T.C.

ALTINBAŞ UNIVERSITY

Electrical and Computer Engineering

OPTIMIZATION AND CONTROL OF DC MOTOR USING FUZZY LOGIC

NOORULDEN BASIL MOHAMADWASEL

Master Thesis

Supervisor

Assoc.Prof. Dr. Oguz Bayat

Istanbul (2019)

OPTIMIZATION AND CONTROL OF DC MOTOR USING FUZZY LOGIC

by

Noorulden Basil Mohamadwasel Alhamadani

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.

Assoc. Prof. Dr. Oğuz Bayat

Supervisor

Examining Committee Members

Assoc. Prof. Dr. Oguz Bayat	School of Engineering and Natural Sciences, Altinbas University	_____
Asst. Prof. Dr. Muhammad Ilyas	School of Engineering and Natural Sciences, Altinbas University	_____
Asst. Prof. Dr. Dilek Göksel Duru	Biomedical Engineering, Istanbul Arel 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: __/__/__

Assoc. 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.

Noorulden Basil Mohamadwasel

DEDICATION

I would like to dedicate this work to my first teacher, my mother, my first supporter and role model, my father and my companion throughout the journey. Without you, this dream would never come true and to my brother and my sister who stood with me in order to achieve my dream.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all the instructors that have taught me more than just science, especially my supervisor Assoc.Prof.Dr.Oguz Bayat, 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

OPTIMIZATION AND CONTROL OF DC MOTOR USING FUZZY LOGIC

ALHAMADANI, NOORULDEN BASIL MOHAMADWASEL

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

Supervisor:Assoc.Prof. Dr.Oguz Bayat

Date: Mar/2019

Pages:81

In,our system design, we used Particle Swarm Optimization (PSO) Algorithm because we want to design a DC Motor system with Fuzzy logic control to provide high angular speed (ω) and low error. The DC Motor system is configured by MATLAB SIMULINK platform R2012a Particle Swarm Optimization (PSO) Algorithm has been utilized to improve from The performance of the Designed System.In this thesis,three tests are taken to demonstrate between different proposals to studying the-DC Motor system based the Proportional Derivative (PD) controller with various formats by use Proportional Derivative (PD) controller with Fuzzy logic control,In the first experimental work, the Proportional derivative (PD) controller used in DC Motor system by setting the values of scale factors,150 of Kp is utilized and the Kd is 0.01,the

overshoot(MP), steady state error (ess),rising time and settling time.In the second proposed experiment., The DC MOTOR System is used based on the fuzzy logic control. In the last experiment, The DC MOTOR System is used based on the fuzzy logic control by using Particle Swarm Optimization (PSO) algorithm.In this work, all experimental results outcome from our simulation present respectively: the maximum Overshoot for Proportional derivative(PD) controller with DC Motor system is (32.54%) but when we used Proportional derivative(PD) controller with DC Motor system by using fuzzy logic control is (4.07%) but when we use Proportional derivative(PD) controller with DC Motor system by use fuzzy logic control and,Particle Swarm Optimization(PSO) Algorithm there is (No-Overshoot) by using PSO. The steady state error for PD with DC Motor is (0.06), The steady state error for PD with Fuzzy based on DC Motor is (0.05), The steady state error for PD with Fuzzy and Particle Swarm Optimization(PSO) Algorithm based on DC Motor is (0.04).They are good results due to choosing an attractive solution methods and these methods will improve DC Motor System and also improve the efficient of PD controller.

Keywords: Proportional derivative controller , Fuzzy Logic Control ,Particle Swarm Optimization,DC Motor System ,MATLAB.

TABLE OF CONTENTS

Pages

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

LIST OF ABBREVIATIONS

PD : Proportional Derivative controller.

FLC : Fuzzy Logic Control.

FIS : Fuzzy Inference System.

PID : Proportional Integral Derivative controller.

PSO :Particle Swarm Optimization Algorithm.

SSO :Sperm Swarm Optimization Algorithm.

GSO :Glowworm Swarm Optimization Algorithm.

CSO :Cat Swarm Optimization Algorithm.

APSO:Adaptive Particle Swarm Optimization.

GCPSO: Guaranteed Convergence Particle Swarm Optimization.

TVACPSO:Time-Varying Acceleration Coefficients Particle Swarm Optimization.

HPSO:Hybrid Particle Swarm Optimization.

1. INTRODUCTION

Fuzzy logic is a form of logic used in some expert systems and applications of artificial intelligence. This logic originated in 1965 by the Azerbaijani scientist Lutfi Zada of the University of California[1], where he developed it to be used as a better method of data processing. The logic of ambiguity was used to regulate a steam engine, and its applications evolved until it reached the manufacturing of a fuzzy logic chip that was used in many products such as cameras[2].

There are many motives that led scientists to develop the science of fuzzy logic With the development of computer and software arose the desire to invent or programming systems that can deal with inaccurate information like human[3] but this was born a problem since the computer can deal only with accurate and specific data. This trend has resulted in what is known as expert systems or artificial intelligence and fuzzy logic is one of the theories through which such systems can be built[4].

The Fuzzy logic In a broad.sense it is a logical system based on the generalization,of traditional bivalent logic, in order to infer in uncertain circumstances,In narrow sense, it is theories and techniques that use blurry groups that are infinite sets of boundaries. This logic is an easy way to describe and represent human experience, and it provides practical solutions to real problems, solutions.that are cost effective and reasonable, compared to other solutions that offer other technologies[5].

1.1 PROBLEM DEFINITIONS

There are many problems that the researchers still now did not find the practical solutions for it because the input to fuzzy logic when controlled on DC Motor on the angle and speed these physically analog things so the fuzzy does not accept it because fuzzy does not know the speed and also the position it should be converted to value that accepted by fuzzy and this value is limited between zero and one this process called fuzzification more steady state error and little performance in DC motor when using Proportional Derivative (PD) controller and this conventional method to test the DC Motor system there are more method to compare between PD and other advanced types to improve the performance of the system.

1.2 CONTRIBUTION OF THESIS

There are more methods it has been used in this thesis to reduce the steady state error and improve the performance of DC Motor system by comparing the conventional method Proportional Derivative (PD) controller with Fuzzy logic and also it has been used optimization method Particle Swarm Optimization (PSO) it will make fuzzy logic work as best as possible and We have three comparison to test DC Motor System between PD and Fuzzy and Fuzzy with Particle Swarm Optimization(PSO) in additional to that Fuzzy with Particle Swarm Optimization it will give the best results compared the previous methods and this will implement on MATLAB code and the designing of DC Motor and also Comparisons has executed in MATLAB SIMULINK R2012a.

1.3 THESIS ORGANIZATION

The thesis organization fall into six chapters, in chapter two we demonstrate the literature review that was introduced by previous researches, in chapter three give an introductory to fuzzy logic control 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 (2015) [6] Discussed the development of a hybrid system, a fuzzy fracture, slip mode control.for a class of interconnected nonlinear.systems.

In (2009) [7] Discussed the DC Intelligent Controller. Use the Particle Optimization (PSO) method to form the best proportional derivative controller (PID). Adjust parameters. The DC controller is designed. environment. Compared with the mysterious logical controller using smart PSO algorithms, the schema. The graph is more efficient in improving the stability of loop response speed, the fixed state error is reduced, the time high. With no overrun.

In (2014) [8] They discussed the Tuning study of the integral integral derivative (PID). Control for speed control of DC motor.Using soft computing techniques. The AC motor is widely used in industries even if the cost of maintenance is higher than the induction engine. Control of DC motor speed is attracted. Research has evolved and many ways. The PID is a commonly used compensatory control unit used in nonlinear systems. This console is widely used. In many different areas such as space, and process. Control, manufacturing, automation, etc. Setting the PID parameter is very difficult. There are many soft computing techniques that are used to adjust the PID controller to control the DC motor speed. Adjusting PID parameters is important. Because,these parameters has a significant coefficient on the stability.and performance of the control system.

In (2005) [9] studied the system of PMSM multipliers using the Particle Sprain Improvement (PSO). Detailed procedures for designing the PID control are summarized optimally. In terms of the principle of improving the particle swarm. In order to improve overall performance. By responding to the system step, a new Hamming Hazy Distance Assessment is introduced to assess performance. Comparisons are made between the results obtained by GA.method and the results of the PSO method. Simulation results and experimental,results show that the PSO.method can determine the optimum or near parameter area and achieve a higher quality

solution than the.GA method. Provides a new and better optimization tool for optimal PID design in complex systems and coupling.

In (2008) [10] The Fuzzy Logic for DC application discussed engine speed control using Opto Optimization (PSO). First, the control unit is designed according to the rules of foggy logic so that the systems are basically strong. Second, the Fuzzy logic controller (FLC) that was previously used with PSO.so has been optimized to obtain optimal settings for organic functions only. Finally, FLC has been fully optimized with Swarm Intelligence algorithms. Demonstrates digital simulation. The results are that compared with FLC, the FLC-PSO's designed controller gets better behavior. Dynamic and superior performance of the DC motor, as well as perfect tracking of speed without overshoot.

In (2010) [11] Evaluate evolutionary computing approaches to determine optimal values for PID parameters. (LFC) and Automatic Voltage Regulator (AVR) system for the single power system using the Particle Sprain Enhancement technique. The LFC loop controls real energy and frequency and the AVR loop controls the interactive power and voltage. Due to high and declining energy demand, the real and interactive power balance is affected; hence, frequency and voltage deviate from nominal value. This requires design. Of the controller is accurate and quick to maintain system parameters in nominal value. The main purpose of controlling system generation is to balance system generation, load and losses. So that the desired frequency and energy exchange are maintained between neighboring systems. This work demonstrates the application of the PSO method to efficiently search for optimal PID control parameters for the LFC and AVR systems. The proposed method has superior features such as: stable convergence. Characteristics, easy implementation and good.computational efficiency. Simulation results. Demonstrates the effectiveness of the designed system, in terms of reducing the time of stability, speeding and oscillations. Results are compared with PID, Fuzzy, and GA.bas conventional controllers.

In (2010) [12] Discussed the improvement of Particle Squad (PSO). On the basis of optimal profit. Adjust proportional proportional control (PI). In the induction (IM) engine (30hp) with the loader loading crane chart. Optimization.Consider load and speed differences, provides appropriate gains for the speed control to obtain good dynamic performance of the engine. The performance of the chat is examined using optimal gains through simulation studies in the

MATLAB / SIMULINK environment. The results are handled by hand (fixed gain) and FL (Fuzzy) speed control. The Hybrid of FL and the PSO-based PI controller are also implemented to control the specified engine speed to eliminate PI (overflow and under-control) defects and the FL controller (static error). From simulation studies, the hybrid controller produces better performance. In terms of uptime, exceeded the time limit and stability time.

In (2012) [13] He studied the development of an optimum.PID controller to control the system of nonlinear beam cranes. The optimized PSO algorithm is implemented based on a priority-based fitness policy to find the best PID parameters. The dynamic model of the system is derived using the Lagrange equation. A set of PID and PD controllers are used to control the oscillation position of the system. System responses. Including displacement from the vehicle and net load variability are monitored and analyzed. The simulation is performed within the Matlab environment to verify the performance of the control unit, where the control unit has proven effective in moving the vehicle. As soon as possible to the desired position. With low net load fluctuation technique.

The optimized PSO algorithm is implemented based on a priority-based fitness policy to find the best PID parameters.

In (2011) [14] The distributed generation (DG), distributed storage (DS), and loads are considered as an effective cleaning system to mitigate climate change. The microgrid.is run in network mode and intermittent mode according to the initial power grid conditions. The role of energy storage system (ESS) is particularly important to maintain the voltage and voltage frequency of the island microgrid network. For this reason, different methods of controlling the ESS have been introduced. In this paper. , Suggests a fuzzy PID controller to improve ESS frequency control performance. This opaque PID consists of a fuzzy logic controller and a conventional PI controller connected in a series. A fuzzy logic controller has input flags. The output signal for the confused logic controller is the input signal for the traditional PI control. To compare control performance, the gains of each PI controller and PID controller are ambiguous by the PSO algorithm. In the simulation study, FID fuzzy control performance was tested under different operating conditions, using PSCAD / EMTDC simulation platform.

3. FUZZY LOGIC CONTROL

3.1 MOTIVATION

Fuzzy logic control is has 3 inputs and rules and 3 output and it deals with fuzzy set it's a math function such as (and gate) take the less significant value and (or gate) take the high significant value and the math of fuzzy is logic its connected with Boolean that related with fuzzy when controlling on DC Motor the control it will be not 2 actions the speed and angle these analog things that deals with physical partition in additional to that Fuzzy will not accept the speed and also angle it should be converted to avalue that fuzzy understand it and these values between zero and one and the process that convert from physical values to the values that fuzzy accept it and deal with it this process called Fuzzification it will compared with Proportional Derivative (PD) controller and also use fuzzy with,particle swarm optimization (PSO).and the use of PSO with Fuzzy it will improved the DC Motor System and find the best solutions for this system to minimize the steady state error of the system response.

3.2 THE CONTROLLER OF FUZZY LOGIC

This is a controlled system so that this system takes a set of entries and conditions to the rules understood by the system and translated into formulas so that this formula is understood by the system dealing with the fuzzy and control idea you are equal to the values inside with the outside values in the sense that you are trying to equal values inside with the outside values without errors or losses during Process conversion values[15].

It is a controller but its main function in software has the advantage of minimizing errors or losses in the system as well as accelerating the responsiveness and performance and gives better performance and high efficiency of the system as best as possible[16].

3.2.1 The General Concept of Fuzzy Logic

A broad.sense is a logical system based on the generalization,of traditional bivalent logic, in order to infer.in uncertain circumstances. In narrow sense, it is theories and techniques that use blurry groups that are infinite sets of boundaries[17]. This logic is an easy way.to describe and

represent human,experience, and it introduce practical solutions to real problems, solutions that are cost effective and reasonable, compared to other solutions that offer other technologies.

Traditional group: In the traditional group, the element is either a member of the group or not a member of the group. In the fuzzy group, the element has membership scores in that group. For example, if U is a set and S is a subset of U and a function gives each element of group U a class of belonging to group S, if element x belongs to group S, if element x does not belong to group S, U=0,1.

The main characteristics of the Fuzzy logiz are the handling of unverified information that is not known from the ground or not, and the use of this information and special paralysis when the assigned parameters do not provide good statistical treatment with satisfactory results. Therefore, the social, Different parameters and variables, whether you (environment, political economy, etc.), this is to deal with the information stamped on the one hand and on the other hand and the variety of a word, and on the other hand to say that The vagueness is the effective tool when taking steps that are characterized by uncertainties resulting from missing and obscured data and information. The general rule is the only one in the heat and inference[18].

The ambiguity is defined as a special type of multicolored variant (VALED). The logical top of the Logic variant is based on the meaning of the vague assemblies. In the ambiguous area, Which are used to express the degree of belonging represented by the use of 1 - 0 within the closed period that allows for the conclusion of the variable 10.. In addition to methods to transform non-output inputs into outputs, Shell relied on the sense of matter, and elements (different data and knowledge) were represented using the opaque group view and identified the element belonging to the opaque group through the functions of belonging. It provides an easy way to obtain conclusions and solutions from the environment that is not achieved (mprecision), and it offers many advantages:

- the ease of understanding as it depends on a satisfied understanding.
- the return to the point of departure; flexibility in terms of the integrity of the amendment and change without it.
- the representation of unverified data.

- the honesty of non-plan types (functions).
- Experience of the experts in a particular field.
- Integrity of the integration of the ambient with the traditional technologies used.
- The integrity of the ambiguity in analyzing and solving issues that depend on linguistic or verbal variables.
- The implication is that a vague language is a philosophical and artificial type of artifact, one of the tools used to take the keys and help to manage and resolve crises, facing various business organizations in circumstances where uncertainties, ambiguities and ambiguities make it possible to integrate the variable variables at the same time Even changing the word, and helping to solve the issues that are difficult to decide in the answers to the solution, the principle is to make the scale and variables take several, or 0 Classic class that makes the top Take either 1 or 0 and be limited to each other by specifying vague groups of ambiguous variables and 1 and 0 are determined to be members of that group[19].

3.2.2 Principle of Fuzzy Logic

Is one of the forms of mystery that puzzled scientists, but it is not necessary now full explanation of the fuzzy logic, but just define and clarify its uses in 1995 Lutfi Zada discovered the fuzzy logic when he was working at the University of California where he noted that the correctness and error is not enough to represent all logical forms, Which we are currently facing. Classical logic is based solely on 0 or 1, and this is what many relations depend on while other relationships exist where the position in which it is considered to be partially true or partially incorrect at the same time[20].

In the fuzzy logic, the transition between the two situations is gradual so that at this stage we can consider the situation taking both cases together as a small change in the value of income causes an increase in the change, not a complete change.

3.3 FUZZY INFERENCE SYSTEM

The system has the ability to convert the physical values to the values that understood by Fuzzy when physical value input to fuzzy inference system it should input to the rules that dealing fuzzy with it in additional to that the fuzzification it will convert the normal values to fuzzy value between zero and one the rules of fuzzy working as the following if the speed is high then reduce it and the steady state error is high and reduce it and so on[21]. And this called control system as a mind that control on values and enter it to rules so the mechanism of rules its receives the values and understand it and change it to a values that fuzzy inference system understand it and then covert it to a values that the DC Motor systems understand it and then make a decision on it and then the output will change it from fuzzy values to a value that DC Motor dealing with it such as the work of compiler the output process called Defuzzification this process is inverse of fuzzification.the system is work as translater to the DC Motor System.they are three process of Fuzzy system that shown below.

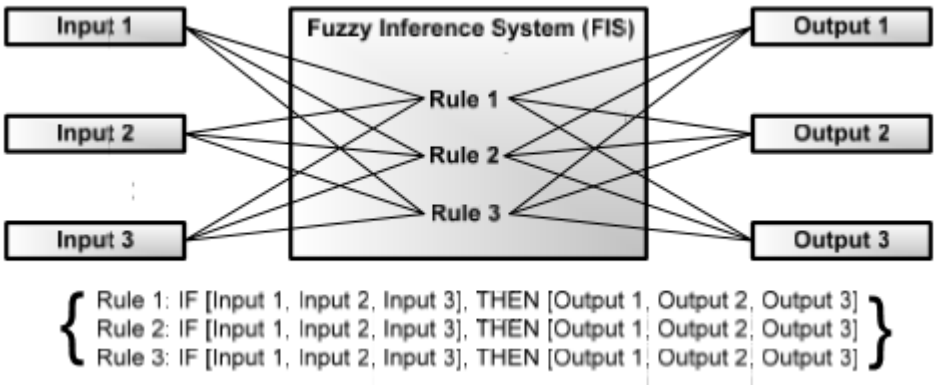


Figure 3.1: Fuzzy Inference System [21].

The fuzzy inference system it's a method that working on control and depending on principle of fuzzy set and fuzzy logic have 3 partitions so it change the variable from normal value ti a value that fuzzy accept it and fuzzy inference system make a rules that take the variables and convert it to fuzzy extension.

There are two types of Fuzzy inference system in the Toolbox of matlab

1-Mamdani

2-Sugeno

3.3.1 Fuzzy Reasoning

Inference and ambiguous reasoning is a speculative model used in the modeling of polarization properties and nonlinear systems, especially when the partial knowledge and the thumb are defined as the variable of the object. In the case of taking the signals that the human is more involved in, it is determined that it is a model based on an understanding of 12 variables (Fuzzy Set Theory), and the one who knows it (from Fuzzy Set Theory) It is the input and the output of the input and output parameters that affect the input and outputs of the input-output system, as well as the bases of the input-output systems.There are two types of assistance (Mamlani), which are the most common 7 in use and suitable for special inputs involving human and human knowledge. - Sugemo Type iPhone Outputs in Shell Enclosed Opaque Warnings on a Shell Set fixed straight[22].

The Discrete Induction and Inference Machine or what we call the specular model is an expert system that illustrates the relationship between input and output of a system through a set of rules and conditions an the goals

The primary objective of using the ambiguous system is to control the complex currencies. Using a sequential analysis strategy based on knowledge derived from experience and experience, the machine has adopted the vagueness of the series, which are often serialized as the following:

- Fuzzification: This is the first step in building the ambiguous model by converting the sharp inputs into opaque inputs through the functions of belonging which take different forms through this step. The inputs expressed in the linguistic terms are converted to the numerical variables in the form of the functions of belonging.

- Appling Fuzzy Operators: When inputs enter into the process of constancy and thumb, we will know the degree of factors to be determined and will collect the existing rules. If the identification has more than one part, the vague logical processes will apply to it to obtain one result.

- Applying Implication Method: At this stage weight is given to each base between input and output, this weight is limited to 0 and 1.

- Aggregating All Outputs: This stage is to gather and collect the collected outputs from all rules to produce the final result.
- Defuzzification: This is the last stage in the construction of the ambiguous model in which the opaque output is converted to outputs with real numerical values[23].

3.3.2 Fuzzy Logic Application

Artificial intelligence:Fuzzy logic is used in the design and analysis of some artificial neural networks.

Operational control: Operational control its mean process control and also relates to automatic control. Most applications include controlling the motor (mechanical) variables of the machine based on input from environmental sensors. Some applications are as follows:

Video cameras: sensor movement of objects that the camera photographed and also any vibration by the camera.

Cars: Provide cruise control where the fuzzy logic circuit calculates the acceleration and control the effect of injecting more fuel or running the brakes.

Air conditioning: Do the heat reduction gradually until you reach the desired level.

Vessels: Control of temperature, pressure and chemical content to maintain stability.

Washing machines: load control, tissue quality and detergent quantity to optimize the cycle cycle[24].

3.4 PROPORTIONAL DERIVATIVE LIKE FUZZY LOGIC CONTROL

It’s controller a type of controlling types that make the process of proportional derivative by using fuzzy logic it has been used in implement the fuzzy and then connect the DC Motor system to each other and the name of PD Controller Fuzzy Logic Control (FLC) in MATLAB is Mamdani and has executed and built in MATLAB Simulink as block diagram and this type has implemented in MATLAB R2012a[25].

3.5 PROPORTIONAL INTEGRAL DERIVATIVE CONTROLLER

Is a comprehensive control loop of reactive feed commonly used in industrial control systems.It is responsible for correcting the error resulting from the difference between the required value and the measured value. The PID algorithm is made up of three separate coefficients: P, Integrity (I), and Differential (D). Proportional value shows the reaction with the current error. The integrative value corresponds to the continuity of the error with time. The differential value corresponds to the rate of error change.

The control loop can be summarized in three basic functions:

Measurement Function: This function is performed by sensors or sensors, sometimes called transmitters. These sensors include heat sensor, pressure sensor, motion sensor, current sensor, resistance probe, frequency sensor, etc. The sender type is selected according to the operation to be controlled.

Comparison and calculation function: The measured value is compared to the sender and the preset value to be accessed. The measured value is called PV while the exact value is SP. The result of the comparison results in a value called error e, which passes to one or the group of previous controls and the result of processing or calculation is sent to the final control section of the work[26].

Final control function: The types of final controls vary depending on the process. For example, valves for opening and locking pipes, motors for speed and direction control, heat control heaters and so on.

On the other hand, it is possible to classify jobs according to the principle of their work. There are three categories:

Pneumatic air function: This type is indicated in the industrial drawings of continuous lines.

Function of working with hydraulic pressure. The symbol for this type is in continuous lines but is cut by letters L – L.

Electric current function: This type is encoded with intermittent lines.

Available technologies is as the following:

PID Pneumatic Controller: PID Pneumatic Controller is one of the earliest technologies to be applied to the application of the differential integral proportional control theory.

Electronic PID Control Technology: It depends mainly on the operational amplifier.

Digital Controller: The latest and most popular in the industry, which adopts advanced programmable control techniques and self-tuning.

PID type control theory: The Types of PID is The elements or boundaries of I and D are completely based on the P boundary, meaning that they do not work without it.

Parallel PID: Functions P, I, and D function independently of each other and then combine into a single output.

Mixed Type PID: A combination of serial and parallel.

3.6 PARTICLE SWARM OPTIMIZATION CONCEPT

Particle swarm optimization is presented previously by Kennedy and Eberhart [27], Particle swarm optimization can be visualized its behavior by Comparing them with swarms of bird by searching For the sources than excisting inside it the best food so the direction of movements the Birds are Affected by their movement of current. The best source for the food that passed in any time and The best source for the food so the birds looking for any quadron for find the food In other words the birds that driving by the inertia their knowledge of personal and the knowledge squadron.in PSO terms,the particle motion is affected by inertia. Its best location of personal and the best location that chosen to be used as international location. PSO has multiple molecules so Each particle consists of its objective value that deals with their current position as the following location, speed, and personal value. This is the very best objective value and the best location for it.its the position that find the best personal value, in additional to that the particle swarm optimization maintainings of it is the best global values and it's the best value has putted for any partition on all and the best global value,and this considered as the place that find in it the best global value.

The classic PSO uses the.following repetition to move particles:

$$x(i)[n+1] = x(i)(n) + v(i)(n+1), n = 0,1,2, \dots \dots \dots, N-1, \quad (3.1)$$

The symbol x(i) that refer to the location for the particle I and n is considered as the frequency so the value for n is 0 that refer to the starting and N is considered as the number for total frequencies [28], and the symbol v(i) is consider as speed that deals with particle or velocity i. in particle swarm optimization classic the particle that deals with velocity is finded by using the following iteration:

$$v(i)[n+1] = v(i)(n) + 2r(i)1(n)[x(i)p(n) - x(i)(n)] + 2r(i)2(n)[xg(n) - x(i)(n)],$$
$$= 0,1,2, \dots \dots \dots, N-1, \quad (3.2)$$

the symbol xp is considered as the best location that taken as the personal location and xg is consider as the best location that taken as the global location and its considered as this symbol x(i)p(n)-x(i)(n) it will calculate the vector that taken as a direct vector in additional to that its towards the greastest location that it has taken as the personal location and xg(n)-x(i)(n) and its considered as the vector that taken as direct vector and its taken as towards location so its considered as the greatest location it has taken for global for both r(i)

1.and,r(i)

2, are the vectors that considered as random vectors,that contains distributed values that arrage as uniform between zero and one. So the encoding r(i)1(n) is to indicate that the new randomly vector is considered as the generator for each one of particle I and a frequency n. particle swarm optimizationcan be affect or focus on either Confluence or variety. In any iterations. To focus on a variety Molecules means that are scattered searching for big area to focus on on convergence its means that the particle are together closed and searching for small very spaces.The committed strategies is to make a focuson replications and confluence on variety in replicas of subsequent.

3.6.1 Global Best Particle Swarm Optimization

The best global particle swarm optimization is an variable that considered as a basic for particle swarm optimization so its utilizes the frequency to find the following velocities:

$$v(i)(n+1) = wv(i)(n) + c1r(i)1(n)[x(i)p(n) - x(i)(n)]$$
$$+ c2r(i)2(n)[xg(n) - x(i)(n)], n = 0,1,2, \dots \dots \dots N-1. \quad (3.3)$$

The weights that considered as inertia its also called fast weights, The weights that consider as best weight c1 [29] and the weights that consider as global weight c2 that compared to the classic PSO it utilizes inertia weights and do not require it. The personal c1 and weights that consider as best global c2 are setted up to 2 in equation (2). The velocity for particle is setted randomly within the area of search.in additional to that the particulate value aim are calculated values and sites of target are generated automatically for best values taken for personal and attitudes for best personal,and value for best global.the location is taken to the value for objective and position of particle with best value for objective.

After one step,all of particles shifting to new locations be use Equations (2) all values for objectives are again appraisal.the best for personal that take molecules to have new value for objective the best location is updating if there is any location best than the value has been found or previously value.the algorithm make continues for objective value.the best values that taken as personal and the best attitudes of personal and best global value and best position for global the algorithm it will stop if the termination standard will stop also such as a boundary to the number of the duplicates.

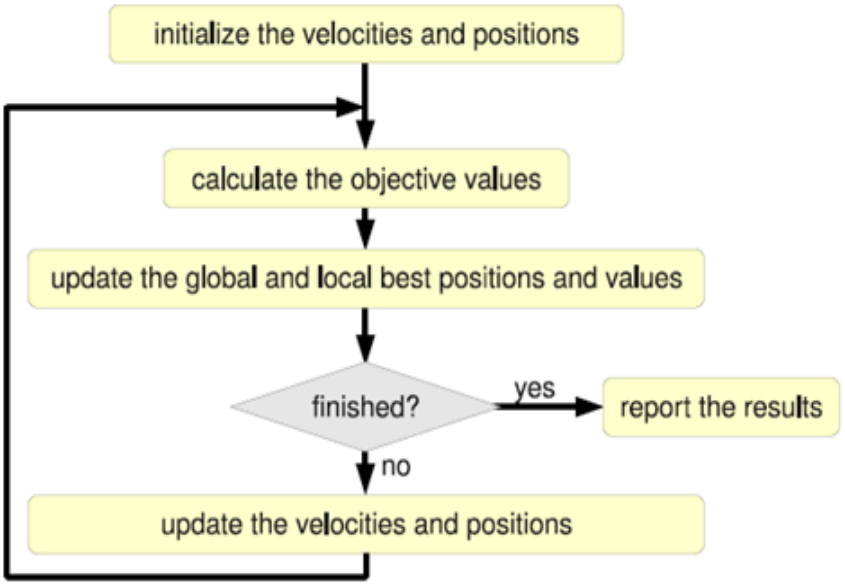


Figure 3.2: FlowChart of PSO [29].

3.6.2 Decreasing Weight Particle Swarm Optimization

The low weight that consider as optimal(DWPSO) particle swarm that comparable to GBPSO but the inertia weight is overtime for linear decline. So the behind of idea the DWPSO is make a focus on the variety in Duplicates that event early and also for confluence in frequencies that event late [30]. DWPSO utilizes the frequency to finde the following velocity:

$$v(i)(n+1) = w(n)v(i)(n) + c1r(i)1(n)[x(i)p(n) - x(i)(n)] + c2r(i)2(n)[xg(n) - x(i)(n)], n = 0,1,2, \dots \dots \dots \dots \dots \dots N-1. \quad (3.4)$$

where the weight inertia,at every iteration n is calculated.using the following equation:

$$w(n) = ws - \frac{(ws - we)n}{N}, \tag{3.5}$$

The symbol w(n) is considered as the weight for inertia at the frequency n, the symbol(ws) is the weights of inertia that putted to the first frequency, so its should to allocated to the last repeat.in additional to that its depending on the number of total times (N),therefore the change of value it will change the behavior of algorithm for each time that repeat.its mean the algorithm not make restarting again from certain point if the sum N duplicates number has been changed.

3.6.3 Time-Varying Acceleration Coefficients PSO

Variable acceleration factors Over time, the PSO (TVACPSO) does not change the weight of inertia only, but also changes acceleration coefficients, namely personal c1 and best global weight c2, over time. The idea is to do a great variety of earlying replicas. And significant Confluence of late recurrences. The weight of the inertia w is changed as in DWPSO.using Equation (5). TVACPSO uses the following frequency to determine the velocities:

$$\begin{aligned} v(i)(n+1) = {} & w(n)v(i)(n) + c1(n)r(i)1(n)[x(i)p(n) - x(i)(n)] \\ & + c2(n)r(i)2(n)[xg(n) - x(i)(n)], n \\ & = 0,1,2, \dots \dots \dots \dots \dots \dots N - 1. \end{aligned} \tag{3.6}$$

Where the character is calculated. The best weight c1 and the best global weight c2 in each frequency n using the following equations:

$$c1(n) = c1s - \frac{(c1s - c1e)n}{N}, \tag{3.7}$$

$$c2(n) = c2s - \frac{(c2s - c2e)n}{N}, \tag{3.8}$$

Where the personal account is calculated. The best weight c1 and the best global weight c2.

3.6.4 Guaranteed Convergence Particle Swarm Optimization

Guaranteed Content Convergence (GCPSO), ensures that the best particle for global, is looking inside The radius is dynamically adapted at all times. This technique addresses the problem of stagnation and increases local convergence by using the best global particle to search randomly in the changing radius of each iteration. GCPSO, as shown in Equation (3), is used to determine velocities v (i) (n) and equation (4) to change the weight of inertia. Best personal

weight c1. The best global weight c2 is kept firmly. GCPSO uses the following redundancy to update the position of the best global particles:

$$v(ig)(n+1) = -x(ig)(n) + xg(n) + w(n)v(ig)(n) + (n)\big(1 - 2r3(n)\big), n = 0,1,2, \dots \dots \dots N-1, \quad (3.9)$$

Where ig is the particle index that has updating the best value of global. The (-x (ig) (n) + xg (n)) expression is utilized to position the ig particles to the best global position. Random numbers in r3 (n) are distributed uniform between zero and one [32]. The search radius is controlled by the parameter of radius search,. The search radius parameter, that calculating using:

$$p(n+1) = \begin{cases} 2p(n), if\ s(n+1) > sc, \\ \frac{1}{2}p(n), if\ a(n+1) > ac, \\ p(n), otherwise, \end{cases}$$

Where sc is the threshold of success and ac is the threshold of .failure. Success means improving the value for best and global location, and means failure. The number of successive successe (s) and failure (a) is calculated using:

$$s(n+1) = \begin{cases} 0, if\ a(n+1) > a(n), \\ s(n)+1, otherwise, \end{cases}$$

$$a(n+1) = \begin{cases} 0, if\ s(n+1) > s(n), \\ a(n)+1, otherwise, \end{cases}$$

3.6.5 Other Particle Swarm Optimization Variants

The local of best PSO GBPSO differention in the best position of universal xg used in the-equation (1) Is replacing with the best location of universal of subset of a certain of particles. Therefore, different Molecules may not be utilized, similar as the best position of universal xg. PSO is informed to fully,To group the molecules in k groups depended on location of their after startation. The size of the neighborhood is increasing with of the persistence algorithm, in an trying to achieving a fully connected like squadron, [33] in additional to that, one group after 80% percent of total number for the replicates N. Local The PSO is studied bested by comparison, the factor of PSO and the PSO factor For the various topology; for example, buildings, living sizes, in the free weight scale PSO Canon do not use any of weights; no weight lack , no best weight of personal, no better global weight. This shows that large neighborhood distribute to Confluence, while small neighborhoods contribute to variety. The boom Factors,

time-varying inertia of weight, mutation, maximum of velocity, and particle restarting. If you install any of particles, it is can not be formatting. If the mutates of the best global particles. In addition to that, a particle is randomly selected and the values are changed in dimensions on random in its location and the velocity is changing. PSO moving attractively between the phase of attractive and phase of repellent depending on diversity in the squadron. The attractive stage focuses on rapprochement, and the stage of abhorrent that Focus on diversity. The attractive phase of global best PSO to calculating the velocities v(i). The phase of obnoxious that uses equation (1) with the best of disgusting and the best universal weight - 2 disgusting to calculate velocities. Two thresholds are used, the minimum threshold of diversity through which the algorithm affect on moving to the phase of dissonance, and the threshold of maximum diversity through which the algorithm moves to the stage of attractive. And collaborative integration, PSO specialized in solving the errors and preventive integers problems. PSO utilizes cooperative and multiple swarms of integrated to progress for different parts of the problem by forward the particles towards the position of best global of their squadron and the locations of best global for neighboring swarms [34].There are PSO variables specialized in solving the problems of dynamic optimization. Multi_Squadron PSO charged and scout self-regulation, PSO was upgraded to solving Problems of dynamic optimization. Multi-charged PSO swarm used sub_ charge, swarms.that repel each of other to maintaining the diversity. PSO scout self-regulation, swarm is divided if optimization of local is found, one part has focuses on optimization of local and the other, on finding the other optimization. PSO Co-op charged is a consists of PSO Co-op and PSO charged. The collaborative PSO that divided the discover, the distance by dimension, and each sub-squadron enhance the values of objective of certain dimensions. The collaborative load-bearing PSO utilizes a context vector that maintaining the values of the best global sub-positions, using vector context to fill the dimensions in a location where its sub-flux is not enhanced. PSO utilizes a concept of synergistic of the charged particles and repel each other as well as neutrals that can not respond. Each other to increasing the diversity and changes of face dynamic in goal function. Vector, PSO evaluated and APSO multigoal, specialized in solve problems of multi-goal. Vector Assessment PSO utilizes one swarm for each target. Each squadron evaluated to values of objective that depended on the target of specific it focused on and utilizes to each other, sweeping the particle of best global as the particle of best global.PSO utilizes the enhanced PSO to find good settings of parameter, for full squadron. Improver

Running PSO runing on multiple PSO of instances and utilizes an optima from the works of individual to improve[35] the settings of parameter.

3.6.6 Hybrid Particle Swarm Optimization

Combining mixed technicals optimization between one or more of techniques,techniques of optimization ,particle swarm optimization that pooling with optimization method its about one or more than one method.PSO that contain linear strategy for search and its utilizes as a global location after each frequency the joint of PSO that utilizes multiple of competing swarms,objective values that calculated by using function of it. The best value of universal of rival squadron. Unlike coovolutionary of PSO,the co-evolution of PSO utilizes flocks of co-operative. PSO utilizes a multi-squadron of co-evolution with one major of squadron that containing the best molecules for universal for all swarmings. Half the particles of all the swarms of slaves. That goes towards her best site of personal, the best location of global, and the best location of global Of the main squadron. The other half utilizes crossover,and mutation. Particles in the Squadron of Master that moves infront of the best position of global and the best global mode for previous replication [36].

3.6.7 Adaptive Particle Swarm Optimization

There is a promisings trend for research within the particle swarm optimization is the utilize of concepts and patterns that adaptive [7, 15,21, 72, 74, 76, 80]. Adaptive PSO variables adapt to their behavior that depended on collected of information as the Improved of revenue.There are many adaptable APSO that stand by changing some or all of speed weights w, c1,C2. PSO utilize adaptation of dynamic as an evolutionary and factor of speed that measuring the better changes for personal. The value and aggregation degree to measure the relative position of particles in the objective space to calculate the weight of inertia. The Adaptive PSO in adapting the weight of each particle inertia depended on its value that taken as objective, the best universal value, and the value of worst universal. APSO changes the intro.in the weight of inertia depnded on a variety of squadron to reduce convergence of premature and thus increasing the total convergence. A variety of the squadron is calculated as previous formula equation (3) [37]. Self-configuration, APSO configuration. The best personal and global weights that are assigned randomly to each of particle and then transmitting to values. For the particle that gave most

updating to the best mode for global, the size of traffic based on the total number of duplicating. In the update. From APSO to self.tuning, the best weights for personal and global are moved at ever steps of smaller to increasing frequencies. In a recent update of APSO self-regulation, the authors explain how self-adjustment of inertia weights with (i), the best personal weights (c).

1, the best in the world

Weights c (i) 2. APSO is controlled, the distance of individual particles is adjusted to the best world level Particles to deal with particle drop problems, rapidly increasing the speed sizes, and catching particles in the local Optima. The parameter is calculated to better adapt the personality

Weights c (i)

1 and the best global weights c (i)

2. APSO inertia gives each molecule its own

It possesses the weight of inertia (w) and controls the weights of inertia as a function of objective values and the best universal objective value. Self-learning for APSO adapts to the best personal weights c (i)

1 of particles, based on reactions to the environment, no feedback provided by the target function. Personal best weights c (i)

1, using the reaction function, that environment is updated.

Adaptive PSO, in control of speed, gave the weights w, c1, and c2 to the squadron depended on speed of average. Adaptive PSO consists of four states in one from the four changes and states The best weight for personal C1 and the best weight of global C2 that depending on the situation of current. The four of The situations utilized are the followings: exploration, exploitation, jumping and convergence by using a set of mysterious on the evolutionary factor so The state is finded for Each iteration. Using a separation of mean of the swarm particles so The evolutionary factor is calculated. Table 3.1 shows changes in the best profile c1 for weight and the best weight for global c2 depended on case [38].

Table 3.1: Four-State APSO:change of c1 and c2 depending on the state.

State	C1	C2
Exploration	Increase	Decrease
Exploitation	Slight Increase	Slight decrease
Convergence	Slight Increase	Slight Increase
Jumping-out	Decrease	Increase

Two states for PSO that avoid Optimas from trap of local and precision lack to find the best global for optimas by placing the molecules in any of exploratory or state of exploitative. In the exploratory, particle, situation, to update. In the state of exploitative, the particles are removed from their better and worse locations of personal. The state of the particles is determined by comparing the distance between its location and the best location for global of the activity threshold that is reduced the over time. Adaptive PSO variants verify the adaptation of entities such as random distributions utilized. Fuzzy Adaptive PSO utilized random gradient for the-speed. To speed up the particles that have a velocity (i) smaller than the minimum speed of the-velocity minimum.This accelerates the decreasing counters for speed that may be held. The algorithm in local optimization. Fuzzy APSO that calculates the factor of size and velocity minimum utilizing a set of vague and Expert rules.Mutation is utilized by many variables of APSO. In the PSO with an adaptive boom, mutation.is utilized to increasing diversity and reduce the convergence chances. The best location for personal is adjusted. For each particle depended on the difference in values of objective and the aggregation of degree Of particles,the relative locations of particles in the area of solution. A mutation is utilized to avoid the early convergence. The best global particles have changed with a certain of probability, which is finded by the variation in values of objective [39]. Low variation that leads to many mutations, while high results for variability in few of mutations. Discovery and response APSO is utilized to solve problems of dynamic optimization by monitoring the best global and second placements. Detection and Response APSO supposes that the change in the function of objective has occurred if either of these two positions has changed its objectivity. If the function of objective changes the particles and the best particles of global are reconfigured in randomly locations, the

Adaptive Dimension of PSO addresses the problems of dynamic optimization. dimension by utilizing the probability equation to calculate the number of positions dimensions. The adaptive vector is utilized to find the dimension that is added or deleted in the case.the number of position dimensions changes Multiple APSO targets address very problems of restrictive. Multi-targets using gradient, search on particles of non-dominant. The technology weight is changed depended on the level of adaptive depended on the distribution of molecules of non-dominant.

3.6.8 The Advantages of PSO

- Insensitive to scaling of design variables.
- Simple implementation.
- Easily parallelized for concurrent processing.
- Derivative free.
- Very few algorithm parameters.
- Very efficient global search algorithm[40].

3.6.9 The Disadvantages of PSO

- Slow convergence in refined search stage (weak local search ability).
- The major drawback of PSO, like in other heuristic optimization techniques, is that it lacks some.what a solid mathematical foundation for analysis to be overcome in the future development of relevant theories. Also, it can have.some limitations for real-time ED and other[41].

3.7 PSO APPLICATIONS

- Training of neural network.
- Optimization of electric power distribution network.
- Structural optimization.
- Process biochemistry.
- System identification in biomechanics.
- Images processing.
- Human tremor analysis for the diagnosis of Parkinson's disease
- Cancer classification
- Examples of human movement biomechanics

- Prediction of survival and survival

3.8 THE ELECTRICAL PART OF DC MOTOR

The idea of dc motor that comes from armature has the resistance and mutual inductance connected with back emf(electro-motive force) that generated inside armature coil[43] and the armature is connected with the load of the system the mechanical load and this connected with electrical Motor so the input voltage to the DC Motor is voltage and the output of DC Motor is angle (θ) or speed (ω) $\theta, \omega m = \frac{d\theta}{dt} = \dot{\theta}$ it's a derivatived the angle according to the time in additional to that the Motor is dealing with this section so the-Kirchhoff Voltage Law (KVL) will taken for the loop circuit from left to right so the input voltage and resistance voltage and coil voltage and also back voltage V.emf inside armature so the KVL it will calculate as this formula $\sum Vi = 0$ the summation of voltages in one loop = 0 so the Equation it will calculate as this formula $Vi - VR - VL - eb = 0$ the armature current inside each element of circuit is series and back emf deal with Motor Torque speed Constant.

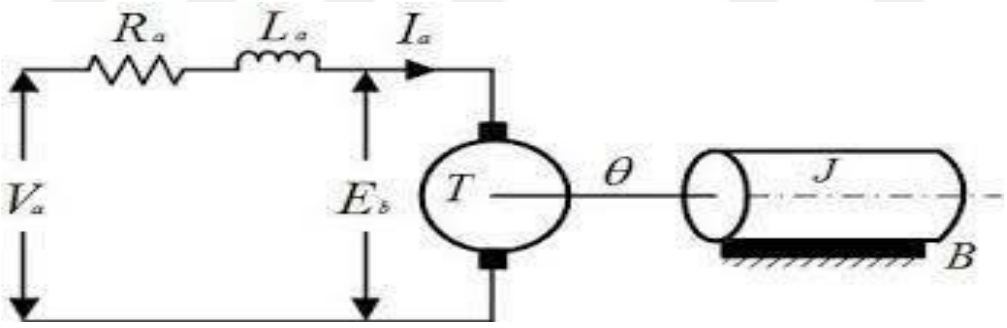


Figure 3.3: DC Motor circuit design [43].

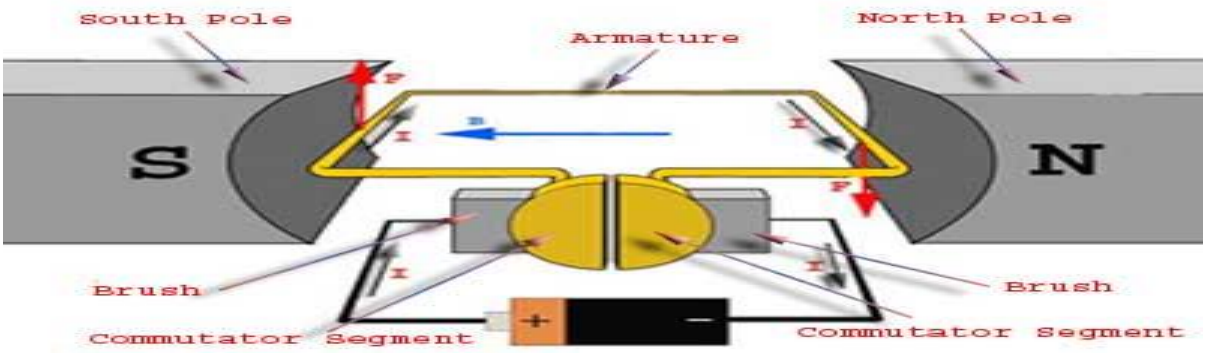


Figure 3.4: DC Motor Electrical Mechanism of work [44].

Back V.emf that dealing with Motor speed ($\dot{\theta}$) and Motor Torque speed Constant is called K so $V.emf = K\omega m$ and ω its $\dot{\theta}$ so its equal to this formula $V.emf = K\dot{\theta} = K\frac{d\theta}{dt}$ and the ωm is the speed of Motor shift[44] and the voltage of coil is the coil factor multiply current derivative of the coil and it's the same armature current because the circuit is series according to the time and $VL = L\frac{dIa}{dt}$ and $VR = Ra\ Ia$ depending on ohm's law and these law's it will be Compensating in Kirchhoff Voltage Law and this will create this formula : $Vi - RaIa - L\frac{dIa}{dt} - k\frac{d\theta}{dt} = 0$ and it should to isolate the electrical part from mechanical part and each part in KVL when theta has inside part of equations its as mechanical part because its dealing with theta so this related with rotation so the new formula it will change because of changing in side of equation and each partition it will transfer to the part that dealing with it and the formula is

$$Vi - k\frac{d\theta}{dt} = Ra\ Ia + La\frac{dLa}{dt} \tag{3.10}$$

so the left part of equation is shown as mechanical part that dealing with back emf and the right part of equation is shown as electrical part and the equation should to convert to laplace transform to be implement and execute in MATLAB SIMULINK by taking laplace transform for equation (1) [45].

Table 3.2: conversion of Electrical element to Laplace Transform.

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

The resistance in above table that used as impedance and Z_L its impedance of the coil that converted to LS and Z_C its impedance of the capacity that converted to $\frac{1}{CS}$ this is the laws will be converted to laplace transform to be executed in Matlab Platform and to be convert to laplace

transform you should to understanding these derivation and also integral fundamentals[46] and its as the following:

$\frac{d}{dt} = s\ so\ any\ derivation\ for\ a\ time\ it\ should\ to\ convert\ to\ S\ (laplace\ formula)$

$\int dt = \frac{1}{s}$ any integral for a time it convert to $\frac{1}{s}$

And these formulas will replaced with equation (1) element so it will create a new formula and it as the following:

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

Every element in the equation refer to derivative it will convert to (S) and the armature current will taken as common factor to be rotate the coil should to take it and this will create a new equation as the following:

$$Vi(S) - KS\theta(S) = Ia(s)[Ra\ + LaS] \tag{3.11}$$

And this equation has electrical current Ia(s) to be used in DC Motor System by divide both sides in [Ra+Las].[47] it will produce this equation as this formula:

$$Ia(s) = \frac{Vi(S) - KS\ \theta(S)}{Ra + LaS} \tag{3.12}$$

And the motor current it has been calculated. In DC Motor the electrical torque that dealing with electrical current it should to convert to be dealing with electrical current and to calculate the Torque and from it we know the speed of DC Motor because the electrical torque it's the same mechanical torque that's why I convert the formula[48].

3.8.1 The Mechanical Part of DC Motor

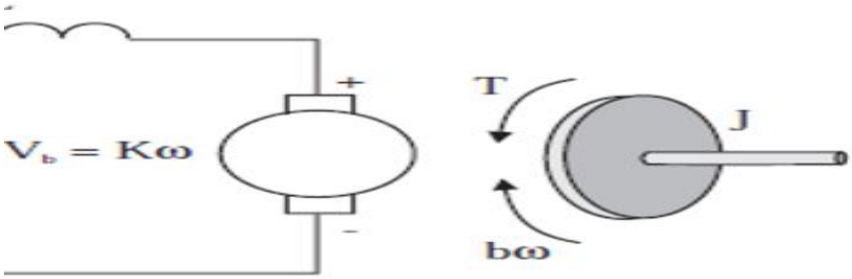


Figure 3.5:DC Motor Mechanical Torque [49].

In mechanical section any part has ability to rotate its called inertia torque (J) and he has the part that rotate surrounding the motor called Friction coefficient (B).

Firstly the electrical torque Te its Torque constant multiply armature current its create a new formula:

$Te = Kt\ Ia$

The electrical Torque dealing with Armature Current and Kt Torque Constant

The main identificate that depended on derivation for dynamic model of DC Motor is Electrical Torque equal Mechanical Torque[49].

Secondly the mechanical torque Tm its inertia torque multiply in double derivation theta over time and also add Friction coefficient multiply by displacement its shown below in formula:

$$Tm = J\ \frac{d2\ \theta}{dt2} + B\ \frac{d\theta}{dt} \qquad (3.13)$$

So if we take the efficiency=100% now all of electrical torque that converted to Mechanical torque this prove that Tm = Te and this concludes the new equation as a new formula:

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

J= polar moment of inertia.

B= damping constant.

K= mechanical torque constant.

TL= load Torque

J,B,K coefficients its changing from motor to another depending on the motor size and current….and so on and the equation that mentioned above dealing with J,B,K we will change it to block diagram to be used in MATLAB SIMULINK and the figure below describe the DC Motor Model[50].

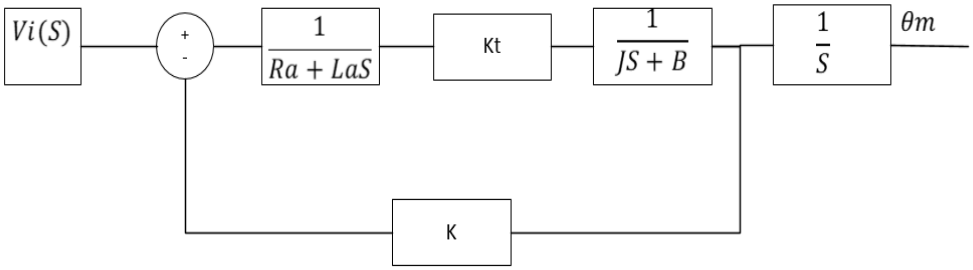


Figure 3.6: theoretical Block diagram design (laplace transform) for DC Motor system.

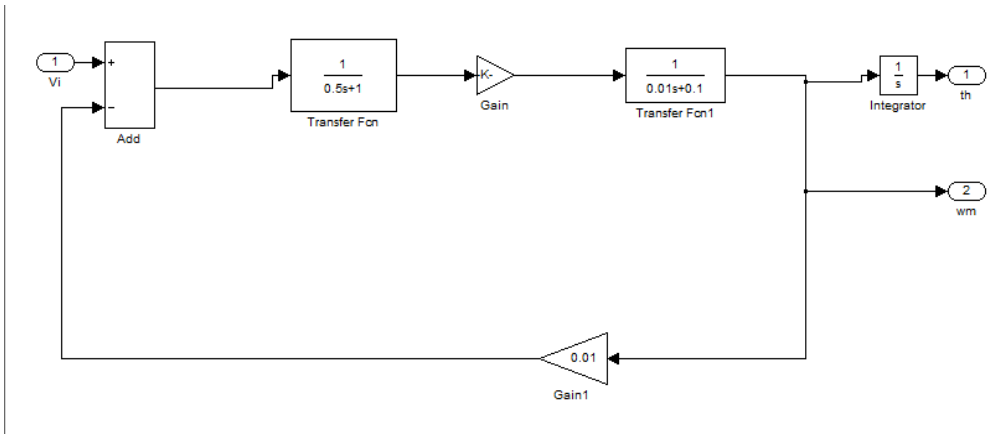


Figure 3.7: DC Motor design in MATLAB SIMULINK.

Firstly we will generate a current so we need voltageVi(S) that input to summation and the feedback will include at last KS $\theta(S)$ so we get this equation Vi(S)-KS $\theta(S)$ it will multiplied by

$\frac{1}{Ra+LaS}$ it will multiply by Kt because the current when multiplied by k it will changed to Torque and take laplace transform for equation (4) so we will get this formula:

$$Tm(S) = JS\ \omega m(S) + B\omega m(S)$$

And the mechanical torque equation its second derivation or first derivation for speed so every $\frac{d\theta}{dt} = \omega m$ we can write it either in $\frac{d\theta}{dt}$ or ωm and $\omega m(S)$ it will taken as common coefficient so should make control on speed and angle it will improve DC Motor System[51].

$$Tm(S) = \omega m(S)\ [JS + B]$$

From torque we will get the speed so the torque will multiply by $\frac{1}{JS+B}$ so in MATLAB SIMULINK the torque will multipled by previous above block $\frac{1}{JS+B}$ we get the speed and the speed when take integral for it we get angle.

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

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

$$\omega m(S) = S\theta(S)$$

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

The values of the parameters that used to modeling the DC Motor System in the table that mentioned 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 we should to input these value in MATLAB m file firstly to implement these values in desktop of matlab to know the matlab what we want to do.

So the MATLAB SIMULINK can to take the values of DC Motor System from MATLAB Desktop so its shown as a symbol and also in MATLAB SIMULINK when want to call it by Symbols because its also dealing with these symbols so we need as the following to implement DC Motor in MATLAB SIMULINK:

1-unit step: to compare the results that shown to us with unit step.

2-adder: addition for add the DC Motor Parameters and make operation on it.

3-subtraction: to reply the feedback with Gain that comes from the DC Motor Parameters and integrator to generate the speed.

4- When we completed the DC Motor System Design in MATLAB Simulink so the DC Motor System became subsystem and the outside design describe input voltage and ωm. And the subsystem of DC Motor system that shown below:

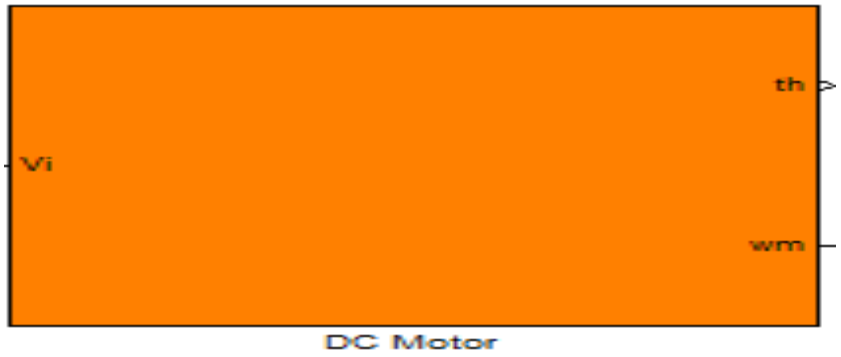


Figure 3.8: The Outside design of DC Motor system in MATLAB SIMULINK.

3.9 PROPORTIONAL DERIVATIVE WITH DC MOTOR

First:build Proportional derivative Controller to connect it with DC Motor System.the block diagram that shown below describe the contents of PD Controller.

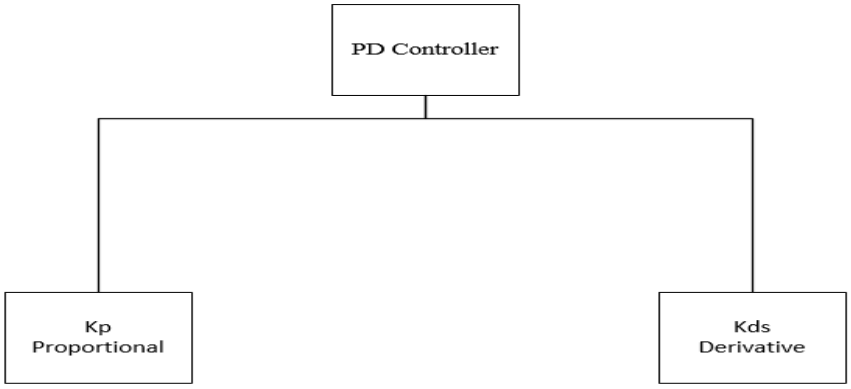


Figure 3.9: Ingredients of Proportional derivative (PD) controller.

The block diagram above describe proportional derivative it has two partitions (kp)proportional and (Kds)derivative so these elements make PD Controller and the control signal that mentioned below as equation.

$$u(t) = Kpe + Kd\frac{de}{dt} \tag{3.14}$$

By take laplace transform for this equation to input it to MATLAB SIMULINK it will introduce to us this formula:

$$U(S) = Kp\,E(S) + Kds\,E(S) \tag{3.15}$$

By take common coefficient

$$U(S) = E(S)[Kp + Kds] \tag{3.16}$$

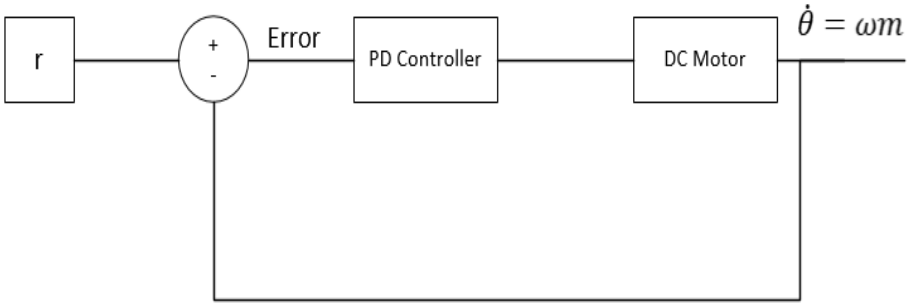


Figure 3.10 : Mechanism of connect Proportional derivative (PD) controller with DC Motor system.

The above block diagram describe the connection between PD Controller and DC Motor System to provide the speed and also to check the steady state error for this system. The block diagram below talking in details the contents of PD Controller when connected with DC Motor System and distribute of PD Controller Kp and Kds and all of these blocks will be input to MATLAB SIMULINK as this format to be accepted it.

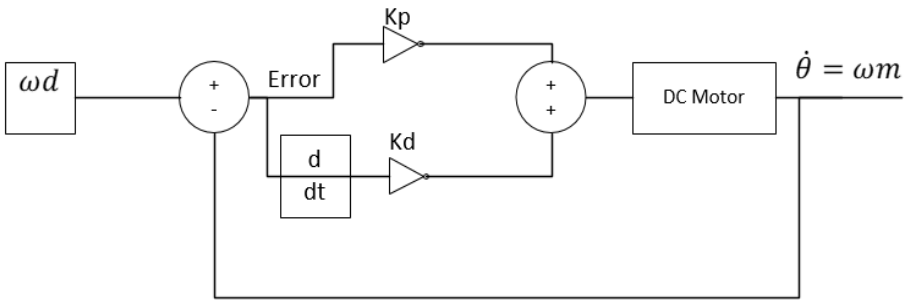


Figure 3.11 : The theoretical inside design for PD controller connected with DC Motor.

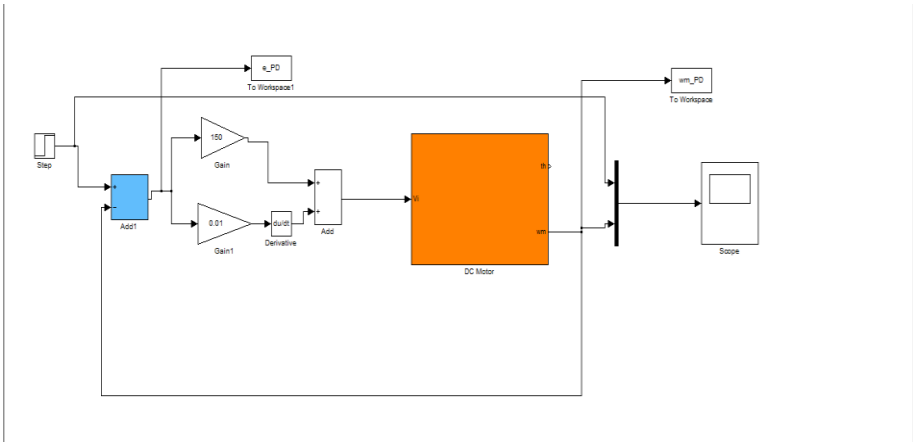


Figure 3.12 : The Outside design for PD controller connected with DC Motor system by MATLAB SIMULINK.

Kp and Kd these values represent the gains these gains that affect from value to another and then it will connect to DC Motor system to check whats the changes will be on the speed of DC Motor and the gain has setted to p and d coefficients its as the following:

Kp=150

Kd=0.01

3.9.1 Fuzzy Logic With DC Motor

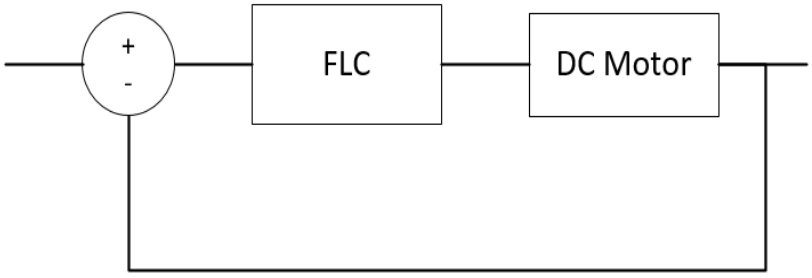


Figure 3.13 : The theoretical design for Fuzzy logic control with DC Motor system.

The fuzzy type that used in the system it's a popular fuzzy:mamdani that Existing inside MATLAB Toolbox.

The system above describe the connection between Fuzzy logic control and DC Motor so the desired signal input to fuzzy and connect with DC Motor.the type of fuzzy that Existing inside fuzzy inference system (FIS) its called PD it take two inputs the error (e) and the derivation of error ($\dot{e}$) as input and product as output is actual signal after normalize it so its product this formula:

$e = d - a \qquad d = desired \quad a = actual$

$\dot{e} = \frac{de}{dt} \; when \, \dot{e} = change \; of \; error$

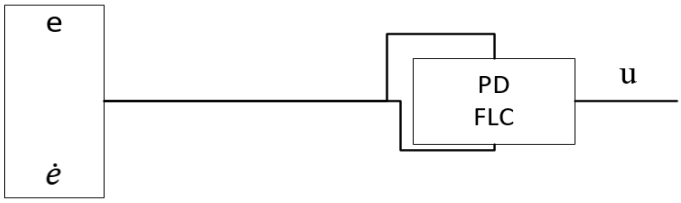


Figure 3.14: the theoretical connection between error and its derivative function with FLC.

The fuzzy controller or Fuzzy inference system its represent the mind of fuzzy that depending on the rules that contains an instructions that govern it.the rule it has been used in this thesis (5*5) its called rule matrix and this depended on membership for the function that used in fuzzy logic and the type of membership that used is triangular so the 5 elements for error and 5 other to the change of error and 5 to the output of fuzzy logic so we get 25 input value and output 5 values its called member ship function for each variable.

Firstly the design for fuzzy logic model and membership of error signal e(t) and change of error d(e) and output is mentioned below.

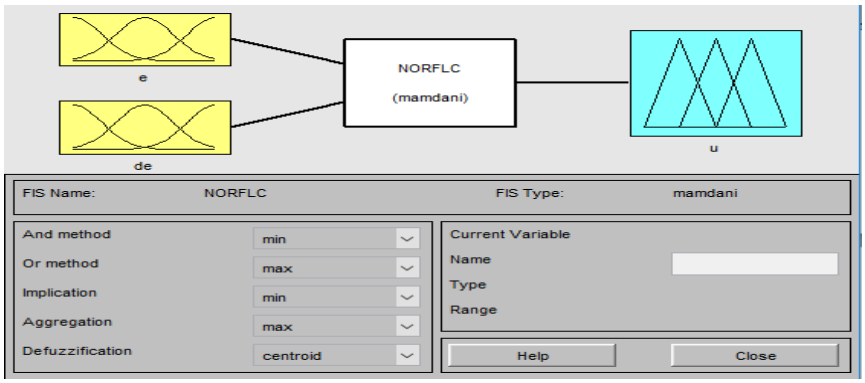
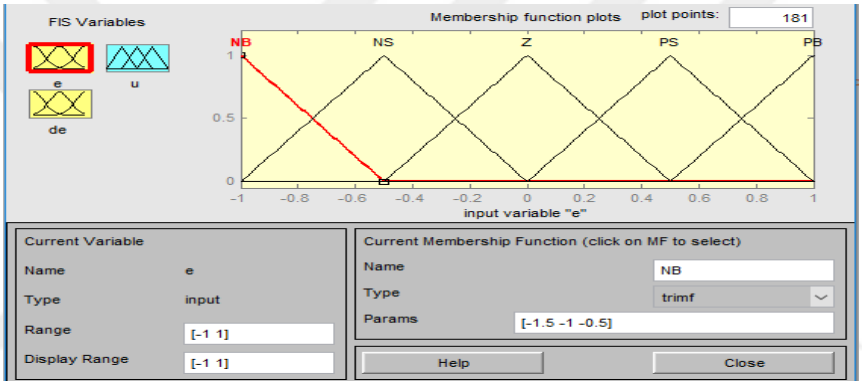


Figure 3.15: Fuzzy logic control function (mamdani) designed in MATLAB SIMULINK.



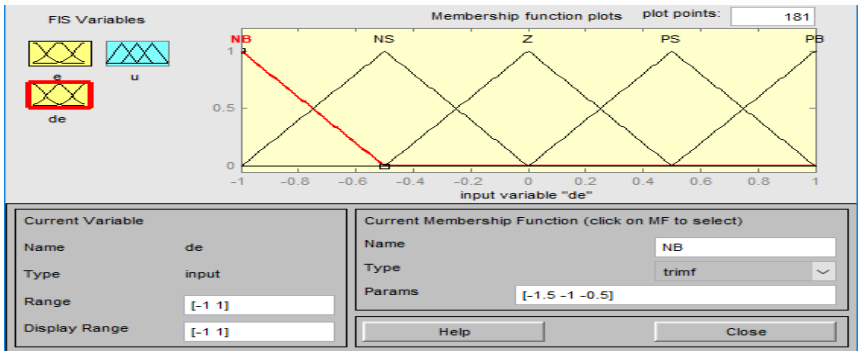


Figure 3.16:the input(error) and its function(derivative of error) designed in FLC function(mamdani).

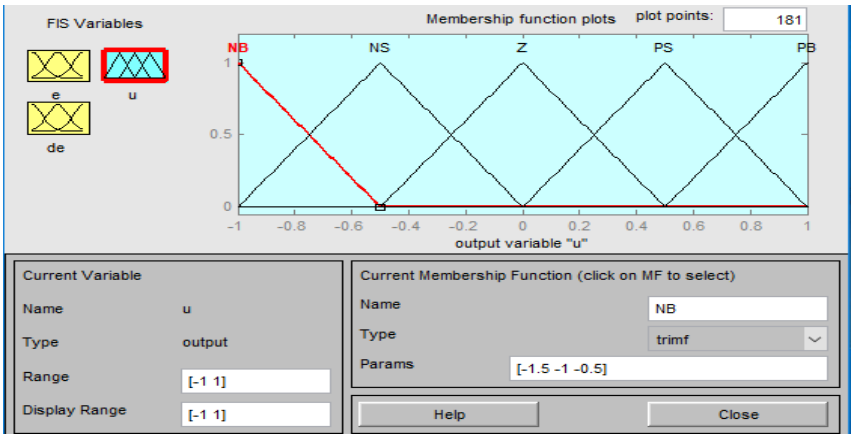


Figure 3.17: The output of fuzzy logic control (u) designed in FLC function.

The second input for fuzzy is d(e) the change of error or rate of error the normalize to make standard for data and control on gains and choose the range that I need it and can to increase the gain this depending on the normalization.

The output for fuzzy it taken the same mechanism of work such as in previous cases in error signal and the change of error.

The rules of fuzzy logic control is called fuzzy inference rules its govern the input signals and normalize it to make it as a rules and the table(5*5) that describe reducing the error down to zero or if it high it will reduce it or if it down its increase it up to zero and so on…

The table contains fuzzy rules. So the input has 5 input and also the derivation of error and the probability theory it will implement on this table that describe down as a fuzzy table.

Table 3.4: Fuzzy rules probabilities for FLC Function.

U	NB	NS	Z	PS	PB
NB	NB	NB	NS	NS	Z
NS	NB	NS	NS	Z	PS
Z	NS	NS	Z	PS	PS
PS	NS	Z	PS	PS	PB
PB	Z	PS	PS	PB	PB

The first case to make normalize for the signal so limit the values between -1 ,1 because the error possible its value in -1 or +1 so the zero its position in middle and the distribution method that used is symmetric function and distribute the signals by shifting it from -1 to +1 and make normalize for it to limit the values and not go out of the range of control signal.

So the middle value is called (Z) and the side that approaches from zero called PS or NS and the P or N this depending of the number is it a small or big and the side that far away from zero called NB or PB and the same meaning that mentioned above to this case.the gain factor in the fuzzy block diagram it will difference between error signal and change of error and also the output and between -1 to zero in the middle= -0.5 and also in +1 to 0 in the middle= +0.5 so to make the normalization working well in error signal divide the signal into 4 partitions.

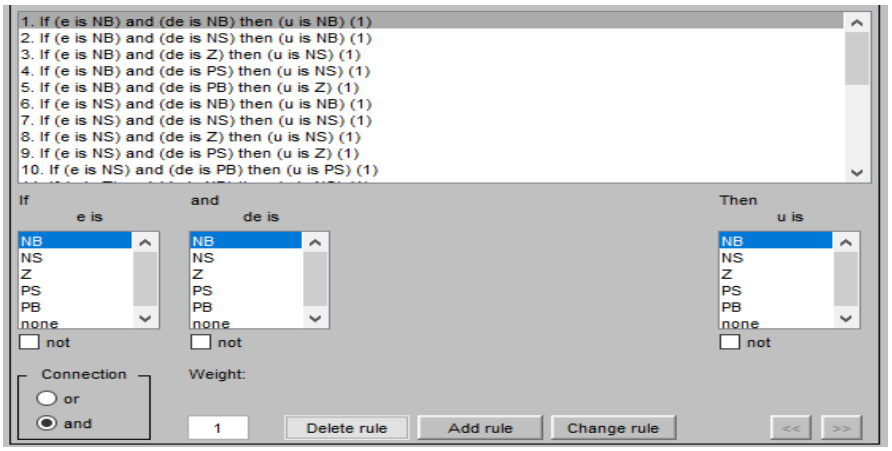


Figure 3.18: the designed rules lists in fuzzy logic control.

After input the rules in tables to rule editor when we want to check from the rules that we have input it by click on View>Rules and then will show us the figure that mentioned below:

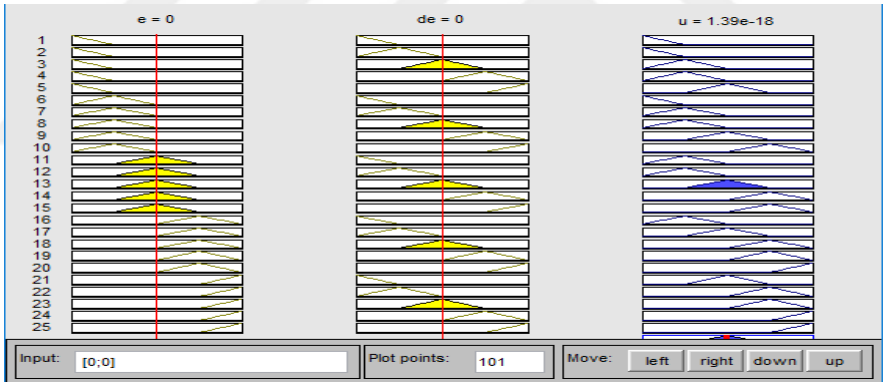


Figure 3.19: the outside of rules lists in fuzzy logic control.

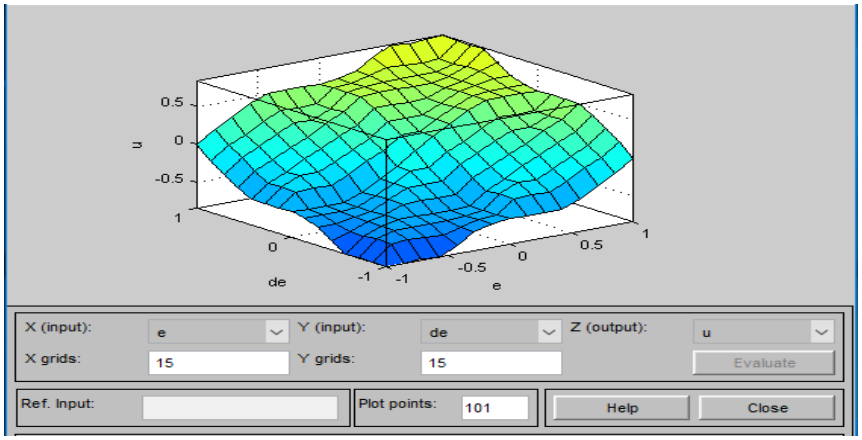


Figure 3.20: the outside Surface for inputs and output values for fuzzy logic control.

The figure above describe the surface that collect the error signal and also the change of error and output of the signal.

When we completed the output of the signal (u) and then export it to the workspace of the MATLAB and then when can call it be command window and then write a word in workspace (‘your function’) so inside this word it will be your function you have design it in MATLAB and normalize it by rules you just want to have put your function name in the command window such as the example that mentioned above and below there are more example to this cases.

in surface the x-grid and y-grid they has been setted when do the fuzzy logic during the normalization so the parameters that shown above in surface as two inputs and one output as the following X,Y,Z.

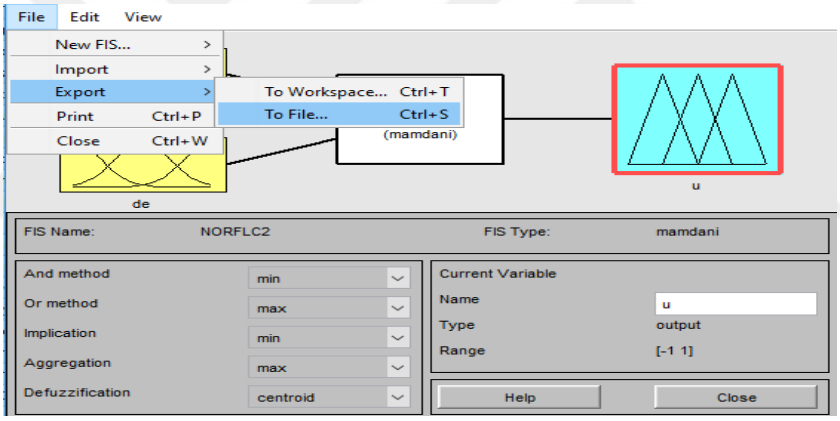


Figure 3.21: Save and export the designed mamdani function to Workspace.

When complete and input all rules in fuzzy inference system and get the output.so this stage to call the fuzzy system from command window you should to export it as a file and named it and then put the name in command window as the same name that you save the function.

The scaling factors that setted for K1,K2,K3 =[5,0.1,50] these Gains that improve the response of the system and its also reduce the steady state error for DC Motor system the figure below describe The Model of DC Motor system that connected with Fuzzy Logic Control(FLC).

The saturation that used after two gains to normalize the values between -1 and +1 and do not change the values or goes out of the range these also help the fuzzy logic in input operation and also in rules because these values its so important to show the best results than PD Controller

when used with out fuzzy.the benefit of using fuzzy logic to arrange the input signals that comes from the equipement that deal with it and change it to a language that by it can understand anything any this language called Fuzzy Inference Rules and Matrix of it is (5*5) as two input values the error and its derivation and one output only will appear from the Math calculation rules so.the fuzzy will check the system when used PD with fuzzy and check what the fuzzy that improve it during Math Operation Rules.

And the figure that shown below describe the usage of Fuzzy controller with PD and DC Motor system.

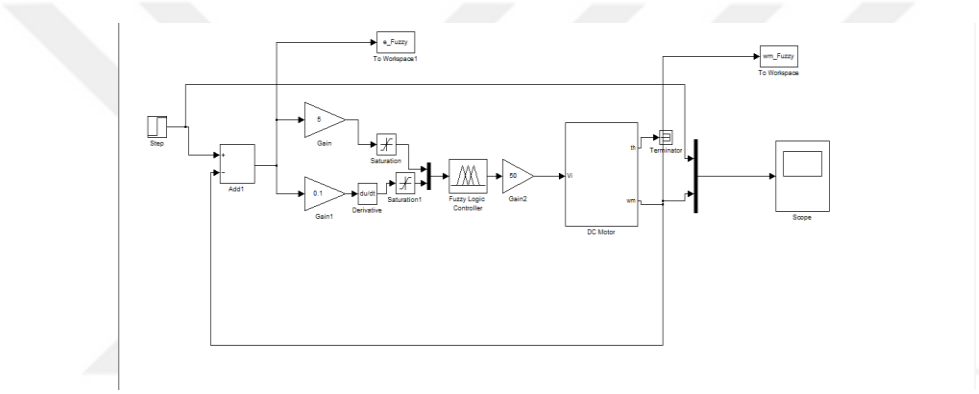


Figure 3.22: The outside design for fuzzy logic control connected with DC Motor system.

The proportional derivative controller has been connected using fuzzy logic control to be improved the DC Motor System and also reduce the steady state error and the maximum no. of overshoot and make the system stable more than in PD controller and the figure above describe the fuzzy logic that connected before DC Motor systemand also improve the main coefficient the angular speed and also the error it will be compared with other controllers to see whats these controllers that improved in DC Motor system.the scope used to see the final results for the system and the others that connected by adder the first is ωm to check the angular speed what the thing that improved for this system but the error that connected in gains to check how many error has been occurred during applied the fuzzy logic control and what normalization has been improve.

3.9.2 Fuzzy Logic Using Particle Swarm Optimization With DC Motor

In this partition make the scale factor (K1,K2,K3) by Tuning using Particle Swarm Optimization(PSO) by taking the same DC Motor parameters that used in Proportional derivative controller and Proportional derivative controller using Fuzzy Logic Control to check what PSO that improved.

So the tuning will give us the best values for (K1,K2,K3) the task of PSO algorithm in these values will setted dynamically this will give us the best results than other controller this process called Optimization in which give the perfect values for the gains this will give the best response and give it to gains of fuzzy logic control.

The working of Particle Swarm Optimization(PSO) it has object function that work on the system it create minimization for the steady state error or the function derivated from steady state error.

The principle of Particle Swarm Optimization(PSO) working its optimization method this method find the best solution for scale factor of fuzzy logic control.

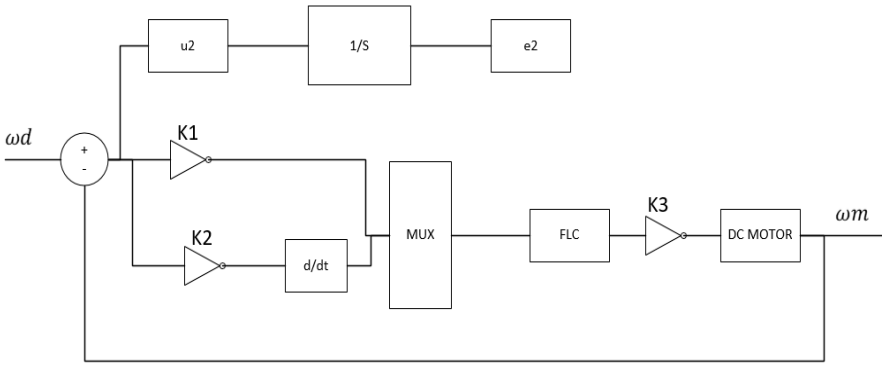


Figure 3.23: the theoretical design for PD,FLC,PSO that connected with DC Motor system.

The figure above describe the connection of Scale Factors(K1,K2,K3) with Fuzzy in DC Motor system using Particle Swarm Optimization(PSO). and ωd its desired speed that give it to the scale factors and ωm it's the real speed.the particle swarm optimization its object function that depended on the steady state error or the derivation of error so its take the error and take square

for it and input it to a function that named u2 and then integrate it and then save it as name e2 in the workspace and it should be modify on it in MATLAB SIMULINK by set it as Array its important to be useful in the workspace.

The idea of topic is make tuning for the scale factor that excisting inside Fuzzy Logic Control.

The Particle Swarm Optimization (PSO) in every execute operation that give the new values for Fuzzy Logic Control (FLC) and the number of birds is :50 and iteration is 10 so the process is 500 times that make process on scale factors for best new values in every round.

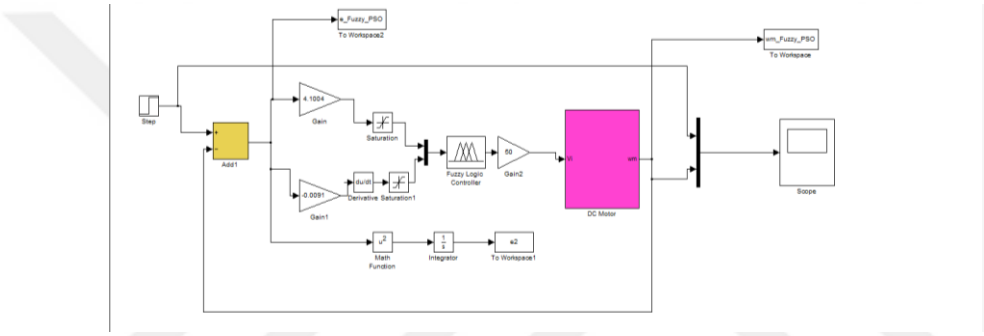


Figure 3.24: The design of PD(proportional derivative) with Fuzzy logic control support PSO Algorithm connected with DC Motor System.

The figure above describe the scale parameters that has been setted in (K1,K2,K3) that taken from Particle Swarm Optimization during the tuning process when was the number of birds:50 and iteration number:10 so it will looking for the best 3 gains and give it to the MATLAB and when make sure that the results is the best and then these values that PSO has give it to us and after that take these values has been putted Dynamically from PSO Algorithm and insert it stactically to K1,K2,K3 in this step,the PSO Algorithm Its role ends because that benefits of using it is to find the best gains and then we can put these gains in a normal case as well as the values that has been setted Previously in Fuzzy Logic Control.the values of (K1,K2,K3) has been putted as the best gains that received from PSO Algorithm its as the Following:

K1 = 4.1004

K2 = -0.0091

K3 = 50

The mechanism of Particle Swarm Optimization (PSO) Algorithm work ,firstly that parameters has been setted and after that to can call it,the current_position that has been putted in algorithm:17.5,the current position is the positions of birds currently and it make it as local best position.

in evaluate initial population: its calculate the generations so the number of generations from 1:n so this will call the function of tuning that has been maked in MATLAB and the result of it will put it in current fitness,and the tuning the number of birds rotation and the mechanism of tuning it will call MATLAB SIMULINK and calculate the error inside this function this process that make it and generate the gains and also choose the best suitable gains and the current fitness that make it local best fitness and take the lower value in it and then will take it its mean the lower error and make loop to the function is it any one that give lower than global will take it and then will update the speed depending on the values and also update the position,the first step that will take from zero up to bird step about 50 and will call the function as the number of birds and then put these values from tuning to current fitness and then comparing if the current fitness less than local after replace the local by current fitness position values to save the best gains that comes from PSO Algorithm and then take minimum of minimum and make it as current global best fitness,and then if there is the lower values in current global best fitness will take it and if the lower values is in global best fitness then will take it instead of current global best fitness values and modify in the velocity on the best case and this will be on the global and also local and take the best among them and then save the best of global and show it in the gains values.

4. SIMULATION MODEL

In this thesis, many of the processes are implemented on the DC Motor system by applying the proportional derivative controller and compared it with in and with out fuzzy logic control and also used the optimization method its the Particle Swarm Optimization has been tested with fuzzy logic control and Proportional Derivative Controller. The MATLAB platform is utilized to achieve and design control system by MATLAB SIMULINK libraries and has the possibility and facility to each of the following: (design.simulating.testing.verifying).

Two experiments have been applied that suggested to competition the proportional derivative Controller (PD the conventional controller) that existing in MATLAB SIMULINK,at the end point there are more drawback in (PD) controller such as high Overshoot,High Error that might be affect on the efficiency of DC Motor instead of using the (PD) controller alone,The proposed system to suggest another method such as Fuzzy Logic Control using Particle Swarm Optimization (PSO) that will be improve the angular speed and reduce the error percent.

4.1 PD CONTROLLER WITH DC MOTOR

This controller consists of two scale factors as (Kp its refer to the proportional and Kd,its refer to the derivative).

The main scale factors that used in this system is (Kp= 150) , (Kd= 0.01) for the System of DC Motor.by utilizing the Previous Scale Factors and these factors has been discussed in the past and checked for the DC Motor system. These Factor has been putted to check the efficient of PD controller and the following figures describe the angular speed and the error for Proportional Derivative when applied on DC Motor System and its as the following:

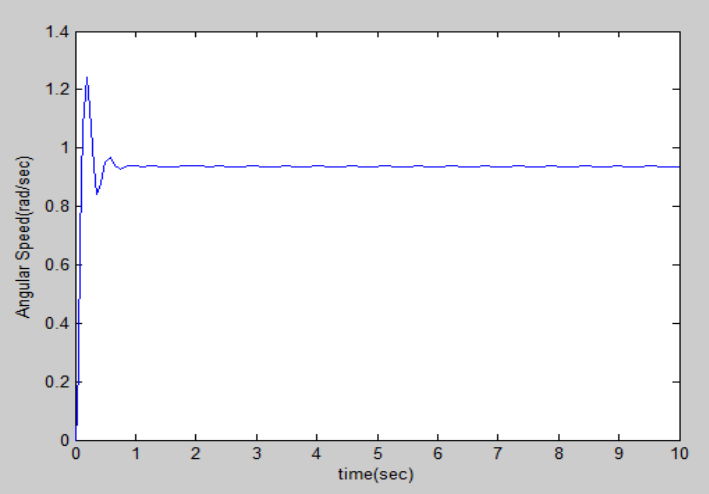


Figure 4.1: The angular speed for proportional derivative controller with DC Motor system.

In figure above describe the angular speed and what affect that excisting in this curve so as we see there is a high overshoot that start from zero and pass the 1 rad/sec its represent the y-axis so the overshoot that event at 1.243 and this is a high percent will reduce the efficient if will not find a best solution for it in additional to that the maximum overshoot will be calculate depending on Math Equation by this equation we can know the percent of overshoot what will be,so when make Math calculation for this cases multiply it by 100% the different between Yp it's the overshoot value in this figure and Yss ,so we getted a high maximum overshoot its about 32.54% for PD Controller when connected with DC Motor System and steady state error has been found about 0.0622 and according to rising time(Tr) = 0.112 sec and settling time (Ts) = 2.5 sec,these results has been extracted from the above figure by make a mathematically calculation on it as we see in the figure in 1(sec) the angular speed is stabled and also the figure below for the error stable at (1)sec.

in the figure below describe the error curve for Proportional Derivative Controller(PD) that connected and tested on DC Motor System and the figure as the following:

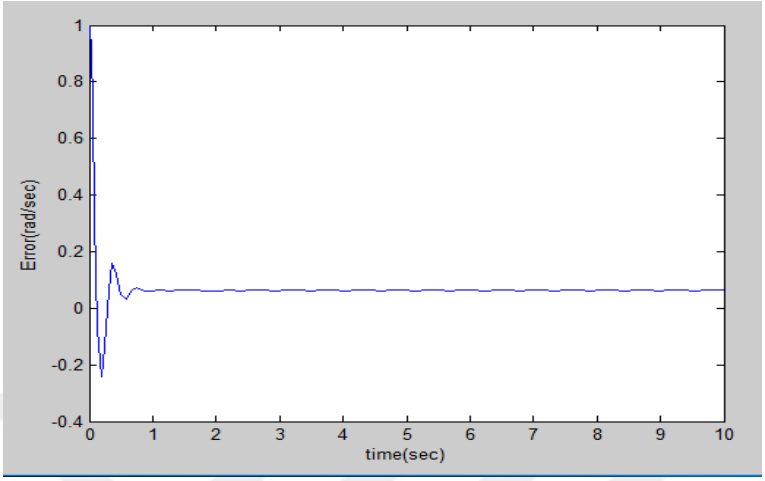


Figure 4.2: The error for proportional derivative controller with DC Motor system.

4.2 PD-FUZZY LOGIC CONTROL WITH DC MOTOR

The general scale factors that used in this system is (K1= 5) , (K2= 0.1), (K3= 50) for the System of DC Motor.by utilizing the Previous Scale Factors and these factors has been discussed in the past and checked for the DC Motor system. These Factor has been putted to check the efficient of PD controller with fuzzy logic control and the following figures describe the angular speed and the error for Proportional Derivative with fuzzy logic control when applied on DC Motor System and its as the following:

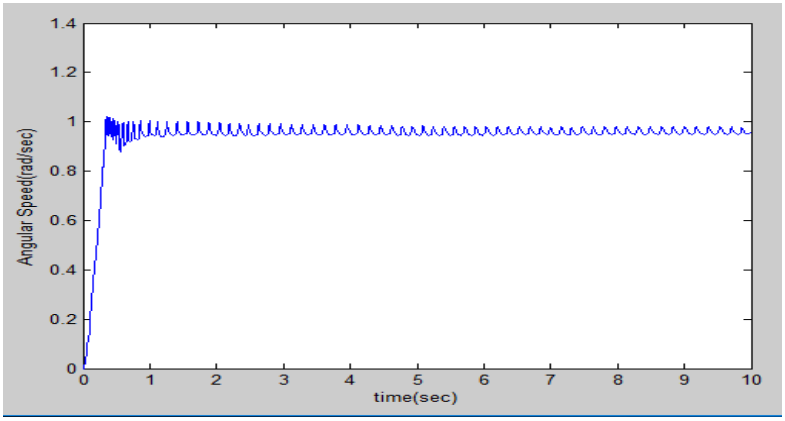


Figure 4.3: The angular speed for proportional derivative controller with DC Motor system using Fuzzy logic control.

In figure above describe the angular speed and what affect that excisting in this curve so as we see there is a low overshoot that start from zero up to the 1 rad/sec its represent the y-axis so the overshoot that event at 1.135 and this is a low percent than the result of angular speed in Proportional Derivative controller (PD) but in additional to that the general curve is not stable this will reduce the efficient but its still need to improve in additional to that the maximum overshoot will be calculate depending on Math Equation by this equation we can know the percent of overshoot what will be,so when make Math calculation for this cases multiply it by 100% the different between Yp it's the overshoot value in this figure and Yss,so we getted a low maximum overshoot its about 13.2 % for PD Controller with fuzzy logic control when connected with DC Motor System and steady state error has been found about 0.05 and according to rising time(Tr) = 0.325 sec and settling time (Ts) = 0.72 sec,these results has been extracted from the above figure by make a mathematically calculation on it as we see in the figure in 1(sec) the angular speed is stabled and also the figure below for the error stable at (1)sec.

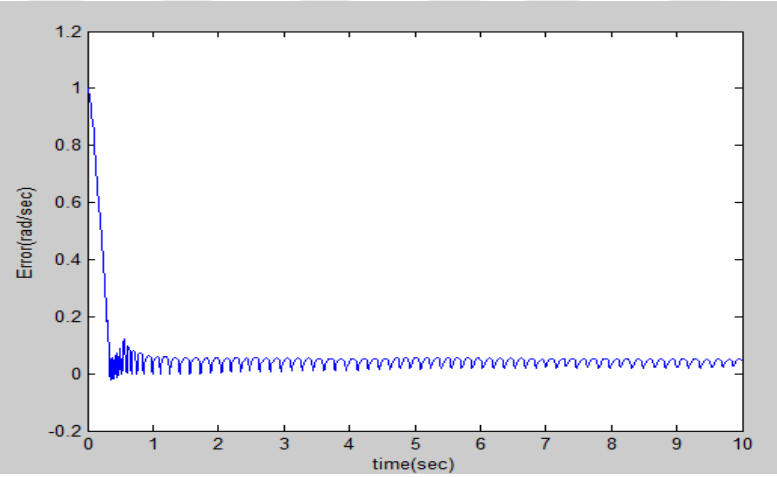


Figure 4.4: The error for proportional derivative controller with DC Motor system using Fuzzy logic control.

4.3 PD-FUZZY LOGIC CONTROL USING PSO ALGORITHM WITH DC MOTOR

The main scale factors that used in this system is (K1= 4.1004) , (K2= -0.0091), (K3= 50) for the System of DC Motor.by utilizing the Previous Scale Factors and these factors has been discussed in the past and checked for the DC Motor system and also the algorithm of particle swarm optimization has been generate these factors dynamically and make the DC Motor system working in the best case. These Factors has been putted to check the efficient of PD controller with fuzzy logic control using particle swarm optimization and the following figures describe the angular speed and the error for Proportional Derivative with fuzzy logic control and particle swarm optimization when applied on DC Motor System and its as the following:

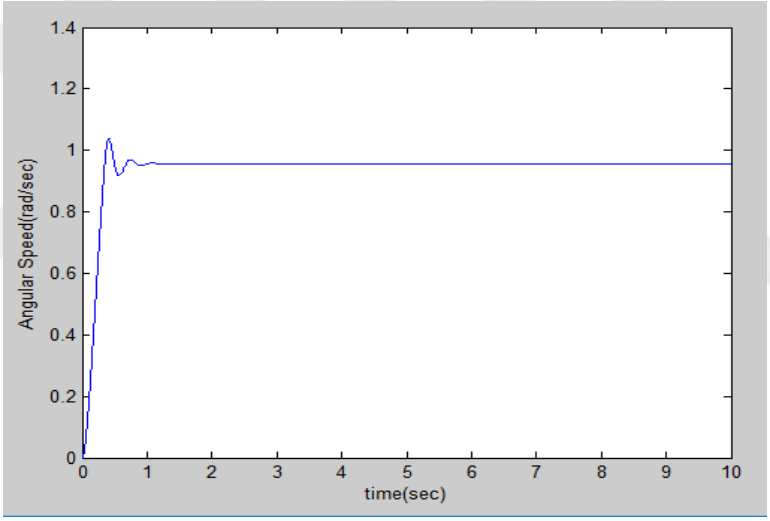


Figure 4.5: The angular speed for proportional derivative controller with DC Motor system using Fuzzy logic control with PSO Algorithm.

In figure above describe the angular speed and what affect that excisting in this curve so as we see there is no overshoot when used particle swarm optimization (PSO) and this is a good percent than the both result of angular speed in Proportional Derivative controller (PD) and also Proportional Derivative controller (PD) when used with fuzzy logic control, in additional to that the general curve is a very stable this will improve the efficient ,and steady state error has been found about 0.0435 and according to rising time(Tr) = 0.326 sec and settling time (Ts) = 1.05 sec,these results has been extracted from the above figure by make a mathematically calculation

on it as we see in the figure in 1(sec) the angular speed is stabled and also the figure below for the error stable at (1)sec.

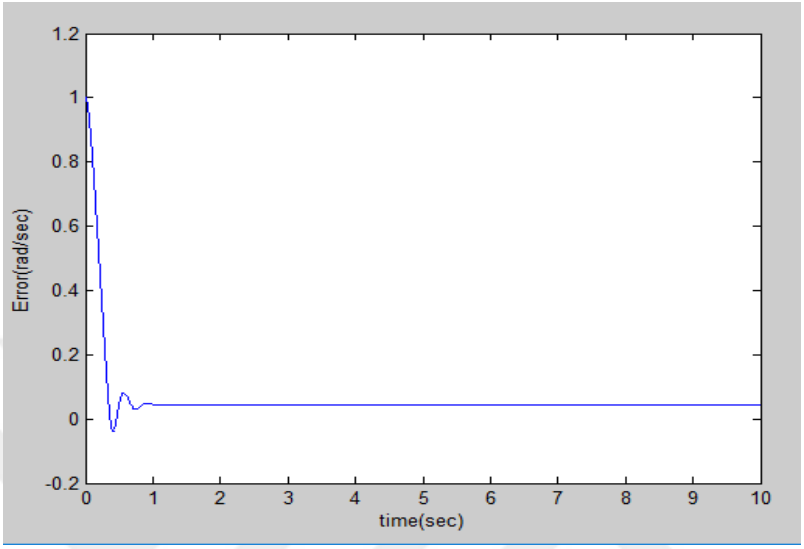


Figure 4.6: The error for proportional derivative controller with DC Motor system using Fuzzy logic control with PSO Algorithm.

5. ANALYSIS OF SIMULATION RESULTS AND DISCUSSION

The use of Proportional derivative (PD) controller with fuzzy logic by applied particle swarm optimization (PSO) ,its attractive solution for DC Motor system its leads to improve the performance of DC Motor system,in additional to that improve the response by FLC in using particle swarm optimization (PSO).The MATLAB Platform is used for these benefits design,simulate and also testing for the our models of us.

5.1 PD CONTROLLER WITH DC MOTOR

The results are obtained that depending on the MATLAB Platform has been used and applied in DC Motor System and the results are check and verified for Proportional derivative (PD) controller to check what this controller that improved for DC Motor System,the figures below that describe the overshoot,rising time,settling time and also the steady state error such as the-following:

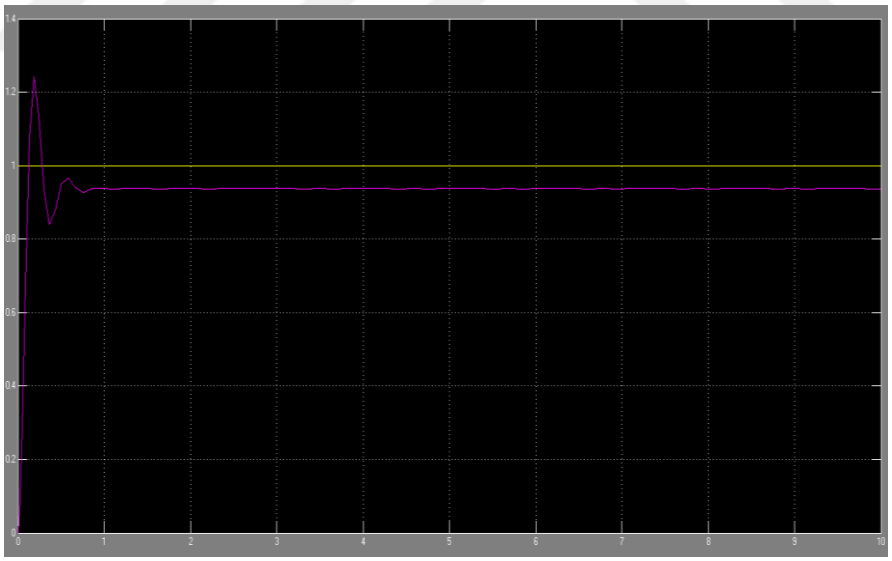


Figure 5.1: The general curve for proportional derivative controller with DC Motor system with unit step by scope in MATLAB.

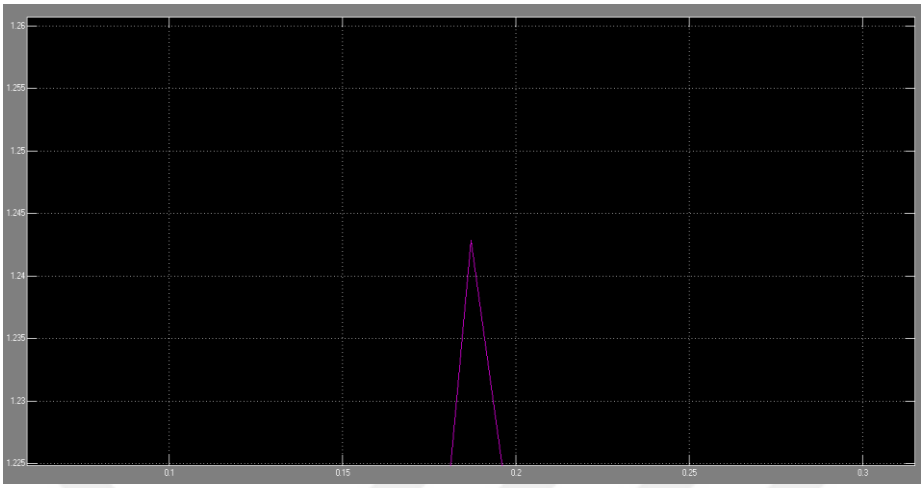


Figure 5.2: The overshoot curve for proportional derivative controller with DC Motor system by scope in MATLAB.

The figure above describe the Overshoot of proportional derivative (PD) controller its stopped in 1.243 to make the mathematics calculation for this overshoot and its considered as the different between the maximum overshoot and steady state(in any time that the curve has been stabled).

The math calculation it will take by the following law:

$$MP = \frac{yp - yss}{yss} * 100\%$$

$$MP = \frac{1.243 - 0.9378}{0.9378} * 100\%$$

MP= 32.54 % this value that refer to the maximum overshoot for PD controller that applied on DC Motor system.

The figure below describe yss value that taked as a stable value because there are no changes has been found,so this value its include inside the maximum overshoot law as the calculation that mentioned above.

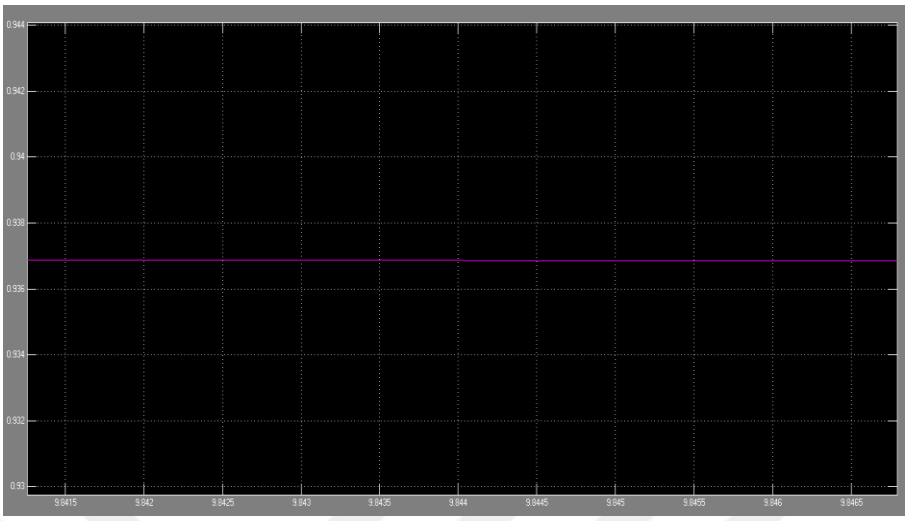


Figure 5.3: The steady state error curve for proportional derivative controller with DC Motor system by scope in MATLAB.

$$steady\ state\ error\ (ess) = 1 - yss = 1 - 0.9378 = 0.0622$$

Above the Math calculation of steady state error for Proportional derivative (PD) controller.

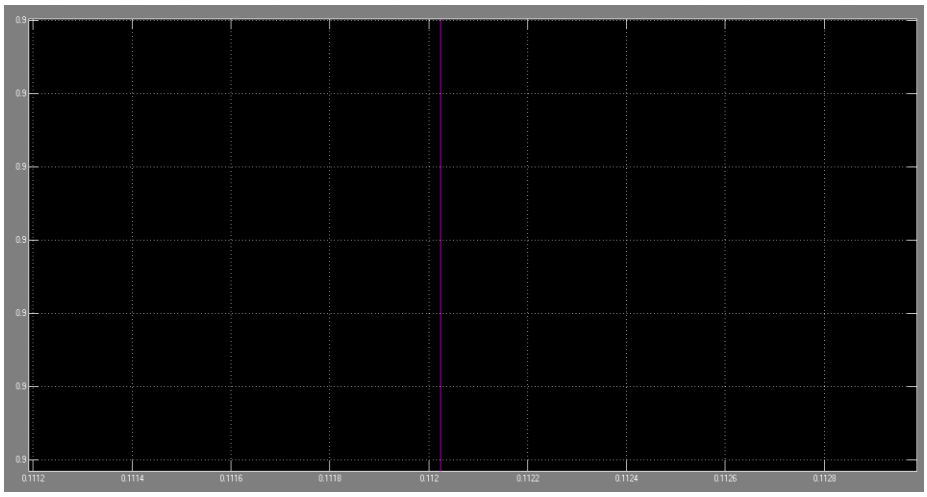


Figure 5.4: The rising time curve for proportional derivative controller with DC Motor system by scope in MATLAB.

The above figure describe the time that the curve of Proportional derivative (PD) controller has been rising and this considered as 90% from the upper value in the beginning of the curve and its has settled in 0.112 as shown in the above figure.

In the figure below describe the settling time that the curve still repeat the same value as shown in the figure that mentioned below its start from 2.5 (sec) and its considered as the time of the response when settled.

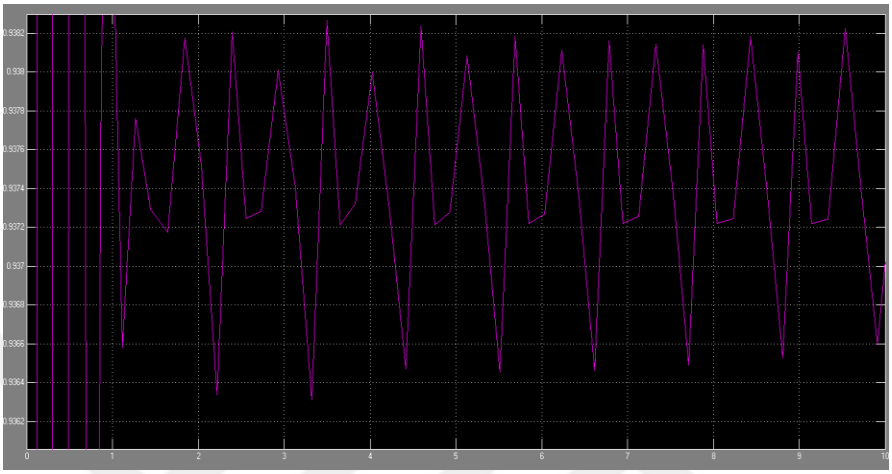


Figure 5.5: The settling time curve for proportional derivative controller with DC Motor system by scope in MATLAB.

5.2 PD-FUZZY LOGIC CONTROL WITH DC MOTOR

The results are obtained that depending on the MATLAB Platform has been used and applied in DC Motor System and the results are checked and verified for Proportional derivative (PD) controller with Fuzzy Logic Control (FLC) to check what this controller that improved for DC Motor System by using Fuzzy Logic Control (FLC),the figures below that describe the overshoot,rising time,settling time and also the steady state error such as the following:

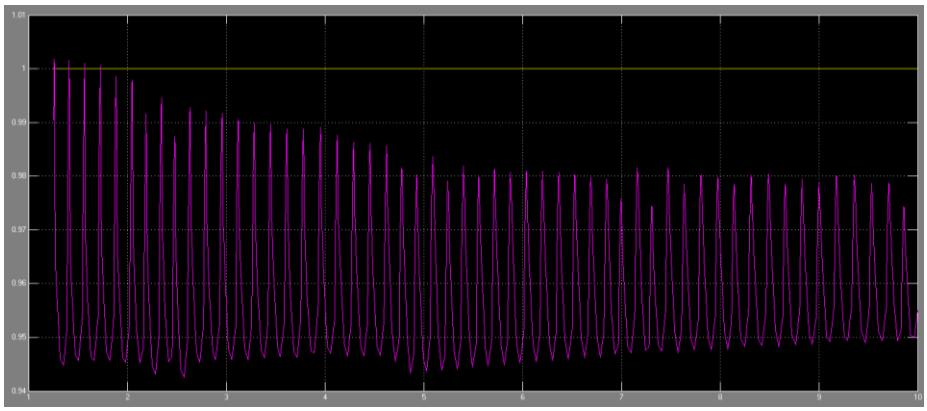


Figure 5.6: The general curve for proportional derivative controller by using fuzzy logic control with DC Motor system with unit step by scope in MATLAB.

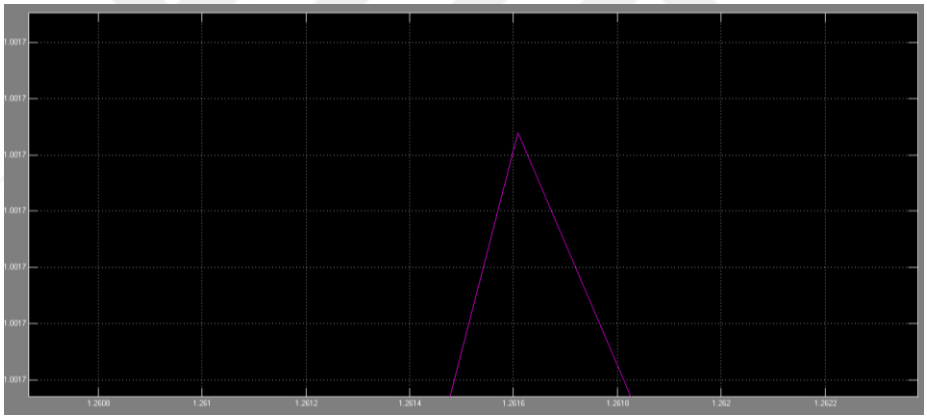


Figure 5.7: The overshoot for proportional derivative controller by using fuzzy logic control with DC Motor system by scope in MATLAB.

The figure above describe the maximum overshoot for Proportional derivative (PD) controller with Fuzzy Logic Control (FLC),the math calculation that mentioned below for overshoot.

$Yp = 1.0017$

$$yss = \frac{0.98 + 0.945}{0.945} = 0.9625$$

$$MP = \frac{yp - yss}{yss} * 100\% = \frac{1.0017 - 0.9625}{0.9625} * 100\% = 4.07272727\%$$

The calculation of overshoot that mentioned above that prove this percent is the best than in proportional derivative (PD) controller.

$ess = 1 - yss = 0.05$

The figure below describe the steady state yss the upper and lower that shown in the figure.

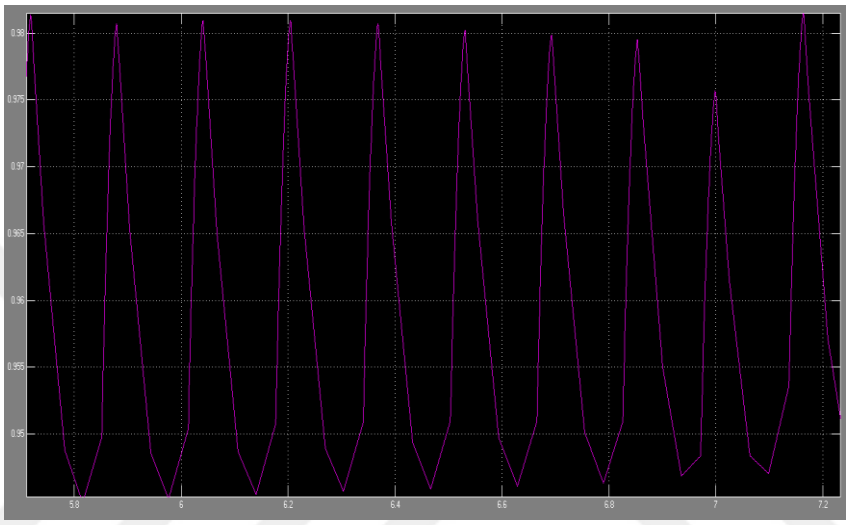


Figure 5.8: The steady state error for proportional derivative controller by using fuzzy logic control with DC Motor system by scope in MATLAB.

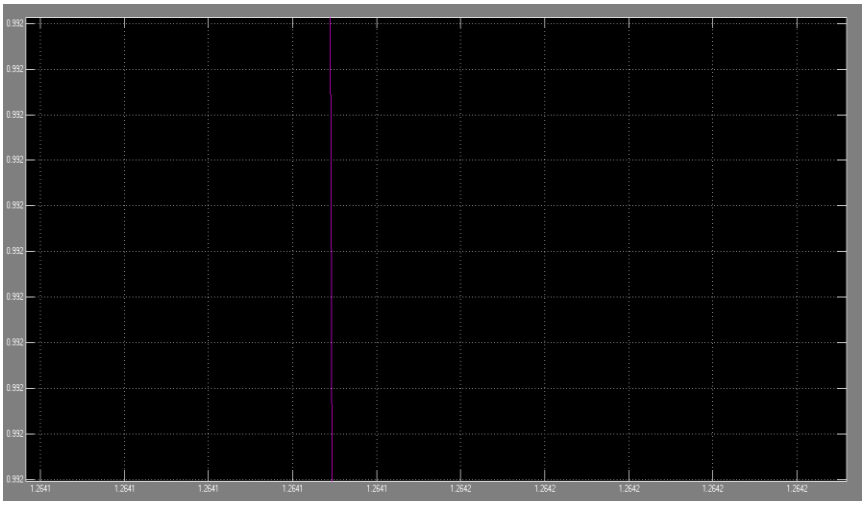


Figure 5.9: The rising time for proportional derivative controller by using fuzzy logic control with DC Motor system by scope in MATLAB.

The above figure describe the time that the curve of Proportional derivative (PD) controller with Fuzzy logic control has been rising and this considered as 90% from the upper value in the beginning of the curve and its has settled in 1.2641 as shown in the above figure.

In the figure below describe the settling time that the curve still repeat the same value as shown in the figure that mentioned below its start from 0.72 (sec) and its considered as the time of the response when settled.

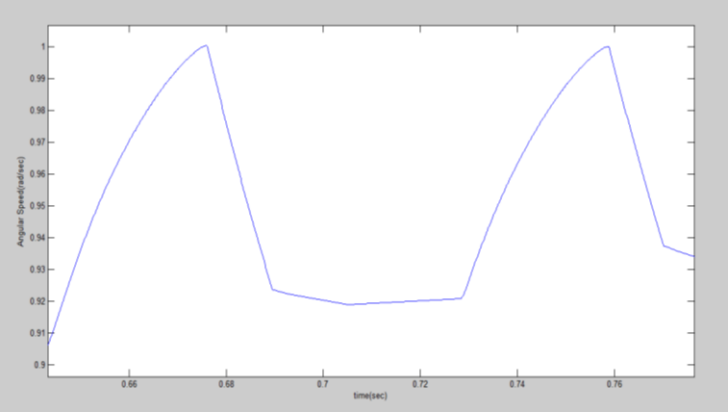


Figure 5.10: The settling time for proportional derivative controller by using fuzzy logic control with DC Motor system in MATLAB.

5.3 PD-FUZZY LOGIC CONTROL USING PARTICLE SWARM OPTIMIZATION ALGORITHM WITH DC MOTOR

The results are obtained that depending on the MATLAB Platform has been used and applied in DC Motor System and The results are checked and verified for Proportional derivative (PD) controller with Fuzzy Logic Control (FLC) using Particle Swarm Optimization (PSO) to check what this controller that improved for DC Motor System by using Fuzzy Logic Control (FLC) with Particle Swarm Optimization (PSO),The figures below that describe the rising time and (there is no overshoot),settling time and steady state error such as the following:

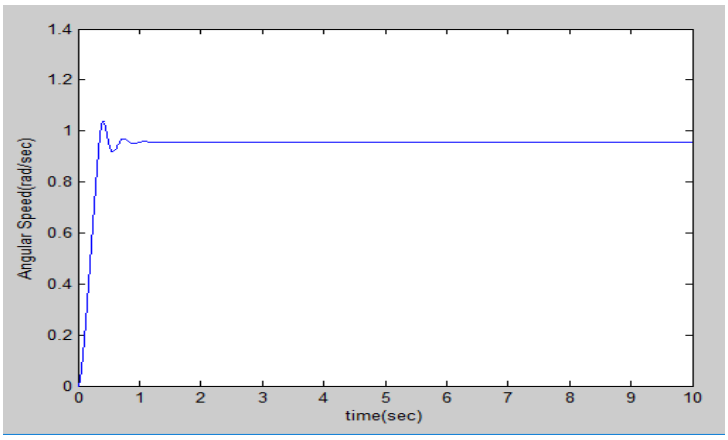


Figure 5.11: The general curve for proportional derivative controller by using fuzzy logic control with PSO Algorithm with DC Motor system in MATLAB.

As we see in the figure above there is on overshoot in the figure of particle swarm optimization when used in fuzzy logic control to improve the DC Motor system so this considered as best results,in additional to that the system is also very stable.

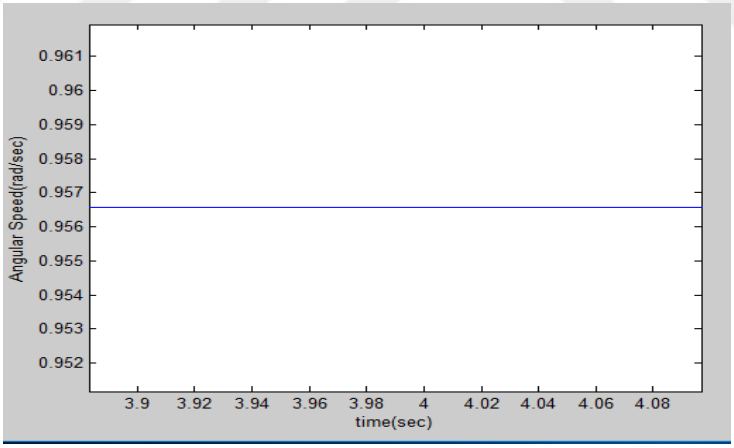


Figure 5.12: The steady state error for proportional derivative controller by using fuzzy logic control with PSO Algorithm with DC Motor system in MATLAB.

In the figure above describe the steady state value when stabled in this figure so the value is 0.9565=yss,and the math calculation will be taken below in this equation.

$ess = 1 - yss = 1 - 0.9565 = 0.0435$

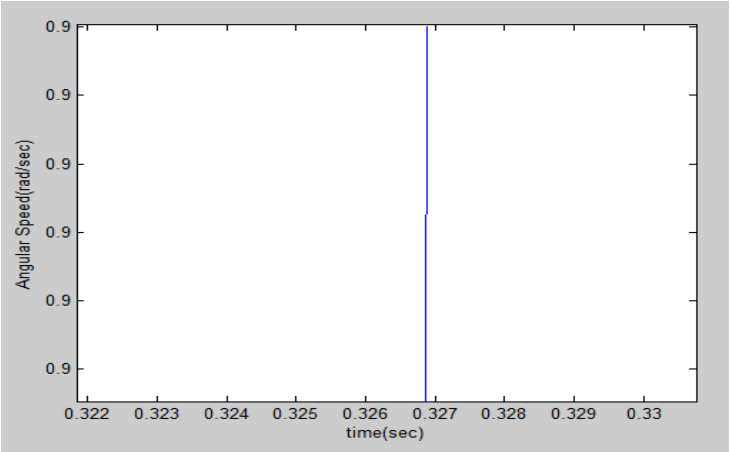


Figure 5.13: The rising time for proportional derivative controller by using fuzzy logic control with PSO Algorithm with DC Motor system in MATLAB.

The above figure describe the time that the curve of Proportional derivative (PD) controller with Fuzzy logic control by using Particle swarm optimization (PSO) has been rising and this considered as 90% from the upper value in the beginning of the curve and its has settled in 0.326 (sec) as shown in the above figure.

In the figure below describe the settling time that the curve still repeat the same value as shown in the figure that mentioned below its start from 1.05 (sec) and its considered as the time of the response when settled.

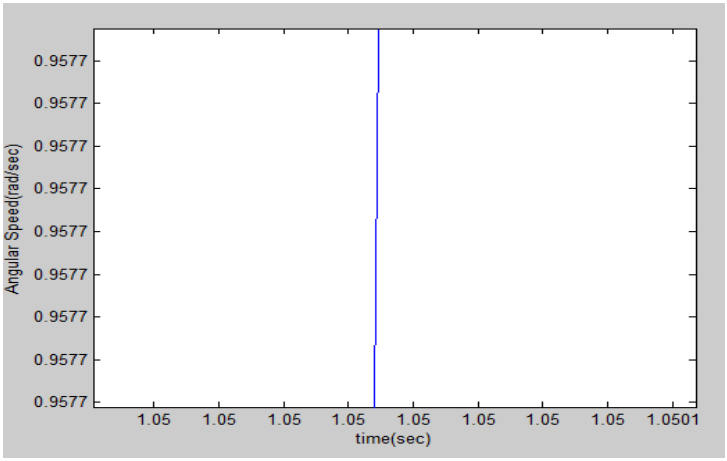


Figure 5.14: The settling time for proportional derivative controller by using fuzzy logic control with PSO Algorithm with DC Motor system in MATLAB.

The figure below describe the final figures,and it's the comparison between Proportional Derivative (PD) controller and when Proportional Derivative (PD) controller used the fuzzy logic control and finally is the best than them is Proportional Derivative (PD) controller used the fuzzy logic control by using particle swarm optimization (PSO) Algorithm.

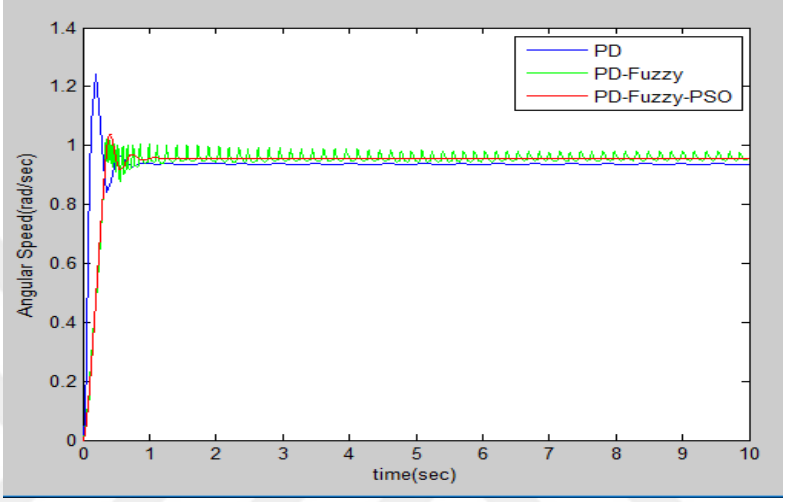


Figure 5.15: the comparison on Angular speed between PD(Proportional Derivative)and PD(Proportional Derivative) with Fuzzy logic control and PD(Proportional Derivative) with Fuzzy logic control using PSO(Particle Swarm Optimization).

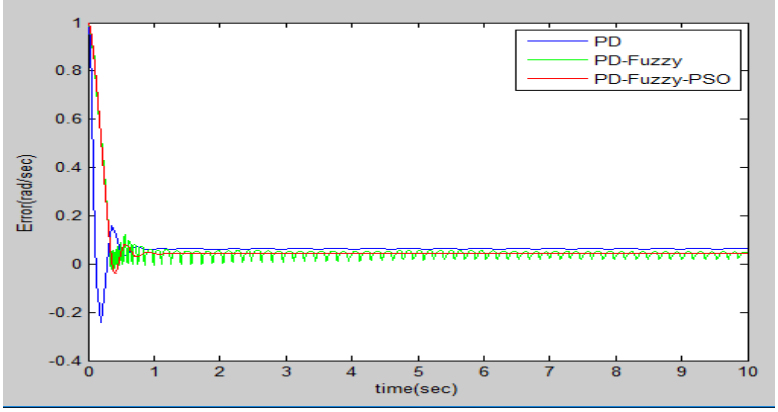


Figure 5.16: the comparison on Error between PD(Proportional Derivative)and PD(Proportional Derivative) with Fuzzy logic control and PD(Proportional Derivative) with Fuzzy logic control using PSO(Particle Swarm Optimization).

The figure above describe the comparison between PD and PD-Fuzzy and PD-FUZZY-PSO when we used the particle swarm optimization (PSO) Algorithm this algorithm that improved the

system of DC Motor so the Red curve in the curve is the best than them because there is no overshoot and also its stable than the others and when the Proportional derivative used the fuzzy logic control we received the good results also but the curve is not stable because of normalization so every signal that input to the fuzzy its should to input to the rules and the mechanism of rules is normalization so the result its also best than PD results but its not stable like it,the improvement of the system comes with PSO because it will looking for the best 50 birds by 10 iteration to get the best 3 scale factors to be used instead of the values that has been putted in the fuzzy logic control so the algorithm give us the best gains for each execution operation so in PSO, we getted the best high angular speed and the best low error so we improve the system by PSO Algorithm as shown in both curves the time that PSO settled in 1 (sec).

In the table below describe the comparison between PD and PD-FUZZY and PD-FUZZY-PSO Algorithm.

Table 3.5: The final Results between PD and PD-FUZZY and PD-FUZZY-PSO

Control Method	**PD**	**PD-FUZZY**	**PD-FUZZY-PSO**
Overshoot %	32.54 %	4.07272727%	NO-Overshoot
SteadyState Error	0.0622	0.05	0.0435
Rising Time	0.112 (sec)	1.2641 (sec)	0.326 (sec)
Settling Time	2.5 (sec)	0.72 (sec)	1.05(sec)

6. CONCLUSION

The proposed system that we have create it is DC Motor system and this system has created by using MATLAB SIMULINK and we used the basic tools and also calculate a math model to create it and also the laplace transform we used it to convert all the equations to blocks because MATLAB SIMULINK not accept equations that not deals with laplace transform and the Math model its similar to the model that used in popular papers and we based on Proportional Derivative (PD) controller by scale factor values for (Kp=150 and Kd=0.01)and these values has been checked and putted in the scale factors this depending on previous studying so we getted a high overshoot for PD controller (32.54 %) with DC Motor System and high steady state error (0.0622) and this results considered as drawback because of its weak so we have improve it by two methods firstly by use the fuzzy logic control with PD controller and getted a low overshoot (4.07272727%) and low steady state error (0.05) but in use PSO with Fuzzy logic control and PD controller so we getted (No-Overshoot) in use PSO and a low steady state error (0.0435) and this considered as a good result, and now we have proved that PSO with fuzzy logic control based on Proportional derivative (PD) controller is better than use PD or PD with Fuzzy.

In future work, I would recommend any one want to completing on my thesis is to use another optimization method and apply it on DC Motor with fuzzy logic control, such as use SSO Algorithm (Sperm Swarm Optimization) or CSO (Cat Swarm Optimization) or GSO (Glowworm Swarm Optimization) and other optimization methods ,I think these Optimization methods it will give the best results than PSO (Particle Swarm Optimization) when Applied in Fuzzy logic control based on PD Controller.

REFERENCES

[1] F. Salas, M. Llama, and V. Santibanez, "A stable self-organizing fuzzy PD control for robot manipulators," *Int. J. Innov. Comput. Inf. Control*, vol. 9, no. 5, pp. 2065–5086, 2013.

[2] D. Preeti and N. Beniwal, "Comparison of Conventional and Fuzzy P/PI/PD/PID Controller for Higher Order Non Linear Plant with High Dead Time," *Int. J. Sci. Res.*, vol. 2, no. 8, p. 217, 2012.

[3] G. Eappen and T. Shankar, "Optimization of two area AGC based power system using PSO tuned Fuzzy PID controller and PSO trained SSSC and TCPS," *Int. J. Eng. Technol.*, vol. 7, pp. 163–168, 2018.

[4] S. Lin and G. Wang, "Fuzzy PID control algorithm based on PSO and application in BLDC motor," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 69, no. 1, 2017.

[5] H. Kala, D. Deepakraj, P. Gopalakrishnan, P. Vengadesan, and M. Karumbal Iyyar, "Performance Evaluation of Fuzzy Logic and PID Controller for Liquid Level Process," *Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng.*, vol. 2, no. 3, pp. 2321–2004, 2014.

[6] N. Bouarroudj, D. Boukhetala, and F. Boudjema, "A hybrid fuzzy fractional order PID sliding-mode controller design using PSO algorithm for interconnected nonlinear systems," *Control Eng. Appl. Informatics*, vol. 17, no. 1, pp. 41–51, 2015.

[7] R. Rahmani, M. S. Mahmodian, S. Mekhilef, and A. A. Shojaei, "Fuzzy logic controller optimized by particle swarm optimization for DC motor speed control," *SCOReD 2012 - 2012 IEEE Student Conf. Res. Dev.*, pp. 109–113, 2012.

[8] B. Allaoua, A. Laoufi, B. Gasbaoui, and A. Nasri, "Intelligent Controller Design for DC Motor Speed Control based on Fuzzy Logic-Genetic Algorithms Optimization," *Sci. York*, no. 13, pp. 90–102, 2008.

[9] S. Wu, "A PID controller parameter tuning method based on improved PSO," *Int. J. Adv. Comput. Res.*, vol. 8, no. 34, pp. 41–46, 2018.

[10] O. Karasakal, E. Yeşil, M. Güzelkaya, and I. Eksin, "Implementation of a new self-tuning fuzzy PID controller on PLC," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 13, no. 2, pp. 277–286, 2005.

[11] M. Nasri, H. Nezamabadi-pour, and M. Maghfoori, "Design of a PID controller using PSO algorithm incorporating fuzzy objective function," *6th Iran. Conf. Fuzzy Syst. 1st Islam. world*

fuzzy Syst. Conf., no. SEPTEMBER 2006, pp. 157–168, 2006.

[12] X. L. Xie and Z. Long, "Fuzzy PID Temperature Control System Design Based on Single Chip Microcomputer," *Int. J. Online Eng.*, vol. 11, no. 8, pp. 29–33, 2015.

[13] V. Kumar, K. P. S. Rana, and V. Gupta, "Real-Time Performance Evaluation of a Fuzzy PI + Fuzzy PD Controller for Liquid-Level Process," *Int. J. Intell. Control. Syst.*, vol. 13, no. 2, pp. 89–96, 2008.

[14] S. Panda, B. K. Sahu, P. K. Mohanty, T. K. Pati, and S. K. Kar, "Design and analysis of fuzzy PID controller with derivative filter for AGC in multi-area interconnected power system," *IET Gener. Transm. Distrib.*, vol. 10, no. 15, pp. 3764–3776, 2016.

[15] C.-T. Chao, N. Sutarna, J.-S. Chiou, and C.-J. Wang, "Equivalence between Fuzzy PID Controllers and Conventional PID Controllers," *Appl. Sci.*, vol. 7, no. 6, p. 513, 2017.

[16] F. Van Den Bergh, "An Analysis of Particle Swarm Optimizers," *PhD thesis*, no. November, p. University of Pretoria, 300 pp, 2001.

[17] H. Mojallali, R. Gholipour, A. Khosravi, and H. Babaee, "Application of Chaotic Particle Swarm Optimization to PID Parameter Tuning in Ball and Hoop System," *Int. J. Comput. Electr. Eng.*, vol. 4, no. 4, pp. 452–457, 2012.

[18] I. M. M. El Emary, W. Emar, and M. J. Aqel, "The adaptive fuzzy designed PID controller using wavelet network," *Comput. Sci. Inf. Syst.*, vol. 6, no. 2, pp. 141–163, 2009.

[19] J. X. Xu, C. C. Hang, and C. Liu, "Parallel structure and tuning of a fuzzy PID controller," *Automatica*, vol. 36, no. 5, pp. 673–684, 2000.

[20] S. Gharghory and H. Kamal, "Modified PSO for Optimal Tuning of Fuzzy PID," *Int. J. Comput. Sci.*, vol. 10, no. 2, pp. 462–471, 2013.

[21] R. Shakya, K. Rajanwal, S. Patel, and S. Dinkar, "Design and Simulation of PD , PID and Fuzzy Logic Controller for Industrial Application," *Int. J. Inf. Comput. Technol.*, vol. 4, no. 4, pp. 363–368, 2014.

[22] J. Jantzen, "Tuning Of Fuzzy PID Controllers," *Tech. Univ. Denmark, Dep. Autom.*, vol. 871, no. 98–H, pp. 1–22, 1998.

[23] O. Karahan and Z. Bingül, “PD-fuzzy controller tuned with PSO for robot trajectory control,” *IFAC Proc. Vol.*, vol. 2, no. PART 1, 2009.

[24] Pardeep Rohilla and Abhishek Pratap Singh, “Analysis of a Fuzzy Self Tuning PID Controller for Nonlinear Pressure Control System.,” *Int. J. Eng. Res.*, vol. V4, no. 04, pp. 665–668, 2015.

[25] S. Vaishnav and Z. Khan, “Design and performance of PID and fuzzy logic controller with smaller rule set for higher order system,” *Proc. World Congr. …*, pp. 24–27, 2007.

[26] K. F. Hussein, I. Abdel-Qader, and M. K. Hussain, “Hybrid fuzzy PID controller for buck-boost converter in solar energy-battery systems,” *IEEE Int. Conf. Electro Inf. Technol.*, vol. 2015–June, pp. 70–75, 2015.

[27] E. C. Chang, Z. Su, Z. Xu, and R. C. Wu, “Particle swarm optimization tuned fuzzy terminal sliding mode control for UPS inverters,” *J. Intell. Fuzzy Syst.*, vol. 29, no. 6, pp. 2483–2488, 2015.

[28] A. Ganguly, A. B. Lane, W. Bengal, and P. Singh, “Design of Tuning Methods of PID Controller Using Fuzzy Logic,” *Int. J. Emerg. trends Eng. Dev.*, vol. 5, no. 3, pp. 239–249, 2013.

[29] S. Das, I. Pan, and S. Das, “Performance comparison of optimal fractional order hybrid fuzzy PID controllers for handling oscillatory fractional order processes with dead time,” *ISA Trans.*, vol. 52, no. 4, pp. 550–566, 2013.

[30] M. Z. Al-Faiz and S. A. Sadeq, “Particle Swarm Optimization Based Fuzzy-Neural Like PID Controller for TCP/AQM Router,” *Intell. Control Autom.*, vol. 03, no. 01, pp. 71–77, 2012.

[31] M. B. B. Sharifian, H. Afsharirad, and S. Galvani, “A particle swarm optimization approach for optimum design of PID controller in linear elevator,” *2010 9th Int. Power Energy Conf. IPEC 2010*, vol. 19, no. 2, pp. 451–455, 2010.

[32] A. R. Abdulnabi, “PID Controller Design for Cruise Control System using Particle Swarm Optimization,” *Iraqi J. Comput. Informatics*, vol. 43, no. 2, pp. 30–35, 2018.

[33] I. Atacak and B. Küçük, “PSO-based PID controller design for an energy conversion system using compressed air,” *Teh. Vjesn. - Tech. Gaz.*, vol. 24, no. 3, 2017.

[34] J. Jantzen, “Design of Fuzzy Controllers,” *Zhonghua Kou Qiang Ke Za Zhi*, vol. 21, no. 5, p. 264–266,317, 1986.

[35] S. M. Morkos and H. A. Kamal, “Optimal tuning of PID controller using adaptive hybrid particle swarm optimization algorithm,” *Int. J. Comput. Commun. Control*, vol. 7, no. 1, pp. 101–114, 2012.

[36] K. Premkumar and B. V. Manikandan, “Bat algorithm optimized fuzzy PD based speed controller for brushless direct current motor,” *Eng. Sci. Technol. an Int. J.*, vol. 19, no. 2, pp. 818–840, 2016.

[37] M. S. Abou Omar, T. Y. Khedr, and B. A. Abou Zalam, “Particle swarm optimization of fuzzy supervisory controller for nonlinear position control system,” *Proc. - 2013 8th Int. Conf. Comput. Eng. Syst. ICCES 2013*, pp. 138–145, 2013.

[38] M. L. Hernández-Nieto, J. de J. Moreno-Vázquez, R. Antonio-Ortiz, and A. R. Sartorius-Castellanos, “A novel method for fuzzy scale factors scheduling in fuzzy PD+I with anti-windup system controllers,” *Rev. Fac. Ing. Univ. Antioquia*, no. 80, pp. 142–151, 2016.

[39] A. H. Gomaa Haroun and Y. ya Li, “A novel optimized hybrid fuzzy logic intelligent PID controller for an interconnected multi-area power system with physical constraints and boiler dynamics,” *ISA Trans.*, vol. 71, pp. 364–379, 2017.

[40] A. Patel and K. Parikh, “Speed Control of DC Motor Using PSO Tuned PI Controller,” *IOSR J. Electr. Electron. Eng.*, vol. 9, no. 2, pp. 04–08, 2014.

[41] Bassi, Mishra, and Omizegba, “Automatic Tuning Of Proportional-Integral-Derivative (Pid) Controller Using Particle Swarm Optimization (Pso) Algorithm,” *Int. J. Artif. Intell. Appl.*, vol. 2, no. 4, pp. 25–34, 2011.

[42] Y. Zennir and M. S. Larabi, “A comparative study of PID-PSO and Fuzzy-PID controller design for Time Delay System,” *Int. J. FUZZY Syst. Adv. Appl.*, vol. 4, pp. 55–60, 2017.

[43] K. W. Yu and J. H. Hsu, “Fuzzy gain scheduling PID control design based on particle swarm optimization method,” *Second Int. Conf. Innov. Comput. Inf. Control. ICICIC 2007*, pp. 2007–2010, 2008.

[44] 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.

[45] 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.

[46] H. Kala, D. Deepakraj, P. Gopalakrishnan, P. Vengadesan, and M. Karumbal Iyyar, "Performance Evaluation of Fuzzy Logic and PID Controller for Liquid Level Process," *Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng.*, vol. 2, no. 3, pp. 2321–2004, 2014.

[47] G. Eappen and T. Shankar, "Optimization of two area AGC based power system using PSO tuned Fuzzy PID controller and PSO trained SSSC and TCPS," *Int. J. Eng. Technol.*, vol. 7, pp. 163–168, 2018.

[48] Ł. Niewiara and K. Zawirski, "Auto-tuning o f PID controller based on fuzzy logic," *Comput. Appl. Electr. Eng.*, vol. 11, pp. 230–240, 2013.

[49] Y. Al-Dunainawi and M. F. Abbod, "PSO-PD fuzzy control of distillation column," *IntelliSys 2015 - Proc. 2015 SAI Intell. Syst. Conf.*, pp. 554–558, 2015.

[50] 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.

[51] B. Akbiyik, I. Eksin, M. Guzelkaya, and E. Yesil, "Evaluation of the Performance of Various Fuzzy PID Controller Structures on Benchmark Systems," *4th Int. Conf. Electr. Electron. Eng.*, no. 1, 2005.

[52] 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.

APPENDIX A

[DC MOTOR PARAMETERS IN MATLAB CODE]

```
clc;
clear all;
close all;
%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
```

A.1 PROPORTIONAL DERIVATIVE CONTROLLER VALUES IN MATLAB CODE

```
Kp=150;

Kd=0.01;
```

A.2 SCALE FACTORS VALUES FOR FUZZY LOGIC IN MATLAB CODE

```
K1=5;

K2=0.1;

K3=50;
```

A.3 SCALE FACTORS VALUES FOR FUZZY LOGIC WITH PARTICLE SWARM OPTIMIZATION IN MATLAB CODE

```
K1=4.1004;

K2=-0.0091;

K3=50;
```

APPENDIX B

[PARTICLE SWARM OPTIMIZATION ALGORITHM IN MATLAB CODE]

```
%% Initialization
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
readfis('NORFLC');
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)







%
```

B.1 PARTICLE SWARM OPTIMIZATION TUNNING IN MATLAB CODE

```
function F = NOR_Tunning(SF)
         % Track the output of optsim to a signal of 1

         % Variables a1 and a2 are shared with RUNTRACKLSQ
         K1 = SF(1);
         K2 = SF(2);
          K3 = SF(3);
```

```
         sprintf('The value of interation K1= %3.0f, K2= %3.0f,K3= %3.0f',
SF(1),SF(2),SF(3));
         % Compute function value
         simopt =
simset('solver','ode5','SrcWorkspace','Current','DstWorkspace','Current');  %
Initialize sim options
         [tout,xout,yout] = sim('DC_Motor_FLC_PSO',[0 10],simopt);
%          e=yout-1 ;  % compute the error
%           sys_overshoot=max(wm)-1; % compute the overshoot
 end1=length(e2);
%         alpha=10;beta=10;
%          F=e2(end1)*beta+sys_overshoot*alpha;
     F=e2(end1);
    end
```

B.2 FUZZY INFERENCE SYSTEM RULES IN MATLAB CODE

```
[System]
Name='NORFLC'
Type='mamdani'
Version=2.0
NumInputs=2
NumOutputs=1
NumRules=25
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'

[Input1]
Name='e'
Range=[-1 1]
NumMFs=5
MF1='NB':'trimf',[-1.5 -1 -0.5]
MF2='NS':'trimf',[-1 -0.5 0]
MF3='Z':'trimf',[-0.5 0 0.5]
MF4='PS':'trimf',[0 0.5 1]
MF5='PB':'trimf',[0.5 1 1.5]

[Input2]
Name='de'
Range=[-1 1]
NumMFs=5
MF1='NB':'trimf',[-1.5 -1 -0.5]
MF2='NS':'trimf',[-1 -0.5 0]
MF3='Z':'trimf',[-0.5 0 0.5]
MF4='PS':'trimf',[0 0.5 1]
MF5='PB':'trimf',[0.5 1 1.5]

[Output1]
Name='u'
Range=[-1 1]
NumMFs=5
```

```
MF1='NB':'trimf',[-1.5 -1 -0.5]
MF2='NS':'trimf',[-1 -0.5 0]
MF3='Z':'trimf',[-0.5 0 0.5]
MF4='PS':'trimf',[0 0.5 1]
MF5='PB':'trimf',[0.5 1 1.5]

[Rules]
1 1, 1 (1) : 1
1 2, 1 (1) : 1
1 3, 2 (1) : 1
1 4, 2 (1) : 1
1 5, 3 (1) : 1
2 1, 1 (1) : 1
2 2, 2 (1) : 1
2 3, 2 (1) : 1
2 4, 3 (1) : 1
2 5, 4 (1) : 1
3 1, 2 (1) : 1
3 2, 2 (1) : 1
3 3, 3 (1) : 1
3 4, 4 (1) : 1
3 5, 4 (1) : 1
4 1, 2 (1) : 1
4 2, 3 (1) : 1
4 3, 4 (1) : 1
4 4, 4 (1) : 1
4 5, 5 (1) : 1
5 1, 3 (1) : 1
5 2, 4 (1) : 1
5 3, 4 (1) : 1
5 4, 5 (1) : 1
5 5, 5 (1) : 1
```