Design of Fractional Order PID Controller using Particle Swarm Optimization for Speed Control of DC Motor

A Thesis submitted in partial fulfillment of the Requirement for the award of degree of

Master of Engineering

in

Electronic Instrumentation and Control

Submitted by

Rinku Kumar
Roll No: 801051013

Under the Guidance of

Dr. Gagandeep Kaur
Assistant Professor

Department of Electrical and Instrumentation Engineering

Thapar University
(Established under the section 3 of UGCact,1956)

Patiala, 147004, Punjab, India
July 2012

DECLARATION

I hereby certify that the work is being presented in the thesis work entitled **"Design of Fractional Order PID Controller Using Particle Swarm Optimization for Speed Control of DC Motor"** in partial fulfillment of award of degree of **Master of Engineering** in **Electronics Instrumentation and Control** submitted in Electrical and Instrumentation Engineering department, Thapar University, Patiala is an authentic record of my own work carried under the supervision of **Dr. Gagandeep Kaur,** Assistant Professor, Department of Electrical and Instrumentation Engineering, Thapar University, Patiala, Punjab.

Date: 25/06/2012

Rinku Kumar
801051013

I certify that the above statement made by the student is correct to the best of my knowledge and belief.

Date:

Dr. Gagandeep Kaur
Assistant Professor,
Department of Electrical
and Instrumentation Engineering,
Thapar University, Patiala,
Punjab

Countersigned by

Dr. Smarajit Ghosh
Head of Department,
Department of Electrical and
Instrumentation Engineering,
Thapar University, Patiala,
Punjab

Dr. Saroj Kumar Mohapatra
Dean of Academic Affairs,
Thapar University, Patiala,
Punjab

ABSTRACT

Fractional order calculus is a emerging area and is used in many real life applications. Fractional order calculus is used in control system to design controllers which perform in a superior manner than the conventional controller. Fractional order PID controller is used in most of the applications because it gives more flexibility than the conventional controller.

This thesis implements conventional PID controller and fractional PID controller to control the speed of DC motor. It performs a comparative study of both the controllers. Fractional order controller has two main parameters λ and μ. To find out the optimal values of λ and μ, particle swarm optimization technique is used. It is observed that the fractional order PID controller gives superior performance in speed control of DC motor as compared to conventional PID controller.

ACKNOWLEDGEMENT

The real spirit of achieving a goal is through the way of excellence and austere discipline. I would have never succeeded in completing my task without the cooperation, encouragement and help provided to me by various personalities.

I am very thankful to **Dr. Abhijit Mukherjee**, Director of Thapar University, Patiala for providing the facilities for the completion of this thesis.

I express my deep sense of gratitude towards **Dr. Smarajit Ghosh**, Professor and Head of the Department of Electrical & Instrumentation Engineering, Thapar University, Patiala who has been a constant source of inspiration for me throughout this work.

With deep sense of gratitude I express my sincere thanks to my esteemed and worthy supervisor **Dr. Gagandeep Kaur**, Department of Electrical and Instrumentation Engineering, Thapar University, Patiala for her valuable guidance in carrying out this work under her effective supervision, encouragement, enlightenment and cooperation. Most of the novel ideas and solutions found in this thesis are the result of our numerous stimulating discussions. Her feedback and editorial comments were also invaluable for writing of this thesis.

We express our deep sense of gratitude to Mr. Subhransu Padhee for his valuable and encouraging guidance, which played a major role in the formation of this project work. Their co-oporation and timely suggestions have unparallel stimuli for us to travel eventually towards the completion of this project.

I am also thankful to all the staff members of the Department for their full cooperation and help. This acknowledgement would be incomplete if I do not mention the emotional support and blessings provided by my friends. I had a pleasant enjoyable and fruitful company with them.

Place: Thapar University, Patiala Rinku Kumar

Date: 801051013

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBERIVATIONS

DC	Direct current motor
R	Resistance of armature
L	Inductance of armature winding
i_a	Armature current
i_f	Field current
e_a	Applied armature voltage
e_b	Back emf
T_m	Motor torque
Θ	Angular displacement of motor shaft
Ω	Angular speed of motor shaft
J	Moment of inertia
B	Friction coefficient
Φ	Flux
P	Proportional controller
PI	Proportional integral controller
PD	Proportional derivative controller
PID	Proportional integral derivative controller
FOPID	Fractional order proportional integral derivative cotroller
$\mathbf{K_p}$	Proportional gain
$\mathbf{K_i}$	Integral gain
$\mathbf{K_d}$	Derivative gain
$\mathbf{K_{cu}}$	Controller gain
Λ	Fractional order integral
M	Fractional order derivative
ISE	Integral square error
IAE	Integral absolute error
IE	Integral error
ITAE	Integral time absolute error
M	Number of iterations
F	Fitness function
B	Scaling factor
$\mathbf{M_p}$	Peak overshoot

$\mathbf{T_p}$	Peak time
$\mathbf{T_s}$	Settling time
SI	Swarm intelligence
CI	Computational intelligence
EC	Evolutionary computation

RELATED PUBLICATION

Rinku Singhal, Subhransu Padhee, Gagandeep Kaur, “Design of Fractional Order PID Controller for Speed Control of DC Motor” in International Journal of Scientific and Research Publications (ISSN:2250–3153,Volume-2,Issue-6) June 2012.

CHAPTER 1
INTRODUCTION

1.1 Overview

Electrical drives are the essential part of any manufacturing industry and so as their controlling but it should be controlled in an efficient way so that the losses should be minimum without making any compromise with controlling accuracy. Most of the industries uses PID controller as controlling device. Fractional order calculus has gained acceptance in last couple of decades. J Liouville made the first major study of fractional calculus in 1832. In 1867, A. K. Grunwald worked on the fractional operations. G. F. B. Riemann developed the theory of fractional integration in 1892. Fractional order mathematical phenomena allow us to describe and model a real object more accurately than the classical "integer" methods. Earlier due to lack of available methods, a fractional order system was used to be approximated as an integer order model. But at the present time, there are many available numerical techniques which are used to approximate the fractional order derivatives and integrals. A typical example of a non-integer (fractional) order system is the voltage-current relation of a semiinfinite lossy transmission line.

The past decade has seen an increase in research efforts related to fractional calculus and use of fractional calculus in control system. For a control loop perspective there are four situations like (i) integer order plant with integer order controller, (ii) integer order plant with fractional order controller, (iii) fractional order plant with integer order controller, (iv) fractional order plant with fractional order controller. Fractional order control enhances the dynamic system control performance

1.2 Objective and Scope of Dissertation

The objective of this dissertation can be stated in three points

- The main objective of this dissertation to control the speed of DC motor by different type of control techniques.
- Design and implementation of PID and fractional order PID (FOPID) controllers
- Design the fractional order PID controller using soft computing techniques, here particle swarm optimization is used to tune the parameters of FOPID.

- Evaluate the performance of conventional and intelligent controllers. To evaluate the performance of the controller, time response analysis is carried out. The time response analysis consists of transient response and performance indices analysis.

1.3 Organization of Thesis

The dissertation is organized as follows.

Chapter 1 gives the introduction of the thesis.

Chapter 2 discuss about different type of electric devices, mainly this chapter deals with the DC motor, its controlling method and mathematical modeling.

Chapter 3 gives a relevant literature review regarding speed control of DC motor.

Chapter 4 gives the basic overview of control system and controller designing .this chapter mainly deals with tuning of PID and FOPID controller using conventional method.

Chapter 5 discuss soft computing techniques and designing of fractional order PID controller using particle swarm optimization algorithm.

Chapter 6 gives the detailed study of unit step response of DC motor speed control using different controlled techniques according to performance indices.

Chapter 7 provides conclusion of entire thesis work and proposes the future scope

CHAPTER 2
ELECTRIC DRIVES

2.1 Introduction

Drives can be defined as systems used for motion control e.g.-fans, robots, pumps etc. Prime mover is the main part of any drive which provides the movement, so it can be diesel engines, petrol engines, hydro motors, electric motors etc.

Drivers using electric motor as the prime movers are known as electrical drives. There are several advantages of electrical drives:

- Easy to control.
- High efficiency (switch mode converters and electrical motors are very efficient).
- Lesser pollution.
- Easy to store or transport energy

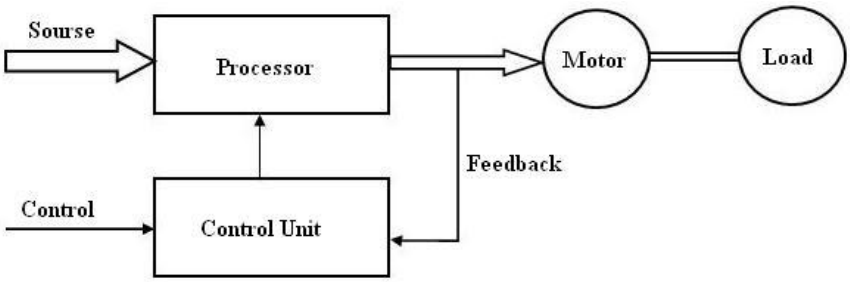

Figure 2.1 Components of electrical drives

2.2 Motors

Motors take the power from electrical source and convert that energy into mechanical energy, so it can be considered as energy converters. There are many types of motors which used in electrical drives, choice of motor depends on electrical source and application.

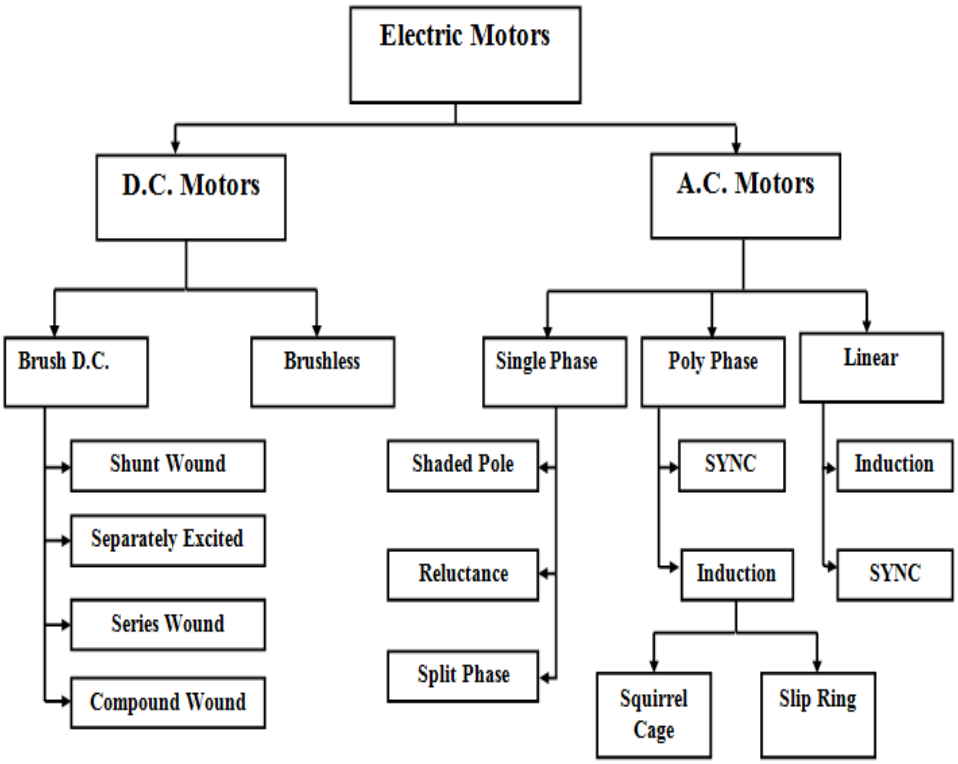

Figure 2.2 Classification of electric motors

2.3 DC Motors

A machine that converts dc power into mechanical power is known as a DC motor. Its operation is based on the principle that when a current carrying conductor is placed in a magnetic field, the conductor experiences a mechanical force. The direction of this force is given by Fleming's left hand rule and magnitude is given by equation 2.1

$$F = BIl \text{ newtons} \qquad (2.1)$$

B = magnetic flux density in Wb/m^2.

l = length of the conductor in meters.

I = current in the conductor in Amp.

2.4 Working of DC Motor

Consider a part of a multipolar DC motor as shown in Figure 2.3. When the terminals of the motor are connected to an external source of DC supply,

(i) The field magnets are excited developing alternate N and S poles.

(ii) The armature conductors carry currents. All conductors under N-pole carry currents in one direction while all the conductors under S-pole carry currents in the opposite direction.

Let the conductors under N-pole carry currents into the plane of the paper and those under S-pole carry currents out of the plane of the paper as shown in Figure 2.3. Since each armature conductor is carrying current and is placed in the magnetic field, mechanical force acts on it, referring to Figure 2.3 and applying Fleming's left hand rule, it is clear that force on each conductor is tending to rotate the armature in anticlockwise direction. All these forces add together to produce a driving torque which sets the armature rotating. When the conductor moves from one side of a brush to the other, the current in that conductor is reversed and at the same time it comes under the influence of next pole which is of opposite polarity. Consequently, the direction of force on the conductor remains the same [1].

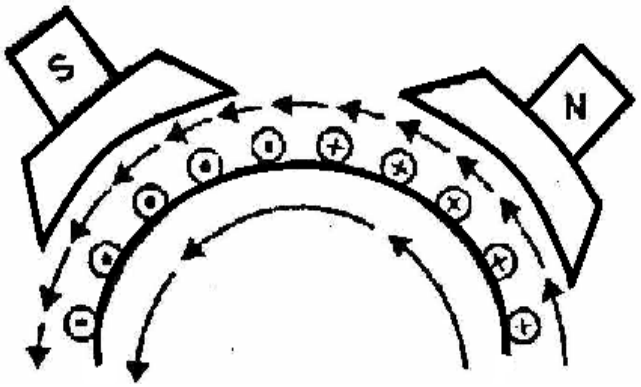

Figure 2.3 Multipolar DC motor

2.5 Types of DC Motor

The types of DC motors characterized by the connections of field winding in relation to the armature viz. DC motor can be classified as following-

- Separately excited
- Shunt wound motor
- Series wound motor
- Compound wound motor

2.5.1 Separately Excited

The field coil contains a relatively large number of turns which minimizes the current required to produce a strong stator field. It is connected to separate DC power supply, thus making field current independent of load or armature current. The connection diagram of separately excited DC motor is shown below in Figure 2.4.

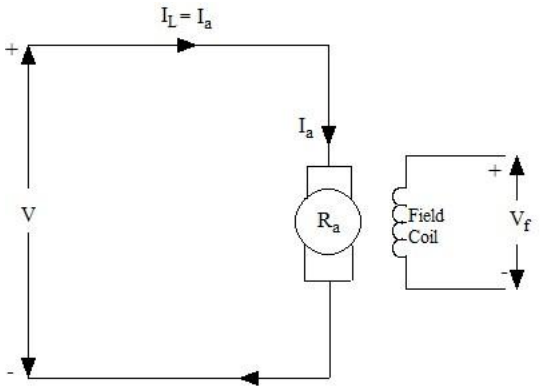

Figure 2.4 Separately excited dc motor

2.5.2 Shunt-Wound Motor

In shunt wound motor the field winding is connected in parallel with the armature as shown in figure 2.5. The current through the shunt field winding is not the same as the armature current. Shunt field windings are designed to produce the necessary m.m.f. by means of a relatively large number of turns of wire having high resistance. Therefore, shunt field current is relatively small compared with the armature current

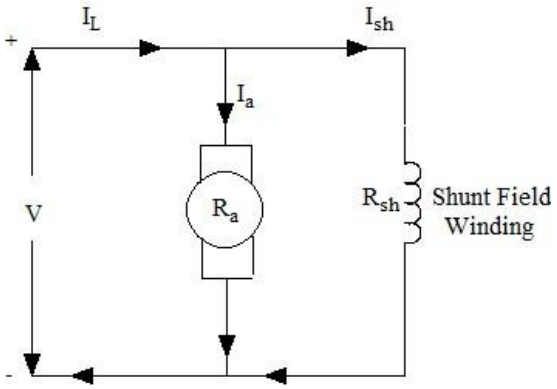

Figure 2.5 Shunt wound DC motor

2.5.3 Series-Wound Motor

In series wound motor the field winding is connected in series with the armature as shown in Figure 2.6. Therefore, series field winding carries the armature current. Since the current passing through a series field winding is the same as the armature current, series field windings must be designed with much fewer turns than shunt field windings for the same m.m.f. Therefore, a series field winding has a relatively small number of turns of thick wire and, therefore, will possess a low resistance.

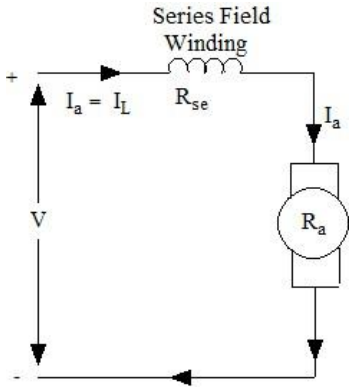

Figure 2.6 Series wound DC motor

2.5.4 Compound-Wound Motor

Compound wound motor has two field windings; one connected in parallel with the armature and the other in series with it. There are two types of compound motor connections. When the shunt field winding is directly connected across the armature terminals it is called short-shunt connection shown in Figure 2.7. When the shunt winding is so connected that it shunts the series combination of armature and series field it is called long-shunt connection shown in Figure 2.8. The compound machines are always designed so that the flux produced by shunt field winding is considerably larger than the flux produced by the series field winding. Therefore, shunt field in compound machines is the basic dominant factor in the production of the magnetic field in the machine.

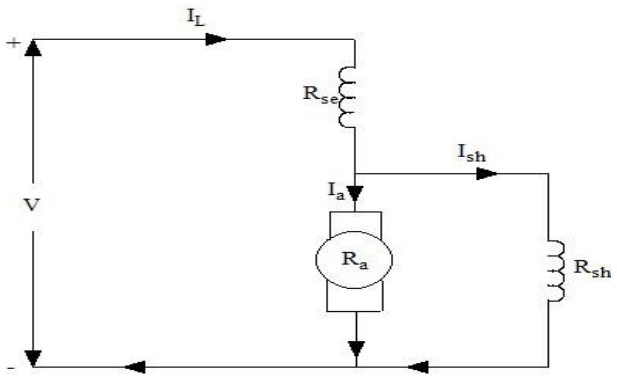

Figure 2.7 Sort shunt connection of compound wound DC motor

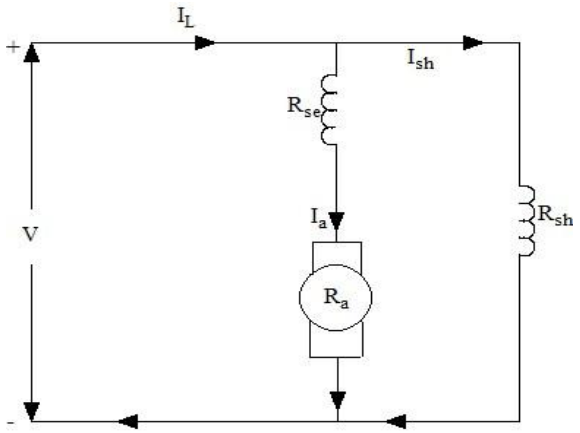

Figure 2.8 Sort shunt connection of compound wound dc motor

2.6 Comparison of Characteristics of Motors

- The speed regulation of a shunt motor is better than that of a series motor. However, speed regulation of a cumulative compound motor lies between shunt and series motors as shown in Figure 2.9.

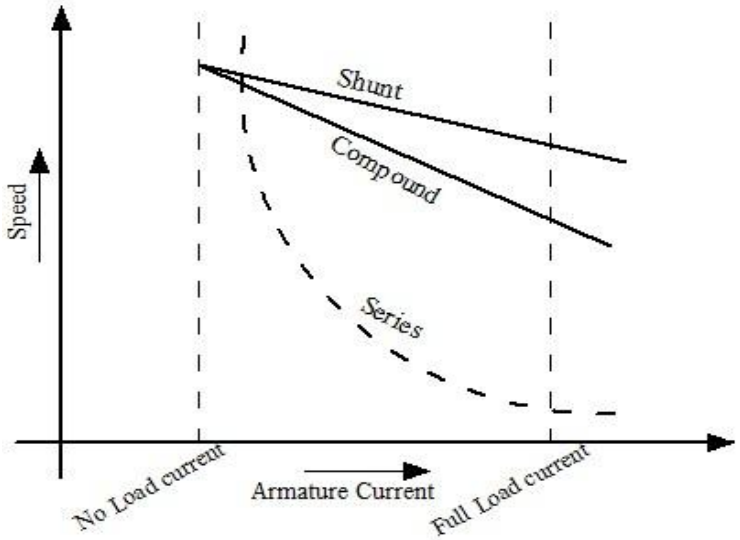

Figure 2.9 Current-Speed curves for different motors

- For a given armature current, the starting torque of a series motor is more than that of a shunt motor. However, the starting torque of a cumulative compound motor lies between series and shunt motors as shown in Figure 2.10.

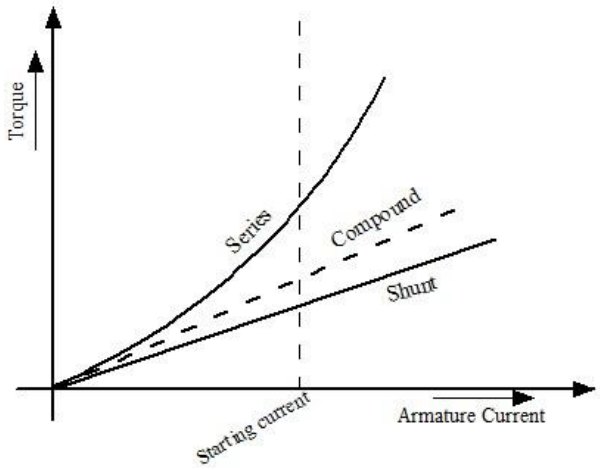

Figure 2.10 Current-Torque curves for different motors

- Both shunt and cumulative compound motors have definite no-load speed. However, a series motor has dangerously high speed at no-load.
- Excellent speed regulation is characteristic of separately excited DC motor which lends itself well to speed control by variation of field or armature current.

2.7 Applications of DC Motors [2]

Separately Excited

Separately excited DC motor can race to dangerously high speeds (theoretically infinity) if current to the field coil is lost. Because of this, applications should include some form of field protection as an unprotected motor could literally fly apart.

Industrial use: Train and automotive traction applications.

Shunt Motors

The characteristics of a shunt motor reveal that it is an approximately constant speed motor. It is, therefore, used

- where the speed is required to remain almost constant from no-load to full-load
- the load has 10 be driven at a number of speeds and any one of which is required to remain nearly constant.

Industrial use: Lathes, drills, boring mills, shapers, spinning and weaving machines etc.

Series Motors

It is a variable speed motor i.e., speed is low at high torque and vice-versa. However, at light or no-load, the motor tends to attain dangerously high speed. The motor has a high starting torque. It is, therefore, used

- Where large starting torque is required e.g., in elevators and electric Traction.

- Where the load is subjected to heavy fluctuations and the speed is automatically required to reduce at high torques and vice-versa.

Industrial use: Electric traction, cranes, elevators, air compressors, vacuum cleaners, hair drier, sewing machines etc.

Compound Motors

Differential-compound motors are rarely used because of their poor torque characteristics. However, cumulative-compound motors are used where a fairly constant speed is required with irregular loads or suddenly applied heavy loads.

Industrial use: Presses, shears, reciprocating machines etc.

Table 2.1 Comparison between DC and AC drives [3]

DC Drives	AC Drives
The power circuit and control circuit is simple and inexpensive.	The power circuit and control circuit are complex.
It requires frequent maintenance.	Less maintenance.
The commutator makes the motor bulky , costly and heavy.	These problems are not there in these motors and are inexpensive, particularly squirrel cage induction motors.
Fast response and wide speed range of control can be achieved smoothly by conventional and solid state control.	In solid state control the speed range is wide and conventional method is stepped and limited.
Speed and design ratings are limited due to commutations.	Speed and design ratings have upper limits.

2.8 Control Methods for Separately Excited DC Motor

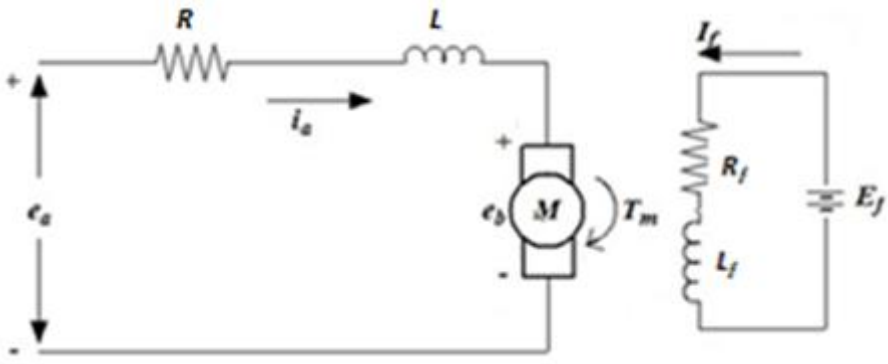

Figure 2.11 Detailed circuitry of DC drives

The torque is produced as a result of interaction of field flux with current in armature conductors and is given by equation 2.2

$$T_m = K_t \phi i_a \tag{2.2}$$

Where K_t a constant depending on motor windings and geometry is Φ is the flux per pole due to the field winding.

The direction of the torque produced depends on the direction of armature current. When armature rotates, the flux linking the armature winding will vary with time and therefore according to Faraday's law, an emf will be induced across the winding. This generated emf, known as the back emf, depends on speed of rotation as well as on the flux produced by the field and given by equation 2.3

$$e_b = K_t \phi \omega \tag{2.3}$$

Using KVL,

$$e_a = i_a R + L\frac{di_a}{dt} + e_b \tag{2.4}$$

In steady state condition

$$E_a = I_a R + E_b \tag{2.5}$$

In terms of torque and speed the steady state equation will be,

$$E_a = \frac{T_m}{K_t \phi} R + K_t \omega \phi \tag{2.6}$$

Which gives

$$\omega = \frac{E_a}{K_t \phi} - \frac{T_m}{(K_t \phi)^2} R \tag{2.7}$$

Thus from the above equation 2.7 it is clear that there are three methods of speed control, those are by varying E_a, R_a, and ϕ, i.e.:-

- Armature voltage controlled (E_a)
- Armature resistance controlled (R)
- Flux controlled (ϕ)

Speed control using armature resistance by adding external resistor R_{ext} is not used very widely because of the large energy lose due to the R_{ext} that can be given as ${I_a}^2 R_{ext}$. Armature voltage control is normally used for speed up to rated speed (base speed). Flux control is used for speed beyond rated speed but at the same time the maximum torque capability of the motor is reduced since for a given maximum armature current, the flux is less than the rated value and so as the maximum torque produced is less than the maximum rated torque.

Here the main attention is given to the armature voltage control method. In the armature voltage control method, the voltage applied across the armature e_a is varied keeping field voltage constant. As equation 2.7 indicates, the torque-speed characteristic is represented by a straight line with a negative slope when the applied armature voltage is ideal, that ideal torque speed characteristic is illustrated in Figure 2.12 [4].

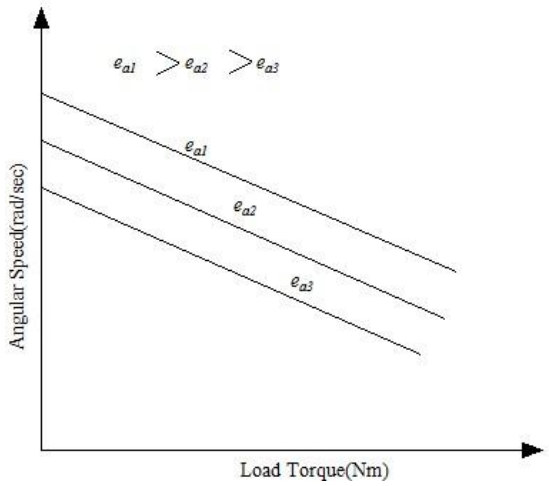

Figure 2.12 Torque -Speed characteristics of the separately excited DC motor at different armature voltages

Depending upon the type of AC source or the method of voltage control, dc drives are classified as,

i. Single Phase dc drives.
ii. Three Phase dc drives.
iii. Chopper drives.

2.9 Single Phase DC Drives

The general circuit arrangement for the speed control of a separately excited dc motor from a single phase source is shown in Figure 2.13. The firing angle control of converter 1 regulates the armature voltage applied to dc motor armature. Thus, the variation of delay angle α_1 of converter 1 gives speed control below base speed and converter 2 gives the speed control above base speed. The inductor *L* is used to achieve continuous armature current as the discontinuous armature current cause [5].

- More losses in armature.
- Poor speed regulation.

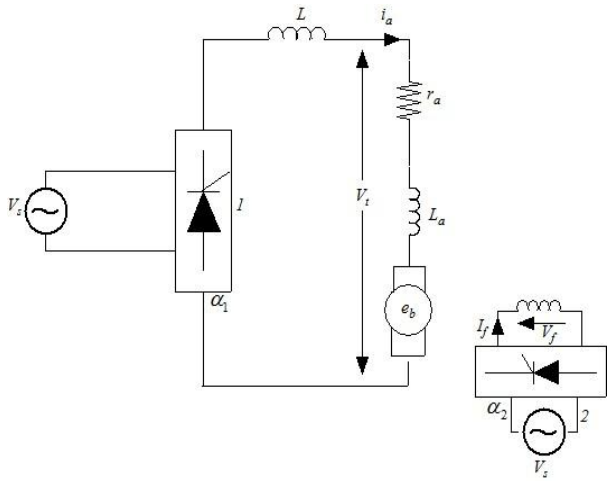

Figure 2.13 General circuit arrangement for single phase DC drives

Depending upon the type of power electronic converter used in the armature circuit, single phase dc drives may be subdivided as below

i. Single phase half wave converter drives.
ii. Single phase semiconverter drives.
iii. Single phase full converter drives.
iv Single phase dual converter drives.

2.9.1 Single Phase Half Wave Converter Drives

A separately excited dc motor fed through single phase half wave converter, is shown in Figure 2.14. Single phase half wave half wave converter feeding a dc motor offers only one quadrant drive. Such type of drives is used up to about 1/2 kW dc motor.

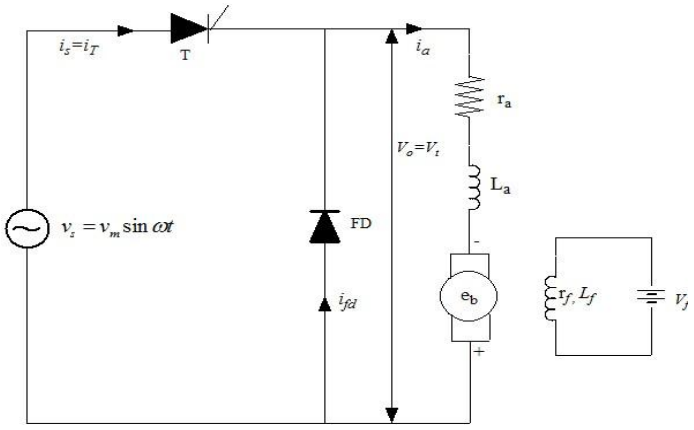

Figure 2.14 Single phase half wave converter drive

For single phase half wave converter, average output voltage of converter can be calculated as,

$$V_o = V_t = \frac{V_m}{2\pi}(1 + \cos\alpha) \qquad \text{for} \quad 0 < \alpha < \pi \tag{2.8}$$

A half wave converter in the field circuit will increase the magnetic losses of the motor due to high ripple content on the field excitation current, so an ideal dc source is preferred over half wave converter for field circuit.

2.9.2 Single Phase Semiconverter Drive

A separately excited dc motor fed through single phase semiconverter is shown in Figure 2.15. This converter also offer only one quadrant drive and is used up to 15 kW dc drives.

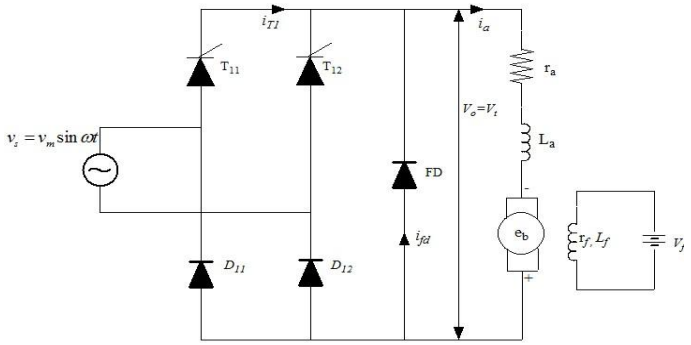

Figure 2.15 single phase semiconverter drive

With a single phase semi converter in the armature circuit, equation 2.9 gives the average armature voltage as

$$V_o = V_t = \frac{V_m}{\pi}(1 + \cos\alpha) \qquad \text{for} \quad 0 < \alpha < \pi \tag{2.9}$$

2.9.3 Single Phase Full Converter Drive

The armature voltage is varied by single phase full wave converter as shown in Figure 3.16. It is a two quadrant drive, and it is limited to application up to 15kW. The armature converter gives $+V_o$ or $-V_o$ and allows operation in the first and fourth quadrant. The converter in the field circuit could be semi, full or even dual converter. The reversal of the armature or field voltage allows operation in the second and third quadrant.

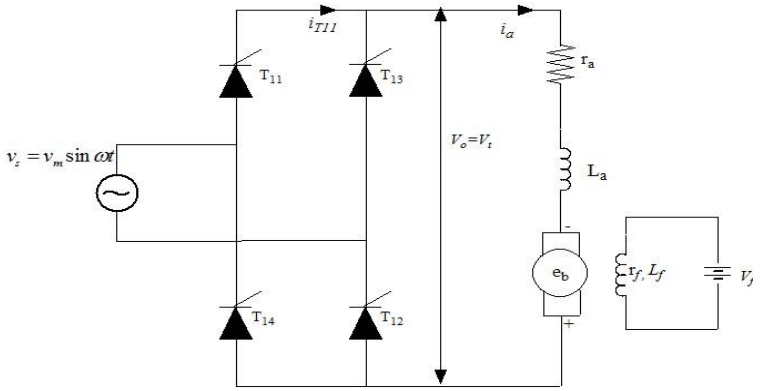

Figure 2.16 Single phase full converter drive

The average armature voltage in armature circuit for single phase full converter drive can be given by equation 2.10

$$V_o = V_t = \frac{2V_m}{\pi}(1 + \cos\alpha) \qquad \text{for} \quad 0 < \alpha < \pi \qquad (2.10)$$

2.9.4 Single Phase Dual Converter Drive

To realize single phase dual converter two single phase full converters are connected as shown in Figure 2.17.

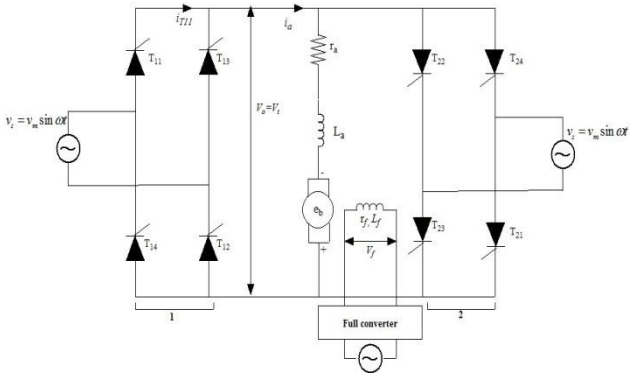

Figure 2.17 Single phase dual converter drive

As shown in above figure there are two single phase full wave converters either converter 1 operates to supply a positive armature voltage V_o, or converter 2 operates to supply negative armature voltage $-V_o$. Converter 1 provides operation in first and fourth quardants, and converter 2 provides operation in second and third quardants.It is four quardant drive and provides four modes of operation: forward powering, forward braking (regenration), reverse powering, and reverse breaking (regeneration). The field converter could be a full wave converter, a semiconverter, or a dual converter.

If converter 1 operates at a firing angle of α_1 then equation 2.11 gives the armature voltage as,

$$V_o = V_t = \frac{V_m}{\pi}(1 + \cos\alpha_1) \qquad \text{for} \quad 0 < \alpha_1 < \pi \qquad (2.11)$$

And similarly, if converter 2 operates at a firing angle of α_2 then equation 2.12 gives the armature voltage as,

$$V_o = V_t = \frac{V_m}{\pi}(1 + \cos\alpha_2) \qquad \text{for} \quad 0 < \alpha_2 < \pi \qquad (2.12)$$

Where $\alpha_2 = \pi - \alpha_1$

2.10 Mathematical Modeling of DC Motor

In control systems, the DC motors are used in two different control modes:-

- Armature control mode with fixed field current.

- Field control mode with fixed armature current.

Here armature control mode with fixed field current considered i.e. also known as armature-control [6,7]. In this system

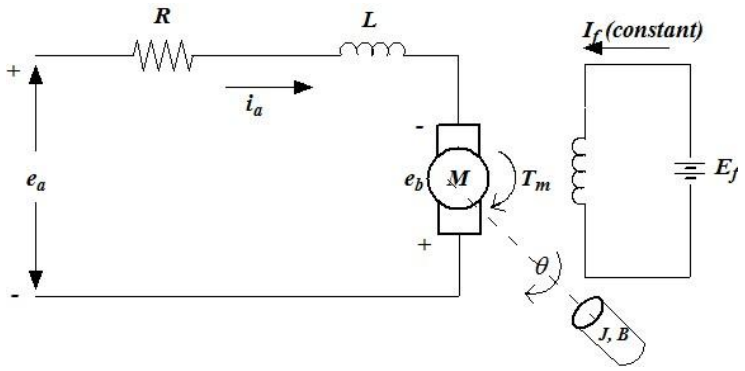

Figure 2.18 Separately excited constant field current DC motor

R = resistance of armature (Ω).

L =inductance of armature winding (H).

i_a = armature current (A).

i_f = field current (A).

e_a = applied armature voltage (V)

e_b = back emf (V)

T_m = torque developed by motor (Nm)

θ = angular displacement of motor shaft (rad).

ω = angular speed of motor shaft (rad/sec.)

J = equivalent moment of inertia of motor and load referred to motor shaft (kg-m^2)

B = equivalent friction coefficient of motor and load referred to motor shaft (Nm*s/rad)

In servo applications, the dc motors are generally used in the linear range of magnetization curve. Therefore, the air gap flux ϕ is proportional of field current, i.e.

$$\phi = K_f i_f \tag{2.13}$$

Where K_f is constant.

The torque T_m developed by the motor is proportional to the product of armature current and air gap flux, i.e.

$$T_m = K_1 K_f i_f i_a \tag{2.14}$$

Where K_1 is constant.

In armature controlled dc motor, the field current is kept constant, so

$$T_m = K_T i_a \tag{2.15}$$

Where K_T is known as the motor torque constant. The motor back emf being proportional to speed is given

$$e_b = K_b \frac{d\theta}{dt} \tag{2.16}$$

Where K_b is the back emf constant.

The differential equation of the armature circuit is

$$L\frac{di_a}{dt} + Ri_a + e_b - e_a = 0 \tag{2.17}$$

The torque equation is

$$J\frac{d^2\theta}{dt^2} + B\frac{d\theta}{dt} = T_m = K_T i_a \tag{2.18}$$

Taking the laplace transform of equations, assuming initial condition zero.

$$E_b(s) = K_b s\theta(s) \tag{2.19}$$

$$(Ls + R)I_a(s) = E_a(s) - E_b(s) \tag{2.20}$$

$$(Js^2 + Bs)\theta(s) = T_m(s) = K_T I_a(s) \tag{2.21}$$

So the final transfer function will be

$$\frac{\theta(s)}{E_a(s)} = \frac{K_T}{s[(R + sL)(Js + B) + K_T K_b]} \tag{2.22}$$

Or

$$G(s) = \frac{\omega(s)}{E_a(s)} = \frac{K_T}{(R + sL)(Js + B) + K_T K_b} \tag{2.23}$$

Block diagram can be realize using equation 2.23

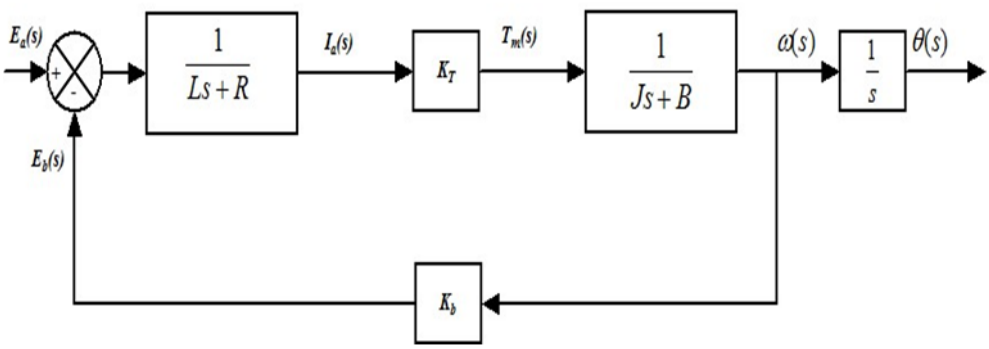

Figure 2.19 Block diagram of DC motor

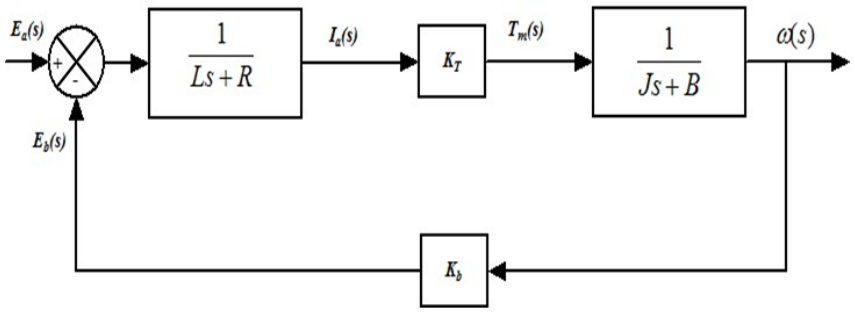

Figure 2.20 Block diagram of speed control of DC motor

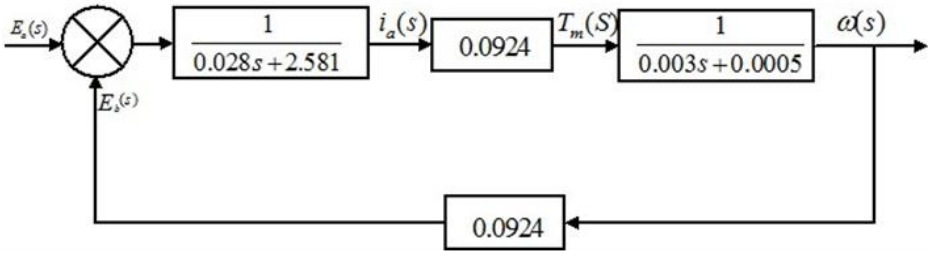

Figure 2.21 Block diagram of speed control of DC motor with transfer function

From the above figure 2.21 the transfer function can be given as equation 2.24

$$G(s) = \frac{0.0924}{8.49\times10^{-7}s^2 + 0.00585s + 0.01729} \tag{2.24}$$

Here in table 2.2 standard parameters are given of separately excited DC motor which are used to control the speed of DC motor in my research work:-

Table 2.2 The parameters of separately excited DC motor

Rated Power (P)	5 Hp
Rated Armature Voltage	240 V
Armature Resistance R	1.75 Ω
Armature Inductance L	0.028 H
Field Resistance R_f	281.3 Ω
Field Inductance L_f	156 H
Back EMF Constant K_b	0.0924
Motor Constant K_t	0.0924
Friction Coefficient of Motor B	0.0005 Nm*/rad
Moment of Inertia of Motor J	0.003 Kg-m^2
Rated Speed	1750 RPM
Rated Field Voltage	300 V

Above table 2.2 shows the parameters of separately excited DC motor. These are used to control the speed of DC motor using different type of control techniques.

CHAPTER 3
LITERATURE REVIEW

Schlegel Milos et.al, performed a comparison between classical controller and fractional controller and summarized that the fractional order controller outperforms the classical controller [8].

D Xue et.al, implemented fractional order PID control in DC motor [9].

Chuang Zhao et.al [10],Ramiro S Barbosa et.al [11] and Ying Luo et.al [12], implemented fractional order controller in position servo mechanism. FPID's experimental time response is compared with model predictive control. Theoretical analysis and experimental analysis are compared which shows the effectiveness of fractional order PID control.

Varsha Bhambhani et.al , implemented fractional order controller in water level control Experimental results confirmed that fractional order PI controller is a promising controller in terms of percentage overshoot and system response in liquid-level control in face of nonlinearities introduced by pumps, valves and sensors [13].

Hyo-Sung Ahn et.al implemented fractional order integral derivative controller for temperature profile control [14].

Concepcion A. Monje et.al [15], and Fabrizio Padula et.al [16], devised methods for tuning and auto tuning of fractional order controller for industry level control.

Venu Kishore Kadiyala ,Ravi Kumar Jatoth et.al , implemented fractional order controller for aerofin control The EMA is realized with permanent magnet brush DC motor which is driven by a constant current driver. Using the non-linear model of EMA-AFC system which includes the non-linearities of DC motor, an FOPID position controller is designed using different soft computing techniques like PSO [17].

Jun-Yi Cao et.al , implemented genetic algorithm and particle swarm optimization methods for optimization of fractional order controller. Fractional calculus can provide novel and higher performance extension for FOPID controllers. However, the difficulties of designing FOPID controllers increase, because FOPID controllers append derivative order and integral order in comparison with traditional PID controllers [18, 19].

Majid Zamani et.al implemented particle swarm optimization for robust performance of fractional order controller [20].

Deepyaman Maiti, et.al , implemented particle swarm optimization for design of fractional order controller Particle Swarm Optimization technique offers optimal or suboptimal solution to multidimensional rough objective functions [21].

Li Meng et.al, implemented multi objective genetic algorithm optimization for fractional order PID controller. The fractional-order PID controller provides more adjustable parameters in the controller optimization than conventional PID controller. Therefore, FOPID is designed to achieve more goals, and multi-objective optimization based genetic algorithm is adopted [22].

Arijit Biswas, et.al , implemented improved differential evolution techniques for design of fractional PID controller In order to digitally realize the fractional-order closed-loop transfer function of the designed plant, Tustin operator-based continuous fraction expansion (CFE) [23].

Ammar A Aldair et.al implemented evolutionary algorithm based fractional order controller for a full vehicle nonlinear activation suspension system [24].

CHAPTER 4
CONTROL SYSTEM

4.1 Introduction

Control system engineers are responsible to control a part of an environment known as a system or plant to produce desired products for society. The prior knowledge of system always played an important role in effective controlling. A control engineer should have the knowledge of different engineering principles like electrical, mechanical, and chemical etc.

Control system can be categorized as open-loop or close-loop feedback control and feedback control systems can be further classified a single-input-single-output (SISO) or multiple-input-multiple-output (MIMO), commonly known as multivariable system [25].

4.2 Open –Loop Control System

An open-loop control system is designed to meet the desired output by using a reference signal that operates the actuators that directly control the process output. Output feedback is not present in this type of system. Usually input is the ideal form of output. Figure 4.1 below shows the general structure of an open loop control system.

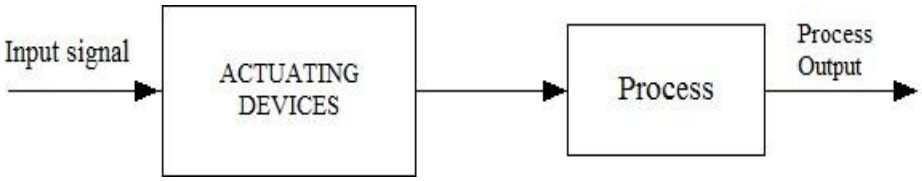

Figure 4.1 Open loop control system

4.3 Closed –Loop Control System

The close loop control systems consists output feedback controller .The controller processes the error signal which is the difference of current output and the desired output to achieve the desired output. Figure 4.2 shows the general structure of closed loop control system.

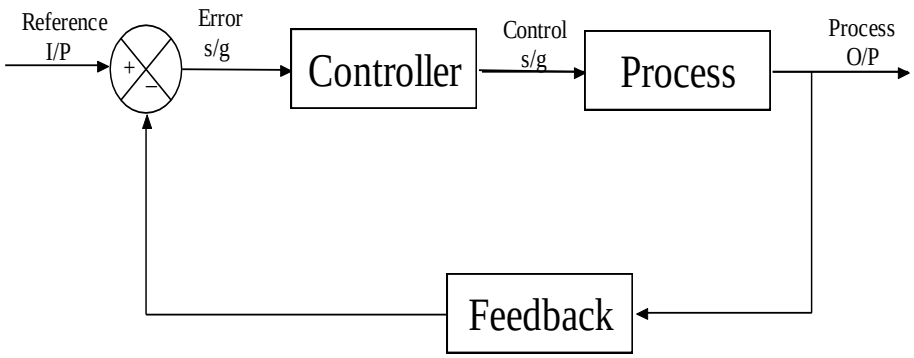

Figure 4.2 Closed loop control system

4.4 Types of Control Structures

There are various types of control structures are available, selection of control structure is very important. Some of control structures are listed below and further discussed with their respective advantages and disadvantages [26].

- Feedback control structure
- Feedforward control structure
- Feedback + Feedforward control structure
- Cascade control structure

4.4.1 Feedback Control Structure

Feedback control is a mechanism which regulates the controlled variable by taking negative feedback from the output and taking regulatory action through the controller and changing the manipulated variable accordingly. Figure 4.3 shows the feedback control structure.

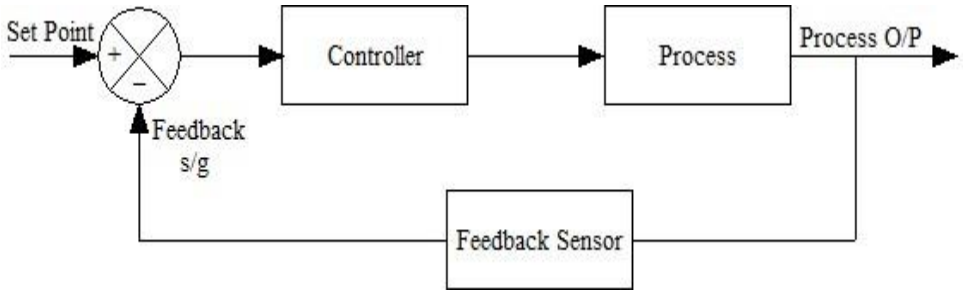

Figure 4.3 Feedback control structure

4.4.2 Feedforward Control Structure

The basic concept of the feed forward controller is to measure the disturbances and take corrective action before the disturbance upset the process that means the

corrective action will takes place before the error occurred. Figure 4.4 shows the feedforward control structure.

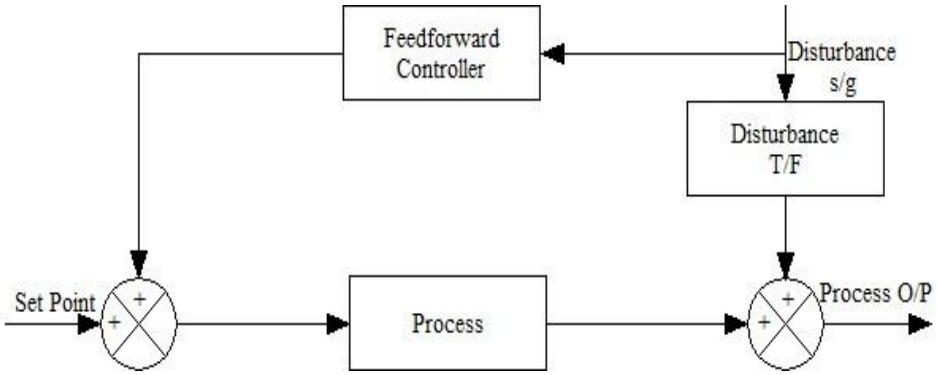

Figure 4.4 Feedforward control structure

4.4.3 Feedback and Feedforward Control Structure

In some systems the disturbance can be predicated there feedforward structure gives good result but in a process some unknown sources of disturbances are always there so for that kind of systems only feedforward structure is not sufficient, for that one should use feedback structure with feedforward structure. By using feedforward structure with feedback structure user get the advantages of both type of structure. Figure 4.5 shows the feedback and feedforward control structure.

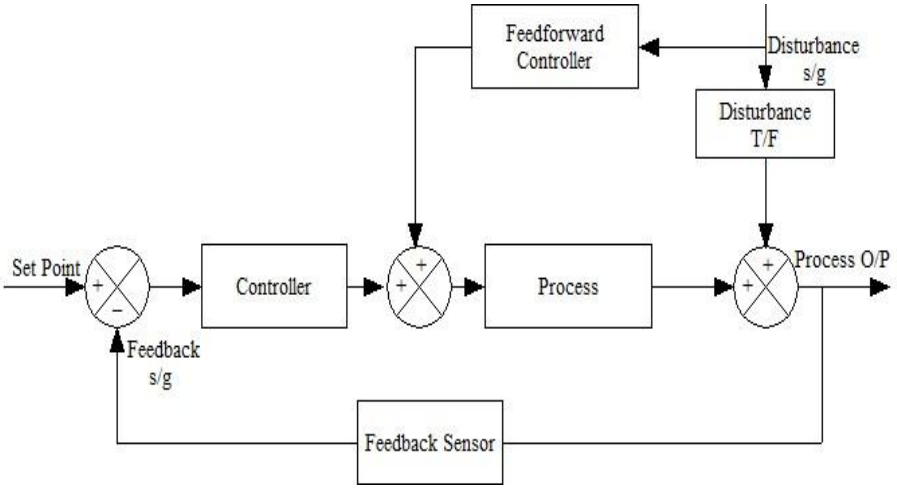

Figure 4.5 Feedback and feedforward control structure

4.4.4 Cascade Control Structure

In a cascade control, there are two or more controllers of which one controller's output drives the set point of another controller. The controller driving the set point is called the primary, outer, or master controller. The controller receiving the set point is called the secondary, inner or slave controller. Cascade control is beneficial only if the dynamics of the inner loop are fast compared to those of the outer loop. Figure 4.6 shows the cascade control structure.

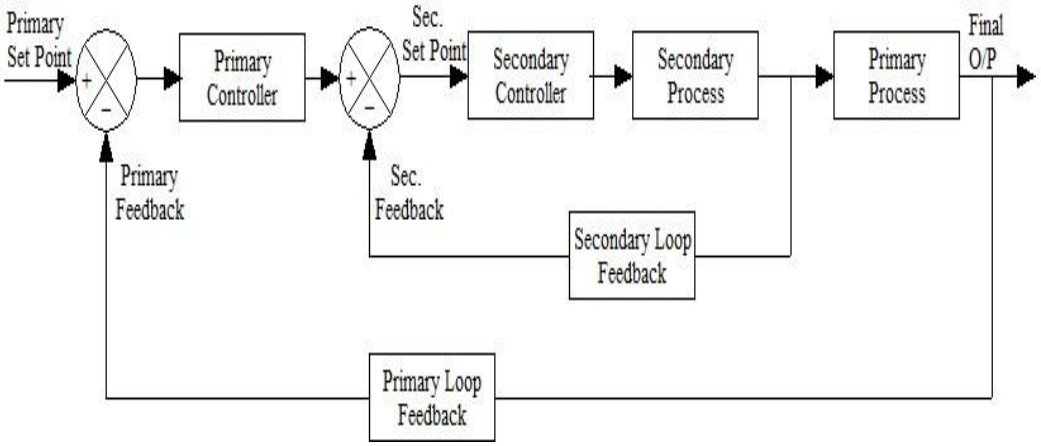

Figure 4.6 Cascade control structure

4.5 PID Controller

The Proportional-Integral-Derivative (PID) controllers have been the most commonly used controller in process industries for over 50 years even though significant development have been made in advanced control theory. According to a survey conducted by Japan Electric Measuring Instrument Manufacturers Association in 1989, 90 % of the control loops in industries are of the PID type [27]. A PID controller calculates an "error" value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error by adjusting the process control inputs.

PID control is a name commonly given to three-term control .The PID controller calculation involves three separate constant parameters [28].

- Proportional term
- Integral term
- Derivative term

PID controller can be describe by the following equation 4.1

$$u(t) = K_p e(t) + K_d \frac{de(t)}{dt} + K_i \int_0^t e(t) \tag{4.1}$$

The mnemonic PID refers to the first letters of the names of the individual terms that make up the standard three-term controller. These are P for the proportional term, I for the integral term and D for the derivative term in the controller.

The theoretical basis for analyzing the performance of PID control is considerably aided by the simple representation of an Integrator by the Laplace transform, 1/s, and a Differentiator usings.

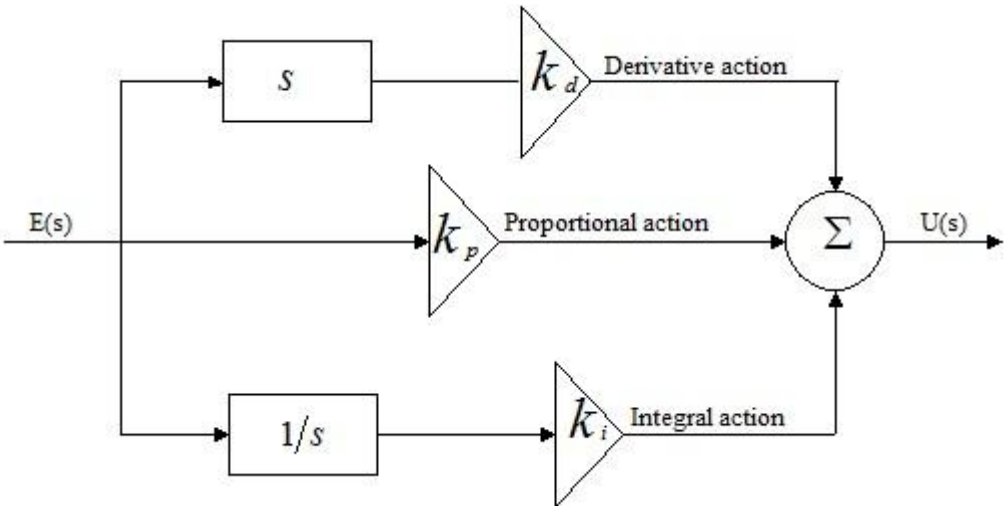

Figure 4.7 Integer PID controller

Proportional Action

The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant K_p, called the proportional gain constant. The proportional term is given by equation 4.2

$$p(t) = K_p e(t) \tag{4.2}$$

A high proportional gain results in a large change in the output for a given change in the error. If the proportional gain is too high, the system can become unstable. In contrast, a small gain results in a small output response to a large input error, and a less responsive or less sensitive controller. If the proportional gain is too low, the control action may be too small when responding to system disturbances. Tuning theory and

industrial practice indicate that the proportional term should contribute the bulk of the output change.

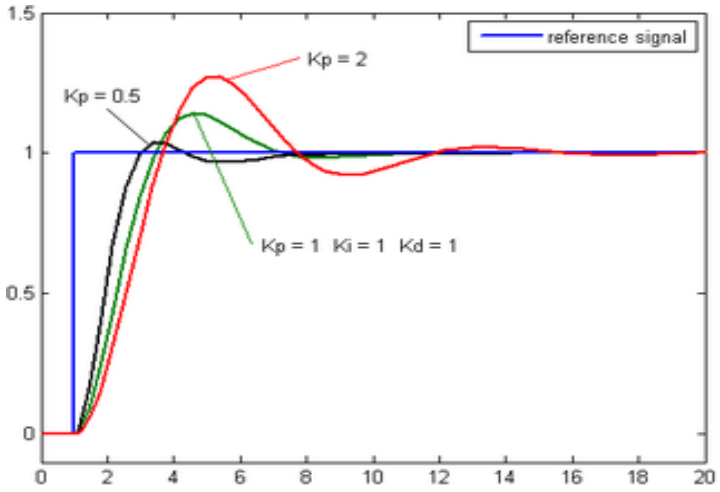

Figure 4.8 Transient response of second order system for different values of Kp

Integral Action

The contribution from the integral term is proportional to both the magnitude of the error and the duration of the error. The integral in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously. The accumulated error is then multiplied by the integral gain (K_i) and added to the controller output. The integral term is given by

$$I(t) = K_i \int_0^t e(t) \tag{4.3}$$

The integral term accelerates the movement of the process towards setpoint and eliminates the residual steady-state error that occurs with a pure proportional controller. However, since the integral term responds to accumulated errors from the past, it can cause the present value to overshoot the setpoint value.

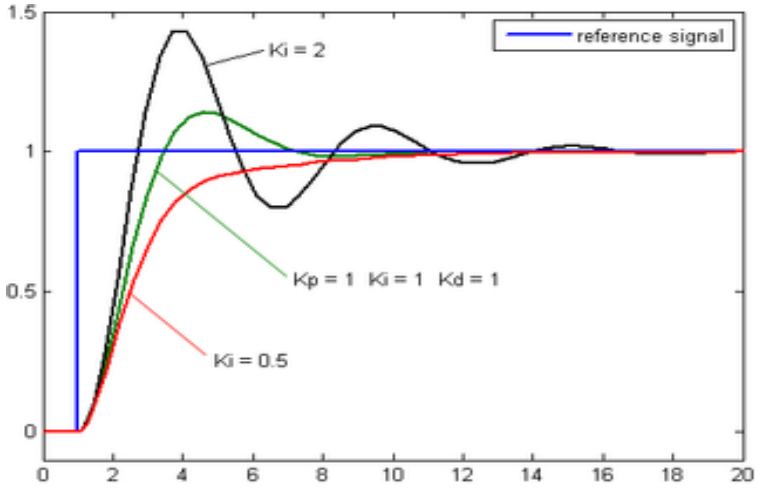

Figure 4.9 Transient response of second order system for different values of Ki

Derivative Action

The derivative of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain K_d. The magnitude of the contribution of the derivative term to the overall control action is termed the derivative gain, K_d .The derivative term is given by

$$D(t) = K_d \frac{de(t)}{dt} \tag{4.4}$$

The derivative term slows the rate of change of the controller output. Derivative control is used to reduce the magnitude of the overshoot produced by the integral component and improve the combined controller-process stability. However, the derivative term slows the transient response of the controller. Also, differentiation of a signal amplifies noise and thus this term in the controller is highly sensitive to noise in the error term, and can cause a process to become unstable if the noise and the derivative gain are sufficiently large. Hence an approximation to a differentiator with a limited bandwidth is more commonly used [28].

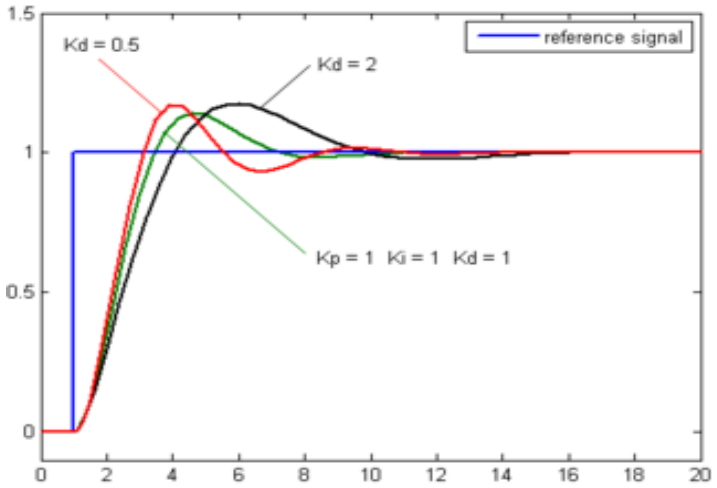

Figure 4.10 Transient response of second order system for different values of Kd

4.6 Performance Indices

The optimal control cannot be defined precisely in general. A solution, which is optimum for one particular application for given set of conditions, may not be optimal for another problem. In such case, defining performance criterion and minimizing the value of such performance index, optimal constants can be obtained as required for economic and practical problem.

One of such index is cost function, which is used by most of the control engineers. Such performance indices help us in assessing the quality of a control system [29].

Performance Index (P.I.)

A performance index is a quantitative measure of the performance of the system. A system is considered an optimal control system when the system parameters are adjusted so that the index reaches an extreme value, commonly a minimum value. Now we will discuss some of various error performance indices which are used in my thesis work.

- **ISE (Integral of Square Error)** A suitable fitness function (Performance index) is the integral of square error.

$$ISE = \int_0^\infty e^2(t)dt \tag{4.5}$$

- **IAE(Integral of Absolute Error)** Another fitness function is the integral of the absolute error as follow:

$$IAE = \int_0^\infty |e(t)|\, dt \tag{4.6}$$

This criterion penalized large errors less heavily and small errors more heavily compared to ISE.

- **ITSE(Integral of Time Multiplied Square Error Criterion)**

It is generally desirable to step response patterns when used as a basis for optimum design. This has a characteristic that in unit step response of the system a large initial error is weighted lightly, while errors occurring late in the transient response are penalized heavily.

$$ITSE = \int_0^\infty te^2(t)dt \tag{4.7}$$

- **ITAE(Integral of Time Multiplied by Absolute Value of Error)**:

It is achieved by adding time weighting to IAE and defined as:

$$ITAE = \int_0^\infty t|e(t)|\, dt \tag{4.8}$$

This generally leads to systems with reasonable transient characteristics. It is comparable in many respects to ITSE but is not mathematically analytic. This also has a property that initial large errors gets weighted lightly and late occurring error gets penalized heavily.

4.7 Characteristics of PID Controller

A proportional controller (Kp) reduces the rise time and steady state error. An integral control (Ki) eliminates the steady-state error but it may make the transient response worse. A derivative control (Kd) increases the stability of the system, reducing the overshoot, and improving the transient response. Effects of each of controllers Kp, Kd, and Ki on a closed-loop system are summarized in the table shown below [30].

Table 4.1 Effects of independent P, I, and D tuning

Controller Response	Rise Time	Overshoot	Settling Time	Steady State Error	Stability
Increasing K_P	Decrease	Increase	Small Increase	Decrease	Degrade
Increasing K_i	Small Decrease	Increase	Increase	Large Decrease	Degrade
Increasing K_d	Small Decrease	Decrease	Decrease	Minor Change	Improve

Note that these correlations may not be exactly accurate, because Kp, Ki, and Kd are dependent of each other. In fact, changing one of these variables can change the effect of the other two. For this reason, the table should only be used as a reference when you are determining the values for Ki, Kp and Kd [31].

4.8 Limitations of PID Controller

Several limitations are associated with PID controller [32]. Some of these are following

- Feedback controller
- Linearity
- Noise in derivative

4.8.1 Feedback Controller

The fundamental difficulty with PID control is that it is a feedback system, with constant parameters. It has no direct knowledge of the process, and thus overall performance is reactive and a compromise.

4.8.2 Linearity

Another problem with PID controller is faced that they are linear in nature .Thus performance of PID controllers in non-linear systems is variable. For example, in temperature control, a common use case is active heating (via a heating element) but passive cooling (heating off, but no cooling), so overshoot can only be corrected slowly – it cannot be forced downward. In this case the PID should be tuned to be overdamped, to prevent or reduce overshoot, though this reduces performance (it increases settling time).

4.8.3 Noise in Derivative

A problem with the derivative term is that small amounts of measurement or process noise can cause large amounts of change in the output. It is often helpful to filter the measurements with a low-pass filter in order to remove higher-frequency noise components. However, low-pass filtering and derivative control can cancel each other out, so reducing noise by instrumentation is a much better choice

4.9 Tuning of PID Controller

In the 1980s, when analogue control was being replaced by digital processing hardware, industrial control companies took the opportunity to develop new PID controller methods for use with the new ranges of controller technology appearing. Consequently, the Ziegler–Nichols methods became the focus of research and have since, become better understood. New versions of the Ziegler–Nichols procedures were introduced, notably the Åström and Hägglund relay experiment. In many applications, the implicit underdamped closed-loop performance inherent in the original Ziegler–Nichols design rules was found to be unacceptable. The result was an extensive development of the rule-base for PID controller tuning. O'Dwyer has published summaries of a large class of the available results. Continuing competitive pressures in industry have led to a constant need for continual improvements in control loop performance. One result of these trends is that industry is much better at being able to specify the type of performance that a control system has to deliver [33].

4.10 Ziegler-Nichols Tuning

In 1942, Ziegler and Nichols, both of them are the employees of Taylor Instruments. They described simple mathematical procedures in the form of the open loop and close loop methods, for tuning PID controllers. Their methods were used for non-first

order plus dead time situations, and involved intense manual calculations. With improved optimization software, most manual methods such as these are no longer used. However, even with computer aids, the following two methods are still employed today, and are considered among the most common. There are two methods of Ziegler-Nichols tuning

- Open loop method
- Closed loop method

These methods are explained in the following sections [34].

4.10.1 Open Loop Method

The Ziegler-Nichols open-loop method is also referred to as a process reaction method, because it tests the open-loop reaction of the process to a change in the control variable output. This basic test requires that the response of the system be recorded, preferably by a plotter or computer. Once certain process response values are found, they can be plugged into the Ziegler-Nichols equation with specific multiplier constants for the gains of a controller with either P, PI, or PID actions. The response is characterized by two parameters, L the delay time and T the time constant. These are found by drawing a tangent to the step response at its point of inflection and noting its intersections with the time axis and the steady state value. The plant model is therefore

$$G(s) = \frac{Ke^{-sL}}{Ts} \tag{4.9}$$

Where:

G(s) = Transfer function of the process

K = Gain of the process

T = Time constant of the process

Ziegler and Nichols derived the following control parameters based on this model

Table 4.2 Ziegler-Nichols recipe- open loop method

PID Type	Kp	Ti=Kp/Ki	Td=Kd/Kp
P	$\frac{T}{L}$	∞	0
PI	$0.9\frac{T}{L}$	$\frac{L}{0.3}$	0
PID	$1.2\frac{T}{L}$	$2L$	$0.5L$

In the above table 4.2, proportional gain, integral time and derivative time for different controller types are derived using ultimate gain and ultimate period using Ziegler-Nichols open loop method.

4.10.2 Close Loop Method

The Ziegler-Nichols closed-loop tuning method allows to use the ultimate gain value, K_{cu}, and the ultimate period of oscillation, P_u, to calculate K_p . You can obtain the controller constants K_p , T_I , and T_D in a system with feedback.

The steps for tuning a PID controller via the close loop method are as follows:
Using only proportional feedback control.
Step1- Remove the integral and derivative actions.
Step2- Adjust the proportional gain by increasing or decreasing until at which sustained oscillation occurs.
Step3- Record the gain ultimate gain value Kcu and ultimate period of oscillation Pu
The controller gains are now specified as follows

Table 4.3 Ziegler Nichols recipe- closed method

PID Type	**K_p**	**T_i**	**T_d**
P	$0.5K_{cu}$	∞	0
PI	$0.45K_{cu}$	$\frac{P_u}{1.2}$	0
PID	$0.6K_{cu}$	$\frac{P_u}{2}$	$\frac{P_u}{8}$

In the above table 4.3 proportional gain, integral time and derivative time for different controller types are derived using ultimate gain and ultimate period using Ziegler-Nichols second method.

4.11 Fractional Order Controller

Fractional order controller is described by fractional order differential equations. Fractional calculus allows the derivatives and integrals to be any real number. Expanding derivatives and integrals to fractional orders can adjust control system's frequency response directly and continuously. This great flexibility makes it possible to design more robust control system [35]. The fundamental advantage of FOC is that the fractional order integrator weights history using a function that decays with a power-law tail. The effect is

that the effects of all time are computed for each iteration of the control algorithm. This creates a 'distribution of time constants,' the upshot of which is there is no particular time constant, or resonance frequency, for the system.

Fractional order control shows promise in many controlled environments that suffer from the classical problems of overshoot and resonance, as well as time diffuse applications such as thermal dissipation and chemical mixing. Fractional controller can be designed in both frequency and time domain [36].

4.12 Fractional Calculus

Fractional order calculus has gained acceptance in last couple of decades. J Liouville made the first major study of fractional calculus in 1832. In 1867, A. K. Grunwald worked on the fractional operations. G. F. B. Riemann developed the theory of fractional integration in 1892. Fractional order mathematical phenomena allow us to describe and model a real object more accurately than the classical “integer” methods. Earlier due to lack of available methods, a fractional order system was used to be approximated as an integer order model. But at the present time, there are many available numerical techniques which are used to approximate the fractional order derivatives and integrals. A typical example of a non-integer (fractional) order system is the voltage-current relation of a semiinfinite lossy transmission line [37].

The past decade has seen an increase in research efforts related to fractional calculus and use of fractional calculus in control system. For a control loop perspective there are four situations like

1. Integer order plant with integer order controller.
2. Integer order plant with fractional order controller.
3. Fractional order plant with integer order controller.
4. Fractional order plant with fractional order controller.

Fractional order control enhances the dynamic system control performance.

4.13 Fractional Order Calculus: Mathematical

Fractional order calculus is an area where the mathematicians deal with derivatives and integrals from non-integer orders. Gamma function is simply the generalization of the factorial for all real numbers. The definition of the gamma function is given by

$$\Gamma(x) = \int_0^{\infty} z^{x-1} e^{-z} dz \tag{4.10}$$

$$\Gamma(x) = (x-1)! \tag{4.11}$$

Differintegral Operator

Differintegral operator is denoted by ${}_aD_t^{\alpha}$. It is the combination of differentiation and integration operation commonly used in fractional calculus. Reimann- Liouville definition for ${}_aD_t^{\alpha}$ is

$${}_aD_t^{\alpha} = \begin{cases} \dfrac{d^{\alpha}}{dt^{\alpha}} & \alpha > 0 \\ 1 & \alpha = 0 \\ \displaystyle\int_a^t (d\tau)^{-\alpha} & \alpha < 0 \end{cases} \tag{4.12}$$

Here α is the fractional order. a and t are the limits. There are two commonly used definitions for general Differintegral ${}_aD_t^{\alpha}$.

(1) Grunwald - Letnikov

(2) Riemann- Liouville

Grunwald – Letnikov Definition

$${}_aD_t^{\alpha} f(t) = \lim_{h \to 0} \frac{1}{h^{\alpha}} \sum_{j=0}^{\left[\frac{t-a}{h}\right]} (-1)^j \binom{\alpha}{j} f(t - jh) \tag{4.13}$$

$[.]$ is a flooring-operator here

$$\binom{\alpha}{j} = \frac{\Gamma(\alpha+1)}{\Gamma(j+1)\Gamma(\alpha-j+1)} \tag{4.14}$$

Riemann- Liouville Definition

$${}_aD_t^{\alpha} f(t) = \frac{1}{\Gamma(n-\alpha)} \frac{d^n}{dt^n} \int_a^t \frac{f(\tau)}{(t-\tau)^{\alpha-n+1}} d\tau \tag{4.15}$$

The condition for above equation is $n-1 < \alpha < n$. $\Gamma(.)$ is called the gamma function.

Laplace Transform of Differintegral Operator

Differintegral operator is denoted by ${}_aD_t^\alpha$ and the Laplace transform of Differintegral operator is represented as

$$L\left[{}_aD_\alpha^t f(t)\right] = \int_0^\infty e^{-st}\,{}_aD_\alpha^t f(t)\,dt \tag{4.16}$$

$$L\left[{}_aD_\alpha^t f(t)\right] = s^\alpha F(s) - \sum_{m=0}^{n-1} s(-1)^j\,{}_0D_t^{\alpha-m-1} f(t) \tag{4.17}$$

Here n lies in between $n-1<\alpha\le n$

4.14 Properties of Fractional Calculus

The main properties of fractional derivative and integrals are following [38]

- If f(t) is an analytical function of t ,its fractional derivative ${}_0D_t^\alpha f(t)$ is an analytical function of *z* and α
- For a =*n*, where *n* is an integer, the operation ${}_0D_t^\alpha f(t)$ gives the same result as classical differentiation of integer order *n*.
- For α= 0 the operation ${}_0D_t^\alpha f(t)$ is the identity operator:

$${}_0D_t^0 f(t) .= f(t). \tag{4.18}$$

- Fractional differentiation and fractional integration are linear operations

$${}_0D_t^\alpha af(t) + bg(t) = a\,{}_0D_t^\alpha f(t) + b\,{}_0D_t^\alpha \tag{4.19}$$

- The additive index law (semigroup property)

$${}_0D_t^\alpha\,{}_0D_t^\beta = {}_0D_t^\beta\,{}_0D_t^\alpha = {}_0D_t^{\alpha+\beta} f(t) \tag{4.20}$$

Holds under some reasonable constraints on the function $f(t)$.

4.15 Fractional Order Calculus in Control

Fractional order controller can be broadly classified as following

- Fractional order PID controller
- CRONE controller (command robust d'ordre non entire)
- TID controller (tilted proportional integral controller)
- Fractional lead leg controller

4.15.1Fractional Order PID Controller

One of the primary controllers is PID controller, which is widely used. Fractional controller is denoted by $PI^{\lambda}D^{\mu}$ was proposed by Igor Podlubny in 1997 [39], here λ and μ have non-integer values. Figure 1 shows the block diagram of fractional order PID controller.

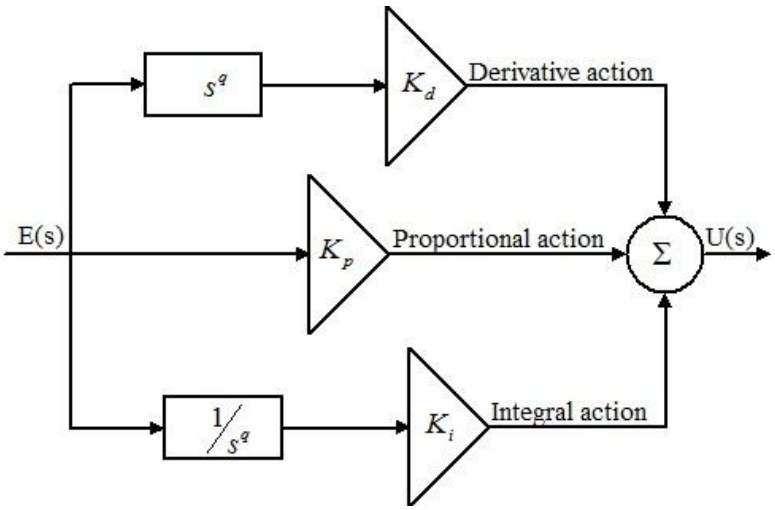

Figure 4.11 Fractional order PID controller

The transfer function for conventional PID controller is

$$G_{PID}(s) = \frac{u(s)}{e(s)} = K_c\left(1 + \frac{1}{\tau_i s} + \tau_d s\right) \tag{4.21}$$

Transfer function for fractional order PID controller is

$$G_{FOPID}(s) = \frac{u(s)}{e(s)} = K_c\left(1 + \frac{1}{\tau_i s^{\lambda}} + \tau_d s^{\mu}\right) \tag{4.22}$$

Where λ and μ are an arbitrary real numbers, K_p is amplification (gain), T_i is integration constant and T_d is differentiation constant. Taking λ=1 and μ=1, a classical PID controller is obtained. For further practical digital realization, the derivative part has to be complemented by first order filter. The filter is used to remove high frequency noise.

$$G_{FOPID}(s) = K_c\left(1 + \frac{1}{\tau_i s^{\lambda}} + \frac{\tau_d s^{\mu}}{\frac{\tau_d}{N} s + 1}\right) \tag{4.23}$$

The $PI^{\lambda} D^{\mu}$ controller is more flexible and gives an opportunity to better adjust the dynamics of control system. Its compact and simple but the analog realization of fractional order system is very difficult.

Intuitively, the FOPID has more degree of freedom than the conventional PID. It can be expected that the FOPID can provide better performance with proper choice of controller parameters. However, with more parameters to be tuned, the associated optimization problem will be more difficult to deal with. It is motivated to develop a systematic procedure for the FOPID optimization to achieve a certain performance.

The fractional order $PI^{\lambda}D^{\mu}$ generalizes the conventional integer order PID controller and expands it from point to plane. This expansion could provide much more flexibility in PID control design.

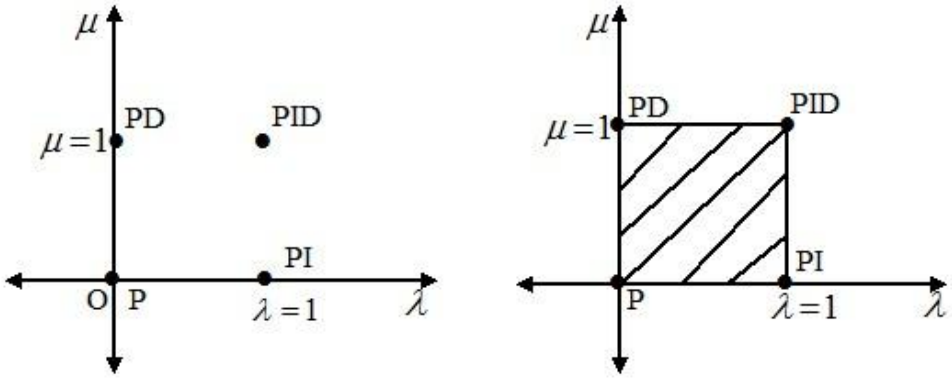

Figure 4.12 PID controller with fractional order

As shown in the Riemann-Liouville definition, fractional order systems have an infinite dimension. To realize fractional order controllers perfectly, all the past inputs should be memorized. Several proper approximation methods by finite differential or difference equation were proposed in recent researches, such as Sampling Time scaling, Short memory principle, Tustin Taylor Expansion, Lagrange function interpolation method. Because fractional order systems have an infinite dimension, the digital realization of FOPID keeps somewhat difficulty. The above FOPID controller can be approximated using different discrete methods, which is given by

$$G(z) = k_p + k_i w_i(z) + k_d w_d(z) \tag{4.24}$$

Where $w_i(z)$ is the discrete approximation equation of fractional order integral $s^{-\lambda}$, $w_d(z)$ is the discrete approximation equation of s^{μ}. The higher the order of

approximation equation, the closer the discrete model is approximates the real fractional order systems [40].

4.15.2 CRONE Controller

The CRONE control was proposed by Oustaloup in pursuing fractal robustness . CRONE is a French abbreviation for "Contrˆole Robuste d'Ordre Non Entier" (which means non-integer order robust control). In this subsection, we shall follow the basic concept of fractal robustness, which motivated the CRONE control, and then mainly focus on the second generation CRONE control scheme and its synthesis based on the desired frequency template which leads to fractional transmittance. Fractal robustness is used to describe the following two characteristics: the iso-damping and the vertical sliding form of frequency template in the Nichols chart. This desired robustness motivated the use of fractional-order controller in classical control systems to enhance their performance. With a unit negative feedback, for the characteristic equation given as

$$1+(\tau s)^{\alpha}=0 \tag{4.25}$$

the forward path transfer function, or the open-loop transmittance, is given as

$$\beta(s)=\left(\tfrac{1}{\tau s}\right)^{\alpha}=\left(\tfrac{\omega_u}{s}\right)^{\alpha} \tag{4.26}$$

This is the transmittance of a non integer integrator in which $\omega_u=\frac{1}{\tau_s}$ denotes the unit gain (or transitional frequency). In controller design, the objective is to achieve such a similar frequency behavior, in a medium frequency range around ω_u, knowing that the closed-loop dynamic behavior is exclusively linked to the open-loop behavior around ω_u .Synthesizing such a template defines the non-integer order approach that the second generation CRONE control uses [41].

4.15.2.1Application of CRONE Controller

It has many real life problem .some of these are following

- Car suspension control
- Flexible transmission
- Hydraulic actuator

4.15.3TID Controller

TID controller has the same structure as classical PID controller, but the proportional gain is replaced with a function $s^{-\alpha}$. The resulting transfer function of the entire compensator more closely approximates an optimal loop transfer function, thereby achieving improved feedback control performance. Further, as compared to conventional PID compensators, the TID compensator allows for simpler tuning, better disturbance rejection, and smaller effects of plant parameter variations on the closed-loop response. The objective of TID is to provide an improved feedback loop compensator having the advantages of the conventional PID compensator, but providing a response which is closer to the theoretically optimal response. In the TID patent an analog circuit using op-amps plus capacitors and resistors is introduced with a detailed component list which is useful in some cases where the computing power to implementing *T*3(*s*) digitally is not possible [42].

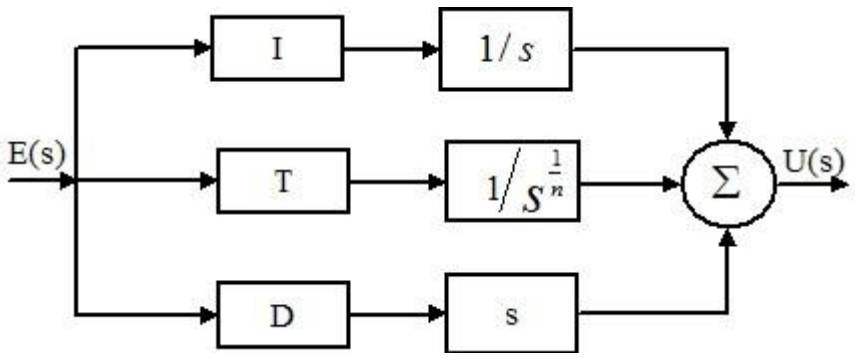

Figure 4.13 Fractional TID controllers

4.15.4 Fractional Lead-Leg Compensator

Fractional lead-lag controller is an extension of classical lead-lag controller [42]. The fractional lead – leg compensator is given by equation 4.27

$$c_r(s) = c_0\left(\frac{1+s/\omega_b}{1+s/\omega_n}\right) \tag{4.27}$$

Where $0 < \omega_b < \omega_h$, $C_0 > 0$ and $r \in (0,1)$.

4.16 Simulation Results

Here in the research work PID controller is used to control the speed of DC motor. A Ziegler-Nichols tuning method is used to design the PID controller. After tuning the PID controller, the values for control parameters Kp,Ki,and Kd are 0.5,0.98 and 0.0525 respectively.

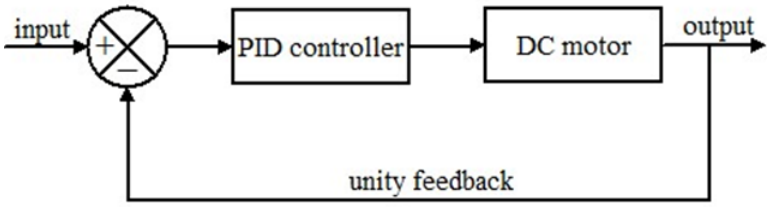

Figure 4.14 Block diagram of speed control of DC motor with PID controller

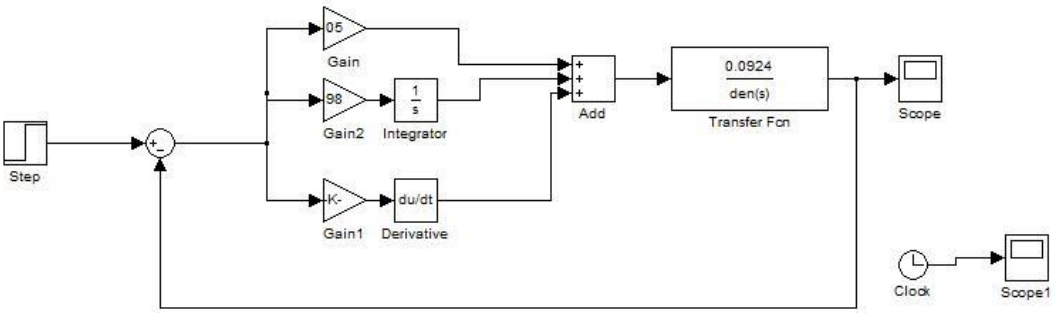

Figure 4.15 Simulink of PID controller with DC motor

Figure 4.15 shows the simulink model of PID controller for speed controller of DC motor.

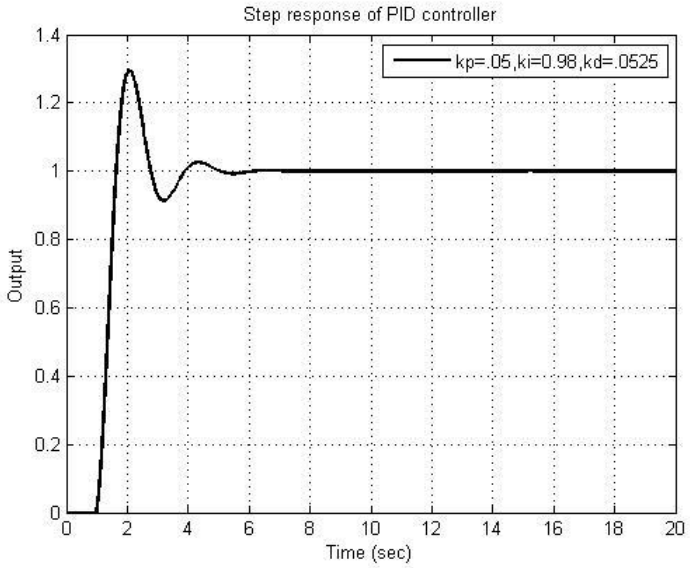

Figure 4.16 Unit step response of PID controller

Above Figure 4.16 shows the unit step response of PID controller for speed control of DC motor for the control parameters K_p=0.5,K_i=0.98 and K_d =0.0525. According to this figure peak overshoot (Mp) will be 29.5939 which is very far from ideal unit step Overshoot .This response also shows an unconsiderable error, settling time and peak time**.**

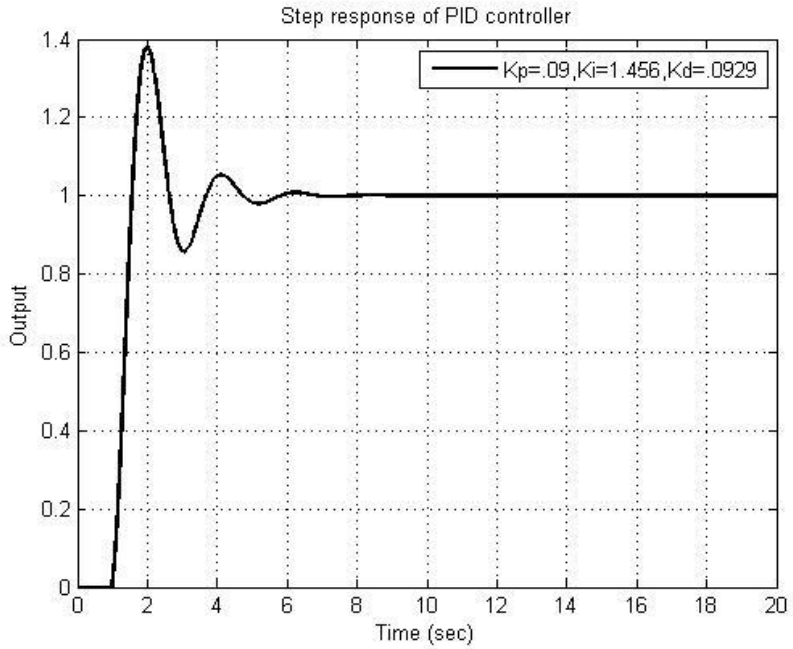

Figure 4.17 Unit step response of PID controller

Above Figure 4.17 shows the unit step response of PID controller for speed control of DC motor for the control parameters K_p=0.9,K_i=1.456 and K_d =0.0929. According to this figure peak overshoot (Mp) will be 37.9562 which is very far from ideal unit step Overshoot .This response also shows an unconsiderable error, settling time and peak time. it is not the desirable result **.**

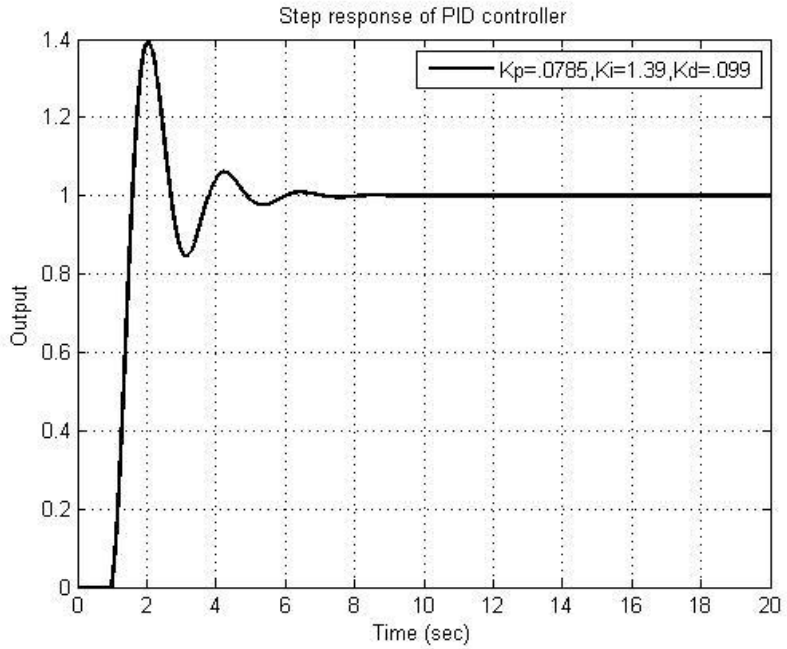

Figure 4.18 unit step response of PID controller

Above Figure 4.18 shows the unit step response of PID controller for speed control of DC motor for the control parameters K_p=0.0785,K_i=1.39 and K_d =0.099. According to this figure peak overshoot (Mp) will be 39.9546 which is very far from desirable unit step respnse .This response also shows an unconsiderable error, settling time and peak time . it is not the desirable result .

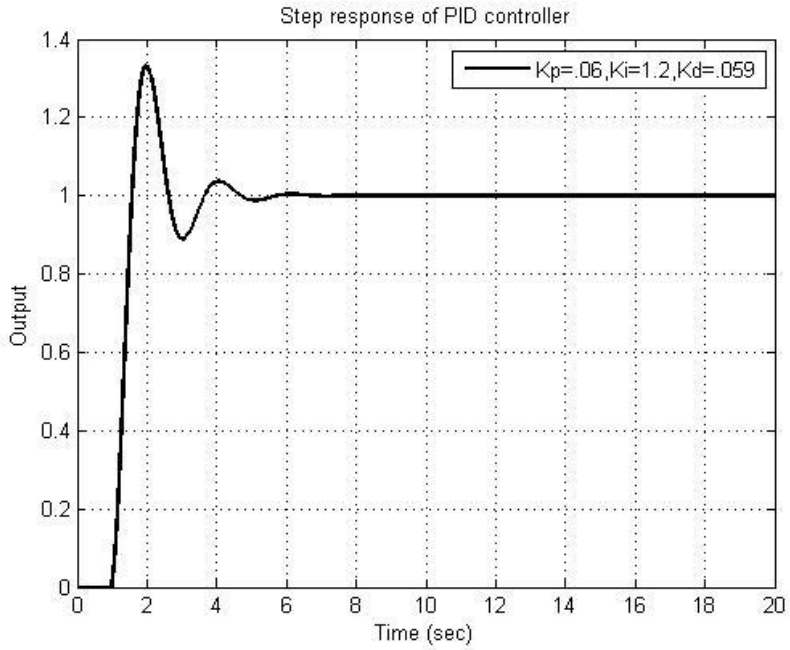

Figure 4.19 unit step response of PID controller

Above Figure 4.19 shows the unit step response of PID controller for speed control of DC motor for the control parameters K_p=0.06,K_i=1.2 and K_d =0.059. According to this figure peak overshoot (Mp) will be 33.3067 which is very far from desirable unit step response. This response also shows an unconsiderable error, settling time and peak time . it is not the desirable result.

For optimization of all these parameters such that peak overshoot, error, settling time and peak time we will use fractional order PID controller (FOPID). Because it also consider the fractional part of the process ,almost real life process are fractional order. Fractionality of process and controller improve the performance of the process and controller both. It will give the most robust and accurate results.

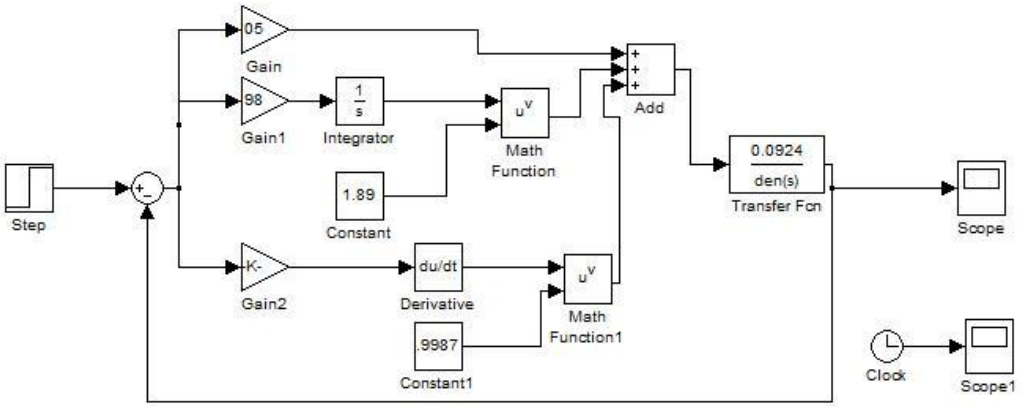

Figure 4.20 Simulink model for fractional controller PID controller

Above Figure 4.20 shows the FOPID controller for speed control of DC motor. According to this figure we have to optimize five parameters K_p, K_i, K_d, λ and μ . Here we will consider the previous optimize values of K_p ,K_i ,and K_d by 'Ziegler-Nichols tuning method' as for PID controller. Remaining two parameters are optimized by 'hit and trial' method in range [0 2]. Following results shows the step response for different values of λ and μ.

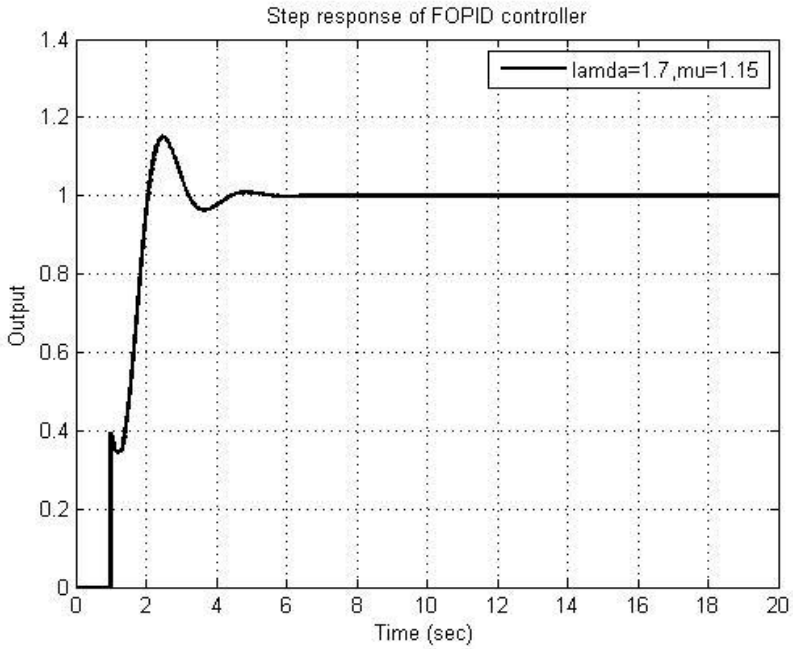

Figure 4.21 Step response of FOPID controller

Above Figure 4.21 shows the step response of FOPID controller for speed control of DC motor for the values of parameters λ=1.7 and μ =1.15. the reponse shows the less peak overshhot , error , settling time and peak time but it is also too far from desirable response**.**

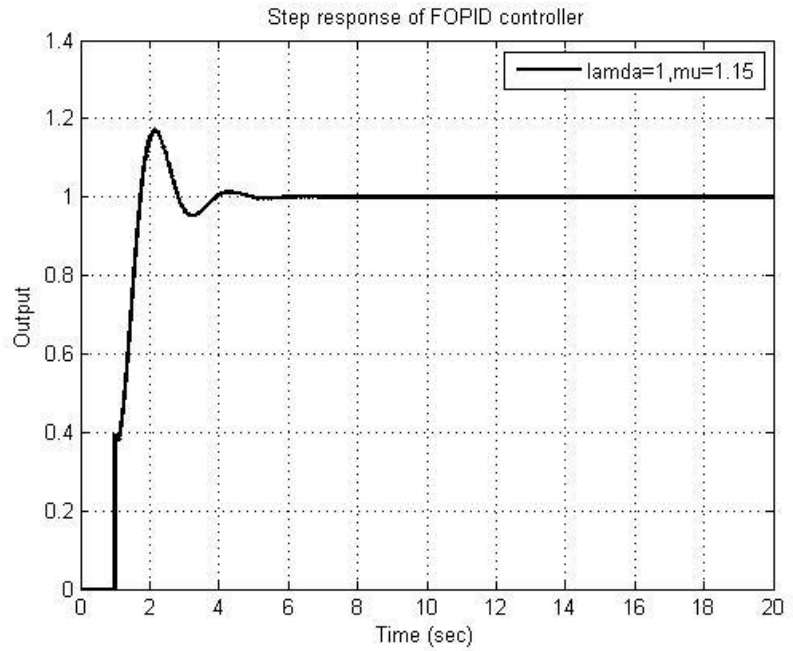

Figure 4.22 step response of FOPID controller

Above Figure 4.22 shows the step response of FOPID controller for speed control of DC motor for the values of parameters λ=1 and μ=1.15.here we are consider constant

λ=1 and variable μ > 1 . the reponse shows the less peak overshhot = 17.1389, error , settling time and peak time but it is also too far from desirable response.

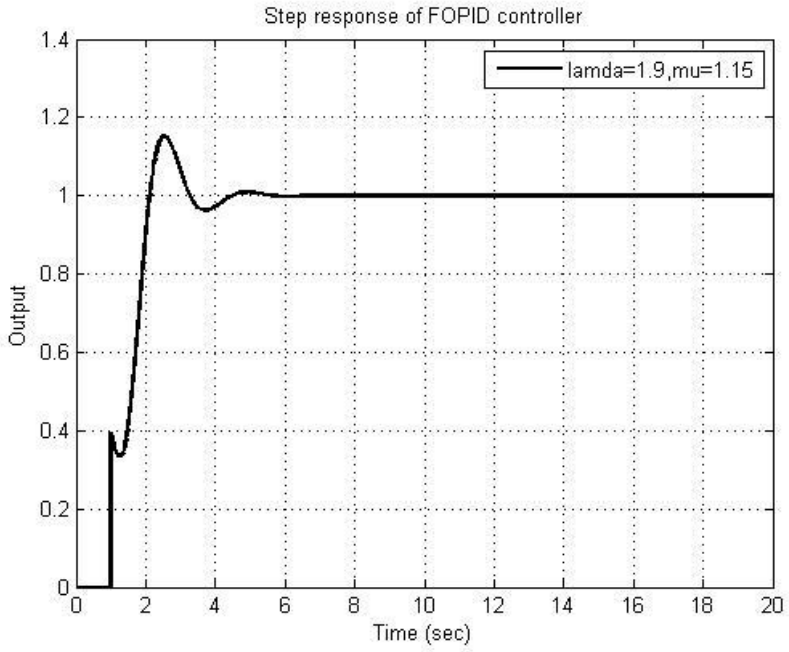

Figure 4.23 unit step response of FOPID controller

Above Figure 4.23 shows the step response of FOPID controller for speed control of DC motor for the values of parameters λ=1.9 and μ=1.15. Here we are consider variable λ > 1 and variable μ > 1. The reponse shows the less peak overshhot(15.3788) , error , settling time and peak time but it is also too far from desirable response**.**

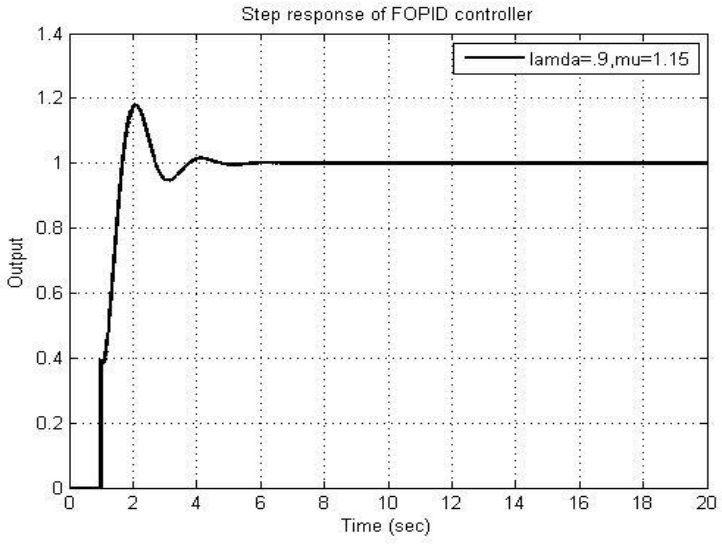

Figure 4.24 unit step response of FOPID controller

Above Figure 4.24 shows the step response of FOPID controller for speed control of DCmotor for the values of parameters λ=.9 and μ=1.15. Here we are considering λ< 1

and $\mu > 1$. the reponse shows the less peak overshhot (18.1406) , error settling time and peak time but it is also too far from desirable response.

CHAPTER 5
SOFT COMPUTING

5.1 Introduction

Soft computing, according to Prof. Zadeh, is "an emerging approach to computing, which parallels the remarkable ability of the human mind to reason and learn in an environment of uncertainty and imprecision" [43]. Soft computing includes neural network, fuzzy logic, evolutionary computation and probabilistic computing. Later soft computing is followed by chaos theory and learning theory. Earlier computational approaches can precisely analyze only relatively simple systems. More complex systems arising in biology, medicine, the humanities, management sciences and similar fields often remained intractable to conventional mathematical and analytical methods. That said, it should be pointed out that simplicity and complexity of systems are relative, and many conventional mathematical models have been both challenging and very productive. Soft computing is a new multidisciplinary field to construct new generation of artificial intelligence known as computational intelligence (CI).The main goal of soft computing to develop intelligent machine to provide solutions to real world problem which are not too difficult to model mathematically. Soft computing deals with imprecision, uncertainty, partial truth, and approximation to achieve practicability, robustness and low solution cost.

5.2 Classification of Soft Computing

Soft computing is broadly classified into following types [44].

- Neural network
- Fuzzy logic
- Evolutionary computation

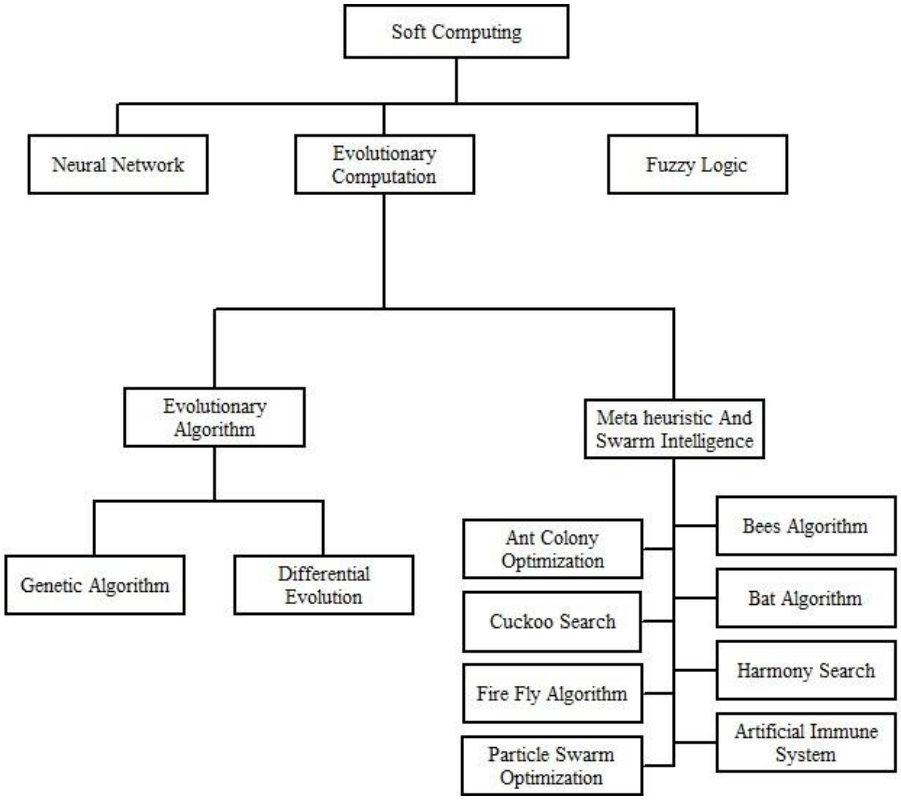

Figure 5.1 Classification of soft computing

5.2.1 Neural Network

Artificial neural nets (ANN) are electrical analogues of the biological neural nets.The electrical model of a typical biological neuron consists of a linear activator, followed by a non-linear inhibiting function. The linear activation function yields the sum of the weighted input excitation, while the non-linear inhibiting function attempts to arrest the signal levels of the sum. The resulting signal, produced by an electrical neuron, is thus bounded (amplitude limited). An artificial neural net is a collection of such electrical neurons connected in different topology. The most common application of an artificial neural net is in machine learning. In a learning problem, the weights and or non-linearities in an artificial neural net undergo an adaptation cycle. The adaptation cycle is required for updating these parameters of the network, until a state of equilibrium are reached, following which the parameters no longer change further. The ANN support both supervised and unsupervised types of machine learning. The supervised learning algorithms realized with ANN have been successfully applied in control, automation robotics and computer vision. The unsupervised learning algorithms built with ANN, on the other hand, have been applied in scheduling, knowledge acquisition, planning and analog to digital conversion of data.

5.2.2 Fuzzy Logic

Fuzzy logic is a form of many logic valued or probabilistic logic. It deals with reasoning that is approximate rather than fixed and exact. In contrast with traditional logic theory, where binary sets have two-valued logic true or false, fuzzy logic variables may have a truth value that ranges in degree between 0 and 1. Fuzzy logic has been extended to handle the concept of partial truth, where the truth value may range between completely true and completely false . Furthermore, when linguistic variables are used, these degrees may be managed by specific functions. Fuzzy logic began with the 1965 proposal of fuzzy set theory by Lotfi Zadeh .Fuzzy logic has been applied to many fields, from control theory to artificial intelligence.

5.2.3 Evolutionary Computation

Evolutionary computation (EC) uses iterative progress, such as growth or development in a population. This population is then selected in a guided random search using parallel processing to achieve the desired end. Such processes are often inspired by biological mechanisms of evolution. As evolution can produce highly optimized processes and networks.

Evolutionary computation further can be classified into following two categories.

- Evolutionary algorithm
- Meta heuristic and Swarm Intelligence

5.2.3.1 Evolutionary Algorithm

Evolutionary algorithms form a subset of evolutionary computation in that they generally only involve techniques implementing mechanisms inspired by biological evolution such as reproduction, mutation, recombination, natural selection and survival of the fittest. Candidate solutions to the optimization problem play the role of individuals in a population, and the cost function determines the environment within which the solutions "live". Evolution of the population then takes place after the repeated application of the above operators.

In this process, there are two main forces that form the basis of evolutionary systems: Recombination and mutation create the necessary diversity and thereby facilitate novelty, while selection acts as a force increasing quality. Many aspects of such an evolutionary process are stochastic. Changed pieces of information due to recombination and mutation are randomly chosen. On the other hand, selection operators can be either

deterministic, or stochastic. In the latter case, individuals with a higher fitness have a higher chance to be selected than individuals with a lower fitness, but typically even the weak individuals have a chance to become a parent or to survive.

5.2.3.2 Swarm Intelligence

Swarm Intelligence (SI) is an artificial intelligence technique based around on the study of collective behavior in decentralized, self-organized systems. The expression "swarm intelligence" was introduced by Beni & Wang in 1989, in the context of cellular robotic systems. SI systems are typically made up of a population of simple agents interacting locally with one another and also with their environment. Usually there is no centralized control structure dictating how the individual agents should behave, but local interactions between such agents often lead to the emergence of a global behavior. Examples of systems like this can be found in nature, including ant colonies, bird flocking, bee swarming, animal herding, bacteria molding and fish schooling [45].

Computational intelligence [46] (CI) is a successor of artificial intelligence relying on Evolutionary computation, which is a famous optimization technique. Computational intelligence combines elements of learning; adaptation and evolution to create programs that are, in some sense, intelligent. Computational intelligence research does not reject statistical methods, but often gives a complementary view.Computational intelligence finds its fundamental application in the area of fitness function design, methods for parameter control, and techniques for multimodal optimization. The importance of CI lies in the fact that these techniques often find optima in complicated optimization problems more quickly than the traditional optimization methods .Swarm intelligence further can be classified into following

- Ant colony optimization(ACO)
- Particle swarm optimization(PSO)
- Bees search
- Cuckoo search

Here in the research work we are dealing with particle swarm optimization (PSO)

5.3 Particle Swarm Optimization

The particle swarm optimization (PSO) was originally designed by Kennedy and Eberhart in 1995 is a new intelligent optimization algorithm developed in recent years which simulates the migration and aggregation of bird flock when they seek for food [47].PSO is similar to evolution algorithm. PSO algorithm adopts a strategy based on particle swarm and parallel global random search. PSO algorithm determines search path according to the velocity and current position of particle without more complicated evolution operation. PSO algorithm has better performance than early intelligent algorithms on calculation speed and memory occupation, and has less parameters and is easier to realize.

At present, PSO algorithm is attracting more and more attention of researchers, and has been widely used in fields like function optimization, combination optimization, neural network training, robot path programming, pattern recognition, fuzzy system control and so on [48]. In addition, The study of PSO algorithm also has infiltrated into electricity, communications, and economic fields. Like other random search algorithms, there is also certain degree of premature phenomenon in PSO algorithm. So in order to improve the optimization efficiency, many scholars did improvement researches on basic PSO algorithm, such as modifying PSO algorithm with inertia weight, modifying PSO algorithm with contraction factor, and combined algorithm with other intelligent algorithm. These modified algorithms have further improvement in aspects like calculation efficiency, convergence rate and so on.

5.4 Standard PSO Algorithm

PSO algorithm is swarm intelligence based evolutionary computation technique. The individual in swarm is a volume-less particle in multidimensional search space. The position in search space represents potential solution of optimization problem, and the flying velocity determines the direction and step of search. The particle flies in search space at definite velocity which is dynamically adjusted according to its own flying experience and its companions' flying experience, i.e., constantly adjusting its approach direction and velocity by tracing the best position found so far by particles themselves and that of the whole swarm, which forms positive feedback of swarm optimization.

Particle swarm tracks the two best current positions, moves to better region gradually, and finally arrives to the best position of the whole search space.

Particle Swarm has two primary operators: Velocity update and Position update. During each generation each particle is accelerated toward the particles previous best position and the global best position. At each iteration a new velocity value for each particle is calculated based on its current velocity, the distance from its previous best position, and the distance from the global best position. The new velocity value is then used to calculate the next position of the particle in the search space. This process is then iterated a set number of times,or until a minimum error is achieved [49].

Suppose in PSO, a swarm consists N number of particles moving around in a D dimensional search space .the ith particle denoted as

$$X_i = (x_{i1}, x_{i2}, \ldots\ldots x_{iD}) \tag{5.1}$$

Whose previous best solution (p_{best}) is represented as

$$P_i = (p_{i1}, p_{i2}, \ldots\ldots p_{iD}) \tag{5.2}$$

And the current velocity is described by

$$V_i = (v_{i1}, v_{i2}, \ldots\ldots v_{iD}) \tag{5.3}$$

Finally ,the best solution of whole swarm (g_{best}) also called global best is represented as

$$P_g = (p_{g1}, p_{g2}, \ldots\ldots p_{gD}) \tag{5.4}$$

At each time step, each particle moves towards (p_{best}) and (g_{best}) locations. The fitness function evaluates the performance of particles to determine whether the best fitting solution is achieved.

The detailed operation of particle swarm optimization is given below:

Step 1: Initialization The velocity and position of all particles are randomly set to within pre-defined ranges.

Step 2: Fitness Function Evaluate the fitness of each particle of the swarm.

Step 3: Velocity Updating At each iteration, the velocities of all particles are updated according to the following equation 5.5

$$v_{id} = v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \tag{5.5}$$

Where c_1 and c_2 are two positive constants, called cognitive learning rate and social learning rate respectively; *rand*() is a random function in the range [0,1]. The velocity of the particles are limited in [*Vmin,Vmax*]. Since the original formula of PSO lacks velocity control mechanism, it has a poor ability to search at a fine grain . A time decreasing inertia factor is designed by Eberhart and Shi to overcome this shortcoming in 1998. In PSO, a parameter called inertia weight is introduced in the original equation for balancing the global and local search. Than the velocity equations will be following as

$$v_{id} = w * v_{id} + c_1 * rand() * \left(p_{id} - x_{id} \right) + c_2 * rand() * \left(p_{gd} - x_{id} \right) \tag{5.6}$$

Where *w* is inertia factor which balances the global wide-range exploitation and the local nearby exploration abilities of the swarm.

Step 4: Position Updating Assuming a unit time interval between successive iterations, the positions of all particles are updated according to

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{5.7}$$

After updating, should be checked and limited to the allowed range.

Step 5: Memory Updating Update p_{best} and g_{best} then condition is met.

$$p_{best} = p_i \text{ if } f(p_i) > f(p_{best}) \tag{5.8}$$

$$g_{best} = p_g \text{ if } f(p_g) > f(g_{best}) \tag{5.9}$$

Where *f* (x) is the objective function subject to maximization.

Step 6: Stop Criteria The algorithm repeats Steps 2 to 5 until certain stop conditions are met, such as a pre-defined n

umber of iterations or a failure to make progress for a certain number of iterations. Once terminated, the algorithm reports the values of p_{best} and g_{best} as its solution.

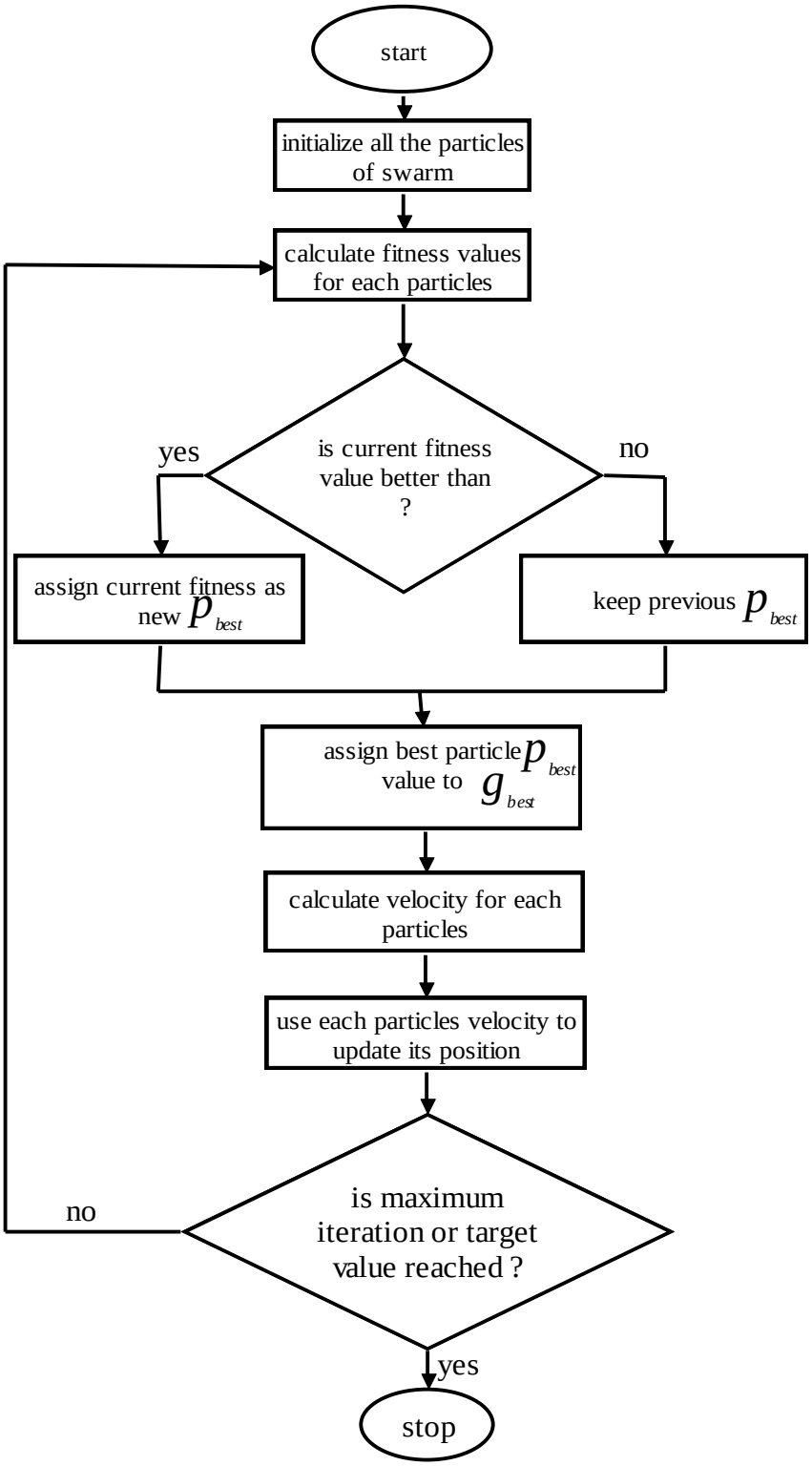

Figure 5.2 Flow chart of particle swarm algorithm

In PSO, each particles of swarm initialize having random values of velocity and position in a predefined range. Each particle has individual fitness function. Fitness value of each particle is calculated and is compared from previous fitness function value if the value of new fitness function is better than previous fitness function than assign current

fitness as new gbest otherwise keep previous pbest.if the new Pbest is available than consider it as gbest. Again calculate the value of velocity of each particles which is used to update the position of the particles. Now algorithm moves towards the termination criteria if minimum fitness function or maximum number of iteration is met than algorithm will terminate otherwise it again repeat the procedure.

5.5 Control Parameter of Particle Swarm Optimization (PSO)

PSO has following control factors [50].

- Acceleration constant and random function.
- Velocity of particles
- Inertia weight
- Swarm size
- Swarm communication topology

5.5.1Acceleration Constant and Random Number

Velocity updation equation has two positive acceleration constant c1 and c2 which is also called cognitive learning rate and social learning rate respectively. These two constant represent the weighting of stochasting acceleration that try to pull each particles towards the pbest and gbest positions. Thus , variations of these constant decreases the 'tension' in the system. A small values of these allow particles far from the target while higher values allow the particles to move abruptly towards the target region or its individual best.Velocity updation equation also has two random function ,rand() . These random function can vary in the range [0 1]. Random function takes the different values for each iteration of the algorithm. These variables affect the fitness function of the swarm.

5.5.2Velocity (Vmax and Vmin)

Particles' velocities on each dimension are clamped to a maximum velocity *Vmax.* If the sum of accelerations would cause the velocity on that dimension to exceed *Vmax,* which is a parameter specified by the user, then the velocity on that dimension is limited to *Vmax.*

Vmax is therefore an important parameter. It determines the resolution, or fineness, with which regions between the present position and the target (best so far) position are searched. If *Vmax* is too high, particles might fly past good solutions. If *Vmax* is too small, on the other hand, particles may not explore sufficiently beyond locally good

regions. In fact, they could become trapped in local optima, unable to move far enough to reach a better position in the problem space.

5.5.3 Inertia Weight

A few years after a new parameter was introduced to balance between global exploration and local exploitation while avoiding clamping the particle velocity through the *Vmax* method.Which was viewed as both artificial and difficult to balance. A single value for *Vmax* was not necessarily applicable to all problem spaces - very large spaces required larger values to ensure adequate exploration, while smaller spaces required very small values to prevent explosion-like behavior on their scale. Finding the appropriate value for *Vmax* for the problem being solved was critical, as a poorly-chosen *Vmax* could result in extremely poor performance, yet there was no simple, reliable method for choosing this value beyond trial and error. This new *inertia weight* parameter *w* was designed to replace *Vmax* by adjusting the influence of the previous particle velocities on the optimization process. By adjusting the value of *w*, the swarm has a greater tendency to eventually constrict itself down to the area containing the best fitness and explore that area in detail. This control of the "inertia" of particles, similar to the role of friction in a physical setting, resulted in improved performance of the algorithm and removed the need for velocity limiting. Inertia factor can be given as

$$w = \left(w_{\max} - w_{\min}\right) \times \frac{\left(Iter_{\max} - Iter_{now}\right)}{Iter_{\max}} + w_{\min} \tag{5.10}$$

5.5.4 Swarm Size

The number of particles in the swarm affect the run time of PSO algorithm. The swarm size can influence the resulting performance by a varying amount , depending upon the being optimized. Some test model give the improved performance as the size of the swarm is increased, while others tend to be better optimized by smaller swarms. Thus there is no definitive value for the swarm size that is optimal across all problems, so to avoid tuning the algorithm to each specific problem, a compromise must be reached.

5.5.5 Swarm Communication Topology

Generally we use two type of communication topology

- Local Topology
- Global Topology

The *lbest* swarm model is referred as a *local topology*. Local topology is not widely used in the community up to this point because it gives the inferior performance when compared to the global gbest model or global topology using the original PSO algorithm. But according to recent research lbest swarms gives the improved results across many standard problem sets when used in conjunction with other improvements to the algorithm.

The lbest model is perhaps the simplest form of a local topology. Which is known as the *ring* model, shown in figure 5.3(b). The lbest ring model connects each particle to only two other particles in the swarm, on the other hand the gbest model connects every particle with each other. In gbest model figure 5.3(a), each particle can convey the information with each other which make able to obtain information from the very best particle in the entire swarm population.

The inclusion of the local ring topology as part of a standard algorithm for particle swarm optimization comes with a caveat, however. Given the slower convergence of the lbest model, more function evaluations are required for the improved performance to be seen. This is especially important on unimodal functions, where the fast convergence of the gbest model combined with single minima in the feasible search Space results in quicker performance than that of the lbest swarm with its limited Communication.

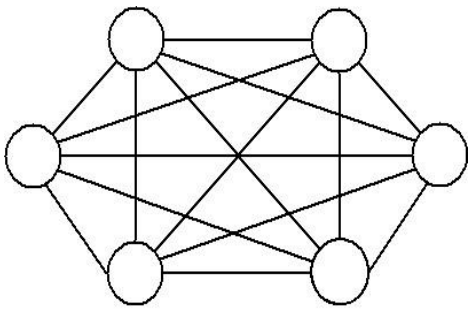

Figure 5.3(a) Global best model

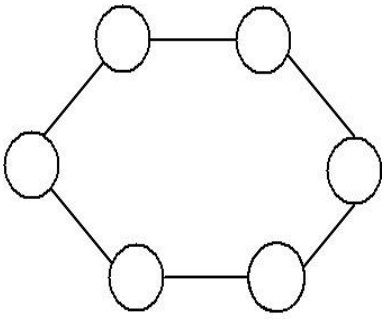

Figure 5.3(b) Local best model

5.6 Advantage of PSO

- PSO is based on the intelligence. It can be applied into both scientific research and engineering use.
- PSO has no overlapping and mutation calculation. The search can be carried out by the speed of the particle. During the development of several generations, only

the most optimist particle can transmit information onto the other particles, and the speed of the researching is very fast.

- The calculation in PSO is very simple. Compared with the other developing calculations, it occupies the bigger optimization ability and it can be completed easily.
- PSO adopts the real number code, and it is decided directly by the solution. The number of the dimension is equal to the constant of the solution [51].

5.7 Disadvantage of PSO

- The method easily suffers from the partial optimism, which causes the less exact at the regulation of its speed and the direction.
- The method cannot work out the problems of scattering and optimization (Chen Yonggang,Yang Fengjie,Sun Jigui ,2006,(In Chinese).
- The method cannot work out the problems of non-coordinate system, such as the solution to the energy field and the moving rules of the particles in the energy field [51].

5.8 Application of PSO

Particle swarm optimization can be used across a wide range of applications. It can be used for the multimodal problem and problem for which there is no specialized method available or all specialized method gives unsatisfactory results. Some of applications are described following [52].

- The first application represents an approach, or method, that can be used for many applications: evolving artificial neural networks. Particle swarm optimization is being used to evolve not only the network weights, but also the network structure (Eberhart and Shi 1998a, Kennedy, Eberhart and Shi 2001). The method is so simple and efficient that we have almost completely ceased using traditional neural network training paradigms such as backpropagation. Instead, we evolve our networks using particle swarms. The approach is effective for any network architecture.
- Another application is the use of particle swarm optimization for reactive power and voltage control by a Japanese electric utility (Yoshida et al., 1999). Here,

particle swarm optimization was used to determine a control strategy with continuous and discrete control variables, resulting in a sort of hybrid binary and real-valued version of the algorithm. Voltage stability in the system was achieved using a continuation power flow technique.

- Finally, one of the most exciting applications of PSO **is** that by a major American corporation to ingredient mix optimization. In this work, "ingredient mix" refers to the mixture of ingredients that are used to grow production strains of microorganisms that naturally secrete or manufacture something of interest. Here, PSO was used in parallel with traditional industrial optimization methods.

Generally speaking, particle swarm optimization, like the other evolutionary computation algorithms, can be applied to solve most optimization problems and problems that can be converted to optimization problems. Among the application areas with the most potential are system design, multi-objective optimization, classification, pattern recognition, biological system modeling, scheduling (planning), signal processing, games, robotic applications, decision making, simulation and identification. Examples include fuzzy controller design, job shop scheduling, real time robot path planing, image segmentation, EEG signal simulation, speaker verification, time-frequency analysis, modeling of the spread of antibiotic resistance, burn diagnosing, gesture recognition and automatic target detection, to name a few.

5.9 Optimization of Fractional Order PID Controller Using PSO

Here in the research work particle swarm optimization is used to optimize the control parameter of FOPID controller to control the speed of DC motor in a best way. Optimization of FOPID controller firstly need the design the optimization goal and then the optimization to be searched. PSO algorithm runs until the stop condition is satisfied. The particles of last generation are the optimized parameters of FOPID controller.

5.9.1Representation of Parameter

FOPID controller has five parameters K_P, K_i, K_d, λ and μ are required to be designed, according to control objectives. For the conventional PID controller design, we should ensure that all the poles of the close-loop transfer function are confined in the left half of the s plane. Based on the stability condition of fractional order system, a controller that stabilizes the integer order system stabilizes the fractional order system. The FOPID

controller parameter can adopt the parameter of integer order controller and add fractional orders of integral and derivative. The initial positions of the *i*th particles of the swarm can be represented by a 5 dimensional vector, and then the initial values are randomly generated based on extreme values.

5.9.2 Selection of PSO Factors

According to PSO algorithm, PSO needs to predefine numerical coefficients. PSO consists acceleration constant c1 and c2, random function rand(), and the maximum and minimum velocity [Vmin Vmax] ,inertia weight ,size of swarm and finally communication topology. In my research work , acceleration constant c1=c2=1.5 , random function will be in the range , rand() =[0 1] and the inertia weight ω=.9, and the initial population of swarm will be n=100.

5.9.3 Fitness Function

For our case of design, we had to tune all the five parameters of FOPID such that it gives the best output results or in other words we have to optimize all the parameters of the FOPID for best results. Here we define a five dimensional search space in which all the five dimensions represent five different parameters of the FOPID. Each particular point in the search space represents a particular combination of [K_p K_i K_d , λ ,$\boldsymbol{\mu}$] for which a particular response is obtained.

The performance of the point or the combination of FOPID parameters is determined by a fitness function or the cost function. For the case of our design, we have taken four component functions to define fitness function. The fitness function is a function of steady state error, peak overshoot, rise time and settling time. However the contribution of these component functions towards the original fitness function is determined by a scale factor that depends upon the choice of the designer. For this design the best point is the point where the fitness function has the minimal value. The chosen fitness function is given as by equation 5.11

$$F = (1-\exp(-\beta))\,(Mp + ESS) + (\exp(-\beta))(Ts - Tr) \qquad (5.11)$$

Where F: Fitness function

Mp: Peak Overshoot

Ts: Settling Time

Tr: Rise Time

β: Scaling Factor(Depends upon the choice of designer) For our case of design we have taken the scaling factor β = 1.

In the matlab library we have defined a fitness function. It has the format

Function [F] = fitness (K_d K_P K_i,λ,μ)

Which has FOPID parameters as input values and it returns the fitness value of the FOPID based controlled model as its output.

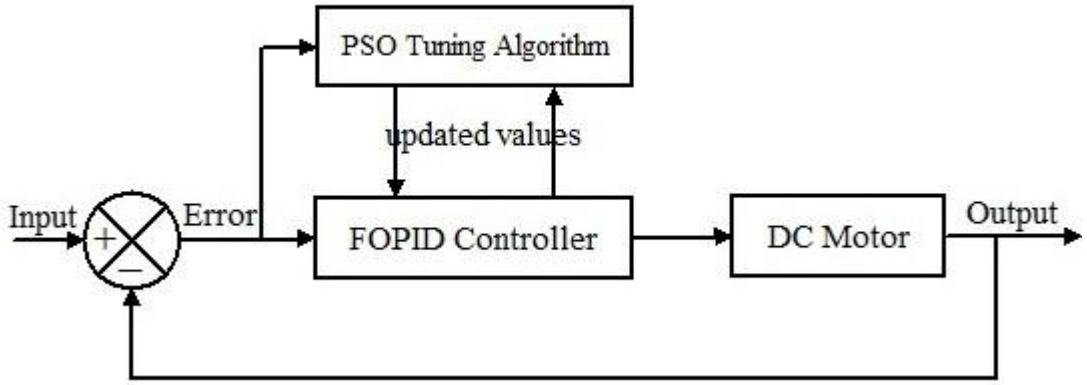

Figure 5.4 PSO tuned FOPID controller

Figure 5.4 shows the PSO tuned FOPID controller for the purpose of speed control of DC motor. this will give the best optimization result for the control parameter of fractional order controller which are needed to tune the controller.

5.10 Simulation Results

In my research work, we are dealing with fractional order PID controller which controls the speed of DC motor. Here fractional order PID controller is tuned by PSO which gives the different fitness function for each iteration because it uses the random function. Here we are using M=1000 number of iterations to optimize parameters control. It is also the termination condition of PSO algorithm. Here we are using 'Tustin method' to obtain the transfer function of FOPID controller by discretization. Here in the research work we are using sampling period = .01 and the range of λ and μ in between [0 2]. M=number of iterations, figure 5.5 shows the simulink mode of FOPID controller.

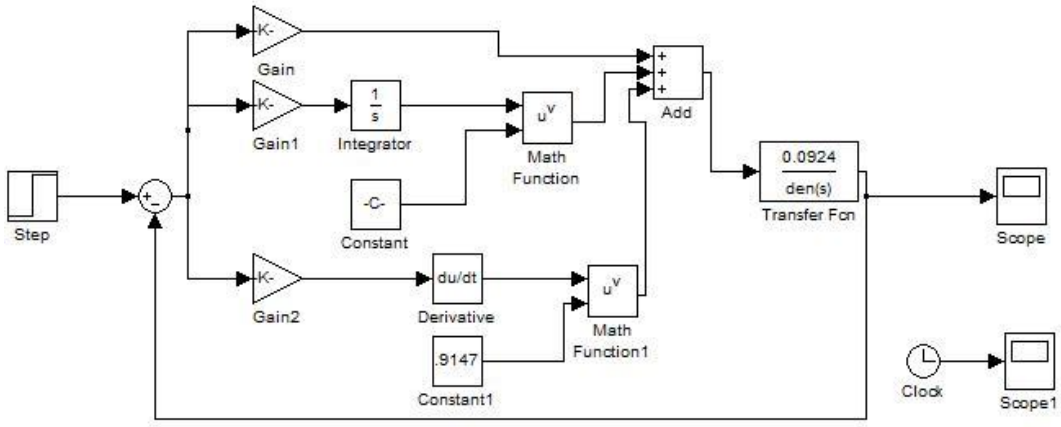

Figure 5.5 Simulink model of PSO based FOPID controller

This simulink model shows the fractional order PID controller (FOPID) for speed control of DC motor. This simulink model gives the step response of FOPID controller for the different values of all five parameters of FOPID which are required to design for optimize step response.

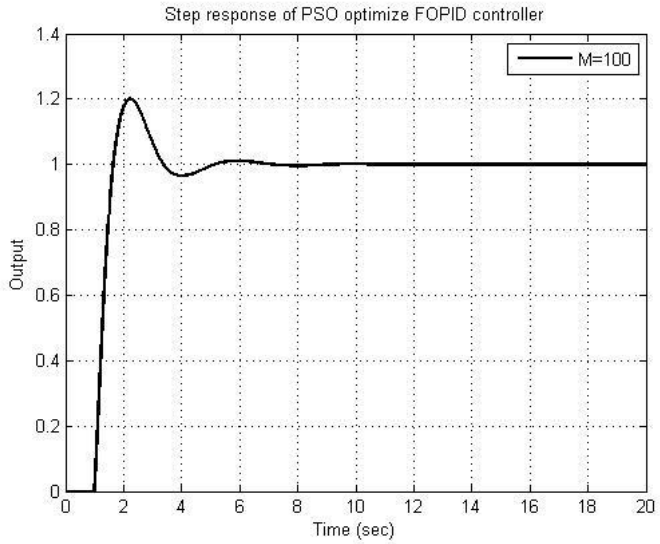

Figure 5.6 Unit step response of FOPID controller

Figure 5.6 shows the step response of PSO tuned FOPID controller for the M=100 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 0.3540, 1.3869, 0.0600 ,0.9374 and 0.5584 respectively. Step response shows very high peak overshoot (20.764%) , settling time (2.2248 sec) ,error and peak time(2.2248) that is not desirable for the purpose of industrial use. This controller is designed for fitness function(Fbest =.2001) and elapsed time (6.27593) sec.

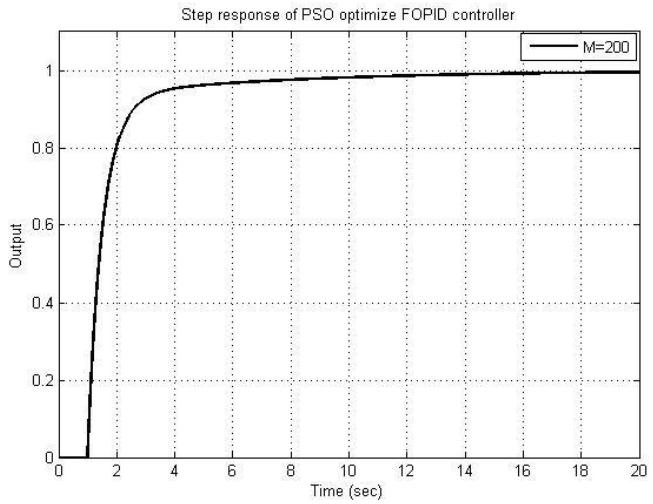

Figure 5.7 Unit step response of FOPID controller

Figure 5.7 shows the step response of PSO tuned FOPID controller for the M=200 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 1.1038, .1403, .5491 ,0.8378 and 0.9561 respectively. Step response shows acceptable peak overshoot (-0.4735%) but settling time (19.6100 sec) and peak time (10.8775) are very high that is not desirable for the purpose of industrial use. This controller is designed for fitness function(Fbest =.1989) and elapsed time (18.043227) sec.

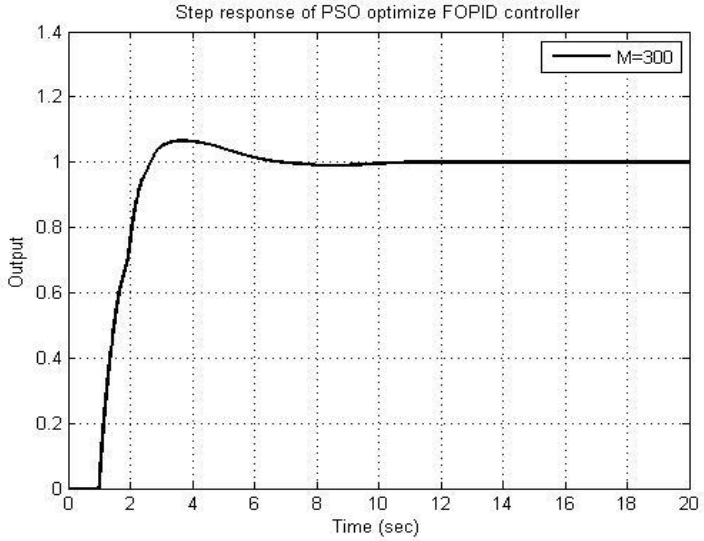

Figure 5.8 Unit step response of FOPID controller

Figure 5.8 shows the step response of PSO tuned FOPID controller for the M=300 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 0.3674, .4360, 0.0385 ,0.9843 and 0.4413 respectively. Step response shows high peak

overshoot (Mp=6.506%) , settling time (Ts=9.8633 sec) ,error and peak time(Tp=3.6897). it is not desirable response . This controller is designed for fitness function(Fbest =.1998) and elapsed time (21.012049) sec.

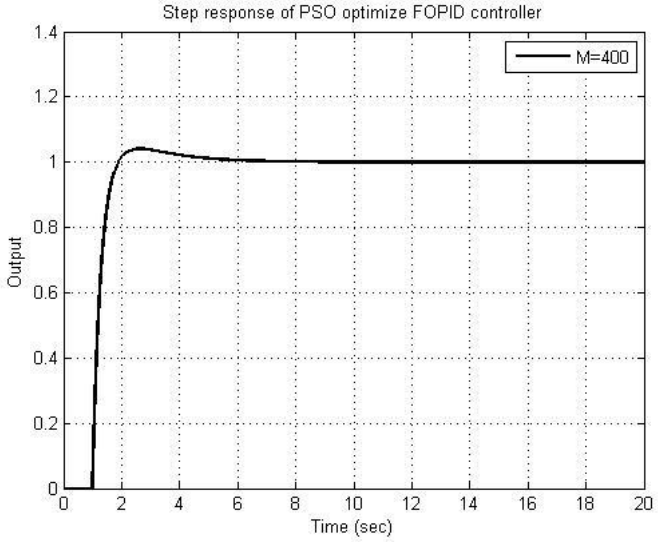

Figure 5.9 Unit step response of FOPID controller

Figure 5.9 shows the step response of PSO tuned FOPID controller for the M=400 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p), integral gain (K_i), differential gain (K_d), λ and μ are 1.5487, .7366, 0.3934 ,0.7595 and 0.9387 respectively. Step response shows high peak overshoot (Mp=4.0813%), settling time (Ts=6.2218 sec) and peak time(Tp=2.6326). it is moving towards desirable response, but is not optimal result . This controller is designed for fitness function (Fbest =.1981) and elapsed time (24.907939) sec.

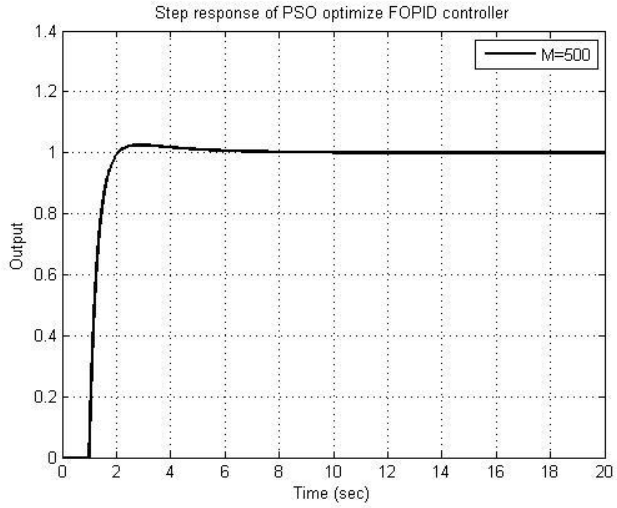

Figure 5.10 Unit step response of FOPID controller

Figure 5.10 shows the step response of PSO tuned FOPID controller for the M=500 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 1.136, .3008, 0..2759 ,0.5846 and 0.9130 respectively. Step response shows peak overshoot (Mp=2.5445%) , settling time (Ts=6.5589 sec) and peak time(Tp=2.8334). it is moving towards desirable response, but is not optimal result . This controller is designed for fitness function(Fbest =.1975) and elapsed time (67.759827) sec.

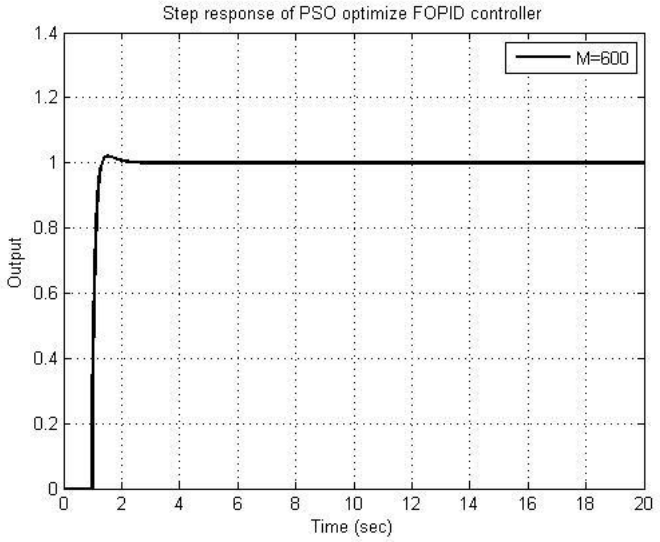

Figure 5.11 Unit step response of FOPID controller

Figure 5.11 shows the step response of PSO tuned FOPID controller for the M=600 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 0.4907, 0.5887, 0.0042 ,0.4927 and 1.205 respectively. Step response shows a fine peak overshoot (Mp=2.1015%) , settling time (Ts=2.1015 sec) and peak time(Tp=1.5125). it is moving towards desirable response, it is a good response . This controller is designed for fitness function(Fbest =.1970) and elapsed time (47.710881) sec.

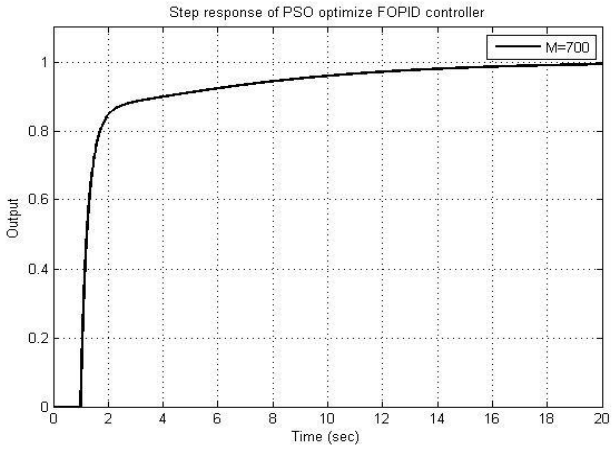

Figure 5.12 Unit step response of FOPID controller

Figure 5.12 shows the step response of PSO tuned FOPID controller for the M=700 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 1.0828, .2591, 0.2577 ,1.4594. and 0.9069 respectively. Step response shows negative overshoot (Mp= -.6846%) , settling time (Ts=20 sec) and peak time(Tp=20). This reponse gives better overshoot but the settling time and peal time are too high . This is designed for fitness function(Fbest =.1968) and elapsed time (60.679252) sec.

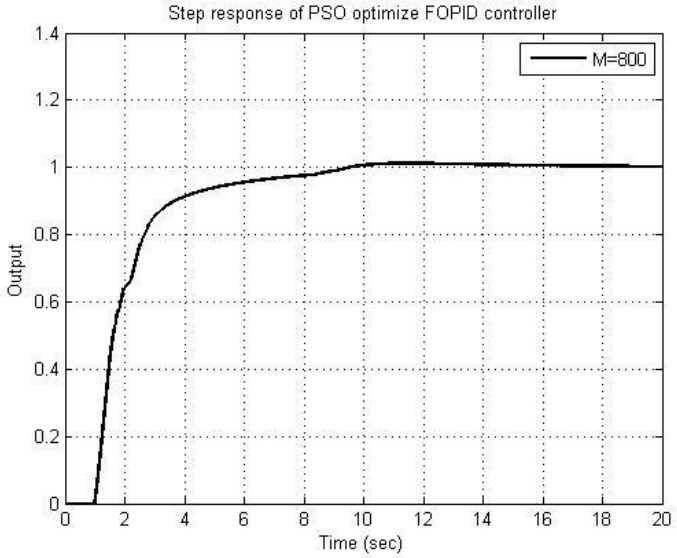

Figure 5.13 Unit step response of FOPID controller

Figure 5.13 shows the step response of PSO tuned FOPID controller for the M=800 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d), λ and μ are 0.9519, .1986, 1.0390 ,0.8956 and 0.2674 respectively. Step response shows considerable

peak overshoot (Mp=1.1869%) but settling time (Ts=16.4917 sec) and peak time(Tp=11.4874) are too high. This controller is designed for fitness function (Fbest =.1973) and elapsed time (60.534406) sec.

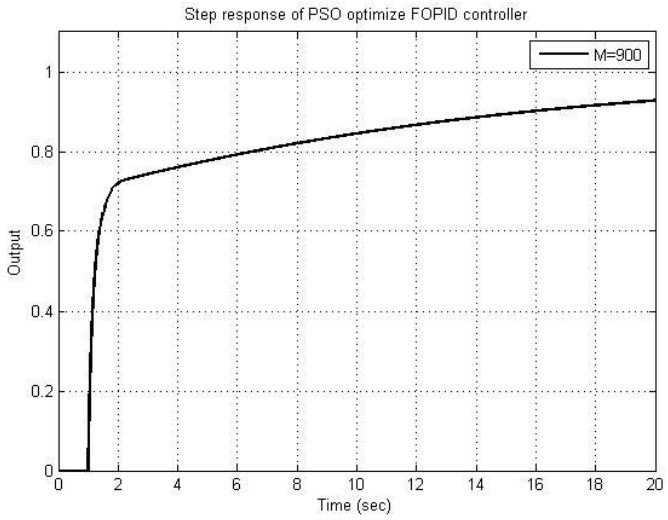

Figure 5.14 Unit step response of FOPID controller

Figure 5.14 shows the step response of PSO tuned FOPID controller for the M=900 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 0.5715, .0582, ,0.0313, 1.0559 and 0.05579 respectively. Step responseshows very high peak overshoot (Mp= -7.220%) and settling time (Ts=29 sec) and peak time(Tp=19.99) .This controller is designed for fitness function(Fbest =.1969) and elapsed time (63.134011) sec**.**

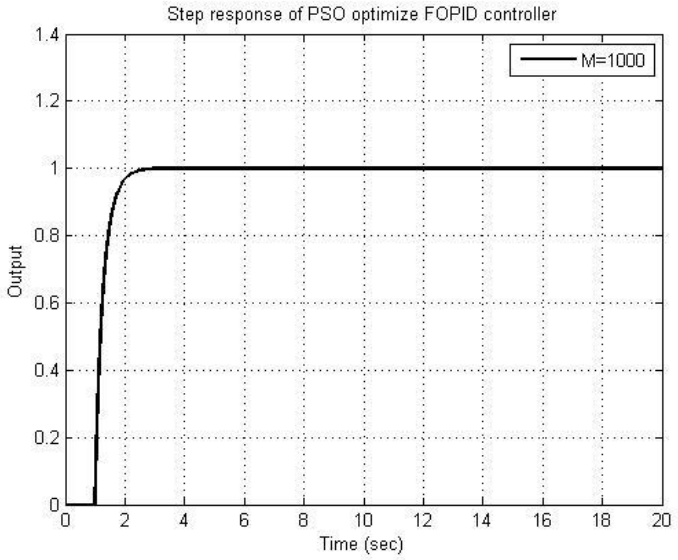

Figure 5.15 Unit step response of FOPID controller

Figure 5.15 shows the step response of PSO tuned FOPID controller for the M=1000 number of iterations. According to PSO algorithm control parameters of FOPID controller proportional gain (K_p) , integral gain (K_i) , differential gain (K_d) , λ and μ are 1.1082, .7882, .2259 ,1.0940 and 0.9147 respectively. Step response shows desirable peak overshoot (Mp=.0410%) and settling time (Ts=2.5358 sec) and peak time(Tp=3.8196) .This controller is designed for fitness function(Fbest =.1965) and elapsed time (76.102968) sec.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Step Response Analysis of PID Controller

Here in the research work we will compare peak overshoot , errors , settling time and peak time to find the best step response for speed control of DC motor. Figure 6.1 shows the simulink model of PID controller with error block diagram

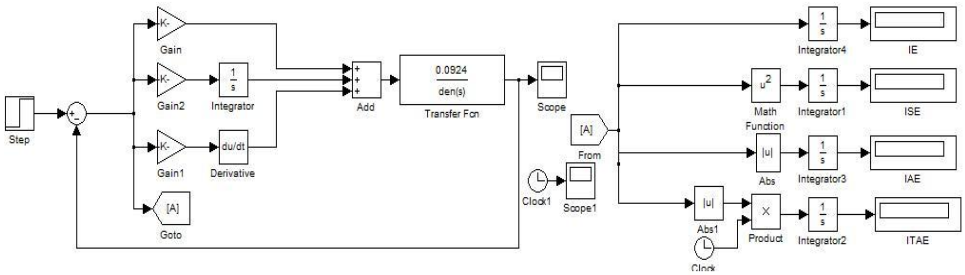

Figure 6.1 Simulink of PID controller with error

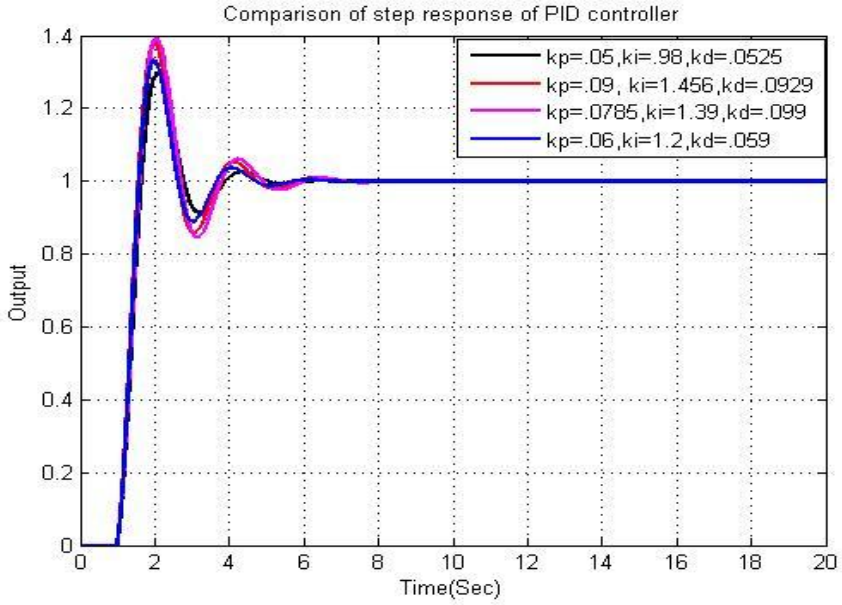

Figure 6.2 Comparison of step response of PID controller for different values of K_p,K_i, and K_d

The above Figure 6.2 shows the unit step response of PID controller for different values of parameters K_p, K_i, K_d for the purpose to find best unit step response. A detailed comparative study can be done by using table 1 which is given below.

Table 6.1 Comparison of different Parameters for PID controller

K_p	K_i	K_d	M_p	T_p	T_s	IE	ISE	IAE	ITAE
0.05	0.98	0.0525	29.5939	1.0736	5.4706	.1109	0.272	0.628	1.181
0.09	1.456	0.0929	37.9562	1.9937	6.5158	.1286	0.3075	0.7088	1.456
0.785	1.39	0.099	39.3546	2.0241	6.7980	0.1346	0.3341	0.7708	1.653
0.06	1.2	0.059	33.3067	1.9835	5.5025	0.1559	0.2893	0.6335	1.191

From the above table 6.1, we can say that for the values of proportional gain K_p(0.05), integral gain K_i(0.98) and derivative gain K_d(0.0525) all the control parameters peak overshoot(M_p=29.5939), peak time (Tp=1.0736 sec) , settling time (Ts=5.4706) and all the errors are minimum as compare to other proportiaonal gain, integral gain and derivative gain. Hence combination of these values is taken for PID controller.

6.2 Step Response Analysis of FOPID Controller

Here in the research work we will compare all the performances parameters of the FOPID controller by varying the values of 'λ and μ' and consider the best values of K_p ,K_i, and K_d For which PID has best unit step response. Figure 6.3 shows the simulink diagram of FOPID controller with error.

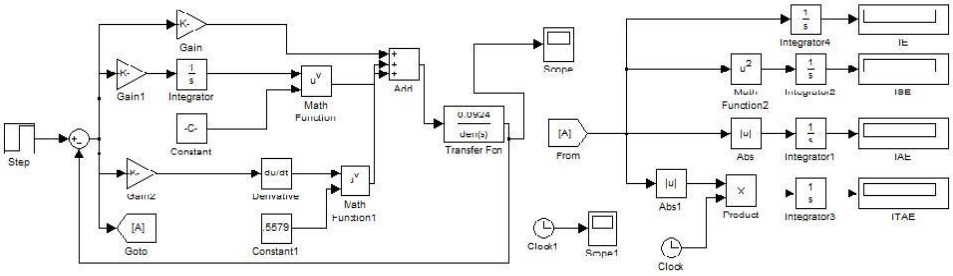

Figure 6.3 Simulink of FOPID controller with error

The above Figure 6.3 is the simulink block of DC motor control using fractional PID controller . In a fractional PID controller, for different values of integral order (λ) and derivative order (μ), different unit step responses come out. In the next section, different combinations of integral order and derivative order are taken and unit step responses and different control parameters are calculated.

(1) With λ=1 and Varying Values of μ<1

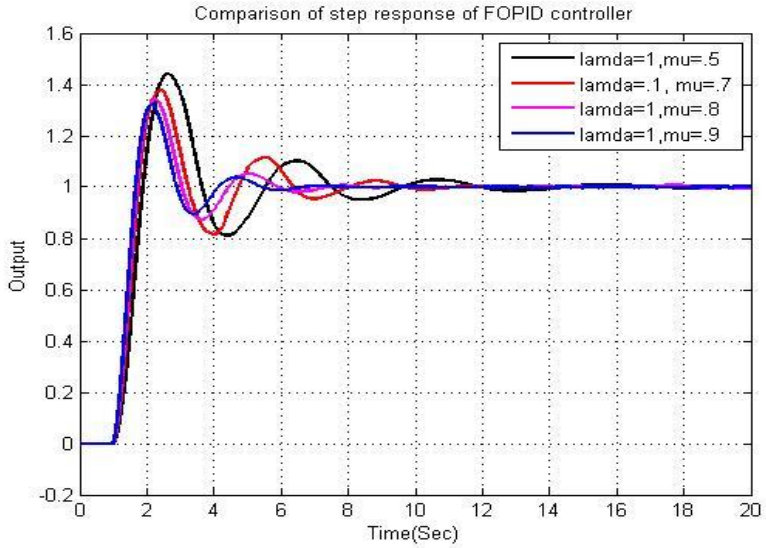

Figure 6.4 Unit step response of speed control of DC motor using FOPID for Different Values of μ<1

The above Figure 6.4 shows the unit step response of speed control of DC motor using fractional order PID controller for λ=1 and μ<1.

Table 6.2 Comparison of Parameters for Different Combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
1	.5	44.0002	2.6109	20	.1548	.6693	2.072	13.2
1	.7	37.9076	2.4184	20	.1909	.4894	1.504	7.794
1	.8	34.2127	2.2361	19.2436	.1938	.3864	.8966	2.351
1	.9	31.9307	2.1435	6.4481	.1909	.3397	.7420	1.53

It can be seen from the above table 6.2 that with the increase in the value of μ, control parameters are minimized. The best unit step response comes out for λ=1 and μ =.9

(2) With varying values of λ<1 and μ=1

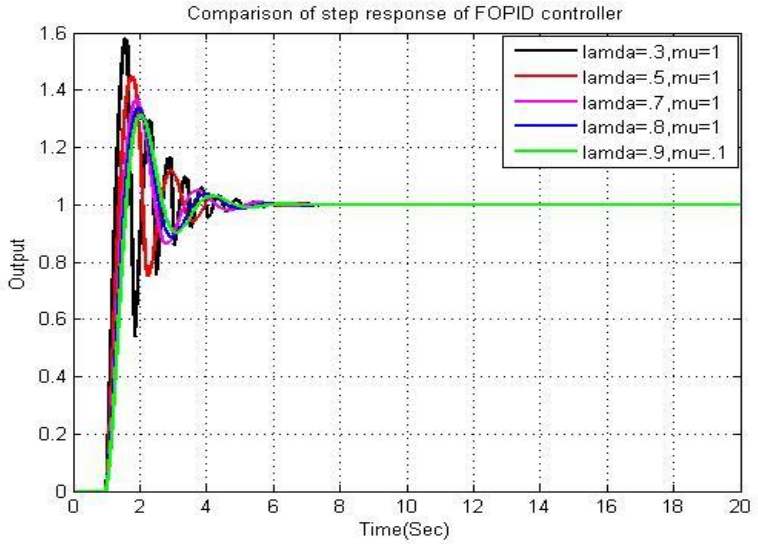

Figure 6.5 Unit step response of DC moror control using FOPID for different Values of λ <1

The above Figure 6.5 shows the unit step response of speed control of DC motor using fractional order PID controller for λ<1 and μ=1.

Table 6.3 Comparison of Parameters for Different Combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
.3	1	57.8344	1.578	6.0182	.003823	.2274	.61	1.235
.5	1	44.9034	1.7552	5.7181	.03573	.2361	.5899	1.133
.7	1	36.6742	1.9012	5.7881	.0931	.2569	.6036	1.147
.8	1	33.7712	1.9640	5.3927	.1256	.2707	.6131	1.154
.9	1	31.4517	2.0222	5.6193	.1585	.2859	.6247	1.166

It can be seen from the above table 6.3 that with the increase in the value of λ, peak overshoot decreases and all other control parameters changes minutely.

(3) With varying values of λ<1 and μ<1

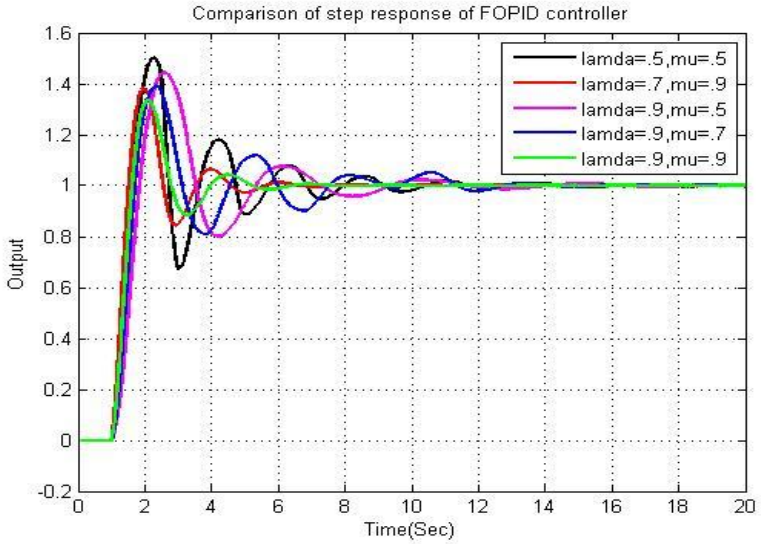

Figure 6.6 Unit step response of DC motor control using FOPID for Different Values of λ<1and μ<1

The above Figure 6.6 shows the unit step response of speed control of DC motor using fractional order PID controller for λ<1 and μ<1.

Table 6.4 Comparison of Parameters for Different Combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
.5	.5	50.4591	2.2573	20	.02828	.4779	1.421	6.702
.5	.7	47.6677	2.0182	19.9729	.0198	.3849	1.445	8.026
.5	.9	45.9049	1.8137	6.9300	.03573	.2656	.678	1.431
.7	.5	46.4460	2.4501	20	.06876	.5575	1.639	8.389
.7	.9	38.3353	1.7679	6.4501	.02313	.2106	.6006	1.421
.9	.5	44.1697	2.5774	20	.1269	.6057	1.704	8.775
.9	.7	38.9747	2.3607	20	.1231	.4783	1.52	8.147
.9	.9	38.5954	2.0397	7.1159	.1585	.3223	.7257	1.501

It can be seen from the above table 6.4 that from all the different combinations of λ and μ, control parameters for the values of λ=0.7 and μ=0.9 are less than other values of λ and μ.

(4) With λ=1 and varying values of μ>1

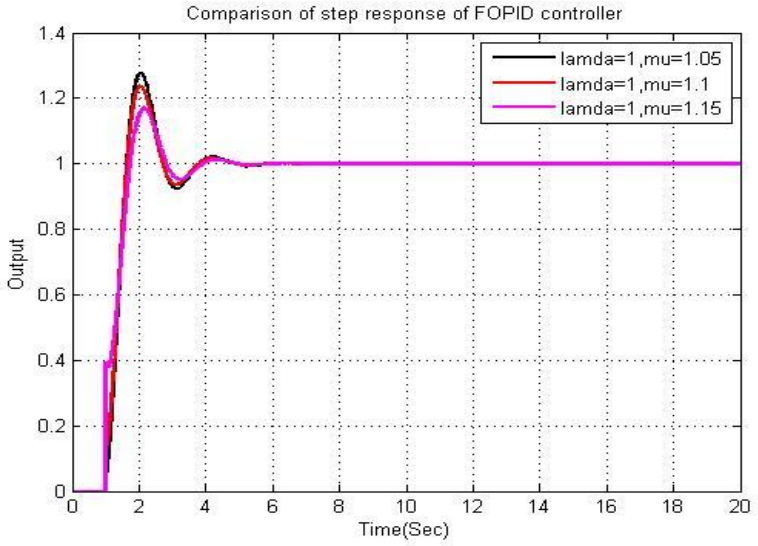

Figure 6.7 Unit step response of DC motor control using FOPID for Different Values of μ>1

The above Figure 6.7 shows the unit step response of speed control of DC motor using fractional order PID controller for λ=1 and μ>1.

Table 6.5 Comparison of Parameters for Different Combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
1	1.05	27.8089	2.0473	5.5645	.191	.2777	.5891	.9141
1	1.1	23.7612	2.0413	4.6560	.191	.232	.5226	.8725
1	1.15	17.1389	2.1465	4.7784	.1906	.1523	.4428	1.052

It can be seen from the above table 6.5 that with the increase in the value of , peak overshoot decreases and settling time ,peak time increases and the performances indices also decreases. So it is not a satisfactory result .

(5) With Varying Values of λ>1 and μ=1

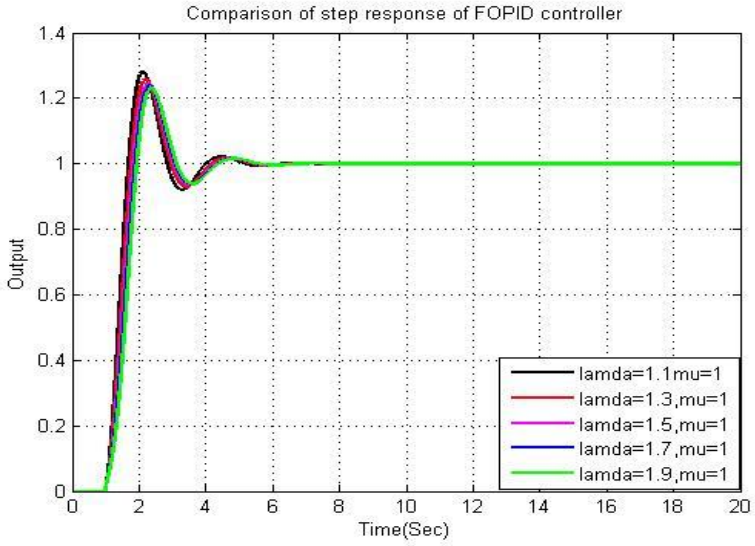

Figure 6.8 Unit step response of DC motor using FOPID for Different Values of λ >1

The above Figure 6.8 shows the unit step response of speed control of DC motor using fractional order PID controller for λ>1 and μ=1

Table 6.6 Comparison of Parameters for Different Combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
1.1	1	28.1038	2.213	5.8632	.3224	.3186	.6525	1.199
1.2	1	26.9088	2.1637	5.906	.2525	.3354	.668	1.22
1.3	1	25.9522	2.2026	5.8922	.2811	.3521	.684	1.243
1.4	1	25.1895	2.2377	5.2537	.3082	.3686	.7003	1.267
1.5	1	24.5857	2.2695	5.2965	.3338	.3847	.7167	1.293
1.6	1	24.1127	2.2978	5.3295	.358	.4005	.733	1.319
1.7	1	23.7982	2.3228	5.3536	.3807	.4158	.7491	1.345
1.8	1	23.476	2.3457	5.3705	.4022	.4307	.965	1.373
1.9	1	23.2754	2.3669	5.3817	.4224	.445	.7805	1.4

It can be seen from the above table 6.6 that with the increase in the value of , peak overshoot decreases and settling time , peak time increases and the performances indices also increases. So it is not a satisfactory result.

(6) With Varying Values of λ>1 and μ>1

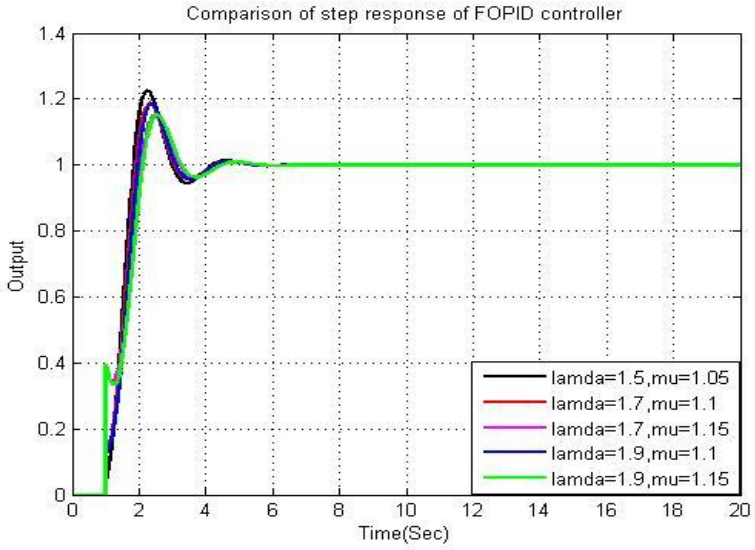

Figure 6.9 Unit step response of DC motor using FOPID for Different Values of λ >1 and μ>1

The above Figure 6.9 shows the unit step response of speed control of DC motor using fractional order PID controller for λ>1 and μ>1.

Table 6.7 Comparison of parameters for different combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
1.5	1.05	22.6561	2.2488	5.1095	.3339	.3588	.669	1.166
1.5	1.1	19.3614	2.2628	5.0276	.3338	.3143	.6149	1.054
1.5	1.15	15.2327	2.4100	5.1690	.3342	.2349	.5083	1.091
1.7	1.05	21.7979	2.3060	5.1665	.3806	.3895	.7018	1.222
1.7	1.1	18.7298	2.3235	5.0846	.3807	.3453	.652	1.116
1.7	1.15	15.2073	2.1774	5.0176	.321	.266	.6141	1.033
1.9	1.05	21.3166	2.3520	5.1959	.4224	.4183	.7342	1.275
1.9	1.1	18.4303	2.3738	5.1230	.4223	.3744	.6873	1.178
1.9	1.15	15.3788	2.5293	5.3213	.4225	.2952	.657	1.262

It can be seen from the above table 6 that from all the different combinations of λ and μ, control parameters for the values of λ=1.7 and μ=1.15 are less than other values of λ and μ.

(7) With Varying Values of λ<1 and μ>1

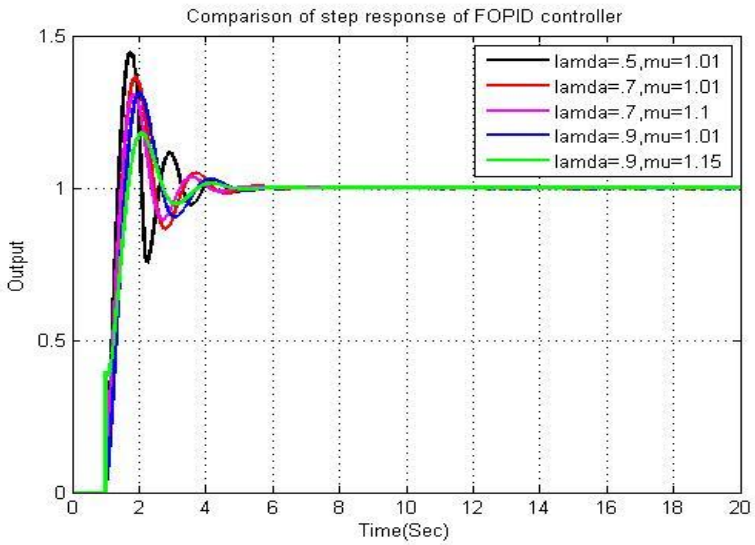

Figure 6.10 Unit step response of DC motor using FOPID for different Values of λ<1 and μ>1

The above Figure 6.10 shows the unit step response of speed control of DC motor using fractional order PID controller for λ<1 and μ>1.

Table 6.8 Comparison of parameters for different combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
.5	1.01	43.5083	1.7294	5.4501	.03574	.2147	.5439	1.012
.5	1.1	38.8578	1.6955	4.8685	.03574	.1687	.4679	.8576
.5	1.15	27.1098	1.6918	4.9308	.0356	.08584	.3453	.7442
.7	1.01	36.4376	1.8956	5.7249	.09309	.2532	.5946	1.12
.7	1.1	30.6041	1.8496	4.7835	.09311	.188	.4819	.866
.7	1.15	21.3701	1.8986	4.8358	.09294	.1074	.3751	.7721
.9	1.01	31.1797	2.0156	5.5637	.1585	.282	.6155	1.139
.9	1.1	25.5206	1.9830	5.1964	.1585	.2162	.5062	.8874
.9	1.15	18.1406	2.0724	4.6213	.1582	.1363	.4186	.836

It can be seen from the above table 6.8 that from all the different combinations of λ and μ, peak overshoot for the values of λ=0.9 and μ=1.15 are less than other values of λ and μ.but all other control parameter are not satisfactory.

(8) With Varying Values of λ>1 and μ<1

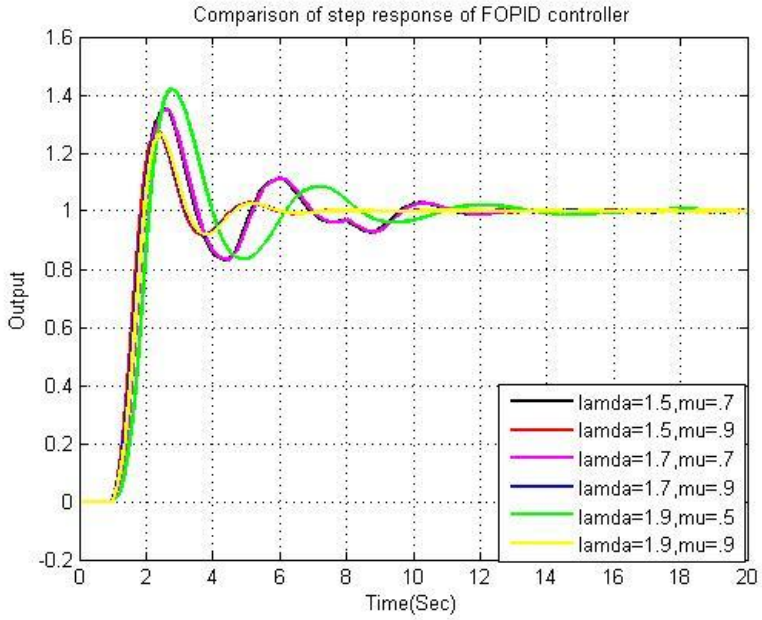

Figure 6.11 Unit step response of DC motor using FOPID for different Values of λ >1 and μ<1

The above Figure 6.11 shows the unit step response of speed control of DC motor using fractional order PID controller for λ>1 and μ<1.

Table 6.9 Comparison of parameters for different combinations of λ and μ

λ	μ	M_p	T_p	T_s	IE	ISE	IAE	ITAE
1.5	.7	35.220	2.5910	20	.3125	.6223	1.937	12.87
1.5	.9	27.588	2.3340	6.920	.3338	.4267	.824	1.652
1.7	.7	34.9529	2.6269	20	.4142	.6187	1.522	6.55
1.7	.9	26.8613	2.3838	6.9712	.3807	.459	.8567	1.705
1.9	.5	42.0534	2.7607	20	.48	.8181	1.989	11.18
1.9	.7	34.8113	2.6542	20	.4107	.6517	1.666	8.283
1.9	.9	26.5052	2.4195	6.9815	.4224	.4893	.8878	1.761

It can be seen from the above table 6.9 that from all the different combinations of λ and μ, control parameters for the values of λ=1.9 and μ=.9 are less than other values of λ and μ.

From the above graphs and tables for different combinations of integral order and derivative order, for λ=1.7and μ=1.15, all the parameters are minimum. Hence combination of λ=1.7 and μ=1.15 is taken for fractional PID controller.

6.3 Step Response Analysis of PSO Based FOPID Controller

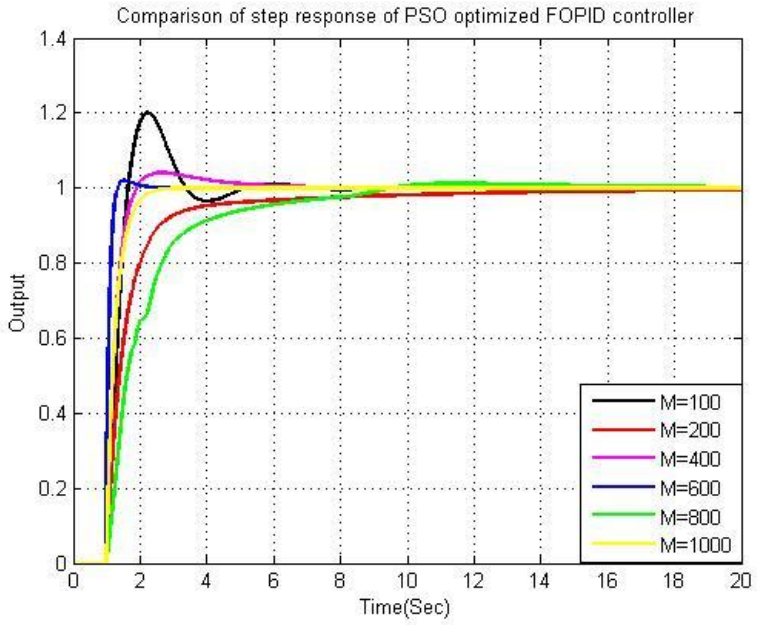

Figure 6.12 Unit step response of speed control of DC motor using FOPID for different number of iteration.

The above Figure 6.12 shows the unit step response of speed control of DC motor using FOPID controller for the iteration M=100,200,300,400,……,1000.

Table 6.10 comparison of parameter of different number of iterations

M	K_p	K_i	K_d	λ	μ	f_{best}	t(sec)
100	.3540	1.3869	.0600	.9374	.5584	.2001	6.27593
200	1.1038	.1403	.5491	.8378	.9561	.1989	18.043227
300	.3674	.4360	.0385	.9843	.4413	.1998	21.012049
400	1.5487	.7366	.3934	.7595	.9387	.1981	24.907939
500	1.1136	.3008	.2759	.5846	.9130	.1975	67.759827
600	.4907	.5887	.0042	.4927	1.205	.1970	47.710881
700	1.0828	.2591	.2577	1.4594	.9069	.1968	60.679252
800	.9519	.1986	1.0390	.8956	.2674	.1973	60.534406
900	.5715	.0582	.0313	1.0559	.5579	.1969	63.134011
1000	1.1082	.7882	.2259	1.0940	.9147	.1965	76.102968

It can be seen from the above table 6.10 that from all the number of iterations M, fitness function (fbest) for the iteration M=1000 is less than other number of iterations.

Table 6.11 Comparison peak overshoot, peak time and settling time response

(M)	(M_p)	(T_p)	(T_s)
100	20.764	2.2248	6.6684
200	-0.4735	10.8775	19.6100
300	6.506	3.6897	9.8633
400	4.0813	2.6326	6.2218
500	2.5445	2.8334	6.5589
600	2.1015	1.5125	2.1048
700	-0.6846	20	20
800	1.1869	11.4874	16.4917
900	-7.220	19.999	20
1000	.0410	3.8196	2.5358

It can be seen that from the above table 6.11 that from all the number of iteration M=1000, peak overshoot, peak time and settling time for M=1000 are less than other number of iterations.

Table 6.12 Comparison of performance indices for different number of iterations

M	IE	ISE	IAE	ITAE
100	.105	.2254	.7782	4.035
200	.9292	.3009	.9292	3.706
300	.3585	.3601	1.063	5.739
400	.1496	.1369	.3488	.6912
500	.1927	.1426	.344	.6671
600	.05675	.02818	.08073	.09507
700	1.195	.2399	1.195	6.445
800	.5592	.5582	1.82	12.16
900	3.19	.6944	3.19	25.64
1000	.02708	.01352	.0286	.0451

It can be seen that from the above table 6.12 that from all the number of iteration M=1000, performances indices for M=1000 are less than other number of iterations.

Hence from all the graphs and tables of PSO based FOPID controller, we can say that all the control parameters for M=1000 iterations are less than other number of iterations. Hence combinations for M=1000 are taken for PSO based FOPID controller.

6.4 Comparison of Different Control Techniques

Unit step response of speed control of DC motor is taken using different control techniques.

- Conventional PID controller with $\boldsymbol{\lambda}$=1 and $\boldsymbol{\mu}$=1
- Fractional order PID contoller (FOPID) with $\boldsymbol{\lambda}$=1.7 and $\boldsymbol{\mu}$=1.15
- PSO based FOPID controller for M =1000 iterations.

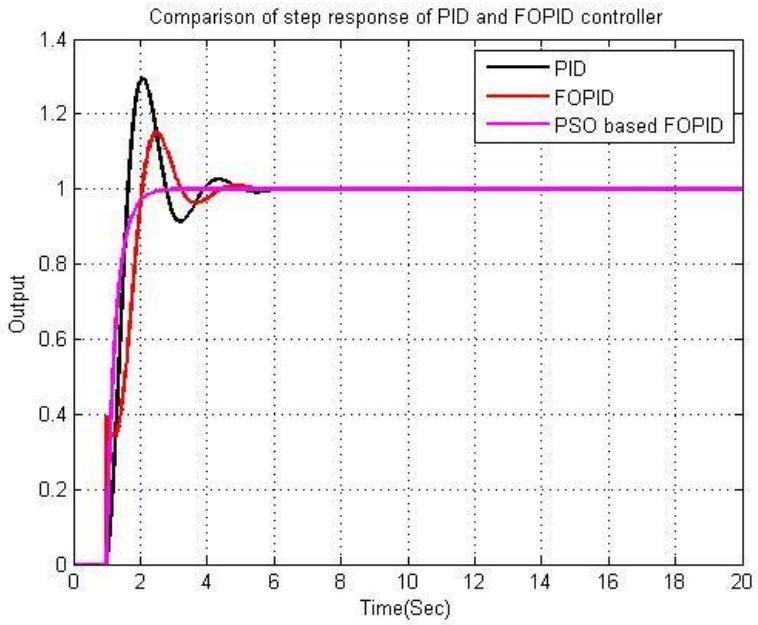

Figure 6.13 Comparison of transient response for different control techniques.

Above Figure 6.13 shows the unit step response of speed control of DC motor using PID, FOPID and PSO optimize FOPID for different combinations of all the parameters .these are best unit step response individually for PID, FOPID and PSO based FOPID. From the tables given below we can conclude which response is best

Table 6.13 Proportional, integral, derivative gain, λ and μ for controllers

Different control techniques	K_p	K_i	K_d	Λ	μ
PID	0.05	0.98	0.0525	1	1
FOPID	0.05	0.98	0.0525	1.7	1.15
PSO based FOPID	1.1082	.7882	.2259	1.0940	.9127

It can be seen from above table 6.13 shows the different type of gains and fractional orders of integral term and derivative term of fractional order PID controller and integer order PID controller.

Table 6.14 Comparison of peak overshoot, peak time and settling time

Different control techniques	M_p	T_p	T_s
PID	29.5939	1.0736	5.4706
FOPID	15.2073	2.1779	5.0176
PSO based FOPID	.0410	3.8196	2.5358

From the above table 6.14,the best peak overshoot (Mp) =.0410 comes for the PSO optimize FOPID controller for the values of proportional gain (K_p) =1.0182,integral gain(K_i)=.7882, derivative gain(K_d)=.2259, λ=1.0940 and μ=.9127.

Table 6.15 Comparison of error performance indices for different controllers

Different control techniques	IE	ISE	IAE	ITAE
PID	.1109	.0272	.628	1.181
FOPID	.321	.266	.6141	1.116
PSO based FOPID	.02706	.01352	.0286	.0451

It can be seen that from the above table 6.15, all the errors for PSO based FOPID are less than all other control techniques.

Hence from all the above tables and figures we can say that all the control parameters peak overshoot (Mp)=0.041,peak time (Tp)=3.8196,settling time (Ts)=2.5358 and all the errors are minimum for iteration M =1000 for the proportional(K_p), integral(K_i) and derivative(K_d) gain those are given in above table 6.13.

Hence particle swarm optimization based fractional order controller (FOPID) gives the best unit step response for speed control of DC motor

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

Servo control is one of the research area, which demands design of an optimal controller. There are many controllers architecture available in control literature, but PID controller is one of the most versatile and widely used controller. More than 90% of the process plants use this as controlling element. But the recent advancement in fractional calculus has introduced applications of fractional order calculus in control theory. One of the prime applications of fractional calculus is fractional order PID controller and it has received a considerable attention in academic studies and in industrial applications. Fractional order PID controller is an advancement of classical integer order PID controller. In many a cases fractional order PID controller has outperformed classical integer order PID controller. This dissertation, studies the control aspect of fractional order controller in speed control of DC motor.

This dissertation considers an DC motor model with unity feedback to control speed of DC motor using feedback controller . The primary goal of the controller is to control the speed of DC motor while manipulating the parameters of controller .This dissertation designs a PID controller, a fractional order PID controller and optimization of FOPID controller using particle swarm optimization. The gain values and differenintegral values obtained through PSO based FOPID controller using particle swarm optimization, provides best transient response than other controller such as integer PID controller, fractional order PID controller.

In the future, the following work may be carried out. Hybrid Neuro Fuzzy (HNF) approach can be used to improve the performance of PSO based controller. Tabu Search (TS) algorithm can be used to optimize PSO based controller. In the present work, I have used simulation to show superiority of the algorithm. Further works may be done to apply it to real system.

REFERENCES

[1] Alan L. Sheldrake, "Hand book of Electrical Engineering", John Wiley & Sons Ltd, 2003.

[2] V. K. Mehta and Rohit Mehta, "Hand book of Principle of Electrical Machines", S. Chand & Company Ltd, 2002.

[3] Theodore Wildi, "Hand book of Electrical Machines, Drives, and Power Systems", Prentice Hall, 2002.

[4] A. Gelen and S. Ayasun, "Effects of PWM Chopper Drive on the Torque-Speed Characteristic of DC motor" 43rd International Universities Power Engineering Conference, pp.1-4, 2008.

[5] P. S. Bhimbhra, "Hand book of Power Electronic", Khanna publishers, 2010.

[6] B.L. Theraja, A.K. Theraja, " A Text Book of Electrical Technology, Volume II, AC and DC Machines", S. Chand & Company Ltd, 1990.

[7] Eric L. Obestar, "DC Motor With Inertia Disk Model Development, Proportional Controller and State Feedback Controller with Full State Estimator", 2005.

[8] Schlegel Milos ,C Ech Martin, "The Fractional Order PID Controller Outperforms the Classical one," Proc. 7th Int. Sci. & Tech. Conf.- Process Control, pp. 1-7, 2006.

[9] Dingyu Xue, Chunna Zhao, Yang Quan Chen, "Fractional Order PID Control of a DC Motor with Elastic Shaft: A case study," Proc. American Control Conference, pp. 3182-3187, 2006.

[10] Chuang Zhao, Xiangde Zhang, "The Application of Fractional Order PID Controller to Position Servomechanism," Proc. 7th World Congress Intelligent Control and Automation, pp. 3380-3383, 2008.

[11] Ramiro S Barbosa, J A Tenreiro Machado, Isabel S Jesus, "On the Fractional PID Control Laboratory Servo system," Proc. 17th World Congress IFAC, pp. 15273-15278, 2008.

[12] Ying Luo and Yang Quan Chen, "Fractional Order [PD] Controller for Robust Motion Control: Tuning Procedure and Validation," American Control Conf., pp. 1412-1417, 2009.

[13] Varsha Bhambhani and YangQuan Chen, "Experimental Study of Fractional Order PI Controller for Water Level Control," Proc. IEEE Conf. Decis. Cont., pp.

1791-1796, 2008.

[14] Hyo-Sung Ahn, Varsha Bhambhani and Yang Quan Chen, "Fractional-Order Integral and Derivative Controller Design for Temperature Profile Control," in Proc. 2008 CCDC, pp. 4767-4771, 2008.

[15] Concepcion A. Monje, Blas M. Vinagre, Vicente Feliu, YangQuan Chen, "Tuning and Auto-Tuning of Fractional Order Controllers for Industry Applications," Control Engineering Practice, pp. 792 – 812, 2008.

[16] Fabrizio Padula, Antonio Visioli, "Tuning Rules for Optimal PID and Fractional Order PID Controllers," Journal of Process Control, pp. 69-81,2011.

[17] Venu Kishore Kadiyala, Ravi Kumar Jathoth, Sake Pothalaiah, "Design and Implementation of Fractional Order PID Controller for Aerofin Control System," Proc. 2009 World Congress NaBIC, pp. 696-701,2009.

[18] Jun-Yi Cao, Jin Liang and Bing-Gang Cao, "Optimization of Fractional PID Controllers based on Genetic Algorithms," Proc. 4th Int. Conf. Machine Learning and Cybernetics, pp. 5686-5689, 2005.

[19] F Gao and H Q Tong, "A Novel Optimal PID Tuning and Online Tuning based on Particle Swarm intelligence," Proc. Int. Conf. Sensing, Comput. and Automation, pp. 182-186, 2006.

[20] Ying Luo and Yang Quan Chen, "Fractional order [PD] Controller for Robust Motion control: Tuning Procedure and Validation," 2009 American control conf., pp. 1412-1417, 2009.

[21] Deepyaman Maiti, Sagnik Biswas and Amit Konar, "Design of a Fractional Order PID Controller using Particle Swarm Optimization Technique," Proc. National Conference on Recent trand in information system, pp. 1-5, 2008.

[22] Li Meng, Dingyu Xue, "Design of an Optimal Fractional-Order PID Controller using Multi-Objective GA Optimization," Proc. Chinese Contr. Desi. Conf., pp. 3849-3853, 2009.

[23] Arijit Biswas, Swagatam Das, Ajith Abraham, Sambarta Dasgupta, "Design of Fractional Order $PI^{\lambda}D^{\mu}$ Controller with Improved Differential Evolution," Engineering Applications of Artificial Intelligence, pp. 343-350, 2009.

[24] Ammar A Aldair, and Weiji J Wang, "Design of Fractional Order Controller based on Evolutionary Algorithm for a Full Vehicle Nonlinear Active Suspension System," Int. J. Control Automation, pp. 33-46, 2010.

[25] Dorf, Richard C. and Robert H. Bishop, “A Text Book of Modern Control Systems”, Prentice–Hall, 2001.

[26] Hans Butler, Ger Honderd, Job van Amerongen, “Model Reference Adaptive Control of a Direct-Drive DC Motor”, IEEE Control Systems Magazine, pp 80-84 1989.

[27] Åström K. J., Hägglund T.“ A Book PID Controllers, Theory, Design and Tuning”, Research Triangle Park: Instrument Society of America, 1995.

[28] B.Wayne Bequette, “A Text Book of Process Control Modeling, Design and Simulation”, Prentice-Hall, 2003.

[29] Bennett, Stuart, “Text Book A History of Control Engineering 1930-1995”, Peter-Peregrinus Ltd, 1993.

[30] Ang, K.H., Chong, G.C.Y., and Li, Y.,”PID Control System Analysis, Design, and Technology”, IEEE Trans Control Systems Tech, pp.559-576, 2005.

[31] Jinghua Zhong ,”PID Controller Tuning: A Short Tutorial”, Spring, 2006.

[32] Derek P. Atherton and S. Majhi-“Limitations of PID Controller” Proceedings American Control Conference , pp.3843-3847, 1999.

[33] M.J. Willis, “Proportional-Integral-Derivative Control”, Dept. of Chemical and Process Engineering University of Newcastle,1998.

[34] Dingyü Xue, YangQuan Chen, Derek P. Atherton, Text Book of Linear Feedback Control Analysis and Design with Matlab”, Society for Industrial and Applied Mathematics, 2007.

[35] Jun-Yi Cao and Bing-Gang Cao, “Design of Fractional Order Controller Based on Particle Swarm Optimization, International Journal of Control, Automation and Systems”, pp. 775-781, 2006.

[36] M. S. Tavazoei, M. Haeri, S. Bolouki, and M. Siami, "Stability Preservation Analysis for Frequency-Based Methods in Numerical Simulation of Fractional Order systems," SIAM Journal on Numerical Analysis, pp. 321–338, 2008.

[37] Ivo Petras- “Text Book of Fractional-Order Nonlinear Systems,Modeling, Analysis and Simulation”, Springer, 2010.

[38] Yang Quan Chen, Ivo Petras and Dingy¨u Xue, “Fractional Order Control-A Tutorial” American Control Conference, pp.1397-1401, 2009.

[39] Igor Podlubny, “Fractional-Order Systems and $PI^{\lambda}D^{\mu}$ controller”, IEEE Transactions on Automatic Control, pp. 208-214, 1999.

[40] Jun-Yi Cao and Bing-Gang Cao, “Design of Fractional Order Controller Based on Particle Swarm Optimization”**,** International Journal of Control, Automation, and Systems, pp. 775-781, 2006.

[41] A. Oustaloup, X. Moreau, and M. Nouillant, “The CRONE Suspension. Control Engineering Practice”, pp.1101–1108, 1996.

[42] B. J. Lurie “Three-Parameter Tunable Tilt-Integral-Derivative Controller”, United States Patent,USA, pp.371- 670,1994.

[43] Jang, J. R., Sun, C. and Mizutani, E., “ Text Book of Neuro-Fuzzy and Soft Computing-A Computational Approach to Learning and Machine Intelligence” , Prentice-Hall,1997.

[44] Zadeh, Lotfi A., "Fuzzy Logic, Neural Networks, and Soft Computing," Communication of the ACM, pp. 77-84,1994.

[45] Beni, G., Wang, J. Swarm “Intelligence in Cellular Robotic Systems”, Proceed NATO Advanced Workshop on Robots and Biological Systems,1989.

[46] David Poole, Alan Mackworth, Randy Goebel, “Computational Intelligence- Text Book of A Logical Approach” ,by Oxford University Press,1997.

[47] Adel A. A. El-Gammal, Adel A. El-Samahy “A Modified Design of PID Controller For DC Motor Drives using Particle Swarm Optimization PSO”, International Conference on Power Engineering and Drives ,Powereng, 2009.

[48] Riccardo Poli , James Kennedy , Tim Blackwell” An Overview of Particle Swarm Optimization”, Springer Science ,2007.

[49] J Kennedy, R Eberhart. “Particle Swarm Optimization[C]”,.In Proceedings of IEEE International Conference on Neural Networks, pp.1942-1948,1995.

[50] Daniel Bratton, James Kennedy” Defining a Standard for Particle Swarm Optimization”, Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS) 2007.

[51] Qinghai Bai, "Analysis of Particle Swarm Optimization Algorithm," Computer and Information Science, pp.180-184, 2010.

[52] Russell C. Eberhart, Yuhui Shi, “Particle Swarm Optimization Developments, Applications and Resources” IEEE Conference, pp. 81-86,2001.