SPEED CONTROL OF DC MOTOR USING EVOLUTIONARY COMPUTATIONAL BASED IP CONTROLLER

By:

SAJID ALI

317-FET/MSEE/S13

Supervised By:

DR. SUHEEL ABDULLAH MALIK

DEPARTMENT OF ELECTRONIC ENGINEERING,

FACULTY OF ENGINEERING AND TECHNOLOGY

INTERNATIONAL ISLAMIC UNIVERSITY,

ISLAMABAD

1. Electric motors, Direct current - Design and construction
2. Theses

Thesis entitled

SPEED CONTROL OF DC MOTOR USING EVOLUTIONARY COMPUTATIONAL BASED IP CONTROLLER

Submitted to International Islamic University Islamabad

In partial fulfillment of the requirements

For the award of degree of

MASTER OF SCIENCE

IN

ELECTRONICS ENGINEERING

BY

SAJID ALI

317-FET/MSEE/S13

SESSION 2013-2015

DEPARTMENT OF ELECTRONICS ENGINEERING

INTERNATIONAL ISLAMIC UNIVERSITY,

ISLAMABAD

DECLARATION

I, **Mr. Sajid Ali**, Reg. No. **317-FET/MSEE/S13** student of MS electronics engineering in Session 2013-2015, hereby declare that the matter printed in the thesis titled "**SPEED CONTROL OF DC MOTOR USING EVOLUTIONARY COMPUTATIONAL BASED IP CONTROLLER**" is my own work and has not been printed, published and submitted as research work, thesis or publication in any form in any University, Research Institution etc. in Pakistan or abroad.

Sajid Ali: ______________________

Dated: ______________________

Certificate of Approval

This is to certify that the work contained in this thesis entitled, **<u>"Speed Control of DC Motor Using Evolutionary Computational Based IP Controller"</u>**was carried out by**Sajid Ali, Registration# 317-FET/MSEE/S13,** it is fully adequate in scope and quality, for the degree of MS (Electronic Engineering).

Viva Voce Committee

<u>Supervisor</u>
Dr. Suheel Abdullah Malik
~~Assistant~~ Associate Professor
DEE, FET, IIU, Islamabad

<u>Internal Examiner</u>
Dr. Adnan Umer Khan
Assistant Professor
DEE, FET, IIU, Islamabad

<u>External Examiner</u>
Dr. Abdul Basit
Senior Scientist
PINSTECH, Islamabad

<u>Chairman DEE FET IIU, Islamabad</u>
Prof. Dr. Muhammad Amir

<u>Dean FET, IIU Islamabad</u>
Prof. Dr. Aqdas Naveed Malik

Dean
Faculty of Engineering & Technology
International Islamic University Islamabad
www.iiu.edu.pk

ACKNOWLEDGEMENTS

Praise to Almighty ALLAH, the most Benevolent, the Omnipotent and Merciful, the Creator of the Universe, who enabled me to accomplish this declaration. Blessing on MOHAMMAD (S.A.W), the seal of prophets and his pious progeny. I am grateful to International Islamic University, Islamabad for providing me a chance to enhance my skills and knowledge through this thesis and I wish to express my deepest gratitude to my worthy and honorable supervisor **Dr. Suheel Abdullah Malik** for his kind supervision with keen interest and valuable guidance. His professional understanding and profound knowledge and kind attitude helped me in compiling this dissertation. His constructive criticism and directional guidance always motivated me to work hard.

I am especially grateful to Mr. Aamir Shahzad for a helpful discussion during my research work. And in the end I would like to pay my special thanks to my friends. Amal Daraz, Ahmad Khan, Khalid Ibrahim and Imran Khan who helped me in preparing my thesis.

I ask ALLAH to help all those who have helped me and guided me from beginning to the end of my thesis.

ABSTRACT

The automatic control played main role in the innovation of science and engineering. DC motor is an important device in industries used for driving various loads, thus the implementation of motor speed controller is important. The main function of speed controller of a motor is to keep the rotation of the motor at preset speed and to drive a system at the demanded speed. In this thesis, we have implemented proportional-Integral (PI) and Integral-Proportional (IP) controller for a DC motor to control its speed. The proportional gain (K_P) and Integral gain (K_i) of PI and IP controller are adjusted using Ziegler-Nicholas (ZN) method. Further, the popular evolutionary computing techniques like Simulated Annealing (SA) and Genetic Algorithm (GA) have been used for the tuning of the PI and IP controllers. To adjust the gain of PI and IP controller using ZN tuning method, first we take the open loop step response of the system and extract the value of delay time (L_t) and time constant (τ) from the open loop step response of the system and calculate the value of K_P and K_i.

For the calculation of gain parameters of a controller using soft computing techniques like SA and GA, first formulate the fitness function of PI and IP controllers and then choose the value of K_P and K_i at which the fitness function gives minimum value. Comparison between the ZN, GA and SA output was done on the basis of the simulation results obtained by MATLAB (Simulink). The simulation results demonstrate that the response of GA based PI and IP controller in terms of overshoot, settling time and rise time is better as compared to ZN, SA based PI and IP controller. The response of SA based PI and IP controller is better than the ZN based PI and IP controller. Furthermore the overshoots in IP controller are less than the PI controller either tuned by ZN, GA or SA.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

An electric motor is a device which transform electrical energy in to mechanical energy. On the basis of motor input supply electric motors are divided in to two main categories: Alternating Current (A.C) and Direct current (DC) motors. In DC motor the input supply is provided by direct current source while in case of A.C motor the input supply is provided by an alternating current source.

DC motor are highly used in industry where wide speed control range is required. Some of the significant applications of DC motor drives are paper mills, textile mills, hoists, printing press, braking, rolling mills, excavators, machine tools and cranes. DC motor are simple in construction and less expensive than A.C motor. DC motor drives are highly used for position and adjustable speed control system [1]. Now a days in industry to control the speed of motor is very essential, for example if we use a DC motor in radio car controller then it is not possible to get the desired speed for giving a constant power. It will run slower above rough way, uphill and quicker on downhill, which make essential to design a controller for DC motor to drive it in the desired speed [2].

The mathematical equation describing the speed of a DC motor is given as [3].

$$N \propto \frac{V - I_a R_a}{\varphi} \qquad (1)$$

Here N=speed of motor, V=applied voltage in volts, $I_a R_a$= voltage across armature in volts, φ=flux in weber

From equation (1) we see that speed of a DC motor depends upon the following three factors:

i. Flux
ii. Voltage across the armature
iii. The voltage applied

Hence the speed of a DC motor can be change by changing any one of the above three factors. The objective of a controller designed for motor to control its speed is to receive a signal that represent the desired speed and to drive it at that speed [4]. There are two different type of a controller which are used to control the speed of DC motor: one is closed loop and second is open loop speed controller. To measure the actual speed of motor we used closed loop controller while open loop controller cannot measure the actual speed. The response of close loop controller is better as compared to the open loop controller but it is more complex and expensive due to feedback components. Closed loop control system is mostly used for precise speed control of a motor. Figure 1.1 show closed loop block diagram of DC motor to control its speed [5].

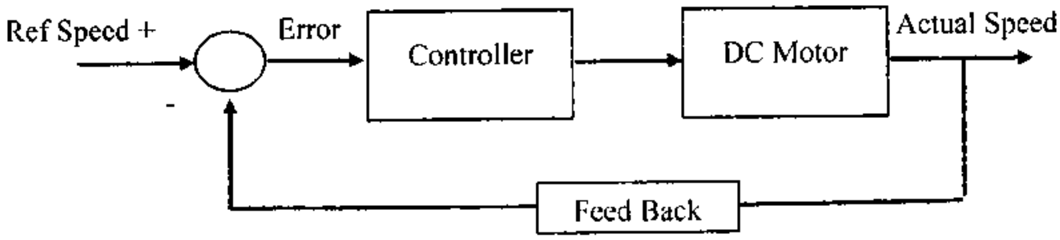

Fig 1.1: DC Motor Closed Loop Block Diagram

From the block diagram we see that to generate the error signal we take the difference between the reference and actual speed and vary the voltage of a motor to control its speed.

A DC motor give brilliant speed control response for both speeding up and down. DC motors are usually less expensive for high power rating. DC motors are highly used for variable speed machines and a wide range choices for this purpose have been evolved. For these purpose the DC motor should give the precision control to get the desired result. Different controller like PI, Proportional-Integral-Derivative (PID), Proportional-derivative (PD), Fuzzy logic controller (FLC), Evolutionary computational based PI, PD and PID controllers or combination of these controllers are used for speed control of DC motor [6].

1.1 Problem statement

Speed control of a DC motor is the main problem in industry. A lot of work has been done for DC motor to control its speed, Different controllers like PI, PID, PD, Particle Swam Optimization (PSO) based PI, Genetic Algorithm (GA) based PID, particle swam optimization (PSO) based PID, PSO based Fractional order PID, FLC etc. has been made for DC motor to control its speed. However IP controller has not been explored very much. Further the combination of evolutionary computing techniques and IP controller has not been explored. The purpose of this research is to explore the application of tuning of IP controller using evolutionary computing techniques for DC motor to control its speed and compare the results with evolutionary computational based PI controller.

IP controller is a very simple and economical controller; its implementation is also very easy. Therefore it is a great worth to control the speed of a DC using IP controller. In this research, PI and IP controllers have been designed and implemented for DC motor speed control. Moreover evolutionary computation has been explored for optimum tuning of the controller.

1.2 Goals and objectives

This thesis is basically conducted to achieve the following goals and objectives

i. Detail analysis of speed control methods for DC motor.
ii. Mathematical modeling of DC motor.
iii. Analysis and modeling of PI and IP controllers.
iv. Comparative analysis between PI and IP controllers.
v. Tuning of PI and IP controller using evolutionary computing techniques like Genetic Algorithm (GA) and Simulated Annealing (SA).
vi. Investigation of transient response Characteristics.

1.3 Thesis Outline

Chapter 1 provides brief introduction of DC motor Speed control. Chapter 2 give the detailed literature survey of speed control of DC motor. In Chapter 3 mathematical modeling and transfer function of DC motor are presented. Chapter 4 explain the PI and IP controller used for DC motor speed control. Tuning of PI and IP controller using Ziegler-Nicholas (ZN) tuning method are also described. Chapter 5 explain the brief introduction of soft computing techniques, SA and GA (Genetic Algorithm). The basic working principle of GA and SA for tuning of PI and IP controllers are discussed, further Fitness function of PI and IP controllers are formulated. Chapter 6 present Simulink model and results of the DC motor speed control. Chapter 7 provide the conclusion and recommended future work.

CHAPTER 2

LITERATURE REVIEW

Speed control of motor is one of the main problem of the industry. Speed control of motor means intentional change in speed to drive the motor at a specific speed to perform the specific work. A lot of work has been done for a DC motor to control its speed. To control the speed of a DC motor different controllers are used, from which some of them are specified as under:

i. P.I controller
ii. P.D controller.
iii. P.I.D controller.
iv. Evolutionary computational based PID controller.
v. PWM Techniques.
vi. Armature voltage control method
vii. Rheostat control method.
viii. Fuzzy logic controller.
ix. Fuzzy adaptive PID controller.
x. Fuzzy logic PI controller.
xi. GA based fuzzy controller.

Atual Dewangan et al. used ATmega8L for controlling the speed of DC motor that is fed by DC chopper, pulse width modulation (PWM) signal are used to drive the chopper circuit. The speed of a motor can be controlled by controlling the motor terminal voltage using PWM duty cycle [4].

Sadiq et al. control the speed of DC motor using voltage control method applied to the field circuit and compare the results with the armature voltage control method. Simulation results show that the transient response of the field circuit voltage control technique is superior than the armature voltage control method at the rated speed of 1500 and 2000 rpm and the designed FLC is efficient for wide range of speed (0-2000rpm), also the transient response of the motor at 2000 rpm is better than 1750 rpm [7].

Anurag Dwivedi used rheostat control method for controlling the field and armature voltage of a DC motor to control its speed. The speed of a motor below normal speed are controlled using Armature rheostat control method while the speed above the normal speed are controlled using field rheostat control method. So by using these two methods together, we can obtain extensive range of speed control for different applications. The main problem to use this system is large size of rheostat across the armature due to which a large amount of the power are wasted [8].

G Mishra et al. do comparative analysis of the performance of DC motor speed control using armature voltage control method with the combined armature voltage and field current control method using PI controller. Simulation results indicate that the speed and efficiency of DC motor at full and half load torque condition with combined field current and armature voltage control method using PI controller is better as compared to armature voltage control. Buck converter are used to control simultaneously field current and armature voltage of a motor [9].

A Adday and othman improve the efficiency of a DC motor using FLC for controlling both the field current and armature voltage simultaneously. To control the field and armature voltage DC-DC converter are used. Results show that motor efficiency is increased in low and medium load torque by using armature fuzzy logic speed controller. We cannot use this for high load torque because at high load motor efficiency decrease [10].

S. Singh et.al. Used PI controller for a DC servo motor to control its speed. Design of PI controller for DC servo motor is simple, economical and powerful. Use of PI controller for DC servo motor is an efficient method to improve the transient response of the motor. PI controller minimize the settling time, rise time and reduce the overshoot approximate to zero [11].

R Kanojiya et.al. Used PSO based PI controller for a DC motor to control its speed. Further the results of PSO based PI controller were compared with PID controller tuned by Ziegler-Nicholas (ZN) and Modified Ziegler-Nicholas (MZN) tuning method. Simulation results indicate that the results of PSO based PI controller are much better than ZN and MZN based PID controller in terms of settling time, overshoots and rise time . Furthermore PSO based PI controller are, economical, efficient and easy in implementation than ZN and MZN based PID controller [12].

Ch Prakash and R Naik used PID controller tuned using Ziegler-Nicholas (ZN) algorithm to control position of DC motor. Comparison of the result of Ziegler-Nicholas based PID controller with the conventional PID controller are made. Simulation results indicate that transient response (i.e. overshoots, settling time and rise time) of ZN based PID controller performance are better as compared to conventional PID controller. Further, the performance of a controller can be improved by using genetic algorithm for tuning purpose [13].

Venkateswarlu and Chchengaiah compare the performance of PID and FLC for a DC motor to control its speed. The transient response of DC motor obtained without controller is not satisfactory. There exit a dead time of one second which is the main drawback of the system. To remove this drawback used PID controller which improve the transient response of the motor but failed to remove the dead time of one second. FLC is used to remove this dead time. The step response of FLC is smoot and ripple free [14].

Kushwah and P. Patra compare the performance comparison of different tuning techniques for tuning of a PID controller used for speed control of DC motor. They compare the results of conventional PID controller with evolutionary computational based PID controller and found that the transient response (i.e. overshoots, settling time and rise time) of computational based controller are much better than the conventional controller tuned by Ziegler-Nicholas tuning method [15].

M Jaiswal and M Phadnis used genetic algorithm (GA) for the tuning of a PID controller to control the speed of a DC motor. The transient response of motor can be enhanced by well tuning of PID controller. They compare the result of PID controller tuned by GA with conventional PID controller and found that the GA based PID show superiority over conventional controller in transient response like rise time, settling time and overshoot [16].

S Dubey and Srivastava analysis the performance of proportional derivative (PD) and PID controller for DC motor to control its speed. From the analysis it is observed that main drawback of PD controller is steady state error as PID controller has zero steady state error therefore PID controller can be used to overcome the drawback of PD controller, to make the steady state error to zero the overshoots are observed. To decrease the overshoot we have to increase the derivative gain but the rise time of the system increases as a consequence. However there is a compromise between the speed response and overshoot. Overall speed response of PID controller is better than PD controllers [17].

M Telbany used artificial bee's colony (ABC) for PID controller to control the speed of a DC motor. He compare the result of artificial bees colony (ABC) based PID controller with conventional PID controller. The rise time, overshoots and settling time of ABC based PID controller are better than PID controller [18].

A Gammal and Asamahy use multi-Objective- PSO for tuning of PID controller for separately excited DC motor to control its speed. Results show that multi-Objective- Particle Swarm optimization (MOPSO) based PID controller minimize overshoot, tracking error, steady state error, rise time and settling time of the system [19].

Salim et.al. Compare the result of FLC based on Laboratory virtual Instrument Engineering Workbench (Lab VIEW) with PI and PID controllers that is tuned using Ziegler-Nicholas tuning method. To control the applied voltage of DC motor a controller is designed on the basis of fuzzy rule to change the speed error and speed control of motor. Simulation results indicate superiority of FLC controller over PI and PID controller. The transient response (settling time, peak time, rise time, dead time) of FLC controller is lower than the PID controller tuned by Zeigler-Nicholas (ZN) tuning method [20].

M Faiz et.al. Use Genetic Algorithm (GA) based type2 FLC for DC motor to control its position. They compare the results of GA based type2 FLC with GA based type1 FLC. The performance of these GA based type2 FLC with GA based type1 FLC are analyze on the basis of several measurements like settling time, overshoot, rise time, integral time multiplied absolute error (ITAE) and Integral absolute error (IAE). In each case the performance of GA based type2 FLC is better than the GA based type1 FLC. GA based type2 FLC has ability to remove the non-linearity of the system especially when a un modelled dynamics are introduced [21].

M Ramesh et.al. Used fuzzy logic (FL) PI controllers to control the speed of brushless direct current (BLDC) motor. Three PI and three FLCs are used. The rule base used for FLC are same for these three controllers. The fuzzy logic and PI controllers used for low, medium and high speed are connected in parallel and the fuzzy logic, PI controllers used to control the same speed like small speed are connected in series. The output of FLC goes to the input of PI controller and the output of PI controller goes to the input of current controller. The simulation results indicate that the speed response of BLDC motor using F.L based PI controller performance is better as compared to the conventional FLC [22].

P Agrawal et.al. Compare the results of PI controller with FLC for brushless DC motor to control its speed. The performance of traditional controllers like PI controllers are better under small set of conditions and highly used in industry due to economical and simplicity in construction but in case of high load disturbance and nonlinearity performance of these controllers is not satisfactory. To overcome these problems a FLC has been introduced. The FLC can be implemented easily to overcome the load disturbance and nonlinearity problem of the system. The simulation results of PI and FL controllers are compared for speed control of BLDC motor and found that the speed and torque response of FLC is better than conventional PI controllers. The response of FLC is better than PI controllers in terms of overshoot, rise time and settling time [23].

K Sujatha et.al. Compare the performance of conventional PI, fuzzy PI controllers with neural network for a brushless DC motor (BLDC) to control its speed. Neural network show better performance than the conventional PI and fuzzy PI controllers. The simulation results show that ITAE (integral time absolute error) and IAE (integral absolute error) are small in neural network as compared to PI and fuzzy PI controllers. The transient response (rise time, overshoot and settling time) of neural network is small the conventional PI and fuzzy PI controllers [24].

C Xia et.al. Use GA based FLC for Brushless DC motor to control its speed and compare the result with PID controller. To control the speed of BLDC motor using genetic algorithm based FLC two loops are made. One is the inner current loop which is used to control the motor torque and second is the FLC whose control rules are adjusted and set the parameters based on genetic algorithm. By comparing the result of GA based FLC with PID controller, it is found that the response of GA based FLC method in terms of robust and dynamic performance of the system is better than the PID controller [25].

Z. Yachen and H. Yueming tune the PID, increment PID and Integral separation PID controllers using SA algorithm. Compare the results of these three controllers on the basis of minimizing the steady state error and integral time absolute error (ITAE). Simulation results show that every controller has its own advantages and are superior in different control requirement. Generally conventional PID controller is so developed and diffused in to many control cases. IPID save from much calculation but in some high precision system. the high overshoot is deva sting. so ISPID controller is best for the combining strategy to accomplish the optimal performance [26].

Y. Soni and R. Bhatt design a SA based PID controller for a stable LTI (Linear time invariant) continuous system using IAE, MSE, ISE and ITAE error criteria. The gainK_p,K_i and K_d are adjusted using SA to meet the desired performance specification. A comparison of PID controller performance is observed on the basis IAE, MSE, ISE and ITAE error criteria. Simulation results show that the response of SA based PID controller is better for IATE which have 0% overshoot and very small settling time 2.26e^-005 seconds [27].

R soni et.al. used conventional PID and F.L controllers for DC motor speed control. As PID controller is very simple and economical due to which it is highly used in industry. FLC are mostly used to control the performance of a controller. In this paper PID controller has been implemented for DC motor to control its speed and then improved controller performance using FLC. from the Simulation results it is observed that transient response of FLC is better than the conventional PID controller [28].

G Huang and S Lee Used Lab VIEW software for designing of a PID controller for DC motor to control its speed and to present its speed response in real time. The application of motor to monitor in real time not only monitor traditional instrument but also monitor either machine is working typically or not. This system is economical and more efficient than other methods as it combine the needed instruction of DC motor and constructed on the assembly of the PC. The actual response of PID controller by proper tuning of gain parameters K_p, $K_i$and K_d can be obtain with this theory [29].

D Geng et.al. Used fuzzy adaptive PID controller for brushless DC motor to control its position. According to the control system to adjust the gain parameters of PID controller fuzzy adoptive control theory is used. Simulation results indicate that the fuzzy adaptive control system has fast response, robust and strong coupling, and the static and dynamic characteristics are much better than the conventional PID controller. The system robustness can be improved by using fuzzy based PID controller. The system designed by this method is feasible, effective and correct [30].

CHAPTER 3

MATHEMATICAL MODELING OF DC MOTOR

In order to derive the transfer function of DC motor, its mathematical model is divided in to two main parts. One is the electrical part and second one is the mechanical part. We derive the differential equation of these two parts separately and then combine these two differential equations to find the resultant transfer function of DC motor. The electrical circuit diagram of the DC motor is given in figure 3.1 [31].

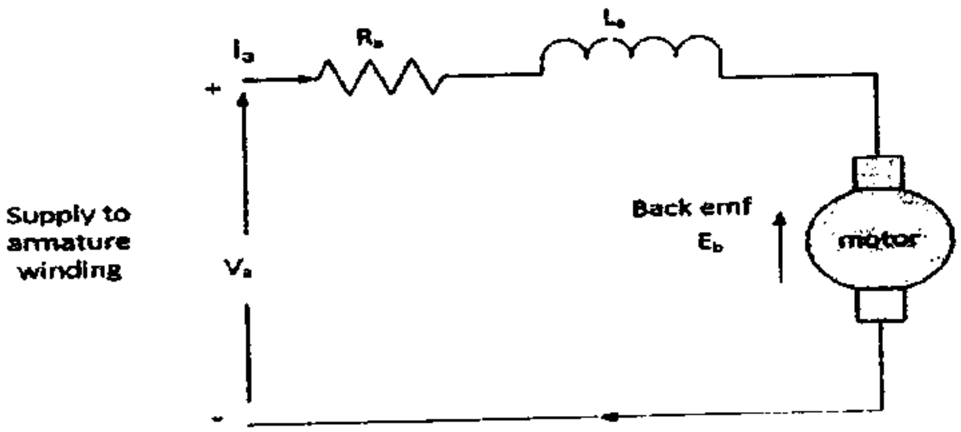

Fig 3.1: Equivalent Electrical circuit diagram of DC motor

3.1 Electrical Equation of DC Motor

Now by applying the Kirchhoff voltage law to the loop in figure 3.1 we have

$$V = V_{R_a} + V_{L_a} + E_b \tag{3.1}$$

Here V_R= voltage across the resistor, V_L = voltage across the inductor, E_b = back emf of motor

As

$$V_{L_a} = L_a \frac{dI}{dt}, \quad V_R = IR_a, \quad E_b = K_e \frac{d\theta}{dt}$$

By putting these values in (3.1) we get

$$V = IR_a + L_a\frac{dI}{dt} + K_e\frac{d\theta}{dt} \tag{3.2}$$

Rearranging (3.2)

$$L_a\frac{dI}{dt} = V - R_a - K_e\frac{d\theta}{dt}$$

$$\frac{dI}{dt} = \frac{\left[V - IR_a - K_e\frac{d\theta}{dt}\right]}{L_a} \tag{3.3}$$

3.2 Mechanical Equation of DC Motor

Using newton's law the torque equation of a motor can be written as below

$$T_m(t) = J_m\frac{d^2\vartheta}{dt} + B_m\frac{d\vartheta}{dt} \tag{3.4}$$

As torque is proportional to current and flux

$$T_m \propto \varphi I_a$$

$$T_m = K_t I_a$$

By putting the value of torque in (3.4) we get

$$K_t I_a = J_m\frac{d^2\vartheta}{dt} + B_m\frac{d\vartheta}{dt} \tag{3.5}$$

Now by taking Laplace transform of (3.3) we get

$$SI(s) = \frac{[V(s) - R_a I(s) - K_e S\theta(s)]}{L}$$

Rearranging the above equation

$$L_a SI(s) = [V(s) - R_a I(s) - K_e S\theta(s)]$$

$$I(s)[SL_a + R_a] = [V(s) - K_e S\theta(s)] \tag{3.6}$$

By taking Laplace transform of (3.5) we get

$$K_t I(s) = J_m S^2\theta(s) + B_m S\theta(s)$$

Rearranging the above equation

$$K_t I(s) = \theta(s)[J_m S^2 + B_m S] \quad (3.7)$$

Dividing (3.6) by (3.7) yields the transfer function $\frac{\omega(s)}{V(s)}$ as follows

$$\frac{I(s)[SL_a + R_a]}{K_t I(s)} = \frac{V(s) - K_e S\theta(s)}{\theta(s)[J_m S^2 + B_m S]}$$

Rearranging the above equation

$$S\theta(s)(J_m S + B_m)(SL_a + R_a) = K_t V(s) - K_t K_e S\theta(s)$$

$$S\theta(s)[(J_m S + B_m)(SL_a + R) + K_t K_e] = K_t V(s)$$

$$\frac{\omega(s)}{V(s)} = \frac{S\theta(s)}{V(s)} = \frac{K_t}{(J_m S + B_m)(SL_a + R_a) + K_t K_e} \quad (3.8)$$

Here

$\theta(S)$ = angular displacement (radian)

$\omega(S)$ = angular speed

L_a = inductance (henry)

I = current (ampere)

V_a = voltage (volts) across armature

K_e = back emf constant (volts/ (rad/sec))

K_t = torque constant in N.m/ampere

R_u = armature resistance (ohm)

J_m = moment of inertia (kg.m^2/rad)

B_m = fractional constant (N.m/ (rad/sec))

Figure 3.2 show the block diagram of DC motor described by (3.8).

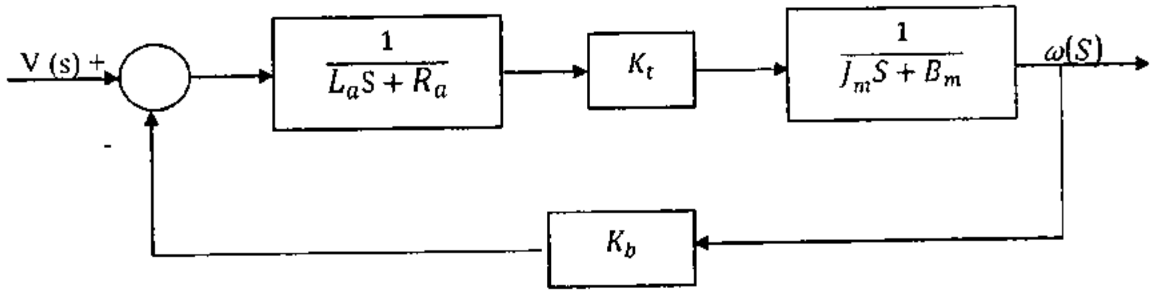

Fig-3.2: Block diagram of armature voltage control system of DC motor

The value of motor parameters used in the research work are given in table 3.1.

Table-3.1: parameters value of DC motor [32].

Parameters	Values
Armature resistance (R_a)	0.6 ohm
Back EMF constant (K_e)	0.55 (volt/(rad/sec))
Torque constant (K_t)	0.55 (N-m/ampere)
Moment of inertia (J_m)	0.0465 (kg-m^2/rad)
Armature inductance (L_a)	0 (henry)
Fractional constant (B_m)	0.004 (N-m/(rad/sec))

Putting the parameters value from Table-3.1 in (3.8) and neglecting inductance L_a we obtain the required transfer function as given in (3.9)

$$\frac{\omega(s)}{V(s)} = \frac{0.55}{(S\times 0+0.6)(0.0465S+.004)+0.550.55}$$

$$\frac{\omega(s)}{V(s)} = \frac{0.55}{0.0279S+0.0024+0.3025}$$

$$\frac{\omega(s)}{V(s)} = \frac{0.55}{0.0279S+0.3049}$$

$$\frac{\omega(s)}{V(s)} = \frac{19.71}{S+10.92} \quad (3.9)$$

CHAPTER 4

PROPORTIONAL INTEGRAL (PI) AND INTEGRAL PROPORTIONAL (IP) CONTROLLERS

4.1 Controller

A controller of a motor is such a device that helps to run the motor at a specific speed to perform the specific work. The controller of a motor may be automatic or manual for starting or stopping a motor, selecting reverse or forward rotation, regulating and selecting the specific speed, diagnosing the fault, protecting against overloads and controlling the torque.

Every motor has to have some sort of controller. type of a controller depends upon the motor task. The simplest controller of a motor is an electric switch that are used to connect the motor with power supply to turn on or off the motor. The switch may be operated manually or some sort of relay with sensor to turn on or off the motor automatically.

A closed loop control system is also known as feedback control system. A closed loop systems are designed to automatically achieve and maintain the desired output by comparing it with the actual value. It is done by generating the error signal which is the difference between desired value and actual value. PI and IP controllers are used in a closed loop whose description are given in the preceding sections.

4.2 Proportional Integral (PI) Controller

A closed loop feedback PI controller is mostly used in industry for speed control of DC motor. The PI controller consist of proportional gain (K_P) and integral gain (K_i). PI controller has its capability to create the zero steady state error in the step change with respect to reference speed [33]. Proportional Integral controller is defined by (4.1)

$$u(t) = K_p \times e(t) + K_i \times \int_0^t e(t)dt \qquad (4.1)$$

Here

$u(t)$ = Reference signal

e (t) = Error signal

K_p= Proportional gain

K_i= Integral gain

Proportional Integral (PI) control algorithm is simple and economical algorithm used for DC motor to control its speed. Figure 4.1 show block diagram of PI controller used for speed control of DC motor. From figure 4.1 we see that in feed forward path both the proportional and integral gain of PI controller are placed. Block diagram indicates that PI controller has two loop, one external speed loop and second internal current loop. E(s) is the speed error which is the change between the real and reference speed which is feed to PI controller. The output of the PI controller $E_1(s)$ acts as a reference input current of motor to control its speed [34].

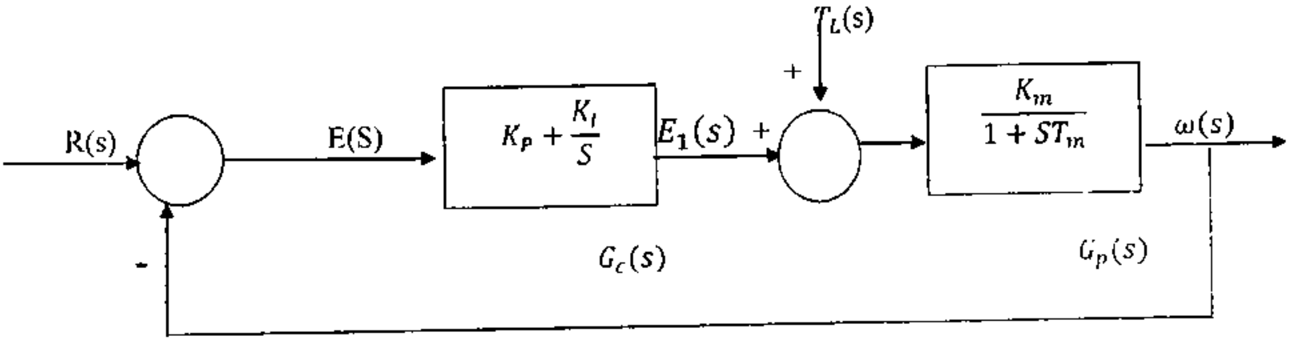

Fig-4.1: Block Diagram of PI Controller

The transfer function of PI controller between the actual speeds $\omega(s)$ to reference speed R(s) is given as follows:

$$\frac{\omega(s)}{R(s)} = \frac{K_m(SK_P+K_i)}{T_m S^2+(1+K_m K_p)S+K_m K_i} \quad (4.2)$$

Here

K_p=proportional gain

K_i= Integral gain

T_m= motor mechanical constant

K_m= gain constant of motor

Now the transfer function obtained between the real speed $\omega(s)$ and the load torque $T_L(s)$ is given as follows:

$$\frac{\omega(s)}{T_L(s)} = \frac{S(1+ST_m)}{T_m\,K_pS^2+(K_m+T_m\,K_i+K_p)S+K_i} \tag{4.3}$$

Figure 4.2 show basic simulink model of PI controller. The error indication which is change between the real and reference speed is set to the input of PI controller, depending upon the gain value of PI controller. The PI controller tries to minimize the error, output of the controller is given to the input of motor.

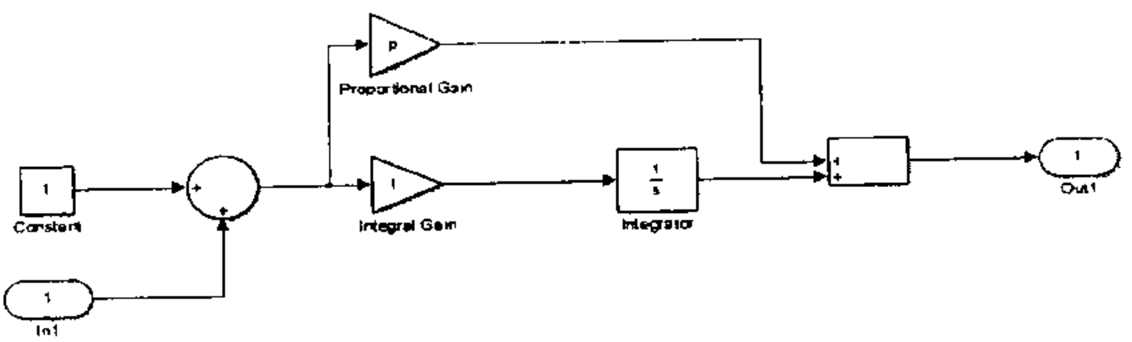

Fig-4.2: Basic Simulink model of PI controller

4.3 Integral Proportional (IP) Controller

Integeral proportional (IP) controller is an modified form of the Proportional Integral (PI). Most of the properties of IP controller are similar to PI controller. The disadvantgae of PI controller like sluggish response due to abrupt variation in disturbance, controller gain sensitivity and high overshoot can be reduced by using comparatively new integral proportional (IP) controller. Integral Proportional (IP) controller was introduced first time in 1978 [35]. Figure 4.3 Show block diagram of IP controller.

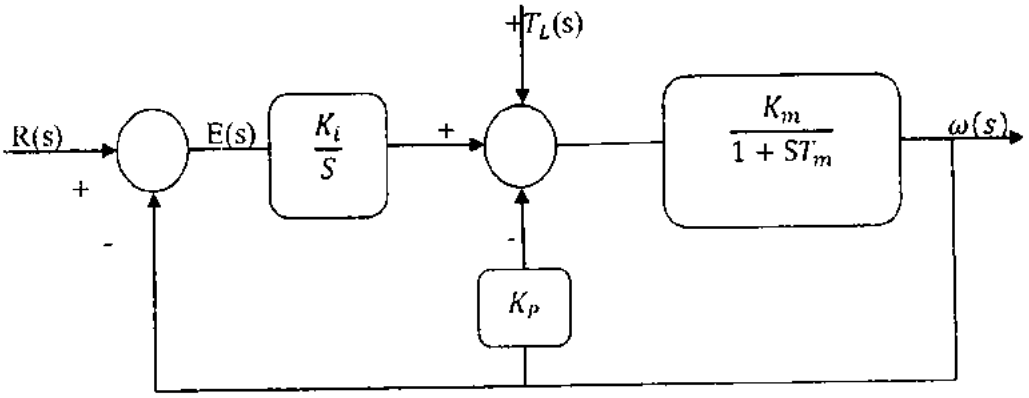

Fig-4.3: Block diagram of IP controller

From figure 4.3 we see that in IP controller proportional gain (K_P) are put in feedback path, while integral gain (K_i) are put in feed forward path. The block diagram shows three loops of the IP controller, one internal current loop, second speed feedback loop and third proportional gain K_P feedback loop. The speed error E(s) is set to an integrator with gain $K_i$and the speed is feedback through a proportional gain K_P[36].

The transfer function obtained for IP controller between the real speed $\omega(s)$ and reference speed $R(s)$is given as follows:

$$\frac{\omega(s)}{R(s)}=\frac{K_m K_i}{T_m S^2+(1+K_m K_p)S+K_m K_i} \tag{4.4}$$

Here

K_p=proportional gain

K_i= Integral gain

T_m= motor mechanical constant

K_m= gain constant of motor

Now the transfer function obtained between the actual speed $\omega(s)$ and the torque disturbance $T_L(s)$ is given as follows:

$$\frac{\omega(s)}{T_L(s)}=\frac{S(1+ST_m)}{T_m K_p S^2+(K_m+T_m K_i+K_p)S+K_i} \tag{4.5}$$

From the (4.2) and (4.4) we observe that the zero which are present in (4.2) is removed in (4.4) which indicates that the IP controller overshoots will be minimium than PI controller. (4.3) and (4.5) of PI and IP controller are same, therefor the load disturbance response of PI and IP controller is likely to be similar to each other. The basic Simulink model of IP controller are shown in figure 4.4.

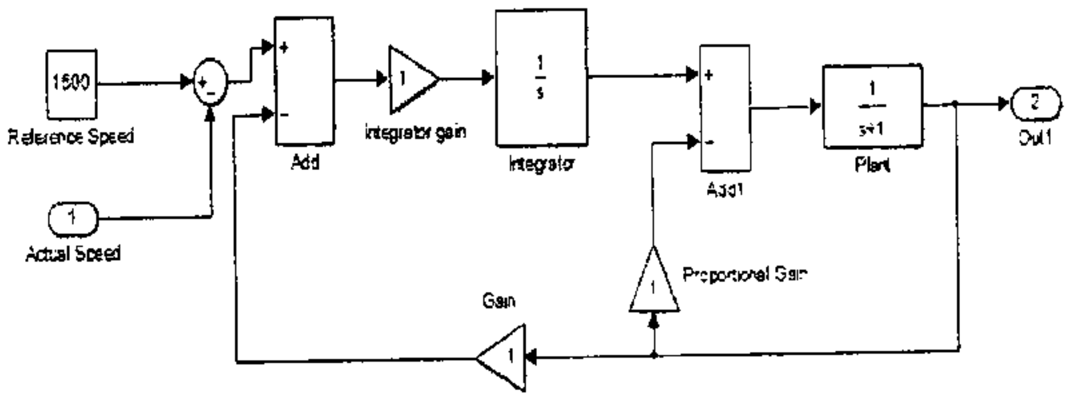

Fig-4.4: Basic Simulink model of IP controller

4.4 Tuning of PI and IP Controllers using Ziegler-Nicholas Method

4.4.1 Tuning

The method of selecting the gain parameters proportional gain (K_P) and integral gain (K_I) of PI and IP controller to get the desired system response called tuning of a controller. Different tuning methods are used for tuning of a controller, which are divided in to two main classes:

i. Close loop methods
ii. Open loop methods

In close loop method a controller is tuned automatically while in an open loop method the tuning of a controller is done manually. The closed loop tuning methods which are mostly used for simulation are as follows:

i. Ziegler Nicholas (ZN) method
ii. Modified Ziegler-Nicholas (MZN) method
iii. Tyreus Luyben method
iv. Damped oscillation method

Open loop tuning methods:

i. Open loop Ziegler Nicholas method
ii. CHR method
iii. Fertik method
iv. Cohen and coon method
v. IMC method
vi. Ciancone Marline method [37].

4.4.2 Ziegler-Nicholas Method

Ziegler-Nicholas tuning method is a trial and error method which was introduced by Ziegler Nicholas in 1942. Ziegler-Nicholas tuning method is a typical method which are used for tuning of all order system. For the tuning of a first order system using Ziegler Nicholas tuning method, first we take the open loop response of the system. From the open loop response of the system which is shown in the figure 4.6, we note the time constant (T) which is the time to reach 63.2% of final value of the system and delay time (L) which is the time to reach half of the final value. These parameters value can be found by drawing a tangent line at the inflection point of the open loop step response system as shown in figure 4.5 [38].

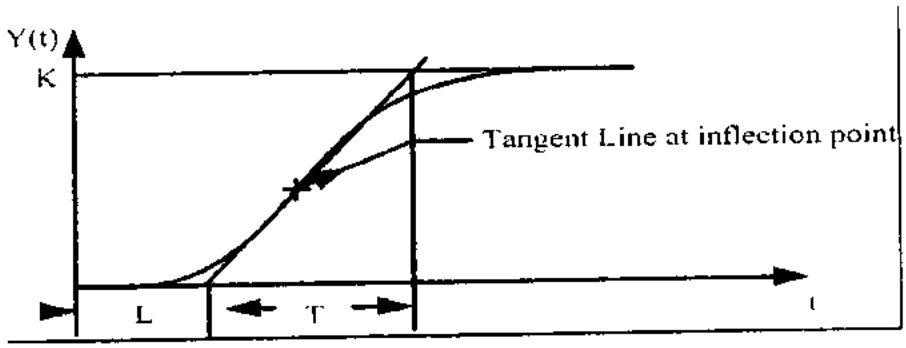

Fig-4.5: Response Curve of system for Ziegler-Nichols Method

Figure 4.6 show open loop response of the system, from this response we extract the value of delay time (L_t) and time constant (τ) to calculate the gain of a controller.

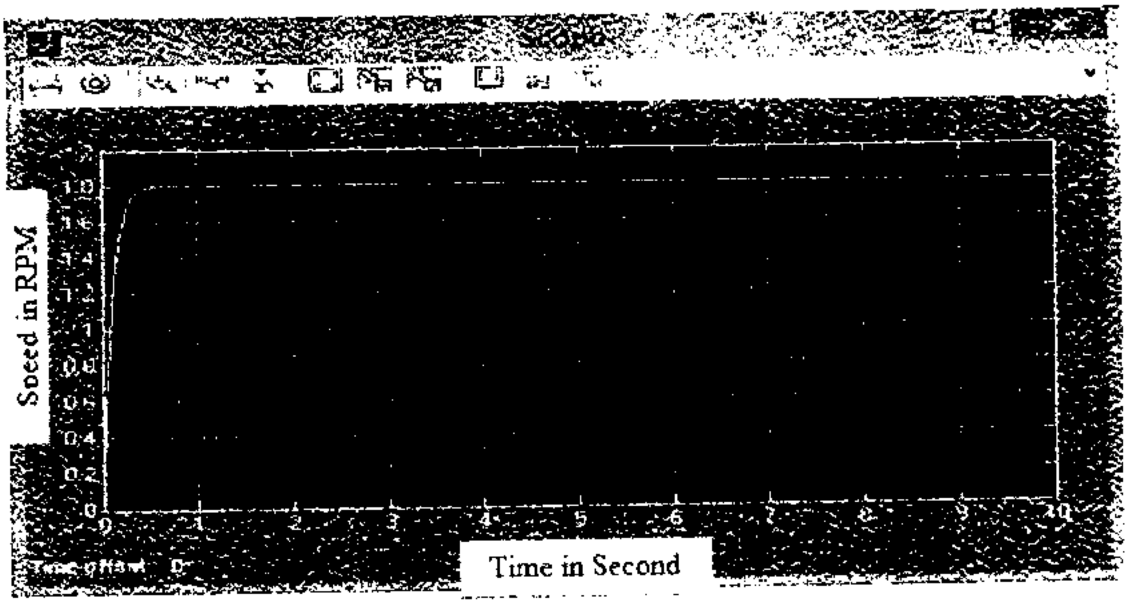

Fig-4.6: open loop response of system

From (3.9) the transfer function of DC motor is given as follows:

$$\frac{\omega(S)}{V(S)} = \frac{19.71}{S+10.92}$$

Comparing (3.9) with the standard equation of transfer function

$$\frac{\omega(S)}{V(S)} = \frac{1}{S+a}$$

We obtained

$$a = 10.92$$

$$\text{Time constant} = \tau = \frac{1}{a} \qquad (4.6)$$

By putting the value of a in (4.6)

$$\text{Time constant} = \tau = \frac{1}{10.92} = 0.0915 \text{ s}$$

Using figure 4.6 we find

$$\text{Delay time} = L_t = 0.068$$

By using Ziegler Nicholas formula [38] for tuning of first order system

$$K_p = \frac{0.9\times\tau}{L_t} \tag{4.7}$$

$$T_i = \frac{L_t}{0.39} \tag{4.8}$$

$$K_i = \frac{K_p}{T_i} \tag{4.9}$$

Putting value of τ and L_t in (4.7) we get

$$K_p = \frac{0.9\times 0.0915}{0.03} = 1.2$$

Putting value of L_t in (4.8) yields

$$T_i = \frac{0.068}{0.39} = 0.1$$

By substituting T_i in (4.9) gives us

$$K_i = \frac{1.2}{0.1} = 12$$

The gain values Proportional gain (K_p) and Integral gain (K_i) obtained using Zeigler-Nichols tuning are given in table-4.1.

Table-4.1: Zeigler Nichole based PI and I-P controllers gain values

Gain Parameters	K_p	K_i
Gain Values	1.2	12

CHAPTER 5

EVOLUTIONARY COMPUTING TECHNIQUES

Evolutionary computing (E.C) techniques is a heuristic computing technique used for optimization of different problems. E.C techniques are generally executed on computer and pretends evolution. The main advantages of E.C techniques are flexibility, simplicity, robustness and good gain parameter optimization of a controller [39]. In the process of E.C techniques initially a random population is created, which is composed of a huge number of chromosomes, while individual chromosomes represents the solution of problem. A fittest is selected during the basic steps of selection, cross over and mutation. The main drawback of this techniques is that there is no guarantee of desired solution within the define limit of time [40]. A large number of E.C techniques algorithm are used for optimization of a problems. Some of them are as follows:

i. Genetic Algorithm (GA)
ii. Particle Swarm Optimization (PSO)
iii. Simulated annealing (SA) algorithm.
iv. Evolutionary Strategy.
v. Differential Evolution (D.E).
vi. Ant Colony Optimization.
vii. Parallel Simulated Annealing
viii. Firefly Algorithm
ix. Self-organization migrating Genetic Algorithm,
x. Harmony Search (HS).

Though, the distinct study is partial to GA and SA whose particulars are given as:

5.1 Genetic Algorithm (GA)

GA is a one of the subclass of Evolutionary Computing (E.C) Techniques. GA is an optimization techniques based on the random selection, which was first time introduce by John Holland in 1962 [40-41]. GA is a computer search optimization techniques that are used to find the local minima or maxima of any problem. In the process of GA initially a random population consisting a large number of chromosomes are selected where each chromosomes represent the solution of the problem. Fitness of each chromosomes is computed using fitness function. Parents are selected from initial population on the basis of their fitness level and then take the crossover of randomly selected parents to find the optimal solution. This process continue until we cannot achieve the desired result [42].

GA has three main evolution operator:

i. Selection
ii. Cross over
iii. Mutation

5.1.1 Selection

In this operator fittest chromosomes from old population are nominated to create the fresh population. The fitness of a chromosomes depends upon evaluation of objective or fitness function. The chromosomes having fittest solution has maximum chances to survive. There are many other selection process which are used for the selection of chromosomes for new population but the basic principle of selection of all the process are same i.e. a chromosomes having fittest value have large selection probability. Selection is a random method, the most common methods which are used for selection are [43]:

i. Normalized geometric selection
ii. Roulette wheel selection
iii. Tournament selection
iv. Stochastic universal sampling.

In this work stochastic universal sampling was used as it gave us best results.

5.1.2 Crossover

After completing the selection stage, cross over operation is applied. In this operation take a cross over between two parent's chromosomes to produce new chromosomes called offspring which have property of both the parent chromosomes. In crossover operation mostly two chromosomes are randomly selected and interchange their some part to form a new chromosomes. Crossover operators have three basic types such as single point crossover, two point crossover and multi point crossover.

5.1.3 Mutation

After the completion of crossover operator, mutation operator started. Mutation are used to sustain the genetic range of chromosomes from one generation to other. Mutation change a small value of chromosomes gene from its initial state to make better solution. In mutation a chromosomes may change completely from the previous chromosomes. Mutation occur within the define mutation probability, usually this value set to low value like 0.1%. If we set the mutation probability too high value then the search will turn to a primitive arbitrary search. Mutation can be performed for all types of encoding techniques like:

i. Binary encoding mutation
ii. Permutation encoding mutation
iii. Value encoding mutation
iv. Tree encoding mutation.

5.1.4 Algorithm for GA and SA

Step 1: (Population Initialization)

A population of P chromosomes are randomly produced, where each chromosomes denotes the solution of the problem.

Step 2: (Evaluation of fitness)

Using the objective function/fitness function compute the fitness value of each chromosomes from current population. According to their fitness values each chromosomes assign rank individually.

Step 3(Stoppage Criteria)

If the maximum number of cycles has been exceeded or predefined fitness value is achieved then the algorithm terminates else repeat steps 2 to 5.

Step 4: (Reproduction)

Using cross over operation a new population containing best fitness value called parents is created which yield offspring to act as parentages for next generation.

Step 5. Mutation

Mutation operation is executed when there is no enhancement in the fitness in a generation.

The working principle of GA used for tuning of a PI and IP controller can be described by a flow chart which are shown in the figure 5.1.

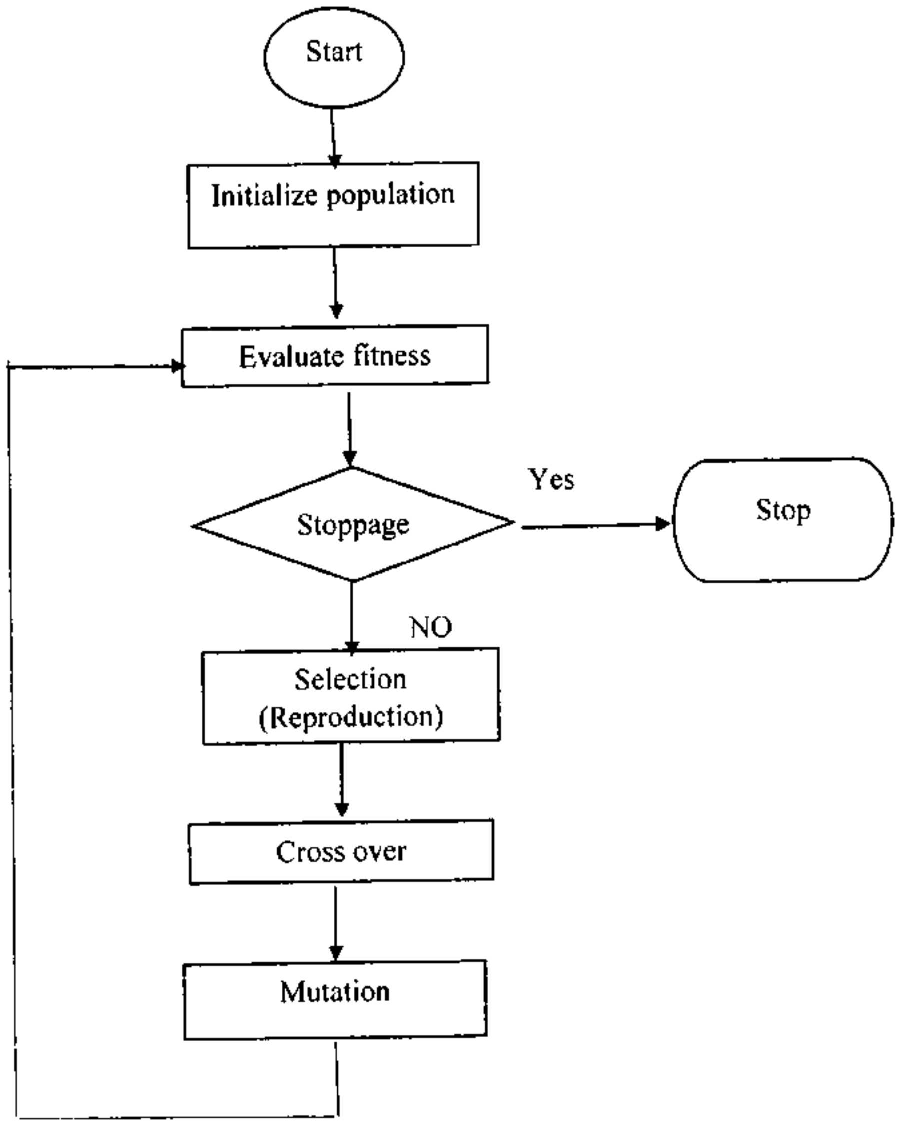

Fig 5.1: Flow chart of Genetic Algorithm

5.2 Simulated Annealing (SA)

SA is an optimization techniques which was introduced by kirk-Patrick in 1982. This techniques is used to find the global minima of a function containing multiple variables [44]. Different methods like Qusai Newton, golden search, conjugate-gradient, davidon-fletcher-powell, steepest descent [45] etc. are considered the aggressive methods used to find the minima of the objective function. The minima found by these techniques is a local minima, we cannot get the global minima by these techniques. To find the global minima of objective function we have to repeat the optimization several time, starting from the different initial guess. The local minima found by this is considered as global minima. In this regard, SA algorithm is a simple and efficient method to find the global minima of an objective function [46].

Simulating Annealing (SA) is a general probabilistic metaheuristic techniques used to find the global optimization of applied mathematics problem. The convergence abilities of SA are extraordinary in intelligent system designs and pattern analysis [47]. SA has a robust capability to find desired results. Different type of simulating annealing like hybrid simulated annealing (HSA), parallel SA, clustering algorithm, division algorithm etc. are applied for optimization problems. Other than to find global minima SA can also be used in primary distributor feeders to find the optimal allocation of capacitors, phase balancing, distribution networks configurations and reactive planning [48].

The main advantages of SA is that it can be used to find the both local and global optimizer and also it is simple, robust and efficient than other mathematical methods [49]. However, SA diverges in the optimization of singular, highly convex and stiff situation. This drawback can be removed by hybrid approaches with SA which provide stability and robustness in the solution. In SA the initial population of solution space is created on the basis of distributed neighborhood points. The size of the population depends upon the nature of problem which is to be optimized. Figure 5.2 indicate the flow diagram of SA process.

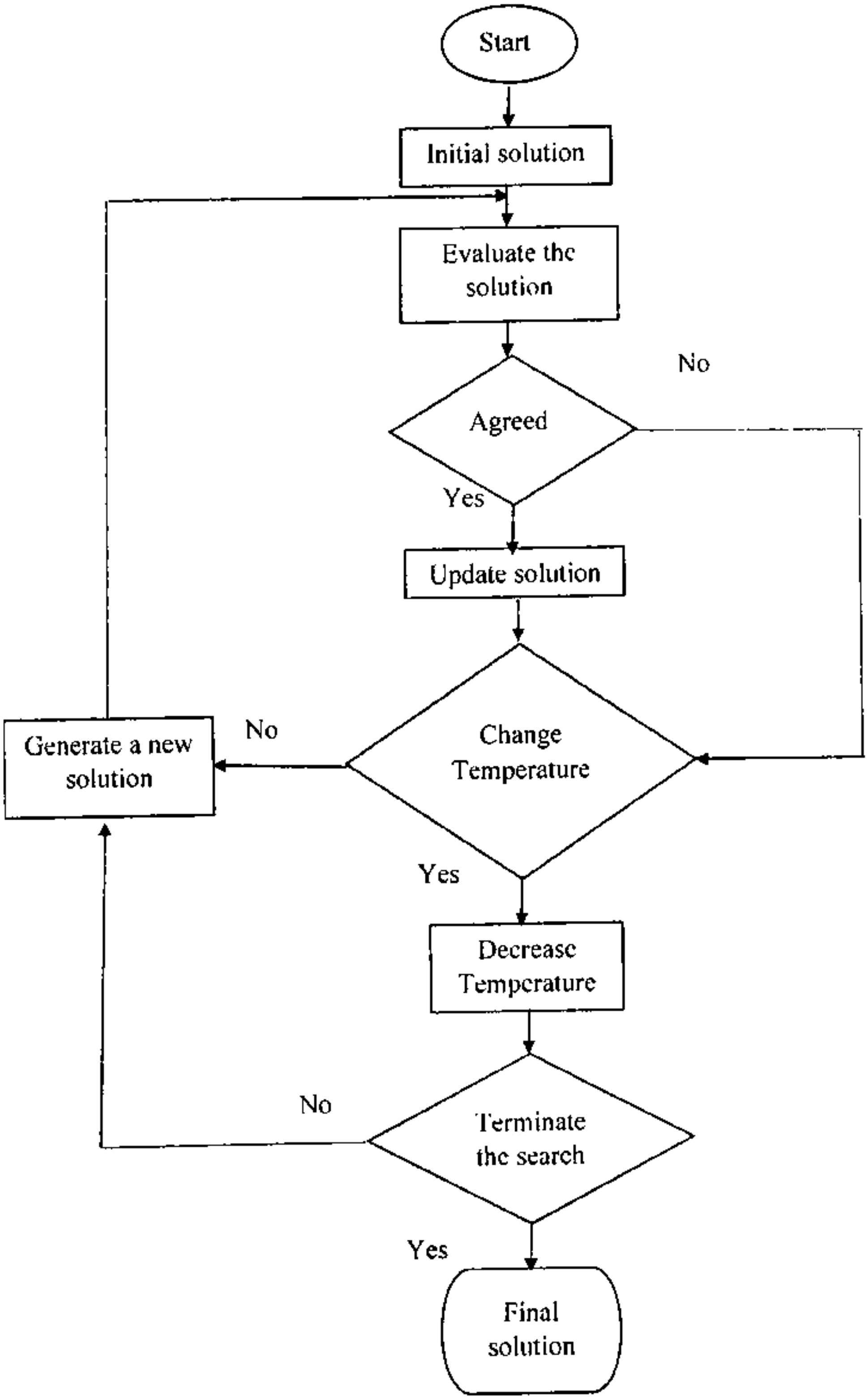

Fig-5.2. Flow Diagram of Simulating Annealing

5.3 Tuning of PI and IP controller using Genetic Algorithm and Simulated Annealing

GA and SA are optimization techniques which are used to find the gain parameters of a controller. For this purpose first we find the fitness function of a controller then find the value of those gain parameters which minimize the error [50].

5.3.1 Formulation of Objective function /Fitness Function for PI controller

Fitness function are calculated to minimize the error of a controller which is the difference between the actual speed and desired speed. Now by using figure 5.3 we calculate the error function which is defined as:

$$E(s) = R(s) - Y(s) \quad (5.1)$$

Here

$E(s)$= error

$R(s)$= desired speed

$Y(s)$= actual speed

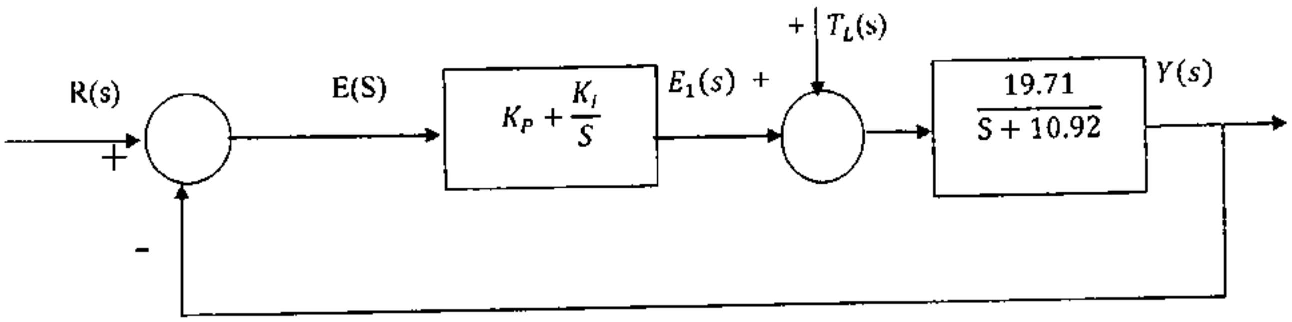

Fig-5.3: Closed loop transfer function for PI controller

From the figure 5.3 we calculate the value of $Y(s)$ which is given as:

$$Y(s) = \left[E(s)\left(K_p + \frac{K_i}{s}\right) + T_L\right]\left(\frac{19.71}{s+10.92}\right) \quad (5.2)$$

Putting (5.2) in (5.1) we have

$$E(s) = R(s) - \left[E(s)\left(K_p + \frac{K_i}{s}\right) + T_L\right]\left(\frac{19.71}{s+10.92}\right). \qquad (5.3)$$

Rearranging (5.3) and putting the value of $R(s)$ and $T_L(s)$ we get

$$E(s) = \frac{-8355\times S+16380}{S^2+10.92\times S+(K_p\times S+K_i)\times 19.71} \qquad (5.4)$$

By taking the inverse Laplace of equation (5.4) using mat lab we obtained $E(t)$ which are derived in Appendix A and put these value of $E(t)$ in (5.5).

$$\text{Integral Square Error} = ISE = \int_0^t (E(t)^2)dt \ \ [48] \qquad (5.5)$$

$E(t)$ Given in appendix (A-4) contains constant K_p and K_i. The values of these constants are obtained using SA and GA.

5.3.2 Formulation of Fitness Function for IP controller

By using figure 5.4 we calculate the fitness function for IP controller to minimize the error which is given as:

$$E(s) = R(s) - Y(s)$$

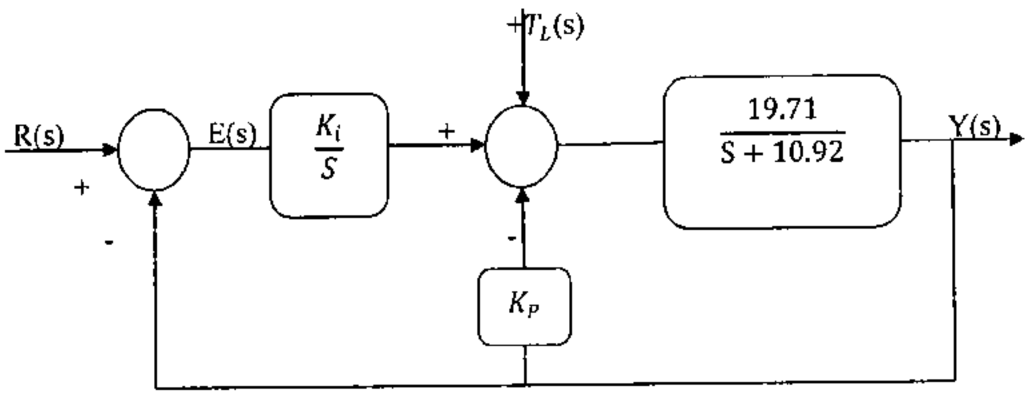

Fig-5.4: Closed loop transfer function for IP controller

Calculate the value of $Y(s)$ using figure 5.4 is given as:

$$Y(s) = \left[E(s)\left(\frac{K_i}{s}\right) + T_L - K_pY(s)\right]\left(\frac{19.71}{s+10.92}\right) \qquad (5.6)$$

Simplifying (5.6) for Y(s)

$$Y(s) = \frac{E(s)K_i19.71+ST_L19.71}{S\times(S+10.92+K_p19.71)} \tag{5.7}$$

Putting value of $Y(s)$ from (5.7) to (5.1) we get

$$E(s) = \frac{-8355\times S+16380+K_p29565}{S^2+10.92\times S+(K_p+K_i)\times S\times 19.71} \tag{5.8}$$

Now by taking the inverse Laplace of (5.8) using mat lab we get $E(t)$ which are given in Appendix (B-5) and put the value of these $E(t)$ in (5.5) to find the value of constants using SA and GA. To obtain the optimal value of K_p and K_i gains SA and GA are executed.

5.6 Tuning of PI and IP using GA

Parameters value which are used for optimization of PI and IP controller are given in table 5.1.

Table-5.1: Parameter Setting of GA

Parameters	GA Settings	
	PI	IP
Population Size	50	30
Number of Generations	100	120
Cross over function	Scattered	Scattered
Bounds	[0 0]-[50 50]	[2.5 2.5]-[20 20]
Selection function	(Stochastic Uniform)	(Stochastic Uniform)
Chromosomes	2	2
Mutation function	Constraint dependent	Constraint dependent

To achieve proportional(K_p) and integral (K_i) gains, GA is implemented according to the given setting in table 5.1. It was saw by several simulations that these setting deliver best fitness. The optimum values of $K_p$and K_i attained using GA are providing in table 5.2.

Table-5.2: GA based PI and IP Controller Gain values

Gain Parameters	K_P	K_I
GA-PI	13.8	8.92
GA-IP	2.5	20

5.7 Tuning of PI and IP Controller using SA

SA parameters value used for minimizing the fitness function of PI and IP are given in table 5.3.

Table-5.3: Parameter Setting of SA

SA		
Parameters	Settings	
	PI	IP
Maximum iteration	1000	1000
Function tolerance	e^{-2}	e^{-2}
Initial temperature	50	50
Bounds	[0 0]-[100 100]	[2.5 2.5]-[20 20]

SA is performed for obtaining optimum value of K_p and K_i for PI and IP controllers. The parameters values of SA used to find optimum values of K_p and K_i are given in table 5.3. From simulation results it was observed that these settings provide best fitness. The optimum values of K_p and K_i achieved using SA are given in table 5.4.

Table-5.4: SA based PI and IP Controller Gain values

Gain Parameters	K_P	K_I
SA-PI	13.93	23.8
SA-IP	2	15

CHAPTER 6

SIMULATIONS AND RESULTS

Simulation results for speed control of a DC motor using proportional integral (PI) and integral proportional (IP) controller are presented in this chapter. Mat lab simulation of PI and IP controllers using Ziegler-Nicholas (ZN) tuning method and evolutionary computing techniques like GA and simulated annealing (SA) used for tuning of controller are also presented. Mat lab R2012b version are used for simulation.

6.1 System Open Loop Response

Open loop Simulink model of DC motor speed control is shown in figure 6.1. In open loop response of the system there is no feedback. DC motor with parameter given in table-3.1 has been used, 1500 r.p.m speed was set to reference speed of motor. The system transfer function was given a step input and observe its output response which is shown in figure 6.2. From figure 6.2 we see that system response set at the value of 2700RPM instead of 1500 RPM. The system does not come to the desired value until at infinity time. Therefore to set the system response at desired value, we need to design a controller.

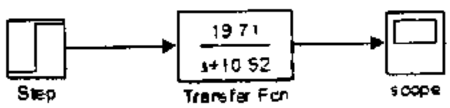

Fig-6.1: Open loop Simulink model of DC motor

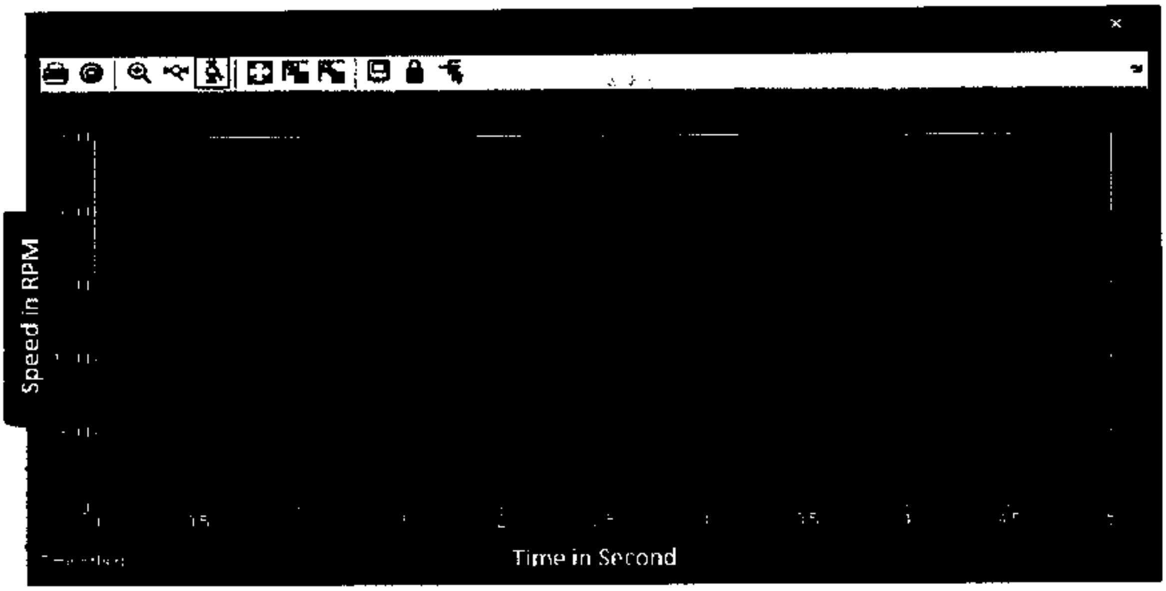

Fig-6.4: Response of System with ZN based PI controller

6.3 Ziegler Nicholas Based IP controller

Figure 6.5 and 6.6 shows Simulink model and result of IP controller. figure 6.6 show that system response with IP controller have very small value of overshoots 11.8%, which are 28.4% less as compared to the PI controller. Thus the response of any system can be improved by using IP controller.

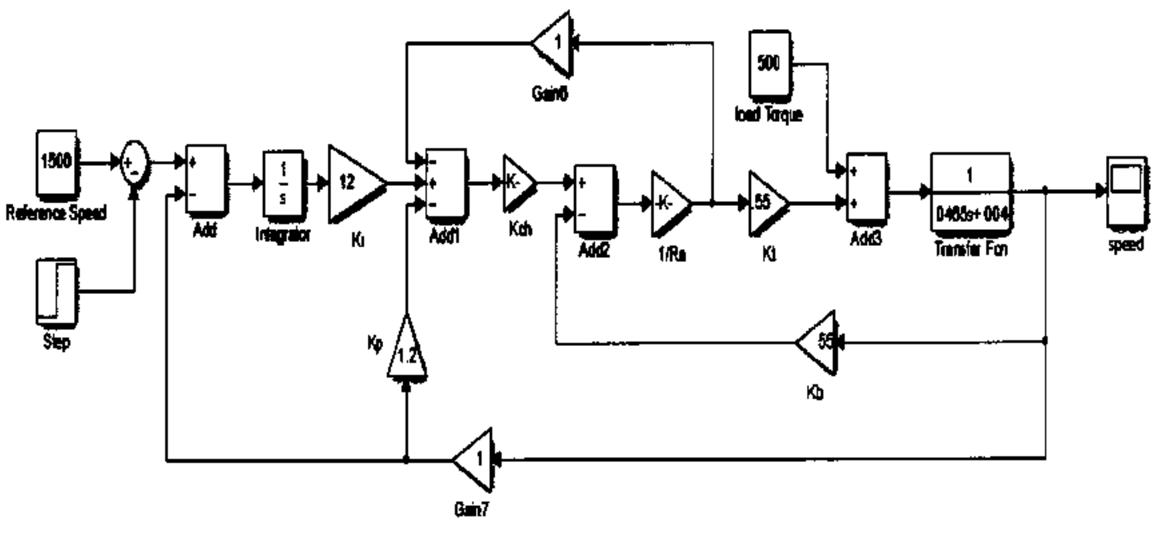

Fig-6.5: Simulink model of ZN based IP controller

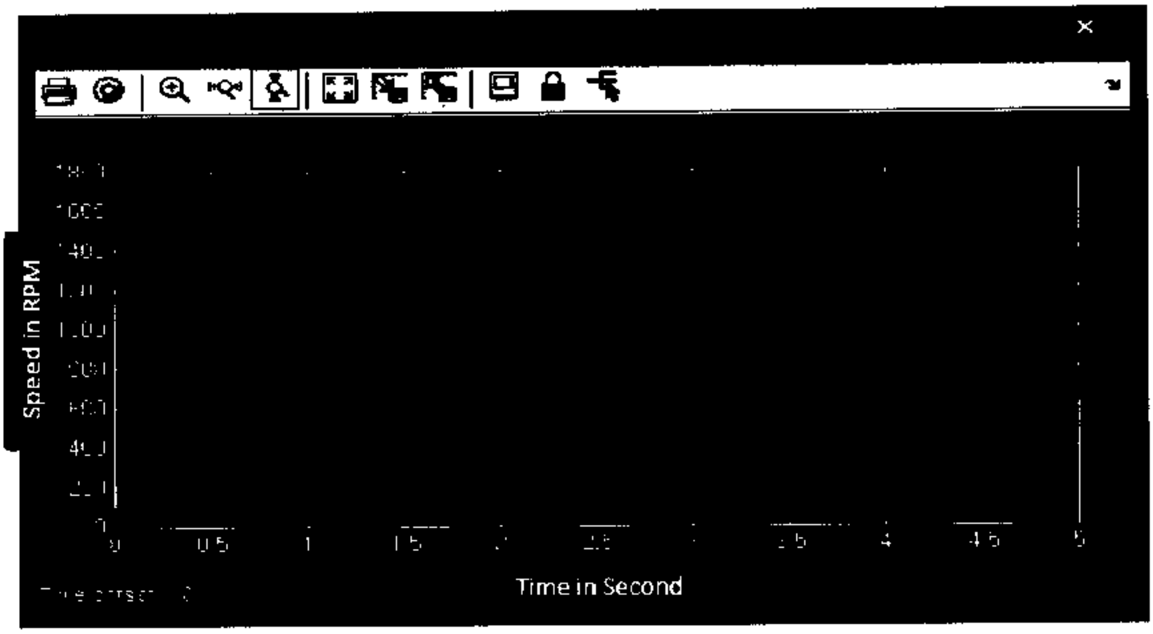

Fig-6.6: Response of System with ZN based IP controller

6.4 Comparatively System Response Analysis

The speed and speed error response of DC motor with and without controller are shown in the figure 6.7 and 6.8. In Figure 6.7 light blue and green color indicates the reference speed and open loop system response. Green color indicates that system response does not set to desire value at 1500 RPM but it set to 2700 RPM which is not desirable response. Red dotted line indicate the response of system with PI controller which shows that system has some overshoot but after some time set to the desire value. Blue line indicate the system response with IP controller which shows that the system response with IP controller has small overshoot as compared to PI controller. Thus the system performance can be improved using IP controller.

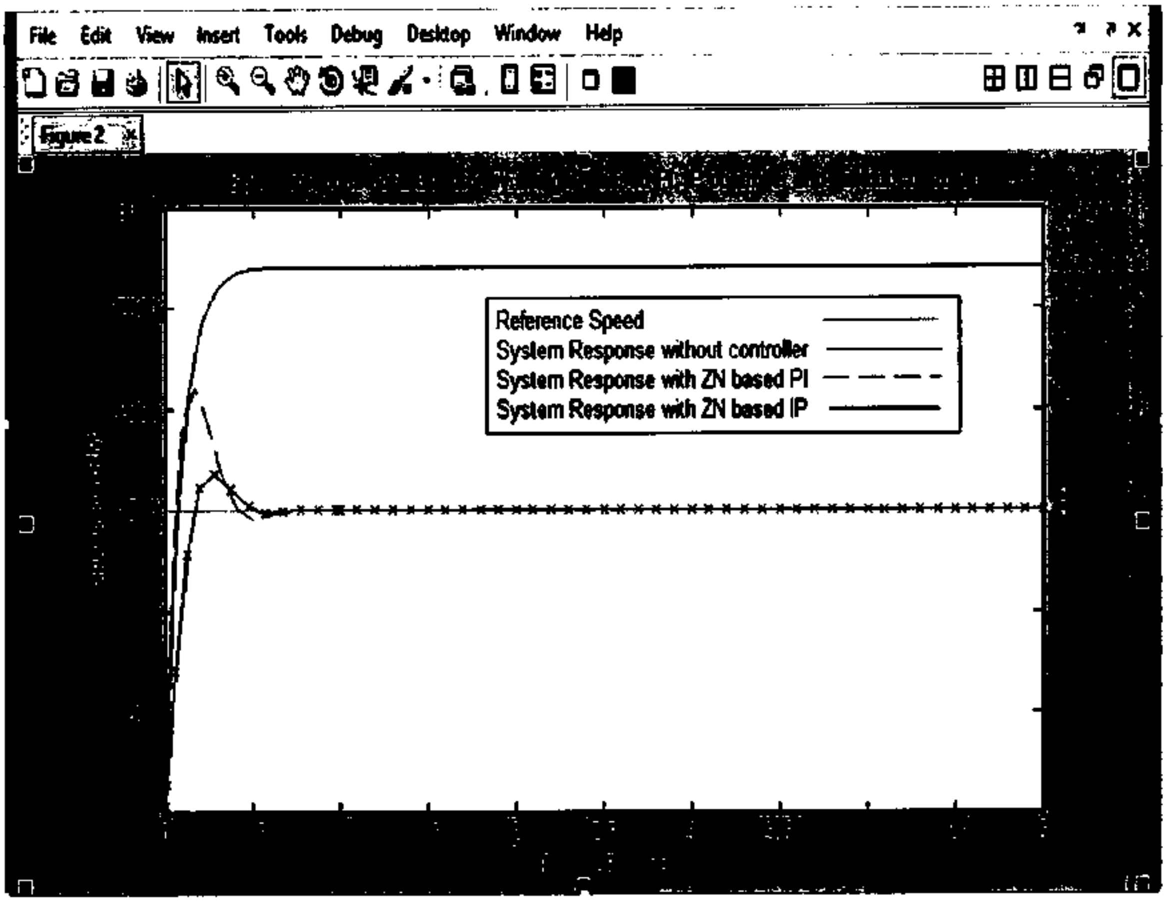

Fig-6.7: ZN Based speed response Analysis

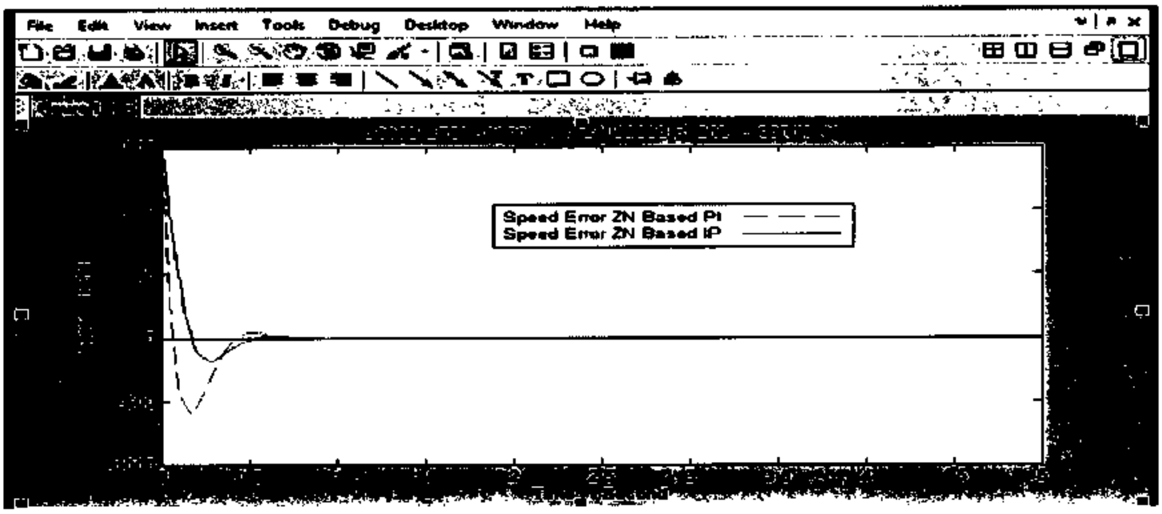

Fig-6.8: ZN based comparatively speed Error analysis of the system

6.5 System Transient Response with Ziegler Nicholas Based PI and IP Controller

Table 6.1 show the gain value and Transient response of speed control of DC motor in terms of overshoot, rise time and settling time using Ziegler Nicholas based PI and IP controllers.

Table 6.1: Gain value and Transient Response of Motor

Method	K_P	K_I	**Over shoot (%)**	**Settling time (Sec)**	**Rise time (Sec)**
System Response without controller			84.48	Infinity	0.057
ZN-PI	1.2	12	40.2%	0.373	0.0515
ZN-IP	1.2	12	11.8 %	0.404	0.13

From the comparison it is observed that ZN based IP controller gives 28.4% less overshoots as compared to PI. However it's settling and rise time are slightly higher than PI.

6.6 Genetic Algorithm Based PI Controller

Figure 6.9 and 6.10 show Simulink model and result of GA based PI controller. Figure 6.10 indicate that the GA based PI controller has negligible overshoots as compared to the Ziegler-Nicholas based PI controller.

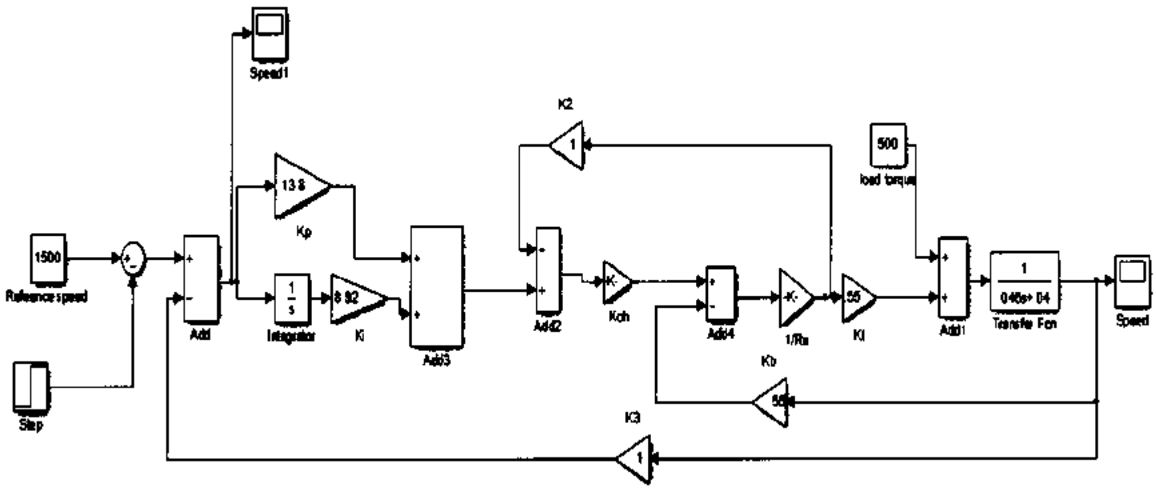

Fig-6.9: GA based PI Controller Simulink model

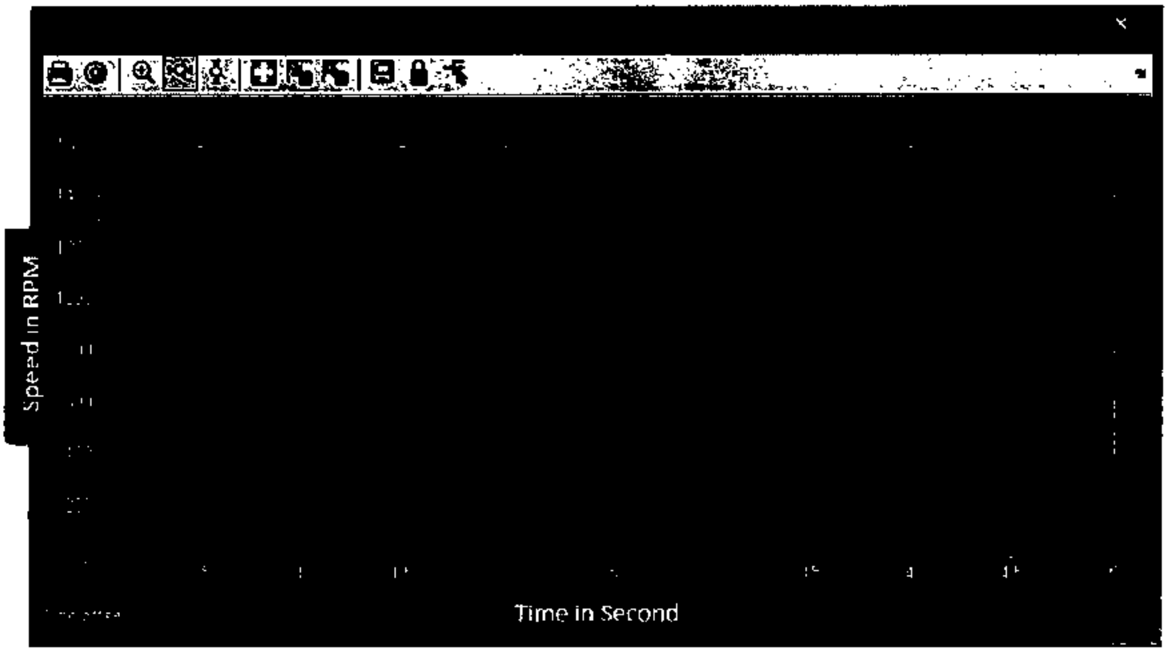

Fig-6.10: GA based PI Controller Simulation Result

6.7 Genetic Algorithm Based IP Controller

Figure 6.11 and 6.12 show Simulink model and result of GA based IP controller. Figure 6.12 show that the GA based IP controller provides zero overshoot as compared to the GA based PI controller.

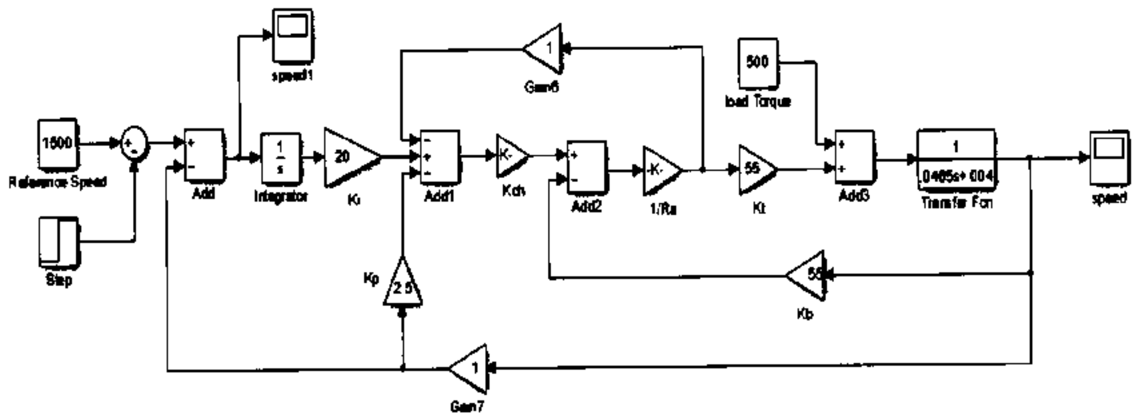

Fig-6.11: GA based IP Controller Simulink model

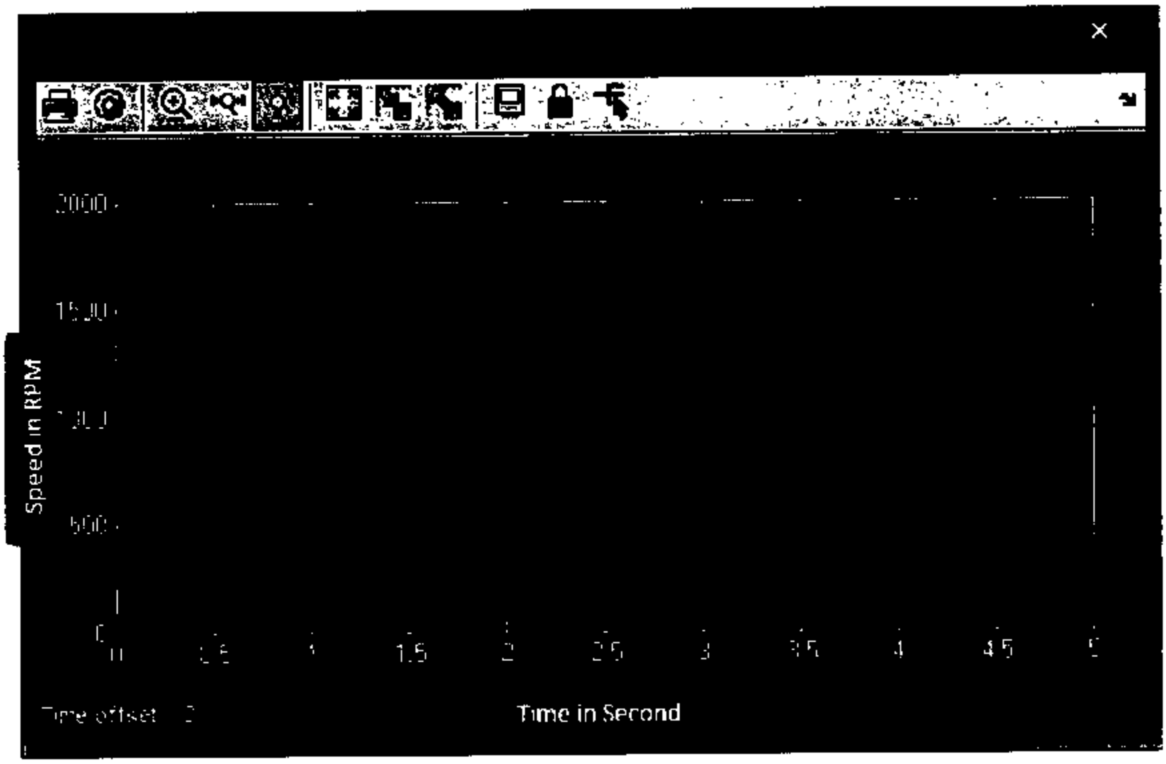

Fig-6.12: GA based IP Controller Simulation Result

6.6 Comparative Analysis of System Response using Genetic- Algorithm (GA)

Comparative analysis of speed control and speed error response of the system using GA based PI and IP controller are shown in the figures 6.13 and 6.14. In figure 6.13 light blue indicate the reference speed, green line indicate the system response without controller, red dotted line indicate the GA based PI controller and green line indicate the GA based IP controller response. From this comparative analysis it is concluded that for any system a controller is necessary to get the desired response. The system response can be improved by using the advance controller like IP controller. The response of the controller can also be improved by proper tuning. Figure 6.13 indicate that the response of the system using GA based PI and IP controller is much better that the response of the system using Ziegler-Nicholas based PI and IP controller which are shown in figure 6.7. The GA based PI has 38% less overshoot than the Ziegler-Nicholas based PI controller, also GA based IP controller has 11.8% less overshoots than the ZN based IP controller. The overshoots of IP controller either tuned by ZN or GA is smaller than the PI controller, however there is slightly high rise time and settling time in IP than PI controllers which can be ignore with respect to overshoots because overshoots are dangerous due to which our mechanical system may be damage. Figure 6.13 also show superiority of IP controller over PI controller.

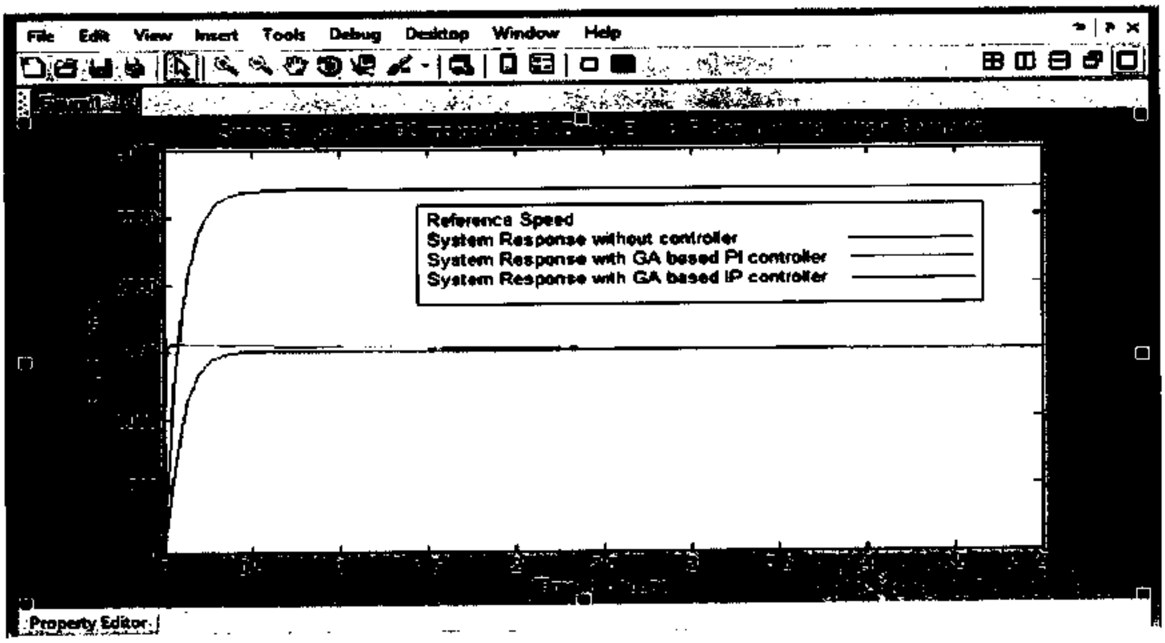

Fig-6.13: GA Based Speed Response Analysis

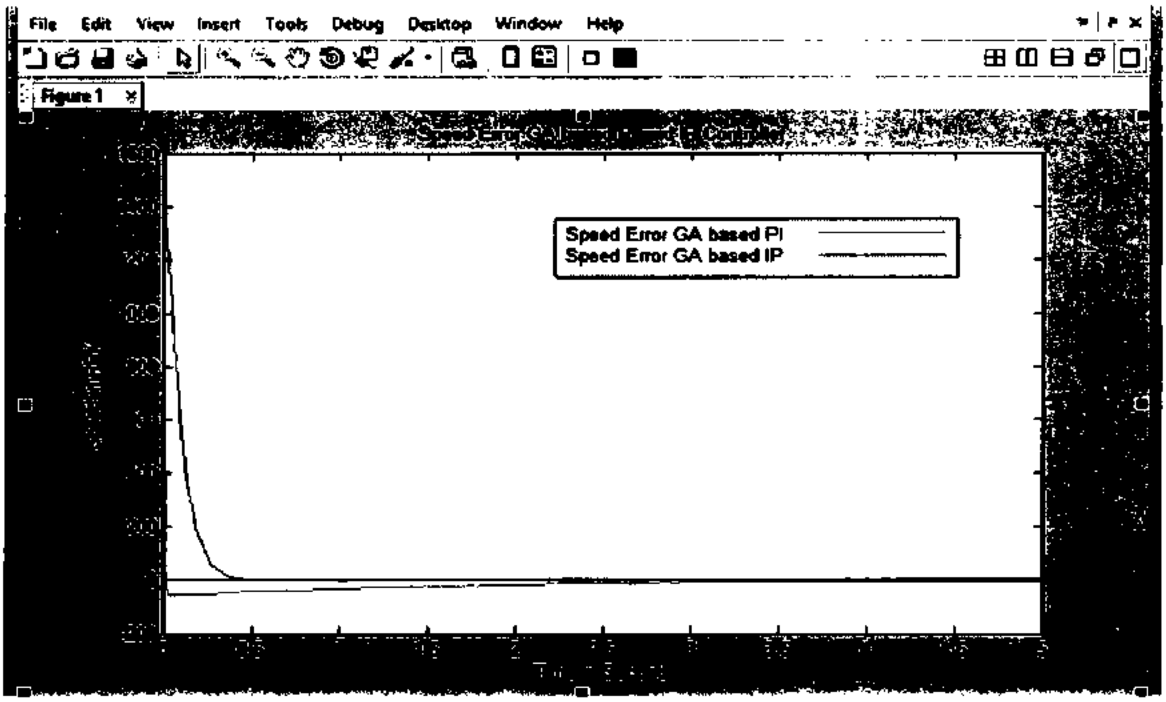

Fig-6.14: GA Based Speed Error Analysis

6.7 Transient Response of the System and Gain Value of Controller Using Genetic Algorithm

Table-6.2: GA based PI and IP Controller Gain values and system response

Method	K_P	K_I	Over shoot (%)	Settling time (Sec)	Rise time (Sec)
GA-PI	13.8	8.92	3.91	0.0156	0.0125
GA-IP	2.5	20	0	0.26	0.196

6.8 Simulated Annealing Based PI Controller

Figure 6.15 show the response of SA based PI controller which clearly indicate that system response with SA based PI controller is better than the Ziegler-Nicholas based PI controller which are shown in figure 6.4.

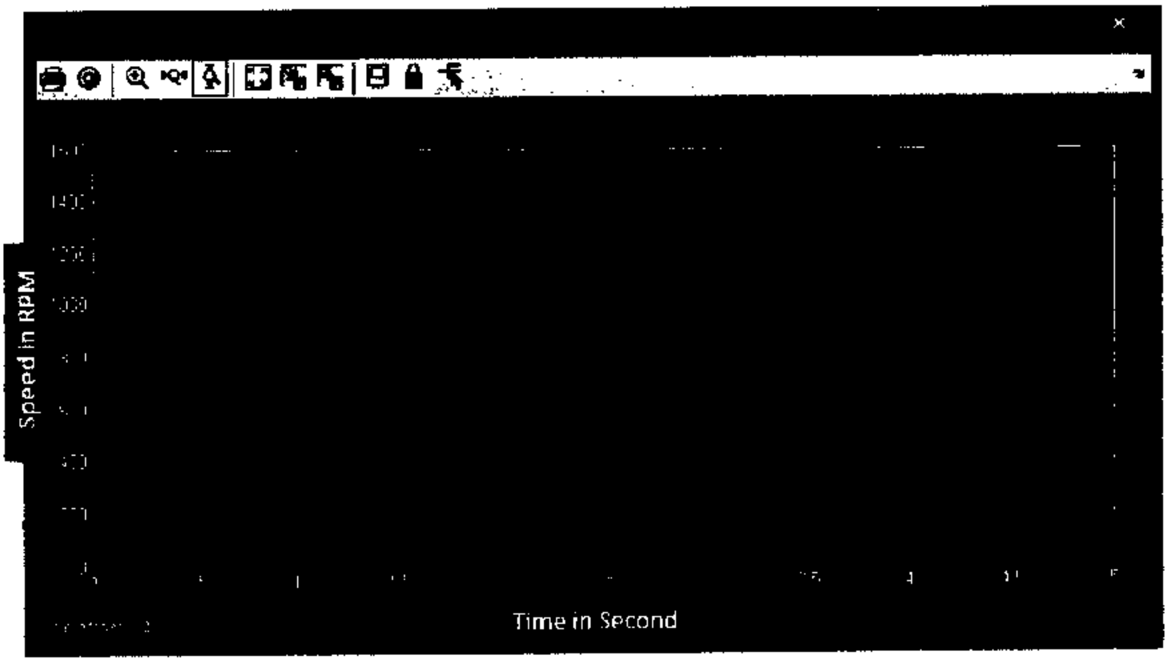

Fig-6.15: Simulated Annealing based PI controller result.

6.9 Simulated Annealing Based IP Controller

Figure 6.16 show response of the system with SA based IP controller which indicate that the overshoots which are present in the SA based PI controller can be removed in SA based IP controller.

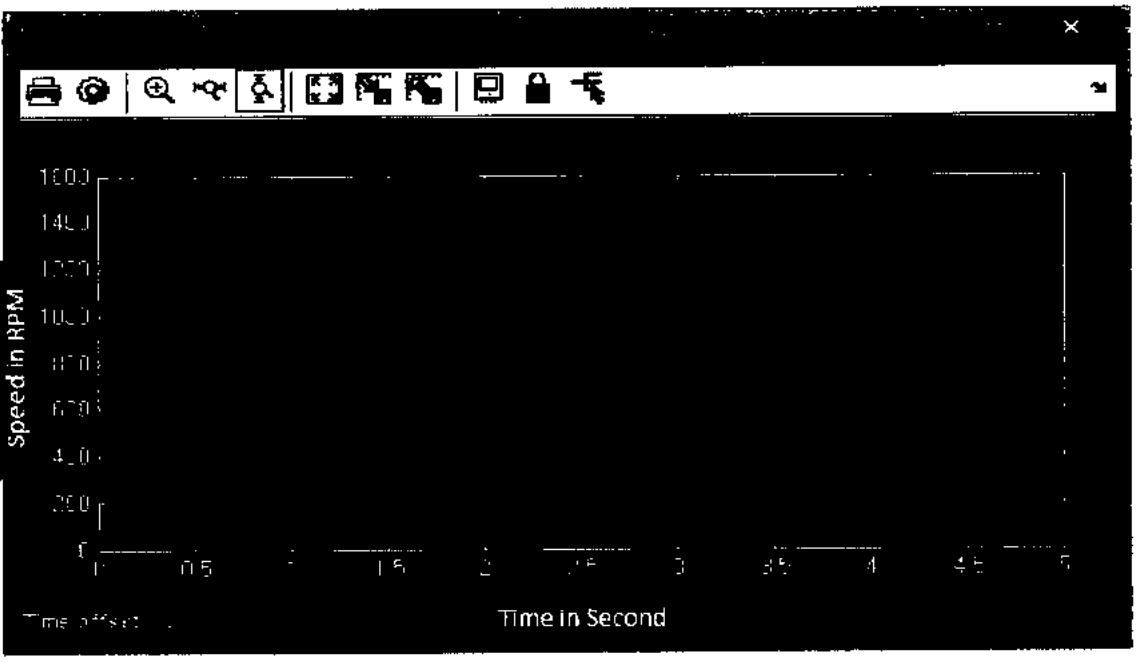

Fig-6.16: Simulated Annealing based IP controller result.

6.10 Comparative Analysis of System Response using Simulated Annealing (SA) algorithm

Figure 6.17 and 6.18 shows Comparative analysis of system speed response and speed error using SA based PI and IP controllers. In figure 6.17 light blue line indicate the reference speed, green line indicate the system response without controller, red line indicate the SA based PI controller and green line indicate the SA based IP controller response. From the analysis we observe that the small overshoot which are present in PI controller can be removed in IP controller, which indicate that the system response can be improved using IP controller. The system response of the system can be improved by proper tuning of a controller. Figure 6.17 and 6.7 show that the response of the system using SA based PI and IP controllers are better than the Ziegler-Nicholas based PI and IP controllers, the SA based PI controller has 37.7% less overshoots than the ZN based PI controller and the SA based IP controller has 11.8% less overshoots than the ZN based IP controller. The speed error response of SA based PI and IP controllers are also smaller than ZN based PI and IP controller. The speed error of IP controller is less as compared to PI controller. The overall response of the system using IP controller is better than the PI controllers.

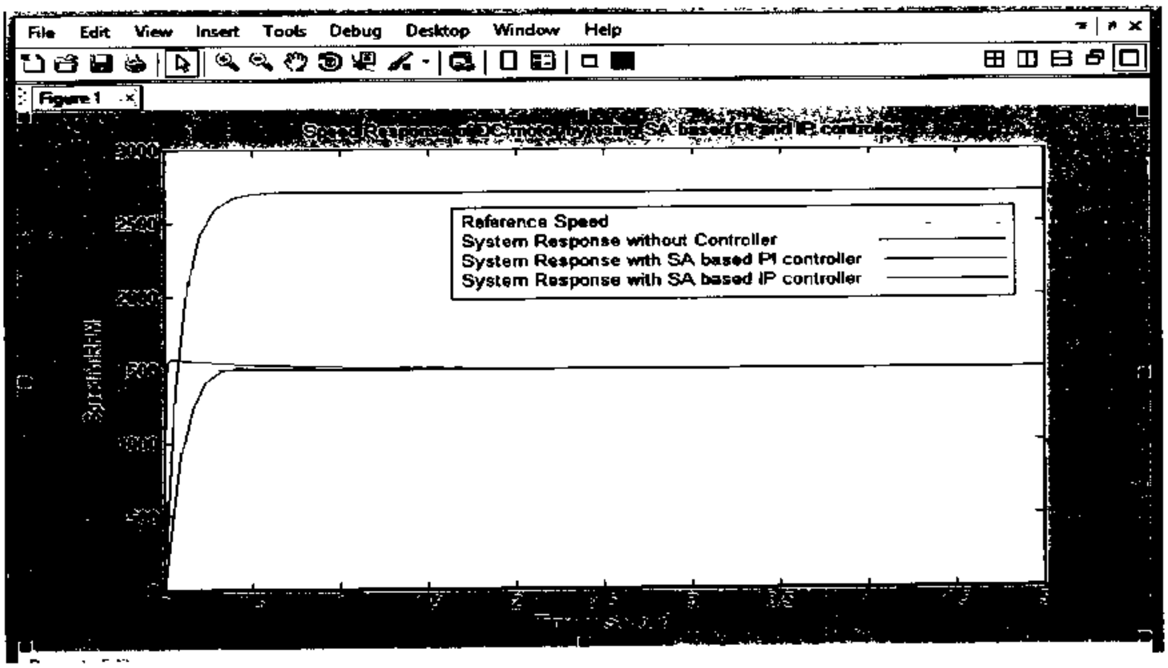

Fig-6.17: SA Based Speed Response Analysis

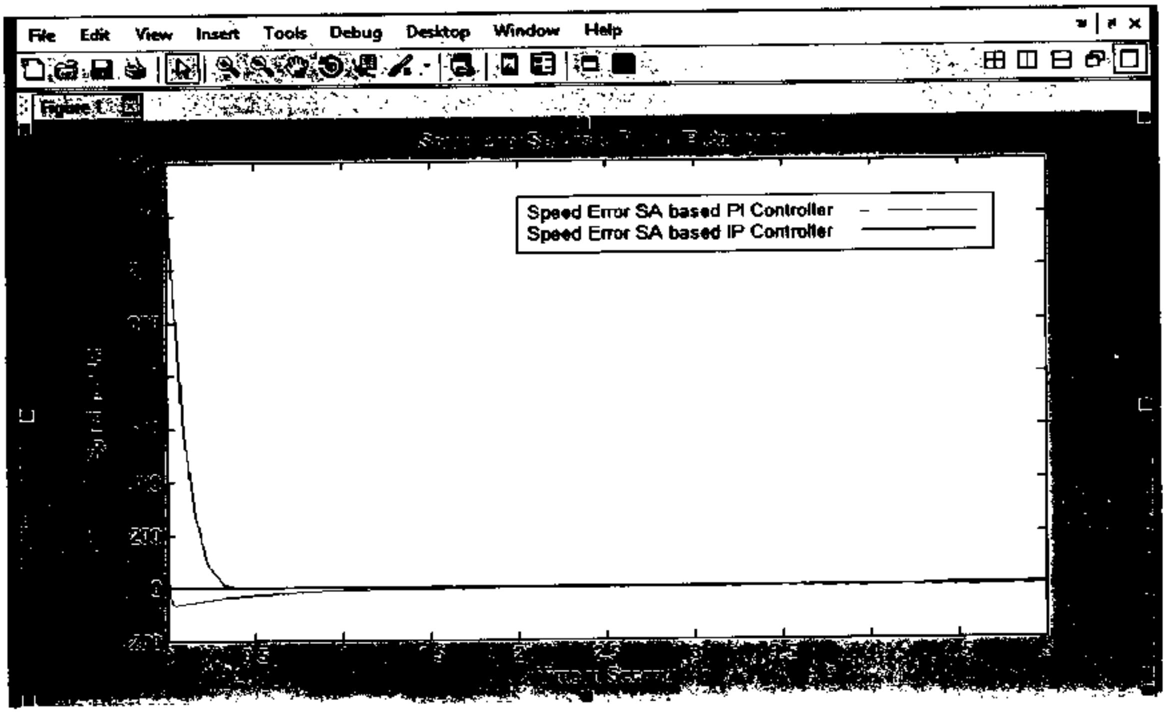

Fig-6.18: SA Based Speed Error Analysis

6.11 Transient Response of the System and Gain Value of Controller Using Simulated Annealing

Table-6.3: Simulated Annealing based PI and IP Controller Gain value and system response

Method	K_P	K_I	Over shoot (%)	Settling time (Sec)	Rise time (Sec)
SA-PI	13.93	23.8	4.4	0.0158	0.0128
SA-IP	2	15	0	0.27	0.198

6.12 Comparative Analysis of System Response using ZN, SA and GA algorithm

Figure 6.19 show the speed response analysis of PI and IP controller using ZN, SA and GA for tuning of controllers. From figure 6.19 we see that the transient response of the IP controller is better than the transient response of the PI controller either tuned by ZN, SA and GA. Also the transient response of the system with PI and IP controller using soft computing techniques like GA and SA for tuning of controllers is better than the ZN tuning method.

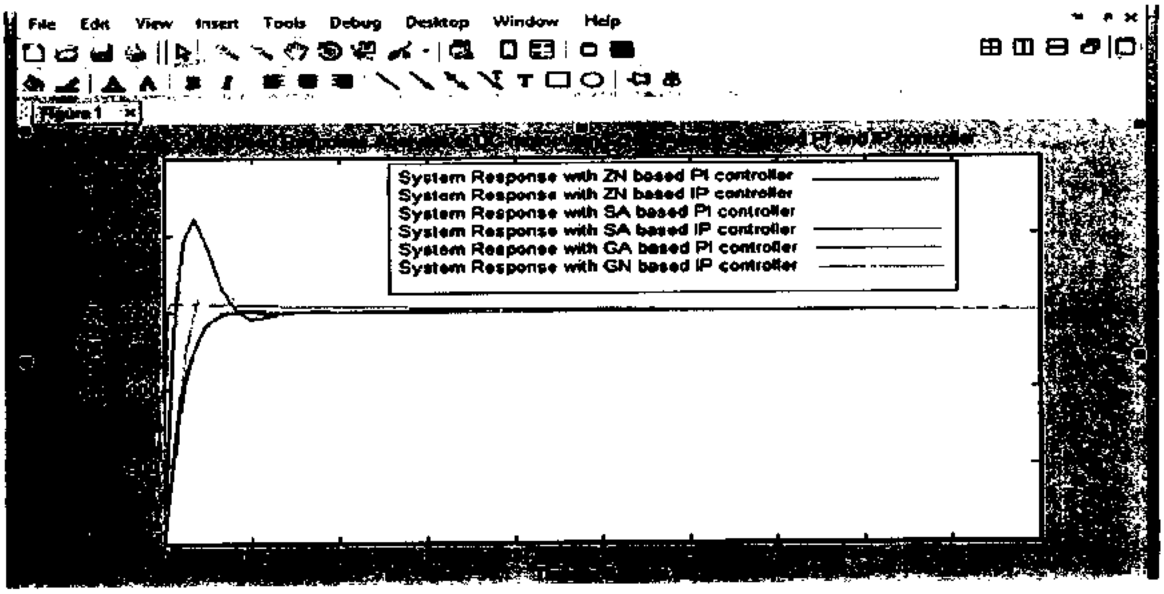

Fig-6.19: ZN, SA and GA Based Speed Response Analysis

6.12 Transient Response Analysis of the System

The transient response of the speed control of DC motor using PI and IP controller tuned by ZN, GA and SA are shown in the table 6.4. From the table 6.4 we can analyze that system response by using GA optimization techniques is better than the SA and Ziegler-Nicholas tuning method, the response of the system using SA algorithm is better than Ziegler-Nicholas tuning method. The GA based PI has 38% less overshoot than the Ziegler-Nicholas based PI controller, also GA based IP controller has 11.8% less overshoots than the ZN based IP controller. The SA based PI controller has 37.7% less overshoots than the ZN based PI controller and the SA based IP controller has 11.8% less overshoots than the ZN based IP controller. The overshoots of IP controller either tuned by ZN, GA or SA are smaller than the PI controller, however there is slightly high rise time and settling time in IP than PI controllers which can be ignore with respect to overshoots because overshoots are more dangerous due to which our mechanical system may be damage. From table 6.4 we see that the transient response of DC motor using ZN, SA and GA based PI and IP controller is much better than the response of [32, 34].