SOFT COMPUTING TECHNIQUES TO CONTROL MAGNETIC LEVITATION SYSTEM

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree

of

MASTER OF TECHNOLOGY

in

ELECTRONICS AND COMPUTER ENGINEERING
(With Specialization in Control and Guidance)

By

VAIBHAV GUPTA

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE-247 667 (INDIA)

JUNE, 2010

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this dissertation report entitled, "**Soft computing techniques to control Magnetic Levitation System**" towards the partial fulfillment of the requirements for the award of degree of **Master of Technology in Electronics and Communication Engineering** with specialization in **Control and Guidance**, submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, is an authentic record of my own work carried out during the period from July 2009 to June 2010, under the guidance of **Dr. Vijay Kumar, Associate Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee.**

The content of this dissertation has not been previously submitted for examination as part of any academic qualifications.

Date: 29/06/2010

Place: Roorkee

Vibhav

VAIBHAV GUPTA

CERTIFICATE

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

Date: 29-06-10

Place: Roorkee

Dr. VIJAY KUMAR

Associate Professor, E&C Department,

Indian Institute of Technology Roorkee,

Roorkee-247667

(India)

ACKNOWLEDGEMENTS

With great sense of pleasure and privilege, I take this opportunity to express my deepest sense of gratitude towards my supervisor and guide; **Dr. Vijay Kumar** for his valuable suggestions, sagacious guidance, scholarly advice and insightful comments and constructive suggestions to improve the quality of the present work. His professionalism, accurate advice, suggestions and his ways of thinking inspired me, and this inspiration guides me in every moment of my life.

I would like to thank all my friends especially Ravichandra, Sajan and Manju ma'am for their help in successful completion of my thesis. I would also like to thank the Lab staff of Control Sytems Lab for their continuous support and well maintained Lab.

Here, I would like to Thank persons, whose presence gives me courage and motivation to keep working. Finally, I would like to extend my gratitude to all those persons who directly or indirectly contributed towards this work.

VAIBHAV GUPTA

ABSTRACT

Magnetic levitation technology has been receiving increasing attention because it helps eliminate frictional losses due to mechanical contact. A Magnetic Levitation System (MLS) is considered as a good test-bed for the design and analysis of control systems since it is a nonlinear unstable plant. Controller design for such unstable system involves optimizing various parameters, such as settling time, peak overshoot etc. Since the system is very prone to disturbances, design must ensure robustness and disturbance rejection performance.

The aim of the thesis is to develop various controlling schemes, which deal with the uncertainties of the system, maintaining optimum time specifications. In order to achieve this, first the System model has been designed. With understanding of system performance, initially the use of conventional methods to control system dynamics employed. Later, Improvement in controlling techniques is performed. It involved employing Proportional-Integral-Derivative (PID) control, followed by application of soft computing techniques including fuzzy logic, genetic algorithms, neuro-computing and particle swarm optimization(PSO).

Firstly, tuning capabilities of genetic algorithm and particle swarm optimization technique for a PID controller are explored. Secondly, fuzzy logic systems are developed based on behavior of system, which is combined with tuning of scaling parameters with genetic algorithm. These applications are analyzed on magnetic levitation system and results showed the capability of proposed algorithms.

CONTENTS

Chapter 1

Introduction

In the industrial domain, magnetic levitation has been successfully implemented for many applications. We can mention, for example, high-speed train suspension in Japan and Germany, vibration isolation systems, magnetic bearings, rocket-guiding projects.[1]–[3].

Due to the features of the open-loop instability and inherent nonlinearities in electromechanical dynamics of the magnetic levitation apparatuses, the development of the high-performance control design for the position control of the levitated object is very important.

Despite the fact that magnetic levitation systems have unstable behavior and are described by highly nonlinear differential equations, most design approaches are based on the linearized model about a nominal operating point. In this case, the tracking performance deteriorates rapidly with increasing deviations from the nominal operating point. However, in order to ensure very long ranges of travel and still obtain good tracking, it is necessary to consider a nonlinear model rather that a linear one. In addition, the plant parameter changes, such as the change of suspending mass and the variations of resistance and inductance due to electromagnet heating, should be taken into consideration.[4]

Control system design must take into account a number of performance issues, such as system stability, the static and dynamic index, and system robustness. Each of these issues strongly depends on the controller structure and parameters. However, this dependence usually cannot be expressed in a mathematical formula. Additionally, often a trade-off has to be made among conflicting performance issues.[5]

Obviously the lack of a systematic and intuitive approach to select values for a large number of control parameters is a big obstacle when attempting to obtain a satisfactory control system. In this Disseratation, various controlling schemes are implemented on Magnetic leviataion system optimizing various performance Index and a comparison is made between various schemes.

1.1 Challenges in the Magnetic Leviatation systems

Due to wide application fields of Magnetic Levitation Systems, it has been the subject of many years research, and continues to attract much attention. The main challenges in the control of Magnetic Levitation Systems are the complexity of the system and disturbances. Following properties make system difficult to control :

- The highly nonlinear system.
- Accurate control required over different operating conditions.
- Open-loop instability

1.2 Brief review of literature survey

Robustness and Disturbance rejection performance are two important performance criterion for any control system and important parameters to compare between various controlling schemes. Conventional control deals with development of mathematical modeling of the system first, than employing controlling methods.

As known, conventional control system which relies on the mathematical model of the underlying system has been successfully implemented to various simple and non-linear control systems. However, it has not been widely used with complicated, non-linear and time varying systems.[6]

Since Magnetic levitation system is a highly nonlinear system, controller implemented using conventional controlling techniques are not able to give satisfactory performance, because it is difficult to describe properly all their nonlinearities.

The proportional-integral-derivative (PID) controllers were the most popular controllers of this century because of their remarkable effectiveness, simplicity of implementation and broad applicability. However, PID controllers are poorly tuned in practice with most of the tuning done manually which is difficult and time consuming. The computational intelligence has purposed genetic algorithms (GA) and particle swarm optimization (PSO) as opened paths to a new generation of advanced process control[7]. These advanced techniques to design industrial control systems are, in general, dependent on achieving *optimum performance* with the controller when facing with various types of disturbance that are unknown in most practical applications.

1.3 Need for soft computing

Soft Computing is a field, which is characterized by the use of inexact solutions to computationally-hard tasks. Modern systems are becoming increasingly more complex, making conventional control algorithms that utilize mathematical models insufficient.[7]-[8] This resulted in the evolution of soft computing techniques based on biological processes such as learning and evolutionary development. Soft computing differs from conventional (hard) computing in that, unlike hard computing, it is tolerant of imprecision, uncertainty, partial truth, and approximation. In effect, the role model for soft computing is the human mind.

The guiding principle of soft computing is: Exploit the tolerance for imprecision, uncertainty, partial truth, and approximation to achieve tractability, robustness and low solution cost. At this juncture, the principal constituents of Soft Computing (SC) are Fuzzy Logic (FL), Neural Computing (NC), Evolutionary Computation (EC) Machine Learning (ML) and Probabilistic Reasoning (PR), with the latter subsuming belief networks, chaos theory and parts of learning theory. What is important to note is that soft computing is not a mélange. Rather, it is a partnership in which each of the partners contributes a distinct methodology for addressing problems in its domain. In this perspective, the principal constituent methodologies in SC are complementary rather than competitive. Furthermore, soft computing may be viewed as a foundation component for the emerging field of conceptual intelligence.

1.4 Thesis Organization

The entire work carried out has been presented in six chapters. Each chapter begins with brief introduction pertaining to the concerned problem and the motivation behind the study. Emphasis has been given to the control algorithm design methodology. The results are summarized and discussed at the end of each problem that is attempted.

Chapter 1 - The challenges in the Magnetic Levitation system control problem and literature survey and brief organization.

Chapter 2 – Magnetic Leviation System modeling, and the stability analysis in both time and frequency domain.

Chapter 3 - Soft computing techniques and about Particle swarm optimization.

Chapter 4 – Tuning strategies of PID controller. Also, results of tuning using GA and PSO are discussed here.

Chapter 5 – Introduction to Fuzzy logic and ANFIS control.

Chapter 6 - deals with the conclusion and future scope of the work carried out in the dissertation.

Chapter 2

Magnetic Levitation System

2.1 Introduction

Magnetic levitation is a method by which an object is suspended with no support other than magnetic fields. The electromagnetic force is used to balance the gravitational force.[9]

Magnetic Levitation system is based on a principle of Electromagnetic Induction in current carrying coil to levitate objects in air. By using sensors for observing position of Levitated objects and then controlling amount of current to be given to Electromagnet, an object can be held into air.

2.2 System Block Diagram

The System Configuration can be best understood by following Figure: [10]

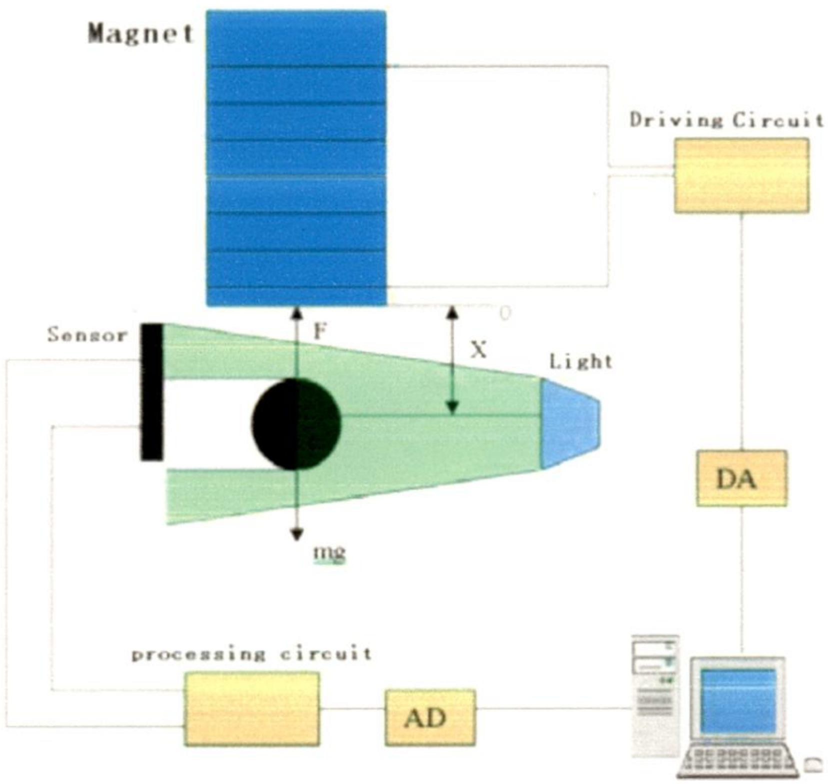

Fig.2.2.1 MLS block diagram

MLS is a platform for the study of magnetic levitation technology. It's mainly composed of solenoid, position sensor, amplifier/compensation device (driver), digital controller (PCI1711) and control object ***steel ball***. It is a typical magnetized levitation system. The system can be divided into two parts, MLS main body, control platform with data acquisition card and PC. System block diagram is shown in Figure 2.2.2 [10]

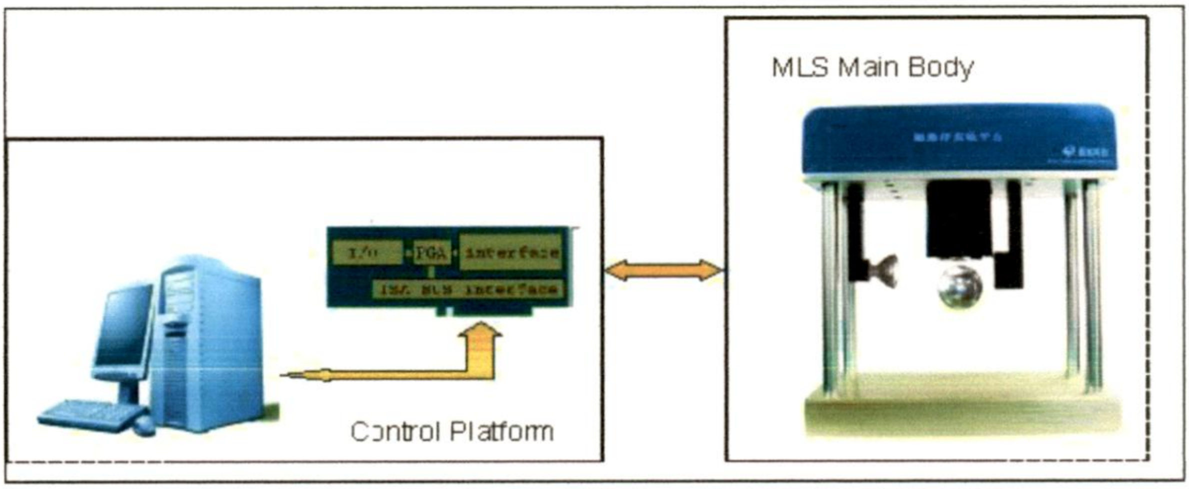

Fig.2.2.2 MLS Experimental setup

Components of System

Magnetic Levitation System main body components:

- Driver
- Solenoid (Magnet)
- Sensor
- LED
- Steel ball

Control Platform

- PC compatible with IBM PC/AT, with PCI slot.
- PCI1711 data acquisition card and its driver
- Demo experimental software.

2.3 Principle and Working

Magnetic Levitation System works on principle of Electromagnetic Induction to control position of ball at required position. When current go through the winding, electromagnetic force F will be generated. By controlling the current in the electromagnet winding to balance the steel ball gravity force *mg* by magnetic force, the steel ball will be levitated in the air. Closed loop control is required for the stability and anti-interference. The distance x from the steel ball to electric magnet is detected by sensor system composed of light source and light sensor. To enhance the performance, the speed of the distance variance can also be considered. The control current is the input for magnetic levitation control object.

2.4 System Modeling

Before system modeling, we have the following assumptions:[10]

1. Assume all the magnetic flux go through the magnetic pore outside air gap, ignore the magnetic flux leak;

2. Magnetic flux is even in the air gap, no edge effect;

3. Ignore the magnetism reluctance of the ball and electromagnet. Then the reluctant is mainly in the air gap between the ball and the electromagnet.

4. Assume all the magnetic force is focused in the mass center of the ball, which is also the ball center.

Physical Model of the Magnetic Levitation System

Differential Equation Inference: This system mathematic model is established based on the ball kinematics, electrical and dynamics equations. Therefore the system model is established from 2 aspects.

Dynamic Equation of the Control Object

Assume the ball is not disturbed by external forces, then it is only affected by the magnetic force F and gravity force mg. The dynamic equation in vertical direction can be described as:

$$m \frac{d^2x(t)}{dt^2} = F(i,x) + mg \tag{2.4.1}$$

x - the air gap between ball centroid and magnetic pore, unit: m;

m - steal ball mass, unit: Kg

F(i,x) - magnetic force, unit: N

g - acceleration of gravity, unit: m/s^2

System Electromagnetic Force Model

Electromagnetic Force can be modeled using following relation:

$$F(i,x) = K\left(\frac{i}{x}\right)^2 \tag{2.4.2}$$

K - constant depending on physical parameters

x - the air gap between ball centroid and magnetic pore, unit: m;

i - current in Electromagnet

By equation (2.4.2), the electromagnetic force $F(i,x)$ is nonlinear negative proportional with the air gap(x), that's the reason why the Magnetic levitation system is unstable.

The Control Voltage and Current Model in Electromagnet

$$U(t) = Ri(t) + L_1\frac{di}{dt} \tag{2.4.3}$$

Where,

L_1 - the static inductance when ball is in the magnetic field;

R - winding Resistance

i - current in electromagnet

U(t) - coil voltage

The Boundary Condition of System Equilibrium

When the ball is in equilibrium point, the acceleration is 0, from Newton's law the composition of force on the ball is 0. The magnetic force is equal to the gravity force of the ball:

$$mg + F(i_0, x_0) = 0 \tag{2.4.4}$$

System Equation Description: The ML system equation can be obtained from above equations:

$$m\,\frac{d^2x(t)}{dt^2} = F(i,x) + mg$$

$$F(i,x) = K\left(\frac{i}{x}\right)^2 \tag{2.4.5}$$

$$U(t) = Ri(t) + L_1\frac{di}{dt}$$

$$mg + F(i_0, x_0) = 0$$

System Model Linearization: This is a typical nonlinear system. If we are going to solve the system by linear theory, it should be linearized first.

The magnetic force *F*, instantaneous current i in the coil and air gap *x* are nonlinear related. For controller design using linear theory the system should be linearized first. The system control is in certain range, therefore the linearization is possible. It is also indicated in the experiment that the system can be linearized at equilibrium point.

Take Tylor's expansion of equation (2.4.2), ignore higher order terms, there is:

$$F(i,x) = F(i_0, x_0) + F_i(i_0, x_0)(i - i_0) + F_x(i_0, x_0)(x - x_0) \tag{2.4.6}$$

In which $F(i_0,\ x_0)$ is the magnetic force equal to the ball gravity force when the air gap is x_0 and the balancing current is i_0. That is:

$$F(i_0, x_0) = mg$$

$$F_i(i_0, x_0) = \frac{\delta F(i,x)}{\delta i}\Big|_{(i=i_0, x=x_0)} \tag{2.4.7}$$

$$F_x(i_0, x_0) = \frac{\delta F(i,x)}{\delta x}\Big|_{(i=i_0, x=x_0)}$$

Now, on assuming,

$$K_i = F_i(i_0, x_0) = \frac{2Ki_0}{x_0^2}$$

$$K_x = F_x(i_0, x_0) = -\frac{2Ki_0^2}{x_0^3} \qquad (2.4.8)$$

Where,

K_i = stiffness coefficient of the magnetic force to current at the equilibrium point.

K_x = stiffness coefficient of the magnetic force to air gap at the equilibrium point.

The equation for the whole system is:

$$m\,\frac{d^2x}{dt^2} = K_i(i - i_0) + K_x(x - x_0) \qquad (2.4.9)$$

System Control Modeling: In system modeling, the input is control current of the electromagnet, the influence of inductance is not considered. Assume the power amplifier output current is strictly linear with input voltage without delay.

After Laplace transform, the system can be described by following equation:

$$s^2X(s) = \frac{2Ki_0}{mx_0^2}I(s) - \frac{2Ki_0^2}{mx_0^3}X(s)$$

$$\frac{X(s)}{I(s)} = \frac{-1}{As^2 - B} \qquad (2.4.10)$$

Define the input variable U_{in} as the input voltage of the power amplifier, output variable U_{out} as the output voltage reflecting (the voltage output of the process circuit at the back of the sensor), the system control object model can be expressed as:

$$G(s) = \frac{U_{out}(s)}{U_{in}(s)} = \frac{K_s\,X(s)}{K_a\,I(s)} = \frac{-(\frac{K_s}{K_a})}{As^2 - B}$$

$$A = \frac{i_0}{2g}, B = \frac{i_0}{x_0} \qquad (2.4.11)$$

The open loop system characteristic equation is: $As^2 - B = 0$

The system open loop pole is:

$$s = \pm\sqrt{\frac{B}{A}} = \pm\sqrt{\frac{2g}{x_0}} \tag{2.4.12}$$

There is an open loop pole at the right plane, by stability criterion, stable system should have all the open loop poles on the left plane. Therefore the ML system is essentially unstable.

The system state variables are $x_1 = u_{out}, x_2 = \dot{u_{out}}$ and the system state equations are as follow:

$$\begin{pmatrix} \dot{x_1} \\ \dot{x_2} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \frac{2g}{x_0} & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ -\frac{2g.K_s}{i_0.K_a} \end{pmatrix} u_{in}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = x_1 \tag{2.4.13}$$

2.5 Real System Model

Put the parameters in equation: [10]

$$\begin{pmatrix} \dot{x_1} \\ \dot{x_2} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 980.0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ 2499.1 \end{pmatrix} u_{in} \tag{2.4.14}$$

The transfer function of u_{in} -Y is:

$$G_0(s) = \frac{Y(s)}{U_{in}(s)} = C^T(sI - A)^{-1}B \tag{2.4.15}$$

Then we have: [10]

$$G_0(s) = \frac{77.8421}{0.0311s^2 - 30.5250} \tag{2.4.16}$$

2.6 Magnetic Levitation System Properties

Though magnetic levitation system has different application with variety of structure and configuration, they have the same basic properties:

a) Nonlinearity

Magnetic levitation is a typical nonlinear complex system. The approximate model can be obtained through linearization. It can also be controlled by nonlinear control theory.

b) Uncertainty

The uncertainty mainly is modeling error, electromagnetism interfere and other outside disturbances. In the experiment, uncertainty can be reduced by adding lens hood, enhancing the background light, or reducing the electromagnet temperature, etc.

c) Open loop instability

There is only one steady state of the magnetic levitation system, which is, when the electromagnetic force balance the gravity force of the levitate object. When the system is open loop, slight disturbance will change the steady state.

Above properties increase the control difficulty of the magnetic levitation system and adding research value for users.

2.7 Magnetic Levitation Technology Application Field

- Magnetic levitation vehicle
- Magnetic levitation bearing
- High speed magnetic levitation motor
- Other area of the magnetic levitation applications
- Wind tunnel magnetic levitation system
- Magnetic levitation anti-vibration system
- Magnetic levitation fusion

2.8 Analysis of Magnetic levitation system

System Step Response Analysis

System step response is obtained by applying step input to transfer function derived above.The following result is obtained:

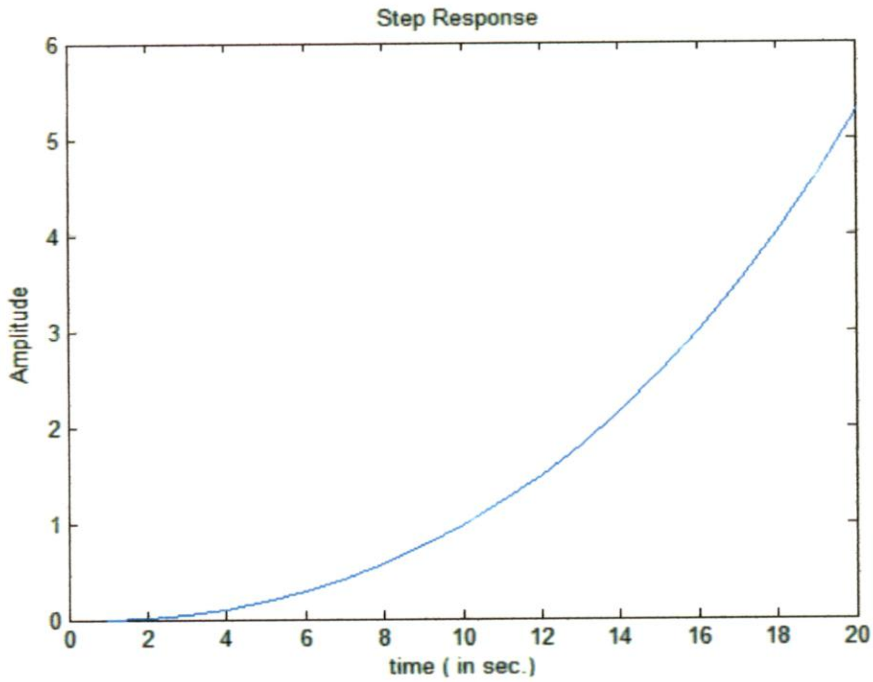

Fig.2.8.1 System Step Response

From above figure, the position of the ball diverges quickly. The open system is a second order unstable system. To stabilize the levitated object, the current in the electric magnet should be changed to prevent the object from leaving the equilibrium point. Therefore, a feedback controller is necessary.

2.8.1 Root Locus Analysis of System

The closed loop system transient response property is closely related to system closed loop poles. If system gain is adjustable, the placement of closed loop pole will depends on the selected system gain. From design point of view, some system could easily place the pole to the required position by adjusting the system gain. However, when simply changing the system gain cannot satisfy the system performance requirement, controller design become important.

Some recently most popular used controller design methods include phase-lead, phase-lag and lead-lag compensator, etc.

The real system open loop transfer function is:

$$G_0(s) = \frac{77.8421}{0.0311s^2 - 30.5250} \tag{2.8.1}$$

We can see that, there are 2 system poles, one is positive, one negative.

p1 = 31.3291 , p2 = -31.3291

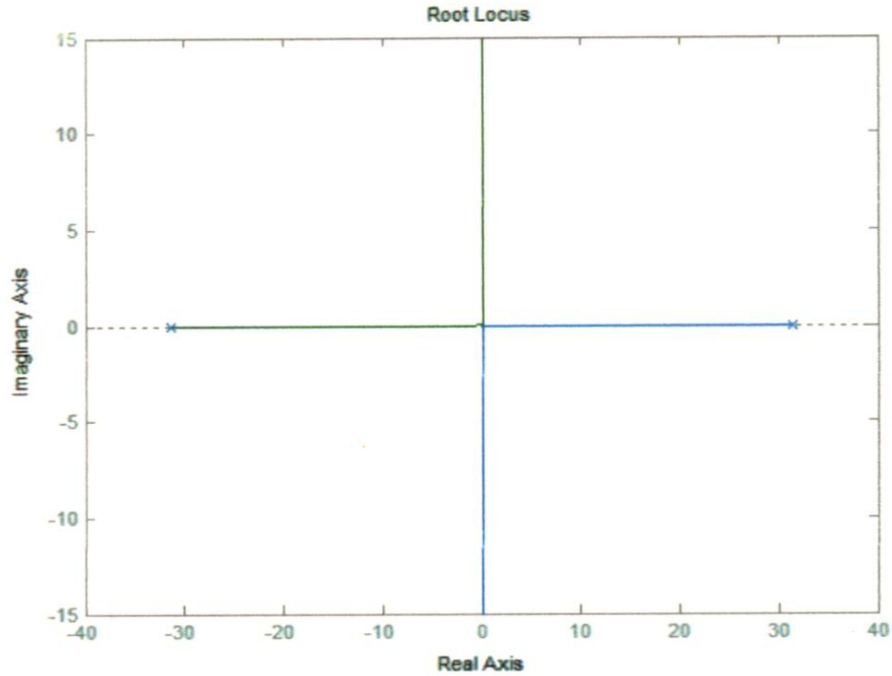

Fig. 2.8.1.1 Open-loop system root locus

There is a Open loop pole on the right half plane. One root locus starts from this point towards left along the real axis and ends at zero. That means however the systems gain change; the locus will stay on the right half plane. The system is always unstable.

Root Locus Design and Simulation

Magnetic Levitation system root locus method can be formulated as the following problem:

Design a controller having following specifications:

- Response time: $t_s = 0.2$ sec (2%)

- Maximal overshoot: $m_p \leq 10\%$

From above specifications, following controller has been designed for the system.

$$G(s) = \frac{0.991112(s+23.424)}{s+48.8648} \tag{2.8.2}$$

Giving the Step response

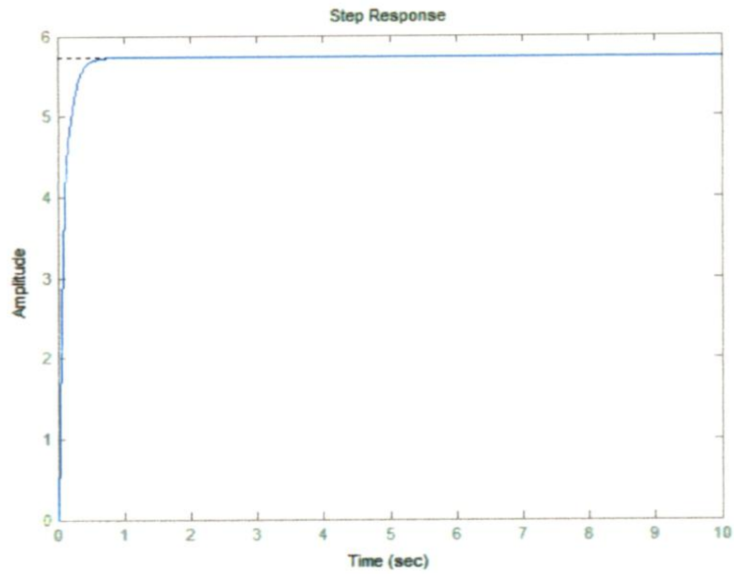

Fig. 2.8.1.2 System Step Response

The compensated system root locus plot is as follow:

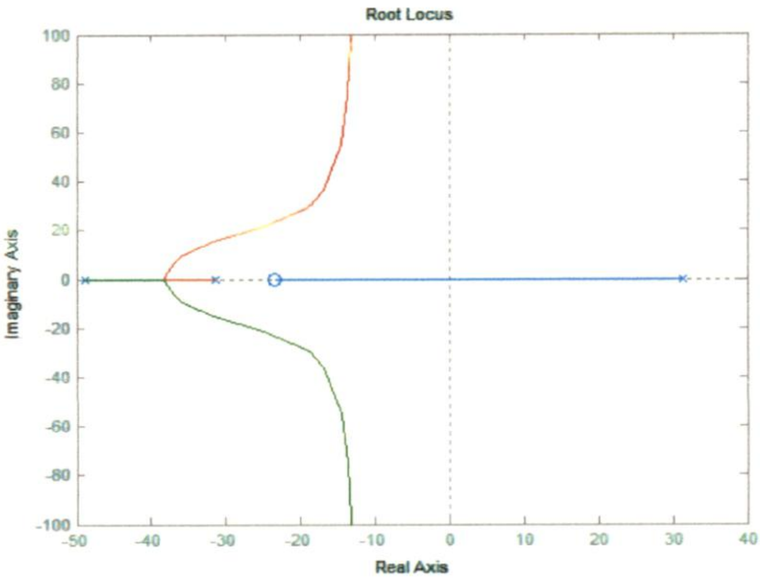

Fig.2.8.1.3 Improved root locus

We can see from the figure, root locus plots are on the left half plane. The system can be stabilized by proper choice of K.

2.8.2 Frequency Response Analysis

The System response to sinusoidal signal is called frequency response. In frequency response method, we change the input signal frequency in certain range to study the system response.

The polar coordinates plot which gives the locus of vector G(jw) /_G(jw) , when ω varies from 0 to infinity. The polar coordinates plot is also called Nyquist plot. The Nyquist stability criterion helps us decide the close loop system absolute stability and relative stability according to the open loop frequency response property information.

Bode plot, which use two separate figure: one plots the relation between magnitude and frequency and another plots the relation between phase and frequency.

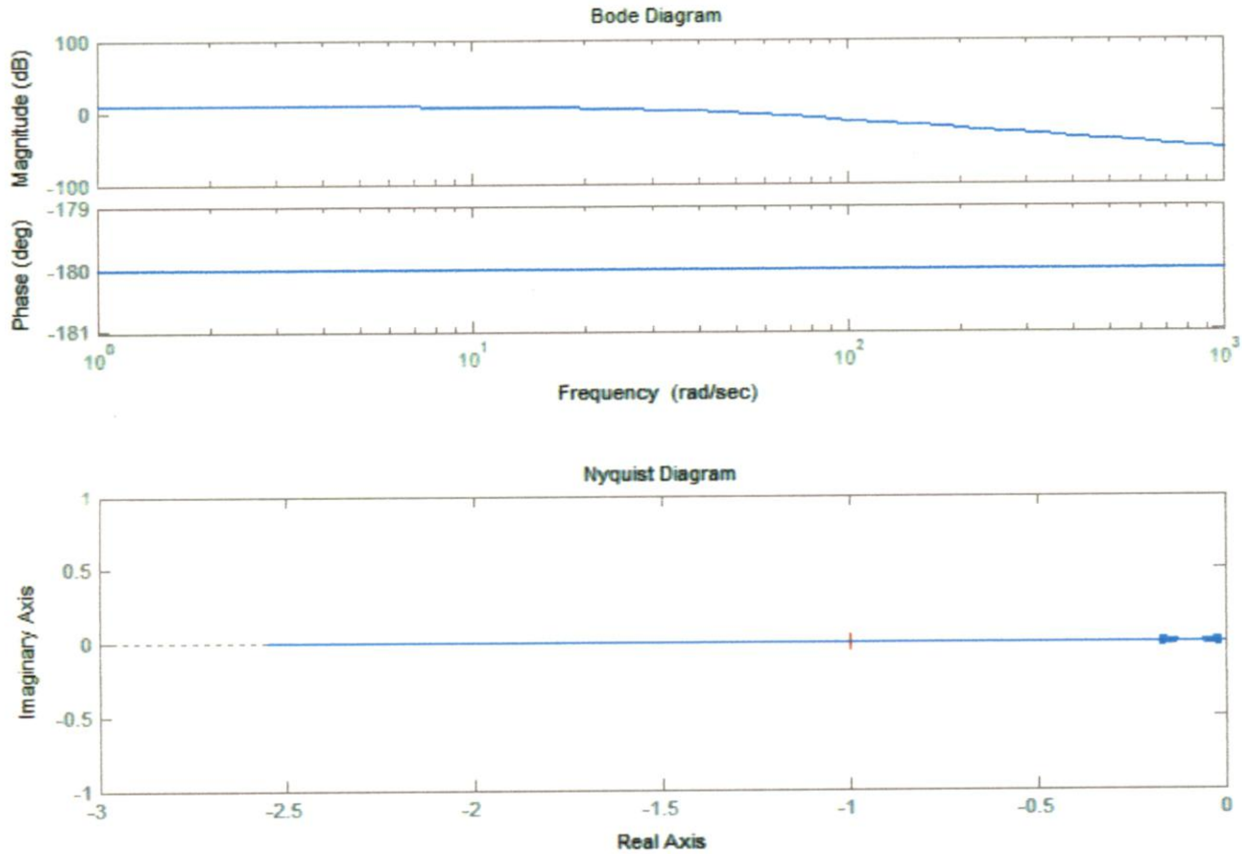

Fig.2.8.2.1 System Bode and Nyquist Plot

We could see that the system has 2 poles, one of which is on the right half S plane. According to Nyquist stability criterion, the sufficient and necessary condition of close loop system be stable is: when ω change from -∞ to +∞ , the open loop transfer function G(jw) encircle the point -1 p times, in which p is the number of poles of open loop transfer function on the right half plane. For ML system, the open loop transfer function has a pole on the right half S plane, so *G(jw)* needs to encircle the point -1 once. We could see from Figure the system Nyquist plot does not encircle the point -1 once, so the system is unstable. Further controller design is required to stabilize the system.

Design controller, let system static position error constant be 5, phase margin be 50^0, gain margin be larger or equal to 10dB.

From above specifications, following controller has been designed for the system.

$$G(s) = \frac{19.7254(s+38.0855)}{s+383.1543} \tag{2.8.3}$$

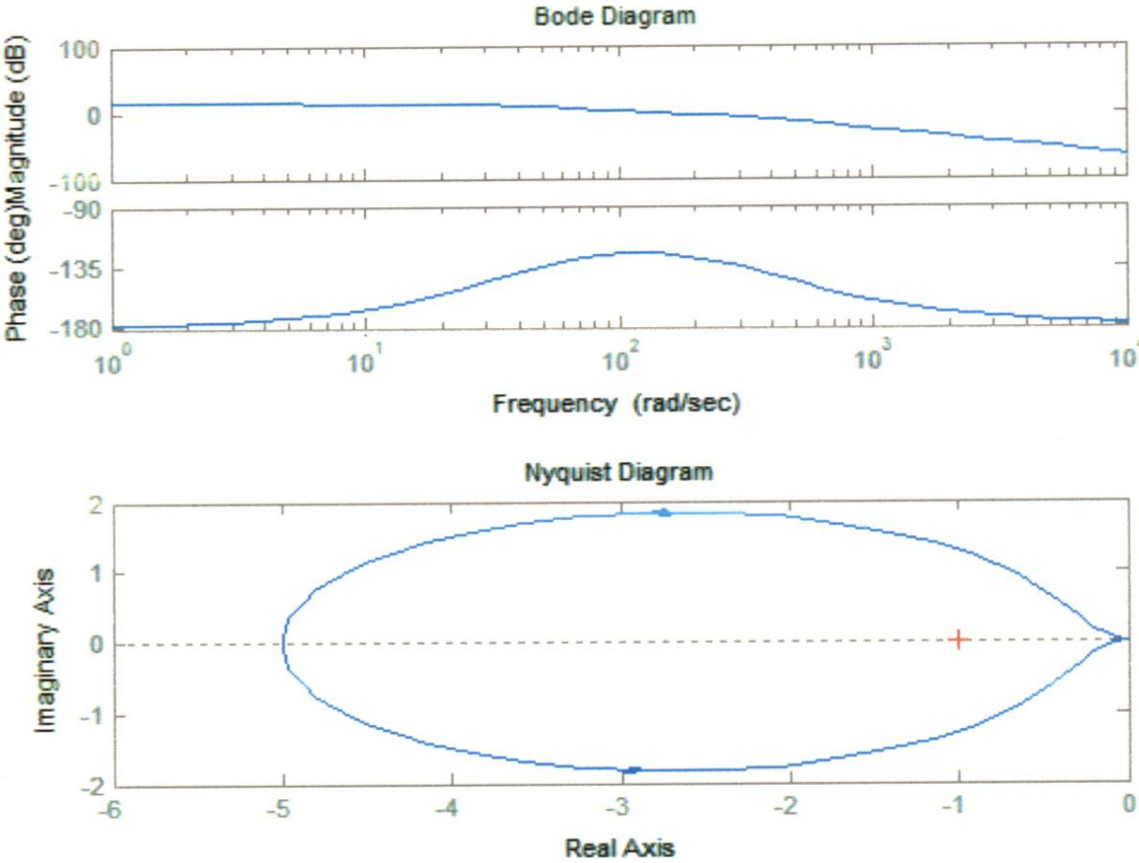

Fig.2.8.2.2 Improved Bode and Nyquist Plot

We can see from Bode plot that system has satisfactory phase and gain margin. From Nyquist plot, the compensated system is stable.

and the compensated system gives step response as:

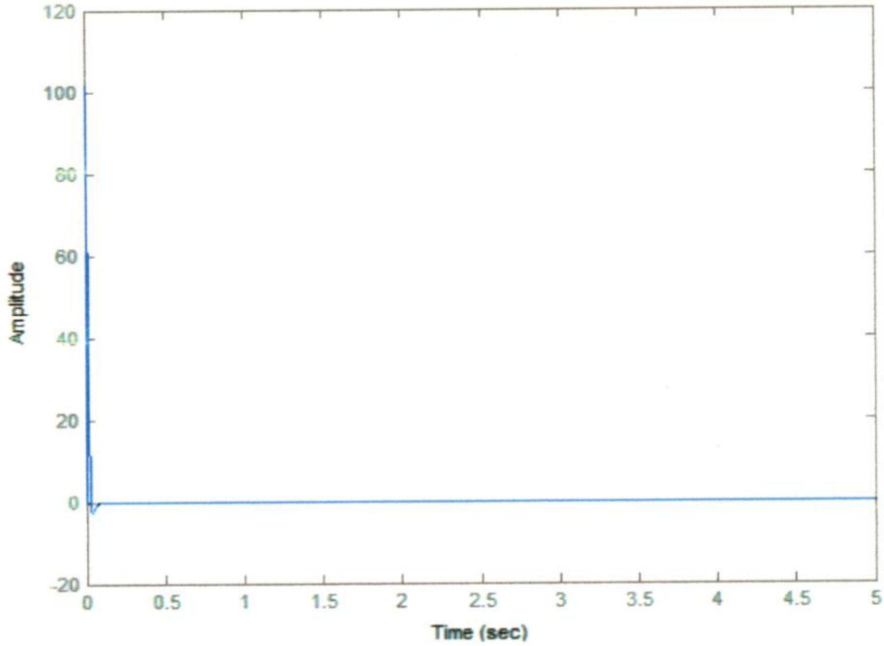

Fig.2.8.2.3 System step response

2.8.3 PID control analysis

In industrial control, the most mature and widely used controller is PID controller. PID controller is a kind of linear controller, which feedback the combination of proportion, integral and differential of the control difference to control the object.

PID Controller Principle

PID controller consists of Proportional Action, Integral Action and Derivative Action. It is commonly refer to Ziegler-Nichols PID tuning parameters. It is by far the most common control algorithm.[11] In this chapter, the basic concept of the PID controls will be explained.

PID controller's algorithms are mostly used in feedback loops. PID controllers can be implemented in many forms. It can be implemented as a stand-alone controller or as part of Direct Digital Control (DDC) package or even Distributed Control System (DCS). The latter is a hierarchical distributed process control system which is widely used in process plants such as pharmaceutical or oil refining industries.

It is interesting to note that more than half of the industrial controllers in use today utilize PID or modified PID control schemes. Below is a simple diagram of illustrating the schematic of the PID controller. Such set up is known as non interacting form or parallel form.

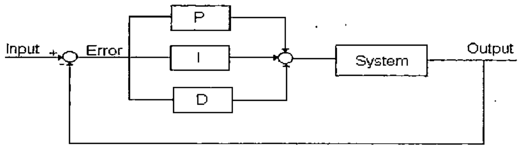

Fig.2.8.3.1 Schematic of the PID controller

In proportional control

$$P_{term} = K_p \times error$$

It uses proportion of the system error to control the system. In this action an offset is introduced in the system.

In Integral control

$$I_{term} = K_i \times \int error.dt$$

It is proportional to the amount of error in the system. In this action, the I-action will introduce a lag in the system. This will eliminate the offset that was introduced earlier on by the P-action.

Derivative control

$$D_{term} = K_d \times (d(error))/dt$$

It is proportional to the rate of change of the error. The D-action will introduce phase lead in the system. This will eliminate the lag in the system that was introduced by the I-action earlier on.

Continuous PID

The three controllers when combined together can be represented by the following transfer function.

$$G_c(s) = K(1 + \frac{1}{sT_i} + sT_d)$$

This can be illustrated below in the following block diagram

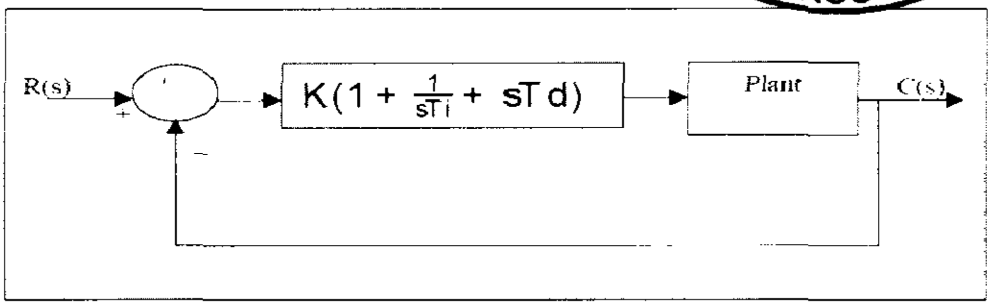

Figure 2.8.3.2 Block diagram of Continuous PID Controller controlling plant

Fig.2.8.3.2 shows PID control of plant. If mathematical model of a plant can be derived then it is possible to apply various design techniques for determining parameters of controller that will meet the transient & steady state specifications of closed loop system. However if the plant is so complicated that its mathematical model cannot be easily obtained, then an analytical approach to the design of PID control is not possible. Then we must resort to the experimental approach to the design of PID controllers.

The process of selecting controller parameters to meet given performance specifications is known as controller tuning.

Now, the First set PID controller as proportion control, let K_p=0.3, K_i=0, K_d=0. the following result is obtained:

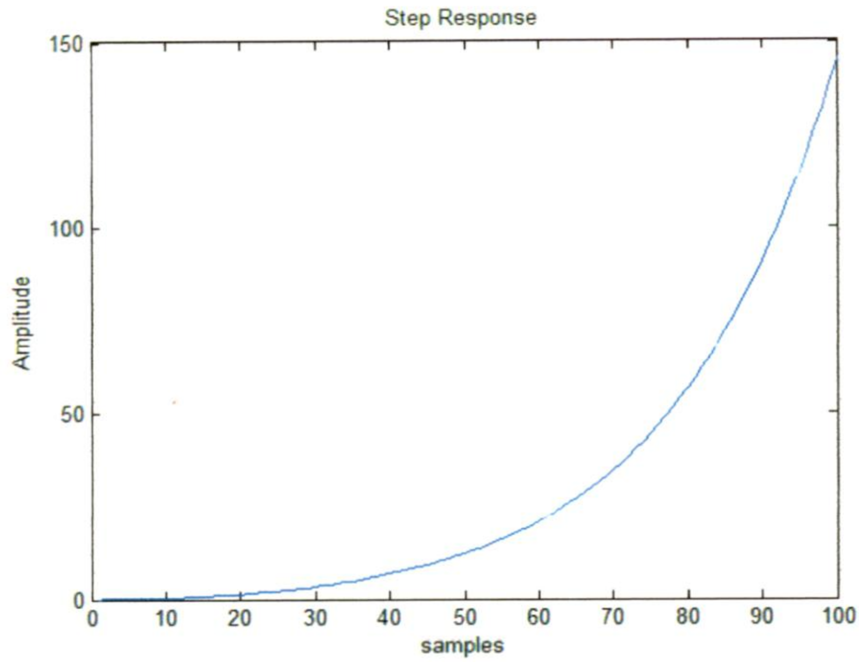

Fig.2.8.3.3 Unstable Plant output with Kp=0.3, Ki-=0,Kd=0

It can be observed from the figure that the system output diverges. Increase control gain, K_p=0.8, K_i=0, K_d=0, the simulation result is as follow:

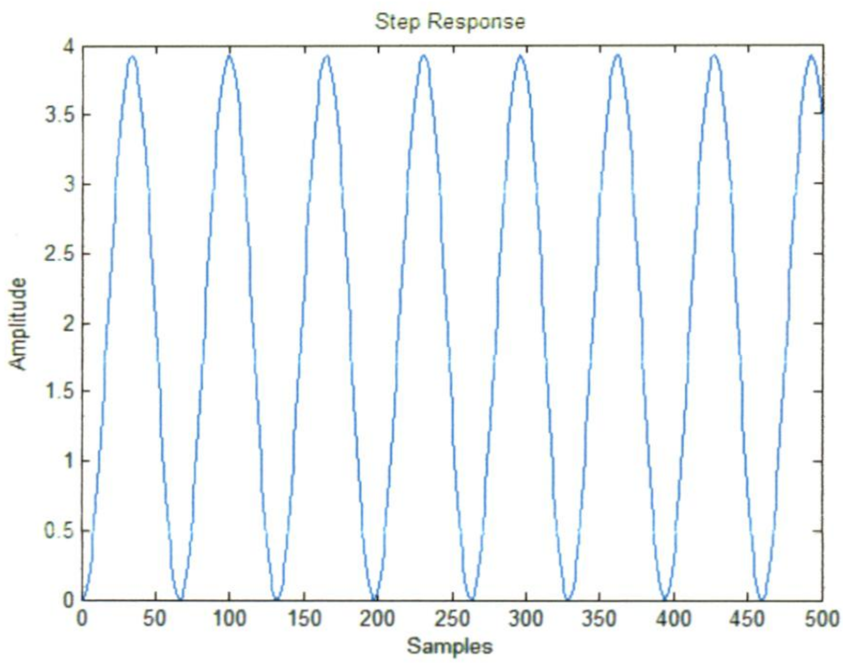

Fig.2.8.3.4 Marginally stable output having Kp=0.8, Ki-=0,Kd=0

In real control, the design parameters are for ideal system. The nonlinearity, noise, etc are ignored in the design process. Therefore, the parameters should be tuned for several times until satisfactory result is obtained. Tune the PID parameters again to get the final response curve as follow

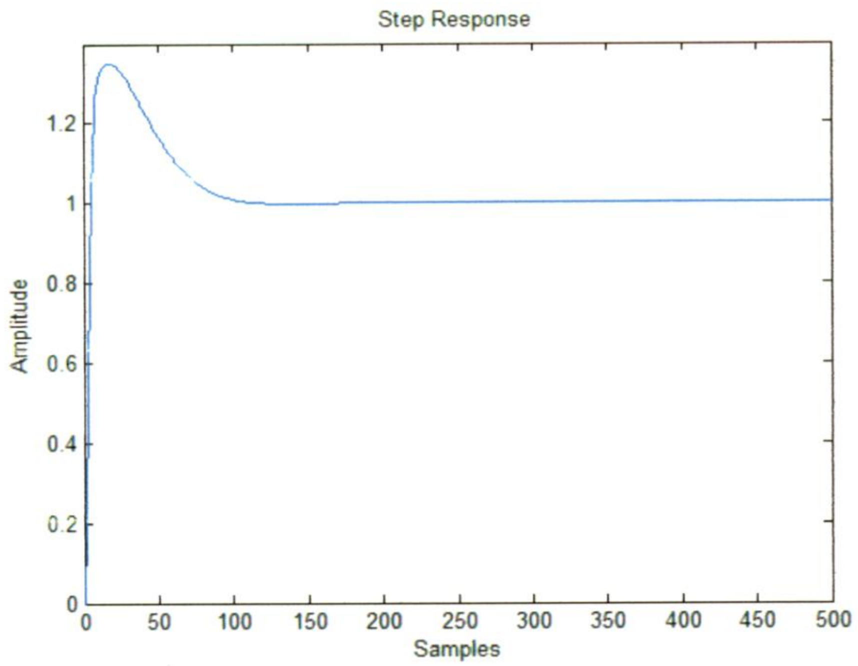

Fig. 2.8.3.5 System step response with Kp=5, Ki-=0.03,Kd=15

It can be observed that system will be stabilized in 0.5 second, but the over shoot is large. User may adjust the parameters by themselves to get the best performance.

Chapter 3

Soft Computing – A perspective

3.1 Soft Computing

Soft Computing is a field, which is characterized by the use of inexact solutions to computationally-hard tasks .Soft computing differs from conventional (hard) computing in that, unlike hard computing, it is tolerant of imprecision, uncertainty, partial truth, and approximation. In effect, the role model for soft computing is the human mind. [12]

At this juncture, the principal constituents of Soft Computing (SC) are Fuzzy Logic (FL), Neural Computing (NC), Evolutionary Computation (EC) Machine Learning (ML) and Probabilistic Reasoning (PR), with the latter subsuming belief networks, chaos theory and parts of learning theory. What is important to note is that soft computing is not a mélange. Rather, it is a partnership in which each of the partners contributes a distinct methodology for addressing problems in its domain. In this perspective, the principal constituent methodologies in SC are complementary rather than competitive. Furthermore, soft computing may be viewed as a foundation component for the emerging field of conceptual intelligence.

- Fuzzy Systems
- Neural Networks
- Evolutionary Computation
- Machine Learning
- Probabilistic Reasoning

The Soft Computing techniques are based on the information processing in biological systems. The complex biological information processing system enables the human beings to survive with accomplishing tasks like recognition of surrounding, making prediction, planning, and acting accordingly. Human type information processing involves both logical and intuitive information processing. Conventional computer systems are good for the former, but their capability for the later is far behind that of human beings. For a computing system to have human like information processing facility, it should be flexible enough to support three features: openness, robustness, and real time processing. Openness of a system is its ability to adapt or extend on its own to cope with changes encountered in the real world. Robustness of a system means its stability and tolerability when confronted with distorted, incomplete, or imprecise information. The real time characteristic implies the ability of the system to react within a reasonable time in response to an event. Information processing systems with all these three characteristics are known as real world computing (RWC) systems.[13]

Advantages of Soft computing techniques :

The complementarily of FL, NC, GC, and PR has an important consequence: in many cases a problem can be solved most effectively by using FL, NC, GC and PR in combination rather than exclusively. A striking example of a particularly effective combination is what has come to be known as "neurofuzzy systems." Such systems are becoming increasingly visible as consumer products ranging from air conditioners and washing machines to photocopiers and camcorders. Less visible but perhaps even more important are neurofuzzy systems in industrial applications. What is particularly significant is that in both consumer products and industrial systems, the employment of soft computing techniques leads to systems which have high MIQ (Machine Intelligence Quotient). In large measure, it is the high MIQ of SC-based systems that accounts for the rapid growth in the number and variety of applications of soft computing.

3.2 Fundamentals of soft computing

3.2.1 Fuzzy Logic

Fuzzy logic is a form of multi-valued logic derived from fuzzy set theory to deal with reasoning that is approximate rather than precise. In contrast with "crisp logic", where binary sets have binary logic, fuzzy logic variables may have a truth value that ranges between 0 and 1 and is not constrained to the two truth values of classic propositional logic. Furthermore, when linguistic variables are used, these degrees may be managed by specific functions.

Fuzzy Logic Control Theory

Fuzzy logic is derived from fuzzy set theory dealing with reasoning that is approximate rather than precisely deduced from classical predicate logic. It can be thought of as the application side of fuzzy set theory dealing with well thought out real world expert values for a complex problem

3.2.2 Fuzzy logic Controller

Fuzzy control system realizes the control based on system error and its variance. The control system basic structure is shown in Figure 3.2.2.1[10], which includes fuzzification, fuzzy rule, fuzzy inference, defuzzification and output quantification,etc.

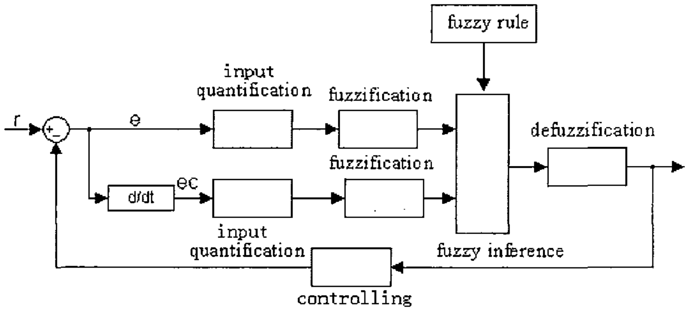

Fig.3.2.2.1 Fuzzy Control Logic

Where,

r - fixed value, y is the output,

e and e_c are control error and its derivative respectively.

E and E_c is the language variable after input quantification.

U is the basic fuzzy controller language variable,

u is the actual output after output quantification.

Fuzzification: Fuzzification transforms from an accurate data variable to a fuzzy variable according to its membership function. Consider a fuzzy set of error with 7 members: negative large, negative medium, negative small, zero, small, medium, large. Triangular or trapezoidal shape function is widely used.

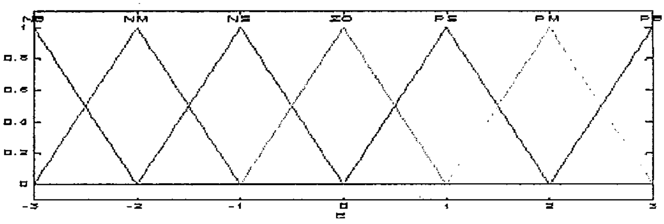

Fig 3.2.2.3 Membership function for error

Fuzzy Rule:

Fuzzy rule set is an important part of the fuzzy controller. The knowledge and operation experience are all saved in the rule set. Table 3-5 gives an example of the fuzzy rule set. In the table: NB-Negative large; NM-Negative Medium; NS-Negative Small; Z-Zero; PS-Small; PM-Positive Medium; PB-Positive Large.[14]

Table 3.2.2.1 Fuzzy Rule Base

E \ Ec	NB	NM	NS	Z	PS	PM	PB
NB	PB	PB	PM	PS	PS	Z	Z
NM	PB	PB	PM	PM	PS	PS	Z
NS	PB	PB	PM	PS	PS	Z	Z
Z	PB	PM	PM	Z	NM	NM	NB
PS	Z	Z	Z	NS	NS	NM	NB
PM	Z	Z	NS	NS	NM	NB	NB
PB	Z	Z	NS	NS	NM	NB	NB

Fuzzy Inference:

Fuzzy inference means, according to input fuzzy variables, solve for the fuzzy relation equation by fuzzy rules. Fuzzy inference is the most basic problem in fuzzy logic theory. The most commonly used method is max-min inference. Next we will introduce the inference method on the fuzzy set with triangle membership function.

For 2 input variables E and E_c, a fuzzy controller with output U, the control rules are

If E is A_{11} and E_c is A_{12}, then U is U_1

If E is A_{21} and E_c is A_{22}, then U is U_2

In which A_{11} & A_{12} and A_{21} & A_{22} are the neighboring fuzzy set of input variable E and E_c ; which U_1 and U_2 are two neighboring subset of output variable U . If there is $E=e_0$ and $E_c = e_{c0}$, then according to the membership function, w_i is :

The result of rule I is,

$$W_1 = \mu_{A11}(e_0) * \mu_{12}(e_{c0}) \tag{3.2.2.1}$$

And that of rule II is,

$$W_2 = \mu_{A21}(e_0) * \mu_{22}(e_{c0}) \tag{3.2.2.2}$$

Where * is min(minimal) or algebra product.

And combined output of two rules is given as :

$$\mu_u(u) = W_1 \vee W_2 \tag{3.2.2.3}$$

The inference result shows the fuzzy rule inference is complete. However, the result is still a fuzzy vector, which can not be directly used as control value. Further transformation is required to obtain clear control output, which is defuzzification.

Defuzzification:

Defuzzification is the process of producing a quantifiable result in fuzzy logic. There are several defuzzification methods, the most commonly used are:

a) Max membership method

b) Bisector

c) Centroid

The above 3 methods have their advantages and disadvantages respectively. In real application, user should choose the proper method according to the application.

Input Output Quantification

The I/O quantification changes the input variable from the basic domain to the language variables domain. Take error e for example, it's domain is determined by the real process and operation experience, the corresponding domain of language variable E is determined by the number of fuzzy set defined on E . For example 7 fuzzy sets, then the domain of E is [-3 , 3], difference input quantification is to multiply e with a proportional gain to obtain E, then the domain of e is also become [-3 , 3]. Similarly, e_c should be quantified by multiplying proportional gain. The output language variable *U* should also be quantified by output proportional gain, which change the domain of to the domain of real output variable *u*.

3.2.3 Structure of Fuzzy logic Controller

Magnetic levitation system is a typical nonlinear hysteretic system, which is hard to build precise mathematical model. Traditional PID controller is also hard to get precise mathematical model, thus the dynamic performance and control effect is not satisfactory. The advantage of fuzzy control is that it doesn't require the precise model of the control object, but organize the control rule decision table to decide the control input. Such system has good dynamic performance however the steady state performance is not good, which is subject to the control rule and the quantification level. What's more, ordinary fuzzy control is similar with PD control, which has a non-zero steady state error. Fuzzy PID controller combines the advantage of fuzzy control (dynamic performance) and PID control (steady state performance). Regular PID controller can not realize the parameters online adjustment, but the fuzzy PID controller adding fuzzy parameters self-tuning control will auto tune the 3 PID parameters K_p , K_i , K_d online to gain better performance. The fuzzy PID controller mainly includes fuzzy parameter tuning control and PID control.

Fuzzy PID Controller Structure Self-tuning fuzzy PID controller mainly composed of self-tuning PID and fuzzy control. The diagram is shown below [14]

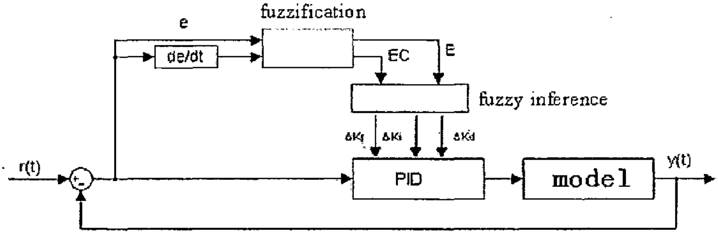

Fig.3.2.3.1 Fuzzy PID controller

PID control realizes the system control part. Fuzzy logic system utilizes error e and its derivative e_c as the input to auto-tune the PID parameters K_p, K_d, K_i online by fuzzy logic, thus to obtain good dynamic and steady state performance.

Basic Form of Fuzzy PID Controller According to fuzzy logic output physical meaning, the fuzzy PID controller can be classified as: gain adjustment, direct control and hybrid type.

a) Gain adjustment fuzzy PID controller

The output signal of this kind of controller directly corresponds to gain parameters. 3 gain parameters are tuned on line by fuzzy rules. One type of the gain adjustment fuzzy PID controller is based on the performance index as follow:

If ("Perform Index is ...") then (ΔKp is ...) and (ΔKi is ...) and (ΔKd is...)

the "Perform Index" can be overshoot, steady state error or other index.

Another kind of gain adjustment fuzzy PID controller is based on error. The formulation is as follow:

If (e is ...) and ($\Delta e$is ...) then (ΔKp is ...) and (ΔKi is ...) and (ΔKd is...)

The gain parameter of this kind of controller is nonlinear function of error and its variation. For example the proportional gain can be expressed as: . In recent years, this kind of fuzzy PID controller is most widely explored and used. According to the difference ways of tuning, there are fuzzy self-tuning PID controller and fuzzy self-compensating PID controller.

b) Directly control fuzzy PID controller

If the fuzzy logic output is the control variable of PID theory, then this kind of controller belongs to directly control type. The kind of controller can be classified to 12 structure units. Each unit has different control effect, which can be regarded as independent.

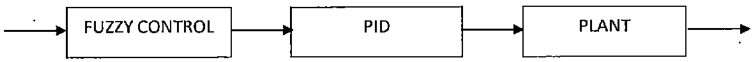

Fig.3.2.3.2 Directly control fuzzy PID controller

c) Hybrid fuzzy PID controller

There are many form of hybrid fuzzy PID controller, like the combination of gain adjustment and directly control, or combination of traditional PID control and fuzzy control.[16] Some researchers proposed to first use fuzzy control for the fast response and then use PID controller for detailed adjustment. This Fuzzy-PID controller has faster dynamic response than PID controller as well as less overshoot. The steady state precision is also higher than fuzzy control. However, how to realize the smooth switch is a problem. On the other hand, to solve the problem of PD controller that can not eliminate steady state error, integral unit can be added. This is called fuzzy PD+ linear I. [15]

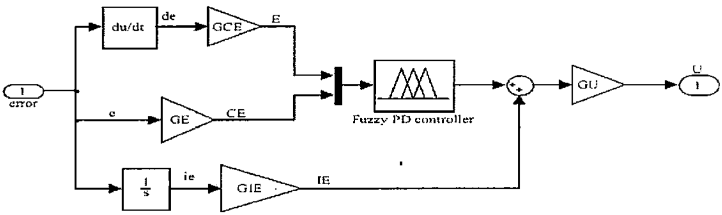

Fig. 3.2.3.3 Fuzzy PD + Linear I Hybrid controller

3.2.4 Sugeno type Fuzzy Logic control

The structure of the Takagi-Sugeno type fuzzy model in general is similar to Mamdani type fuzzy model. Here instead of Mamdani type sub-models Takagi-Sugeno type sub-models are used. The first two parts of the fuzzy inference process, fuzzifying the inputs and applying the fuzzy operator, are exactly the same. The main difference between Mamdani and Sugeno is that the Sugeno output membership functions are either linear or constant. From the experiments it was noticed that Mamdani type fuzzy model is more accurate than Takagi-Sugeno, but it has one big drawback – huge amount of parameters that need to be identified. The process of identification is time consuming so it would not be possible to implement this model in real time control.[16]

Takagi-Sugeno FS essentially performs a nonlinear interpolation between linear mappings.

A typical rule in a Sugeno fuzzy model has the form

If Input 1 = x and Input 2 = y, then Output is $z = ax + by + c$

For a zero-order Sugeno model, the output level z is a constant $(a=b=0)$.

The output level z_i of each rule is weighted by the firing strength w_i of the rule. For example, for an AND rule with Input 1 = x and Input 2 = y, the firing strength is

$$w_i = AndMethod(F_1(x), F_2(y)) \tag{3.2.4.1}$$

where $F_{1,2}$ $(.)$ are the membership functions for Inputs 1 and 2.

The final output of the system is the weighted average of all rule outputs, computed as

$$\text{Final Output} = \frac{\sum_{i=1}^{N} w_i z_i}{\sum_{i=1}^{N} w_i} \tag{3.2.4.2}$$

where N is the number of rules.

Comparisons between Mamdani and Sugeno type inference system

Advantages of the Mamdani Method

- It is intuitive.
- It has widespread acceptance.
- It is well suited to human input.

Advantages of the Sugeno Method

- It is computationally efficient.
- It works well with linear techniques (e.g., PID control).
- It works well with optimization and adaptive techniques.
- It has guaranteed continuity of the output surface.
- It is well suited to mathematical analysis.

3.2.5 Genetic Algorithm

Genetic Algorithms (GA) are a stochastic global search method that mimics the process of natural evolution. It is one of the methods used for optimization. John Holland formally introduced this method in the United States in the 1970 at the University of Michigan. The genetic algorithm starts with no knowledge of the correct solution and depends entirely on responses from its environment and evolution operators such as reproduction, crossover and mutation to arrive at the best solution. By starting at several independent points and searching in parallel, the algorithm avoids local minima and converging to sub optimal solutions.

a) Characteristics of Genetic Algorithm

Genetic Algorithms are search and optimization techniques inspired by two biological principles namely the process of natural selection and the mechanics of natural genetics. GAs manipulates not just one potential solution to a problem but a collection of potential solutions. This is known as population. The potential solution in the population is called chromosomes. These chromosomes are the encoded representations of all the parameters of the solution. Each chromosomes is compared to other chromosomes in the population and awarded fitness rating that indicates how successful this chromosomes to the latter.

To encode better solutions, the GA will use genetic operators or evolution operators such as crossover and mutation for the creation of new chromosomes from the existing ones in the population. This is achieved by either merging the existing ones in the population or by modifying an existing chromosome.

The selection mechanism for parent chromosomes takes the fitness of the parent into account. This will ensure that the better solution will have a higher chance to procreate and donate their beneficial characteristic to their offspring genetic algorithm is typically initialized with a random population consisting of between 20-100 individuals. This population or also known as mating pool is usually represented by a real-valued number or a binary string called a chromosome. For illustrative purposes, the rest of this section represents each chromosome as a binary string. How well an individual performs a task is measured and assessed by the objective function. The objective function assigns each individual a corresponding number called its fitness. The fitness of each chromosome is assessed and a survival of the fittest strategy is applied. In this, the magnitude of the error is used to assess the fitness of each chromosome.

There are three main stages of a genetic algorithm; these are known as reproduction, crossover and mutation. This will be explained in details in the following section.

b) Population Size

Determining the number of population is the one of the important step in GA. There are many research papers that dwell in the subject. Many theories have been documented and experiments recorded [17]. However the matter of the fact is that more and more theories and experiments are conducted and tested and there is no fast and thumb rule with regards to which is the best method to adopt. For a long time the decision on the population size is based on trial and error [18].

In this seminar the approach in determining the population is rather unscientific. It is suggested that the safe population size is from 30 to 100. In this an initial population of 20 was used and the result was observed. The result was not promising. Hence initiatives of 40, 60, 80 and 90 size of population were experimented. It was shown that the population of 80 seems to be a good guess. Population of 90 and above does not results in any further optimization.

c) Reproduction

Reproduction is usually the first operators applied on population. Chromosomes are selected from the population to be parents to cross over and produce offspring. According to Darwin's evolution theory of Survival of fittest, the best one should survive and create new offspring that is why reproduction operator is sometime known as selection operator .The essential idea is that above average strings are picked from the current population and their multiple copies are inserted in the mating pool in a probabilistic manner.

During the reproduction phase the fitness value of each chromosome is assessed. This value is used in the selection process to provide bias towards fitter individuals. Just like in natural evolution, a fit chromosome has a higher probability of being selected for reproduction.

An example of a common selection technique is the Roulette Wheel selection method as shown in Figure 3.2.5.1.

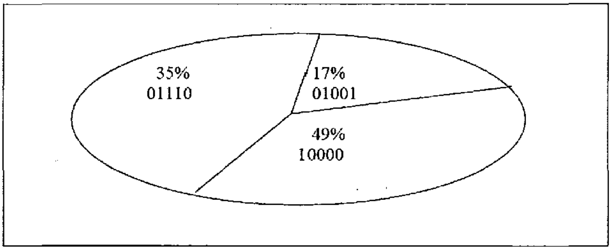

Fig.3.2.5.1 Depiction of roulette wheel selection

Each individual in the population is allocated a section of a roulette wheel. The size of the section is proportional to the fitness of the individual.[19] A pointer is spun and the individual to whom it points is selected. This continues until the selection criterion has been met. The probability of an individual being selected is thus related to its fitness, ensuring that fitter individuals are more likely to leave offspring. Multiple copies of the same string may be selected for reproduction and the fitter strings should begin to dominate.

There are a number of other selection methods available and it is up to the user to select the appropriate one for each process. All selection methods are based on the same principal that is giving fitter chromosomes a larger probability of selection.

Four common methods for selection are:

1. Roulette Wheel selection

2. Stochastic Universal sampling

3. Normalized geometric selection

4. Tournament selection

Due to the complexities of the other methods, the Roulette Wheel method is preferred.

d) Crossover

Once the selection process is completed, the crossover algorithm is initiated. The crossover operations swap certain parts of the two selected strings in a bid to capture the good parts of old chromosomes and create better new ones. Genetic operators manipulate the characters of a chromosome directly, using the assumption that certain individuals' gene codes, on average, produce fitter individuals. The crossover probability indicates how often crossover is performed. A probability of 0% means that the offspring will be exact replicas of their parents and a probability of 100% means that each generation will be composed of entirely new offspring.

Uniform crossover is the most disruptive of the crossover algorithms and has the capability to completely dismantle a fit string, rendering it useless in the next generation. Because of this Uniform Crossover will not be used in this and Multi-Point Crossover is the preferred choice.

e) Mutation

Using selection and crossover on their own will generate a large amount of different strings. However there are two main problems with this.

1. Depending on the initial population chosen, there may not be enough diversity in the initial strings to ensure the Genetic Algorithm searches the entire problem space.

2. The Genetic Algorithm may converge on sub-optimum strings due to a bad choice of initial population.

These problems may be overcome by the introduction of a mutation operator into the Genetic Algorithm. Mutation is the occasional random alteration of a value of a string position. It is considered a background operator in the genetic algorithm. The probability of mutation is normally low because a high mutation rate would destroy fit strings and degenerate the genetic algorithm into a random search. Mutation probability values of around 0.1% or 0.01% are common, these values represent the probability that a certain string will be selected for mutation i.e. for a probability of 0.1%; one string in one thousand will be selected for mutation.

Once a string is selected for mutation, a randomly chosen element of the string is changed or mutated.

f) Summary of Genetic Algorithm Process

In this section the process of Genetic Algorithm will be summarized in a flowchart. The summary of the process will be described below.

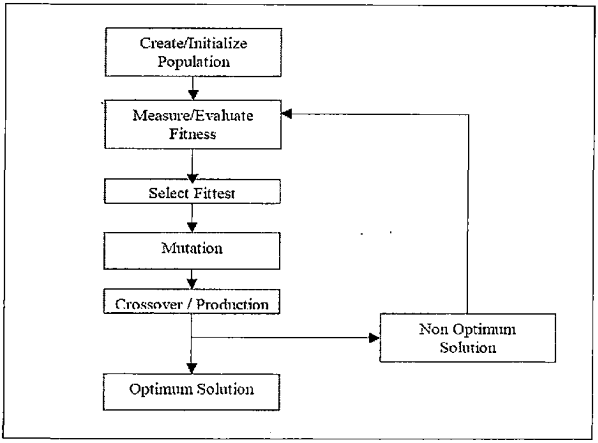

Fig.3.2.5.2 Genetic Algorithm Process Flowchart

The steps involved in creating and implementing a genetic algorithm:

1. Generate an initial, random population of individuals for a fixed size.
2. Evaluate their fitness.
3. Select the fittest members of the population.
4. Reproduce using a probabilistic method (e.g., roulette wheel).
5. Implement crossover operation on the reproduced chromosomes (choosing probabilistically both the crossover site and the mates).
6. Execute mutation operation with low probability.
7. Repeat step 2 until a predefined convergence criterion is met.

The convergence criterion of a genetic algorithm is a user-specified conditions for example the maximum number of generations or when the string fitness value exceeds a certain threshold.

g) Objective Function or Fitness Function

The objective function is used to provide a measure of how individuals have performed in the problem domain. In the case of a minimization problem, the most fit individuals will have the lowest numerical value of the associated objective function. This raw measure of fitness is usually only used as an intermediate stage in determining the relative performance of individuals in a GA. Another function that is the fitness function is normally used to transform the objective function value into a measure of relative fitness, thus where f is the objective function, g transforms the value of the objective function to a nonnegative number and F is the resulting relative fitness. This mapping is always necessary when the objective function is to be minimized as the lower objective function values correspond to fitter individuals. In many cases, the fitness function value corresponds to the number of offspring that an individual can expect to produce in the next generation. A commonly used transformation is that of proportional fitness assignment [20].

3.2.6 Particle Swarm optimization

Particle swarm optimization (PSO) is a method for performing numerical optimization without explicit knowledge of the gradient of the problem to be optimized. PSO is originally attributed to Kennedy, Eberhart and Shi[21] and was first intended for simulating social behavior. The algorithm was simplified and it was observed to be performing optimization. The book by Kennedy and Eberhart describes many philosophical aspects of PSO and swarm intelligence. An extensive survey of PSO applications is made by Poli [22].

The features of the method are as follows :

- The method is developed from research on swarm such as fish schooling and bird flocking.
- It can be easily implemented, and has stable convergence characteristic with good computational efficiency.

PSO optimizes a problem by maintaining a population of candidate solutions called particles and moving these particles around in the search-space according to simple formulae. The movements of the particles are guided by the best found positions in the search-space, which are continually updated as better positions are found by the particles.

The particle swarm optimization algorithms are based on two socio-metric principles. Particles fly through the solution space and are influenced by both the best particle in the particle population and the best solution that a current particle has discovered so far. The best particle in the population is typically denoted by (gobal best), while the best position that has been visited by the current particle is donated by (local best). The (global best) individual conceptually connects all members of the population to one another. That is, each particle is influenced by the very best performance of any member in the entire population. The (local best) individual is conceptually seen as the ability for particles to remember past personal success. The particle swarm optimization makes use of a velocity vector to update the current position of each particle in the swarm. The position of each particle is updated based on the social behavior that a population of individuals adapts to its environment by returning to promising regions that were previously discovered.

Block Diagram

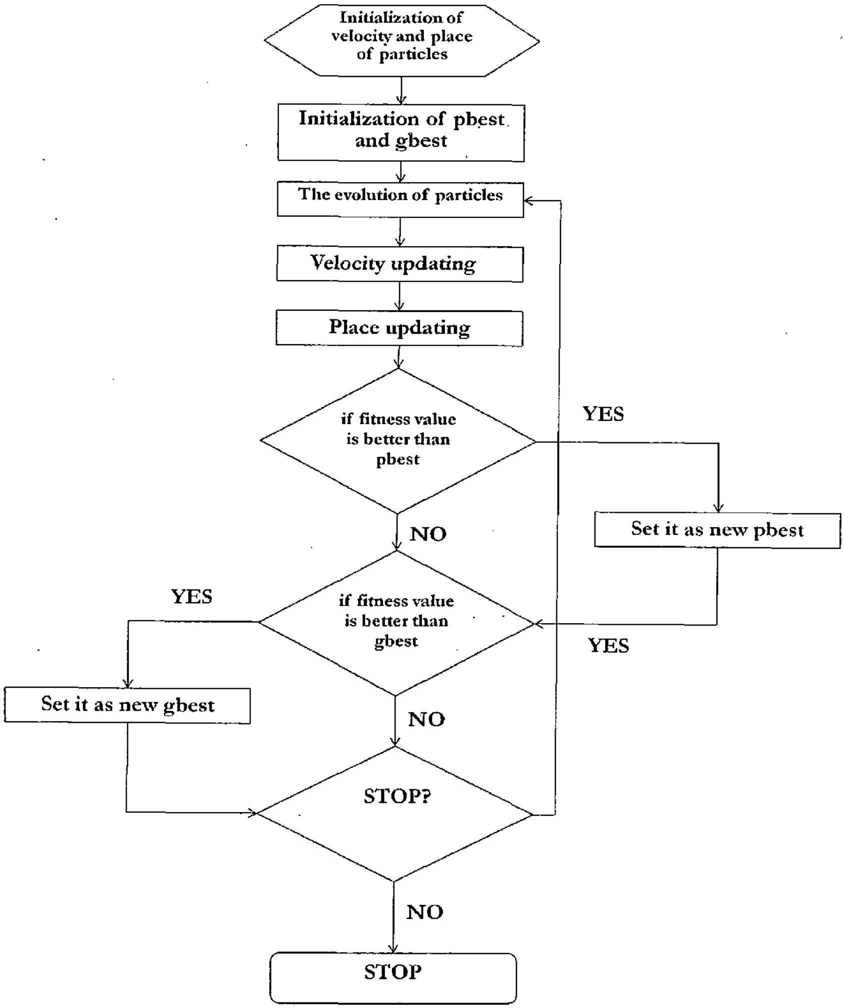

Fig. 3.2.6.1 PSO Flow Diagram

Particle swarm optimization algorithm

Basic algorithm as proposed by Kennedy and Eberhart (1995)[21]

x_k^i - Particle position

v_k^i- Particle velocity

p_k^i- Best "remembered" individual particle position

p_k^g- Best "remembered" swarm position

c_1, c_2- Cognitive and social parameters

r_1, r_2- Random numbers between 0 and 1

Position of individual particles updated as follows:

$$x_{k+1}^i = x_k^i + v_{k+1}^i \qquad (3.2.6.1)$$

with the velocity calculated as follows:

$$v_{k+1}^i = v_k^i + c_1 r_1 (p_k^i - x_k^i) + c_2 r_2 (p_k^g - x_k^i) \qquad (3.2.6.2)$$

Comparison between PSO and Genetic Algorithm

Particle swarm optimization can be used to solve many of the same kinds of problems as genetic algorithms **(GAS)**. This optimization technique does not scffer, however, from some of GA's difficulties; interaction in the group enhances rather than detracts from progress toward the solution. Further, a particle swarm system has memory, which the genetic algorithm does not have Change in genetic populations results in destruction of previous knowledge of the problem, except when elitism is employed, in which case usually one or a small number of individuals retain their "identities." In particle swarm optimization, individuals who fly past optima are tugged to return toward them; knowledge of good solutions is retained by all particles.

Chapter - 4

PID controller tuning strategies

4.1 Introduction and necessity of tuning

Designing and tuning a PID controller appears to be conceptually intuitive, but can be hard in practice, if multiple (and often conflicting) objectives are to be achieved. A conventional PID controller with fixed parameters may usually derive poor control performance when it comes to system complexities. [24]. Since the gain and the time constants of the system changes with the operating conditions so the conventional PID controllers result in sub-optimal corrective actions and, hence, require frequent tuning adjustments. This stimulates the development of tools that can assist engineers to achieve the best overall PID control for the entire operating envelope of a given process. In the past few decades, neural networks have been used to meet system complexities like nonlinearities [4],, but their real time implementation is quite difficult.

Genetic Algorithms (GAs) are a stochastic global search method that mimics the process of natural evolution. Genetic Algorithms have been shown to be capable of locating high performance areas in complex domains without experiencing the difficulties associated with high dimensionality or false optima as may occur with gradient decent techniques. GA finds the optimal solution through cooperation and competition among the potential solution. Using genetic algorithms to perform the tuning of the controller will result in the optimum controller being evaluated for the system every time.

4.2 Techniques for controller tuning

Various techniques can be employed to tune the value of Proportional, Integral and Derivative gains of the system. All methods are giving the value of PID based on some optimum condition of objective function.

4.2.1 Zigler- Nicholas Oscillation Method

For the system under study, Ziegler-Nichols tuning rule based on critical gain K_{cr} and critical period P_{cr} will be used. In this method, the integral time T_i will be set to infinity and the derivative time T_d to zero. This is used to get the initial PID setting of the system. This PID setting will then be further optimized using the steepest descent gradient method.

In this method, only the proportional control action will be used. The K_p will be increased to a critical value K_{cr} at which the system output will exhibit sustained oscillations. In this method, if the system output does not exhibit the sustained oscillations hence this method does not apply.

In this chapter the inefficiency of designing PID controller using the classical method will be shown. This design will be further improved by the optimization method such as steepest descent gradient method as mentioned earlier

Designing PID Parameters

From the response below, the system under study is indeed oscillatory and hence the Z-N tuning rule based on critical gain K_{cr} and critical period P_{cr} can be applied.

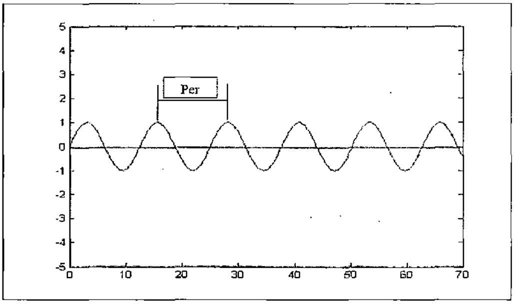

Figure 4.2.1.1 Illustration of Sustained Oscillation with Period (Per.)

The transfer function of the PID controller is

$$G_c(s) = K(1 + \frac{1}{sT_i} + sT_d)$$

The objective is to achieve a unit-step response curve of the designed system that exhibits a maximum overshoot of 25 %. If the maximum overshoot is excessive says about greater than 40%, fine tuning should be done to reduce it to less than 25%.

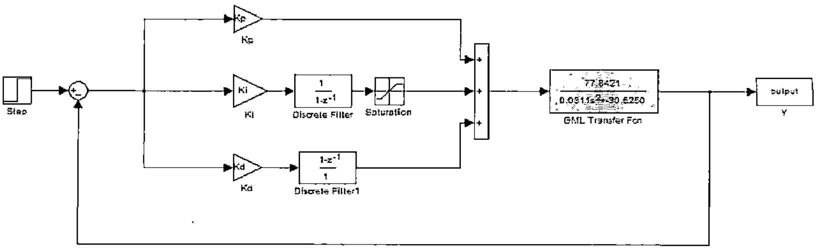

Fig. 4.2.1.2 Simulink implementation of PID control

Method:

1. Keep $T_i = \infty$ and $T_d = 0$,only K_p will remain active.
2. Keep increasing value of K_p till system start oscillating.
3. This value of K_p is known as critical gain, K_{er}.
4. Also calcute time period of the oscillation, P_{er}.
5. Now, from the relation given in table below, value of K_p, K_i and K_d can be calculated.
6. This values are used for implementing PID controller in plant.

Table 4.2.1.1 Recommended PID Value Setting.

Type of controller	K_p	T_i	T_d.
P	$0.5\ K_{er}$	∞	0
PI	$0.45\ K_{er}$	$(1/1.2)\,P_{er}$	0
PID	$0.6\ K_{er}$	$0.5 P_{er}$	$0.125\ P_{er}$

From above given method, system response obtained which is shown below. It is clear from the figure that system response can be tuned better, hence new methods for PID tuning are implemented.

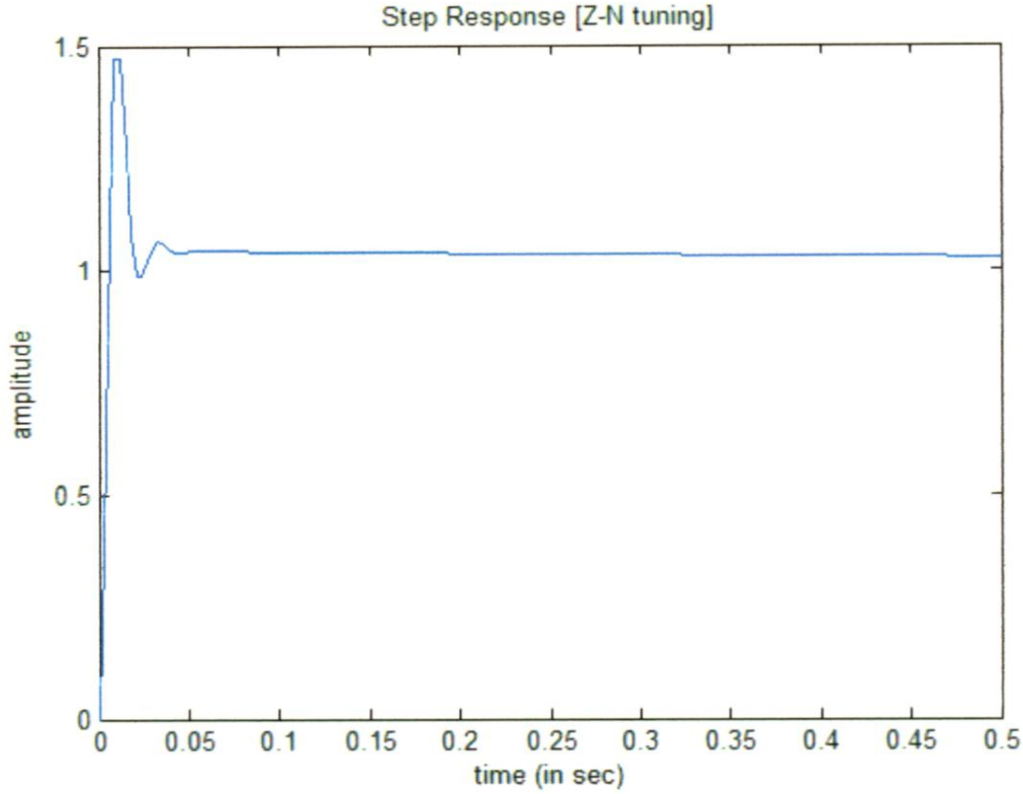

Fig. 4.2.1.3 Step response of Z-N tuned PID controller

4.2.2 Modern techniques

4.2.2.1 Implementation using Genetic Algorithm

Before going into the subject it is good to discuss the differences between Genetics Algorithm against the traditional methods. This will help us understand why GA is more efficient than the latter. Genetic algorithms are substantially different to the more traditional search and optimization techniques. The five main differences are:

1. Genetic algorithms search a population of points in parallel, not from a single point.
2. Genetic algorithms do not require derivative information or other auxiliary Knowledge; only the objective function and corresponding fitness levels influence the direction of the search.
3. Genetic algorithms use probabilistic transition rules, not deterministic rules.
4. Genetic algorithms work on an encoding of a parameter set not the parameter set itself (except where real-valued individuals are used).

5. Genetic algorithms may provide a number of potential solutions to a given Problem and the choice of the final is left up to the user.

Initializing the Population of the Genetic Algorithm

The Genetic Algorithm has to be initialized before the algorithm can proceed. The Initialization of the population size, variable bounds and the evaluation function are required. These are the initial inputs that are required in order for the Genetic Algorithm process to start.

- Population Size - The first stage of writing a Genetic Algorithm is to create a population. This command defines the population size of the GA. Generally the bigger the population size the better is the final approximation.
- Variable Bounds – Since in this we are using genetic algorithms to optimize the gains of a PID controller. There are going to be three strings assigned to each member of the population, these members will be comprised of a P, I and a D string that will be evaluated throughout the course of the GA processes. The three terms are entered into the genetic algorithm via the declaration of a three-row variable bounds matrix. The number of rows in the variable bounds matrix represents the number of terms in each member of the population.
- EvalFN - The evaluation function is the MATLAB function used to declare the objective function. It will fetch the file name of the objective function and execute the codes and return the values back to the main codes.
- Options - Although the previous examples in this section were all binary encoded, this was just for illustrative purposes. Binary strings have two main drawbacks:
  1. They take longer to evaluate due to the fact they have to be converted to and to/from binary.
  2. Binary strings will lose its precision during the conversion process.

  As a result of this and the fact that they use less memory, real (floating point) numbers will be used to encode the population. This is signified in the options command where the 1e-6 terms the floating point precision and the .1 term indicates that real numbers are being used (0 indicates binary encoding used).
- Initialisega . This command is from the GAOT toolbox. It will combine all the previously described terms and creates an initial population of members between bounds with 6 decimal place precision.

The Objective Function of the Genetic Algorithm

The most challenging part of creating a genetic algorithm is writing the the objective function.In this the objective function is required to evaluate the best PID controller for the system. An objective function could be created to find a PID controller that gives the smallest overshoot, fastest rise time or quickest settling time. However in order to combine all of these objectives it was decided to design an objective function that will minimize the error of the controlled system instead. Each chromosome in the population is passed into the objective function one at a time. The chromosome is then evaluated and assigned a number to represent its fitness, the bigger its number the better its fitness. The genetic algorithm uses the chromosome's fitness value to create a new population consisting of the fittest members.

Each chromosome consists of three separate strings constituting a P, I and D term, as defined by the 3-row bounds declaration when creating the population. When the chromosome enters the evaluation function, it is split up into its three Terms. The P, I and D gains are used to create a PID controller according to the equation below.

$$C_{pid} = K_d s^2 + K_p s + K_i$$

The newly formed PID controller is placed in a unity feedback loop with the system transfer function. This will result in a reduction of the compilation time of the program. The system transfer function is defined in another file and imported as a global variable. The controlled system is then given a step input and the error is assessed using an error performance criterion such as Mean square error (MSE), Integral Square Error(ISE), Integral of absolute value of error (IAE),Integral of time absolute error(ITAE) etc.[25]

Results of the Implemented Genetic Algorithm PID Controller

In the following section, the results of the implemented Genetic Algorithm PID Controller will be analyzed. The GA designed PID controller is initially initialized with population size of 20 and the response analyzed. It was then initialized with population size of 40, 60, 80 and 90. The response of the GA designed PID was then be analyzed for the smallest overshoot, fastest rise time and the fastest settling time.

4.2.3 Implementation using Particle Swarm optimization

The particle swarm optimization algorithms are based on two socio-metric principles. Particles fly through the solution space and are influenced by both the best particle in the particle population and the best solution that a current particle has discovered so far. The best particle in the population is typically denoted by (gobal best), while the best position that has been visited by the current particle is donated by (local best). That is, each particle is influenced by the very best performance of any member in the entire population. The (local best) individual is conceptually seen as the ability for particles to remember past personal success. The particle swarm optimization makes use of a velocity vector to update the current position of each particle in the swarm. The position of each particle is updated based on the social behavior that a population of individuals adapts to its environment by returning to promising regions that were previously discovered [26].

4.3 Simulation Results

a) Tuning PID paramteres using Genetic Algorithm

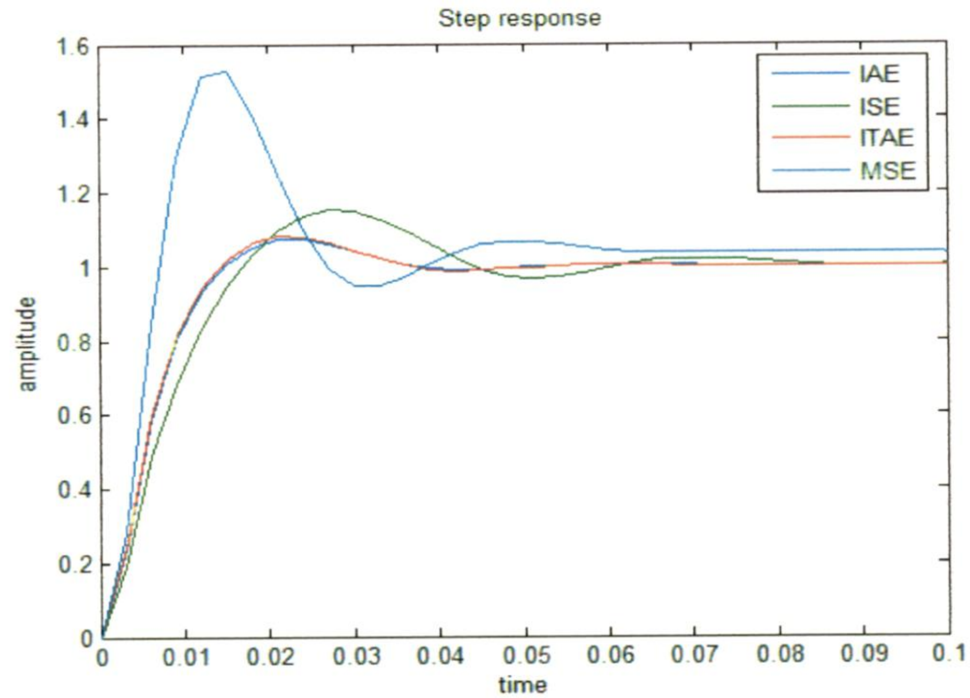

Fig.4.3.1 System step response, PID tuned with GA

System response shows that PID tuned using Genetic Algorithm has very less settling time, when compared with Zigler – Nicholas tuned PID controller. Also, steady state error reduces to zero in very short span of time.

Plot of various performance Index versus time shows that IAE is maximum, while ITAE is minimum.

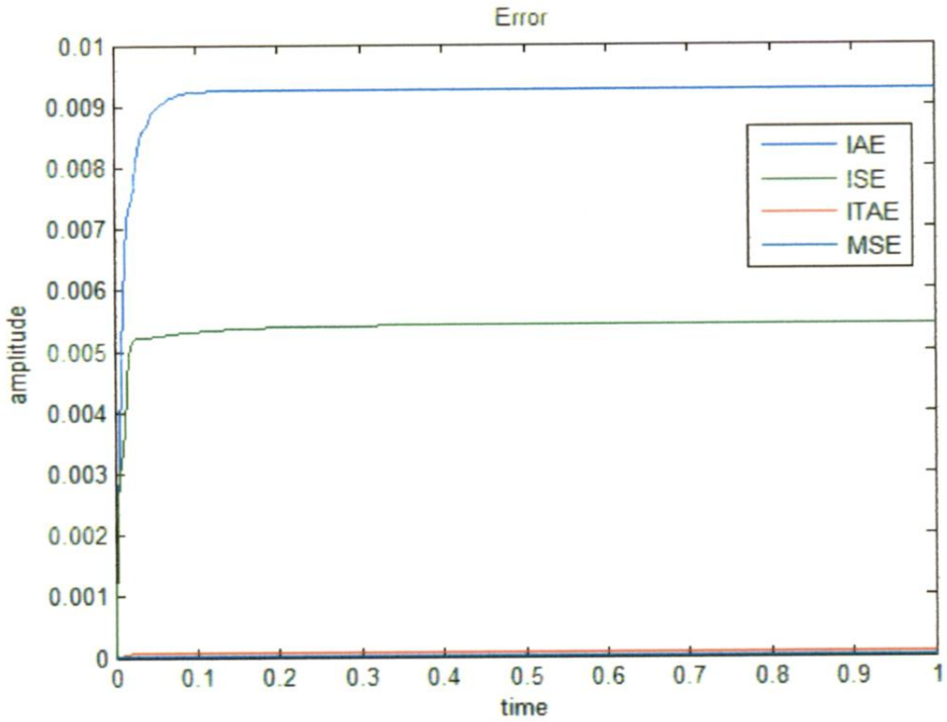

Fig.4.3.2 Plot of various errors with time (GA)

b) Tuning PID parameters using Particle Swarm Optimization

PID Parameter tuning using PSO involved assigning some random velocities and positions to n number of particles. Objective function which is MSE, IAE, ISE and ITAE is calculated for each particle, giving pbest and gbest. Using some parameters, velocity of each particle is updated so as to reach towards optimum value of objective function.

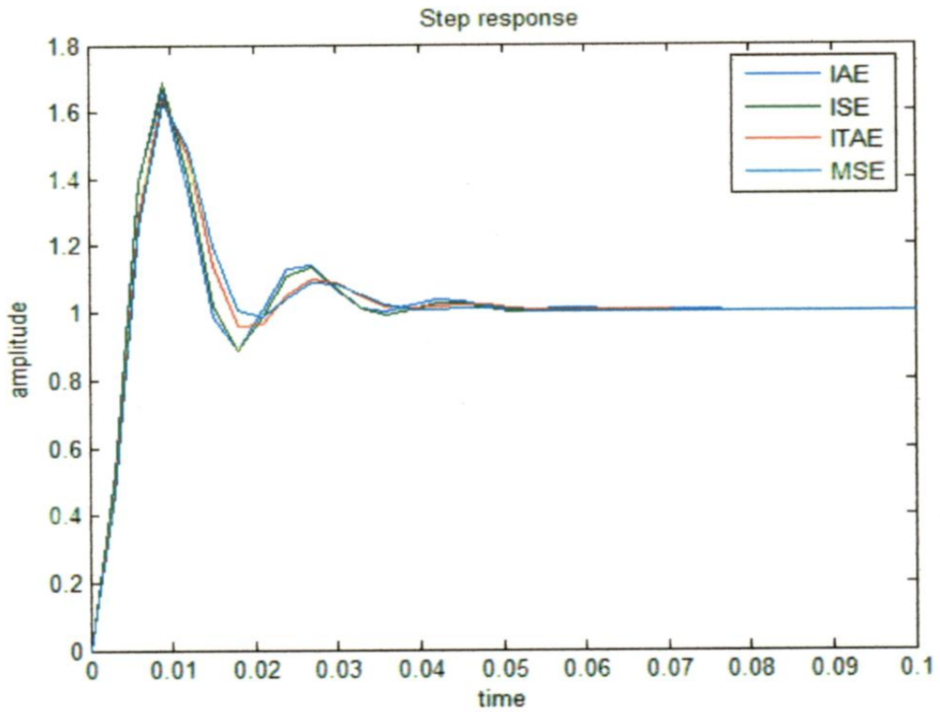

Fig. 4.3.3 System step response, PID tuned with PSO

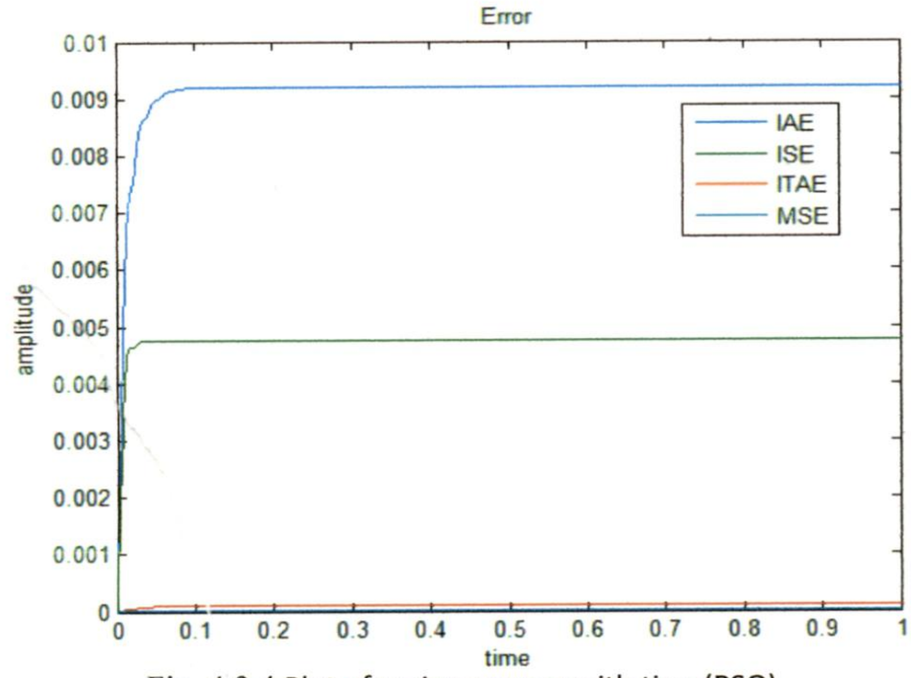

Fig. 4.3.4 Plot of various errors with time(PSO)

Results can be summarized in the following table, which shows that PSO performs better than Genetic Algorithm.

Table 4.3.1 Comparison between GA and PSO tuned PID controller results

	IAE	ISE	ITAE	MSE
GA	3.52561	2.20342	0.9346992	0.008336
PSO	3.12434	1.83247	0.7943232	0.006030

Chapter – 5

Fuzzy logic and ANFIS controller

Many decision-making and problem-solving tasks are too complex to be understood quantitatively, however, people succeed by using knowledge that is imprecise rather than precise. Fuzzy set theory, originally introduced by Lotfi Zadeh in the 1960's, resembles human reasoning in its use of approximate information and uncertainty to generate decisions. It was specifically designed to mathematically represent uncertainty and vagueness and provide formalized tools for dealing with the imprecision intrinsic to many problems. By contrast, traditional computing demands precision down to each bit. Since knowledge can be expressed in a more natural by using fuzzy sets, many engineering and decision problems can be greatly simplified.

Fuzzy set theory implements classes or groupings of data with boundaries that are not sharply defined (i.e., fuzzy). Any methodology or theory implementing "crisp" definitions such as classical set theory, arithmetic, and programming, may be "fuzzified" by generalizing the concept of a crisp set to a fuzzy set with blurred boundaries. The benefit of extending crisp theory and analysis methods to fuzzy techniques is the strength in solving real-world problems, which inevitably entail some degree of imprecision and noise in the variables and parameters measured and processed for the application. Accordingly, linguistic variables are a critical aspect of some fuzzy logic applications, where general terms such a "large," "medium,""small" have more meaning than numerical values.[27]

5.1 Fuzzy logic system

Fuzzy PID Controller Design Procedures From previous principle, we can design the fuzzy PID controller. The steps of designing fuzzy PID controller are as follow:[10]

(1) Determine the input and output variables of fuzzy controller, thus to determine the dimension of the controller. In general, the input variable can be selected as the system error and its variation. The output variables are PID parameters or their increment.

(2) According to the requirement, determine the variation range of each input, output variables, and then determine their quantification level, quantification factor and proportion factor.

(3) Define fuzzy subset in the quantification domain of each variable. First define the number of fuzzy subset to determine the language variable of each fuzzy subset, then choose the membership function for each variable.

(4) Determine the fuzzy rules. This is actually a group of operation experience. The fuzzy rules aim to make the system obtain optimum dynamic and steady state performance.

(5) Solve for the fuzzy control table. From fuzzy control rule and the input, output variables the fuzzy controller output can be obtained. These outputs are related with PID parameters. The fuzzy control table is formulated by putting the input, output variables in a table. PID parameters are independent; therefore there are 3 fuzzy control tables.

(6) Put sampling error and increment into fuzzy control rule table to obtain new PID parameters. The final output will be obtained by PID calculation.

(7) Analyze the fuzzy PID control performance according to simulation or experiment results. Then adjust the quantification and proportion factor to get the best result.

Realization [10]

Table 5.1.1 Fuzzy PID parameters

variables	e	e_c	Δk_p	Δk_i	Δk_d
language variables	E	EC	ΔK_p	ΔK_i	ΔK_d
basic domain	[-2 2]	[-1 1]	[-0.2 0.2]	[-0.001 0.001]	[-3 3]
fuzzy subsets	[NB NM NS ZO PS PM PB]				
fuzzy domain	[-3 3]	[-3 3]	[-0.2 0.2]	[-0.001 0.001]	[-3 3]
quantification factors	1.5	3	1	1	1

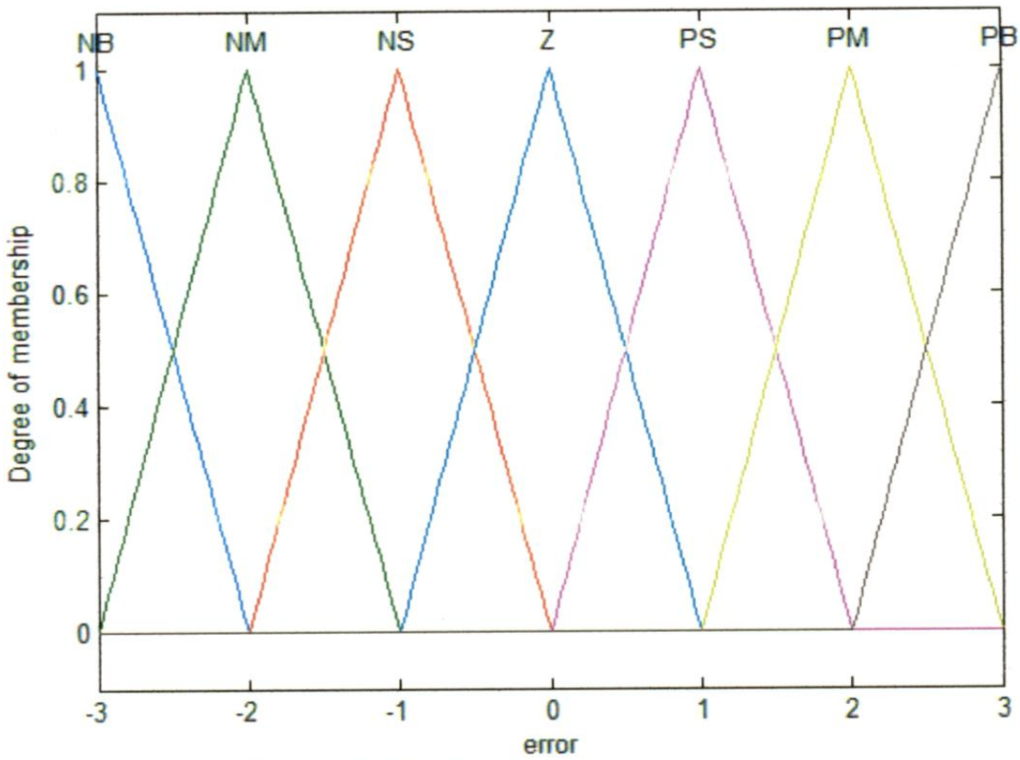

Fig. 5.1.1 Membership function for error

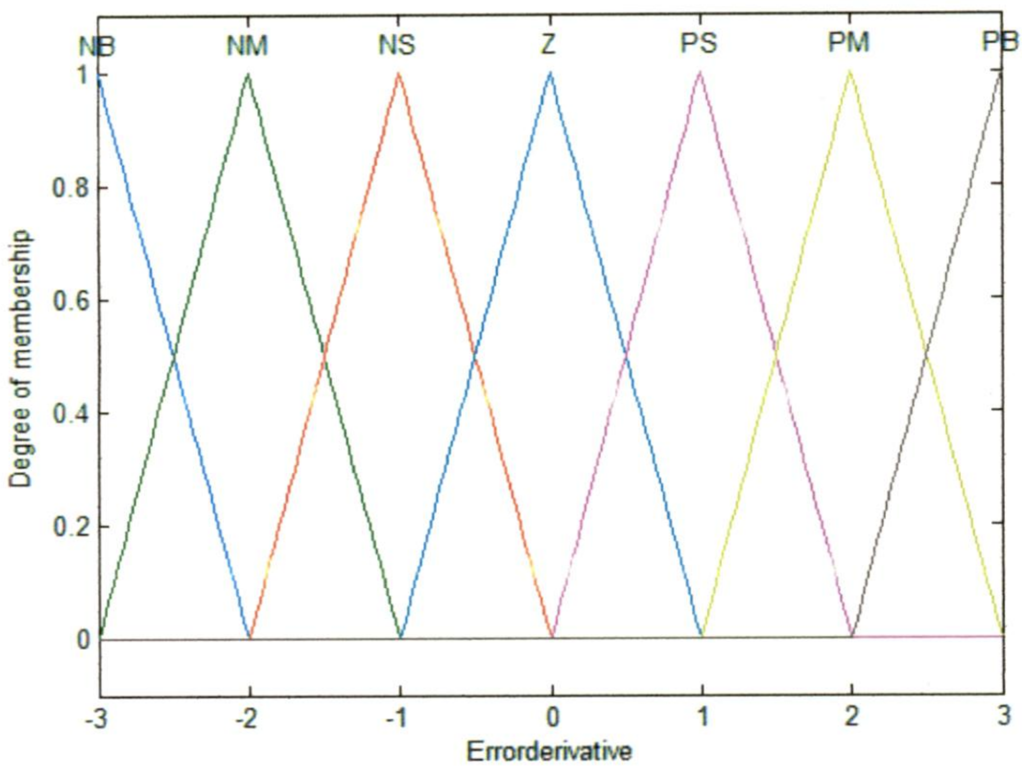

Fig.5.1.2 Membership function for derivative of error

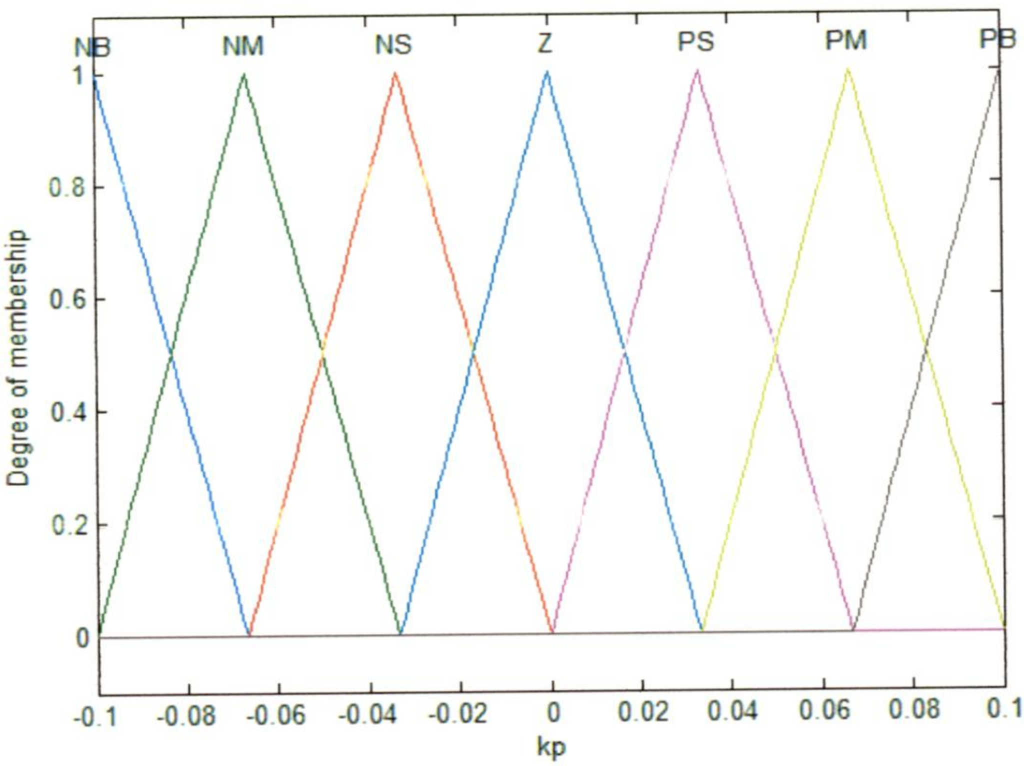

Fig. 5.1.3 Membership function for Proportional Gain

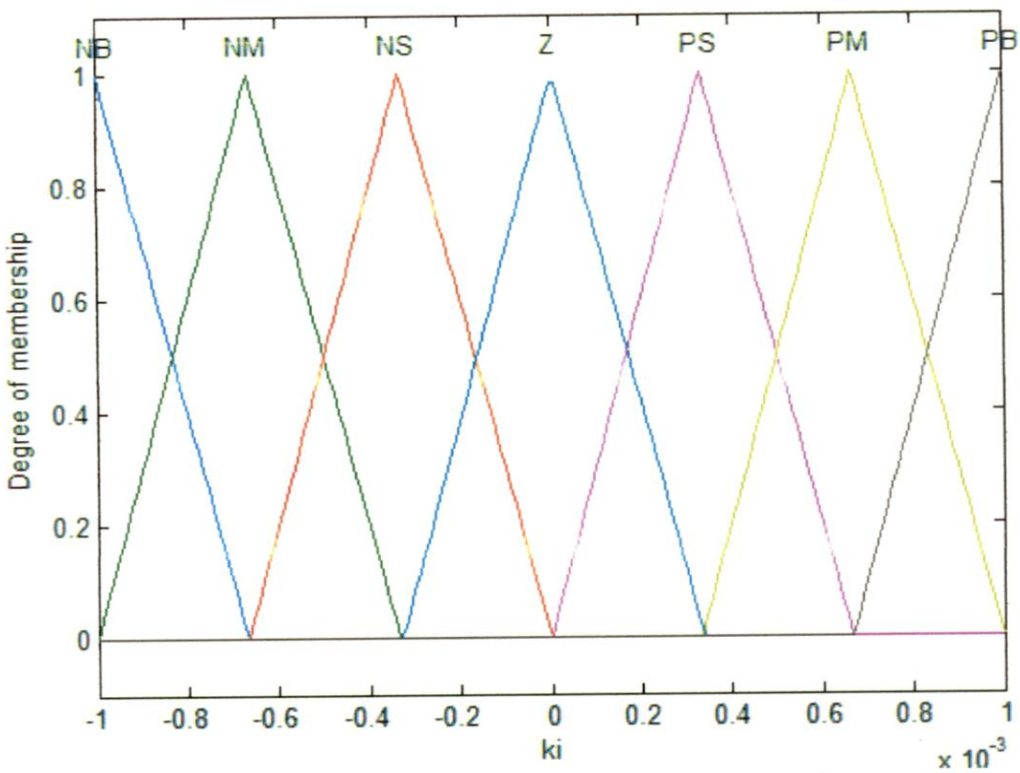

Fig. 5.1.4 Membership function for Integral Gain

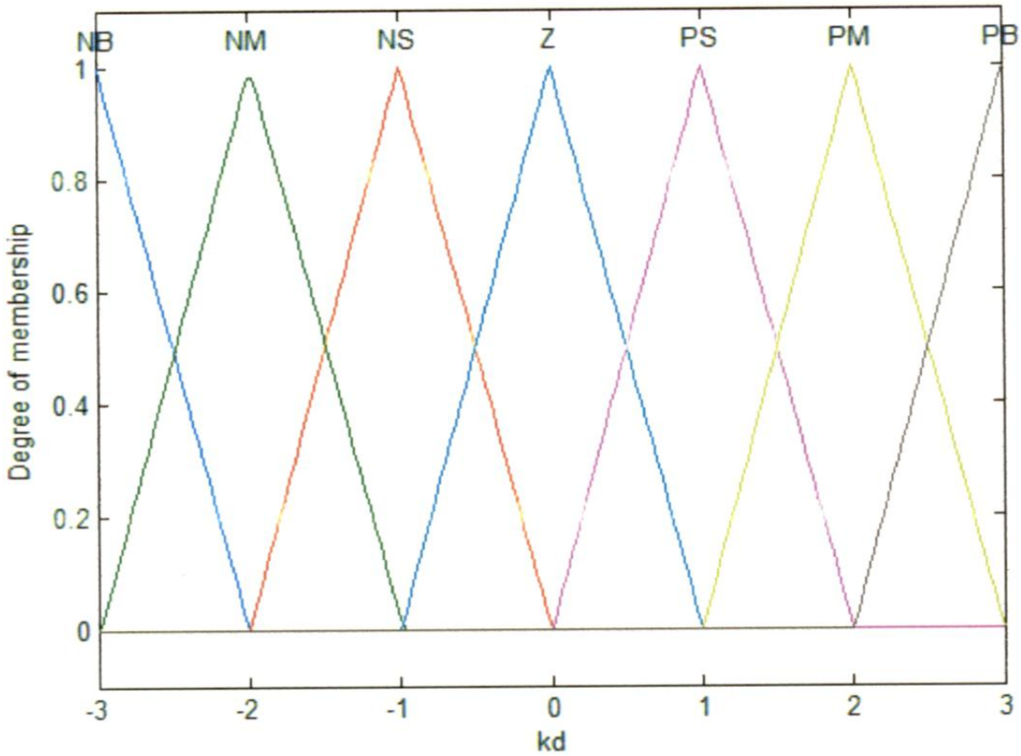

Fig. 5.1.5 Membership function for Derivative Gain

Simulink Architecture

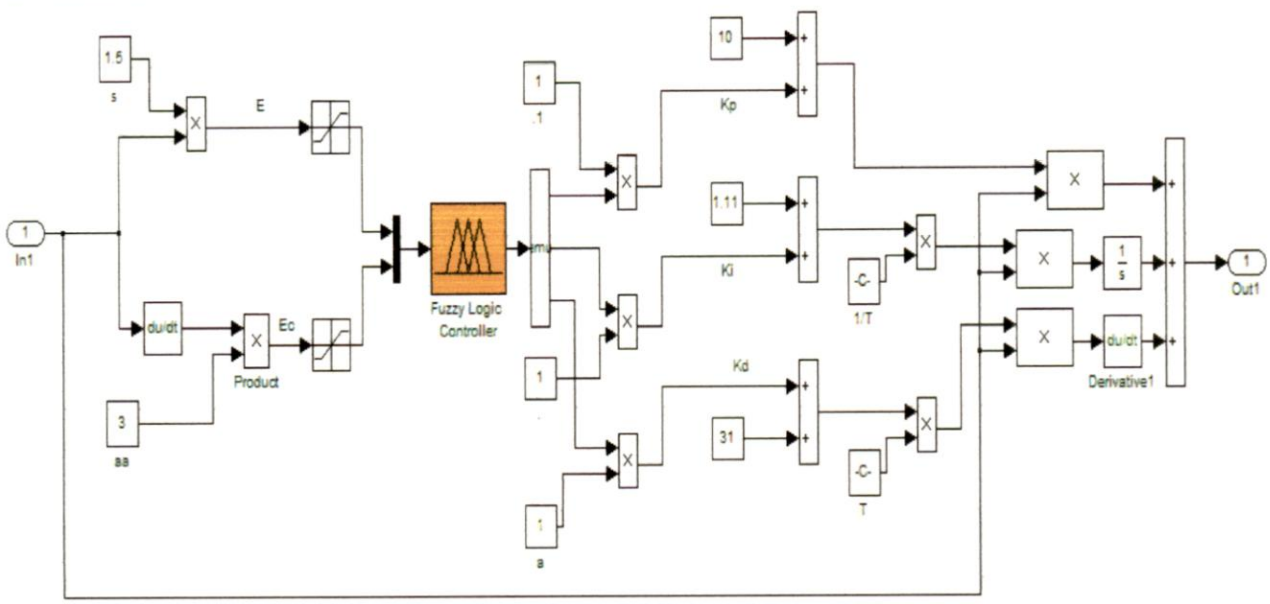

Fig. 5.1.6 Simulink Model for Fuzzy PID controller

5.2 ANFIS architecture

It is a hybrid neuro-fuzzy technique that brings learning capabilities of neural networks to fuzzy inference systems. The learning algorithm tunes the membership functions of a Sugeno-type Fuzzy Inference System using the training input-output data. The ANFIS is, from the topology point of view, an implementation of a representative fuzzy inference system using a BP neural network-like structure.[28] It consists of five layers. The role of each layer is briefly presented as follows: let O_i^l denote the output of node i in layer l, and x_i is the i^{th} input of the ANFIS, $i = 1, 2,...,p$. In layer 1, there is a node function M associated with every node:

$$O_i^1 = M_i(x_i) \tag{5.2.1}$$

The role of the node functions M_1, M_2,..., M_q here is equal to that of the membership functions $\mu(x)$ used in the regular fuzzy systems, and q is the number of nodes for each input. Gaussian shape functions are the typical choices. The adjustable parameters that determine the positions and shapes of these node functions are referred to as the premise parameters. The output of every node in layer 2 is the product of all the incoming signals:

$$O_i^2 = M_i(x_i) AND\ M_j(x_j) \tag{5.2.2}$$

Each node output represents the firing strength of the reasoning rule. In layer 3, each of these firing strengths of the rules is compared with the sum of all the firing strengths. Therefore, the normalized firing strengths are computed in this layer as:

$$O_i^3 = \frac{O_i^2}{\sum_i O_i^2} \tag{5.2.3}$$

Layer 4 implements the Sugeno-type inference system, i.e., a linear combination of the input variables of ANFIS, $x_1,x_2, ...x_p$ plus a constant term, $c_1,c_2, ...,c_p$, form the output of each IF –THEN rule. The output of the node is a weighted sum of these intermediate outputs:

$$O_i^4 = O_i^3 \sum_{j=1}^{p} P_j x_j + c_j \tag{5.2.4}$$

where parameters $P_1, P_2, \ldots, P_p$ and $c_1, c_2, \ldots, c_p$, in this layer are referred to as the consequent parameters. The node in layer 5 produces the sum of its inputs, i.e., defuzzification process of fuzzy system (using weighted average method) is obtained:

$$O_i^5 = \sum_i O_i^4 \tag{5.2.5}$$

The flowchart of ANFIS procedure is shown in Fig. 5.6. ANFIS distinguishes itself from normal fuzzy logic systems by the adaptive parameters, i.e., both the premise and consequent parameters are adjustable. The most remarkable feature of the ANFIS is its hybrid learning algorithm. The adaptation process of the parameters of the ANFIS is divided into two steps. For the first step of the consequent parameters training, the Least Squares method (LS) is used, because the output of the ANFIS is a linear combination of the consequent parameters. The premise parameters are fixed at this step. After the consequent parameters have been adjusted, the approximation error is back-propagated through every layer to update the premise parameters as the second step. This part of the adaptation procedure is based on the gradient descent principle, which is the same as in the training of the BP neural network. The consequence parameters identified by the LS method are optimal in the sense of least squares under the condition that the premise parameters are fixed.

5.3 Results

a) Fuzzy PID controller

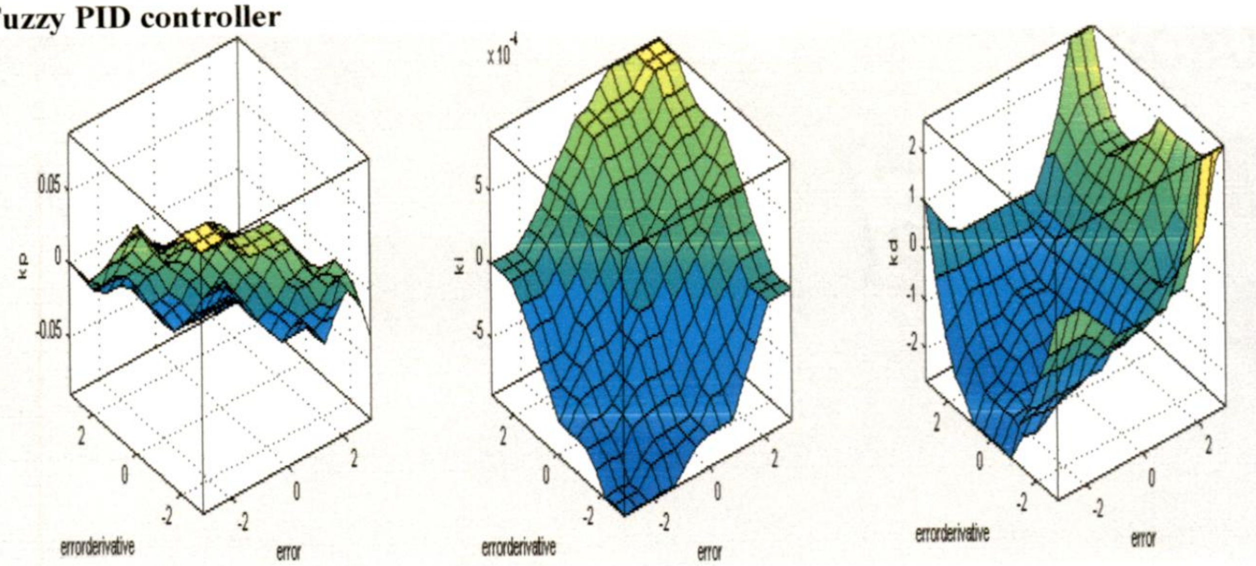

Fig. 5.3.1 Surfaces for Kp,Ki and Kd

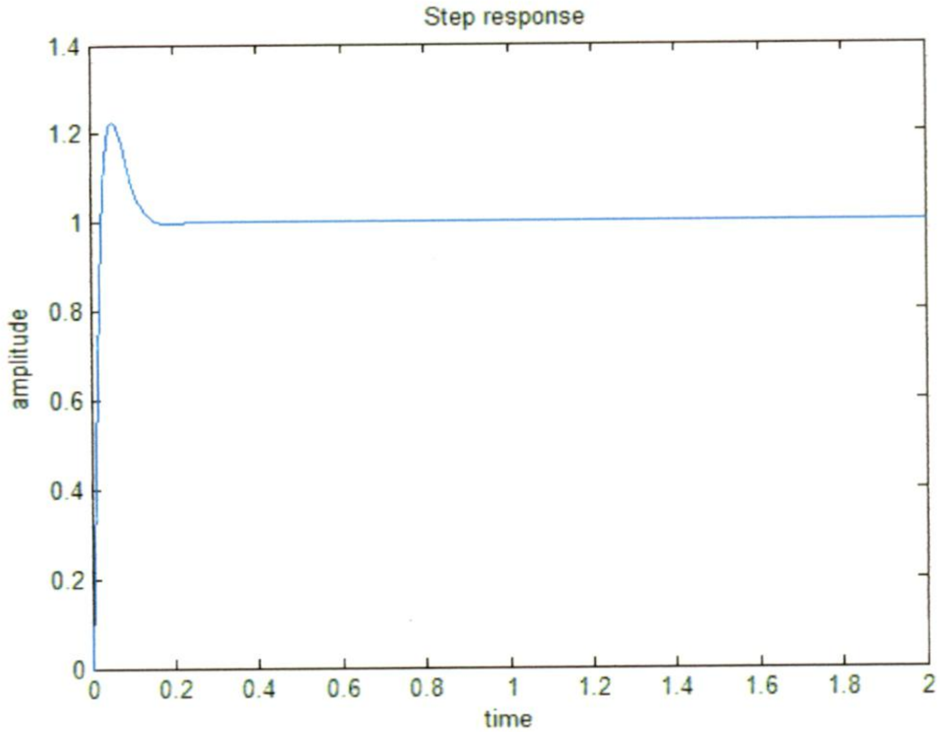

Fig. 5.3.2 Step Response of Fuzzy PID controller

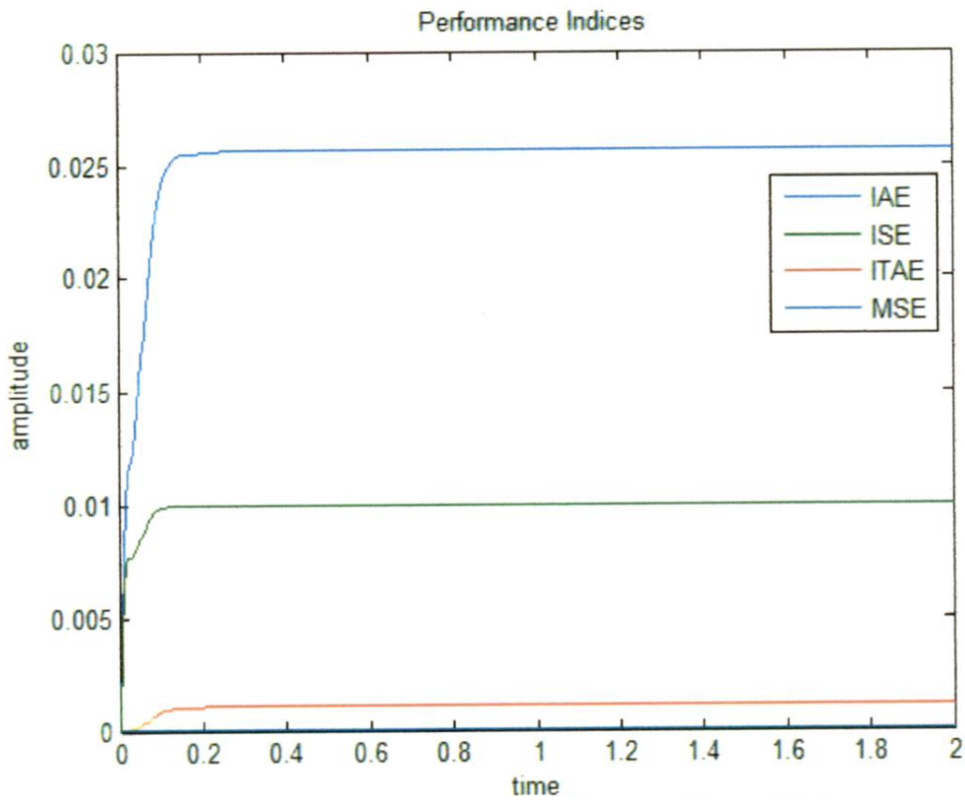

Fig.5.3.3 Plot of various errors with time(Fuzzy PID)

b) ANFIS control

ANFIS structure

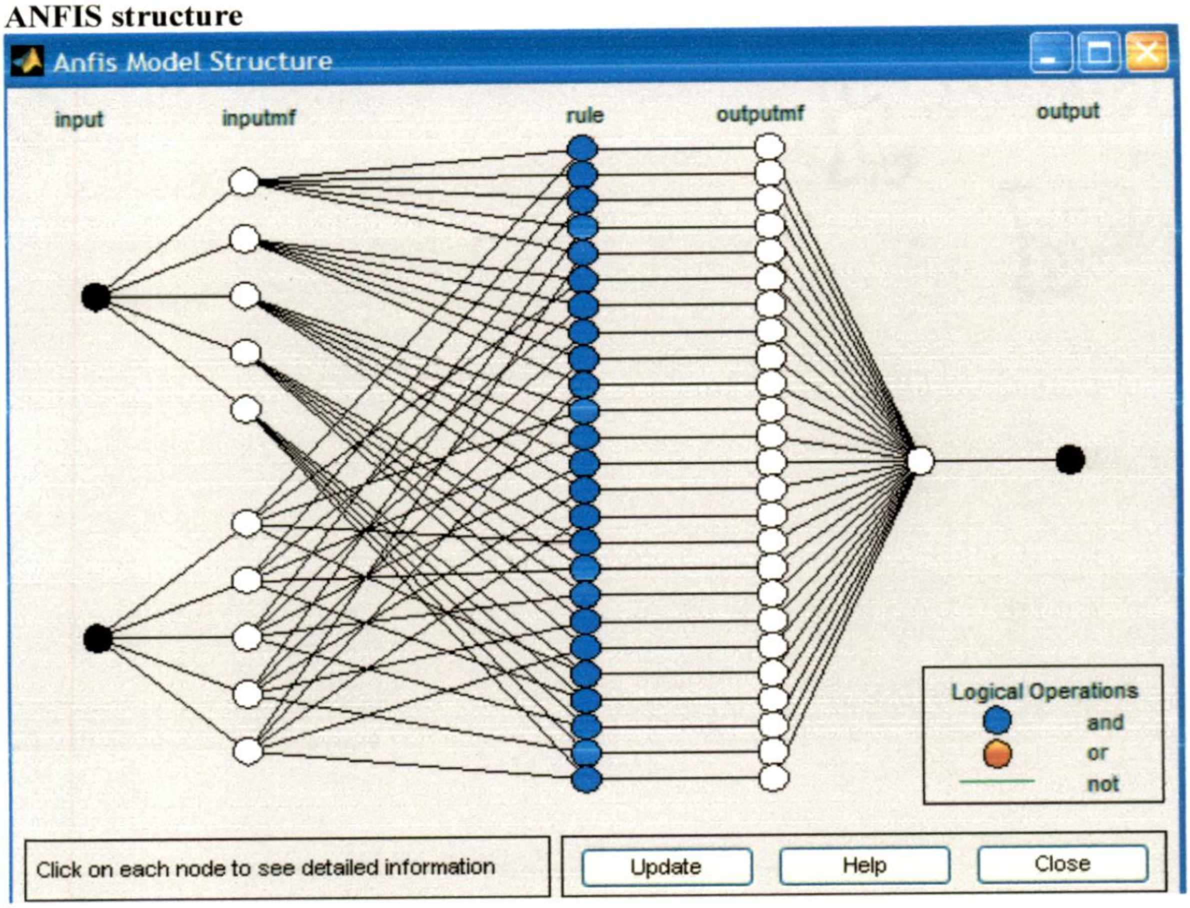

Fig. 5.3.4 ANFIS structure

From this architecture a Fuzzy inference system is being created, this system is exported to workspace in MATLAB window, and used as inference system in system employing following architecture.

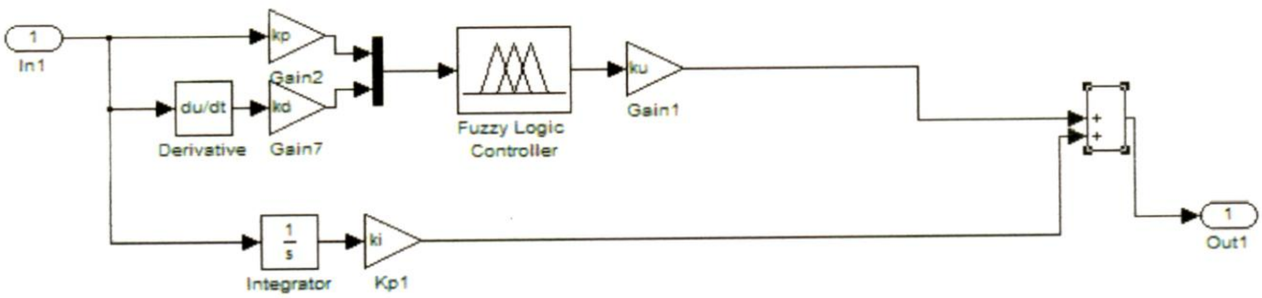

Fig.5.3.5 Fuzzy PD + linear I structure

Now, tuning of scaling parameters of the system is the next task. This is accomplished by applying Genetic algorithm to tune parameters of system, like Kp, Ki, Kd, Ku .
The results obtained for various performance indices are calculated and compared with those obtained using PID tuned using Genetic algorithm and PSO. Results showed that Fuzzy system are more efficient as compared to PID controller.

Table 5.3.1 Comparison between results of PID and Fuzzy PD+ linear I tuned using GA

architecture	IAE	ISE	ITAE	MSE
PID tuned using GA	3.52561	2.20342	0.9346992	0.0083362
Fuzzy PD +linear I tuned using GA	3.02512	2.16031	0.8654213	0.0068289

System step response shows that Fuzzy system provide output with less overshoot, but settling time is more as compared to PID controller system.

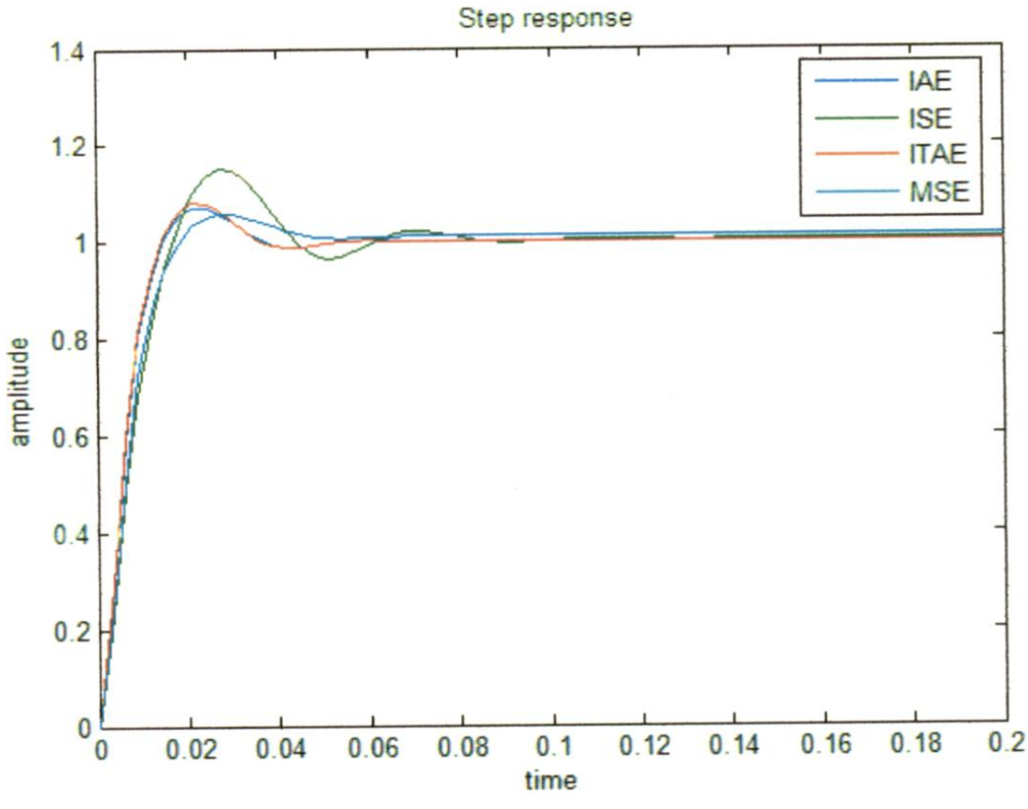

Fig. 5.3.6 Fuzzy PD+ linear I system Step Response

Chapter 6

Conclusion and Future Scope

Conclusion

The Magnetic levitation system being a highly unstable, nonlinear system needs controlling scheme which can control system uncertainties. Position of pole on the right half plane makes system unstable that any amount of compensation will not stabilize the ball in required position, and the effect of environmental disturbances will affect its position everytime.

Step response analysis reveals that there is a compromise always existing between steady state error and peak overshoot, settling time. Controller designed using root locus analysis gives more steady state error and settling time and peak overshoot are also high.PID controller gives better performance, but it has difficulty with properly tuning of the PID parameters value. Tuning PID parameters using Genetic Algorithm and Particle swarm optimization are analyzed.

On comparing between two optimization techniques, the simulation results with PSO techniques prove to be more effective than with GAs. In GAs, the limits defined by the number of parameters gives the search region while in PSO, the search region is independent of the number of parameters, given by the distance between the randomly selected initial position and the position corresponding to optimal fitness value. It is also observed that the speed of computation in PSO is very less in comparison to GAs.

Since better performance shown by PSO technique, combining this technique with other intelligent techniques, such as neural networks, expert systems, and fuzzy logic control systems open a new way to design and construct intelligence control systems adapted to complex real time systems.

Robustness is major advantage of Fuzzy logic controller than PID. Also, these are simpler to implement as no mathematical modelling is required and expert knowledge can be embedded in form of control rules. The achieved results showed that proposed ANFIS and fuzzy logic controllers are more robust to parameter variations when compared to the PID controller.

Future Scope:

Application of various new optimization techniques such as Genetic algorithm, PSO opens a doorway to development of intelligent systems, incorporating advantages of various schemes, like Fuzzy control, Neural control into single system. For future work, various Fuzzy parameters like membership function tuning using Genetic algorithm and PSO is suggested.

Appendix

Parameter values for Magnetic Levitation System

Parameter	**Value**	**Parameter**	Value
m	$22g$	x_0^*	$20.0mm$
Iron core diameter	ϕ22mm	Enameled wire diameter	ϕ0.8mm
R	13.8Ω	r	12.5mm (ball radius)
N	2450 circles	K	2.3142e-004 Nm^2/A^2
i_0^*	0.6105 A	K_f	0.25

REFERENCES

[1] Ono M., Koga S. and Ohtsuki H., “Japan’s superconducting maglev train,” *IEEE Instrum. Meas. Mag.*, vol. 5, no. 1, pp. 9–15, Mar. 2002.

[2] Chen M. Y., Wang M. J., and Fu L. C., “A novel dual-axis repulsive maglev guiding system with permanent magnet: modeling and controller design,” *IEEE/ASME Trans. Mechatronics*, vol. 8, no. 1, pp. 77–86, Mar. 2003.

[3] Rote D. M. and Cai Y., “Review of dynamic stability of repulsive-force maglev suspension systems,” *IEEE Trans. Magn.*, vol. 38, no. 2, pp. 1383–1390, Mar. 2002.

[4] Wang Q., Spronck P. & Tracht R., “An Overview Of Genetic Algorithm Applied To Control Engineering Problems”, Proceedings of the Second Conference on Machine Learning and Cybernetics, 2003.

[5] Hajjaji A. E. and Ouladsine M., “Modeling and nonlinear control of magnetic levitation systems,” *IEEE Trans. Ind. Electron.*, vol. 48, no. 4, pp. 831–838, Aug. 2001.

[6] Zadeh Lotfi A., “Fuzzy logic, neural networks, and soft computing”, Communications of the ACM, v.37 n.3, p.77-84, March 1994.

[7] Cipperfield A. Flemming P., and Fonscea C.,"Genetic Algorithms for Control System Engineering", in Proceedings Adaptive Computing in Engineering Design Control, pp-128-133, 1994.

[8] Sinha. N.K. and Gupta.M.M., “Soft Computing and Intelligent Systems: Theory and Applications”, Academic Press Editors, New York, 1999.

[9] Trisanto Agus, Yasser Muhammad, Lu Jianming, and Yahagi Takashi, “Implementation of a Fuzzy PID Controller Using Neural Network on the Magnetic Levitation System”,*IEEE Transactions on Fuzzy control.*

[10] Googoltech Magnetic Levitation system Experiment Manual.

[11] Astrom, Hagglund K., T., "*PID Controllers; Theory, Design and Tuning*", Instrument Society of America, Research Triangle Park, 1995.

[12] Abraham Ajith, Guo He and Liu Hongbo, "Swarm Intelligence: Foundations, Perspectives and Applications", studies in computational Intelligence (SCI) 26, 3–25 (2006).

[13] Chattopadhyay S (2006), "Soft Computing Techniques in combating the complexity of the atmosphere-a review", Arxiv preprint nlin/0608052.

[14] Jie Ma, Wenjun Fan, Fenghua He, "Parameters Self-adjusting Fuzzy PID Control in Magnetic Levitation System", IEEE Transaction on systems and control in aeronautics and aerospace,December 10-12, 2008,pp 1 – 5.

[15] Bong Joo Kim , Chung Choo Chung "*Design of Fuzzy PD + I Controller for Tracking Control*", Proceedings of the American Control Conference Anchorage, AK May 8-1, 0-7803-7298-0/02/ 0 2002 AACC p:2124-2129

[16] Takagi T., Sugeno M., "Fuzzy identification of systems and its application to modeling and control." IEEE Transactions on Systems, Man and Cybernetics, 15(1), 1985, 116-132.

[17] Gotshall,S. and Rylander, B., "Optimal Population Size And The Genetic Algoithm", Proc On Genetic And Evolutionary Computation Conference,2000.

[18] Luke S., Balan G.C. and Panait L, "Population Implosion In Genetic Programming", Department Of Computer Science, George Mason University.

[19] Sabar Nasser R., Ayob Masri, Kendall Graham and Qu Rong, "Roulette wheel Graph Colouring for Solving Examination Timetabling Problems", COCOA 2009: 463-470.

[20] Chipperfield A. J., Fleming P. J., Pohlheim H. and Fonseca C. M., "A Genetic Algorithm Toolbox for MATLAB" , Proc. International Conference on Systems Engineering, Coventry, UK, 6-8 September, 1994.

[21] Kennedy J., Eberhart R., (1995). "Particle Swarm Optimization",Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942-1948

[22] Poli, R. (2008). "Analysis of the publications on the applications of particle swarm optimisation",Journal of Artificial Evolution and Applications: 1-10.

[23] http://www.swarmintelligence.org/tutorials.php

[24] Houck Chris, Joines Jeff, and Kay Mike, "A Genetic Algorithm for Function Optimization: A Matlab Implementation" NCSU-IE TR 95-09,1995.

[25] Battaglia G.J. and Maynard J.M., "*Mean square Error: A Useful Tool for Statistical Process Management*" ,AMP J. Technol. 2, 47-55, 1992

[26] El-Telbany Mohammed El-Said, "Employing Particle Swarm Optimizer and Genetic Algorithms for Optimal Tuning of PID Controllers: A Comparative Study", Computers & Systems Department, Electronics Research Institute.

[27] Jantzen Jan, "Foundations of Fuzzy Control" , 2007 John Wiley & Sons, Ltd. ISBN: 0-470-02963-3

[28] Jang.J, "ANFIS: adaptive-network-based fuzzy inference system," IEEE Trans. Syst. Man Cybern. 23 (3) (1993) 665685.