

PLC BASED FUZZY LOGIC CONTROLLER FOR SPEED

A DISSERTATION

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

of

MASTER OF TECHNOLOGY

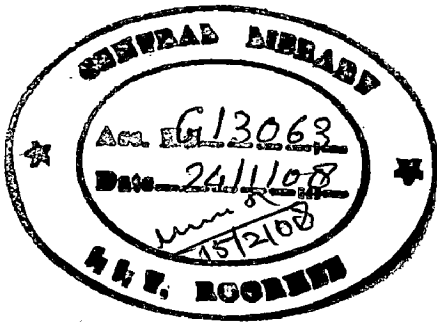
in

ELECTRICAL ENGINEERING

(With Specialization in Measurement and Instrumentation)

By

MAKWANA PRATAPBHAI JAGUBHAI



DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)
JUNE, 2007

12

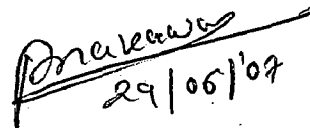
CANDIDATE'S DECLARATION

I hereby declare that the work presented in this dissertation titled "PLC BASED FUZZY LOGIC CONTROLLER FOR SPEED" submitted in partial fulfilment of the requirement for the award of the Master of Technology, in Electrical Engineering, with specialization in MEASUREMENT AND INSTRUMENTATION, in the Department of Electrical Engineering, Indian Institute of Technology, Roorkee, is an authentic record of my own work, carried out effect from June 2006 to June 2007 under the guidance of Dr. S. Mukherjee, Professor, Department of Electrical Engineering, IIT Roorkee and Dr. R. S. Anand, Associate Professor, Department of Electrical Engineering, IIT Roorkee.

The matter embodied in this thesis has not been submitted for the award of any other degree.

Date: 29/06/07


Place: Roorkee


29/06/07

(MAKWANA PRATAP BHAI JAGU BHAI)

CERTIFICATE

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


29.06.07

(Dr. R. S. Anand)
Associate Professor
Dept. of Electrical Engineering
Indian Institute of Technology,
Roorkee – 247 667, India.



(Dr. S. Mukherjee)
Professor
Dept. of Electrical Engineering
Indian Institute of Technology,
Roorkee – 247 667, India.

ACKNOWLEDGEMENTS

I take this opportunity to express my profound gratitude to **Professors, Dr. S. Mukherjee and Dr. R. S. Anand**, Department of Electrical Engineering, IIT Roorkee, under whose guidance this dissertation work was carried out. Without their support, encouragement and vigilant guidance completion of this dissertation would have been impossible.

I would like to acknowledge **Prof. Dr. H. K. Verma**, Group Leader, Department of Electrical Engineering, IIT Roorkee, **Prof. Dr. S. P. Gupta**, Head, Department of Electrical Engineering, IIT Roorkee and **Prof. Dr. Vinod Kumar**, Department of Electrical Engineering, IIT Roorkee, for providing all the facilities to make this Dissertation a successful one.

I am also deeply thankful to all the faculty members of **Measurement & Instrumentation** Group for their inspirational impetus, invaluable suggestions, constructive criticism and constant encouragement.

I am grateful to **Dinesh kumar**, lab assistant, Advanced Instrumentation Lab for providing me all computer and hardware facilities to carry out my dissertation work.

I dedicate this dissertation report to my wife **Aarti** and my parents without whom I have no origin.

(**MAKWANA PRATAP BHAI JAGU BHAI**)

Dedicated
to
My Family

ABSTRACT

Basically, Programmable Logic Controller (PLC) consists of input/output, memory, CPU and power supply. PLC executes program like a microprocessor device and changes output condition after reading input status. PLC is widely used in industries for sequential control and continuous control. In past, PLC was used for sequential control as a replacement of relay logic. Nowadays PLC provides variety of functions like, digital input/output, timers, counters, PWM output, analog input/output and P, PI, PID, 2/3 point regulation for continuous control.

Successful implementation of the fuzzy method for speed control of DC motor using a general-purpose programmable logic controller (PLC) is demonstrated in this work. Speed control of DC motors by a changing armature voltage is a common practice throughout industry at the present time. In addition, general-purpose PLCs are the most common controllers utilized for the control of industries and power plants. However, this work suggests a method for increasing the utilization level of an existing general-purpose PLC. In effect, the PLC is requested to perform the additional task of speed control using the elegant and effective fuzzy scheme. With a control scheme based on fuzzy methods, the speed control algorithm has been implemented within the standard PLC ladder logic.

Application of fuzzy logic controller in control systems proved to be superior to the conventional controllers like P, PI, PID controllers. PID type controllers do work fine when the process under control is in a stable and linear condition. Also they do not cope with the dead time of process. Fuzzy logic controllers can infer coherent results in dead time, nonlinear and unstable conditions.

It is very desirable to integrate conventional control engineering techniques, such as ladder logic or instruction list language for digital logic and PID control blocks tightly together with fuzzy logic functionality. In present work, fuzzy logic algorithm (fuzzy PI and fuzzy PID) is implemented on GE/7 PLC for continuous speed control using ladder logic.

3.2	Structure of a Fuzzy Controller.	22
3.2.1	Preprocessing	22
3.2.2	Fuzzification	23
3.2.3	Rule Base	23
3.2.4	Membership Functions	25
3.2.5	Inference Engine	26
3.2.6	Defuzzification	27
3.2.7	Postprocessing	29
3.2.8	Summary for Fuzzy Logic Processing	29
3.3	Fuzzy Logic Toolbox	30
3.4	Overview on Fuzzy Logic Controller Integrated on a PLC System	36
CHAPTER 4: PLC BASED FUZZY LOGIC CONTROLLER FOR SPEED		41
4.1	PLC based Fuzzy PI Controller for Speed	43
4.1.1	Fuzzification	44
4.1.2	Inference Engine	44
4.1.3	Defuzzification	45
4.2	PLC based Fuzzy PID Controller for Speed	47
4.2.1	Fuzzification	48
4.2.2	Inference Engine	48
4.2.3	Defuzzification	48
4.3	Experimental Results	51
CHAPTER 5: RESULTS AND DISCUSSIONS		53
CHAPTER 6: CONCLUSIONS AND SCOPE FOR FUTURE WORK		55
REFERENCES		57

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
Figure 1.1	Fuzzy logic controller operation	4
Figure 1.2	The PLC connected with process	5
Figure 2.1	PID module	10
Figure 2.2	Practical set up for speed control using PLC	11
Figure 2.3	Speed control using PID controller on PLC	12
Figure 2.4	Speed control module	12
Figure 2.5	PID control action performed by PLC for speed	14
Figure 2.6	PI algorithm on PLC for speed control	15
Figure 2.7	Speed control by PID algorithm on PLC for 1500 rpm	17
Figure 2.8	PLC based PID control for speed with load disturbance	17
Figure 2.9	Speed control by PI algorithm on PLC for 750 rpm	18
Figure 3.1	FLC for direct control	20
Figure 3.2	Fuzzy feed forward control	21
Figure 3.3	Fuzzy parameter adaptive control	21
Figure 3.4	Blocks of a fuzzy controller	22
Figure 3.5	Examples of membership functions	25
Figure 3.6	Graphical construction of the control signal in a fuzzy PD controller	26
Figure 3.7	One input, one output rule base with non-singleton output sets	28
Figure 3.8	Fuzzy logic MATLAB toolbox view	31
Figure 3.9	Membership function of input variables for a FLC	32
Figure 3.10	Membership function of output variable for a FLC	33
Figure 3.11	Fuzzy rule viewer	34
Figure 3.12	Output surface viewer	35
Figure 3.13	Fuzzy rule base editor	36
Figure 3.14	Fuzzy logic control system	37
Figure 3.15	Input data to a fuzzy logic system represented as counts and percentages	37

Figure 3.16	Output data from a fuzzy logic system represented as counts and percentages	38
Figure 3.17	Fuzzy logic system chart showing both input and output grades	39
Figure 3.18	Omron's fuzzy logic controller in a PLC system (The first fuzzy-PLC)	40
Figure 4.1	Membership functions for input variables (E, CE & SE)	42
Figure 4.2	Membership function for output variable (CV)	43
Figure 4.3	An example for LOM defuzzification performed by PLC	45
Figure 4.4	Flow chart of fuzzy PI control for speed realized by PLC software	46
Figure 4.5	Flow chart of fuzzy PID control for speed realized by PLC software	50
Figure 4.6	Fuzzy PID control for 1500 rpm speed realized by PLC software	51
Figure 4.7	PLC based FPID control for speed with load disturbance	51
Figure 4.8	Fuzzy PI control for 1500 rpm speed realized by PLC software	52

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
Table 3.1	A rule base in tabular format	24
Table 4.1	Rule base for fuzzy PI controller	44
Table 4.2	Output variable (CV) action for FPI controller	44
Table 4.3	Rule base for fuzzy PID control	47
Table 5.1	Comparison between conventional and fuzzy logic based speed control realized by PLC software	53

LIST OF ABBREVIATIONS

PID	Proportional-integral-derivative
PLC	Programmable logic controller
GE	General electric (PLC manufacturer)
PI	Proportional-integral
HMI	Human machine interface
RISC	Reduced instruction set computing
Fuzzy PLC	A fuzzy logic controller integrated with a PLC
FLC	Fuzzy logic controller
OMRON	A PLC manufacturer company
HSC	High speed counter
PWM	Pulse width modulated
PTO	Pulse train output
LSB	Least significant bit
PID-ISA	ISA standard PID algorithm
Max-min	A fuzzy inference process
COG	Center of Gravity defuzzification
COGS	Center of Gravity defuzzification method for singletons
LOM	Last of maximum defuzzification
MOM	Mean of maximum defuzzification
FOM	First of maximum defuzzification
LM	Left most maximum defuzzification
RM	Right most maximum defuzzification
GUI	Graphical user interface
FPI	Fuzzy PI control
FPID	Fuzzy PID control

INTRODUCTION

In the majority of all applications involving rotating machinery, a speed controller must be present to govern the speed of the device. Currently, most speed control problems, rotational or linear, use a dedicated digital or microprocessor based controller implementing some combination of PID control scheme. PID based controllers are typically very reliable; however, they often have several limitations that can drastically reduce their performance. These limitations include:

- Most often the PID speed controllers are implemented using a dedicated controller, which can increase the cost of the overall control system.
- The response of standard PID controllers can be severely limited in applications where the process environment is constantly changing. An example of such an environment is that of a steam turbine/generator, where temperatures, levels, and pressures are constantly changing the dynamics of the system.
- The PID controllers are often limited in highly nonlinear cases, dead time and do not respond quickly with the required accuracy for obtaining acceptable control.
- The PID controllers require the system designer to develop an extensive mathematical model of the control process, which is not always easily achieved.

The need for a stand-alone device, one of the main downfalls of PID controllers, can easily be eliminated by implementing the control scheme from within a general purpose programmable logic controller (PLC). Utilizing a PLC as the primary cerebrum is a common practice in industrial plants. However, in order to accomplish PID control from within a PLC, one frequently needs a PID coprocessor module that can be programmed. These modules, which are often obtained from a company other than the PLC manufacturer, can add a significant amount to the overall PLC cost and therefore work against the goal of reducing the cost of the control system. In addition, the performance of the controller is still hindered by the limitations of PID control [1, 2].

On the other hand, in highly nonlinear cases where the process is relatively slow (changes take place on the order of milliseconds rather than microseconds), control schemes based on fuzzy methods often allow much simpler controller design than their

classical PID counterparts. Accurate and responsive control can be achieved using fuzzy control methods without the need for highly involved mathematical modeling and an additional PID coprocessor module.

Thus the thought is to eliminate the need for these standalone controllers and implement the speed controllers from within the PLC using a control scheme other than PID. Due to the inherently slow nature of most process type applications where a general purpose PLC is being utilized, fuzzy control methods offer a desirable alternative to PID controllers [3, 4, 5].

1.1 Fuzzy Control

The recent trend in global manufacturing scenario has resulted in the emergence of many innovative techniques that have revolutionized the manufacturing industry as a whole. A lot of states of art advances have been associated with the increasing use of microprocessors in advanced manufacturing systems. Out of many the one, which has predominantly captured the real time, behavioral use is the fuzzy logic.

Fuzzy logic implementation in real time environment involves the integration of multi-disciplinary area like computer engineering, mechanical and electronic systems tailored towards smart systems that enable a very precise control on the process. The fuzzy logic based project involves the programming of microprocessor for the control of sequence of operation depending upon the change in the values of environment variables.

Fuzzy control methods are based on the idea of fuzzy sets brought about in 1965 by Lotfi Zadeh of the University of California, Berkeley [6, 7]. Fuzzy systems are knowledge-based (or rule-based) systems that use a set of fuzzy rules of the If-Then form to determine the output of the controller. So an engineer does not have to develop an extensive mathematical model for the process of concern to design a successful controller. This is especially time saving when the plant model is highly nonlinear and the differential equations describing the plant are not easily obtained. In fuzzy based control, instead of using mathematics to model the plant, linguistic terms are used to create fuzzy subsets, a fuzzy rule base, and a fuzzy output. The fuzzy rule-base is developed from one or more "expert's" knowledge concerning the device or process that is to be controlled [8].

In conventional control theory, the process being controlled must be mathematically modeled so that a controller can be designed. However, a given system may be ill-defined or too complex to be accurately modeled and thus requires ad hoc controller design methods. An alternative to the ad hoc design methods is to employ fuzzy logic to the control process [8]. Fuzzy control is based on the concept of the fuzzy algorithm, introduced by Zadeh in 1973. Zadeh argued that the conventional quantitative techniques of system analysis are unsuited for systems of high complexity. He defined two major components of the fuzzy algorithm, the linguistic variable and the fuzzy rule, as a means of approximating the behavior of complex systems. A linguistic variable is a variable, written in a natural language format, which represents imprecise information. For example, the terms short, normal, and full might be considered values of a linguistic variable height. Fuzzy rules are conditional statements of the form If A then B, where A and B are fuzzy variables. The fuzzy algorithm is then an ordered sequence of commands in which some of the commands are fuzzy rules [6, 8].

The fuzzy logic foundation is based on the simulation of people's opinions and perceptions to control any system. One of the methods to simplify complex systems is to tolerate to imprecision, vagueness and uncertainty up to some extent. An expert operator develops flexible control mechanism using words like "suitable, not very suitable, high, little high, much and far too much" which are frequently used words in people's life. Fuzzy logic control is constructed on these logical relationships. Fuzzy sets are used to show linguistic variables.

Although the classical controllers depend on the accuracy of the system model and parameters, FLC uses different strategies for motor speed control. Basically, FLC process is based on experiences and linguistic definitions instead of system model. It is not required to know exact system model to design FLC. In addition to this, if there is not enough knowledge about control process, FLC may not give satisfactory results.

For a general fuzzy controller, the fuzzy logic control process will be performed by main three actions as shown in Fig. 1.1. The three main actions performed by a fuzzy logic controller are:

- Fuzzification
- Fuzzy processing

- Defuzzification

As shown in Fig. 1.1, when the fuzzy controller receives the input data, it translates it into a fuzzy form. This process is called fuzzification. The controller then performs fuzzy processing, which involves the evaluation of the input information according to If-Then rules created by the user during the fuzzy control system's programming and design stages. Once the fuzzy controller finishes the rule-processing stage and arrives at an outcome conclusion, it begins the defuzzification process. In this final step, the fuzzy controller converts the output conclusions into "real" output data (e.g., analog counts) and sends this data to the process via an output module interface. If the fuzzy logic controller is realized by PLC ladder logic program [9, 10] and does not have a direct or built-in I/O interface with the process, then it will send the defuzzification output to the PLC memory location that maps the process's output interface module.

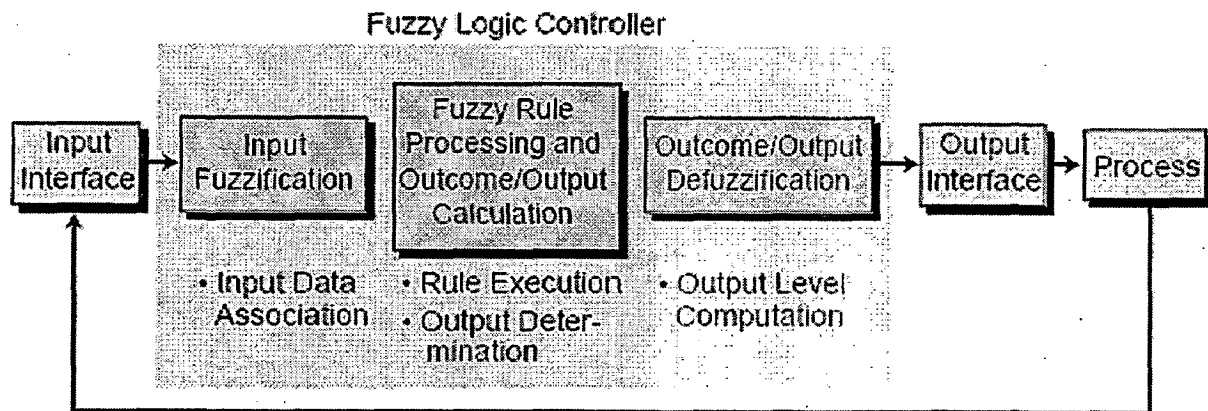


Fig. 1.1. Fuzzy logic controller operation.

1.2 Speed Control using PLC based Fuzzy Logic Controller

When a process is controlled by a PLC, it uses inputs from sensors to make decisions and update outputs to drive actuators, as shown in Fig.1.2. The process is a real process that will change over time. Actuators will drive the system to new states (or modes of operation). This means that the controller is limited by the sensors available, if an input is not available, the controller will have no way to detect a condition.

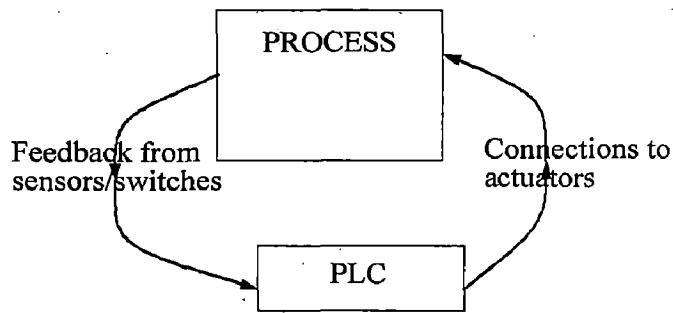


Fig. 1.2 The PLC connected with process

The control loop is a continuous cycle of the PLC reading inputs, solving the ladder logic, and then changing the outputs. Like any computer this does not happen instantly.

The most commonly used control systems today are based on the proportional-integral-derivative (PID) method. The PID method controls a system by reading the input sensors and applying a mathematical equation to the inputs to produce the output. Typically the output from this type of system is used to control some kind of actuators to regulate the flow of something. One example of such actuator is a speed control module which is used to control speed of DC servomotor. A common industrial practice is to purchase a PID unit, connect it to a process, and tune it through trial and error. But here this task is done by PID action performed by GE/7 PLC [9]. The speed control module provides interface between PLC and DC servo motor. The speed is controlled from 0 to maximum up to 1500 rpm using PLC software (ladder diagram/instruction list) [9, 11].

In this work, the effort has been made on the analysis of the conventional (PID) techniques and fuzzy logic based techniques for speed control of DC servo motor. The light is thrown on comparison between traditional (PID) methods and fuzzy methods.

Ladder program has been developed on GE PLC using PID algorithm. Additional features are available such as manual control and PI control [9, 11, 12, 13]. Ziegler-Nicholas tuning is used for PID tuning which is specified in GE PLC manual [1, 9]. It has been emphasized through results that speed control using PID algorithm of GE PLC has responded slowly to the setpoint variations and disturbances near to the set speed of DC motor. The above limitations can be eliminated by designing proper fuzzy logic controller on GE PLC through ladder logic, whose knowledge base is derived from

complete knowledge of the process of the speed control of permanent magnet DC motor [14].

HMI panel is available in order to allow the operator to communicate with the PLC by changing certain control characteristics and setpoints of the fuzzy speed controller before execution of the PLC ladder logic program. The HMI communicates with the PLC via an RS-232 serial communication link and can be used to transmit and receive data to and from the PLC [9, 11, 12, 13].

The system can be controlled by fuzzy control algorithms formed by basic PLC instructions. In this work, Max-min method is used for inference and Center of Gravity method is used for defuzzification [15, 16]. Triangular fuzzy set is considered for inputs and output. Input variables are chosen as speed error, change of error and sum of error while output variable is considered as control variable voltage from PLC. PLC acquires data from the addresses which are assigned for its inputs, fuzzifies and infers by comparing with the rule base written before, it then writes the fuzzified result to the addresses which are assigned for its outputs. Formation of fuzzification, inference and defuzzification algorithms can be done by basic PLC instructions such as multiply, divide, add, subtract, min, compare, move etc.

Fuzzy PI control is known to be more practical than fuzzy PD control because it is difficult for the fuzzy PD controller to remove steady state error [14]. The fuzzy PI control, however, is known to give poor performance in transient response for higher order processes due to internal integration operation. Thus, in practice the fuzzy PID controllers are more useful.

It is emphasized by comparing the results of conventional controllers (PID) and fuzzy (PI and PID) controllers that speed of the DC motor is controlled in few seconds with removal of disturbances. The on-line analysis of speed of DC motor is possible through LABVIEW software and hardware. The output speed is also indicated on rpm indicator of speed control module.

1.3 Fuzzy Logic Applications: State of the Art

Fuzzy logic was given the real meaning by Dr. Lotfi Zadeh in the year 1965 [6]. He viewed it as advanced form of multi valued logic. According to him fuzzy logic deals with more approximate reasoning rather than precise modes of the same, making it a

decision making tool over a range of transitional values instead of precise ones. According to Gupta [7], it can be stated as a superset of conventional (Boolean) logic that has been extended to handle the concept of partial truth which can be stated in other words as truth value between completely true and completely false.

Fuzzy logic foundation is based on the simulation of people's opinions and perceptions to control any system. One of the methods to simplify complex systems is to tolerate to imprecision, vagueness and uncertainty up to some extent [17]. Although the classical controllers depend on the accuracy of the system model and parameters, FLC uses different strategies for motor speed control. Basically, FLC process is based on experiences and linguistic definitions instead of system model. It is not required to know exact mathematical model of system to design FLC [18].

As stated by Mamdani [8], "Fuzzy logic is successful because it replaces the classical PID controller when tuned to required parameters; the parameters of a PID controller affect the shape of the entire control surface. Since the fuzzy logic controller is a rule based controller, the shape of the control surface can be individually manipulated for the different regions of the state space, thus limiting possible effects to neighboring regions only". Hogener.J [19] emphasized that fuzzy logic has become a universal technology and its application ranges over the entire spectrum of electronic control. The development in software tools enables us to implement fuzzy logic system on almost any target hardware, ranging from low cost 8-bit micro controller that have only few bytes of RAM and few hundred bytes of ROM up to distributed process control system that uses multiple 64 bit RISC processors or PLCs.

Application of fuzzy logic has extended to a large number of control features of industrial automation. They can extend from anti-sway control of cranes to wind energy converter machines. Recently, a 64-ton crane that transports concrete modules for bridges and tunnels over a distance of 500 yards has been automated with fuzzy PLC in Germany. The benefit was capacity gain of about 20% due to faster transportation and an increase in safety. The crane was commissioned in spring 1995 and the crane operator has continuously enabled the fuzzy logic anti-sway controller [4, 20].

Fuzzy PI control is known to be more practical than fuzzy PD because it is difficult for the fuzzy PD to remove steady state error. The fuzzy PI control, however, is

known to give poor performance in transient response for higher order processes due to the internal integration operation. Thus, in practice the fuzzy PID controllers are more useful [14]. In electric motor drives and motion control areas, fuzzy logic has been recently proposed to control different systems such as static power converter, induction and dc servo motors, robots and ac servo. Speed controller of DC motors is carried out by means of voltage control frequently because DC motors are used in many applications such as still rolling mills, electric trains, electric vehicles, electric cranes and robotic manipulators require speed controllers to perform their tasks [21, 22, 23].

1.4 Organization of Thesis

Chapter 1: Describes the limitation of PID controllers. It gives an introduction on fuzzy control including the state of art and applications of fuzzy logic. It gives brief introduction on fuzzy logic controller implemented on PLC for speed control of DC motor.

Chapter 2: Describes speed control of DC motor using PID controller realized by GE PLC after discussing the PLC specifications. It also describes the responses and the flow charts for speed control of DC motor using conventional (PI and PID) control on PLC.

Chapter 3: Gives an overview on fuzzy logic control including different control schemes. It also gives an introduction on MATLAB toolbox for FLC. It describes briefly an example of temperature control performed by a FLC module integrated with OMRON's fuzzy PLC.

Chapter 4: Describes steps for the fuzzy control realization on PLC using ladder logic. It describes different FLC operations for speed control using PLC software likewise fuzzification, inference and defuzzification. The flow charts and the responses, of fuzzy PI and fuzzy PID algorithms realized by PLC software for speed control of DC motor, are also described.

Chapter 5: Summarizes results by comparing conventional and fuzzy control realized by PLC for speed.

Chapter 6: Describes conclusions and directions of future works.

SPEED CONTROL USING PID ON PLC

Programmable Logic Controllers are microprocessor based devices which manage a machine or an industrial foundation by means of change of situations and a program edited before. Basically, a PLC is composed of a microprocessor, a power supply, memory and input/output units. Beside these units mentioned above, some certain units were developed which respond to requirement of faster and more effective processing in industrial controlling systems. These special units, such as high speed counters, heat controllers and pH controllers, increase the controlling performance of PLCs. The early logical control was based on relays. It is common to use relays to make simple logical control decisions. The development of low cost computer has brought the most recent revolution, the Programmable Logic Controller (PLC) [9, 10, 11, 12, 13].

2.1 Specifications of 23-Point MICROPLC (GE/7 PLC)

1. **AC Power:** The device is operated with power range of 200V to 240V AC with 10% variations. The frequency range is 50 to 60 Hz with 5% variations.
2. **DC Inputs (Discrete Inputs):** Total 13 numbers of DC inputs are available on GE PLC which accept 24V DC. The input voltage range is 0 to 30V. The 24 V DC inputs can be operated in positive or negative logic. Current into an input point results in a logic 1 in the input status table (%I). Input characteristics are compatible with a wide range of input devices, such as pushbuttons, limit switches and proximity switches.
3. **DC Output (Discrete Output):** One output (Q1) is a 24V DC transistor output. It can be used as a normal DC output or as a high speed counter (HSC) controlled output, pulse train output (PTO), or pulse width modulated (PWM) output.
4. **Relay outputs (Discrete Outputs):** 9 isolated relays are available with 2-amp current capacity; normally open outputs can control devices such as motor starters, solenoids and indicators. AC or DC power to operate field devices must be supplied externally. External fusing is recommended to protect the relay contacts. Relay outputs can be configured as regular outputs or as outputs controlled by HSCs (high speed counters) [9, 11]. They can't be used as PTO (pulse train output) or PWM outputs [9, 10, 11, 12, 13].

5. **High speed counters:** The MicroPLC can be configured to provide built in high speed counter and pulse operation. High speed counter operation can be set up as up to 4 HSCs. Each counter can be enabled independently and provides rapid pulse signals up to 10 kHz for industrial control application.

6. **Analog (I/O):** The MicroPLC provides 2 analog input channels (AI0018 and AI0019) that can be configured to accept inputs from 0 to +10V or from 0 to 20 mA or 4 to 20 mA input signals. The module also has an analog output (AQ0012) configurable for same voltage or current ranges. Resolution is 12 bits (1LSB = 2.5 mV) for 0 to 10 V range or 12 bits (1LSB = 5 μ A) for 0/4 to 20 mA range. The PLC counts from 2 input channels of 12 bit ADC into values (0 to 32000) in references AI0018 and AI0019. To generate the analog output, the value in AQ0012 (0 to 32000) is translated into a count value for 12 bit DAC, which drives the analog output.

2.2 Analysis of PID Controller

The PID task is the core of the application, where the controller calculation is actually made. The PID controller receives as input the error, given by

$$e(t) = r(t) - y(t) \quad (2.1)$$

And controller produces the control variable $c(t)$ that is its output.

The PID controller has three terms: (a) the proportional term P corresponding to proportional control, (b) the integral term I giving a control action that is proportional to the time integral of the error, and (c) the derivative term D, proportional to the time derivative of the error. The control signal $c(t)$ is calculated as follows:

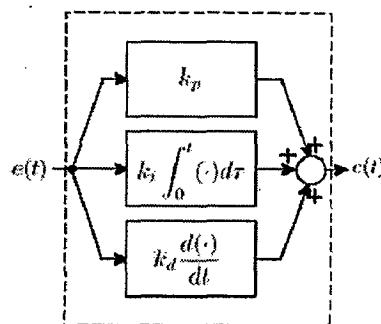


Fig. 2.1: PID module

$$c(t) = k_p e(t) + k_i \int_0^t e(t) dt + k_d \frac{de(t)}{dt} \quad (2.2)$$

It is common to define $K_p=K$, $k_i=K/T_i$ and $k_d=KT_d$. Then the following well-known version of the equation arises:

$$c(t) = K \left(e(t) + \frac{1}{T_i} \int_0^t e(t) dt + T_d \frac{de(t)}{dt} \right) \quad (2.3)$$

It is necessary to discretise the controller, that is, to approximate the integral and derivative terms to forms suitable for computation by a controller. From a purely numerical point of view we can use:

$$\frac{de(t)}{dt} \approx \frac{e(t) - e(t-1)}{T_c} \quad (2.4)$$

$$\int_0^t e(t) dt \approx \sum_{n=0}^k e(n) \quad (2.5)$$

Then, the discrete PID equation is:

$$c[k] = K \left(e[k] + \frac{T_c}{T_i} \sum_{n=0}^k e[n] + T_d \frac{e[k] - e[k-1]}{T_c} \right) \quad (2.6)$$

In the above equations, T_c is the sampling period

2.3 Speed Control using PLC

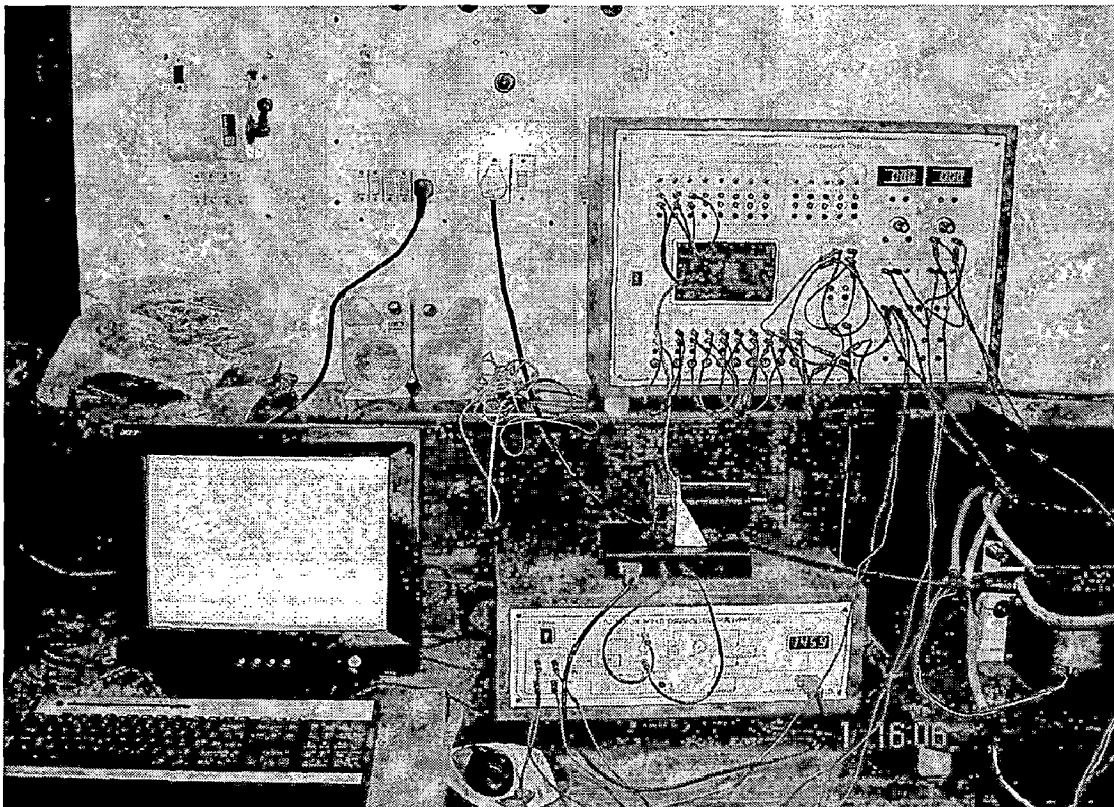


Fig. 2.2 Practical set up for speed control using PLC

The practical set up is shown in Fig. 2.2. The PLC acts as an error detector and controller (PID) as shown in Fig. 2.3. The set point is given to PLC and the PLC compares the set point (SP) and process variable (PV) from the motor and creates an error value and produces the control variable (CV) to the motor unit. Thus the speed of motor is controlled using PLC [9, 12, 13].

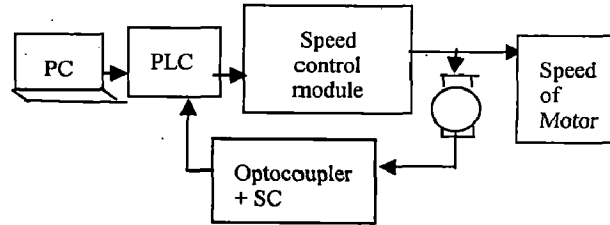


Fig. 2.3 Speed control using PID controller on PLC

2.3.1 Speed Control Module

This module, as shown in Fig. 2.4, maintains the speed of DC motor to the set point using PLC. The speed of the motor is measured using optocoupler sensor. The output of optocoupler will be a series of pulses depending upon speed of the motor. These pulses are converted into voltage using frequency to voltage converter.

This voltage is a process variable (PV) and is applied to the analog input of PLC. The setpoint is given through the software or through other analog input which is fed by an analog potentiometer. The PID control action was taken by the PLC and gives the controlled output through analog output channel.

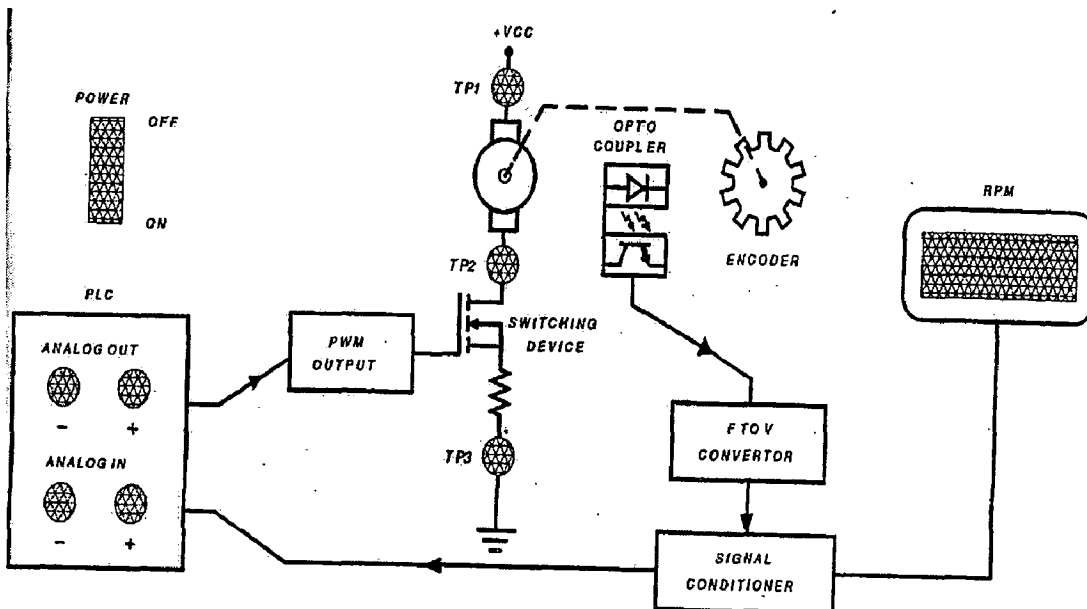


Fig. 2.4 Speed control module

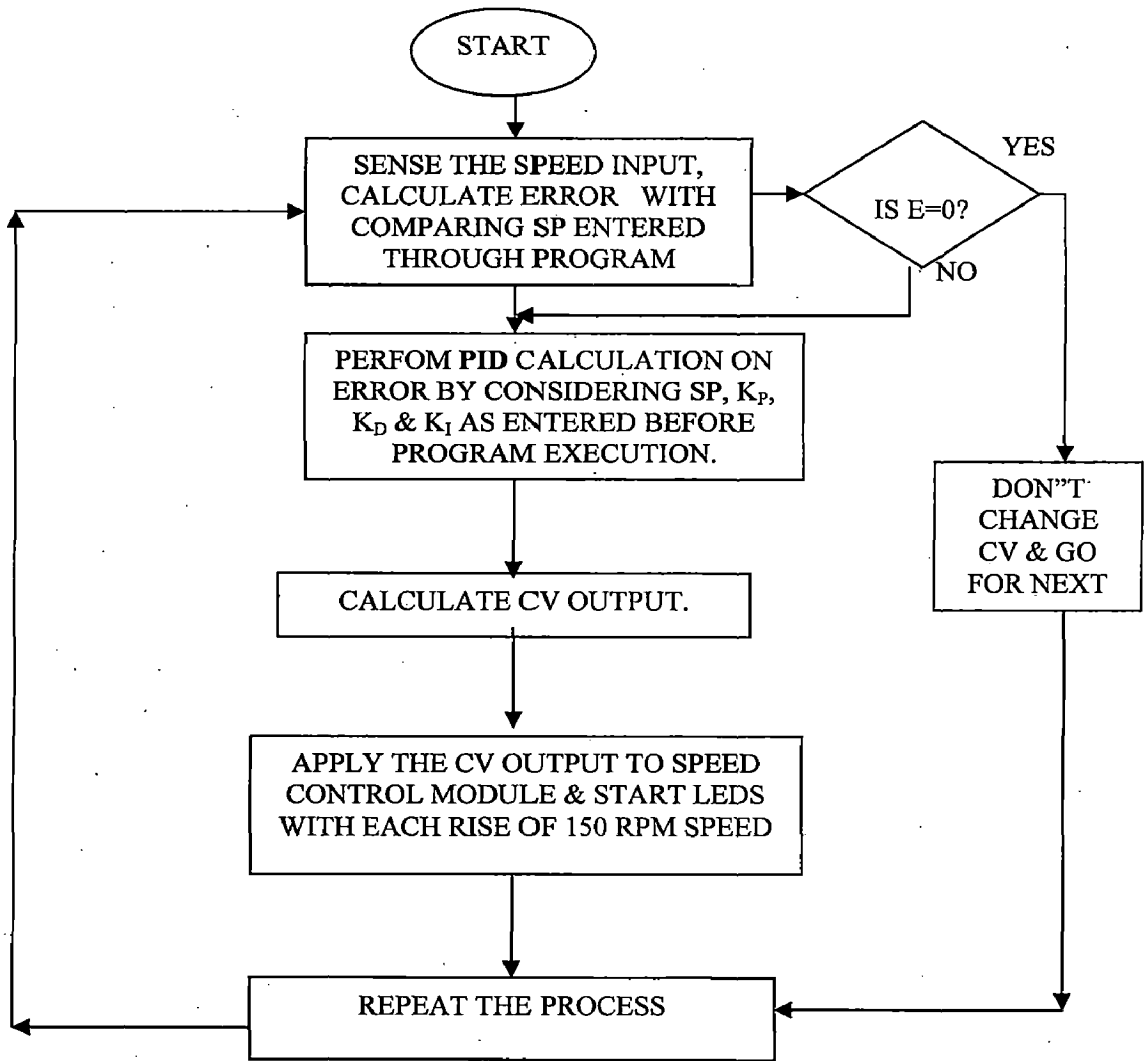


Fig. 2.5 PID control action performed by PLC for speed

2.4.2 Speed Control at 750 rpm using PI Algorithm on PLC

This program maintains speed of motor which set speed is adjusted by reference potentiometer (voltage/current source). The analog voltage/current from potentiometer is applied to remaining analog channel of PLC (AI0019). The PID-ISA algorithm is used for speed control which takes setpoint from voltage/current source. The PI algorithm realization on PLC for speed control of DC motor is explained in Fig. 2.6 in the form of flow chart.

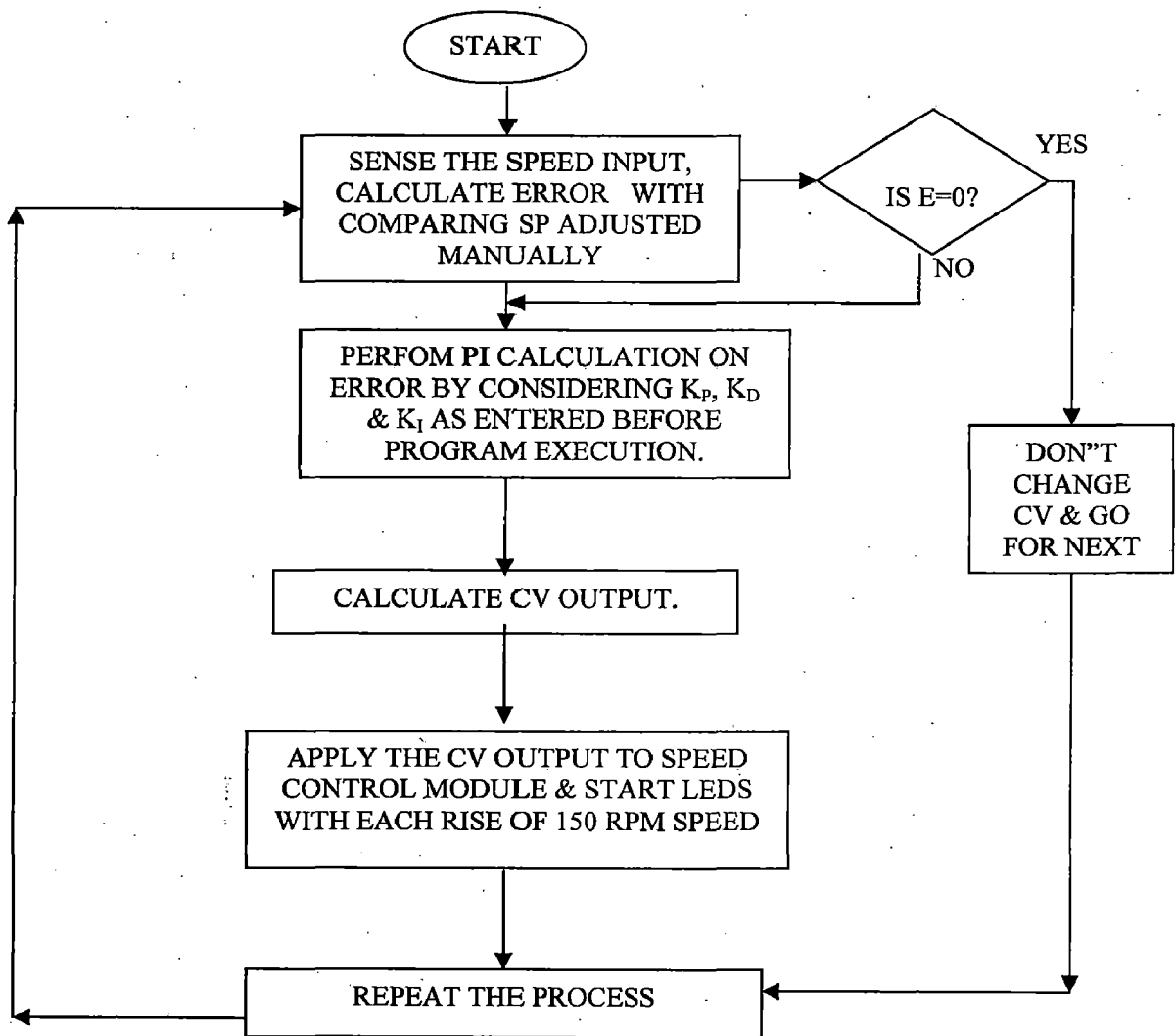


Fig. 2.6 PI algorithm on PLC for speed control

2.4.3 PID Algorithm and its Parameters on PLC

The algorithm (PID-ISA) is the closed loop control algorithm. The PID function has six input parameter: a process setpoint (SP), a manual/auto Boolean switch (MAN), a manual mode up adjustment input (UP), and manual mode down adjustment (DN). It also has an address, which specifies the location of a block of parameters associated with the function. It has two output parameters, a successful Boolean output and the control variable result (CV).

If the manual input is true, the PID block is placed in manual mode and the output control variable is set from the manual command parameter %Ref + 13. If either the UP or DN inputs are true, the manual command word is increased or decreased by one CV count every PID solution. For faster manual changes of the output (CV), it is also possible to add or subtract any CV count value directly to/from the manual command word.

For tuning parameters K_p , K_i , K_d , SP, PV, sampling period and slew time is adjusted. The CV is calculated as per equation (2.2) automatically by PLC with execution of ladder diagram. The PID algorithm parameters are explained as follows [9].

1. Address: The variable's address is the location of the PID control block information, which consists of 40 consecutive registers of memory. Data type: WORD
2. SP: SP is the control loop set point. It can be directly entered in program or can be adjusted by reference potentiometer available in PLC. Data type: WORD
3. PV: PV is the control loop process variable. Data type: WORD
4. MAN: When energized, the PID function is in manual mode.
5. UP: When energized, if in manual mode, the CV output is adjusted up.
6. DN: When energized, if in manual mode, the CV output is adjusted down.
7. Analog input to the PLC- PV from motor.
8. Analog output from PLC- CV to the speed control module.

The setpoint is given to PLC. The process variable (PV) from the motor is given as input to the analog input of PLC. The PLC performs the control operation and gives the control variable (CV) to control DC motor.

2.5 Experimental Results

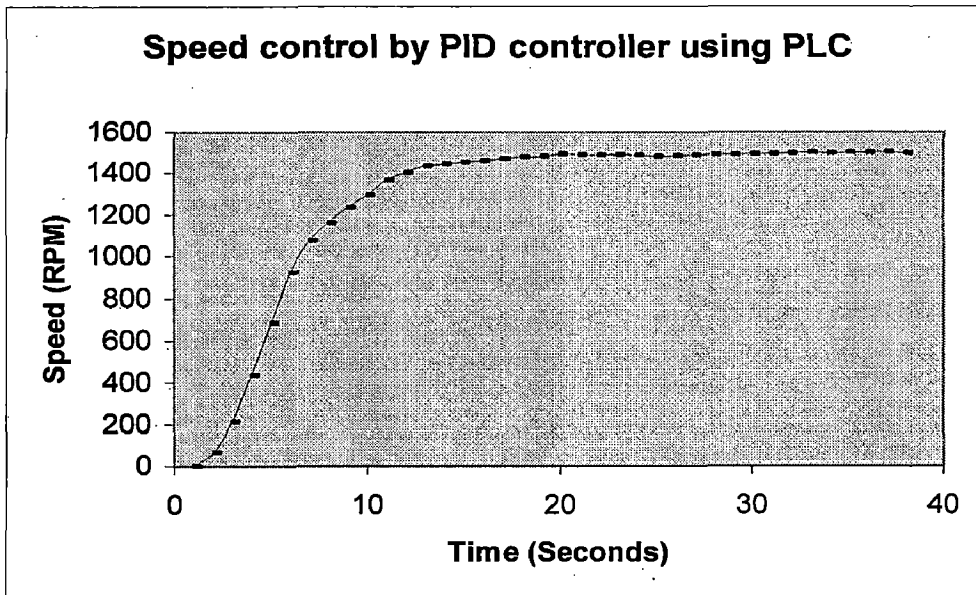


Fig. 2.7 Speed control by PID algorithm on PLC for 1500 rpm

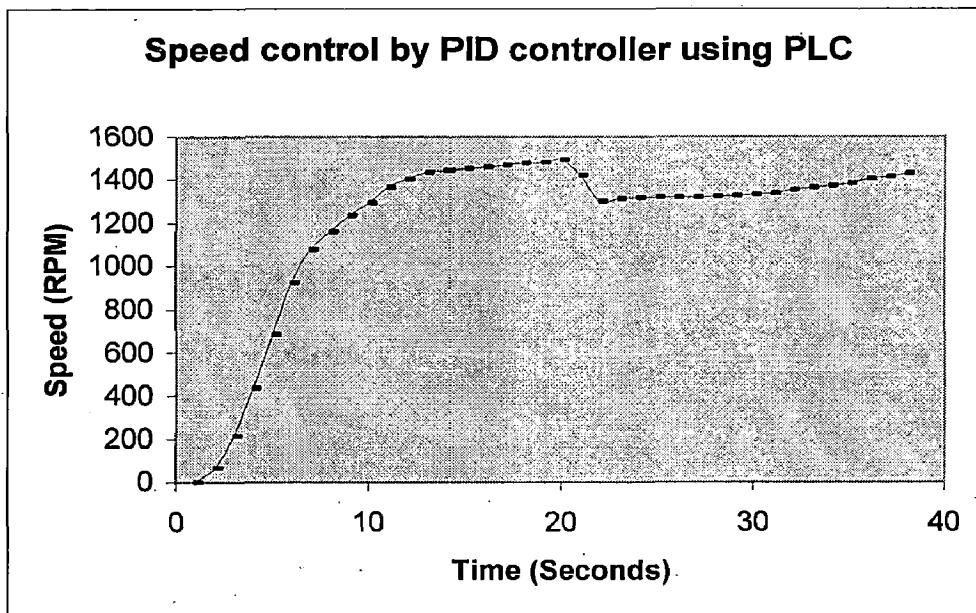


Fig. 2.8 PLC based PID control for speed with load disturbance

As shown in Fig. 2.7, the speed of 1500 rpm is controlled by PID algorithm on PLC. The settling time is 15 seconds and the rise time is 12 seconds. When the load disturbance is applied externally as shown in Fig. 2.8, The speed is reached at setpoint (1500 rpm) because of natural characteristic for removing load disturbances by PID algorithm and PLC keeps speed at setpoint though the load variations.

The same PID algorithm on PLC is converted PI algorithm for 750 rpm speed control by making derivative gain $K_d = 0$. As shown in following Fig. 2.9, The PI control increases overshoot hence oscillation which can be easily seen from Fig. 2.9. The settling time is 5 seconds and rise time is 3 seconds. Because of oscillating nature of PI control, PID control is widely used for processes like speed control and temperature control.

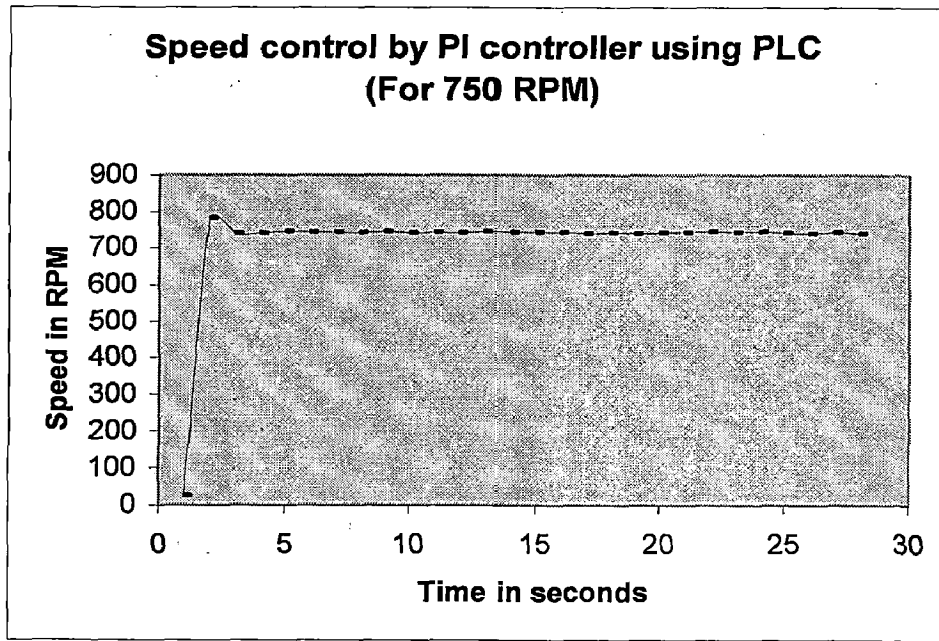


Fig. 2.9 Speed control by PI algorithm on PLC for 750 rpm

OVERVIEW ON FUZZY LOGIC CONTROL

Fuzzy controllers are used to control consumer products, such as washing machines, video cameras, and rice cookers, as well as industrial processes, such as cement kilns, underground trains, and robots. Fuzzy control is a control method based on fuzzy logic [24, 25].

Just as fuzzy logic can be described simply as “computing with words rather than numbers”; fuzzy control can be described simply as “control with sentences rather than equations”. A fuzzy controller can include empirical rules, and that is especially useful in operator controlled plants [26, 27].

Take for instance a typical fuzzy controller

1. If error is Neg and change in error is Neg then output is NB
2. If error is Neg and change in error is Zero then output is NM

The collection of rules is called a rule base. The rules are in the familiar if-then format, and formally the if-side is called the condition and the then-side is called the conclusion (more often, perhaps, the pair is called antecedent - consequence or premise-conclusion). The input value "Neg" is a short word for the linguistic term Negative. The output value "NB" stands for Negative big and "NM" for Negative Medium. The computer is able to execute the rules and compute a control signal depending on the measured inputs (error and change of error).

The objective here is to identify and explain the various design choices for engineers. In a rule based controller the control strategy is stored in a more or less natural language. The control strategy is isolated in a rule base opposed to an equation based description. A rule based controller is easy to understand and easy to maintain for a non-specialist end-user. An equivalent controller could be implemented using conventional techniques. In fact; any rule based controller could be emulated. It is just that it is convenient to isolate the control strategy in a rule base for operator controlled systems [25, 26].

3.1 Control Schemes for Fuzzy Logic Control

Fuzzy controllers are being used in various control schemes [25, 26, 27]: The most obvious one is direct control, where the fuzzy controller is in the forward path in a feedback control system (Fig. 3.1).

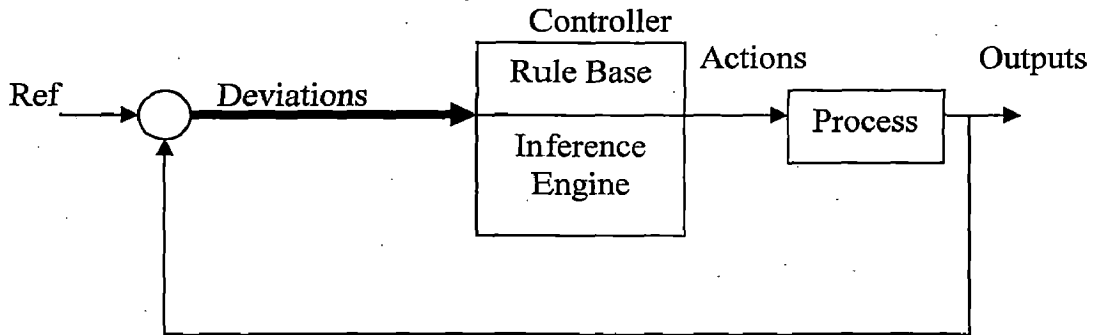


Fig. 3.1 FLC for direct control

The process output is compared with a reference, and if there is a deviation, the controller takes action according to the control strategy. In the figure, the arrows may be understood as hyper-arrows containing several signals at a time for multiloop control. The sub-components in the figure will be explained shortly. The controller is here a fuzzy controller, and it replaces a conventional controller, say, (proportional-integral-derivative) controller.

In feed forward control (Fig. 3.2) a measurable disturbance is being compensated. It requires a good model, but if a mathematical model is difficult or expensive to obtain, a fuzzy model may be useful. Fig. 3.2 shows a controller and the fuzzy compensator, the process and the feedback loop are omitted for clarity. The scheme, disregarding the disturbance input, can be viewed as a collaboration of linear and nonlinear control actions; the controller C may be a linear PID controller, while the fuzzy controller F is a supplementary nonlinear controller. Fuzzy rules are also used to correct tuning parameters in parameter adaptive control schemes (Fig. 3.3). If a nonlinear plant changes operating point, it may be possible to change the parameters of the controller according to each operating point. This is called gain scheduling since it was originally used to change process gains. A gain scheduling controller contains a linear controller whose parameters are changed as a function of the operating point in a preprogrammed way. It requires

thorough knowledge of the plant, but it is often a good way to compensate for nonlinearities and parameter variations. Sensor measurements are used as scheduling variables that govern the change of the controller parameters, often by means of a table look-up.

Whether a fuzzy control design will be stable is a somewhat open question. Stability concerns the system's ability to converge or stay close to equilibrium. A stable linear system will converge to the equilibrium asymptotically no matter where the system state variables start from.

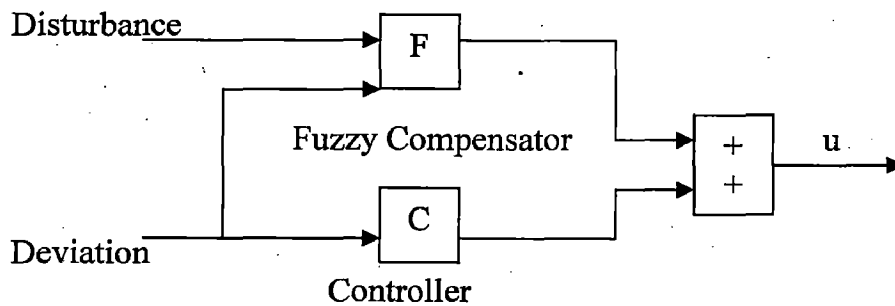


Fig. 3.2 Fuzzy feed forward control

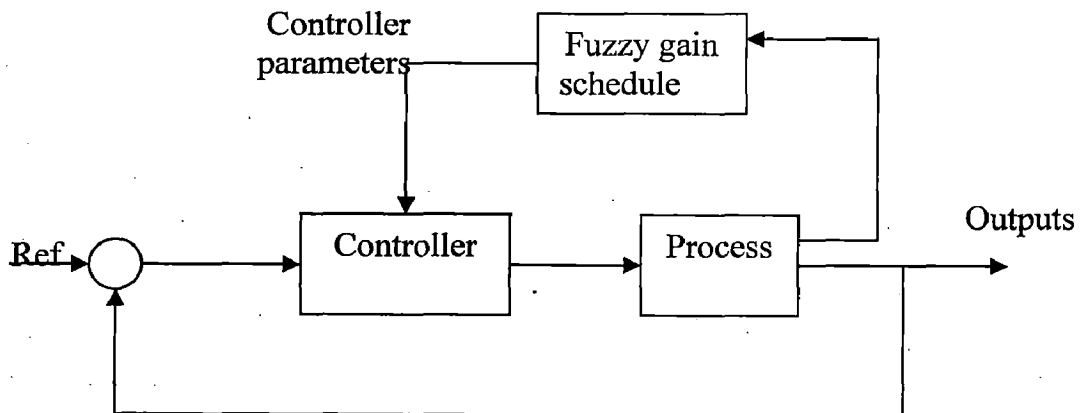


Fig. 3.3 Fuzzy parameter adaptive control

There are at least four main sources for finding control rules [3, 24, 27]

Expert experience and control engineering knowledge: One classical example is the operator's handbook for a cement kiln [3, 28]. The most common approach to establish such a collection of rules of thumb is to question experts or operators using a carefully organized questionnaire.

Based on the operator control actions: Fuzzy If-Then rules can be deduced from observations of an operator's control actions or a log book. The rules express input-output relationships.

Based on the fuzzy model process: A linguistic rule base may be viewed as an inverse model of the controlled process. Thus the fuzzy control rules might be obtained by inverting a fuzzy model of the process. This method is restricted to relatively low order systems, but it provides an explicit solution by assuming that fuzzy models of the open and closed loop systems are available [3, 24, 28].

Based on the learning: The self-organizing controller is an example of a controller that finds the rules itself. Neural networks are another possibility. There is no design procedure in fuzzy control as the procedures available for the conventional control likewise root-locus design, frequency response design, pole placement design, or stability margins, because the rules are often nonlinear [3, 24].

3.2 Structure of a Fuzzy Controller

There are specific characteristics of a fuzzy controller to support a design procedure. In the block diagram in Fig. 3.4, the controller is between a preprocessing block and a post-processing block. The following sections explain the diagram block by block.

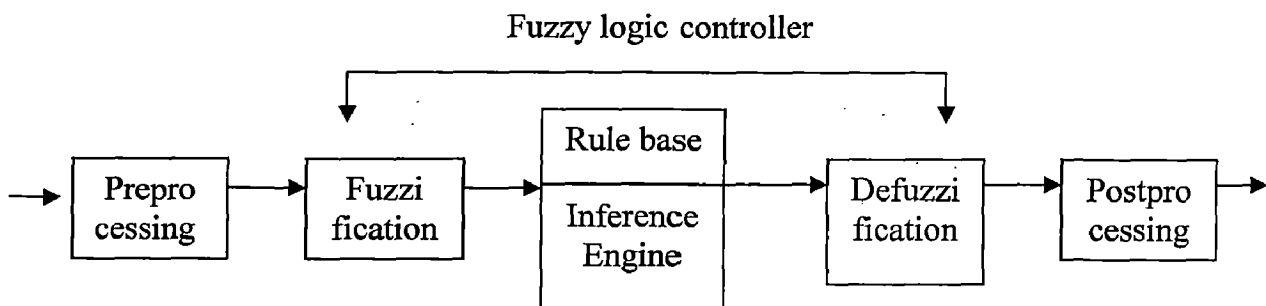


Fig. 3.4 Blocks of a fuzzy controller

3.2.1 Preprocessing

The inputs are most often hard or crisp measurements from some measuring equipment, rather than linguistic. A preprocessor, the first block in Fig. 3.4, conditions the measurements before they enter the controller. Examples of preprocessing are:

Quantization in connection with sampling or rounding to integers; normalization or scaling onto a particular, standard range; filtering in order to remove noise; averaging

to obtain long term or short term tendencies; a combination of several measurements to obtain key indicators; and differentiation and integration or their discrete equivalences.

When the input to the controller is error, the control strategy is a static mapping between input and control signal. A dynamic controller would have additional inputs, for example derivatives, integrals, or previous values of measurements, taken backwards in time. These are created in the preprocessor thus making the controller multi-dimensional, which requires many rules and makes it more difficult to design. The preprocessor then passes the data on to the controller.

3.2.2 Fuzzification

The first block inside the controller is fuzzification, which converts each piece of input data to degrees of membership by a lookup in one or several membership functions. The fuzzification block thus matches the input data with the conditions of the rules to determine how well the condition of each rule matches that particular input instance. There is a degree of membership for each linguistic term that applies to that input variable.

3.2.3 Rule Base

The rules may use several variables both in the condition and the conclusion of the rules. The controllers can therefore be applied to both multi-input-multi-output (MIMO) problems and single-input-single-output (SISO) problems. The typical SISO problem is to regulate a control signal based on an error signal. The controller may actually need the error, the change of error, and the accumulated error as inputs, but we will call it single-loop control, because in principle all three are formed from the error measurement. To simplify, the present work assumes that the control objective is to regulate some process output around a prescribed setpoint or reference [25, 26, 27].

Rule formats: Basically a linguistic controller contains rules in the If-Then format, but they can be presented in different formats. In many systems, the rules are presented to the end-user in a format similar to the below one,

1. If error is Neg and change in error is Neg then output is NB
2. If error is Neg and change in error is Zero then output is NM
3. If error is Neg and change in error is Pos then output is Zero

4. If error is Zero and change in error is Neg then output is NM
5. If error is Zero and change in error is Zero then output is Zero
6. If error is Zero and change in error is Pos then output is PM
7. If error is Pos and change in error is Neg then output is Zero
8. If error is Pos and change in error is Zero then output is PM
9. If error is Pos and change in error is Pos then output is PB

The names Neg, Zer, Pos are labels of fuzzy sets as well as NB, NM, PB and PM (negative big, negative medium, positive big and positive medium respectively). The relational format is certainly suited for storing in a relational database. It should be emphasized, though, that the relational format implicitly assumes that the connective between the inputs is always logical **and** or logical **or** for that matter. Incidentally, a fuzzy rule with an **or** combination of terms can be converted into an equivalent **and** combination of terms using laws of logic (DeMorgan's laws among others). A second format is the tabular linguistic format as shown in Table 3.1.

Table 3.1 A rule base in tabular format

		CHANGE IN ERROR		
		Neg	Zero	Pos
ERROR	Neg	NB	NM	Zero
	Zero	NM	Zero	PM
	Pos	Zero	PM	PB

This is even more compact. The input variables are laid out along the axes, and the output variable is inside the table. In case the table has an empty cell, it is an indication of a missing rule, and this format is useful for checking completeness. When the input variables are error and change in error, as they are here, that format is also called a linguistic phase plane. In case there are $n > 2$, input variables involved, the table grows to an n -dimensional array.

Lastly, a graphical format which shows the fuzzy membership curves is also possible (Fig. 3.5). This graphical user-interface can display the inference process better than the other formats, but takes more space on a monitor.

3.2.4 Membership Functions

Every element in the universe of discourse is a member of a fuzzy set to some grade, may be even zero. The grade of membership for all its members describes a fuzzy set, such as Neg. In fuzzy sets elements are assigned with a grade of membership, such that the transition from membership to non-membership is gradual rather than abrupt. The set of elements that have a non-zero membership is called the support of the fuzzy set [24, 26, 27]. The function that ties a number to each element x of the universe is called the membership function $\mu(x)$.

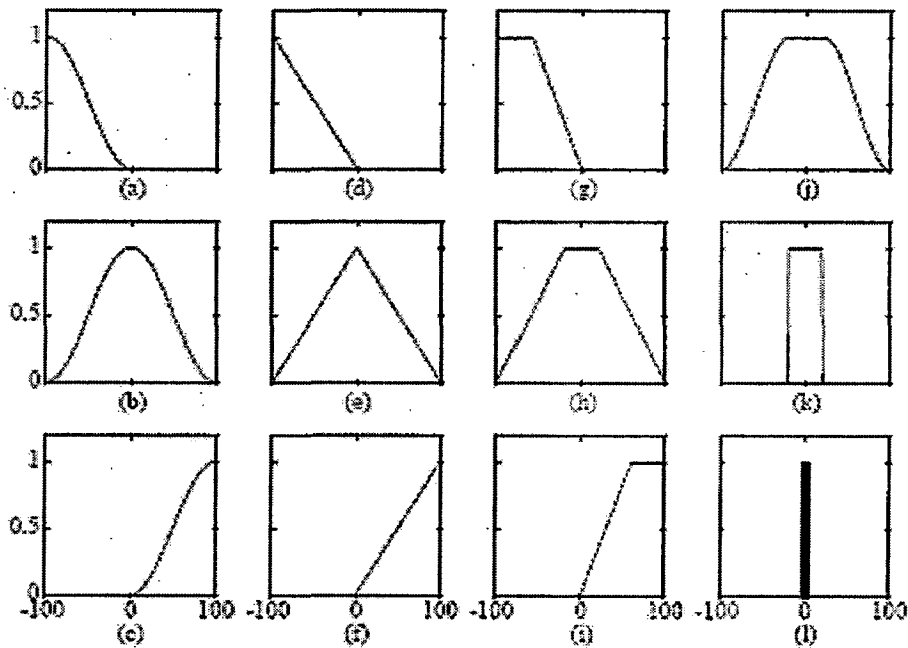


Fig. 3.5 Examples of membership functions. Read from top to bottom, left to right: (a) s-function, (b) π -function, (c) z-function, (d-f) triangular versions, (g-i) trapezoidal versions, (j) flat π -function, (k) rectangle, (l) singleton.

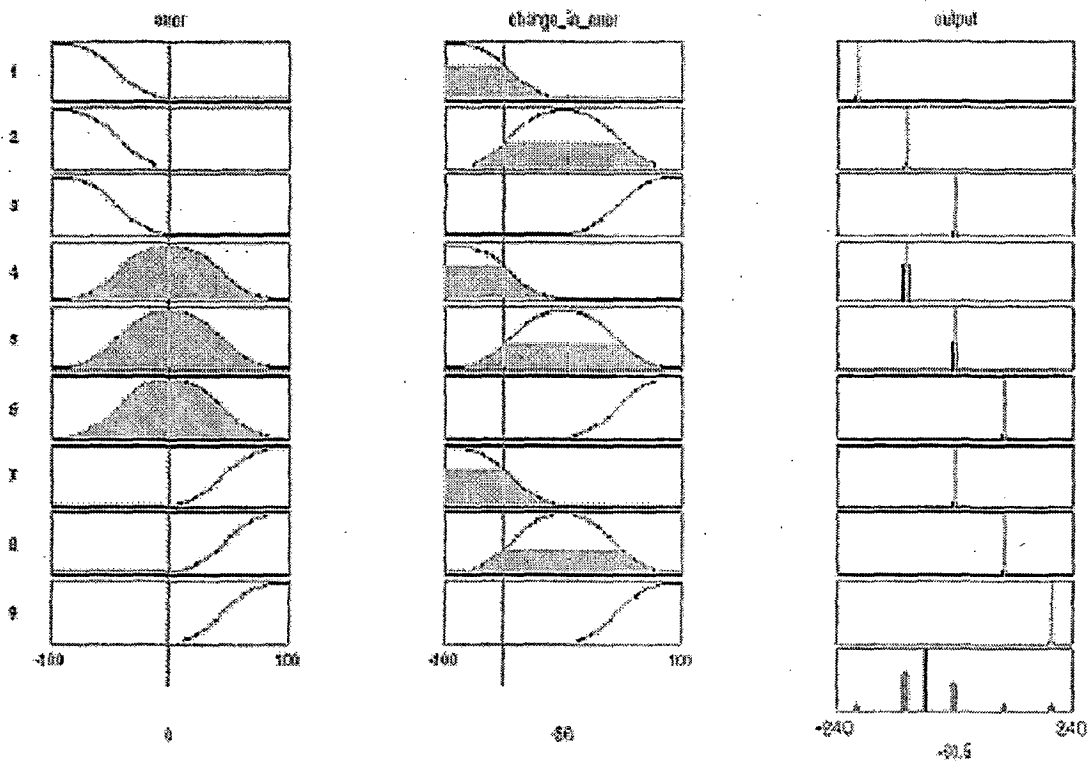
Membership functions can be flat on the top, piece-wise linear, triangle shaped, rectangular, or ramps with horizontal shoulders. Fig. 3.5 shows some typical shapes of membership functions. Strictly speaking, a fuzzy set A is a collection of ordered pairs

$$A = (x, \mu(x)) \quad (3.1)$$

Item x belongs to the universe and $\mu(x)$ is its grade of membership in A .

3.2.5 Inference Engine

Fig. 3.6 and 3.7 are both a graphical construction of the algorithm in the core of the controller.



**Fig. 3.6 Graphical construction of the control signal in a fuzzy PD controller
(generated in fuzzy MATLAB toolbox)**

In Fig. 3.6, each of the nine rows refers to one rule. For example, the first row says that if the error is negative (row 1, column 1) and the change in error is negative (row 1, column 2) then the output should be negative big (row 1, column 3). The picture corresponds to the rule base in Table (3.1). The rules reflect the strategy, that the control signal should be a combination of the reference error and the change in error, of a fuzzy proportional-derivative controller. We shall refer to that figure in the following discussion. The instances of the error and the change in error are indicated by the vertical lines on the first and second columns of the chart. For each rule, the inference engine looks up the membership values for the condition of the rule.

Aggregation: The Aggregation operation is used to calculate the aggregation or firing strength α_k of the condition of a rule k . A rule, say rule 1, will generate a fuzzy membership value μ_{e1} coming from the error and a membership value μ_{ce1} coming from

the change of error measurement. The aggregation is their combination as shown in Equation (3.2), similarly the process continues for the other rules.

$$\mu_{e1} \text{ and } \mu_{ce1} \quad (3.2)$$

Aggregation is equivalent to fuzzification, when there is only one input to the controller. Aggregation is sometimes also called fulfillment of the rule or firing strength.

Activation: The activation of a rule is the deduction of the conclusion, possibly reduced by its firing strength. Thickened lines in the third column indicate the firing strength of each rule. Only the thickened part of the singletons are activated, and **min** or **product** (*) is used as the activation operator. It makes no difference in this case, since the output membership functions are singletons, but in the general case of s , π and z functions in the third column, the multiplication scales the membership curves, thus preserving the initial shape, rather than clipping them as the min operation does. Both methods work well in general, although the multiplication results in a slightly smoother control signal. In Fig. 3.6, only rules four and five are active. A rule k can be weighted a priori by a weighting factor $w_k \in [0, 1]$, which is its degree of confidence. In that case the firing strength is modified to

$$\alpha_k^* = w_k * \alpha_k \quad (3.3)$$

The degree of confidence is determined by the designer or a learning program trying to adapt the rules to some input-output relationship.

Accumulation: All activated conclusions are accumulated, using the **max** operation, to the final graph on the bottom right (Fig. 3.6). Alternatively, **sum** accumulation counts overlapping areas more than once. Singleton output (Fig. 3.6) and **sum** accumulation results in the simple output

$$\alpha_1 * s_1 + \alpha_2 * s_2 + \dots + \alpha_n * s_n \quad (3.4)$$

The alpha's are the firing strengths from the n rules and s_1, \dots, s_n are the output singletons. Since this can be computed as a vector product, this type of inference is relatively fast in a matrix oriented language.

3.2.6 Defuzzification

The resulting fuzzy set (Fig. 3.6, bottom right; Fig. 3.7, extreme right) must be converted to a number that can be sent to the process as a control signal.

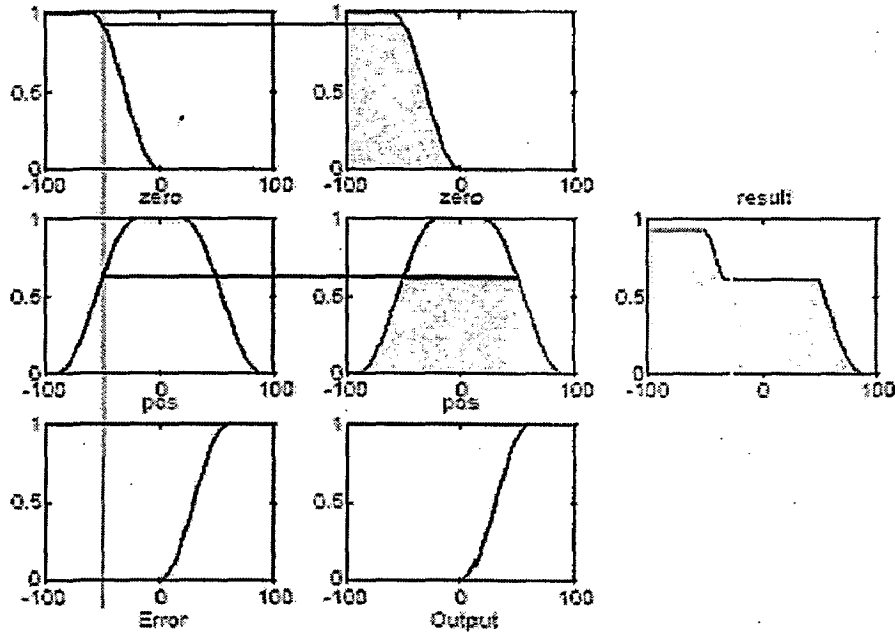


Fig. 3.7 One input, one output rule base with non-singleton output sets.

This operation is called defuzzification shown in Fig. 3.7. The x-coordinate marked by a white, vertical dividing, line becomes the control signal. The resulting fuzzy set is thus defuzzified into a crisp control signal. There are several defuzzification methods, which is described as follows.

Center of gravity (COG): The crisp output value x (white line in Fig. 3.7) is the abscissa under the centre of gravity of fuzzy set,

$$u = \frac{\sum_{i=1}^n \mu(x_i)x_i}{\sum_{i=1}^n \mu(x_i)} \quad (3.5)$$

Here x_i is a running point in a discrete universe, and $\mu(x_i)$, is its membership value in the membership function. The expression can be interpreted as the weighted average of the elements in the support set. For the continuous case, replace the summations by integrals. It is a widely used method although its computational complexity is relatively high. This method is also called center of area.

Center of gravity method for singletons (COGS): If the membership functions of the conclusions are singletons (Figure 3.6), the output value is

$$u = \frac{\sum_{i=1}^n \mu(s_i)s_i}{\sum_{i=1}^n \mu(s_i)} \quad (3.6)$$

Here s_i is the position of singleton i in the universe, and $\mu(s_i)$, is equal to the firing strength α_i of rule i . This method has a relatively good computational complexity, and u is differentiable with respect to the singletons s_i , which is useful in neuro-fuzzy systems.

Mean of Maxima: An intuitive approach is to choose the point with the strongest possibility, i.e. maximal membership. It may happen, though, that several such points exist, and a common practice is to take the Mean of maxima (MOM). This method disregards the shape of the fuzzy set, but the computational complexity is relatively good.

Left most maximum (LM) and Right most Maximum (RM): Another possibility is to choose the leftmost maximum (LM), or the rightmost maximum (RM). In the case of a robot, for instance, it must choose between left or right to avoid an obstacle in front of it. The defuzzifier must then choose one or the other, not something in between. These methods are indifferent to the shape of the fuzzy set, but the computational complexity is relatively small.

3.2.7 Postprocessing

Output scaling is also relevant. In case the output is defined on a standard universe this must be scaled to engineering units for instance, volts, meters, or tons per hour. An example is the scaling from the standard universe $[-1, 1]$ to the physical units $[-10, 10]$ volts. The postprocessing block often contains an output gain that can be tuned, and sometimes also an integrator.

3.2.8 Summary for Fuzzy Logic Processing

In a fuzzy controller the data passes through a preprocessing block, a controller, and a postprocessing block. Preprocessing consists of a linear or non-linear scaling as well as a quantization in case the membership functions are discretised (vectors); if not, the membership of the input can just be looked up in an appropriate function. When designing the rule base, the designer needs to consider the number of term sets, their shape, and their overlap. The rules themselves must be determined by the designer, unless more advanced means like self-organization or neural networks are available [24, 25, 26]. There is a choice between multiplication and minimum in the activation. There is also a choice regarding defuzzification.

The postprocessing consists of a scaling for the output. In case the controller is incremental, postprocessing also includes integration. The following is a checklist of design choices that have to be made:

Rule base related choices: Number of inputs and outputs, rules, universes (continuous/discrete), the number of membership functions, their overlap and width, and singleton output.

Inference engine related choices: Connectives, modifiers, activation operation, aggregation operation, and accumulation operation.

Defuzzification method: COG, COGS, MOM, LM, and RM.

Pre and post-processing: Scaling, gain factors, quantization, and sampling time.

Some of these items must always be considered, others may not play a role in the particular design. The input-output mappings provide an intuitive insight which may not be relevant from a theoretical viewpoint, but in practice they are well worth using. The analysis represented by plots is limited to three dimensions. Various input-output mappings can be obtained by changing the fuzzy membership functions. The linear fuzzy controller may be used in a design procedure based on PID control.

3.3 Fuzzy Logic Toolbox

The fuzzy toolbox is a collection of functions built on the MATLAB numeric computing environment. It provides tools to create and edit fuzzy inference systems within the framework of MATLAB, or we can integrate our fuzzy systems into the simulations with Simulink, or we can even build stand alone C programs that call on fuzzy systems which we can build with MATLAB. This toolbox relies heavily on graphical user interface (GUI) tools to accomplish work, although we can work entirely from the command line if we prefer. The toolbox provides three categories of tools:

- Command line functions
- Graphical interactive tools
- Simulink blocks and examples

The first category of tools is made up of functions that we can call from the command line or from our own applications. Many of these functions are MATLAB M-files or series of MATLAB statements that implement specialized fuzzy logic algorithms. The toolbox provides a number of interactive tools that access many of the functions

through a GUI-based tools which provides environment for fuzzy inference system design, analysis and implementation. The third category of tools is a set of blocks for use with Simulink software. These are specially designed for high speed fuzzy logic inference in the Simulink environment.

The fuzzy logic toolbox is highly impressive in all respects. It makes fuzzy logic an effective tool for the conception and design of intelligent systems. The fuzzy logic toolbox is easy to master and convenient to use. And last, but not least important, it provides a reader-friendly and up-to-date introduction to the methodology of fuzzy logic and its wide-ranging applications.

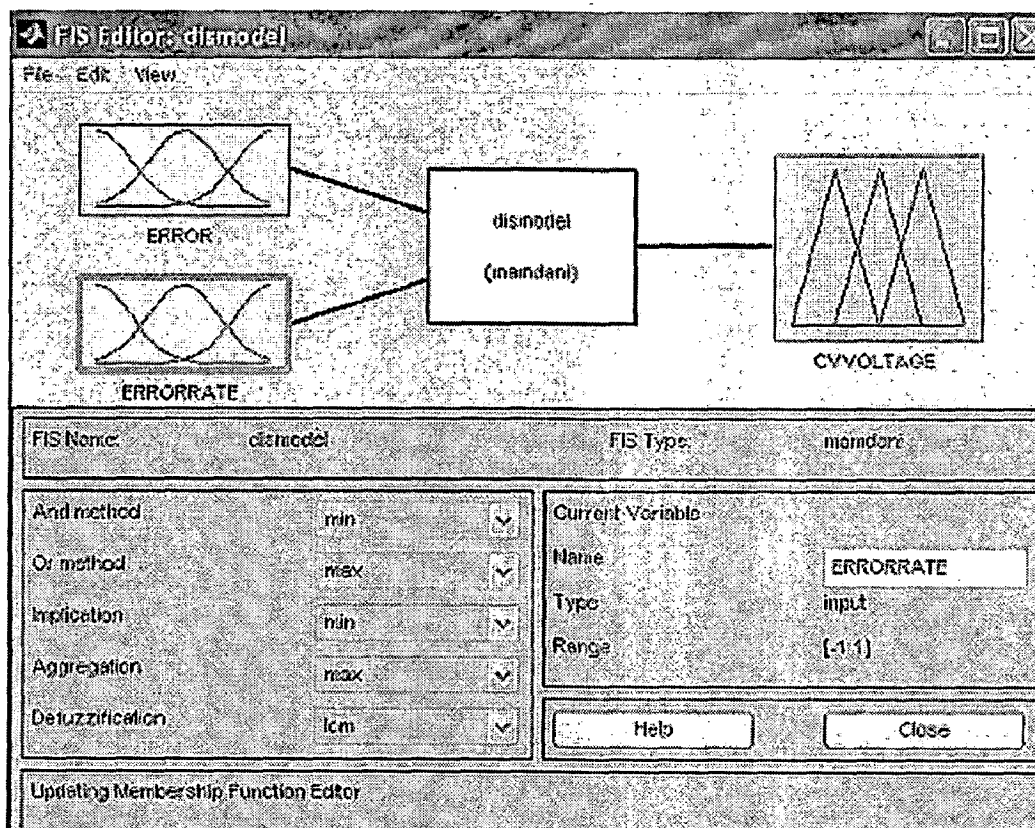


Fig. 3.8 Fuzzy logic MATLAB toolbox view

The process of fuzzy logic control can be easily performed using fuzzy MATLAB toolbox. Inputs and output membership functions, rule base and rule viewer can be easily created and edited using fuzzy MATLAB toolbox. Influence of rules on fuzzy logic process can be easily checked using the rule base editor. Fuzzy MATLAB toolbox also

provides the choices of controllers, inference processes, and defuzzification processes. Fig. 3.8 to Fig. 3.13 provides an overview on fuzzy logic processing using fuzzy MATLAB toolbox.

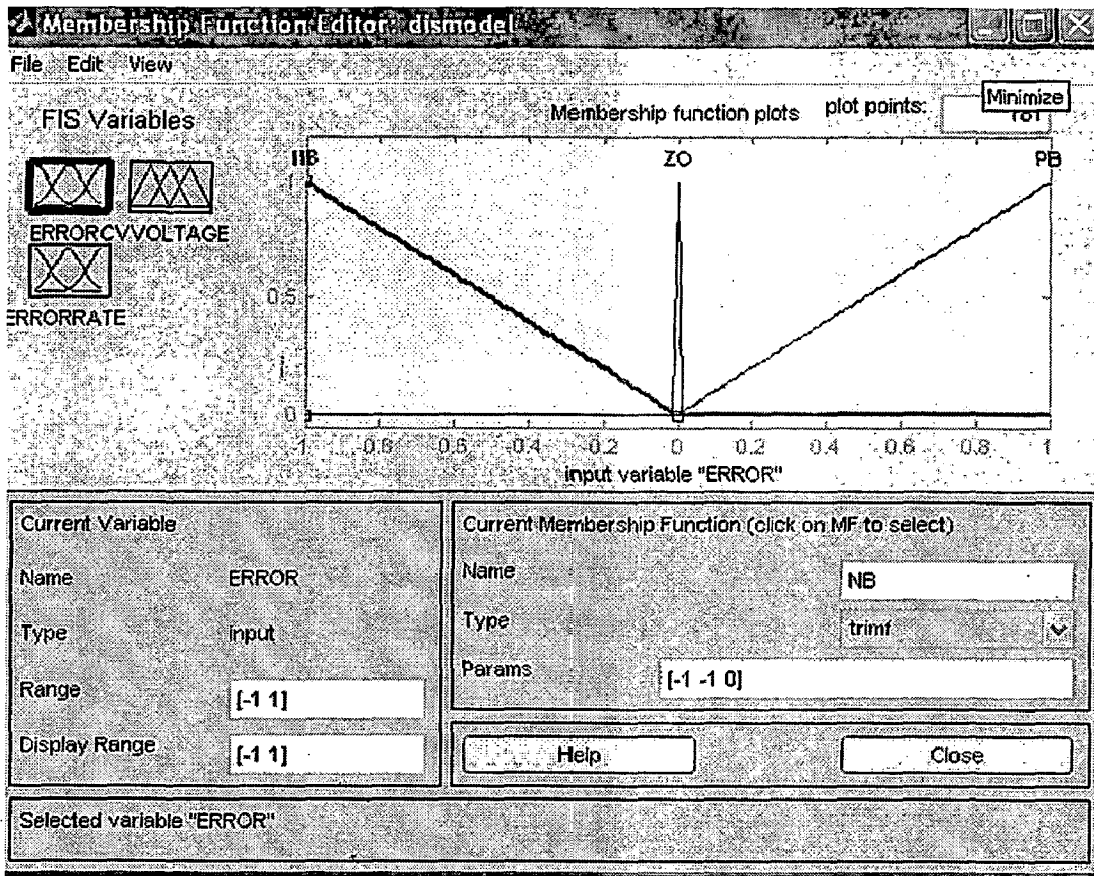


Fig. 3.9 Membership function of input variables for a FLC

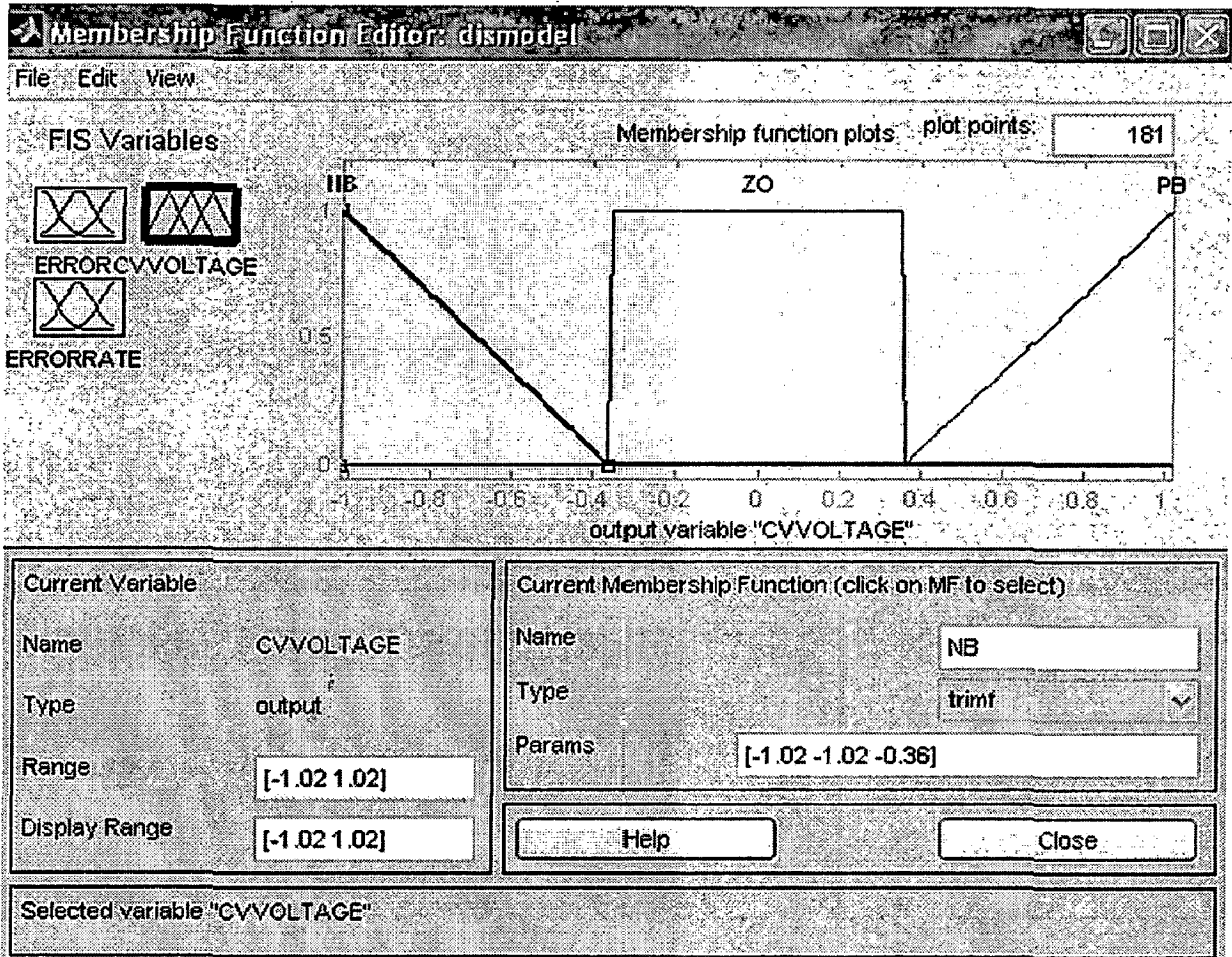


Fig. 3.10 Membership function of output variable for a FLC

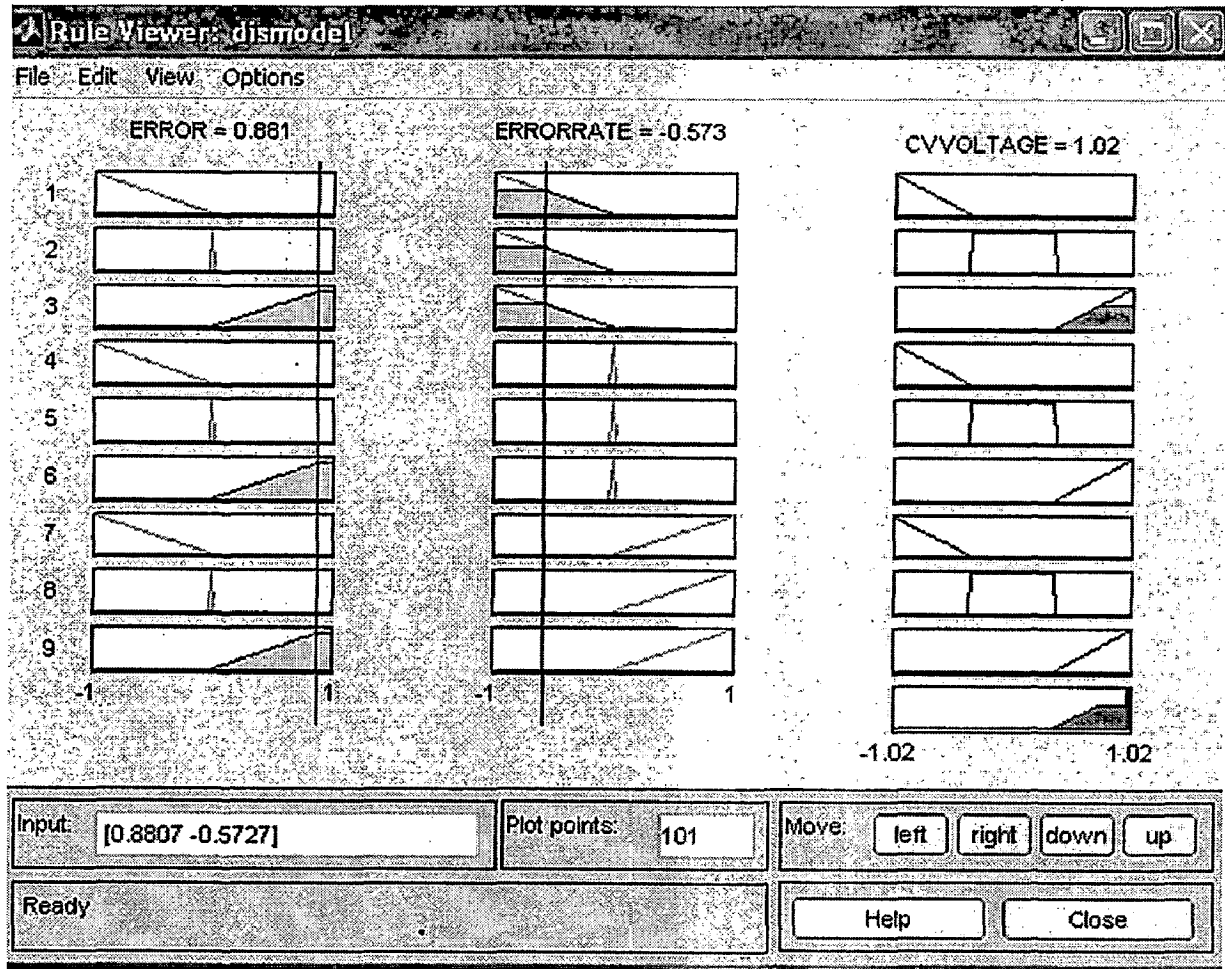


Fig. 3.11 Fuzzy rule viewer

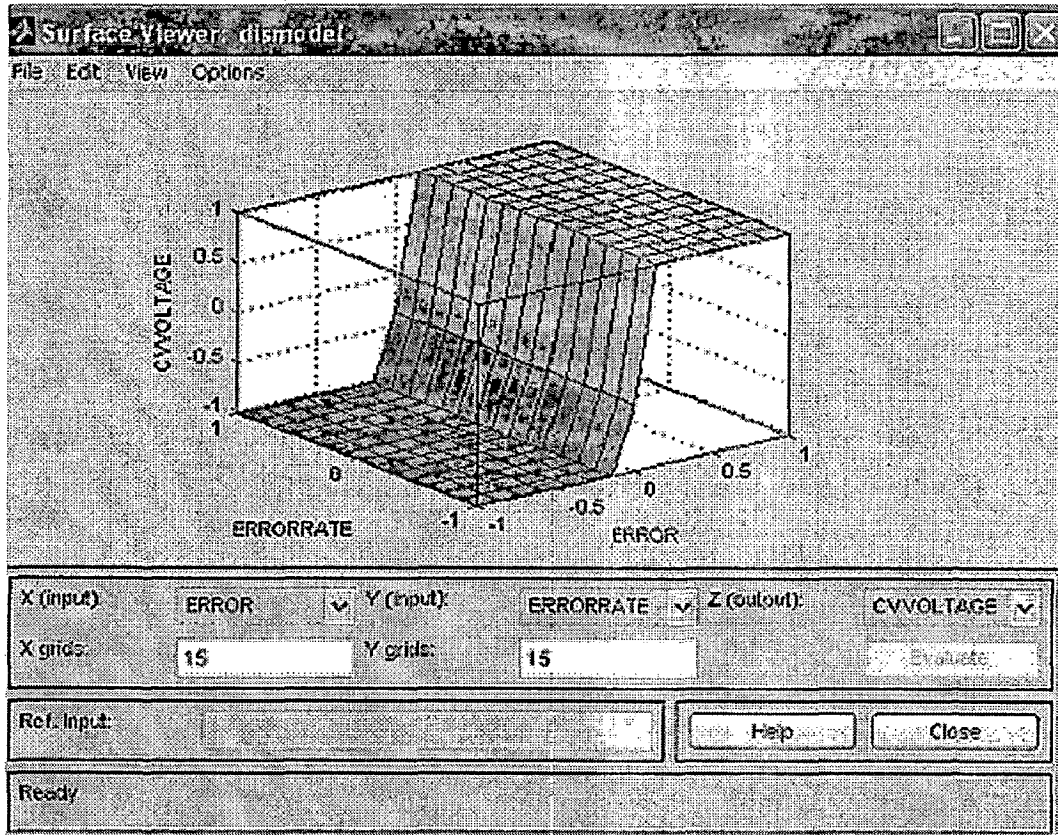


Fig. 3.12 Output surface viewer

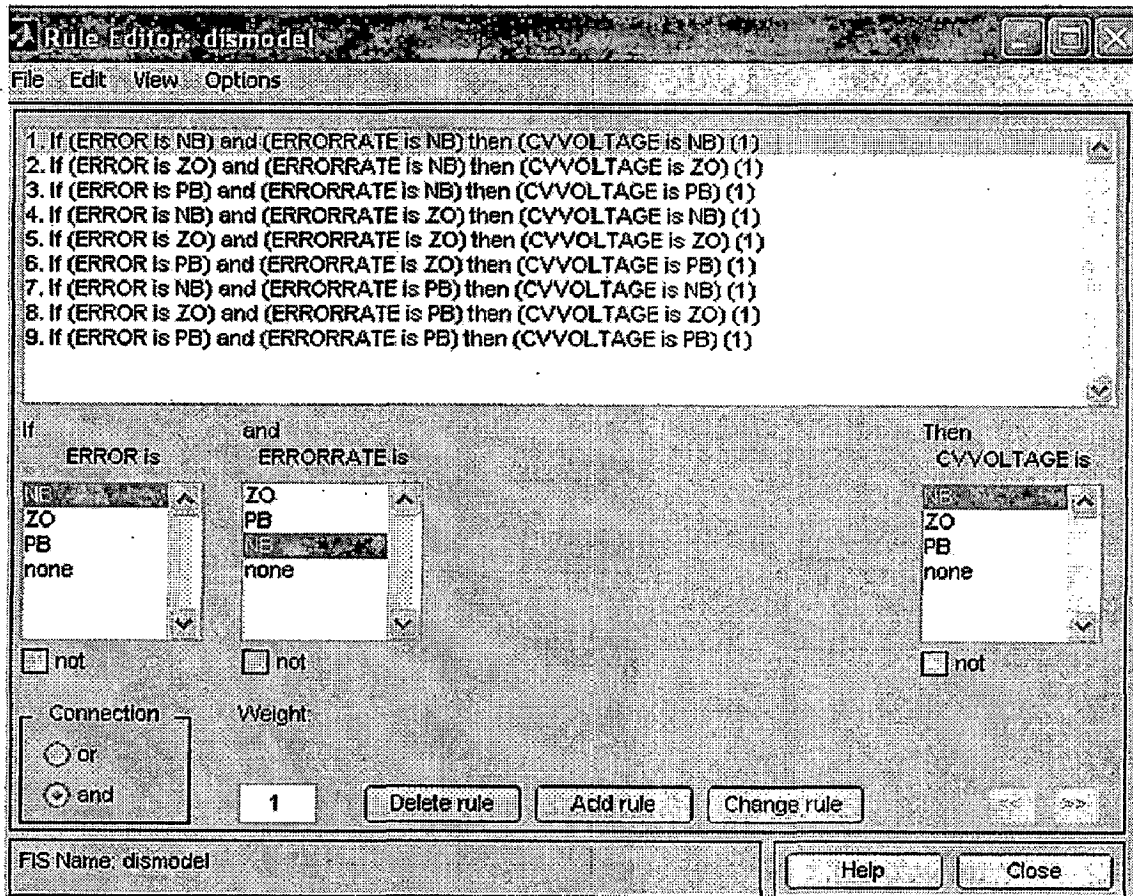


Fig. 3.13 Fuzzy rule base editor

3.4 Overview on Fuzzy Logic Controller Integrated on a PLC System

Fuzzy logic provides the ability to PLCs to make “reasoned” decisions about a process. Fuzzy logic can be used in practical applications to provide real time, logical control of a process. Fig. 3.14 illustrates a fuzzy logic control system integrated on a PLC (OMRON’s PLC). The input to the fuzzy system is the output of the process, which is entered into the system via input interfaces. For example, in a temperature control application, the input data would be entered using an analog input module. This input information would then go through the fuzzy logic process, where the processor would analyze a database to obtain an output. Fuzzy processing involves the execution of If...Then rules, which are based on the input conditions. An input’s grade specifies how well it fits into a particular graphic set (e.g., too little, normal, too much). Input data, as shown in Fig. 3.15, may also be represented as a count value ranging from 0 to 4095 or as a percentage of error deviation. If the fuzzy logic system utilizes an analog input that has

a count range from 0 to 4095, the graphs representing the input will cover the span from 0 to 4095 counts. Furthermore, the analog input information (0– 4095 counts) may represent an error range, from -50% to $+50\%$, of a process.

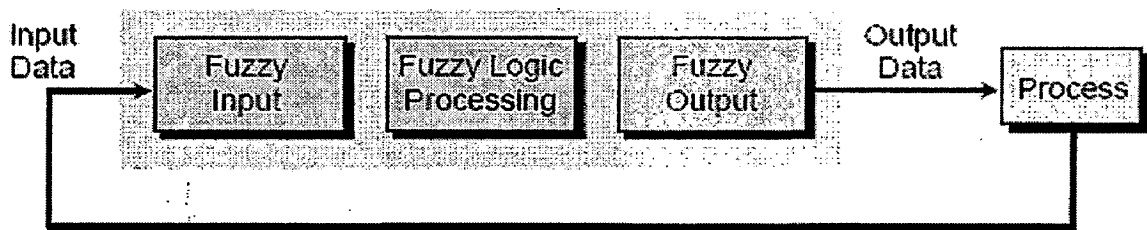


Fig. 3.14 Fuzzy logic control system.

The output of a fuzzy controller is also defined by grades, with the grade determining the appropriate output value for the control element. The output of the fuzzy logic system as shown in Fig. 3.16, for example, controls a steam valve, which opens or closes according to its grade on the output chart.

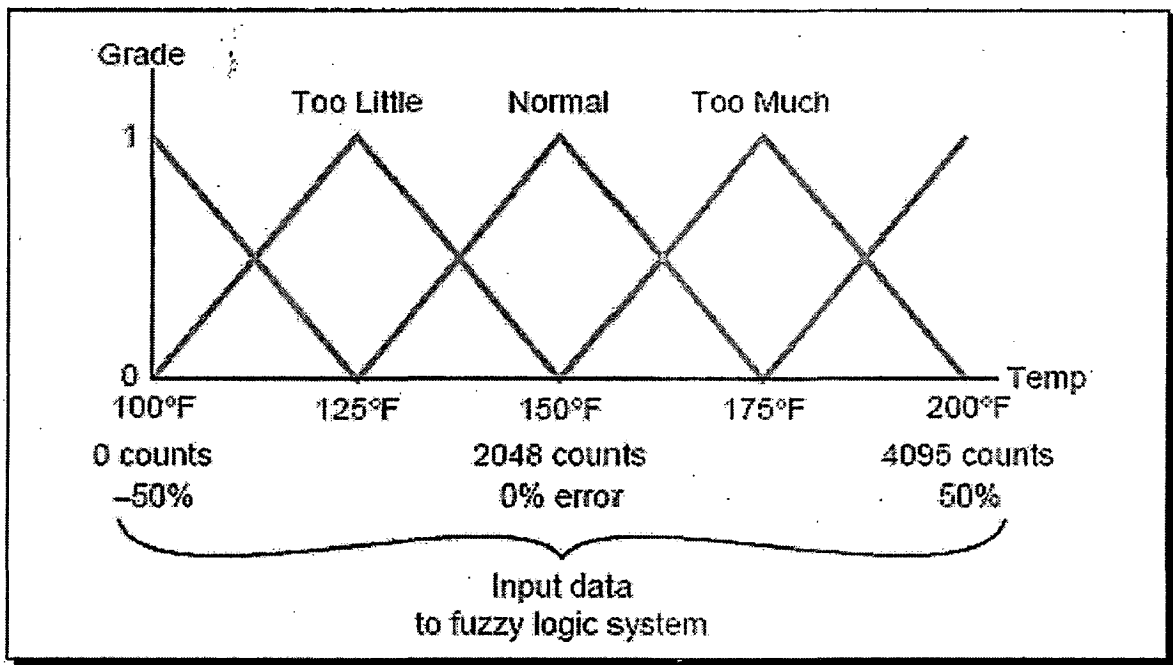


Fig. 3.15 Input data to a fuzzy logic system represented as counts and percentages.

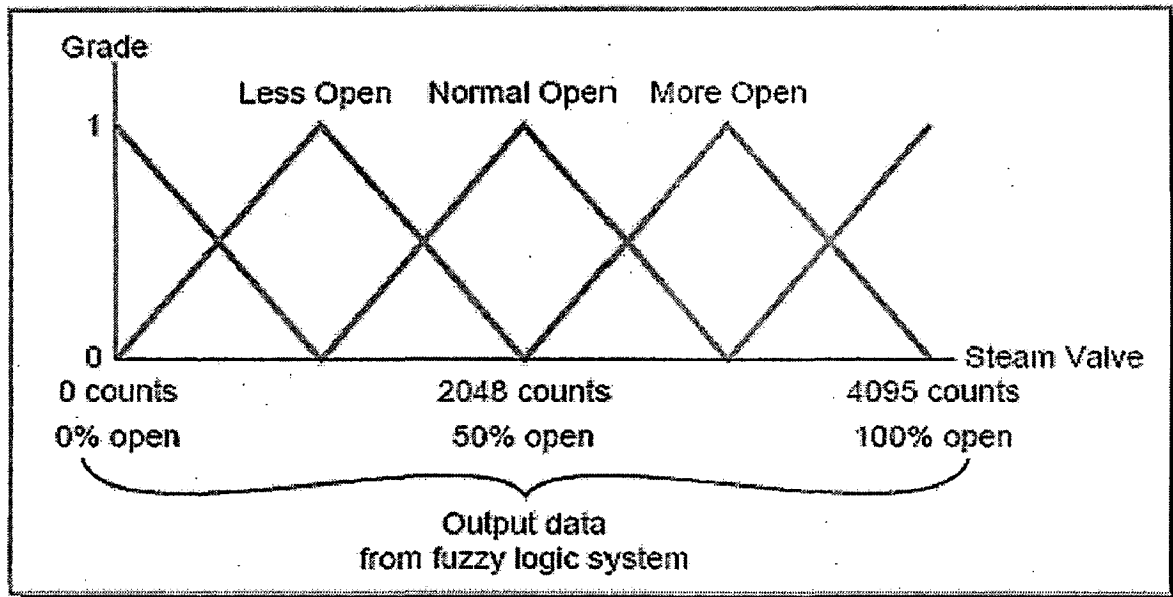


Fig.3.16 Output data from a fuzzy logic system represented as counts and percentages.

Fig. 3.17 illustrates a fuzzy logic cooling system chart with both input and output grades, where the horizontal axis is the input condition (temperature) and the vertical axis is the output (air-conditioner motor speed). In this chart, a single input can trigger more than one output condition. For example, if the input temperature is 137.5°F, then the temperature is part of two input curves—it is 50% too cool and 50% normal. Consequently, the input will trigger two outputs—the too cool input condition will trigger a less speed output, while the normal input will trigger a normal speed output condition. Since the fuzzy logic controller can have only one output, it completes defuzzification to determine the actual final output value [12, 13].

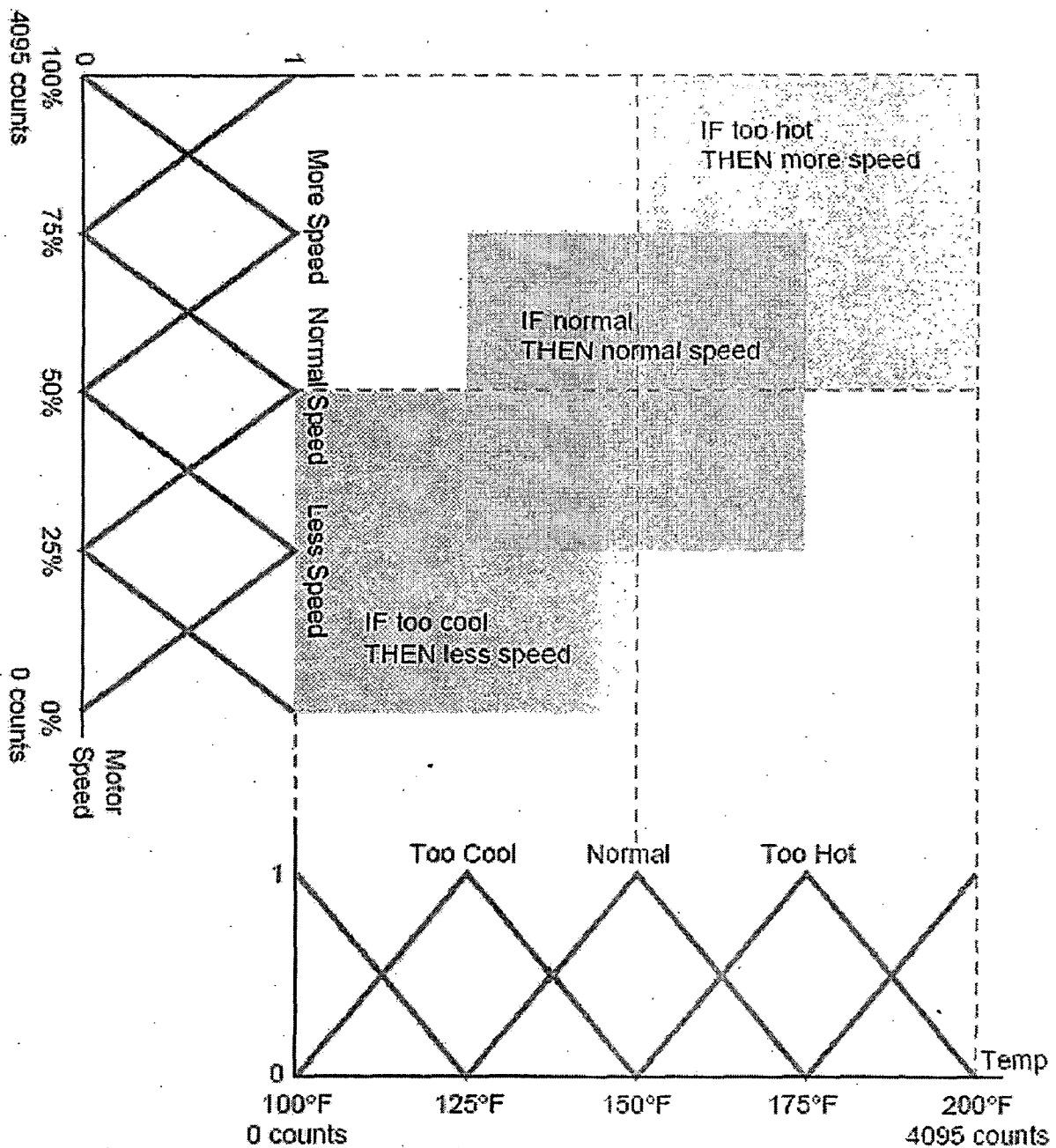


Fig. 3.17 Fuzzy logic system chart showing both input and output grades.

The implementation and operation of a fuzzy logic control system is similar to the implementation of PID control using intelligent interfaces, where the module reads the input, processes the information, and provides an output. However, fuzzy controllers are usually independent interfaces, which plug into the PLC rack [5, 16, 28, 29] and use the PLC's I/O system to communicate with the process under fuzzy control.

Different fuzzy logic controllers have different rule evaluation capabilities. The fuzzy logic controller from Omron Electronics shown in Fig. 3.18, for example, is capable of handling eight inputs and two outputs, where each input can be represented by a maximum of seven membership functions for a total of 56 membership functions (8×7). The controller also allows a maximum of 128 programmed rules. Each rule can have up to eight input conditions (which can be logically ANDed or ORed) and two outcomes [5, 16].

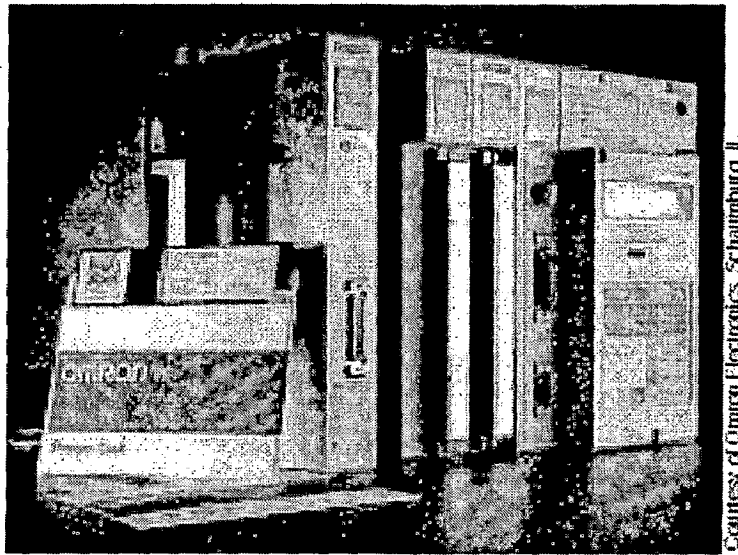


Fig. 3.18 Omron's fuzzy logic controller in a PLC system (The first fuzzy-PLC).

CHAPTER 4

PLC BASED FUZZY LOGIC CONTROLLER FOR SPEED

The implementation and operation of a fuzzy logic control system for speed control of DC motor is similar to the implementation of PID control using ladder logic program of PLC, where the module reads the input, processes the information, and provides an output [4, 5, 16, 19].

In the present context the following set of conditions were implemented in an orderly manner to achieve the accurate output from the fuzzy inference system.

Step 1: Define inputs and outputs for the fuzzy logic controller

Step 2: Define frames for fuzzy variables

Step 3: Assign membership values to fuzzy variable

Step 4: Create a rule base

Step 5: Fuzzify inputs to the FLC

Step 6: Determine which rule fires

Step 7: Infer the output recommended by each rule

Step 8: Aggregate the fuzzy outputs recommended by each rule

Step 9: Defuzzify the aggregated fuzzy set to form crisp output from the FLC .

The input variables error, change in error and sum of errors are given as follows:

Error=Set speed -Measured speed,

Change in error =Current error- Previous error

& Sum of Error = Current error+ Previous errors.

The output is the control variable voltage from PLC which is a crisp value. Through the use of membership functions defined for each fuzzy set for each input variable, the degree of membership for a crisp value in each fuzzy set is determined. In a fuzzy expert system application, each input variable's crisp value is first fuzzified into linguistic values before the inference engine proceeds in processing with the rule base.

These are the points which are considered during design of fuzzy logic controller on PLC through ladder logic [12, 13, 28, 29].

- Use triangular input sets and output set that do not overlap.
- Scale inputs in the measured range of PLC.

- Use Max-min inference method.
- Use LOM & FOM defuzzification.
- Scale output in the controlled variable range of PLC, if necessary.

There are 2 analog inputs available in GE PLC [9, 11]. So one input is used as process variable hence to sense measured speed, while the other can be used to apply reference speed. The speed is controlled in the range of 0 to 1500 rpm, which is measured in the range of 0 to 5 Volts by speed sensor and signal conditioner circuit of speed control module. The PLC converts this 0 to 5 Volts signal (transduced signal from speed sensor) into 0 to 16000 counts via internal 12-bit ADC channel with 12-bit resolution.

The first step is to define fuzzy variable. For fuzzy PID controller, it requires three input variables, Error (E), Change in error (CE) & Sum of Error (SE). These fuzzy inputs are already defined in Equations (2.1), (2.4) and (2.5).

First of all, the input fuzzy variables are scaled to match antecedent of fuzzy rule base. So the E, CE and SE are divided by 4000, 4000 and 8000. Now the membership functions for these variables (Negative (N), Zero (Z) & Positive (P)) are as shown in Fig. 4.1.

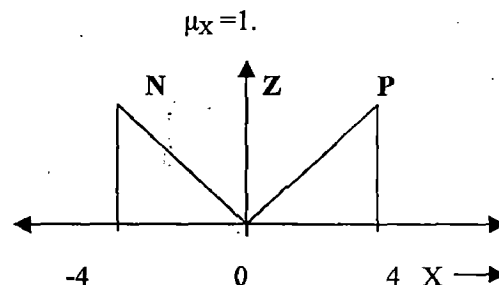


Fig. 4.1 Membership functions for input variables (E, CE & SE)

Now, there is a need to define output fuzzy variable (controlled variable voltage). Basically the output variable is abbreviated as CV. The range of CV is 3600 to 10100 count available from analog output channel of PLC for 0 to 1500 rpm. The membership function for output variable CV is shown in Fig. 4.2 This output is fed to speed control module which drives an output to DC motor such that the actual speed is adjusted to reference speed. There is no need for post scaling because the output from defuzzification is available in the required output range.

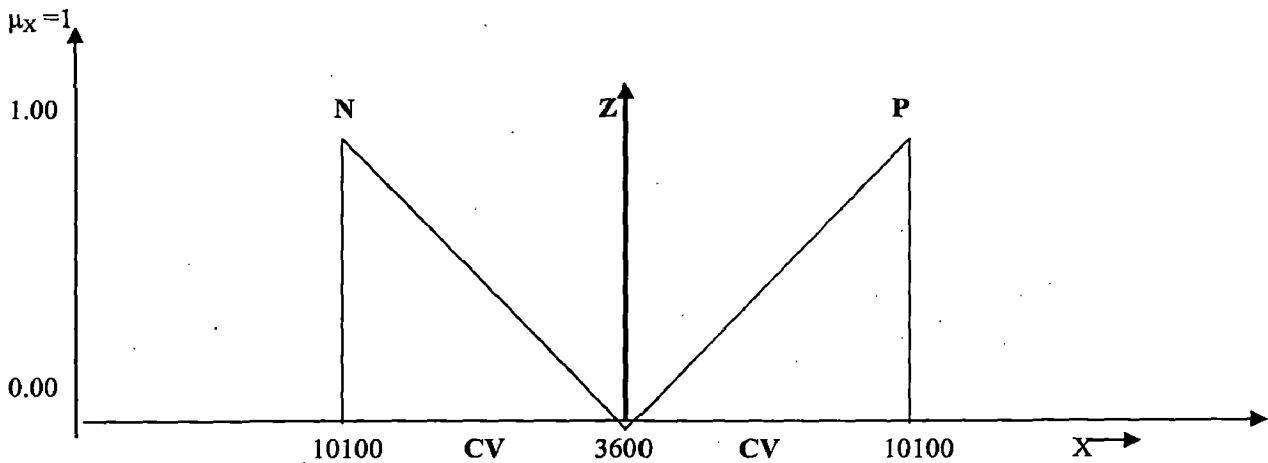


Fig. 4.2 Membership function for output variable (CV)

Now, the next step is to create rule base which can be derived from the complete knowledge and experience of speed control process. The rule base also depends upon types of fuzzy controller which are as follows.

4.1 PLC based Fuzzy PI Controller for Speed

If there is a sustained error in steady state, integral action is necessary. The integral action will increase the control signal if there is a small positive error, no matter how small error is; the integral action will decrease it though the error is negative. A controller with integral action will always return to zero in steady state. The linear approximation to this controller is

$$u(t) = k_p e(t) + k_i \int_0^t e(t) dt \quad (4.1)$$

Where $u(t)$ = CV voltage (3600 to 10100 counts), K_p = proportional gain, K_i = integral gain and SE = sum of error. The two fuzzy inputs (E , SE) are scaled in the range suitable for premises.

$$\int_0^t e(t) dt \approx \sum_{n=0}^k e(n) = SE \quad (4.2)$$

The rule base for fuzzy-PI controller is derived as shown in Table 4.1.

Table 4.1 Rule base for fuzzy PI controller

E	N	Z	P
SE			
N	N	Z	P
Z	N	Z	P
P	N	Z	P

The rules can be read as follows from Table 4.2.

Table 4.2 Output variable (CV) action for FPI controller

E	N	Z	P
SE			
N	Decrease CV	Don't change CV	Increase CV
Z	Decrease CV	Don't change CV	Increase CV
P	Decrease CV	Don't change CV	Increase CV

Then the symbolic expression for a PI like fuzzy controller can be read as

If E is N and SE is N then Decrease the CV.

4.1.1 Fuzzification

FPI (fuzzy PI) controller includes two inputs, E & SE. The range of N, Z & P variables are [-4, 0], [0, 0] & [0, 4]. To fuzzify, division operation is performed on PLC. So each variable is fuzzified and mapped with the membership range in [0, 1]. For N linguistic variable division is performed by -4 while for P linguistic variable division is performed with 4.

4.1.2 Inference Engine

For Inference, Max-min operation is performed. The membership functions for each variable is not overlapping hence single rule is fired at a time. Aggregation is performed by minimum operation. Correspondingly the output is activated as per activation of fuzzy inputs. The aggregation operation for E & SE inputs is expressed as follows.

$$\mu_e \text{ and } \mu_{se} \tag{4.3}$$

Aggregation is equivalent to fuzzification, when there is only one input to the controller. Aggregation is sometimes also called fulfillment of the rule or firing strength, which is denoted by α_k for k^{th} rule. For aggregation, E & SE inputs for a particular sample

are compared for minimum operation through PLC. The minimum degree of membership is denoted as firing strength.

Activation: The activation of a rule is the deduction of the conclusion, possibly reduced by its firing strength. After finding degree of fulfillment, the conclusion is conducted for output variable (CV). Only single rule is activated hence the degree of membership is directly provided by α_k .

4.1.3 Defuzzification

Basically Last of Maxima (LOM) defuzzification method for decreasing membership function of output and First of Maxima (FOM) is used for increasing membership function of output. This technique is used because of computational simplicity. Because of linear relation between membership functions of linguistic variable and output fuzzy variable, the activated α_k is multiplied with the range of CV output variable. The range of CV output variable is 3600 to 10100 counts [total 6500 counts].

$$CV = \alpha_k * 6500 \quad (4.4)$$

For example, one rule is fired with 0.5 degree of membership. So the LOM defuzzified value for output variable CV results in increasing or decreasing of 3250 counts in previous CV output. As shown in Fig. 4.3, the output is activated for decreasing CV count by LOM defuzzification.

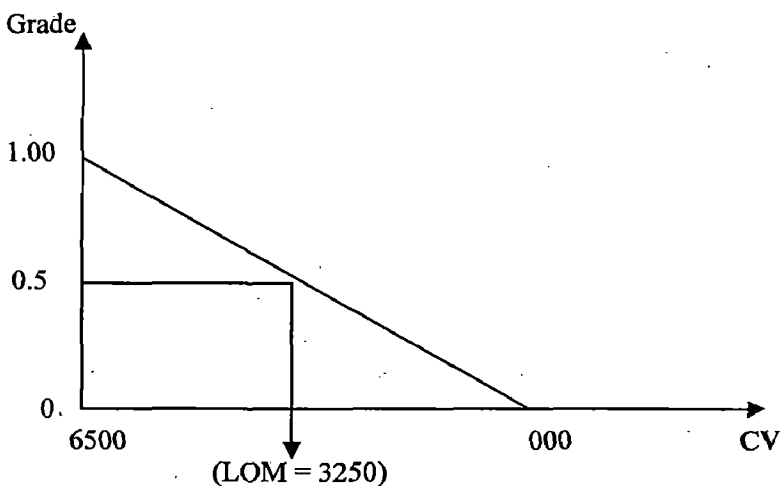


Fig. 4.3 An example for LOM defuzzification performed by PLC

There is no need for post scaling since the output is available in the specified range of PLC analog output required for speed control module.

This fuzzy PI control performed by PLC ladder logic can be described in the form of flow chart as shown in following Fig. 4.4.

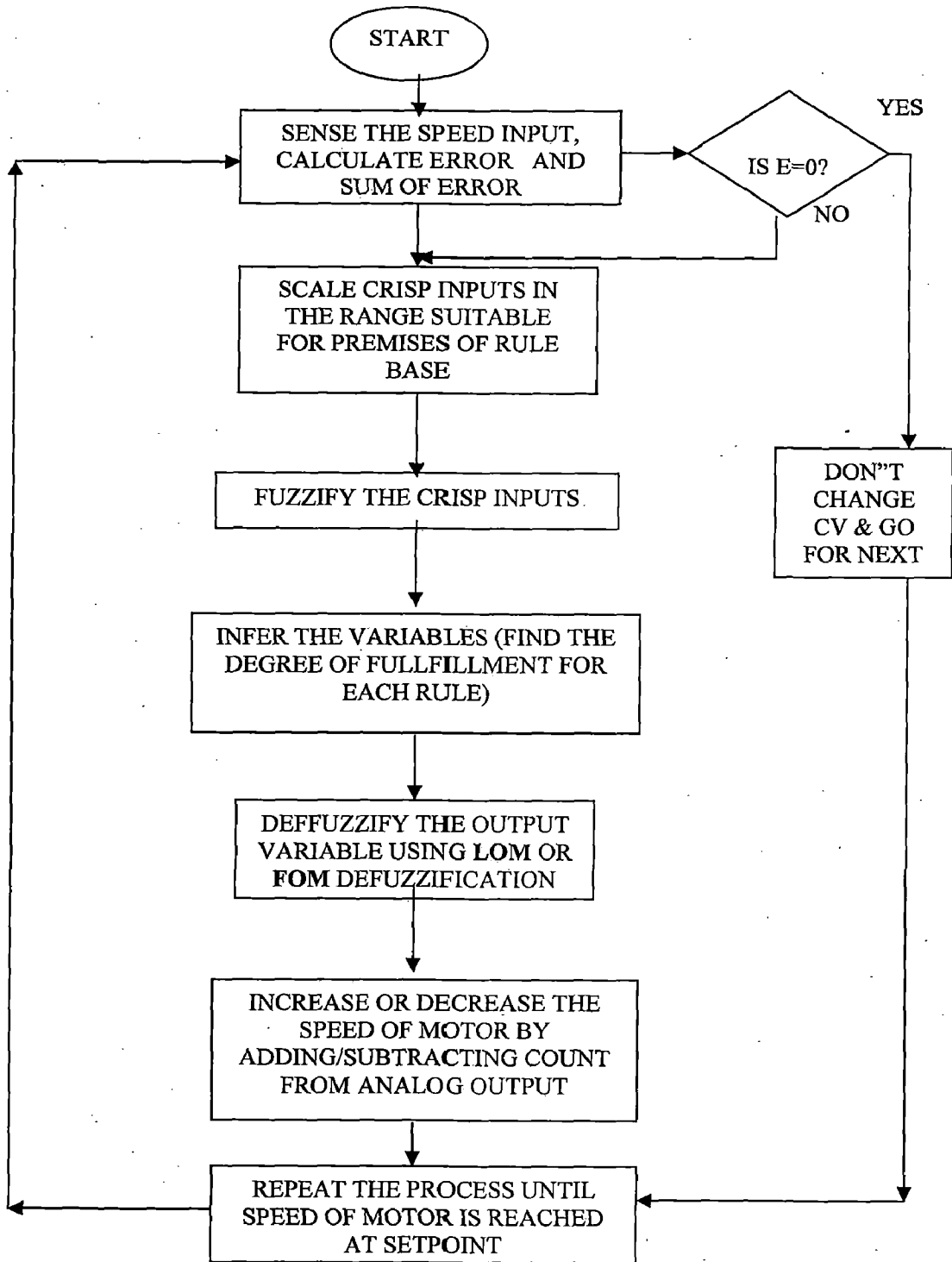


Fig. 4.4 Flow chart of fuzzy PI control for speed realized by PLC software

4.2 PLC based Fuzzy PID Controller for Speed

It is straight forward to envision a fuzzy PID controller with three input terms: error (E), change of error (CE) and sum of error (SE). The linear approximation is as follows:

$$u(t) = k_p e(t) + k_i \int_0^t e(t) dt + k_d \frac{de(t)}{dt} \quad (4.5)$$

Then the symbolic expression for a rule of a PID-like fuzzy controller is “If E is N, CE is N and SE is N, then decrease CV”. This way total 27 rule can be possible for a fuzzy PID Controller because there are 3 linguistic variables for each fuzzy input which can be accumulated in tabular format as shown in Table 4.3.

Table 4.3 Rule base for fuzzy PID control

E	CE	SE	CV
N	N	N	N
N	N	Z	N
N	N	P	N
N	Z	N	N
N	Z	Z	N
N	Z	P	N
N	P	N	N
N	P	Z	N
N	P	P	N
Z	N	N	Z
Z	N	Z	Z
Z	N	P	Z
Z	Z	N	Z
Z	Z	Z	Z
Z	Z	P	Z
Z	P	N	Z
Z	P	Z	Z
Z	P	P	Z
P	N	N	P
P	N	Z	P
P	N	P	P
P	Z	N	P
P	Z	Z	P
P	Z	P	P
P	P	N	P
P	P	Z	P
P	P	P	P

4.2.1 Fuzzification

FPID (fuzzy PID) controller includes three inputs, E, CE & SE. The range of N, Z & P variables are [-4, 0], [0, 0] & [0, 4]. To fuzzify, division operation is performed on PLC. So each variable is fuzzified and mapped with the membership range in [0, 1]. For N linguistic variable division is performed by -4 while for P linguistic variable division is performed with 4.

4.2.2 Inference Engine

For Inference, Max-min operation is performed. The membership functions for each variable is not overlapping hence single rule is fired at a time. Aggregation is performed by minimum operation. Correspondingly the output is activated as per activation of fuzzy inputs. The aggregation operation for E, CE & SE inputs is expressed as follows.

$$\mu_e \text{ and } \mu_{ce} \text{ and } \mu_{se} \quad (4.6)$$

Aggregation is equivalent to fuzzification, when there is only one input to the controller. Aggregation is sometimes also called fulfillment of the rule or firing strength, which is denoted by α_k for k^{th} rule. For aggregation, E, CE & SE inputs for a particular sample are compared for minimum operation through PLC. The minimum degree of membership is denoted as firing strength.

Activation: The activation of a rule is the deduction of the conclusion, possibly reduced by its firing strength. After finding degree of fulfillment, the conclusion is conducted for output variable (CV). Only single rule is activated hence the degree of membership is directly provided by α_k .

4.2.3 Defuzzification

Basically Last of Maxima (LOM) defuzzification method for decreasing membership function of output and First of Maxima (FOM) is used for increasing membership function of output is used because of computational simplicity. Because of linear relation between membership functions of linguistic variable and output fuzzy variable, the activated α_k is multiplied with the range of CV output variable. The range of CV output variable is 3600 to 10100 counts [total 6500 counts].

$$CV = \alpha_k * 6500 \quad (4.7)$$

For example, one rule is fired with 0.5 degree of membership. So the LOM defuzzified value for output variable CV results in increasing or decreasing of 3250 counts in previous CV output. As shown in fuzzy PI controller, the output is activated for decreasing CV count by LOM defuzzification.

There is no need for post scaling since the output is available in the specified range of PLC analog output required for speed control module.

This fuzzy PID control performed by PLC ladder logic can be described in the form of flow chart as shown in Fig. 4.5.

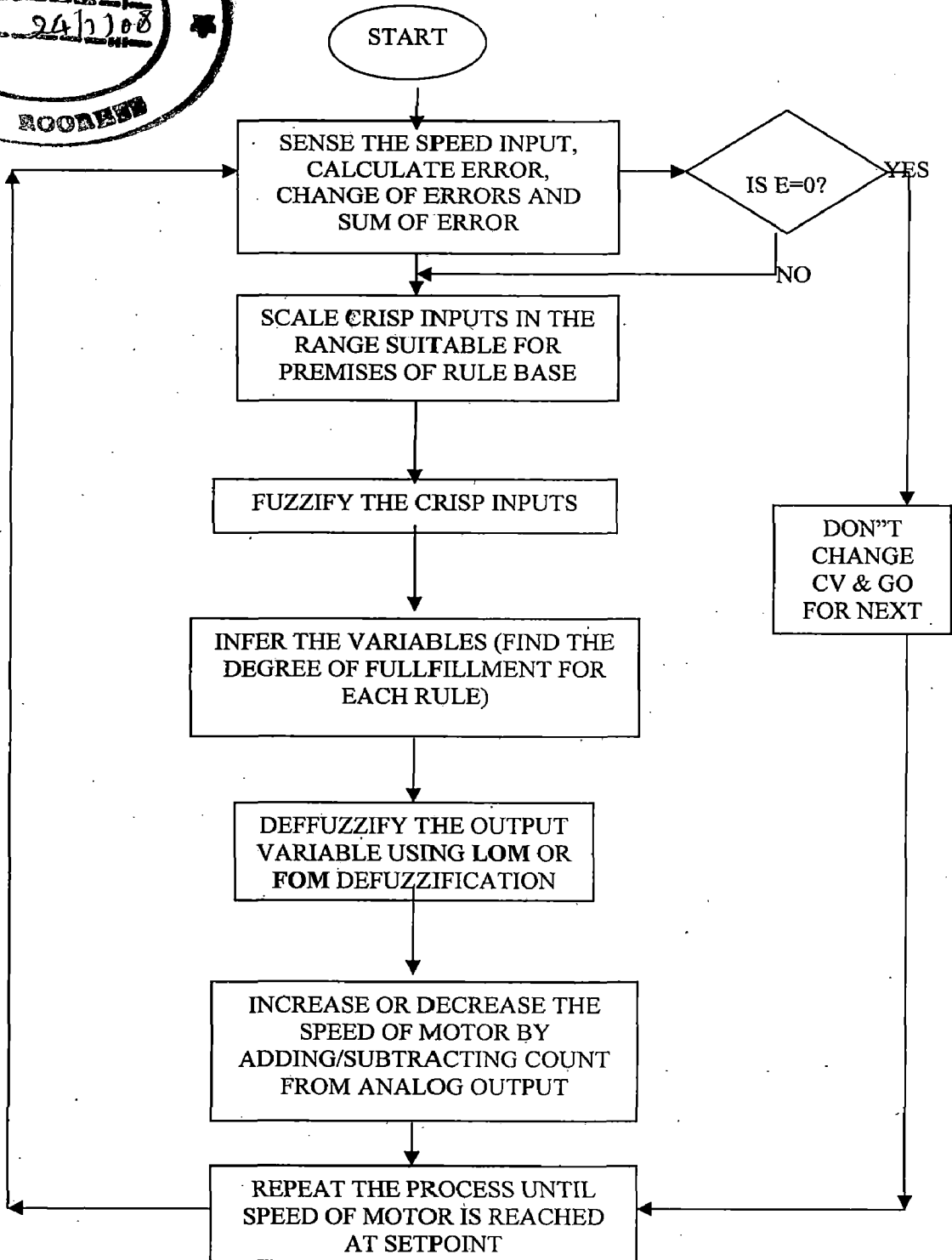
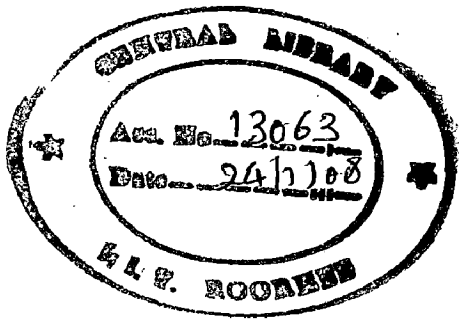


Fig. 4.5 Flow chart of fuzzy PID control for speed realized by PLC software

4.3 Experimental Results

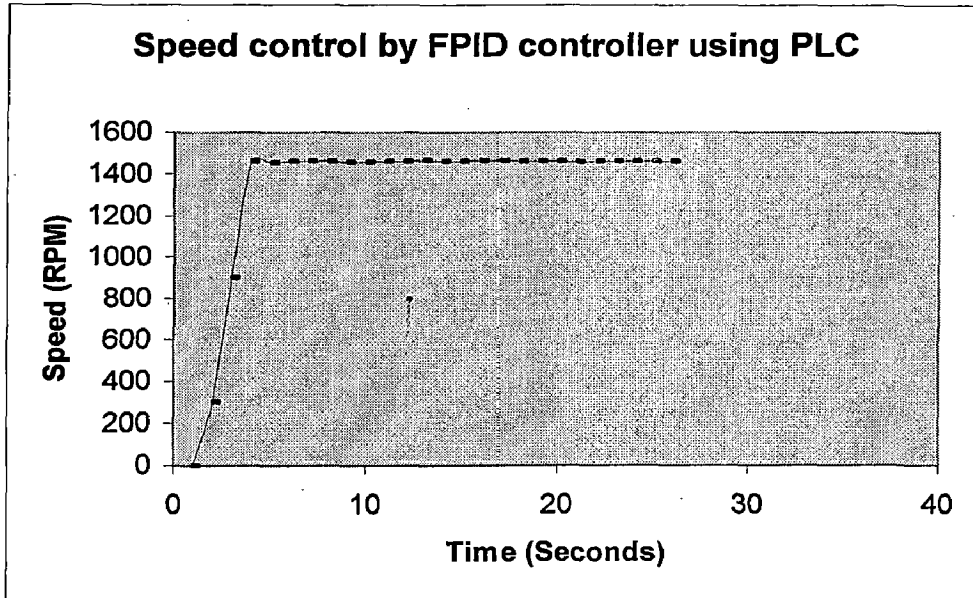


Fig. 4.6 Fuzzy PID control for 1500 rpm speed realized by PLC software.

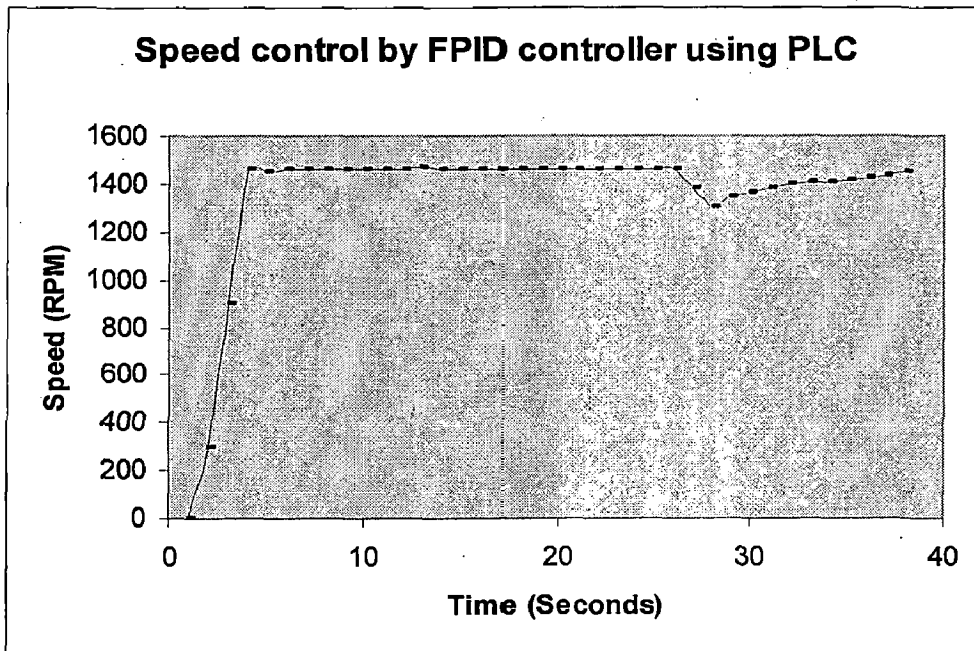


Fig. 4.7 PLC based FPID control for speed with load disturbance

As shown in Fig. 4.6, the speed of 1500 rpm is controlled by fuzzy PID algorithm based on PLC. The settling time is 3 seconds and the rise time is 2.5 seconds. When the load disturbance is applied externally as shown in Fig. 4.7, the speed is reached at setpoint (1500 rpm) and PLC keeps speed at setpoint though the load variations.

As shown in following Fig. 4.8, the speed of motor is controlled at 1500 rpm by fuzzy PI control implemented by PLC ladder program. Here slight oscillation and overshoot are easily highlighted near setpoint from following figure. The settling time is 6 seconds and rise time is 5 seconds.

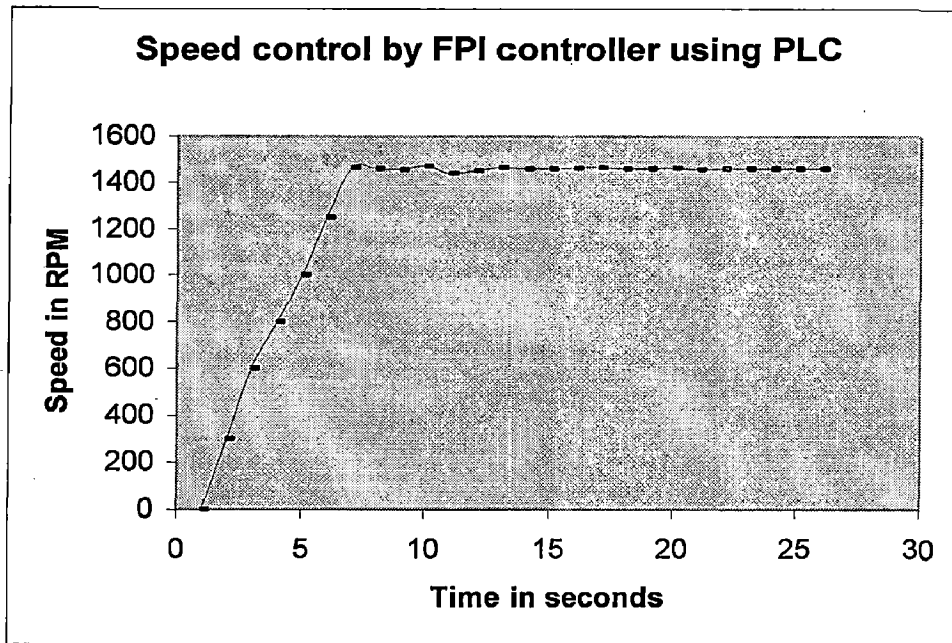


Fig. 4.8 Fuzzy PI control for 1500 rpm speed realized by PLC software

RESULTS AND DISCUSSIONS

In the present work, four different types of controllers such as PI, PID, FPI, and FPID have been implemented. The results for these techniques have already been presented in Fig. (2.7), (2.9), (4.6) and (4.8), respectively. A comparative performance table is presented in Table 5.1.

Table 5.1 Comparison between conventional and fuzzy logic based speed control realized by PLC software

Control	PI	PID	FPI	FPID
Set Point	750 RPM	1500 RPM	1500 RPM	1500 RPM
Overshoot	8 %	0 %	1%	0%
Settling time	5 Seconds	15 Seconds	6 Seconds	3 Seconds
Rise time	3 seconds	12 Seconds	5 Seconds	2.5 Seconds

An observation on these Figures reveals that the PID control is better than PI control because of oscillating characteristic of integral action. The overshoot is reduced in FPI control than ordinary PI control. Fig. 2.8 and Fig. 4.7 present response to load disturbances. As indicated in Fig. 2.8, the PID control on PLC responds to disturbances and gradually reaches to setpoint. Comparing Fig. 2.8 and Fig. 4.7, it is observed that the FPID responds faster than PID algorithm.

In both speed control strategies (conventional and fuzzy) which are realized by a PLC, the setpoint is adjusted either manually via second analog input which receives voltages through potentiometer or entered through program before PLC program execution.

Manual setpoint adjustment is more suitable because we can easily track how fast the controlled variable (speed output) is available from PLC. We can change the setpoint constantly during PLC program execution and PLC based FLC or PID responds to these setpoint changes. Thus keeps the speed of motor at a set speed available at that instant. While in the case of set speed adjusted through program, it remains fixed during PLC

program execution and the operator is not allowed to change the setpoint during program execution.

In the present work, the load disturbance for speed is applied through magnetic coupling (attraction) of DC motor shaft and magnet which is externally applied to shaft of motor. Hence we can increase or decrease the load by increasing or decreasing the magnetic attraction between shaft and magnet.

It is easily revealed from above comparison between conventional and fuzzy logic control implemented by PLC for speed control of DC motor that PLC based fuzzy PI or fuzzy PID speed control follows setpoint change, which is adjusted through program or varied manually via analog potentiometer and second analog input, and load disturbances more quickly than PLC based PI and PID speed control. It is also observed from Fig. 4.6 and 2.7 that fuzzy PID control provides more smooth response than PID control near setpoint. In PLC based PID control, though the speed is controlled at setpoint, after some time speed deviates slightly and PID controller responds to this deviation for maintaining the speed at setpoint. This limitation can be easily overcome in fuzzy logic based speed control realized by a PLC.

In overall the fuzzy PID control based on PLC is faster than other conventional control strategies for the DC motor speed control and it quickly responds to load disturbances.

CHAPTER 6

CONCLUSIONS AND SCOPE FOR FUTURE WORK

Programmable Logic Controllers (PLCs) are industrial control system and they have an important role especially for medium and large scaled industries. Nowadays fuzzy logic control has become one of the most popular control methods and the industries have great expectations from this control schematic.

In the present work, speed of DC motor is controlled by a PLC based fuzzy logic controller which performs fuzzy logic processes. The speed of DC servomotor is controlled up to 1500 rpm by using ladder logic program of PLC (GE/7 MicroPLC) which performs intelligent fuzzy logic control. The speed is indicated constantly on rpm indicator available with module. The speed is also indicated online by giving controlled variable to LABVIEW hardware card.

In PLC based PID control for speed, the speed of DC motor is controlled by sensing an analog input (PV) by PLC. The PLC applies the PID mathematical equation on error and then feeds the control variable (CV) to speed control module by generating analog output of PLC. This analog output from the speed control module is used to drive the DC motor.

In PLC based fuzzy logic control for speed, PLC senses the analog input; calculates Error (E), Change in error (CE) & Sum of Error (SE); scales them and fuzzyfies these input variables. Further it infers these input variables as per fuzzy PI or fuzzy PID control by considering rule base; defuzzyfies the controlled variable output and drives the controlled variable of process by modifying (increasing or decreasing) an analog output. The other analog input of PLC is used to provide setpoint through analog potentiometer. This set speed can also be adjusted from 0 to 1500 rpm by adjusting analog potentiometer and giving the voltage to the analog input during program execution.

PLC based fuzzy PI or fuzzy PID speed control follows setpoint changes and load disturbances more quickly than PLC based PI and PID speed control. The setpoint is adjusted through program or varied manually via analog potentiometer and second analog input.

PID controller is suited for stable and linear conditions of process. In the case of load disturbances and sluggish process which is controlled by a PLC, fuzzy logic based PI or PID control is better than conventional PID control. In these cases, the extension or replacement of conventional PID controllers with fuzzy based PI or PID controllers is more feasible.

Fuzzy logic lets engineers to design controllers from operator experience and experimental results rather than from mathematical models. PLC based fuzzy PID controller for speed is faster than fuzzy PI and conventional control (PI or PID) based on PLC.

SCOPE FOR FUTURE WORK

This present work illustrates the speed control of DC motor by PLC having intelligent techniques like fuzzy logic control. This scheme can be applied for real world problems like speed control of DC motors/ drives/ generator or turbine in industries.

A generalized programming approach has been used here for PLC based fuzzy logic controller of speed. Thus if speed control module is replaced by temperature control module with temperature bath and a temperature sensor, this control scheme may become a PLC based fuzzy logic controller for temperature. A work in this direction can also be tried.

Fuzzy logic control can also be used to supervise the process. Since the setpoints of individual PID controllers with or without PLC are increased or decreased by FLC implemented on a PLC, this can eliminate the need of operators. Thus it becomes a supervisory control process which is suitable for multivariable control.

REFERENCES

- [1] Karl Johan Astron and Tore Hagglund, "PID Controller; Theory, Design and Tuning", Instrument Society of America, 2nd Edition. 1934.
- [2] Yangwon Kwon, Jonku park, Haksoo Kang and Taeknon Ahn, "Application of Fuzzy-PID Controller Based on Genetic Algorithm for Speed Control of Induction Motors," Proceedings of the 14th KACC, October 1999, pp 309–312.
- [3] Mizumoto, M, "Realization of PID Controls by Fuzzy Control Methods", First International Conference on Fuzzy Systems, the Institute of Electrical and Electronics Engineers (IEEE), San Diego, 1992, pp. 709–715.
- [4] Gebhardt, J. and Muller, R, "Application of Fuzzy Logic to the Control of a Wind Energy Converter", First European Congress on Fuzzy and Intelligent Technologies (EUFIT-93), Aachen, September, 1993.
- [5] Constantin Von Altrock, "Recent Successful Fuzzy Logic, Applications in Industrial Automation", (www.inform-ac.com)/(www.fuzzytech.com)
- [6] Zadeh.L.A., "Fuzzy Sets, Information and Control Vol. 8", 1965, pp.338–353.
- [7] Gupta.M.M., Kiszka. J.B. and Trojan.G.M., "Multivariable structure of Fuzzy Control Systems", IEEE Transactions on Systems, Man and Cybernetics, 1986, pp.94–102.
- [8] Madami.E.H. and Assilian.S., " A Study on the Application of Fuzzy Set Theory to Automatic Control," Proceeding of the IFAC Stochastic Control, Symp. Budapest, 1974.
- [9] GE/7 (23-point MicroPLC), "User Manual, Programmable Logic Controllers", VI Microsystems pvt. Ltd.
- [10] Hughes Thomas A., "Measurement and Control Basics", ISA publication, 2nd Edition, 2000.
- [11] Allen-Bradley, "Processor manual PLC-5 Family Programmable Controllers", Allen-Bradley Co.

- [12] Bryan L.A and E.A Bryan, "Programmable Controllers: Theory and Implementation", Industrial Text Co., 1998.
- [13] Hugh Jack, "Automating Manufacturing Systems with PLCs", GNU Publication, Version 4.7, 2005,
- [14] Onur Karasakal, Engin Yesilmujde, G U Zelkay and Ibrahim Eksin, "Implementation of a New Self-Tuning Fuzzy PID Controller on PLC", Turk Journal of Electrical Engineering, VOL-13, NO-2, 2005.
- [15] Brehm, "Fuzzy Logic Controller: Analysis and Design", M.S. thesis, Wright State University, winter 1994.
- [16] OMRON, "Sysmac C200H-FZ001 Fuzzy Logic Unit Operation Manual", 1992.
- [17] J.Klir, Yuan, Bo, "Furry Sets and Fuzzy Logic- Theory and Applications", Mc Graw-Hill Inc., 1995.
- [18] Nalunat, Khongkoom, Attapol Kanchanathep and Sutichai Nopnakeepong, "Control of the Position DC Servo Motor by Fuzzy Logic", IEEE conference on Fuzzy Logic, Vol-13, No-3, 2000, pp 354-357
- [19] Hogener.J., " Fuzzy – PLC: A Connection with a Future, " EUFIT' 93 – first European Congress on Fuzzy and Intelligent Technologies in Aachen, 1993, pp.688–691.
- [20] Becker.K., Kasmacher.H., Rau.G., Kalff.G., and Zimmerman.H.J., "A Fuzzy Logic Approach to Intelligent Alarms in Cardio Anesthesia", Third IEEE Conference on Fuzzy Logic, 1994, pp.2072–2076.
- [21] Zakharov Alexei, Halasz Sandor, "Robust Speed Fuzzy Logic Controller for DC Drive", 1965.
- [22] Chan, C. C., "Low Cost Electronic Controlled Variable Speed Reluctance Motors", IEEE Transactions on Industrial Electronics, Vol-34, No-I, February 1987, pp 95–100
- [23] LaMeres B.J. and Nehrir M.H., "Fuzzy Logic Based Voltage Controller for a Synchronous Generator," IEEE Computer Applications in Power Systems, Vol-12, No-2, April 1999, pp 46–49.

- [24] Lee, C. C., "Fuzzy logic in control systems: Fuzzy logic controller", IEEE Transactions on Systems, Man and Cybernetics, Vol-20, No-2, 1990, pp 404–435.
- [25] Sabharwal D and Rattan K.S, "A Proportional- Plus-Derivative Rule-Based Fuzzy Controller," proceedings of the IEEE International Conference on Systems Engineering, Dayton, H, August 1991, pp 229–233.
- [26] Driankov, Hellendroom, Reinfrank, "An Introduction to Fuzzy Control, Narosa Publishing House, 1993.
- [27] Ross J.T, "Fuzzy Logic within Engineering Applications", Mc Graw-Hill Inc., 1995.
- [28] IEC, "Programmable Controllers: Fuzzy Control Programming", Technical Report, IEC 1131, International Electrotechnical Commission. (Draft.), Part 7, 1996.
- [29] Salcic Z, "High Speed Customizable Fuzzy Logic Processor: Architecture and Implementation," IEEE Transactions on Systems, Man and Cybernetics, Part A, Vol-31, pp. 731–737.