ABSTRACT

Design and Implementation of Fuzzy Logic Controller for a Process Control Application

by

Sujit S. Nerurkar

Many industrial applications of fuzzy logic control have been reported. This thesis studies and reports the problems associated with the Heat-exchanger temperature control via conventional PID control implemented with Programmable Logic Controllers (PLC) and provides an example of design and implementation of fuzzy logic controllers (FLC's) for a Heat exchanger in a Water for Injection (WFI) system.

After a basic FLC was designed and tested, it is shown how its rule base evolved to achieve superior performance by utilizing additional low-cost sensing information in the process and its environment. A method for the implementation of FLC's into the existing PLC is discussed. The system performance of the five designed FLC rule-base strategies is compared with that of the existing PID controller and it is concluded that better performance can be achieved by using the fuzzy logic control technology.

Finally, this thesis discusses some blocking problems in widespread industrial applications of FLC's and the possible solutions to them.

DESIGN AND IMPLEMENTATION OF FUZZY LOGIC CONTROLLER FOR A PROCESS CONTROL APPLICATION

by
Sujit S. Nerurkar

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

January 1994

APPROVAL PAGE

Design and Implementation of Fuzzy Logic Controller for a Process Control Application

Sujit S. Nerurkar

Dr. Constantine N. Manikopoulos, Thesis Advisor Date
Associate Professor, Dept. of Electrical and Computer Engineering, NJIT

Dr. MengChu Zhou, Committee Member Date
Assistant Professor, Dept. of Electrical and Computer Engineering, NJIT

Dr. E.S.H. Hou, Committee Member Date
Assistant Professor, Dept. of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Sujit S. Nerurkar

Degree: Master of Science in Electrical Engineering

Date: January 1994

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 1994

- Bachelor of Science in Electronics and Telecommunications,
  Karnataka University, Belgaum, India, 1986

Major: Electrical Engineering

ACKNOWLEDGMENT

I would like to express my sincere thanks to my advisor Prof. Constantine N. Manikopoulos for his encouragement, broadmindedness, and warmth of personality. It is nice to be allowed to step to the beat of your own drummer.

I am very grateful to Dr. MengChu Zhou for reviewing this thesis and providing motivation, guidance, and constructive criticism, each in the proper dosage.

Special thanks to Harish, who got me going, was always ready to help and answered so many questions.

TABLE OF CONTENTS

Chapter **Page**

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 General

Fuzzy logic is a technique for doing computations on a digital computer without specifying explicit mathematical formulas. These processes are concerned with continuous phenomena that are not easily broken down into discrete segments, and the concepts involved are difficult to model, sometimes extraordinarily so, along mathematical model-based lines.

Fuzzy logic permits the use of different degrees of truth and allows for a multitude of variables to be considered in the membership functions. Though the word "fuzzy" hints vagueness, there is actually nothing fuzzy about the logic itself. It's the concepts and words used in fuzzy rules that are ambiguous or fuzzy. Instead of a black-or-white alternative, it accommodates the gray areas that also contribute towards the membership functions. Fuzzy logic makes sense through a gradation of numerical values that range between 0 and 1. These gradations are finally used to derive a crisp solution which has a weighted contribution of each process variable.

The "fuzzy philosophy" uses human intuitive expertise to solve a control problem. Instead of complex and sometimes impractical mathematical equations, it relies on qualitative descriptions rather than precise quantitative numbers to describe variables.

1.2 Fuzzy Control

In a conventional proportional, integral, and differential (PID) controller, what is modeled is the system or process being controlled, whereas in a fuzzy logic controller, the focus is the human operator's behavior. In the first case, the system is modeled analytically by a set of differential equations, and their solution tells the PID controller how to adjust the system's control parameters for each type of behavior required. In the fuzzy controller, these adjustments are handled by a fuzzy rule-based expert system, a logical model of the thinking processes a person might go through in the course of manipulating the system. This shift in focus from the process to the person involved, changes the entire approach to automatic control problems.

The inference rules in the fuzzy expert system may take the form "if observed variable x is 'positive medium,' then change the control variable y by the amount 'negative medium.'" The model derives the designation "fuzzy" from its use of such terms as "positive medium," "positive large," and "no change," which in turn form a fuzzy subset of the associated measurement domain. As such, the system being controlled is formally viewed as a fuzzy system. This is why fuzzy controllers are simpler than conventional PID controllers.

Fuzzy logic works by accepting and processing analog input signals to judge or infer reliable conclusions from a combination of a few variable inputs, either calculated or received from an analog output device. It is best used in applications such as set point control (error nulling), discrimination (sorting), identification, and image processing.

1.3 When to use Fuzzy Logic?

Fuzzy logic has rapidly become one of the most successful of today's technologies for developing sophisticated control systems. With its aid, complex requirements may be implemented in amazingly simple, easily maintained, and inexpensive controllers.

An example might be distillation of crude oil in the petroleum industry to obtain various grades of fuels, from the purest form of gasoline to the lowest grade of lubricants. The crude oil is sprayed into the distillation column, which is heated under pressure. The purest form of gasoline being the most volatile is distilled first followed by unleaded gasoline, leaded gasoline, diesel and other crude oil products. The control rules for a system of this nature might include such variables as the crude oil spraying rate, the temperature of the distillation column and the pressure at which the distillation column is compressed. These variables are all continuous, and the range of their values subject to interpretation by the system designer.

Thus the variable "temperature" might have a range of states: cold, cool, moderate, warm, hot, very hot. Yet, the change from one state to another is not precisely defined. At no point can an increase of a tenth of a degree be said to change "this is warm" into "this is hot". Consequently, the idea of what is cold, what is warm, and what is hot is subject to different interpretations by different experts at different points in the variable's domain and is analyzed qualitatively rather than quantitatively.

With fuzzy logic, control statements are written in terms of these imprecise ideas of what constitutes the states of the variable. For example, a fuzzy rule in a crude oil distillation system might be:

> *If column temperature is* **Warm** *AND pressure is* **Not very high**, *then octane rating is* **High**,

while a rule in a conventional proportional-integral-derivative (PID) controller would need to be very specific:

> *If column temperature is greater than 40degC AND pressure is less than 45, then octane rating is 93.*

Although these look a lot like traditional computer program statements, the adjectives in the rules (Warm, Not very high, High) don't correspond to precisely defined ranges. Thus, the temperature of 40 degC could be described as "warm", but the temperature of 45 degC is "warm" to a greater degree. Fuzzy logic takes this into account, so the same rule can work for many different situations.

Fuzzy logic rules are more natural, and technically more expressive, for designers who have no mathematical background of the system models. In the case of the crude oil distillation system the fuzzy logic rules will fire over a wider range of temperatures and pressures. Thus one fuzzy rule can replace many conventional rules. Fuzzy logic creates a control surface by combining rules and fuzzy sets, thus allowing designers to build controllers even though their knowledge of the mathematical model of the system is incomplete.

So, when is it appropriate to use fuzzy logic? When one or more of the control variables are continuous; when a mathematical model of the process does not exist, or exists but is too difficult to encode, or is too complex to be evaluated fast enough for real-time operation, or involves too much memory on the designated chip architecture; when ambient noise levels must be dealt with or it is important to use inexpensive sensors and/or low precision microcontrollers; and perhaps above all, when an expert is available who can specify the rules underlying the system behavior and the fuzzy sets that represent the characteristics of each variable.

1.4 Overview of Thesis

This thesis is organized as follows:

Chapter 2 describes the conventional PID controller. The problems associated with its control action, e.g., offset error, overshoot, and cycling are discussed.

Chapter 3 provides the basic terminology and underlying concepts of the fuzzy set theory in brief.

Chapter 4 presents the idea of a fuzzy logic controller and analyzes the different strategies involved in fuzzification, defuzzification, and generation of rule-bases.

Chapter 5 presents a structured methodology to follow in developing a fuzzy logic application.

Chapter 6 presents the basic design of a fuzzy logic controller for a Heat-exchanger in a Water for Injection (WFI) system. Several rule-bases were evolved to achieve superior performance and the problems that came forth and the strategies used to overcome them are also presented.

Chapter 7 concludes the topic, discussing the advantages and disadvantages of incorporating fuzzy logic in software. Some of the problems present in industrial applications of FLC's and their possible solutions are also discussed.

CHAPTER 2

CONVENTIONAL PROCESS CONTROL

2.1 General

Any study of process control must begin by investigating the concept of a "process". A process [11] is identified as having one or more variables associated with it, which are important enough for their values to be known and for them to be controlled.

We will concentrate on processes having only one controlled variable, such as temperature in the heat-exchange process as shown in Figure 1.

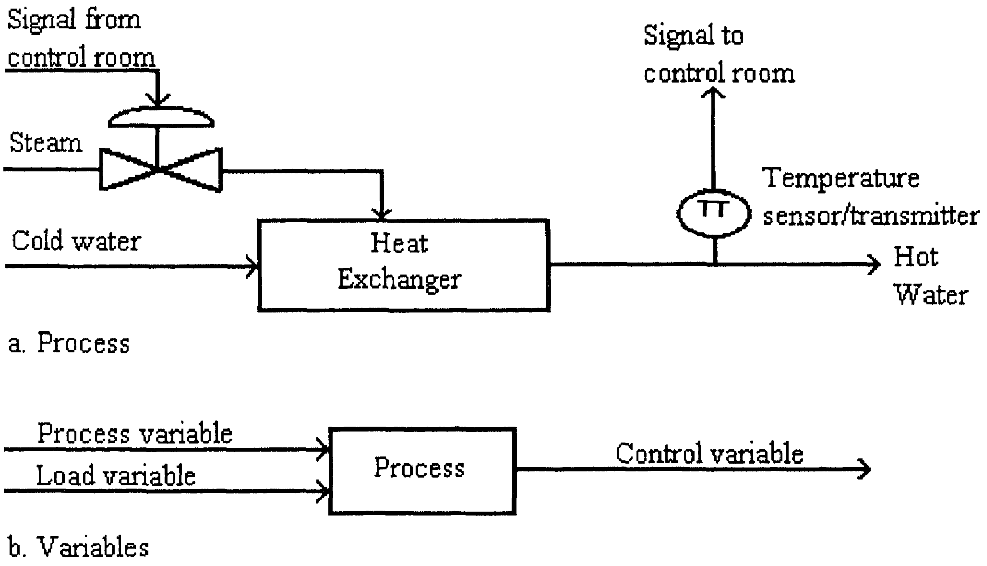


Figure 1 Process - Heat exchanger.

To maintain the temperature of the product (hot water) in this process, another variable influencing the variable being controlled must be available for manipulation by the control system. In this example, the control system manipulates the position of a steam valve. However, the temperature of the water depends not only on the position of this valve but also on the flowrate of the water, its inlet temperature, the enthalpy of the steam, the degree of fouling in the exchanger, and the ambient temperature.

The parameters that indicate product quality of the operating condition of the process are called process variables (PV), such as pressure, level, temperature, pH, motor speed, and other variables, depending on the process. Parameters that control the product quality are called control variables (CV), such as valve position, damper position, motor speed, etc.

All variables affecting a PV, other than the CV, are defined as loads. Both loads and the CV may influence a PV from either the supply side or the demand side of the process. For example, the outlet temperature of a heat exchanger can be controlled by manipulating the steam valve, while tank level can be controlled by manipulating a valve on the outflow from the tank. Often, a PV in one process is a load variable for another. For example, the temperature of the outlet stream from a heat exchanger will almost certainly affect other plant variables -- otherwise, it would not be important enough to control.

2.2 The Control Problem

Changes in the process variable reflect the balance between the loads and the controlled variable. For the heat exchanger, increases in steam-valve opening, steam enthalpy, inlet temperature, and ambient temperature tend to raise the product temperature, while it is lowered by increases in flowrate. The temperature responds to the net effect of these influences. If the positive influences are greater than the negative, the temperature will rise. If the reverse is true, the temperature will fall. If load variables were to remain constant, the steam valve could then be adjusted until the product temperature was constant at the desired value, and would remain there indefinitely.

Process control equipment is needed because these variables do not remain constant. For example, variations in inlet temperature and flowrate both upset product temperature, and require a different steam-valve position in order for the water temperature to be maintained at the desired value. The job of the control system is to determine and continuously update this valve position as load conditions change.

Generally, the control problem is to determine the one value of the control variable that establishes a balance among all the influences on the process variable and keep the variable steady at a desired value. Other factors such as speed of response and shape of response are also important in designing control systems. The control problem can be solved in only two ways, each of which corresponds to a basic control-system design philosophy. *Feedback* systems generate the control signal on the difference between the actual and the reference measurement values. For *feedforward* systems, the control signal is generated from values based on the various load variables as they affect the process.

2.3 Feedback Control Systems

Feedback systems are more common than feedforward ones. The structure of a feedback loop [12] is shown in Figure 2.

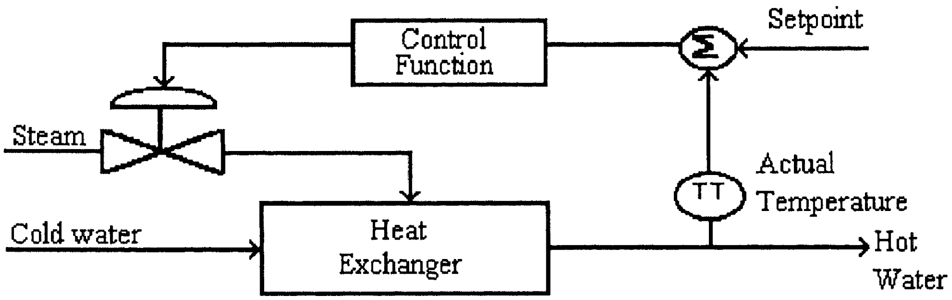


Figure 2 Feedback Control.

Here, the value of the process variable responds to the net effect of the loads and the control variable. A sensor/transmitter measures the current value of the process variable and sends a signal to the feedback controller, where the signal is compared (by subtraction) to a reference value (setpoint). The control function within the controller generates a signal, which positions a valve on the basis of the sign and magnitude of the difference between the measurement and the reference or setpoint values.

In the example for the heat exchanger, a temperature transmitter continuously generates a signal that represents the actual temperature of the hot water. At the controller, this signal is subtracted from an operator-set value that represents the desired temperature. If these values are the same, the current position of the steam valve is correct, and the controller will not change its output. However, if the actual value is below the reference value, the controller will change its output in the direction that opens the steam valve and raises the actual temperature. Conversely, if the actual temperature is above the desired one, the controller will change its output in the direction that closes the steam valve, to lower the actual temperature.

Thus, a feedback controller solves the control problem through a trial-and-error procedure [12]. A change in the load variables upsets the temperature, and a new valve position is required. The controller becomes aware of the upset when the imbalance between the loads and the control variable begins to change the process variable. The controller immediately begins to make corrective changes in its outputs -- even as it monitors the effect of these changes on the process variable. When the controller sees that its corrections have returned the process variable to the desired value (i.e., difference equals zero), it holds the output steady and continues to observe the controlled variable, and waits for the next upset.

2.3.1 Inside A Feedback Controller

Regardless of the hardware used for implementation, the concept of feedback control remains the same [12]. The state of the art today is distributed control through digital systems, and controllers now often exist in software. Digital systems have an extensive selection of features such as automatic alarming, output clamps, and built-in linearization or signal compensation.

All feedback controllers have certain common elements. The feedback-control function always has two inputs and one output. One input is the process variable from the transmitter; the other, the reference value. The reference signal is called the setpoint, which usually represents the desired value of the measurement.

Within the controller, measurement and setpoint values are compared by subtraction. The difference is called the error and is the input to the mechanism, circuit or algorithm that generates the output. Generally, this response contains proportional, integral and derivative (PID) components, although they may not all be present in every controller. Proportional or integral responds to error, while derivative usually responds directly to measurement. The sum of the individual responses forms the automatic control signal.

2.4 PID Controller

One of the most powerful but complex controller mode operations combines the proportional, integral, and derivative modes. The analytic expression [16] for a standard PID controller is

$$P = KpEp + KpKi\int_0^t Epdt + KpKd\frac{d}{dt}Ep + Po$$

where,

P = the controller output (0 - 100%)

Kp = Proportional gain constant

Ki = Integral gain constant

Kd = Derivative gain constant

Ep = Error i.e. PV - SP or SP - PV, depending on whether it is a direct acting or reverse acting controller

Po = Bias or feedforward.

The proportional constant *Kp* expresses a linear relationship between the controller output and the error. Thus, over some range of errors about the setpoint, each value of error has a unique value of controller output in one-to-one correspondence.

The integral constant *Ki* relates the changes in the rate of controller output with changes in error. Thus, a large value of *Ki* means that a *small* error produces a *large* rate of change of output and vice versa. For example, in the case of the heat exchanger, the controller output value initially begins to change very rapidly, but as the valve opens, the error decreases and slows the valve opening rate.

The derivative constant *Kd* relates the extent of the controller output to the rate at which the error is changing and not on the value of the error. The derivative action is also known as rate of anticipatory control.

2.4.1 Problems With The Standard PID Control

An important characteristic of the proportional control mode is that it produces a permanent *residual error* in the operating point of the controlled variable when a change in load occurs. This error is referred to as *offset* [16]. It can be minimized by a larger constant Kp, which also reduces the proportional band. The proportional band is defined as that positive and negative error for which the output will be driven to 0% and 100%. To see how offset occurs, consider a system under nominal load with the controller at 50% and the error zero as shown in Figure 3.

If a transient error occurs, the system responds by changing controller output in correspondence with the transient to effect a return to zero error. Suppose, however, a load change occurs that requires a permanent change in controller output to produce the zero error state. Because a one-to-one correspondence exists between controller output and error, it is clear that a new, zero error controller output can never be achieved. Instead, the system produces a small permanent offset in reaching a compromise position of controller output under new loads.

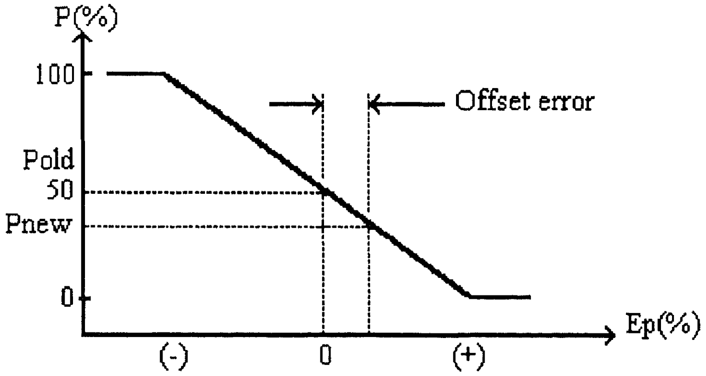


Figure 3 Offset Error in a Proportional Controller.

The effect of process and control system lag is shown as simple delays in the controller output change and in the error reduction when the controller action occurs. If the process lags are too large, the integral action causes a considerable overshoot of the error and output before settling to the operation point. Also, the error can oscillate about zero or even be cyclic. This is shown in Figure 4, where we see the proportional band as a dashed band. The effect of the integral action can be viewed as a shifting of the whole proportional band [16].

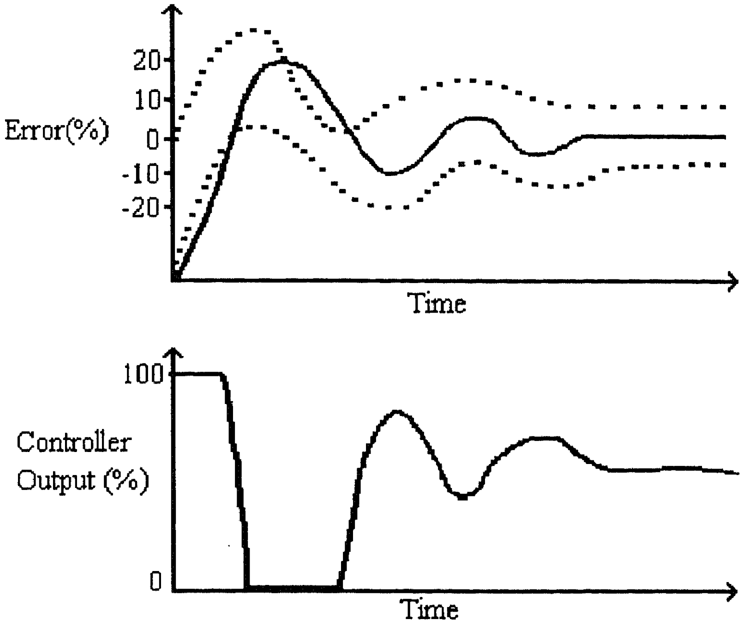


Figure 4 Overshoot and Cycling in a PI Controller.

CHAPTER 3

FUZZY SET THEORY AND TERMINOLOGY

3.1 General

Conventional set theory principles, are based on Aristotelian logic, and form the basis of modern mathematics [1]. Fundamental to this basic set theory is the notion that an item is either a member or it is not a member of a given set; $x \in A$ or $x \notin A$. However, in the real world, the membership of an item in a set is not always so distinct. Fuzzy set theory is based on a recognition that certain sets have imprecise boundaries. Fuzzy sets or subsets are those collections of items with unclear boundaries of vagueness in which the transition from membership to nonmembership in a subset of a reference set is gradual rather than abrupt.

Membership in such sets is not characterized by either/or, but are sets in which membership can be more adequately considered in terms of degrees. A fuzzy set is characterized by a membership function, defined as a real number in the interval [0 , 1]. For example, a membership measure $\mu A(x)=0.8$ suggests that x is a member of set A to a degree 0.8 on a scale where zero is no membership at all; and one is complete membership.

Just as traditional set theory operations can be precisely defined, fuzzy set operations can be precisely defined. Some basic concepts of fuzzy set theory and fuzzy logic are briefly summarized in the following section. A more detailed discussion may be found in [17] [19] [20].

3.2 Fuzzy Terminology

Let U be a collection of objects denoted generically by {u}, which could be discrete or continuous. U is called the universe of discourse and u represents the generic element of U.

Definition: Fuzzy Set: A fuzzy set F [19] in a universe of discourse U is characterized by a membership function μ_F which takes values in the interval [0 , 1] namely, $\mu_F : U \rightarrow [0,1]$. A fuzzy set may be viewed as a generalization of the concept of an ordinary set whose membership function only takes two values $\{0,1\}$. Thus a fuzzy set F in U may be represented as a set of ordered pairs of a generic element u and its grade of membership function:

$F = \{(u, \mu_F(u)) | u \in U\}$

When U is continuous, a fuzzy set F can be written concisely as

$$F = \int_U \frac{\mu_F(x)}{u}$$

When U is discrete, a fuzzy set F is represented as

$$F = \sum_{i=1}^{n} \frac{\mu_F u_i}{u_i}.$$

Definition: Support, Crossover Point, and Fuzzy Singleton: The support of a fuzzy set F [19] is the crisp set of all points u in U such that $\mu_F(u) > 0$. In particular, the element u in U at which $\mu_F = 0.5$, is called the crossover point. A fuzzy set whose support is a single point in U with $\mu_F = 1.0$ is referred to as fuzzy singleton.

3.3 Set Theoretic Operations

Let A and B be two fuzzy sets in U with membership functions μ_A and μ_B, respectively. The set theoretic operations of union, intersection, and complement for fuzzy sets are defined via their membership functions. More specifically, see the following.

Definition: Union: The membership function $\mu_{A \cup B}$ [19] of the union $A \cup B$ is pointwise defined for all $u \in U$ by

$$\mu_{A \cup B}(u) = \max\{\mu_A(u), \mu_B(u)\}.$$

Definition: Intersection: The membership function $\mu_{A \cap B}$ [19] of the intersection $A \cap B$ is pointwise defined for all $u \in U$ by

$$\mu_{A \cap B}(u) = \min\{\mu_A(u), \mu_B(u)\}.$$

Definition: Complement: The membership function $\mu_{\bar{A}}$ [19] of the complement of a fuzzy set A is pointwise defined for all $u \in U$ by

$$\mu_{\bar{A}}(u) = 1 - \mu_A(u).$$

Definition: Cartesian Product: If A1, A2,..........., An are fuzzy sets in U1, U2,..........., Un respectively, the Cartesian product [19] of A1, A2,.........., An is a fuzzy set in the product space U1, U2,......, Un with the membership function

$$\mu_{A_1 x \ldots\ldots\ldots A_n}(u_1, u_2, \ldots\ldots\ldots, u_n) = \min\{\mu_{A_1}(u_1), \ldots\ldots\ldots\ldots\ldots \mu_{A_n}(u_n)\}$$

or

$$\mu_{A_1 x \ldots\ldots\ldots x A_n}(u_1, u_2, \ldots\ldots\ldots, u_n) = \mu_{A_1}(u_1) \cdot \mu_{A_2}(u_2) \ldots\ldots\ldots\ldots\ldots\ldots \mu_{A_n}(u_n)$$

Definition: Fuzzy Relation: A n-ary fuzzy relation [19] is a fuzzy set in U1 x...........x Un and is expressed as

$$R_{U_1 x \ldots\ldots x U_n} = \{((u_1,\ldots\ldots\ldots\ldots,u_n),\mu_R(u_1,\ldots\ldots\ldots\ldots,u_n))|(u_1,\ldots\ldots\ldots\ldots,u_n) \in U_1 x \ldots\ldots\ldots x U_n\}.$$

Definition: Sup-Star Composition: If R and S are fuzzy relations in UxV and VxW, respectively, the composition [19] of R and S is a fuzzy relation denoted by R S and is defined by

$$\mathrm{R\ \ S} = \{[(u,v),\sup_{v}(\mu_R(u,v) * \mu_s(u,w))], u \in U, v \in V, w \in W\},$$

where * could be any operator in the class of triangular norms, namely, minimum, algebraic product, bounded product, or drastic product.

3.4 Linguistic Variables

The use of fuzzy sets provides a basis for a systematic way for the manipulation of vague and imprecise concepts. In particular, we can employ fuzzy sets to represent linguistic variables. A linguistic variable can be regarded as a variable whose values are defined in linguistic terms.

Definition: Linguistic Variables: A linguistic variable [19] is characterized by a quintuple (x, T(x), U, G, M) in which x is the name of variable; T(x) is the term set of x, that is, the set of names of linguistic values of x with each value being a fuzzy number defined on U; G is a syntactic rule for generating the names of values of x; and M is a semantic rule for associating with each value its meaning.

For example, if *speed* is interpreted as a linguistic variable, then its term set T(speed) could be

T(speed) = {very slow, slow, moderate, fast, very fast,............}

where each term in T(speed) is characterized by a fuzzy set in a universe of discourse U = [0 , 100]. We might interpret "very slow" as "a speed below 10 mph," "slow" as "a speed close to 30 mph," "moderate" as "a speed close to 55 mph," and "fast" as "a speed above

70 mph." These terms can be characterized as fuzzy sets whose membership functions are shown in Figure 5. E.g., here the speed of 41 mph has a higher degree of membership in the set "slow" (90%), than in the set "medium" (10%).

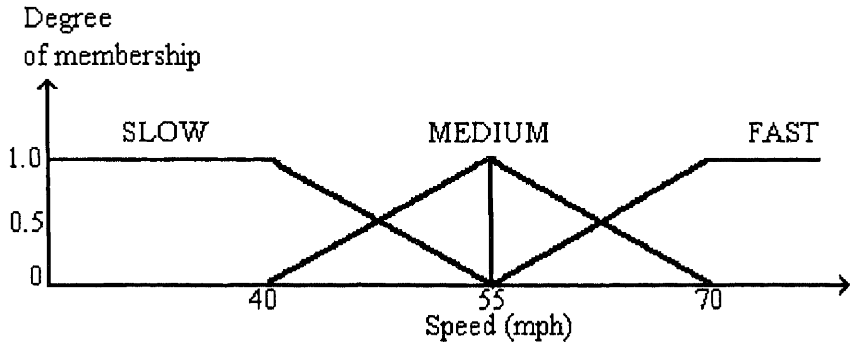


Figure 5 Diagrammatic representation of Fuzzy Speeds.

CHAPTER 4

FUZZY LOGIC CONTROLLER

4.1 General

The main ideas underlying a Fuzzy Logic Controller (FLC) are covered in this chapter. Figure 6 shows the basic configuration of an FLC, which comprises three principal components: a fuzzification function, a knowledge base, and a defuzzification function.

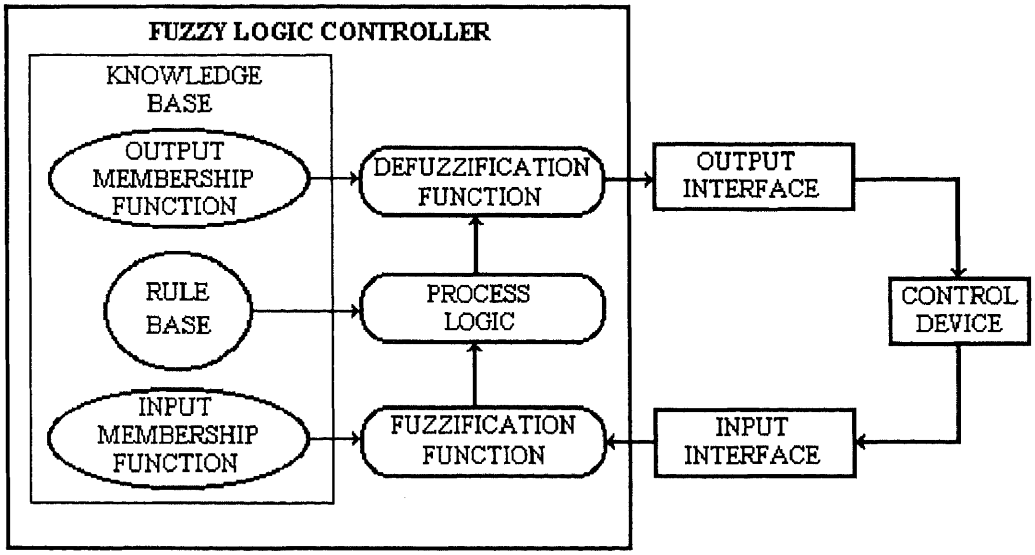


Figure 6 A Typical Fuzzy System.

The **Input interface** maybe of the form of an A/D converter, that measures real-time process variables and digitizes them. It also performs the scaling function to put the process variable into the desired numerical perspective, e.g., in process control applications an analog input module performs the above functions by converting a 4-20mA electrical signal into a numerical value of 0-4095.

The **Fuzzification function** also referred to as a fuzzifier decomposes these real-time input variables to corresponding subsets in the defined universe of discourse. These subsets, and the degree of membership within them are predefined by the rule base programmer in the **Input membership function**. Thus, referring to the input membership function, the fuzzification function describes the numerical input variable with adjectives called linguistic labels that make sense to the designer, e.g., at a particular instance the temperature might be 'cold', 'warm', or 'hot'.

The **Rule base** defines rules that describe the relationships between the results desired and the data available. The **Process logic** executes the rules defined in the rule-base to obtain a separate control output in the domain of the **Output membership function**, for each real-time process variable. It evaluates the degree to which each rule's situation applies. The rule is "active" to the degree that its IF part is true; then in turn determines the degree to which each THEN part applies. Since multiple rules can be active simultaneously, all of the active rules are combined by the **Defuzzification function** to create the final crisp result.

The **Output interface** is of the form of a D/A converter that converts the nonfuzzy control output to be applied to the control device. For example, an analog output module converts the control solution to a 4-20mA signal. This signal may in turn be used to modulate a control valve via a current-to-pressure (I/P) transducer.

4.2 Fuzzification Function

4.2.1 Fuzzification Operator

A fuzzification operator [19] has the effect of transforming crisp input data into fuzzy sets. Symbolically,

$$x_0 = fuzzifier\ (x),$$

where x_0 is a crisp input value from a process; x is a fuzzy set and *fuzzifier* represents a fuzzification operator.

4.2.2 Fuzzy Conditional Statements and Control Rules

The dynamic behavior of a fuzzy system is characterized by a set of linguistic description rules based on expert knowledge. The expert knowledge is usually of the form

IF (a set of conditions are satisfied)

THEN (a set of consequences can be inferred).

Since the antecedents and the consequents of these IF-THEN rules are associated with fuzzy concepts (linguistic terms), they are often called *fuzzy conditional statements*. A *fuzzy control rule* is a fuzzy conditional statement in which the antecedent is a condition in its application domain and the consequent is a control action for the system under control. Fuzzy control rules provide a convenient way of expressing control policy and domain knowledge. Furthermore, several linguistic variables might be involved in the antecedents and the conclusions of these rules. When this is the case, the system is referred to as multi-input-multi-output (MIMO) fuzzy system. For example, in the case of two-input-single-output (MISO) fuzzy systems, fuzzy control rules have the form:

R1 : if *x* is A1 and *y* is B1 then *z* is C1,

R2 : if *x* is A2 and *y* is B2 then *z* is C2,

....,

....,

Rn : if *x* is An and *y* is Bn then *z* is Cn.

where *x*, *y*, and *z* are linguistic variables representing two process state variables and one control variable; Ai, Bi, and Ci are linguistic values of the linguistic variables *x*, *y*, and *z* in the universes of discourse U, V, and W, respectively, with i = 1, 2, ..., n; and an implicit sentence connective also links the rules into a rule set or a rule base.

Rule evaluation, or fuzzy inference, uses a technique called min-max inference to calculate numerical conclusions to linguistic rules based on system input values. The numerical conclusions from this process are called fuzzy outputs.

Although rules look like free-form natural language, they are confined to a limited set of linguistic terms and a strict syntax. The rule language described here is very simple and can be easily implemented in ladder logic for a programmable logic controller (PLC) based control system. Following is an example of the syntax of a rule.

IF IN_X1 IS LABEL_A1 AND IN_X2 IS LABEL_A2 THEN OUT_Y1 IS LABEL_C1 AND OUT_Y2 IS LABEL_C2

Each rule begins with the keyword IF.

Each antecedent is of the form IN_X IS LABEL_A

Where IS is a required keyword,

IN_X is a previously defined system input such as TEMPERATURE,

and LABEL_A is a previously defined label of input X such a WARM.

Any number of antecedents is allowed between the keywords IF and THEN. Each additional antecedent is separated from the previous antecedents by the keyword AND.

The keyword THEN is used to separate antecedents from the consequents.

Each consequent is of the form OUT_Y is LABEL_C

Where IS is a required keyword,

OUT_Y is a previously defined system output such as CONTROLVALVE_POSITION,

and LABEL_C is previously defined label of output Y such as MEDIUM.

Any number of consequents is allowed after the keyword THEN. Each additional consequent is separated from previous consequents by the keyword AND.

More complex fuzzy logic systems could expand on this with a corresponding increase in software overhead to process the more complex rules.

4.2.3 Fuzzification Strategies

Fuzzification is related to the vagueness and imprecision in a natural language. It is a subjective valuation which transforms a measurement into a valuation of a subjective value, and hence it could be defined as a mapping from an observed input space to fuzzy sets in certain input universes of discourse. Fuzzification plays an important role in dealing with uncertain information which might be objective or subjective in nature [19].

In fuzzy control applications, the observed data are usually crisp. Since the data manipulation in an FLC is based on fuzzy set theory, fuzzification is necessary during an earlier stage. The principal ways of dealing with fuzzification are as follows.

1. A fuzzification operator "conceptually" converts a crisp value into a fuzzy singleton [19] within a certain universe of discourse. Basically, a fuzzy singleton is a precise value and hence no fuzziness is introduced by fuzzification in this case. This strategy has been widely used in fuzzy control applications since it is natural and easy to implement. It interprets an input x_0 as a fuzzy set A with the membership function $\mu_A(x)$ equal to zero except at a point x_0, at which $\mu_A(x_0)$ equals one.

2. Observed data are disturbed by random noise. In this case, a fuzzification operator should convert the probabilistic data into fuzzy numbers, i.e., fuzzy (possibilistic) data. In this way, computational efficiency is enhanced since fuzzy numbers are much easier to manipulate than random variables. A variety of fuzzification functions are used, the two most common being the isosceles triangle and the trapezoid functions, as will be more clear in further discussion.

3. In large scale systems and other applications, some observations relating to the behavior of such systems are precise, while others are measurable only in a statistical sense, and some, referred to as "hybrids," require both probabilistic and possibilistic modes of characterization. The strategy of fuzzification in this case is to use the concept of "hybrid numbers", which involves both uncertainty (fuzzy numbers) and randomness (random numbers) and is beyond the scope of this report.

4.3 Knowledge Base

The knowledge base of an FLC is comprised of two components, namely, a data base and a fuzzy control rule base. Some issues relating to the data base are addressed in this section and issues relating to the rule base are discussed in the next section.

The concepts associated with a data base are used to characterize fuzzy control rules and fuzzy data manipulation in an FLC. These concepts are subjectively defined and based on experience and engineering judgment. In this connection, it should be noted that the correct choice of the membership (fuzzification) function of a term set plays an essential role in the success of an application. Some of the important concepts relating to the construction of the data base are discussed below.

4.3.1 Discretization / Normalization of Universes of Discourse

The representation of uncertain information with fuzzy sets brings up the problem of quantifying such information for digital computer processing [19] [20]. In general, the representation depends on the nature of the universe of discourse. A universe of discourse in an FLC is either discrete or continuous. If the universe is continuous, a discrete universe may be formed by a discretization of the continuous universe. Furthermore, a continuous universe may be normalized, as will be seen at a later point in this section.

1. Discretization of a Universe of Discourse:

Discretization of a universe of discourse is frequently referred to as quantization. In effect, quantization discretizes a universe into a certain number of segments (quantization levels). Each segment is labeled as a generic element, and forms a discrete universe. A fuzzy set is then defined by assigning grade of membership values to each generic element of the new discrete universe. A look-up table based on discrete universes, which defines the output of a controller for all possible combinations of the input signals is generated and used as a library function. In the case of an FLC with continuous universes, the number of quantization levels should be large enough to provide an adequate approximation and yet small to save memory storage. The choice of quantization levels has an essential influence on how fine a control can be achieved and is necessary.

For the purpose of discretization, we need a scale mapping, which serves to transform measured variables into values in the discretized universe. The mapping can be uniform (linear), nonuniform (nonlinear), or both. The choice of quantization levels reflects some prior knowledge of the system model. For example, coarse resolution could be used for large errors and fine resolution for small errors. Thus, in a three-input-one-output fuzzy system, we may have control rules of the form:

Ri : if error (e) is *Ai* , sum of errors *(ie)* is *Bi*, and change of error (*de*) is *Ci* then output is *Di*.

A simple instance of an FLC can be represented by

$$K_4[u(k)] = F[K_1e(k), K_2ie(k), K_3de(k)] \ ,$$

where F denotes the fuzzy relation defined by the rule base and *Ki*, i = 1, 2, 3, 4, represents an appropriate scaling mapping. In this relation, we see an analogy to the parameters of a conventional PID controller, in which as a special case F is a linear function of its arguments [21]. In general, due to discretization, the performance of an FLC is less sensitive to small deviations in the values of the process state variables.

2. Normalization of a Universe of Discourse:

The normalization of a universe requires a discretization of the universe of discourse into a finite number of segments, with each segment mapped into a suitable segment of the normalized universe. In this setting a fuzzy set is then defined by assigning an explicit function to its membership function. The normalization of a continuous universe also involves a priori knowledge of the input/output space. The scale mapping can be uniform, nonuniform, or both.

4.3.2 Fuzzy Partition of Input and Output Spaces

A linguistic variable in the antecedent of a fuzzy control rule forms a fuzzy input space with respect to a certain universe of discourse, while that in the consequent of the rule forms a fuzzy output space. In general, a linguistic variable is associated with a term set, with each term in the term set defined on the same universe of discourse. A fuzzy partition, then, determines how many terms should exist in a term set. This is equivalent to finding the number of primary fuzzy sets. The number of primary fuzzy sets determines the granularity of the control obtainable with an FLC [19].

The primary sets (linguistic terms) usually have a meaning, such as NB: negative big; NM: negative medium; NS: negative small; Z: zero; PS: positive small; PM: positive medium; and PB: positive big. A typical example is shown in Figure 7 depicting two fuzzy partitions in the same normalized universe [-1, +1]. Membership functions having the forms of triangle-shaped and trapezoid-shaped functions are used here. Since a normalized universe implies the knowledge of the input/output space via appropriate scale mappings, a well-formed term set can be achieved as shown. If this is not the case, or a nonnormalized universe is used, the terms could be asymmetrical and unevenly distributed in the universe.

The cardinality of a term set in a fuzzy input space determines the maximum number of fuzzy control rules that we can construct. For example, in the case of two-input-one-output fuzzy system, if the cardinalities of T(x) and T(y) are 3 and 7, respectively, the maximum number of rules is 3x7=21. It should be noted that the fuzzy partition of the fuzzy input/output space is not deterministic and has no unique solution. A heuristic cut and trial procedure is usually needed to find the optimal fuzzy partition.

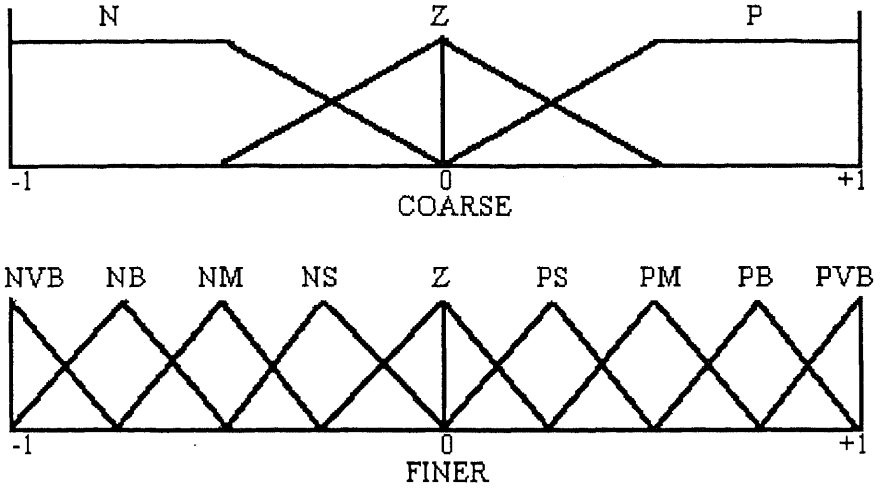


Figure 7 Representation of Fuzzy Partitions.

4.3.3 Completeness

Intuitively, a fuzzy control algorithm should always be able to infer a proper control action for every state of the process. This property is called "completeness" [19]. The completeness of an FLC relates to its data base, rule base, or both.

1. Data Base Strategy:

The data base strategy is concerned with the supports on which primary fuzzy sets are defined. The union of these supports should cover the related universe of discourse in relation to some level set ε . This property of an FLC is called ε -completeness. In general, we choose the level ε at the crossover point, implying that we have a strong belief in the positive sense of the fuzzy control rules which are associated with the FLC. In this sense, a dominant rule always exists and is associated with the degree of belief greater than 0.5. In the extreme case, two dominant rules are activated with equal belief 0.5.

2. Rule Base Strategy:

The rule base strategy has to do with the fuzzy control rules themselves. The property of completeness is incorporated into fuzzy control rules through design experience and engineering knowledge. An additional rule is added whenever a fuzzy condition is not included in the rule base, or whenever the degree of partial match between some inputs and the predefined fuzzy conditions is lower than some level, say 0.5. The former shows that no control action will result. The latter indicates that no dominant rule will be fired.

4.3.4 Membership Functions of a Primary Fuzzy Set

There are two methods used for defining fuzzy sets, depending on whether the universe of discourse is discrete or continuous: 1) numerical and 2) functional.

1. Numerical Definition:

In this case, the grade of membership function of a fuzzy set is represented as a vector of numbers whose dimension depends on the degree of discretization. In this case, the membership function of each primary fuzzy set has the form of

$$\mu_f(u) = \sum_{i=1}^{5} \frac{a_i}{u_i} .$$

2. Functional Definition:

A functional definition expresses the membership function of a fuzzy set in a functional form, typically triangle-shaped function, trapezoid-shaped function, bell-shaped function [1] [17], etc. Such functions are used in FLC because they lead themselves to manipulation through the use of fuzzy arithmetic. The functional definition can readily be adapted to a change in the normalization of a universe. Figure 8 shows an example of a Bell-shaped function expressed as:

$$\mu_f(x) = \exp\{\frac{-(x-\mu_f)^2}{2\sigma_f^2}\}$$

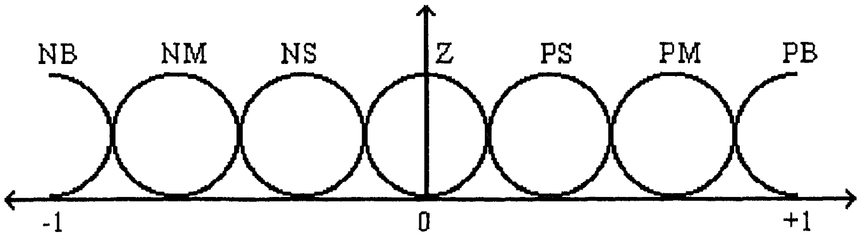


Figure 8 Bell-shaped function.

Note that if the normalized universe is changed, the parameters μ_f, σ_f should be changed accordingly.

Either a numerical definition or functional definition may be used to assign the grades of membership to the primary fuzzy sets. The choice of grades of membership is based on the subjective criteria of the decision. In particular, if the measurable data is disturbed by noise, the membership functions should be sufficiently wide to reduce the sensitivity to noise. This raises the issue of the fuzziness or, the specificity of a membership function, which affects the robustness of an FLC.

4.4 Rule Base

A fuzzy system is characterized by a set of linguistic statements based on expert knowledge. The expert knowledge is usually in the form of "if-then" rules, which are easily implemented by fuzzy conditional statements in fuzzy logic. The collection of fuzzy control rules that are expressed as fuzzy conditional statements forms the rule base or the knowledge repository of the FLC. In this section, we shall examine the following topics related to fuzzy control rules: choice of process state (input) variables and control (output) variables, source, derivation, and types of fuzzy control rules.

4.4.1 Choice of Process State Variables and Control Variables

Fuzzy control rules are more conveniently formulated in linguistic rather than numerical terms. The proper choice of process state variables and control variables is essential to the characterization of the operation of a fuzzy system. Furthermore, the selection of the linguistic variables has a substantial effect on the performance of an FLC. The choice of linguistic variables and their membership function have a strong influence on the linguistic structure of an FLC. Typically, the linguistic variables are the state, state error, state error derivative, state error integral, etc.

4.4.2 Source and Derivation of Fuzzy Control Rules

There are different modes of derivation of fuzzy control rules. These modes are not mutually exclusive, but it is likely that a combination of them would be necessary to construct an effective method for the derivation of fuzzy control rules [19]. Some of the rules are summarized as follows:

1. Expert experience and control engineering knowledge:

Fuzzy control rules have the form of fuzzy conditional statements that relate the state variables in the antecedents and process control variables in the consequents. In this connection, it should be noted that in our daily life most of the information on which our decisions are based is linguistic rather than numerical in nature. Seen in this perspective, fuzzy control rules provide a convenient way to express their domain knowledge.

The formulation of fuzzy control rules can be achieved by means of two heuristic approaches. The most common one involves an introspective verbalization of human expertise. A typical approach is interrogation of experienced experts or operators using a carefully organized questionnaire. This way we can form a prototype of fuzzy control rules for a practical application domain. Obviously, for optimized performance, the use of cut and trial procedures is usually a necessity.

2. Based on Operator's Control Actions:

In many industrial man-machine control systems, the input-output relations are not known with sufficient precision to make it possible to employ classical control theory for modeling and simulation. And yet skilled human operators can control such systems quite successfully without having any quantitative models in mind. In effect, a human operator employs - consciously or subconsciously - a set of fuzzy if-then rules to control the process. In practice, such rules can be deduced from the observation of human controller's actions in terms of the input-output operating data.

3. Based on the Fuzzy Model of a Process:

In the linguistic approach, the linguistic description of the dynamic characteristics of a controlled process may be viewed as a fuzzy model of the process. Based on the fuzzy model, we can generate a set of fuzzy control rules for attaining optimal performance of a dynamic system. The set of fuzzy control rules forms the rule base of a fuzzy logic controller. This approach is said to be somewhat complicated, but yields better performance and reliability, and provides a more tractable structure for dealing theoretically with the fuzzy logic controller.

4.4.3 Types of Fuzzy Control Rules

Depending on their nature, two types of fuzzy control rules are currently in use in the design of a FLC. They are, state evaluation fuzzy control rules and object evaluation fuzzy control rules [19].

1. State Evaluation Fuzzy Control Rules:

Most FLC's have state evaluation fuzzy control rules which, in the case of MISO systems, are characterized as a collection of rules of the form

R1 : if x is A1 and y is B1 then z is C1,

R2 : if x is A2 and y is B2 then z is C2,

....,

....,

Rn : if x is An and y is Bn then z is Cn.

where x, y, and z are linguistic variables representing two process state variables and one control variable; Ai, Bi, and Ci are linguistic values of the linguistic variables $x,....y$, and z in the universes of discourse U,......V, and W, respectively, with i = 1, 2, ..., n.

In a more general version, the consequent is represented as a function of the process state variables $x,....y$, i.e.,

Ri : if x is Ai and y is Bi then $z = fi\ (x,....y)$.

Fuzzy control rules of this type, which are referred to as "state evaluation fuzzy control rules," evaluate the process state, e.g., state, state error, or state integral at time t and compute a fuzzy control action at time t as a function of (x,.......,y) and the control rules in the rule set.

2. Object Evaluation Fuzzy Control Rules:

Algorithms which predict present and future control actions and evaluate control objectives are called "object evaluation fuzzy control," or "predictive fuzzy control." The rules in question, which are derived from skilled operator's experience, are referred to as "object evaluation fuzzy control rules." A typical rule is described as

Ri : if (u is *Ci* --(x is *Ai* and y is *Bi*)) then u is *Ci*.

A control command is inferred from an objective evaluation of a fuzzy control result that satisfies the desired states and objectives. A control command u takes a crisp set as a value, and x, y are performance indices for the evaluation of the i th rule, taking values such as "good" or "bad". The most likely control rule is selected through predicting the results (x,y) corresponding to every control command *Ci*.

In linguistic terms, the rule is interpreted as: "if the performance index x is *Ai* and index y is *Bi*, when a control command u is chosen to be *Ci*, then this rule is selected and the control command *Ci* is taken to be the output of the controller.

4.5 Defuzzification Function

4.5.1 Defuzzification Operator

Basically, defuzzification is a mapping from a space of fuzzy control actions defined over an output universe of discourse into a space of nonfuzzy (crisp) control actions. It is employed because in many practical applications a crisp control action is required. Consequently, one must defuzzify the fuzzy control action (output) inferred from the fuzzy control algorithm, namely

$$z_0 = defuzzifier\ (z),$$

where z_0 is the nonfuzzy control output and *defuzzifier* is the defuzzification operator.

4.5.2 Defuzzification Strategies

A defuzzification strategy is aimed at producing a nonfuzzy control action that best represents the possibility distribution of an inferred fuzzy control action. Unfortunately, there is no systematic procedure for choosing a defuzzification strategy.

At present, the commonly used strategies may be described as the max criterion, the mean of maximum, and the center of area.

1. The max criterion method:

The max criterion produces the point at which the possibility distribution of the control action reaches a maximum value.

2. The mean of maximum method (MOM):

The MOM strategy generates a control action which represents the mean value of all local control actions whose membership functions reach the maximum.

3. The center of area method (COA):

The widely used COA strategy or the centroid method of defuzzification, generates the center of gravity of the possibility distribution of a control action. In the case of a discrete universe, this method yields

$$Z_0 = \frac{\sum_{j=1}^{n} \mu_z(w_j) \cdot w_j}{\sum_{j=1}^{n} \mu_z(w_j)}$$

where n is the number of quantization levels of the output.

CHAPTER 5

DESIGN METHODOLOGY

5.1 General

Achieving superior plant performance requires meeting two types of economic objectives, one called strategic; the other, operating. The strategic objective is primarily concerned with external rather than internal problems of a company, and specifically with selection of a company's product mix and target markets. In most companies, operating objectives inevitably absorb the bulk of the company's energy and attention. Everyone in the organization is concerned with the myriad of recurring operating problems.

The design of a control system and its level of use determine how well a unit in a process plant is run. Therefore, this directly affects the profitability of that unit. The major problems concern how to apply effective optimizing control for operating conditions. The key decisions involve: inventory control for mass and energy in order to maintain steady-state operation; establishing optimal setpoints for efficient operating conditions and consistently maintaining them.

Optimizing control is difficult to accomplish because disturbances continually upset the process. Disturbances are variables that adversely affect the process operations and over which there is no direct control. Values of such variables are often determined by some known or unknown factors external to the process boundary. Control action is required to overcome adverse effects. Uncontrolled disturbances may be associated with raw materials, ambient conditions, changes in load conditions, fuel and energy costs, and other economic factors. It is very difficult to identify all the disturbance variables that affect the process. Numerous factors or conditions are constantly changing. Some are important, some are not. The best procedure, when practical, is to include all major process disturbances and ignore those involved with second-order effects. This approximation will render the minor disturbances indistinguishable from error.

5.1.1 Multiple variables and the human factor

My observations as an Electrical Engineer working in the field of Facilities Automation via Systems integration has led me to believe that the area of control system is the most often neglected for improving profitability.

The disturbances discussed above have prevented several processes from being automated because of lack of precise control techniques. Such processes, prevalent in a Chemical or a Pharmaceutical facility need rigid operating conditions, invisible to the multiple variables or factors affecting it. As an example, chemicals/compounds have to be maintained at precise levels of concentration, pH, temperature, conductivity, etc. before being used or mixed with other chemicals. The conductivity of a substance is interrelated to its temperature or concentration and these variables have to be taken into consideration before adding any reagents to maintain the conductivity.

Many such processes are handled manually by skilled human operators. It is a common consensus that the human factor is prone to inconsistencies in product quality and a wasted chemical batch could cause losses worth several thousand dollars.

5.1.2 Software validation

The Food and Drug Administration (FDA) requires the validation of computer systems [15] used in the manufacture of pharmaceuticals. There are written procedures for production and process control, designed to assure that drug products have the identity, strength, quality and purity they purport or are represented to possess. These procedures and designs have to be exercised over computer systems (or PLCs or distributed control systems) that are used in the manufacture of drugs.

All of the software installed and operating within a distributed control system or other computer based automation system has to be documented with the proper paper trail and qualified to assure that it is free of any defects and consistently operates within the set tolerances under production conditions. Validation of the control system includes validating the following sub components [15]:

- Operating system and file managers
- Network information management software
- Database management software
- Software modules for batch and continuous control
- Graphical operator workstation software
- Data collection and report generation software
- Statistical process control packages.

As a vendor, I have to play a significant role in assuring that the system designed is of high quality at the time of commissioning and remains so throughout its life cycle. I was concerned about oscillations and responses to upsets or setpoint changes (Chapter 1) in the PID loops I commissioned. Some typical disturbances that can upset a temperature loop are [8] [14]:

- Opening a door in a climate chamber causes a sudden drastic change in the chamber's temperature.
- Introducing moisture into a temperature vessel speeds up the heating process by changing its enthalpy factor.
- Sudden increase in the flow around a heater element means that more energy has to be applied to the heater to maintain the setpoint.
- Sudden decrease in the flow around a heater element means that less energy has to be applied to the heater to maintain the setpoint.

Most of the loops (temperature and humidity) being related to HVAC (Heating Ventilation and Air Conditioning) didn't need any critical time considerations as long as the variable parameter stabilized within an acceptable amount of time. But, with the exposure to Pharmaceutical industries, governed by strict FDA regulations, the approach to tackle some of the PID loops had to be changed. I believed that the philosophy of fuzzy logic had the potential to overcome the problems mentioned earlier.

A fairly complicated system was chosen for experimentation and a design methodology was formulated as explained in the sections that follow. Chapter 6 explains in detail the fuzzy logic controller designed for a Heat Exchanger used in the process of Water for Injection (WFI).

5.2 Design Methodology

It is important to have a good design methodology for fuzzy systems because they are new to most designers, who therefore have no hoard of experience on which to rely for guidance [10].

Fuzzy models, be they employed in process control or information technology, tend to follow somewhat similar development cycle. The methodology explained in the following section attempts to formalize and structure a procedure in which the conceptual design is done on paper, and the later steps are an iterative cycle of modeling and simulation, carried out on a computer using software development and simulation tools, and continued until the model behaves as desired.

The paper portion of the process is critical: understanding the mechanics behind a system's behavior, and identifying the system dynamics in terms of the conventional input-process-output model is an absolutely essential part of fuzzy system design [10].

5.2.1 System Design Steps

Step 1: Define the system's functional requirements:

This is the first step in the development phase and locates the need of fuzzy control in place of conventional control. Systems with unusual behavior or susceptible to a multitude of disturbances are studied with fuzzy control philosophy in mind. The disadvantages of the conventional PID control are studied and the possible advantages that could results via fuzzy control are listed for comparison purposes. If the advantages of fuzzy control outweigh the conventional system, we proceed to the next step of modeling the system.

It is a possible that the complexity of fuzzy control is not cost-effective for some applications. As an example, the HVAC system of an office building is merely an on-off control but the same is not true in the case of a hospital intensive-care unit where, relatively a greater degree of precision is necessary.

Step 2: Define the system parameters:

Here, we define the system in terms of an input-process-output relationship. The variable that needs to be controlled is pointed out and its relationship with variables affecting it and loads or disturbances is tabulated.

Even though a mathematical model of the system is not available, the designer should have an in-depth knowledge of the process and its effectiveness in the systems overall performance [10]. The designer should define what process variables (PV) will be read by the system and what are the calculations to be performed on them to generate the control variable (CV). The operating ranges of the process variables have also to be estimated.

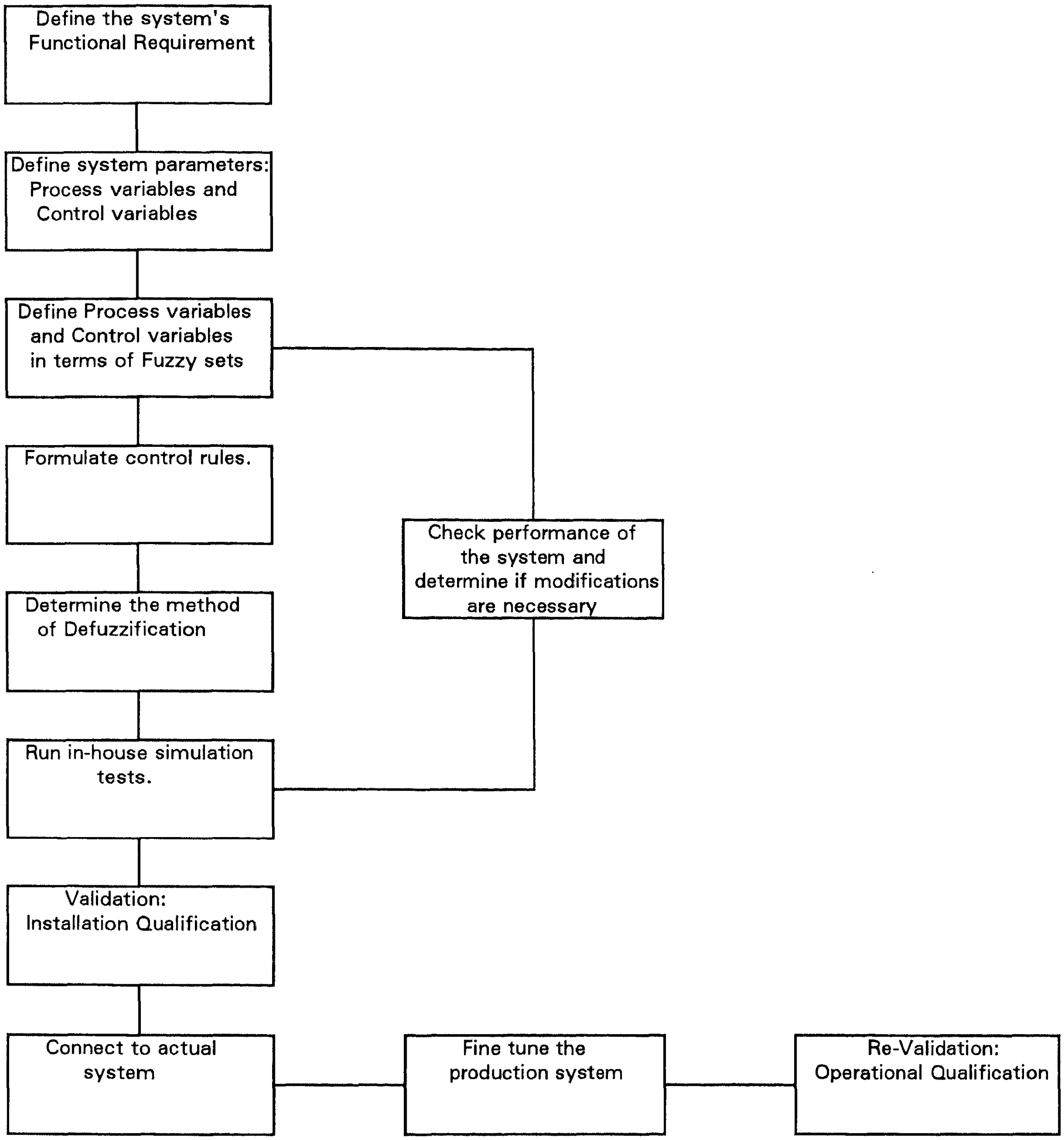

Define the system's Functional Requirement
Define system parameters: Process variables and Control variables
Define Process variables and Control variables in terms of Fuzzy sets
Formulate control rules.
Check performance of the system and determine if modifications are necessary
Determine the method of Defuzzification
Run in-house simulation tests.
Validation: Installation Qualification
Connect to actual system
Fine tune the production system
Re-Validation: Operational Qualification

Figure 9 FLC Design Steps.

Step 3: Define the system parameters in terms of fuzzy sets:

In this step each process variable and control variable in the system is decomposed into a set of fuzzy regions. These regions are given unique names, called labels, within the domain of the variable. These names express the relative quality of the variable instead of the precise quantitative factor. Finally, a fuzzy set that semantically represents the concept associated with the label is created.

Some rules of thumb help in defining fuzzy sets [10]. First, the number of labels associated with a variable should generally be an odd number between five and nine. Second, each label should overlap somewhat with its neighbors. This overlap, in fact, is what gives a fuzzy controller its smooth, stable surface. The overlap should be between 10 and 50 percent of the neighboring space, and the sum of the vertical points of the overlap should always be less than or equal to one.

Finally, the density of the fuzzy sets should be highest around the control setpoint of the system and should thin out as the distance from that point increases.

Step 4: Formulate control rules:

This step involves writing the rules that tie the input variables to the output variables. These rules are expressed in an English-like simple language with a syntax like:

-- *If <fuzzy proposition>, then <fuzzy proposition>,*

where the fuzzy propositions are of the form, "x is Y" or "x is not Y," x being a scalar variable and Y being a fuzzy set associated with that variable.

Such a group of rules forms a fuzzy associative memory. When a set of input values are read, each of the rules that has any truth in its premise will be executed. Since these rules are declarative rather than procedural, their order in the knowledge repository is unimportant.

Generally, the number of rules a system requires is simply related to the number of process variables. In some cases, it is possible to use fewer rules, but there are dangers in doing so. The rules represent knowledge, so if any are deleted or omitted, knowledge is removed from the system--knowledge that may become important if the system is later modified.

Step 5: Select a method of defuzzification:

This is the final part of creating the basic fuzzy model. There are several ways to convert an output fuzzy set into a crisp solution variable, but the two most common are the composite maximum and the composite moment, or centroid.

The centroid method takes the center of gravity of the final fuzzy space and produces a result that is sensitive to all the rules, in particular, the results tend to move smoothly across the control surface. By and large, process control applications use the centroid method [19].

The composite maximum, on the other hand, produces a result that is sensitive to the truth produced by the single rule that has the highest predicate truth. Information-based applications like risk evaluation and terrain analysis use the composite maximum method. E.g. evaluating an individual for automobile insurance - the average of his driving violations, criminal history, age, sex, type of car, residential locality, amount of driving etc. is not taken into consideration, but even a single "negative" aspect is given the extreme negative weight.

Step 6: Run in-house simulation tests:

Simulations are carried out in-house to compare results with the conventional system. The fuzzy model, in the least, should provide a response comparable to the existing PID control, if not better. When the results are not as desired, changes are made either to the fuzzy set descriptions or to the mappings encoded in the rules. This process of simulation and modifications continue till acceptable response is obtained.

Step 7: Validation - Installation Qualification:

The Installation Qualification [15] provides a concise description of the control system upgrade. A verification is made that all instrumentation, designs, drawings, procedures and simulation test results are correctly identified, documented, and within specified set of tolerances. This step basically qualifies the system to be installed and is performed in-house.

Step 8: Connect to actual system:

Connect the control system upgrade to the production system on-site and observe its behavior under production conditions. Note down critical parameters like dead time, time for the process variable to reach the setpoint and overshoot.

Step 9: Fine tune the production system:

Tune the system by making appropriate modifications in the fuzzifier, defuzzifier or the rule-base strategies.

Step 10: Re-validation - Operational Qualification:

The Operational Qualification [15] is performed on-site to verify that the system consistently operates within a specified set of tolerances under test production conditions. This is usually done after tuning the system. The objective of this step is to evaluate the sequence controlled by the system. This objective will be confirmed by conducting a series of critical parameter verifications to include sequence of operations.

CHAPTER 6

FUZZY LOGIC CONTROLLER FOR A HEAT EXCHANGER

6.1 The Heat Exchanger

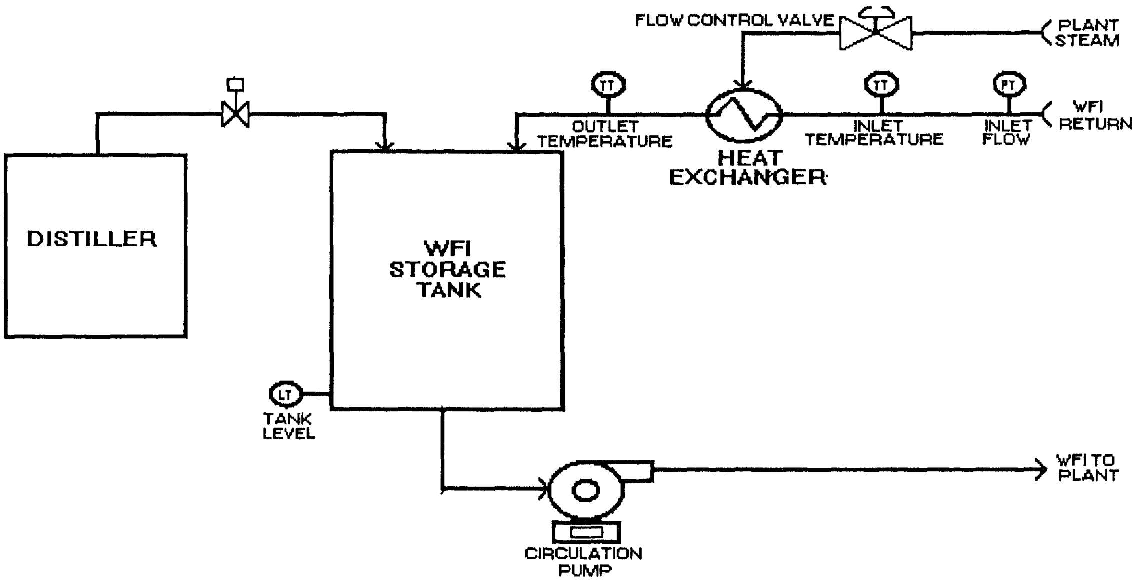


Figure 10 Heat-exchanger in a WFI System.

The Heat-exchanger for a Water for Injection (WFI) system was chosen to study the response of a fuzzy logic controller. The WFI system is shown in the Figure 10. It consists of a Distiller, a WFI storage and distribution tank, the circulation pump, and the Heat exchanger.

The Distiller generates distilled water and operates to keep the WFI storage tank within acceptable production limits. The circulation pump operates on a continuous basis and circulates WFI throughout the plant. Distilled water is used throughout the plant for several purposes, e.g., sterilize equipment used in clean rooms, sterilize piping carrying fluids, clean and sterilize storage vessels, diluting solutions used in the preparation of different buffers. The unused distilled water is returned to the storage tank via the Heat-exchanger. The primary function of the Heat-exchanger is to maintain the temperature of the return WFI at the setpoint. This is done by regulating the flow of plant steam into the Heat-exchanger via a steam flow control valve.

Thus, the Heat-exchanger controls the process variable i.e. temperature via the control variable i.e. steam flow control valve position. The entire heat exchange process is discussed in detail in Chapter 1.

Refering to Figure 6, the implementation of a FLC in a PLC is illustrated in Figure 11.

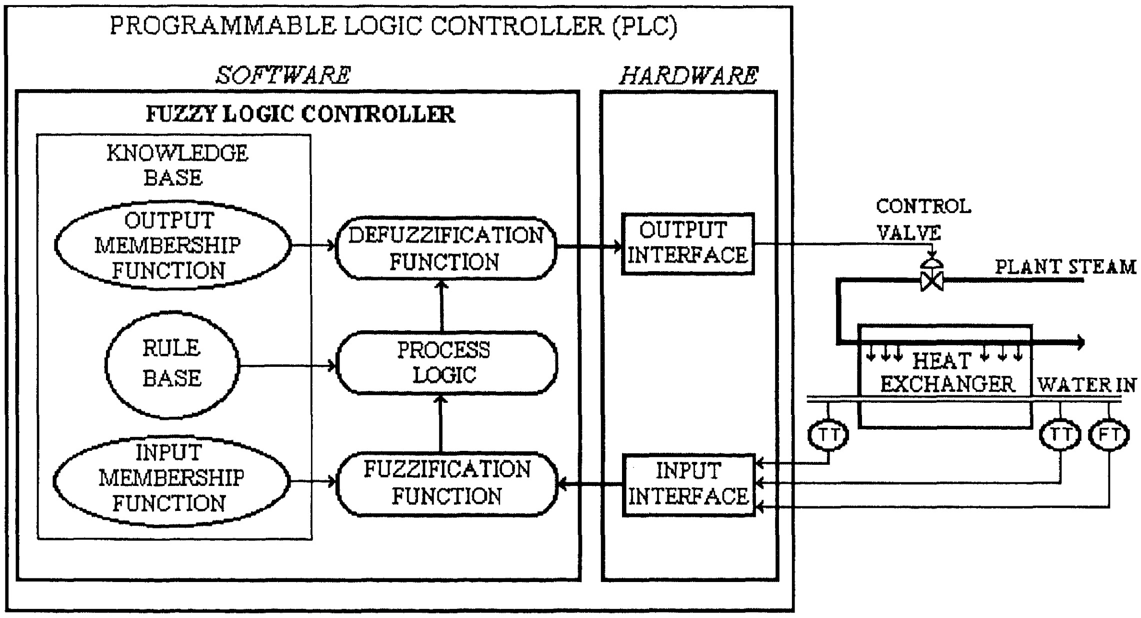


Figure 11 Block diagram of Heat-exchange process.

The hardware specifications of the system are tabulated in Table 1.

Table 1 System Hardware Specifications.

Heat exchanger	Yula Corp. Duty: 90000 MBTU/HR Max. temp: 80 degF Max. temp: 100 psig
WFI storage tank	Capacity : 5000 gallons. Material : 316L Stainless steel. Dimensions : 126" inner diameter, 103" height. Insulation : 3" cerawool.
Steam flow control valve	Worcester carbon steel ball valve - 1/2" 446 PMSE, Worcester 1039SN spring return actuator, Bettis P-2000 electro-pneumatic positioner.
Temperature elements	Burns Engineering RTD with thermowell - XWSPOC21-9-3AW
Flow element	EG&G Sanitary Turbine Flow Meter EG&G Flow Transmitter
Circulation pump	Triclover centrifugal pump - 7.5hp, 3-phase, 230-460vac.

6.2 Design and Implementation of FLC

Step 1: System functional requirements:

The functional requirement of the system was to control the temperature of the return WFI and minimize the effects of process upsets, e.g., overshoots, undershoots due to setpoint and load changes.

The Heat-exchanger is an integral part of the WFI system. The problems associated with the conventional PID control were studied. It was decided to implement the principles of fuzzy logic control in the software of the existing control system and compare its performance to PID control.

Step 2: Define system parameters:

The principle of the Heat-exchange process is based on the control of one process variable namely, water temperature. The PID controller reads the water temperature at the outlet of the Heat-exchanger, applies a mathematical algorithm, and produces a control variable that manipulates the position of the steam flow control valve, which in effect controls the steam flow into the Heat-exchanger.

It was observed that the temperature sensed at the outlet of the Heat-exchanger was utilized to apply corrective action to the return water that was available at a later point in time. Also, water returned from different locations in the plant varied in heat content and hence the temperature, depending on the prevailing ambient conditions along the path it traversed. Hence, it was decided to add another temperature sensor at the inlet of the Heat-exchanger. But using this variable by itself would not suffice the control action as there would be no feedback to the system about the results of the control action. Therefore, the two parameters: inlet temperature and outlet temperature were decided to be used in conjunction to each other in the proposed system.

It was also observed that load changes seriously affected the consistency of the PID controller. Varying usage of the distilled water at different locations in the plant caused varying return water flow rates, and this variable could not be taken into consideration in the existing PID controller. The inclusion of return water flow-meter added one more variable to the new system's design.

The temperatures at the input and output of the Heat-exchanger are measured via RTD's (Resistance Temperature Detector) into an RTD input module housed in the main processor chassis. The flow of the return water is measured via a flow element coupled with a transmitter in terms of a 4 - 20 mA signal into an Analog input module in the main processor chassis.

The control action is via a 4 - 20 mA signal to an I/P (current to pressure) transducer that positions the steam flow control valve and controls the amount of steam into the Heat-exchanger and thus the temperature of the water.

Step 3: Define the system parameters in terms of fuzzy sets:

In order to build a fuzzy controller that represents the relationship between the inputs and the output, each variable must first be decomposed into a set of control regions called fuzzy sets. The input variable is temperature and the output variable is the control output applied to the steam flow control valve.

The input variable temperature is used indirectly in the form of "Error":

where, Error = Setpoint - Actual Process Variable.

The next step was to decide which fuzzification function should be used. Temperature loops are well known to be very slow acting loops in comparison to pressure loops or pH loops and hence the trapezoid function was chosen over the isosceles triangle function. The variable "Error" is decomposed into trapezoidal fuzzy regions as shown in Figure 12. The center of the trapezium corresponds to the mean value of the data set. These trapezoidal regions were given unique names, called labels, within the domain of the variable.

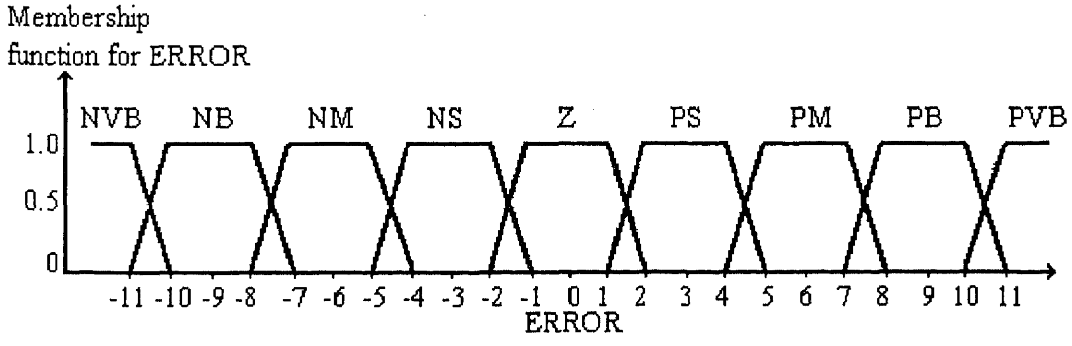


Figure 12 Variable "ERROR" decomposed into labeled fuzzy regions.

The number of labels associated with a variable corresponds to the accuracy of the control action needed. Tighter the control action needed, more the number of labels should be used. Also, each label should overlap somewhat with its neighbors. This overlap, in fact, is what gives a fuzzy controller its smooth, stable control surface. The labels used to describe the fuzzy regions are as follows:

PVB = Error is positive and very big.
PB = Error is positive and big.
PM = Error is positive and medium.
PS = Error is positive and small.
Z = Error is zero.
NS = Error is negative and small.
NM = Error is negative and medium.
NB = Error is negative and big.
NVB = Error is negative and very big.

The labels are self explanatory, and it is clear that more positive the "Error" - further away is the actual temperature from the setpoint - more the steam needed by the Heat-exchanger.

The input variable temperature is also used indirectly in the form of difference in error or rate of change of error "dError":

where, dError = Previous error - Current error.

Similar labels PVB, PB, ...NVB are used to describe the rate of change of "Error", and this is analogous to the derivative action in a PID controller.

The variable "dError" is now decomposed into fuzzy regions as shown in the Figure 13.

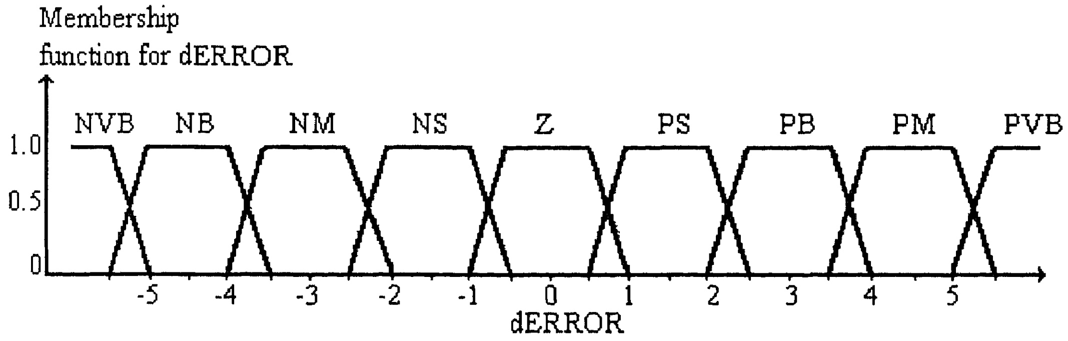


Figure 13 Variable "dERROR" is decomposed into a set of labeled fuzzy regions.

The output variable "Control action" is also decomposed into fuzzy regions as shown in the Figure 14.

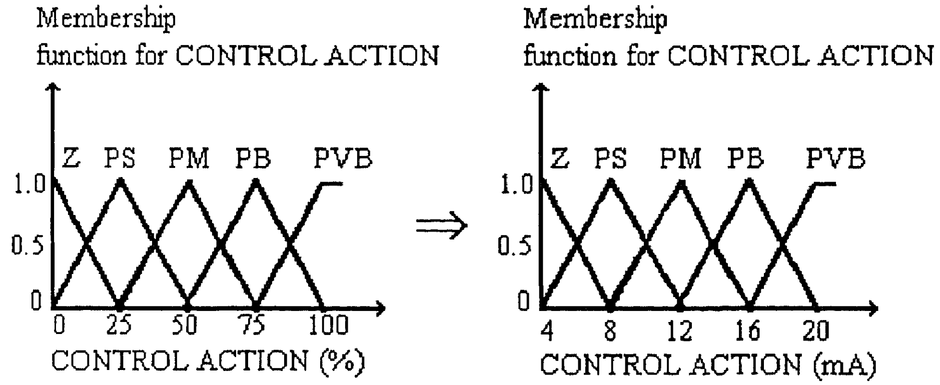


Figure 14 The output variable "CONTROL ACTION" decomposed into labeled fuzzy regions.

Step 4: Formulate control rules:

Here, we make a list of the intuitive rules that govern the heat-exchanger application. Unlike the conventional control method, which would yield a mathematical model, the rules are developed in language of IF-THEN statements. In this model I have nine labels, representing four ranges in the positive direction, four in the negative direction, and a zero, as shown in the figures above.

The rules that describe the "Control action" to be taken on each combination of the control variables "Error" and "dError" are explained in Table 2.

Here,

"Error" = Desired setpoint - Actual temperature at outlet of Heat-exchanger

= SP - PV(out)

and

"dError" = Previous error - Current error.

Table 2 Rule-Base I.

Rule	Antecedent Block	Consequent Block
Rule 1.1	IF Error is PVB and dError is NVB	THEN Output to valve is PVB
Rule 1.2	IF Error is PVB and dError is NB	THEN Output to valve is PVB
Rule 1.3	IF Error is PVB and dError is NM	THEN Output to valve is PVB
Rule 1.4	IF Error is PVB and dError is NS	THEN Output to valve is PVB
Rule 1.5	IF Error is PVB and dError is Z	THEN Output to valve is PVB
Rule 1.6	IF Error is PVB and dError is PS	THEN Output to valve is PB
Rule 1.7	IF Error is PVB and dError is PM	THEN Output to valve is PB
Rule 1.8	IF Error is PVB and dError is PB	THEN Output to valve is PM
Rule 1.9	IF Error is PVB and dError is PVB	THEN Output to valve is PM
....	....	....
Rule 2.1	IF Error is PB and dError is NVB	THEN Output to valve is PB
Rule 2.2	IF Error is PB and dError is NB	THEN Output to valve is PB
Rule 2.3	IF Error is PB and dError is NM	THEN Output to valve is PB
Rule 2.4	IF Error is PB and dError is NS	THEN Output to valve is PB
Rule 2.5	IF Error is PB and dError is Z	THEN Output to valve is PB
Rule 2.6	IF Error is PB and dError is PS	THEN Output to valve is PB

Table 2 (continued)

	IF Error is PB and dError is PM	THEN Output to valve is PM
Rule 2.8	IF Error is PB and dError is PB	THEN Output to valve is PM
Rule 2.9	IF Error is PB and dError is PVB	THEN Output to valve is PS
....	....	....
....	....	....
....	....	....
Rule 9.9	IF Error is NVB and dError is PVB	THEN Output to valve is Z

As can be seen from the Table 2, we have 81 rules covering all possible states of the defined process variables.

The rule numbers are of the form "Rule r.c", on purpose, where "r" and "c" are row and column numbers of the rules when tabulated in a matrix form called Rule-Base.

The Rule-Base I is shown in Figure 15.

d(ERROR) = Previous - Current

ERROR = SP - PV(out)

	NVB	NB	NM	NS	Z	PS	PM	PB	PVB
PVB	PVB	PVB	PVB	PVB	PVB	PB	PB	PM	PM
PB	PB	PB	PB	PB	PB	PB	PM	PM	PS
PM	PB	PB	PB	PB	PB	PM	PM	PS	PS
PS	PM	PM	PM	PM	PM	PM	PS	PS	Z
Z	PM	PM	PS	PS	PS	Z	Z	Z	Z
NS	PS	PS	Z	Z	Z	Z	Z	Z	Z
NM	PS	Z	Z	Z	Z	Z	Z	Z	Z
NB	Z	Z	Z	Z	Z	Z	Z	Z	Z
NVB	Z	Z	Z	Z	Z	Z	Z	Z	Z

Figure 15 Rule-Base I.

Another set of rules was developed to accommodate for the temperature variable at the inlet of the Heat-exchanger. Thus, we have

ERROR(out) = SP - PV(out), and

ERROR(in) = SP - PV(in)

where subscripts "out" and "in" are used to distinguish between the temperature measured at the outlet and inlet of the Heat-exchanger respectively.

The rules developed using PV(out) and PV(in) are summarized in the matrix shown in Figure 16 as Rule-Base II.

ERROR = SP - PV(in)

		PVB	PB	PM	PS	Z	NS	NM	NB	NVB
	PVB	PVB	PB	PM	PS	Z	Z	Z	Z	Z
	PB	PB	PB	PM	PM	Z	Z	Z	Z	Z
	PM	PB	PM	PM	PS	Z	Z	Z	Z	Z
	PS	PM	PM	PS	PS	Z	Z	Z	Z	Z
ERROR =	Z	PM	PS	PS	Z	Z	Z	Z	Z	Z
SP - PV(out)	NS	PM	PS	PS	Z	Z	Z	Z	Z	Z
	NM	PM	PS	PS	Z	Z	Z	Z	Z	Z
	NB	PM	PS	PS	Z	Z	Z	Z	Z	Z
	NVB	PM	PS	PS	Z	Z	Z	Z	Z	Z

Figure 16 Rule Base II.

Step 5: Select a method of defuzzification:

The "center of gravity" or the "centroid" method of defuzzification was chosen, since it weighs the effect of each input variable towards the calculation of the output [19].

Input fuzzy sets and rules are converted into an output fuzzy set, and then into a crisp output for controlling the steam control valve. A fuzzy model is, in a sense, a parallel processor. All the rules that have any truth in their premises will fire and contribute to the output fuzzy set -- the one that will represent the control variable controlling the steam valve. The process of defuzzification is explained below:

Suppose that at a given time, the system sensors determine the "Error" and "dError" to be as shown in Figure 17. As shown in the figure, the "Error" falls into a single region of the Error variable, namely PB. But "dError" has degrees of membership in two regions of its fuzzy set, PB and PVB. This combination causes rules #2.8 and #2.9 to fire.

The two rules have somehow to be combined to form a single system output. The following three-step procedure [10] shows how:

- For each premise expression connected by an AND, take the minimum of the truth of the expressions as the truth level of the premise.

- Truncate the output fuzzy set being built at the truth level of the premise.

- Copy the newly modified fuzzy set into the output variable's fuzzy set. If that region is not empty, combine it with the current contents by taking the maximum of the new fuzzy region and the currently existing fuzzy region at each point in the domain (along the horizontal axis). In other words, if the region is not empty, then OR the outputs together.

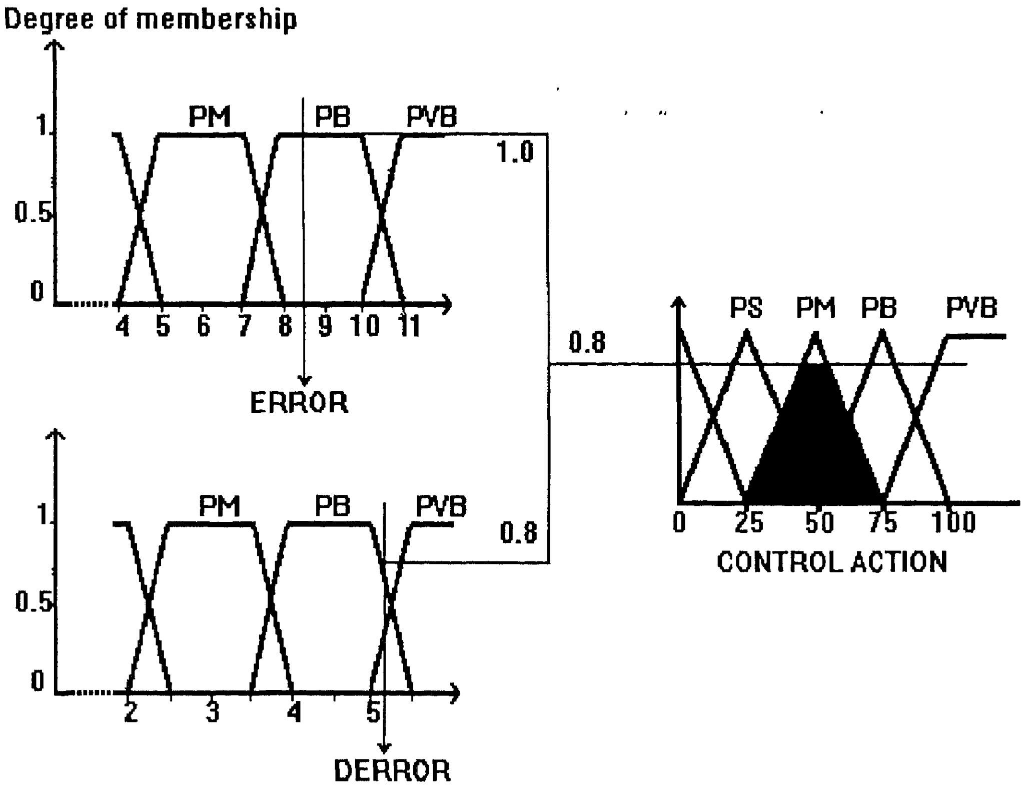


Figure 17 Defuzzification.

For the steam control valve, rule #2.8 (If Error is PB and dError is PB, then control variable is PM), it may be seen that Error has a 1.0 degree of membership in PB, and dError has a 0.8 degree of membership in PB. In accordance with the first step of the procedure, therefore, the smaller of the two figures, 0.8 is taken as the truth level of the premise. Then, in accordance with the second step, the level of the output rule #2.8, namely PM is truncated at that level, and copied to the output variable fuzzy set.

When rule #2.9 fires, the PS fuzzy set is also truncated at the truth of its premise (the smaller of 1.0 and 0.25) and then copied into the output control action region. But, since the region is not empty, this modified fuzzy set is combined with the PM fuzzy set by taking the maximum of their respective membership grades at each point along the horizontal axis.

From this combined region, the defuzzification technique of "center of gravity" is applied to get the combined region, a value of 46.25 is produced as shown in Figure 18. This value is used to adjust the position of the steam control valve. After that, the temperature sensor will make new measurements, starting the cycle over again.

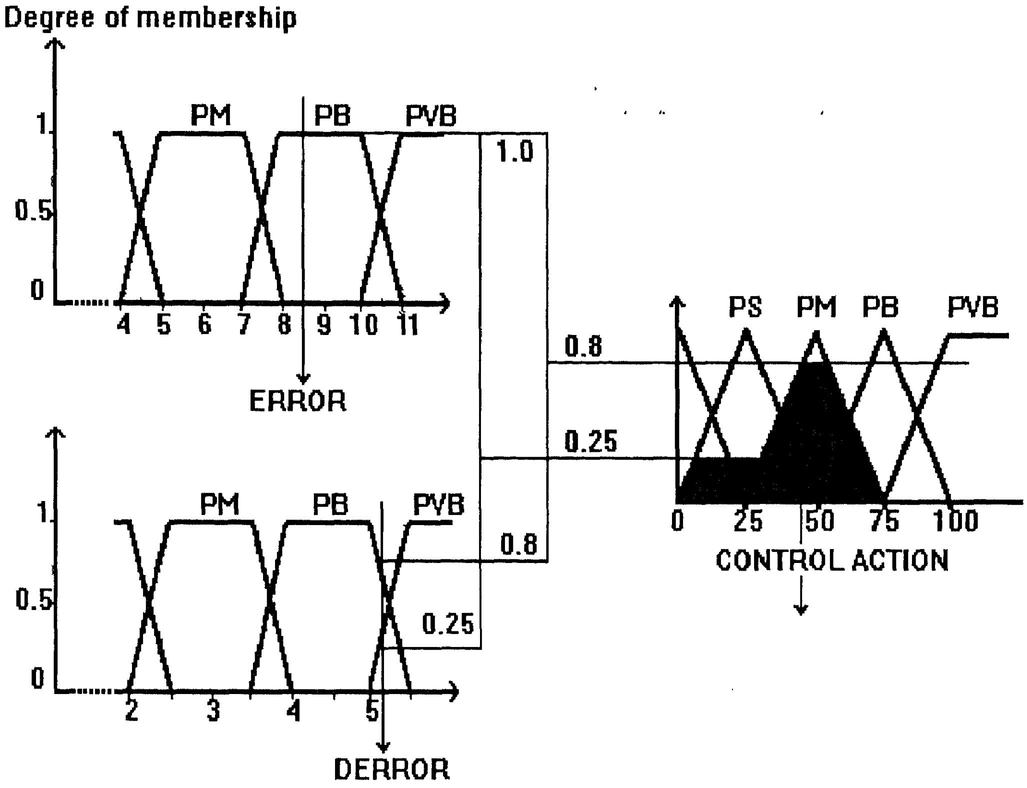


Figure 18 Defuzzification.

Familiarity with ladder logic programming techniques and Allen-Bradley's 6200 programming software instruction set is necessary to write software. A brief outline of the above defuzzification process using the AB-6200 standard instruction set is shown in Figure 19.

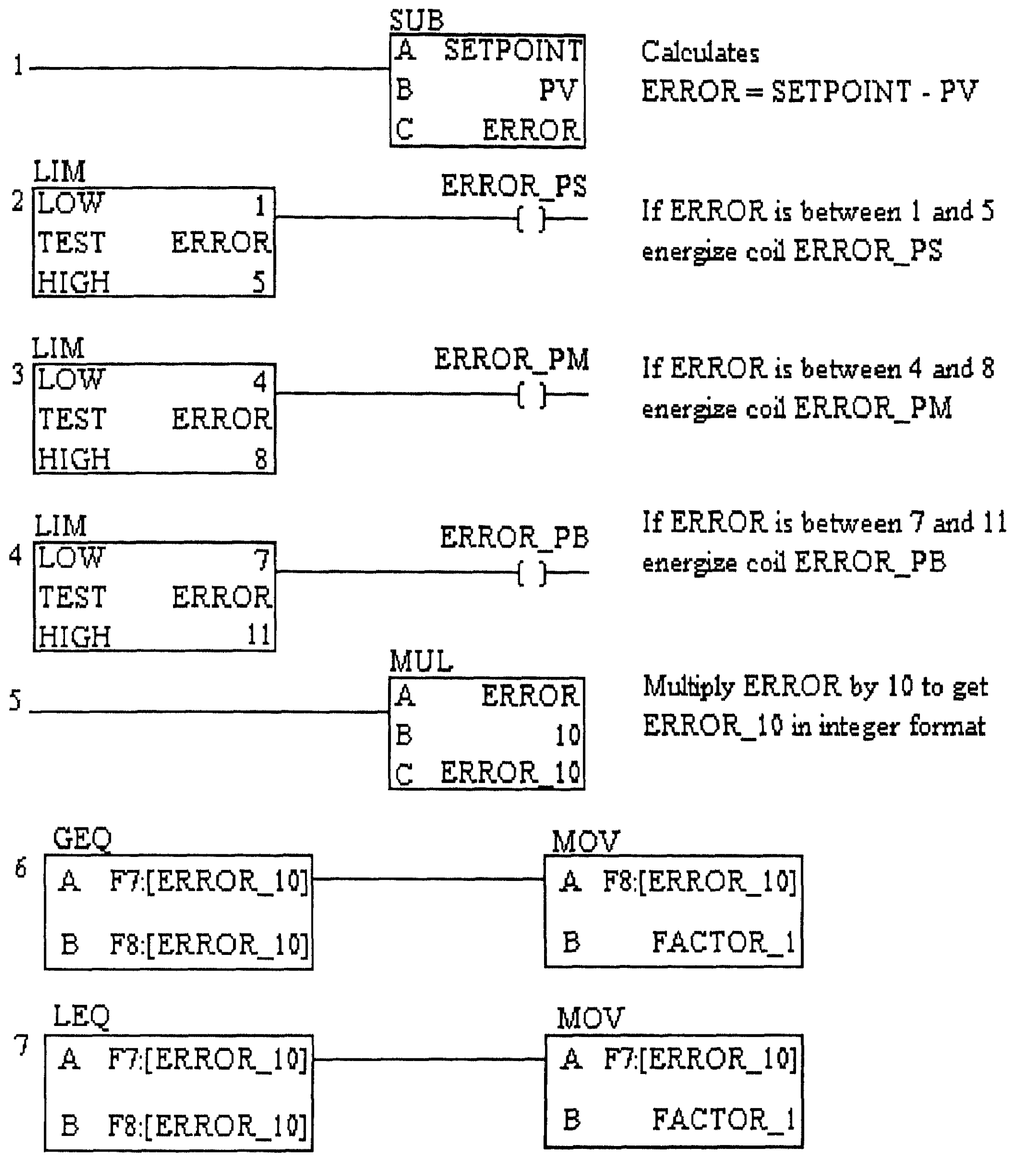
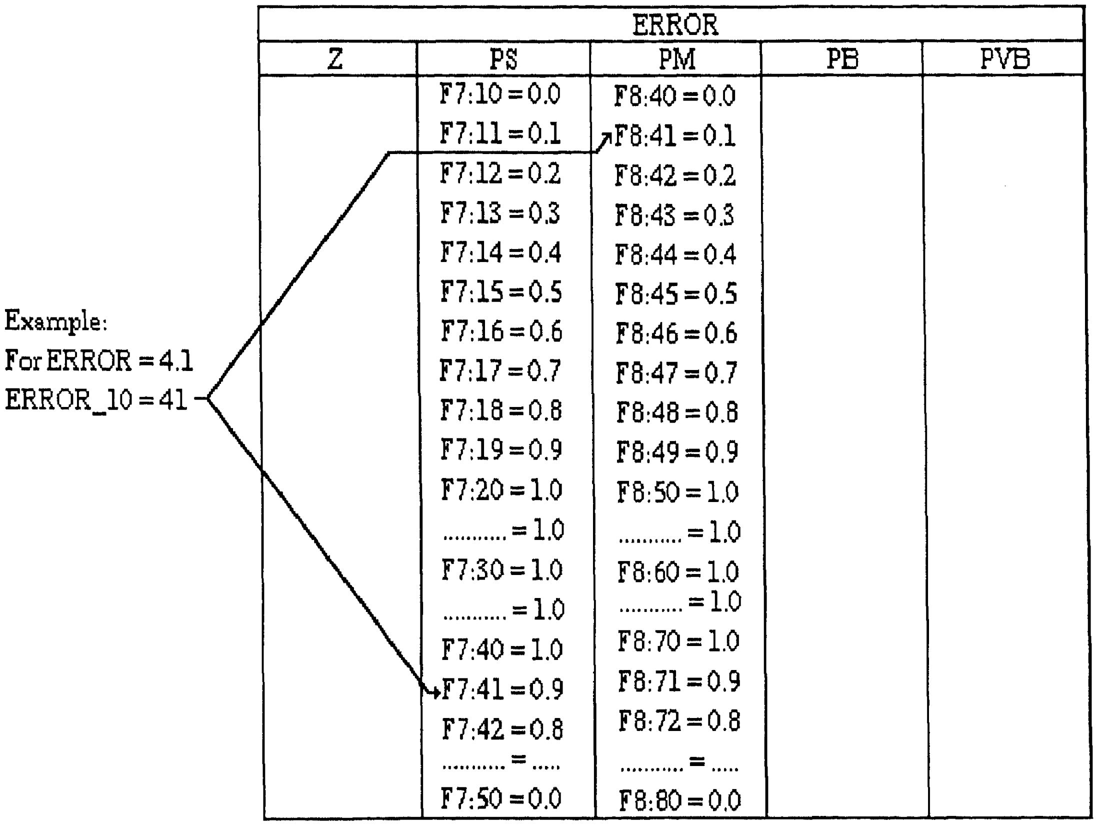


Figure 19 Defuzzification - Sample PLC code.

As can be seen rung 1 calculates the ERROR between the Setpoint and the Process Variable. "LIM" (Limit) instructions like the ones in rungs 2, 3, and 4 are used to energize coils that indicate the membership of that ERROR to a subset within the universe of discourse, e.g., PS, PM, PB, etc. Thus for an ERROR of 4.1, coils ERROR_PS and ERROR_PM are both energized. The degree of membership of this real-time ERROR in the subset PS and PM is calculated by a software pointer procedure in rungs 6 and 7.

E.g., Refering to Figure 20, an ERROR of 4.1 uses this pointer structure to identify itself with 90% of PS and 10% of PM i.e. degree of membership in PS is 0.9 and degree of membership in PM is 0.1.

These values are compared using the GEQ (Greater than or equal to) and LEQ (Less than or equal to) instructions, to move the smaller value in FACTOR_1. Similarly, FACTOR_2 is calculated for the variable DERROR. These two factors contribute towards calculating the final crisp output of the fuzzy controller.



Example:
For ERROR = 4.1
ERROR_10 = 41

ERROR				
Z	PS	PM	PB	PVB
	F7:10 = 0.0	F8:40 = 0.0		
	F7:11 = 0.1	F8:41 = 0.1		
	F7:12 = 0.2	F8:42 = 0.2		
	F7:13 = 0.3	F8:43 = 0.3		
	F7:14 = 0.4	F8:44 = 0.4		
	F7:15 = 0.5	F8:45 = 0.5		
	F7:16 = 0.6	F8:46 = 0.6		
	F7:17 = 0.7	F8:47 = 0.7		
	F7:18 = 0.8	F8:48 = 0.8		
	F7:19 = 0.9	F8:49 = 0.9		
	F7:20 = 1.0	F8:50 = 1.0		
	.... = 1.0	.... = 1.0		
	F7:30 = 1.0	F8:60 = 1.0		
	.... = 1.0	.... = 1.0		
	F7:40 = 1.0	F8:70 = 1.0		
	F7:41 = 0.9	F8:71 = 0.9		
	F7:42 = 0.8	F8:72 = 0.8		
	.... = ...	.... = ...		
	F7:50 = 0.0	F8:80 = 0.0		

Figure 20 Defuzzification - Sample PLC look-up table.

Step 6: In-house simulation tests:

Simulation tests were carried out in-house to make sure that the controller performance was within acceptable stability limits. Tests were conducted using an Omega SCR70

Power Controller for an electrical heater. The controller accepts an 4-20mA dc signal from the Analog output module of the PLC. The temperature was measured by an RTD probe via an Allen-Bradley RTD module.

Step 7: Installation qualification:

Installation qualification was performed on all instrumentation. Data sheets were completed for each instrument, logging important information like model number, serial number, calibration range, calibration interval etc.

Hardware qualification and software simulation test results qualified the system to be installed and tested onto the production system.

Step 8/9: Connect to production system and fine tuning:

The standard PID instruction, available as a user-configurable block was used for testing PID control. The Allen-Bradley processor has four choices of PID algorithms. The independent gains equation was used, i.e.

$$Output = Kp(E) + Ki\int_{0}^{t}(E)dt + Kd\frac{d}{dt}(E) + Bias$$

The temperature loop was tuned by adjusting the loop tuning constants Kp, Ki, and Kd. The software written in ladder logic for achieving fuzzy control was downloaded into the PLC controlling the WFI control system.

The response of the PID controller, Rule-base I, and Rule-base II for a setpoint of 90 degC are shown in Figures 21-23.

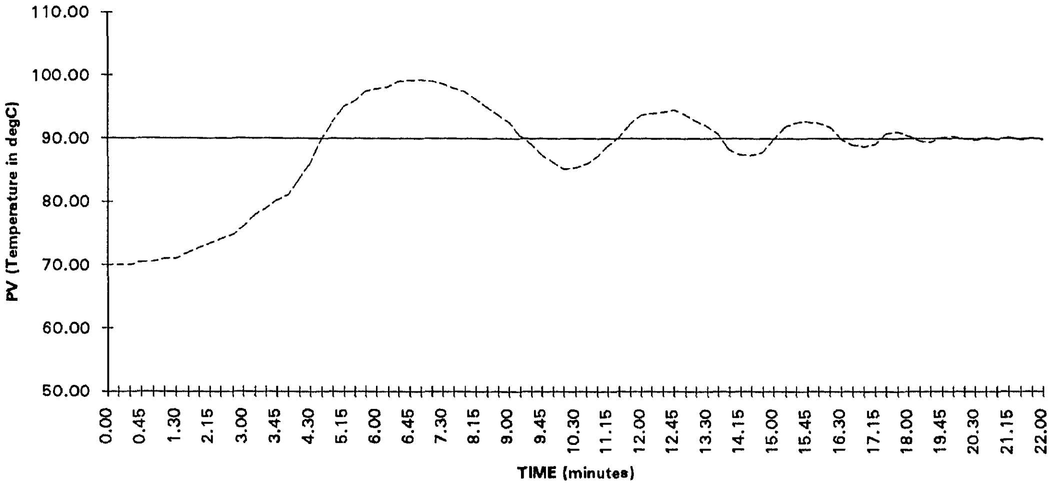


Figure 21 Response of PID controller.

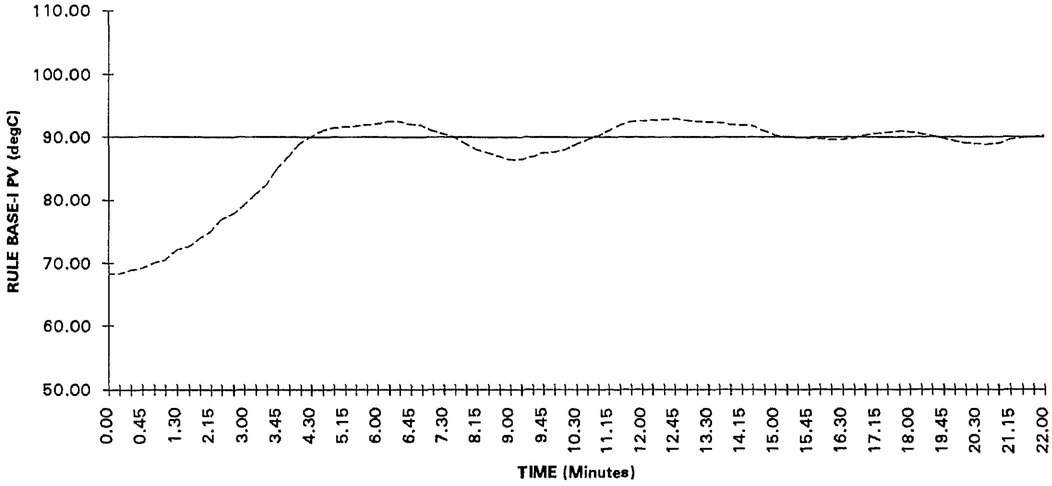


Figure 22 Response of fuzzy logic controller Rule-base I.

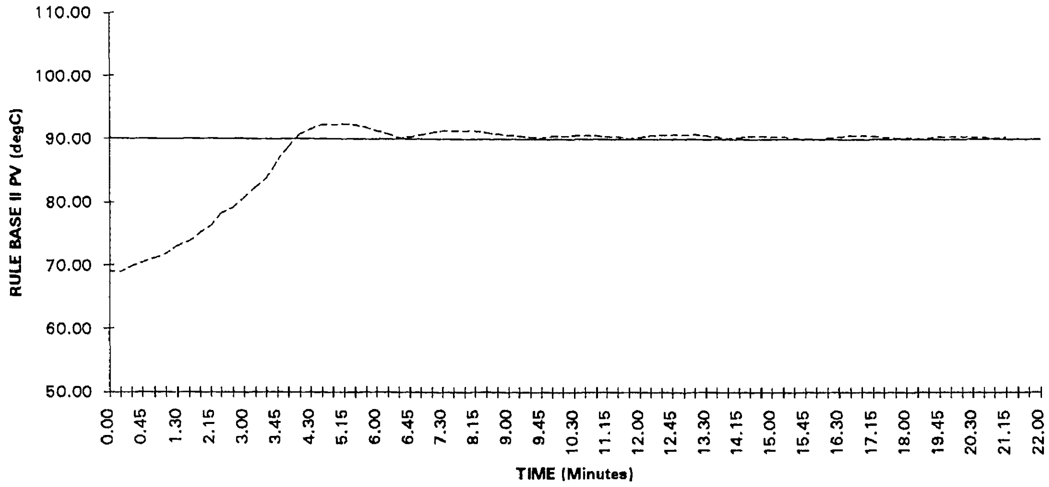


Figure 23 Response of fuzzy logic controller Rule-base II.

It was observed that though there was no significant change in the time required to reach the setpoint, overshoot was drastically reduced by the Rule-base I. Cycling of the process variable around the setpoint as observed in the PID response was also considerably reduced in Rule-base I.

The Rule-base II had similar results. There was no significant change in the time required to reach the setpoint. The overshoot was further reduced in comparison to the Rule-base I. This was due to the fact that the inlet water temperature was taken into consideration before applying control action to it. Though there was practically no cycling around the setpoint, the process variable lingered above the setpoint. This is the result of the centroid method of deriving rules, which kept the steam flow control valve open even though the outlet water temperature was greater than the setpoint.

A heuristic approach was followed to modify the rules. Based on experience the Rule-bases were modified as explained below.

The Rule-base I was modified as shown Figure 24. Since the system was having a very slow response rate, as long as the "Error" was large, I did not care about the difference in error "dError" and applied a large control signal to the steam flow control valve.

d(ERROR) = Previous - Current

ERROR =

SP - PV(out)

	NVB	**NB**	**NM**	**NS**	**Z**	**PS**	**PM**	**PB**	**PVB**
PVB	PVB	PVB	PVB	PVB	PVB	PVB	PVB	PVB	PVB
PB	PVB	PVB	PVB	PVB	PVB	PVB	PB	PB	PM
PM	PB	PB	PB	PB	PB	PM	PM	PM	PS
PS	PB	PB	PB	PB	PB	PM	PM	PS	PS
Z	PM	PM	PS	PS	Z	Z	Z	Z	Z
NS	PS	PS	Z	Z	Z	Z	Z	Z	Z
NM	PS	Z	Z	Z	Z	Z	Z	Z	Z
NB	Z	Z	Z	Z	Z	Z	Z	Z	Z
NVB	Z	Z	Z	Z	Z	Z	Z	Z	Z

Figure 24 Modified Rule-base I for a faster response.

The response for the modified Rule-base I is shown in Figure 25. Though there is some cycling around the setpoint, the response time is lower than the initial Rule-base I and considerably lower than the PID control.

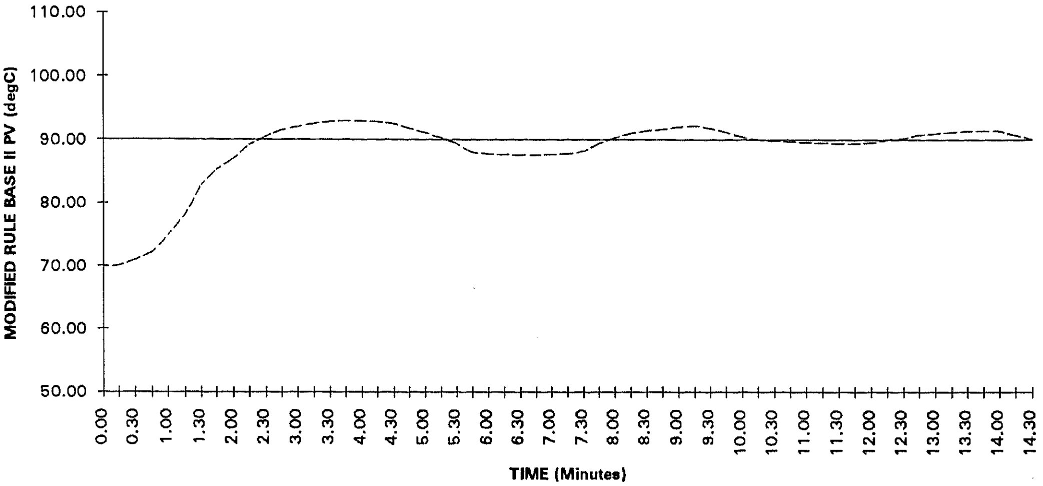


Figure 25 Response for the modified Rule-base I.

A similar heuristic approach was taken in modifying the Rule-base II. It was decided to close the steam flow control valve after the temperature of the outlet water reached the setpoint.

ERROR = SP - PV(in)

		PVB	PB	PM	PS	Z	NS	NM	NB	NVB
	PVB	PVB	PVB	PB	PB	PM	Z	Z	Z	Z
	PB	PVB	PVB	PB	PB	PM	Z	Z	Z	Z
	PM	PVB	PB	PB	PM	PS	Z	Z	Z	Z
	PS	PB	PB	PM	PS	PS	Z	Z	Z	Z
ERROR =	Z	PM	PS	PS	PS	Z	Z	Z	Z	Z
SP - PV(out)	NS	PS	Z	Z	Z	Z	Z	Z	Z	Z
	NM	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NB	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NVB	Z	Z	Z	Z	Z	Z	Z	Z	Z

Figure 26 Modified Rule-base II for a faster and tighter control action.

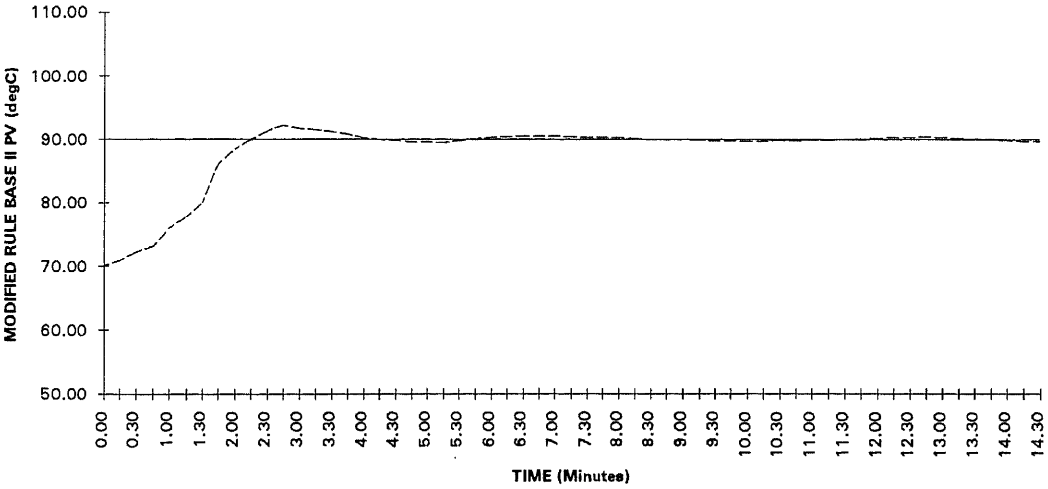


Figure 27 Response for modified Rule-base II.

As seen from Figure 27, the response of the modified Rule-base II is far more superior than the PID control response.

A comparison of the PID controller and the four fuzzy strategies is provided in Table 3. The following response evaluation factors are considered.

Dead time: The dead time is defined as the time delay between a change in the control signal and the beginning of its effect on the measurement.

Time constant: The time constant is defined as the time required to complete 63.2% of the total response. This number is not arbitrary and has significance in terms of the differential equations that model the process.

Peak error: It is the maximum deviation of the process variable from the setpoint.

Settling time: In the event of a transient input or a change in setpoint, the settling time is the time required for the process control loop to bring the process variable back to within the allowable setpoint range.

Processor scan time: The time required to the PLC processor to complete one scan. In this case it is important because the fuzzy control is accomplished through ladder logic software, and the time available to execute other processes is then reduced.

Table 3 Comparison of PID and Fuzzy Strategies.

	Dead time	Time constant	Peak error	Settling time	Processor scan time
PID control	45 secs	4.15 mins	9.3 degC	12.15 mins	28 msec
Rule-base I	30 secs	3.30 mins	2.5 degC	12.30 mins	53 msec
Rule-base II	30 secs	3.15 mins	2.3 degC	5.30 mins	53 msec
Modified Rule-base I	15 secs	1.30 mins	2.9 degC	7.45 mins	53 msec
Modified Rule-base II	15 secs	1.30 mins	2.2 degC	3.30 mins	53 msec

In cases of large load changes or process upsets, the PID controller response exhibited overshoots or undershoots followed by some cycling around the setpoint before settling down.

Process upsets in this case were sudden return water flowrate changes, which were manipulated by controlling a manual flow control valve in the path of the return water. Under normal loads, the return water flow was in the range of 30 - 50 gpm. With heavy loads, the return water flow would drop to a rate of about 10 - 25 gpm. Under very low loads, the return water flow would be greater than 50 gpm.

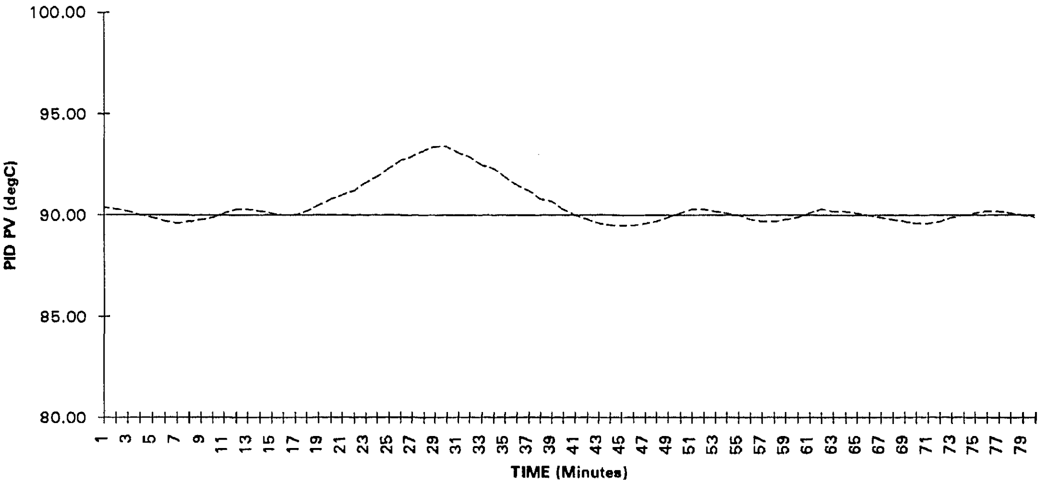


Figure 28 Response of PID under sudden heavy loads.

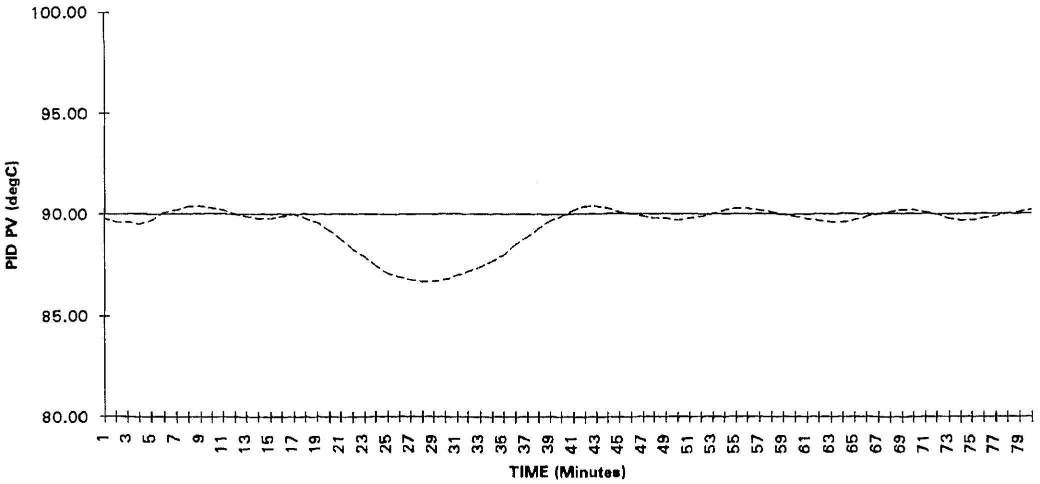


Figure 29 Response of PID under sudden low loads.

It was decided to add a flow-meter in the path of the return water into the heat-exchanger. Thus, the fuzzy logic controller has to incorporate changes in water flow. The third variable "Flow" is decomposed into a set of labeled fuzzy regions as shown in Figure 30.

The isosceles triangle function was chosen unlike the other two variables "Error" and "dError", where the trapezoidal function was used. This is due to the fact that "flow" is a very fast changing variable. The vertex of this isosceles triangle function corresponds to the mean value of the data set, while the base is twice the standard deviation of the data set. In this way, a triangular fuzzy number is formed which is convenient to manipulate.

Only three fuzzy regions are being used. More fuzzy regions can be implemented, but are not necessary since the occurrence of "flow glitches" was not expected to be very often.

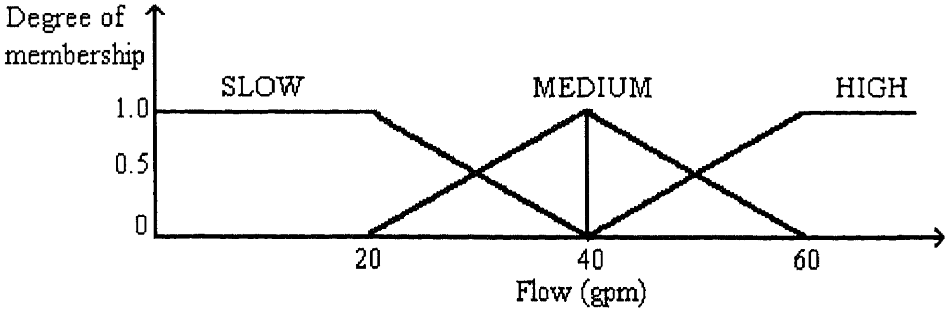


Figure 30 Variable "FLOW" decomposed into labeled fuzzy regions.

The rules associated with the variable "Flow" are explained in Table 4.

Table 4 Rule-Base III.

Rule	Antecedent Block	Consequent Block
Rule 1.1.1	IF Flow is High AND Error(out) is PVB AND Error(in) is PVB	THEN Output to valve is PVB
....	....	....
Rule 2.1.1	IF Flow is Medium AND Error(out) is PVB AND Error(in) is PVB	THEN Output to valve is PVB
....	....	....
Rule 3.1.1	IF Flow is Low AND Error(out) is PVB AND Error(in) is PVB	THEN Output to valve is PVB
....	....	....
Rule 3.9.9	....	....

I decided to use the modified Rule-base II, because it had the best response, and create a sort of three dimensional Rule-base III incorporating the effects of return water flow rates. The idea of a three dimensional Rule-base III will be clear in the Figure 31.

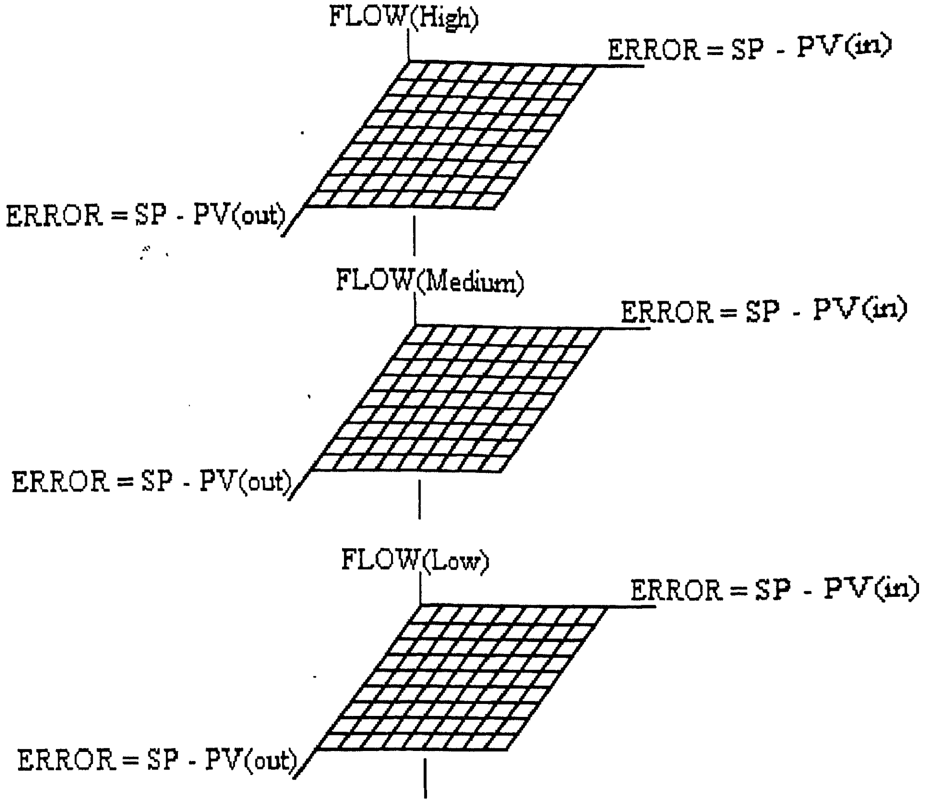


Figure 31 Three dimensional rule-base with three process variables.

The three subsections of the Rule-base III are shown in the Figures 32-34.

		ERROR = SP - PV(in)								
		PVB	**PB**	**PM**	**PS**	**Z**	**NS**	**NM**	**NB**	**NVB**
	PVB	PVB	PB	PM	PS	Z	Z	Z	Z	Z
	PB	PVB	PB	PM	PS	Z	Z	Z	Z	Z
	PM	PVB	PB	PM	PS	Z	Z	Z	Z	Z
	PS	PB	PM	PS	PS	Z	Z	Z	Z	Z
ERROR =	**Z**	PM	PS	PS	Z	Z	Z	Z	Z	Z
SP - PV(out)	**NS**	PS	Z	Z	Z	Z	Z	Z	Z	Z
	NM	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NB	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NVB	Z	Z	Z	Z	Z	Z	Z	Z	Z

Figure 32 Rule-base III for low return flow.

ERROR = SP - PV(in)

		PVB	PB	PM	PS	Z	NS	NM	NB	NVB
	PVB	PVB	PVB	PB	PB	PM	Z	Z	Z	Z
	PB	PVB	PVB	PB	PB	PM	Z	Z	Z	Z
	PM	PVB	PB	PB	PM	PS	Z	Z	Z	Z
	PS	PB	PB	PM	PS	PS	Z	Z	Z	Z
ERROR =	Z	PM	PS	PS	PS	Z	Z	Z	Z	Z
SP - PV(out)	NS	PS	Z	Z	Z	Z	Z	Z	Z	Z
	NM	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NB	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NVB	Z	Z	Z	Z	Z	Z	Z	Z	Z

Figure 33 Rule-base III for medium return flow.

ERROR = SP - PV(in)

		PVB	PB	PM	PS	Z	NS	NM	NB	NVB
	PVB	PVB	PVB	PVB	PB	PM	Z	Z	Z	Z
	PB	PVB	PVB	PVB	PB	PM	Z	Z	Z	Z
	PM	PVB	PVB	PB	PB	PM	Z	Z	Z	Z
	PS	PB	PB	PM	PM	PS	Z	Z	Z	Z
ERROR =	Z	PM	PM	PS	PS	Z	Z	Z	Z	Z
SP - PV(out)	NS	PM	Z	Z	Z	Z	Z	Z	Z	Z
	NM	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NB	Z	Z	Z	Z	Z	Z	Z	Z	Z
	NVB	Z	Z	Z	Z	Z	Z	Z	Z	Z

Figure 34 Rule-base III for high return flow.

The response of the fuzzy logic controller for sudden heavy loads with the Rule-base III in comparison to the PID response is shown in the Figure 35. As can be seen, the overshoot is considerably reduced. Similarly, the response for sudden low loads is shown in Figure 36.

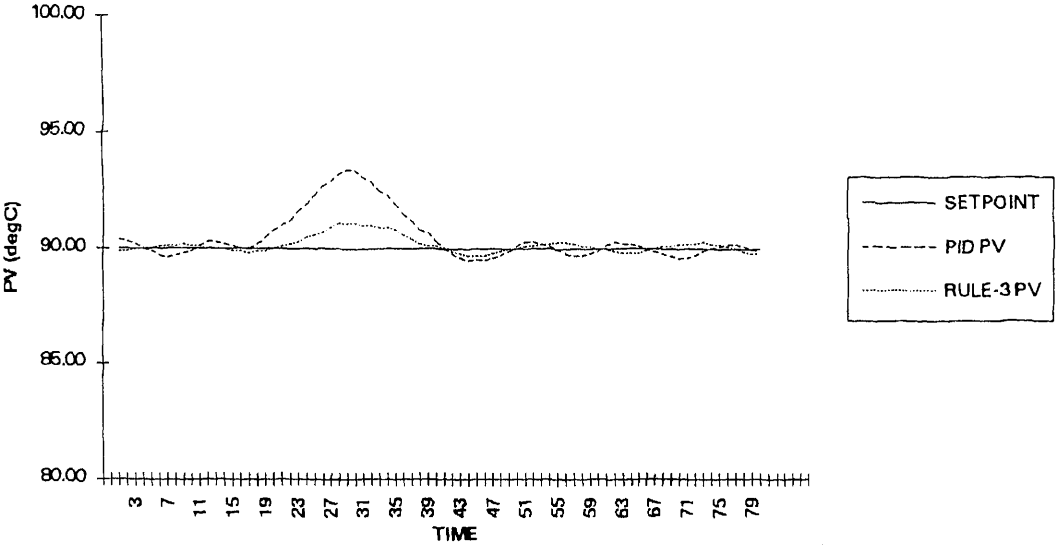


Figure 35 Response comparison under sudden heavy loads.

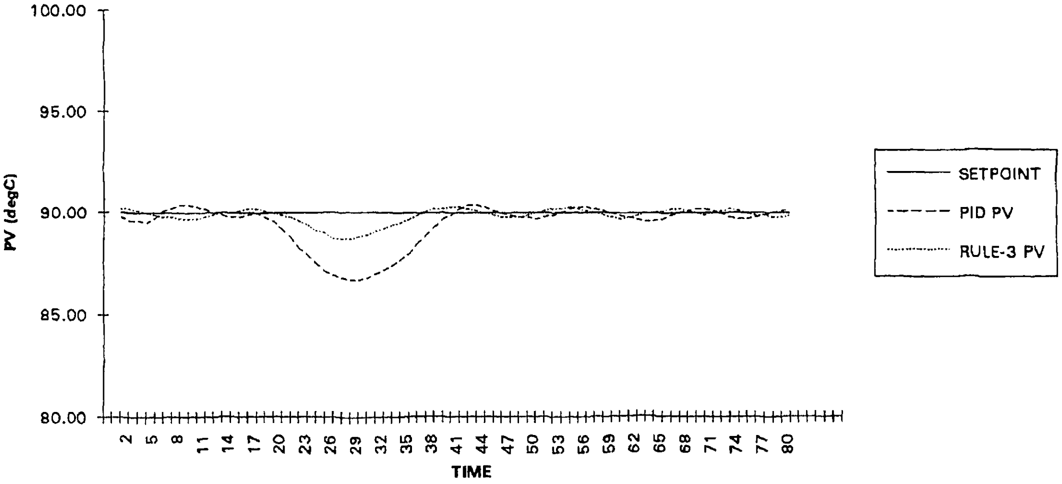


Figure 36 Response comparison under sudden low loads.

Step 10: Operational qualification:

The Operational Qualification was performed to confirm that the system response was within acceptable limits every time an upset was simulated. Loop upsets were purposely induced to verify the response, and the system passed.

CHAPTER 7

CONCLUSIONS

The problems associated with the conventional PID controller were studied for a Heat Exchanger in a WFI system. Multiple variables affecting the stability of the controller were found to be the main problem causing the inconsistent behavior. Several Rule-base strategies were designed and implemented with Allen-Bradley's 6200 PLC ladder logic development software.

Rule-base I utilizes temperature error at the heat exchanger outlet and the rate of change of this error, as the inputs to the FLC. Though the time required to reach the setpoint had no significant improvement the overshoot was drastically reduced in comparison to a PID controller. Rule-base I was further modified for a faster response. Rule-base II utilized the temperature error at the heat exchanger outlet and the temperature error at the heat exchanger inlet as inputs to the FLC. Similar to Rule-base I, though there was no significant improvement in the time required to reach the setpoint, the overshoot was further reduced due to the fact that temperature was sensed at the inlet. Rule-base II was also modified for better response. Finally, the disturbances due to sudden load changes were reduced by developing a third strategy, that utilized three inputs to the FLC, namely, temperature error at inlet, temperature error at outlet, and flowrate at inlet. This evolution of different Rule-base strategies signifies the importance of proper choice of input variables to the FLC.

The usefulness of applying PLC ladder logic to formulate fuzzy logic rules is demonstrated. The ladder logic developed to apply fuzzy logic theory to one process control loop was very lengthy and cumbersome. A fair sized facilities automation project has several such loops and scheming fuzzy logic rules for all of them would not be cost effective. Also, it was experienced that tuning the fuzzy logic controller was a very timeconsuming process involving maintaining records of the responses observed, modifying the rules, modifying the rule-bases, and implementing modifications in the software. In comparison, a PID block is readily available and can be easily configured and tuned.

The PID algorithm has been around for several decades and engineers/operators are educated with its theory, it is easily configurable, and easily tuned. Formal education in fuzzy logic control is not readily available, simulation software written by one engineer is not easily understood by others and making modifications can be a never-ending task. However, well documented software can ease the task of making changes.

Using the PID algorithm, the processor scan time was observed to be 28msecs. The extensive number crunching of Rule-base I and Rule-base II raised the processor scan time to 53msecs whereas Rule-base III further raised the scan time to 67msecs. Obviously, with several fuzzy control loops, the processor scan time will be drastically increased. This could deteriorate the efficiency of the overall processor performance by slowing the processes of remote data transfers, historical trending of critical parameters since recording calculations would be performed less often with slow scan times. Further research can be conducted by implementing fuzzy logic control in operator interfaces and thus freeing up the processor. E.g., the fuzzy logic control algorithm can be formulated as a background task in 'C' language in Intellution's FIX DMACS (Fully Integrated Control System, Distributed Manufacturing Automation and Control Software). The FIX DMACS PLC I/O driver will read the input variables from the PLC, perform fuzzification, implement rules, perform defuzzification, and write the output variable back to the PLC. Graphical representation of rule-bases should be developed as aids to ease and quicken the process of tuning the loops and refining the rule-bases.

As fuzzy logic gets designed into more applications, two important things will happen. First, entirely new applications will emerge that were not considered practical. Second, new chips will be designed that are specifically adapted to do fuzzy logic calculations [22]. These new devices will include processors with some instructions that are specific to fuzzy logic and microcontrollers with hardware fuzzy logic engines as peripherals to a conventional CPU. At the present time a chicken-and-egg problem operates: Applications are waiting for fuzzy logic microcontrollers to become available, but chip manufacturers are waiting for fuzzy logic applications to be developed so they know what fuzzy logic features are important.

Allen-Bradley has several specialty modules that can be used for different applications, Eg. the ASCII-Basic module that can be used for integrating with devices that have an ASCII interface, several communication modules etc. As technology progresses and with the availability of new fuzzy logic processors, Fuzzy control modules should be designed to house in the processor chassis. A new set of instructions and user configurable blocks should be developed for process control applications. User friendly tools for easing the process of tuning and modifications should be developed. These modules can then perform all calculations related to the fuzzy control, and free the main processor for other tasks thus improving the overall system performance in terms of scan times.

Validating the system was a very straightforward process. All the rules were documented and tested. A paper trail is provided for the theory used in deriving rules and the assumptions made in forming the Rule-bases.

Fuzzy logic is not the best approach for every control problem [9]. As designers look at its power and expressiveness, they must decide where to apply it, and also how best to manage software projects based on this new technology. Increasingly, project managers, system architects, and design engineers are asking tough questions: what type of projects can benefit from the use of fuzzy logic? How do I control projects with this technology? What is the best way to design, develop, deliver, and test systems using fuzzy logic and its related technologies?

REFERENCES

1 G. Anderson, "*Fundamentals Of Fuzzy Logic: Part I*", Sensors, March 1993, pp. 40-49.

2 G. Anderson, "*Fundamentals Of Fuzzy Logic: Part II*", Sensors, April 1993, pp. 42-63.

3 G. Anderson, "*Fundamentals Of Fuzzy Logic: Part III*", Sensors, May 1993, pp. 38-42.

4 J. J. Barron, "*Putting Fuzzy Logic Into Focus*", Byte, April 1993, pp.111-118.

5 F. J. Bartos, "*Fuzzy Logic Is Clearly Here To Stay*", Control Engineering, July 1992, pp.45-47.

6 J. A. Bernard, "*Use Of Rule-Based System For Process Control*", IEEE Control System Magazine, Vol. 8, No. 5, 1988.

7 M. Braae and D. A. Rutherford, "*Fuzzy Relations In A Control Setting,*" Kybernetes, Vol.7, No. 3, 1978.

8 D. Brubaker, "*Fuzzy Logic System Solves Control Problem*", EDN, June 1992, pp. 121-127.

9 D. Brubaker, "*An Accelerated Kernel For Fuzzy Systems*", AI Expert, March 1993, pp. 38-43.

10 E. Cox, "*Fuzzy Fundamentals*", IEEE Spectrum, October 1992, pp. 58-61.

11 L. M. Gordon, "*Basic Concepts, Terminology And Techniques For Process Control,*" Chemical Engineering, May 1983.

12 L. M. Gordon, "*Feedback Control Modes,*" Chemical Engineering, Aug. 1983.

13 M. M. Gupta, G. M. Trojan, and J. B. Kiszka, "*Controllability Of Fuzzy Control Systems*", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 16, No. 4, 1986.

14 N. Infelise, "*A Clear Vision Of Fuzzy Logic*", Control Engineering, July 1991, pp.28-30.

References (continued)

15 J. P. King, "*A Vendor's Contribution To Computer System Software Validation*", Pharmaceutical Engineering, Sept/Oct. 1992.

16 C. D. Johnson, "Process Control Instrumentation Technology", John Wiley and Sons Inc. NY, 3RD Edition.

17 A. Kaufmann and M. M. Gupta, "Introduction To Fuzzy Arithmetic", New York: Van Nostrand, 1985.

18 Y. F. Li and C. C. Lau, "*Development Of Fuzzy Algorithms For Servo Systems*", IEEE Control Systems Magazine, April 1989, pp. 65 - 71.

19 C. C. Lee, "*Fuzzy Logic In Control Systems: Part I*", IEEE Trans. on Systems, Man, and Cybernetics, Vol. 20, No. 2, Mar/Apr 1990.

20 C. C. Lee, "*Fuzzy Logic In Control Systems: Part II*", IEEE Trans. on Systems, Man, and Cybernetics, Vol. 20, No. 2, Mar/Apr 1990.

21 K. L. Tang and R. J. Mulholland, "*Comparing Fuzzy Logic With Classical Controller Designs*," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-17, no. 6, pp. 1085 - 1087, 1987.

22 J. M. Sibigtroth, "*Implementing Fuzzy Expert Rules In Hardware*", AI Expert, April 1992, pp. 25-31.