35

Fuzzy Logic in Electric Drives

Ahmed Rubaai, Ph.D.
Department of Electrical Engineering, Howard University, Washington, D.C., USA

35.1 Introduction

Over the years, we have increasingly been on the search to understand the human ability to reason and make decisions, often in the face of only partial knowledge. The ability to generalize from limited experience into areas as yet encountered is one of the fascinating abilities of the human mind. Traditionally, our attempt to understand the world and its functions has been limited to finding mathematical models or equations for the systems under study. This approach has proven extremely useful, particularly in an age when very fast computers are available to most of us with only a minimum amount of capital outlay. And even when these computers are not fast enough, many researchers can gain access to super computers capable of giving numerical solutions to multiorder differential equations that are capable of describing most of the industrial processes.

This analytical enlightenment, however, has come at the cost of realizing just how complex the world is. At this point, we have come to realize that no matter how simple the system is, we can never hope to model it completely. So instead, we select suitable approximations that give us answers that we think, are sufficiently precise. Because our models are incomplete, we are faced with one of the following choices:

1. Use the approximate model and introduce probabilistic representations to allow for the possible errors.
2. Seek to develop an increasingly complex model in the hope that we can find one, that completely describes the systems while being solvable in real time.

This dilemma has led a few, most notably Zadeh [1], to return the decision-making process employed by our brilliant minds when confronted with incomplete information. The approach taken in those cases makes allowances for the imprecision caused by incomplete knowledge and actually embracing the imprecision in forming an analytical framework. This approach involved artificial intelligence using approximate reasoning or fuzzy logic as it now commonly known. As a result, artificial intelligence using fuzzy logic has proven extremely useful in ascribing a logic mechanism to a wide range of topics from economic modeling and prediction to biology analysis to control engineering. In this chapter, an examination of the principles involved in artificial intelligence using fuzzy logic and its application to electric drives is discussed.

35.2 The Fuzzy Logic Concept

Fuzzy logic arose from a desire to incorporate logical reasoning and the intuitive decision making of an expert operator into an automated system [1]. The aim is to make decisions based on a number of learned or predefined rules, rather than

numerical calculations. Fuzzy logic incorporates rule-base structure in attempting to make decisions [1–5]. However, before the rule-base can be used, the input data should be represented in such a way as to retain meaning, while, still allowing for manipulation. Fuzzy logic is an aggregation of rules, based on the input state variables condition with a corresponding desired output. A mechanism must exist to decide on which output, or combination of the different outputs, will be used since each rule could conceivably result in a different output action.

Fuzzy logic can be viewed as an alternative form of input/output mapping. Consider the input premise, x, and a particular qualification of the input x represented by, A_i. Additionally, the corresponding output, y, can be qualified by expression C_i. Thus, a fuzzy logic representation of the relationship between the input x and the output y could be described with the following:

$$\begin{aligned}
&R_1: \text{ IF } x \text{ is } A_1 \text{ THEN } y \text{ is } C_1\\
&\dots \quad \dots \quad \dots \quad \dots\\
&R_2: \text{ IF } x \text{ is } A_2 \text{ THEN } y \text{ is } C_2\\
&\dots \quad \dots \quad \dots \quad \dots\\
&R_n: \text{ IF } x \text{ is } A_n \text{ THEN } y \text{ is } C_n
\end{aligned} \tag{35.1}$$

where

x is the input (state variable).

y is the output of the system.

A_i are the different fuzzy variables used to classify the input x.

C_i are the different fuzzy variables used to classify the output y.

The fuzzy rule representation is based on linguistic [1, 3]. Thus, the input x is a linguistic variable that corresponds to the state variable under consideration. Furthermore, the elements A_i are fuzzy variables that describe the input x. Correspondingly, the elements C_i are the fuzzy variables used to describe the output y. In fuzzy logic control, the term "linguistic variable" refers to whatever state variables the system designer is interested in [1]. Linguistic variables that are often used in control applications include speed, speed error, position, and derivative of position error. The fuzzy variable is perhaps better described as a fuzzy linguistic qualifier. Thus the fuzzy qualifier performs classification (qualification) of the linguistic variables. The fuzzy variables frequently employed include negative large, positive small, and zero. Several papers in the literature use the term "fuzzy set" instead of "fuzzy variable," however, the concept remains the same. Table 35.1 illustrates the difference between fuzzy variables and linguistic variables.

TABLE 35.1 Fuzzy and linguistic variables

Linguistic variables	Fuzzy variables (linguistic qualifiers)
Speed error (SE)	Negative large (NL)
Position error (PE)	Zero (ZE)
Acceleration (AC)	Positive medium (PM)
Derivative of position error (DPE)	Positive very small (PVS)
Speed (SP)	Negative medium small (NMS)

Once the linguistic and fuzzy variables have been specified, the complete inference system can be defined. The fuzzy linguistic universe, U, is defined as the collection of all the fuzzy variables used to describe the linguistic variables [6–8], i.e. the set U for a particular system could be comprised of NS, ZE, and PS. Thus, in this case the set U is equal to the set of [NS, ZE, PS]. For the system described by Eq. (35.1), the linguistic universe for the input x would be the set $U_x = [A_1\ A_2 \dots A_n]$. Similarly, the linguistic universe for the output y would be the set $U_y = [C_1\ C_2 \dots C_n]$.

35.2.1 The Fuzzy Inference System (FIS)

The basic fuzzy inference system (FIS) can be classified as:

Type 1 fuzzy input fuzzy output (FIFO)
Type 2 fuzzy input crisp output (FICO)

Type 2 differs from the first in that the crisp output values are predefined and, thus, built into the inference engine of the FIS. On the contrary, Type 1 produces linguistic outputs. Type 1 is more general than Type 2 as it allows redefinition of the response without having to redesign the entire inference engine. One draw back is the additional step required converting the fuzzy output of the FIS to a crisp output.

Developing a FIS and applying it to a control problem involves several steps:

1. Fuzzification.
2. Fuzzy rule evaluation (fuzzy inference engine).
3. Defuzzification.

The total FIS is a mechanism that relates the inputs to a specific output or set of outputs. First, the inputs are categorized linguistically (fuzziffication), then the linguistic inputs are related to outputs (fuzzy inference), and finally, all the different outputs are combined to produce a single output (defuzziffication). Figure 35.1 shows a block diagram of the fuzzy inference system.

35.2.2 Fuzzification

Fuzzification is the conversion of crisp numerical values into fuzzy linguistic quantifiers' [7, 8]. Fuzzification is performed using membership functions. Each membership function evaluates how well the linguistic variable may be described by

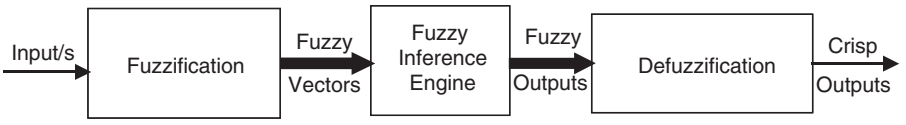

FIGURE 35.1 Fuzzy inference system.

a particular fuzzy qualifier. In other words, the membership function derives a number that is representative of the suitability of the linguistic variable to be classified by the fuzzy variable (set). This suitability is often described as the degree of membership. In order to maintain a relationship to traditional binary logic, the membership values must range from 0 to 1 inclusive. Figure 35. 2 shows the mechanism involved in the fuzzification of crisp inputs when multiple inputs are involved. Since each input has a number of membership functions (one for each fuzzy variable), the outputs of all the membership functions for a particular crisp numerical input are combined to form a fuzzy vector.

Any number of normalizing expressions can perform fuzzification. Two of the more common functions are the linear and Gaussian [4]. In both cases there is one parameter, μ, that indicates the midpoint of the region and another, σ, that defines the width of the membership functions. For the linear function, the width is specified by, σ_L, and the midpoint by, μ_L. Similarly for the gaussian function, the width is specified by, σ_G, and the midpoint by, μ_G. Equations (35.2a) and (35.2b) define the linear and gaussian membership functions, respectively.

Linear function:

$$\begin{cases} 1 - \left|\frac{x-\mu_L}{\sigma_L}\right| & \text{if} \quad x \in [(\mu_L - \sigma_L), (\mu_L + \sigma_L)] \\ 0 & \text{Otherwise} \end{cases} \tag{35.2a}$$

Gaussian function:

$$\exp\left(-\frac{(x-\mu_G)^2}{2(\sigma_G)^2}\right) \tag{35.2b}$$

where

$$\mu_L = \mu_G \tag{35.3a}$$

$$\sigma_L = 3\sigma_G \tag{35.3b}$$

The relations expressed by equations (35.3a) and (35.3b) are made because of the characteristics of the gaussian function. Because a gaussian membership function may never have a membership value of zero, some appropriate value close to zero must be chosen as the cut-off point. At a distance of $3\sigma_G$ from the mean, the gaussian membership function results in a membership value of 0.05. Thus the width of the gaussian function is chosen as $3\sigma_G$.

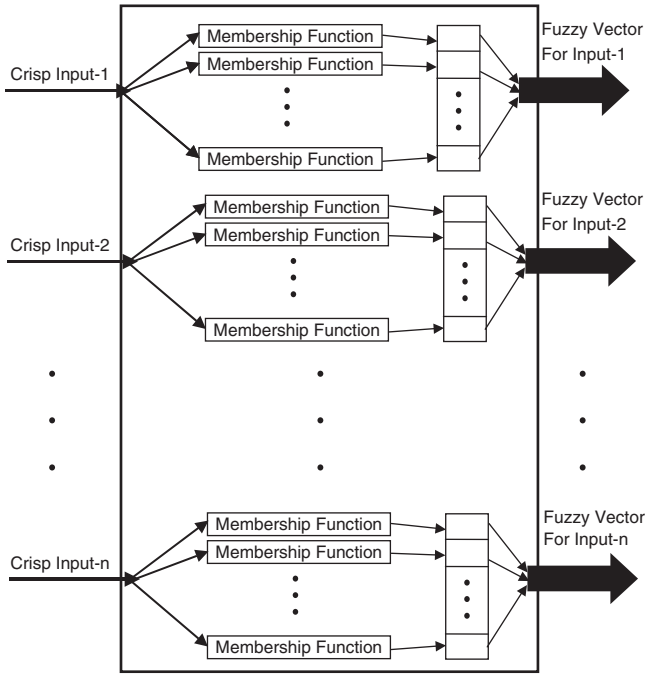

FIGURE 35.2 Fuzzification of the crisp numerical inputs.

As previously mentioned, fuzzification of the input has resulted in a fuzzy vector where each component of this vector represents the degree of membership of the linguistic variables into a specific fuzzy variable's category. The number of components of the fuzzy vector is equal to the number of fuzzy variables used to categorize specific linguistic variable. For illustrative purposes, we consider an example with a linguistic variable x and three fuzzy variables PV, ZE, and NV. If we describe the membership function (fuzzifier) as X, then we will have three membership functions: X_{PV}, X_{ZE}, and X_{NV}. The fuzzy linguistic universe for the input x can be described by the set U_x that is defined in Eq. (35.4).

$$U_x = [X_{NV}\ X_{ZE}\ X_{PV}] \tag{35.4}$$

where

X_{PV} is the membership function for the positive fuzzy variables.

X_{ZE} is the membership function for the zero fuzzy variables.

X_{NV} is the membership function for the negative fuzzy variables.

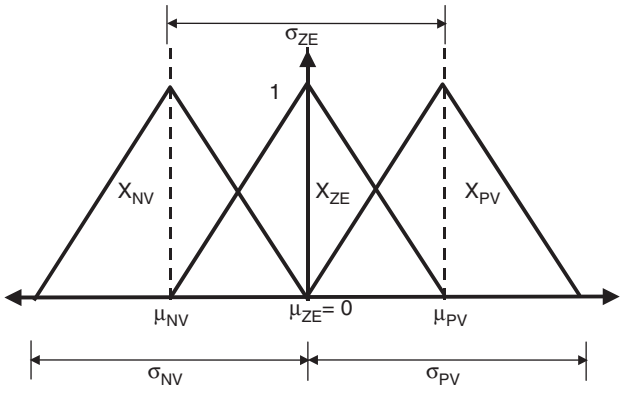

FIGURE 35.3 Linear membership functions.

Thus, the fuzzy vector, which is the output of the fuzzification step of the inference system, can be denoted by $\underline{\mathbf{x}}$.

$$\underline{\mathbf{x}} = [x_{NV}\, x_{ZE}\, x_{PV}] = [X_{NV}(x)\, X_{ZE}(x)\, X_{PV}(x)] \tag{35.5}$$

where

x_{NV} = the membership value of x into the fuzzy region denoted by negative.

x_{ZE} = the membership value of x into the fuzzy region denoted by zero.

x_{PV} = the membership value of x into the fuzzy region denoted by positive.

Equation (35.2a) represents the linear membership functions, which are illustrated in Fig. 35.3. The linear function can be modified to form the linear-trapezoidal function. Under this modification, if the input x falls between zero and the mean, μ_L, of the respective region, then Eq. (35.2a) is used, otherwise, the membership value is equal to one. Thus, we arrive at the membership functions shown in Fig. 35.4. Each region of the linear and trapezoidal-linear membership functions is distinguished from another by the different values of σ_L and μ_L. One important criterion that should be taken into consideration is that the union of the domain of all membership functions for a given input must cover the entire range of the input [9]. Thus the trapezoidal modification is often employed to ensure coverage of the entire input space.

The gaussian membership function is characterized by Eq. (35.2b). The gaussian function can also be modified to form the trapezoidal-gaussian function. In this case, if the input falls between the mean, μ_G and zero, Eq. (35.2b) is used to find the membership value. Otherwise the membership value becomes one. The gaussian function is shown in Fig. 35.5 and the modified version is shown in Fig. 35.6.

A third type of membership function known as the fuzzy singleton is also considered. The fuzzy singleton is a special function in which the membership value is one for only one particular value of the linguistic input variable, and zero otherwise [4]. Thus, the fuzzy singleton is a special case of the membership function with a width, σ, of zero. Therefore, the only parameter that needs to be defined is the mean, μ_s, of the singleton. Thus, if the input is equal to μ_s, then the membership value is one. Otherwise it is zero. We can denote the singleton membership function as $S(\mu_s)$.

The fuzzy singleton function is quite useful in defining some special membership functions. If we would like to dispense with the need for a continuous degree of membership and prefer a binary valued function, the fuzzy singleton is an ideal candidate. We can form the membership function representing the fuzzy variable as a collection of fuzzy singletons ranging within the regions denoted by $[\mu+\sigma/2,\ \mu-\sigma/2]$. A graphical representation using three fuzzy variables (membership functions) is shown in Fig. 35.7 (in Fig. 35.7 the corners are only slanted so that the regions are easier to distinguish from each other). Thus, the membership functions shown in Fig. 35.7

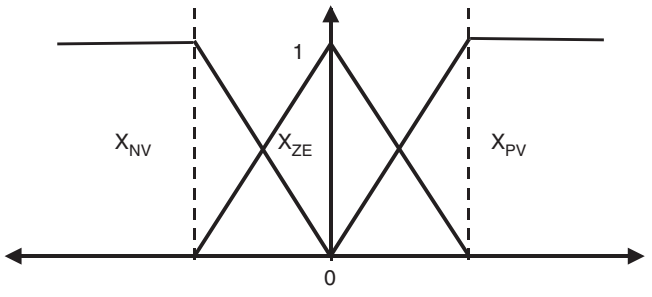

FIGURE 35.4 Linear-trapezoidal membership functions.

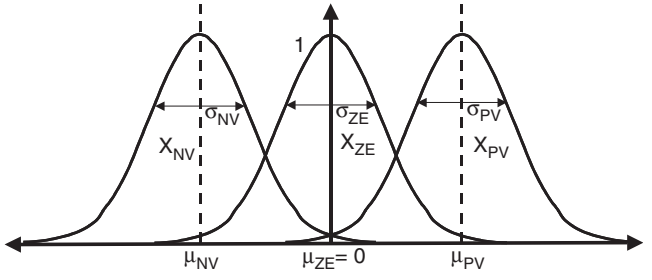

FIGURE 35.5 Gaussian membership functions.

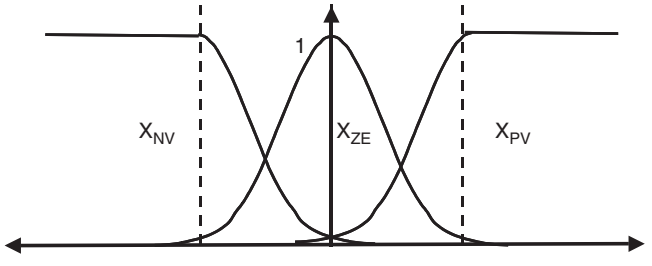

FIGURE 35.6 Gaussian-trapezoidal membership functions.

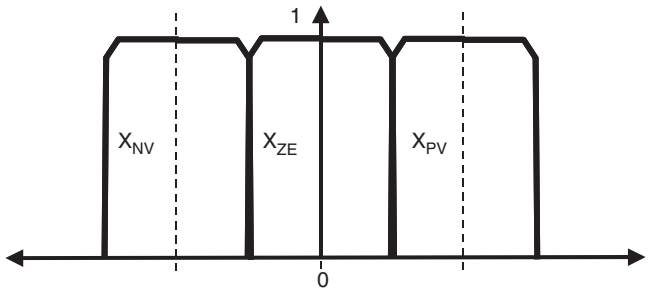

FIGURE 35.7 Membership functions comprised of fuzzy singletons.

would be best defined as an integral of the fuzzy singleton with respect to the mean over the width of the function. Consequently, the membership function X (σ,μ) would be defined as follows:

$$X(\sigma,\mu) = \int_{\mu-\sigma/2}^{\mu+\sigma/2} S(\mu_s)d\mu_s \tag{35.6}$$

where

$S(\mu_s)$ is the singleton function.

35.2.3 The Fuzzy Inference Engine

The fuzzy inference engine uses the fuzzy vectors to evaluate the fuzzy rules and produce an output for each rule. Figure 35.8 shows a block diagram of the fuzzy inference engine. Note that the rule-based system takes the form found in Eq. (35.1). This form could be applied to traditional logic as well as fuzzy logic albeit with some modification. A typical rule R would be:

$$\mathrm{R}_i: \quad \text{IF} \quad x_i \quad \text{THEN} \quad y = C_i \tag{35.7}$$

where

x_i is the result of some logic expression.

The logical expression used in the case of fuzzy inference Eq. (35.7) is of the form

$$x \in X_i \tag{35.8}$$

where

x is the input.

X_i is the linguistic variable.

In binary logic, the expression in Eq. (35.8) results in either true or false. However, in fuzzy logic we often require a continuum of truth-values. Figures 35.9 and 35.10 illustrate the difference between binary logic and fuzzy logic. In traditional logic, there is a single point representing the boundary between true and false. While in fuzzy logic, there is an entire region over which there is a continuous variation between truth and falsehood. The second part of Eq. (35.7), $y = C_i$, is the action prescribed by the particular rule. This portion indicates what value will be assigned to the output. This value could be either a fuzzy linguistic description or a crisp numerical value.

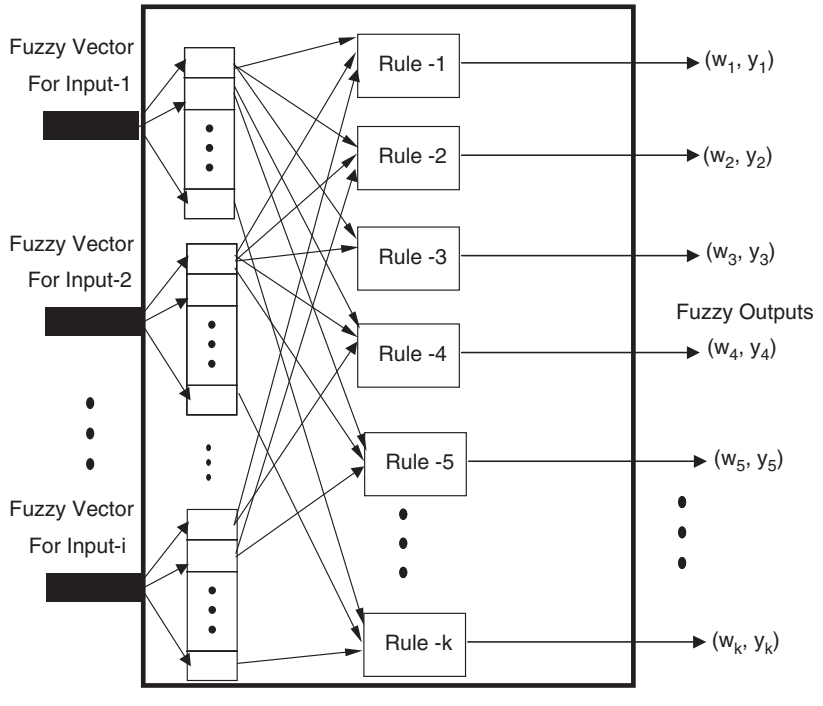

FIGURE 35.8 Fuzzy inference engine.

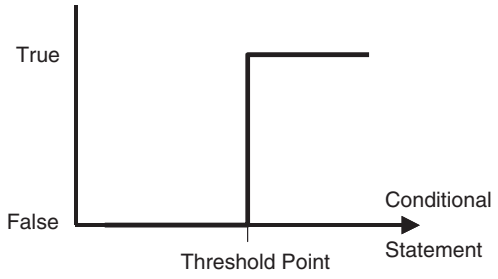

FIGURE 35.9 Binary logic statement evaluation.

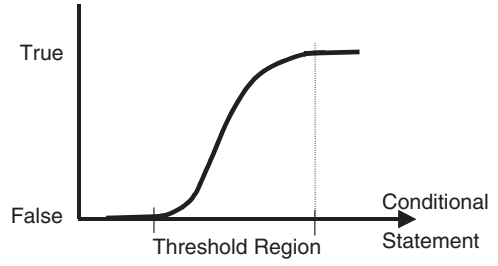

FIGURE 35.10 Fuzzy logic statement evaluation.

The logical expression that dictates whether the result of a particular rule is carried out could involve multiple criteria. Multiple conditions imply multiple input, as is most often the case in many applications of fuzzy logic to dynamic systems. Let us describe a system with two inputs x^1 and x^2. For simplicity of explanation and without loss of generality, we will use three fuzzy variables, namely, PV, ZE, and NV. Although each linguistic variable x^1 and x^2 uses the same fuzzy qualifiers, each input must have its own membership functions since they belong to different spaces. Thus, we will have two fuzzy vectors $\underline{\mathbf{x}}^{\mathbf{1}}$ and $\underline{\mathbf{x}}^{\mathbf{2}}$, for the first and second input, respectively.

$$\underline{\mathbf{X}}^{\mathbf{1}} = \lfloor X^1_{\mathrm{NV}}(x^1) \quad X^1_{\mathrm{ZE}}(x^1) \quad X^1_{\mathrm{PV}}(x^1) \rfloor = \lfloor x^1_1 \; x^1_2 \; x^1_3 \rfloor \tag{35.9a}$$

$$\underline{\mathbf{X}}^{\mathbf{2}} = \lfloor X^2_{\mathrm{NV}}(x^2) \quad X^2_{\mathrm{ZE}}(x^2) \quad X^2_{\mathrm{PV}}(x^2) \rfloor = \lfloor x^2_1 \; x^2_2 \; x^2_3 \rfloor \tag{35.9b}$$

where

X^n_{NV} = Membership function for the negative fuzzy variable for input n.

X^n_{ZE} = Membership function for the zero fuzzy variable for input n.

X^n_{PV} = Membership function for the positive fuzzy variable for input n.

x^1 = First linguistic variable (input-1).

x^2 = Second linguistic variable (input-2).

x^1_i = The degree of membership of input-1 into the ith fuzzy variable's category.

x^2_j = The degree of membership of input-2 into the jth fuzzy variable's category.

The inference mechanism in this case would be specified by the rule R_{ij}:

$$\mathrm{R}_{ij}: \quad \text{IF} \quad (x^1_i \text{ AND } x^2_j) \text{ THEN } y = C_{ij} \tag{35.10}$$

The specification R_{ij} is made so as to emphasize that all combinations of the components of the fuzzy vectors should be used in separate rules. A specific example of one of the rules can be described as follows:

$$\mathrm{R}_{13} \text{ IF } (x^1_1 \text{ AND } x^2_3 \text{ THEN } y = C_{13}$$

This rule can be written linguistically as:

$$\mathrm{R}_{13} \text{ IF}((x^1 \text{is Negative}) \text{ AND } (x^2 \text{ is Positive}))\text{THEN } y = C_{13}$$

where

x^1 is the first linguistic variable.

x^2 is the second linguistic variable.

C_{13} is the output action to be defined by the system designer.

The AND in Eq. (35.10) can be interpreted and evaluated in two different ways. First, the AND could be evaluated as the product of x^1_i and x^2_j [7]. Thus,

$$x^1_i \text{ AND } x^2_j = \quad x^1_i x^2_j$$

The second method is by taking the minimum of the term's [7]. In this case the result is the minimum value of the membership values. Therefore,

$$x^1_i \text{ AND } x^2_j = \quad \min(x^1_i, x^2_j)$$

The most commonly used method in the literature is the product method. Therefore, the product method is used in this chapter to evaluate the AND function. Equation (35.10) can be expanded to multiple input with multiple fuzzy variables. In the most general case of n inputs and k linguistic qualifiers, we would have the rule R_i:

$$\mathrm{R}_i: \quad \text{IF } [x^1_i \text{ AND } x^2_i \text{ AND } \ldots \text{ AND } x^n_i] \text{ THEN } y = C_i \tag{35.11}$$

Recalling that all combinations of input vector components must be taken between fuzzy vector components the system designer could have up to $n^{\star}k$ rules. Where n is the number of fuzzy variables used to describe the inputs and k is the number of inputs. The number of rules used by the fuzzy inference engine could be reduced if the designer could eliminate some combinations of input conditions.

35.2.4 Defuzzification

The fuzzy inference engine as described previously often has multiple rules, each with possibly a different output. Defuzzification refers to the method employed to combine these many outputs into a single output. Using Eq. (35.11) where multiple inputs ($x^1\ x^2\ \ldots\ x^n$) should be evaluated, the product due to the evaluation of the premise conditions (defined by the components of the fuzzy vectors) determines the strength of the overall rule evaluation, w_i.

$$w_i = x_i^1 \text{ AND } x_i^2 \text{ AND } \ldots \text{AND } x_i^n$$
$$w_i = (x_i^1)(x_i^2)\ldots(x_i^n) \tag{35.12}$$

where, x_i^k is the membership value of the kth input into the ith fuzzy variable's category.

This value, w_i, becomes extremely important in defuzzification. Ultimately, defuzzification involves both the set of outputs C_i and the corresponding rule strength w_i.

There are a number of methods used for defuzzification, including the center of gravity (COG) and mean of maxima (MOM) [10]. The COG method otherwise known as the fuzzy centroid is denoted by y^{COG}.

$$y^{\text{COG}} = \frac{\sum_i w_i C_i}{\sum_i w_i} \tag{35.13}$$

where

$\mathrm{w}_i = \prod_k x_i^k$

C_i: is the corresponding output.

The mean of maxima method selects the outputs C_i that have the corresponding highest values of w_i.

$$y^{\text{MOM}} = \sum_{C_i \in G} C_i \Big/ Card(G) \tag{35.14}$$

where, G denotes a subset of C_i consisting of these values that have the maximum value of w_i. Out of the two methods of defuzzification, the most common method is the fuzzy centroid and is the one employed in this chapter.

35.3 Applications of Fuzzy Logic to Electric Drives

High performance drives requires that the shaft speed and the rotor position follow preselected tracks (trajectories) at all times [11, 12]. To accomplish this, two fuzzy control systems were designed and implemented. The goal of fuzzy control system is to replace an experienced human operator with a fuzzy rule-based system. The fuzzy logic controller provides an algorithm that converts the linguistic control maneuvering, based on expert knowledge, into an automatic control approach. In this section, a fuzzy logic controller (FLC) is proposed and applied to high performance speed and position tracking of a brushless DC (BLDC) motors. The proposed controller provides the high degree of accuracy required by high performance drives without the need for detailed mathematical models. A laboratory implementation of the fuzzy logic-tracking controller using the Motorola MC68HC11E9 microprocessor is described in this chapter. Additionally, in this experiment a bang–bang controller is compared to the fuzzy controller.

35.3.1 Fuzzy Logic-based Microprocessor Controller

The first step in designing a fuzzy controller is to decide which state variables representative of system dynamic performance can be taken as the input signals to the controller. Further, choosing the proper fuzzy variables, formulating the fuzzy control rules are also significant factors in the performance of the fuzzy control system. Empirical knowledge and engineering intuition play an important role in choosing fuzzy variables and their corresponding membership functions. The motor drive's state variables and their corresponding errors are usually used as the fuzzy controller's inputs including, rotor speed, rotor position, and rotor acceleration. After choosing proper linguistic variables as input and output of the fuzzy controller, it is required to decide on the fuzzy variables to be used. These variables transform the numerical values of the input of the fuzzy controller, to fuzzy quantities. The number of these fuzzy variables specifies the quality of the control, which can be achieved using the fuzzy controller. As the number of the fuzzy variable increases, the management of the rules is more involved and the tuning of the fuzzy controller is less straightforward. Accordingly, a compromise between the quality of control and computational time is required to choose the number of fuzzy variables. For the BLDC motor drive under study, two inputs are usually required. After specifying the fuzzy sets, it is required to determine the membership functions for these sets. Finally, the fuzzy logic control (FLC) is implemented by using a set of fuzzy decision rules. After the rules are evaluated, a fuzzy centroid is used to determine the fuzzy control output. Details of the design of the proposed controllers are given in the following sections.

35.3.2 Fuzzy Logic-based Speed Controller

In the case of shaft speed control to achieve optimal tracking performance, the motor speed error (ω_e) and the motor acceleration error (α_e) are used as inputs to the proposed controller. The controller output is the change in the motor voltage. For the fuzzy logic-based speed controller, the two inputs required

are defined as

$$\omega_e = \omega_{ref} - \omega_{act} \tag{35.15}$$

$$\alpha_e = \alpha_{ref} - \alpha_{act} \tag{35.16}$$

where

ω_{ref} = the desired speed (rad/s).
ω_{act} = the measured speed (rad/s).
$\alpha_{ref} = 0$, because we want to minimize the acceleration to zero.
α_{act} = the calculated acceleration in (rad/s^2).

For both the speed and acceleration errors, three regions of operation are established according to the fuzzy variables. These regions are positive error, zero, and negative error. The proposed controller uses these regions to determine the required motor voltage, which enables the motor speed to follow a desired reference trajectory. Examples of the broad fuzzy decisions are:

IF speed error is positive, THEN decrease the output.
IF speed error is zero, THEN maintain the output.
IF speed error is negative, THEN increase the output.
IF acceleration error is positive, THEN decrease the output.
IF acceleration error is zero, THEN maintain the output.
IF acceleration error is negative, THEN increase the output.

To achieve a sufficiently good quality of control, the three basic variables must be further refined. Thus the linguistic variable "acceleration error" has seven fuzzy variables: negative large (NL), negative medium (NM), negative small (NS), zero (Z), positive small (PS), positive medium (PM), and positive large (PL). The values associated with the fuzzy variables for the acceleration error are shown in Table 35.2.

The linguistic variable "speed error" has nine fuzzy variables: negative large (NL), negative medium (NM), negative medium small (NMS), negative small (NS), zero (Z), positive small (PS), positive medium (PM), positive medium small (PMS), and positive large (PL). Two fuzzy sets, namely, negative medium small (NMS) and positive medium small (PMS) are added to enhance the tracking performance. The regions defined for each fuzzy variable for the speed error is summarized in Table 35.3.

TABLE 35.2 Fuzzy variables for the acceleration error (α_e) for speed control

Fuzzy variable	Acceleration error (rad/s^2)
(NL)	$\alpha_e \le -738$
(NM)	$-738 < \alpha_e \le -369$
(NS)	$-369 < \alpha_e < 0$
(ZE)	$\alpha_e = 0$
(PS)	$0 < \alpha_e < 369$
(PM)	$369 \le \alpha_e < 738$
(PL)	$738 \le \alpha_e$

TABLE 35.3 Fuzzy variables for the speed error (ω_e) for speed control

Fuzzy variable	Speed error (rad/s)
(NL)	$\omega_e \le -209$
(NM)	$-209 < \omega_e \le -104$
(NMS)	$-104 < \omega_e \le -49$
(NS)	$-49 < \omega_e < 0$
(Z)	$\omega_e = 0$
(PS)	$0 < \omega_e < 49$
(PMS)	$49 \le \omega_e < 104$
(PM)	$104 \le \omega_e < 209$
(PL)	$209 \le \omega_e$

After specifying the fuzzy sets, it is required to determine the membership functions for these sets. The membership function for the fuzzy variable-representing ZERO is a fuzzy singleton. Additionally, the other membership functions are of the type described in Eq. (35.6) and are composed of fuzzy singletons within the region defined for each particular fuzzy variable. Figures 35.11 and 35.12 show the resulting membership function for the acceleration and speed errors, respectively.

The two fuzzy sets illustrated in Figs. 35.11 and 35.12 result in 63 linguistic rules for the BLDC drive system under study. The conditional rules listed in Table 35.4 are clearly implied, and the physical meanings of some rules are briefly explained as follows:

Rule 1: IF speed error is PL AND acceleration is PS, THEN change in control voltage (output of fuzzy controller) is PL. This rule implies a general condition when the measured speed is far from the desired reference speed. Accordingly, it requires a large increase in the control voltage to force the shaft speed to the desired reference speed quickly.

Rule 2: IF speed error is PS AND acceleration is ZERO, THEN change in control voltage is positive very very small PVVS. This rule implements the conditions when the error starts to decrease and the measured speed is approaching the desired reference speed. Consequently, a very small increase in the control voltage is applied.

Rule 3: IF speed error is ZERO AND acceleration is NS, THEN change in control voltage is PVVS. This rule deals with the circumstances when overshoot does occur. A very small decrease in the control voltage is required, which brings the motor speed to the desired reference speed.

These rules comprise the decision mechanism for the fuzzy speed controller. The decision table, Table 35.4, consists of values showing the different situations experienced by the drive system and the corresponding control input functions. It is clear that each entry in Table 35.4 represents a particular rule.

Now it is necessary to find the fuzzy output (change in control voltage). In this experiment the fuzzy centroid is used.

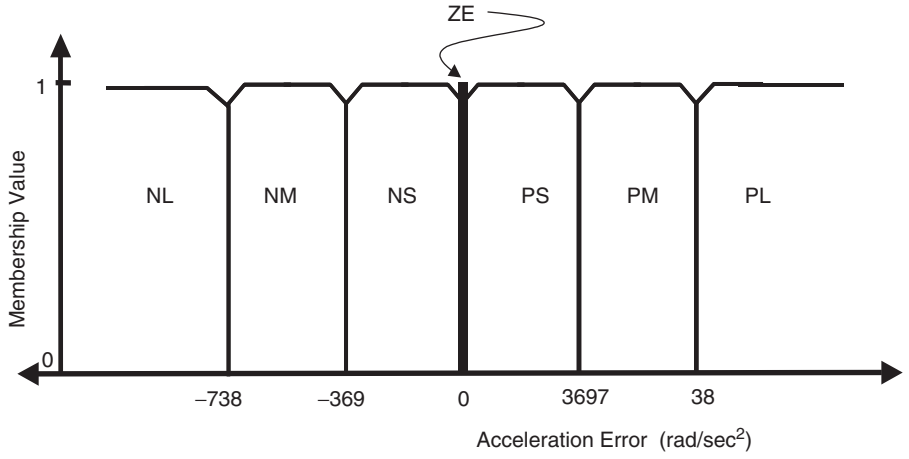

FIGURE 35.11 Membership functions for the acceleration error.

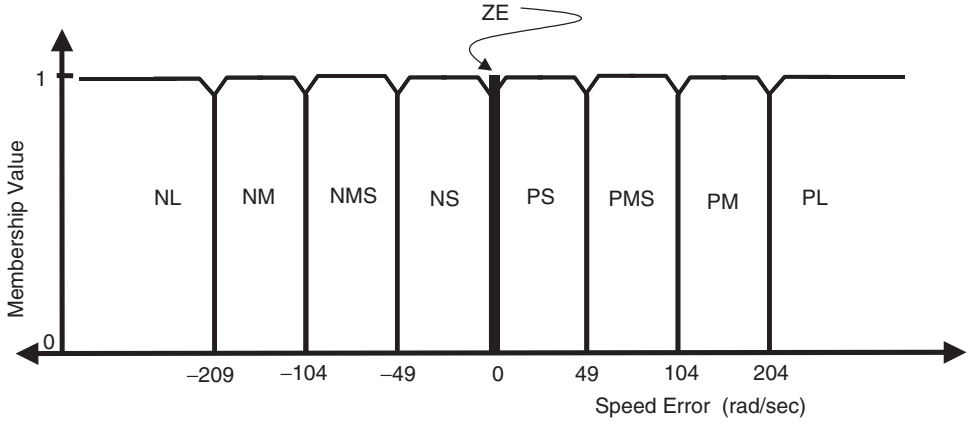

FIGURE 35.12 Membership functions for the speed error.

TABLE 35.4 Decision table for speed control

		Velocity error								
		NL	NM	NMS	NS	ZERO	PS	PMS	PM	PL
Acceleration error	PL	NS	NVVS	PVVS	PVS	PS	PMS	PM	PL	PVVL
	PM	NMS	NVS	ZERO	PVVS	PVS	PS	PMS	PML	PVL
	PS	NMS	NS	NVVS	ZERO	PVVS	PVS	PS	PM	PL
	ZERO	NML	NMS	NVS	NVVS	ZERO	PVVS	PVS	PMS	PML
	NS	NL	NM	NS	NVS	NVVS	ZERO	PVVS	PS	PML
	NM	NVL	NML	NMS	NS	NVS	NVVS	ZERO	PVS	PMS
	NL	NVVL	NL	NM	NMS	NS	NVS	NVVS	PVVS	PS

Equation (35.17) shows the fuzzy centroid used to compute the final output of the controller.

$$\text{Output} = \left(\frac{w_1 o_1 + w_2 o_2 + w_3 o_3 + \cdots + w_{63} o_{63}}{w_1 + w_2 + w_3 + \cdots + W_{63}} \right) \quad (35.17)$$

The weights w_i will be the strength of each particular rule's evaluation. The rule strength is evaluated as the product of the membership values associate with the speed and acceleration errors for the particular fuzzy variables involved in that rule. Since the membership functions were comprised solely of fuzzy singletons and the AND operator is evaluated as a product, the strength of each rules evaluation (w_i) will be either 0 or 1. Additionally, since the membership functions do not overlap, only one rule will be evaluated as true at each sample time. Thus all the weights w_i will be zero except one. So the actual implementation can be simplified.

35.3.3 Fuzzy Logic-based Position Controller

The task of the position control algorithm is to force the rotor position of the motor to follow a desired reference track without overshoot. The same method applied to the fuzzy speed controller is applied to the position controller. The inputs to the fuzzy position controller are, angular position error (θ_e) and motor speed error (ω_e). The output from the controller is

TABLE 35.5 Fuzzy variables of the speed error for position control

Fuzzy variable	Speed error (rad/s)
(NL)	$x^2 \leq -209$
(NM)	$-209 < x^2 \leq -104$
(NS)	$-104 < x^2 < 0$
(ZE)	$x^2 = 0$
(PS)	$0 < x^2 < 104$
(PM)	$104 \leq x^2 < 209$
(PL)	$209 \leq x^2$

TABLE 35.6 Fuzzy variables of position error (θ_e) for position control

Fuzzy variable	Position error (rad)
(NL)	$\theta_e \leq -2.21$
(NM)	$-2.21 < \theta_e \leq -1.05$
(NMS)	$-1.05 < \theta_e \leq -0.53$
(NS)	$-0.53 < \theta_e < 0$
(Z)	$\theta_e = 0$
(PS)	$0 < \theta_e < 0.53$
(PMS)	$0.53 \leq \theta_e < 1.05$
(PM)	$1.05 \leq \theta_e < 2.21$
(PL)	$2.21 \leq \theta_e$

a change in the motor voltage. The two inputs are defined as follows

$$\theta_e = \theta_{ref} - \theta_{act} \tag{35.18}$$

$$\omega_e = \omega_{ref} - \omega_{act} \tag{35.19}$$

where

θ_{ref} = the desired position in radians.
θ_{act} = the measured position in radians.
$\omega_{ref} = 0$, because we want to minimize the speed to zero.
ω_{act} = the measured speed in radians/second.

Nine fuzzy variables were defined for the position error and seven for the speed error. The fuzzy variables for the position error and the speed error are defined in Tables 35.5 and 35.6, respectively. After specifying the fuzzy sets for the position controller, the membership functions can be fully defined, it is required to determine the membership functions for these sets. The membership function for the fuzzy variable ZERO is a fuzzy singleton. Additionally, the other membership functions are of the type described in Eq. (35.6) and are composed of fuzzy singletons within the region defined for each particular fuzzy variable. Figures 35.13 and 35.14 show the resulting membership function for the speed and position errors respectively.

A corresponding output to the motor based on the speed and the rotor position error detected in each sampling interval must be assigned and thereby specifying the fuzzy inference engine. This is done by the fuzzy rules. For example, IF the position error is PL AND the speed is PS, THEN a large positive change in driving effort is used. The crisp (defuzzified) output signal of the fuzzy controller is obtained by calculating the centriod of all fuzzy output variables. Fuzzy rule-base, on which the required change in the motor voltage is generated, is illustrated in Table 35.7.

35.4 Hardware System Description

The BLDC drive system under control consists of the components illustrated in Fig. 35.15. The drive system is made up of several distinct subsystems: the motor, a personal computer (PC), the driving circuit, and the microprocessor evaluation board. The motor is 1/4 HP, 3000-rev/min BLDC motor, and was manufactured by Pittman Company. The BLDC motor is equipped with a hall-effect sensor and an incremental optical encoder. The hall-effect sensors detect the rotor position to indicate which of the three phases of the motor is to be excited as the motor spins. The optical position encoder with

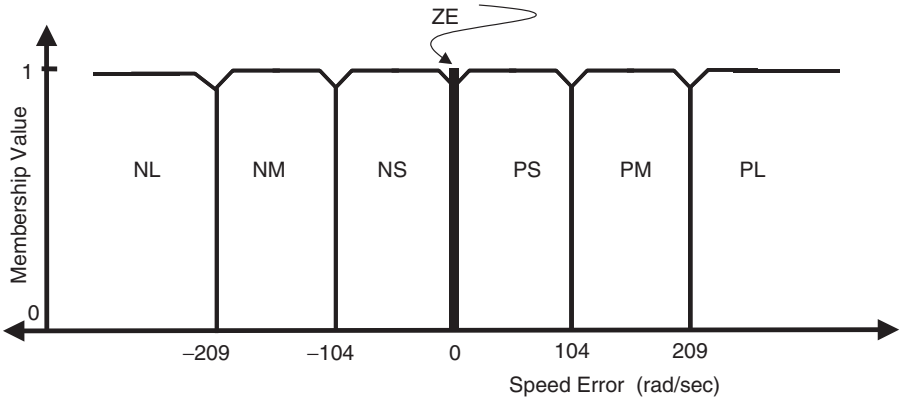

FIGURE 35.13 Membership functions for the speed error.

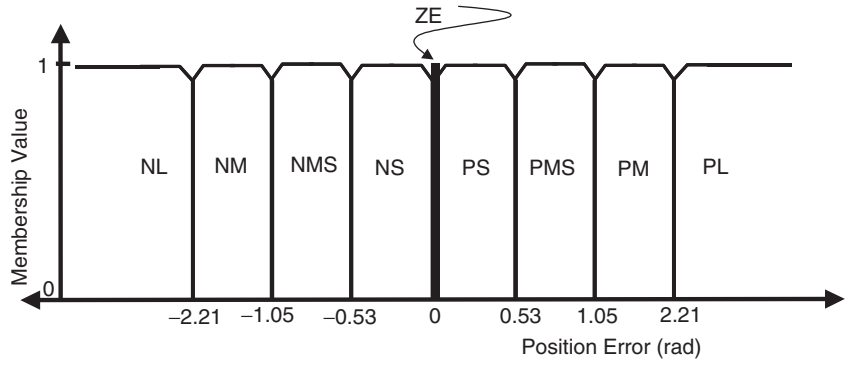

FIGURE 35.14 Membership functions for the position error.

TABLE 35.7 Decision table for position control

		Position error								
		NL	NM	NMS	NS	ZERO	PS	PMS	PM	PL
Speed error	PL	NS	NVVS	PVVS	PVS	PS	PMS	PM	PL	PVVL
	PM	NMS	NVS	ZERO	PVVS	PVS	PS	PMS	PML	PVL
	PS	NMS	NS	NVVS	ZERO	PVVS	PVS	PS	PM	PL
	ZERO	NML	NMS	NVS	NVVS	ZERO	PVVS	PVS	PMS	PML
	NS	NL	NM	NS	NVS	NVVS	ZERO	PVVS	PS	PML
	NM	NVL	NML	NMS	NS	NVS	NVVS	ZERO	PVS	PMS
	NL	NVVL	NL	NM	NMS	NS	NVS	NVVS	PVVS	PS

resolution of 500 pulses/revolution is used to give speed and position feedback.

The controller is implemented by software and executed using a microprocessor. The control algorithm is written and loaded into the microprocessor using the PC. The computer used is an IBM 486 PC. The driving circuit is constructed using two major components: (1) the integrated circuit (UDN2936W-120) designed for BLDC Drives [13] and (2) the digital to analog converter (DAC) and power amplifier. The inputs to the integrated circuit are the rotor position (which is obtained from hall-effect sensors), the direction of desired

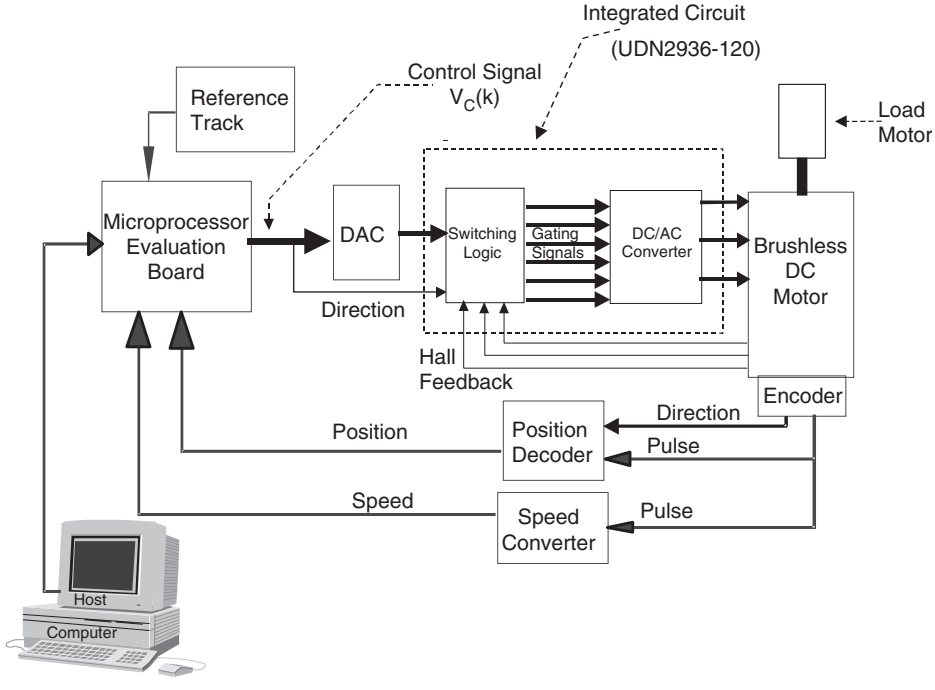

FIGURE 35.15 Block diagram of the laboratory setup.

rotation, and the magnitude of the control voltage. The output of the switching logic section is a sequence of gating signals that are pulse width modulated (PWM). These signals are used to drive the power converter portion of the chip. The power converter is a DC/AC inverter utilizing six MOSFETs. The output of the power converter is a chopped three-phase AC waveform.

The microprocessor used was the Motorola MC68HC11E9 microprocessor [14]. The on board memory system includes 8k bytes of read only memory (ROM), 512 bytes of electrically erasable programmable (ROM) (EEPROM), and 256 bytes of random access memory (RAM). The processor also has four eight bit parallel input/output ports, namely, A, B, C, and E. The program is completely contained in the microprocessor and the computer is only required to load new programs into the processor. Additionally, it has an internal analog-to-digital converter, and can accept up to four analog inputs. Figure 35.15 includes the integrated circuit (UDN2936W-120) designed for operating the BLDC motor. The range of the input voltage (control voltage) is between zero and 25 V. The motor speed for any control voltage less than 7 V is zero. This indicates that the motor requires a minimum of 7 V to start. This limitation is actually a protective feature of the circuit to prevent malfunctions [13]. Thus, the actual output of the microprocessor (the control signal $V_C(k)$) is added to a seven volt DC offset, using a summing amplifier. The current output of the summing amplifier must be first increased before it can be used to drive the motor. The current amplification is accomplished by using a 40 V power transistor (TIP31A). Additionally, when a sudden change in speed or direction occurs, the back EMF produced by the motor can, sometimes, cause large currents. Since the output stage of the integrated circuit (UDN2936W-120) has transistors with a current rating of ± 3 A, some limiting resistors were needed to prevent damage. Figure 35.16 shows a snapshot of the laboratory experiment.

Speed measurements were taken using a frequency to voltage converter LM2907. The LM2907 produces a voltage proportional to the frequency of the pulses it receives at its input. In this experiment, the pulses were sourced from the encoder signal-A. The circuit was constructed such that the maximum speed of the motor was corresponded to 5 V. This value was the input to the microprocessor, via the on chip A/D converter, as an 8-bit word. The direction of rotation was observed using a D-flip-flop. This direction signal was used to indicate the sign of the speed. The two signals, speed and direction, were combined within the software environment to produce an integer. Thus internally the speed could range from +256 to –256. These limits represent +3000 and –3000 R.P.M. respectively. Additionally, the digital output of the microprocessor

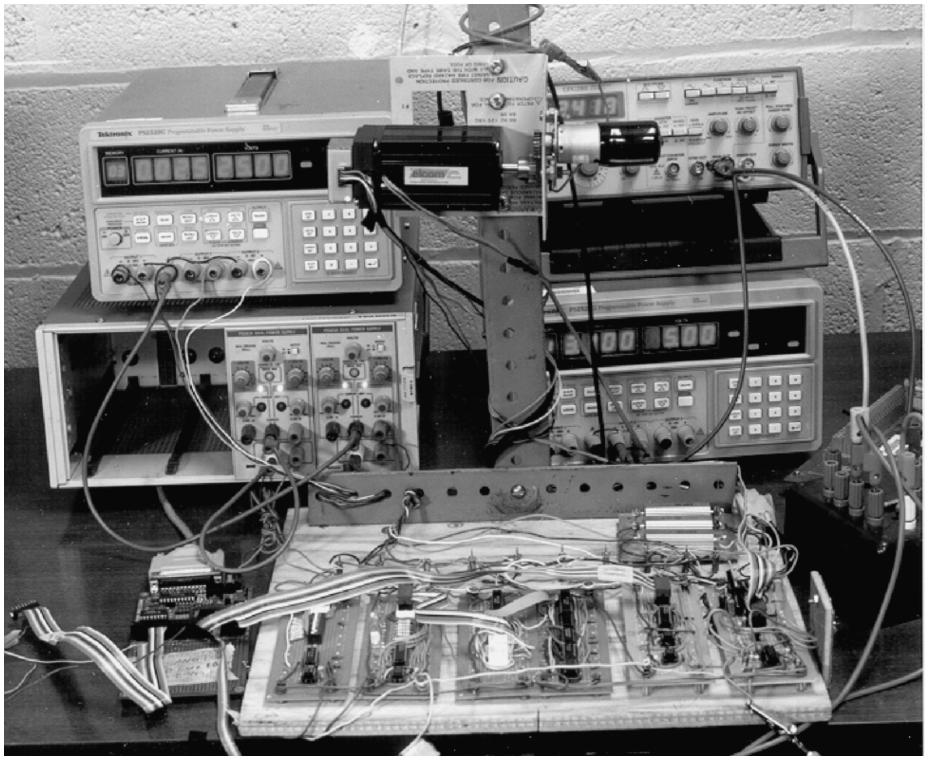

FIGURE 35.16 Photograph of the hardware implementation.

was limited to the resolution provided by seven bits. Therefore, if the desired output voltage was 22 V, then the maximum byte output was 127. Thus each increment in control signal, numerically within the microprocessor, would produce a change of 0.17 V.

35.4.1 Experimental Results

A square-wave followed by sinusoidal reference track was considered. In this experiment, there is a weight attached to the motor via a cable and pulley assembly. Figure 35.17 shows the speed tracking performance of the fuzzy logic controller. The motor is under constant load. The actual motor speed is superimposed on the desired reference speed in order to compare tracking accuracy. High tracking accuracy is observed at all speeds. The corresponding position tracking performance is displayed in Fig. 35.18. Reasonable position tracking accuracy is displayed. One can see from these figures that the results were very successful. For comparison purposes, Figs. 35.19 and 35.20 exhibit cases in which the fuzzy logic controller was replaced by a bang–bang controller. However, when bang–bang controller was applied, the controller could not maintain tracking accuracy. That is, the response using the bang–bang controller was unsatisfactory.

35.5 Conclusion

This chapter describes a fuzzy logic-based microprocessor controller, which incorporates attractive features such as

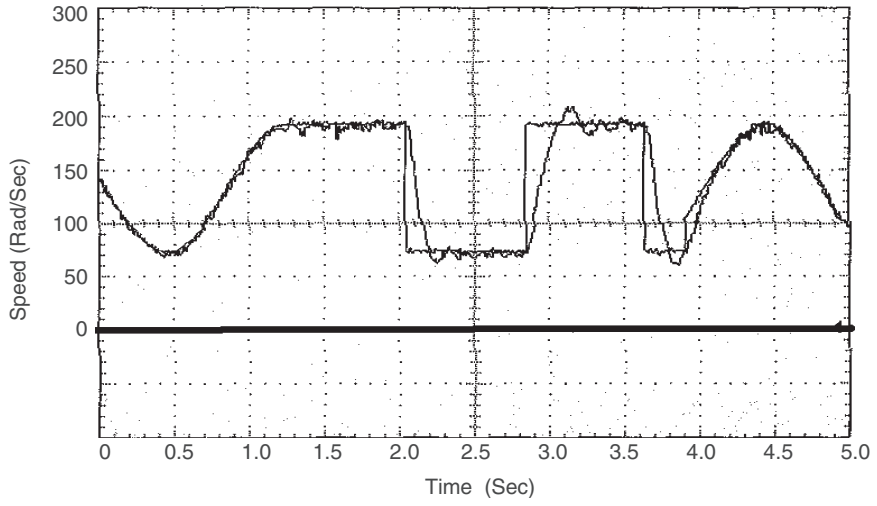

FIGURE 35.17 Fuzzy speed tracking under loading condition.

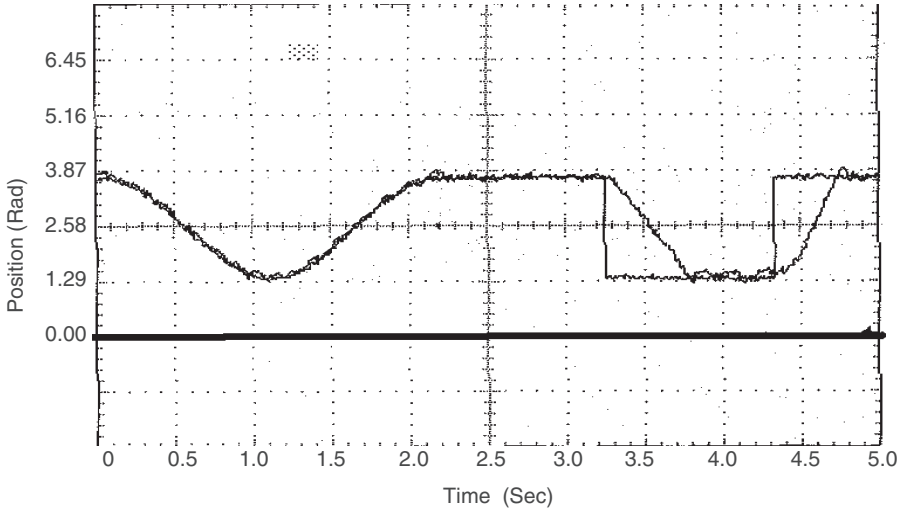

FIGURE 35.18 Fuzzy position tracking under loading condition.

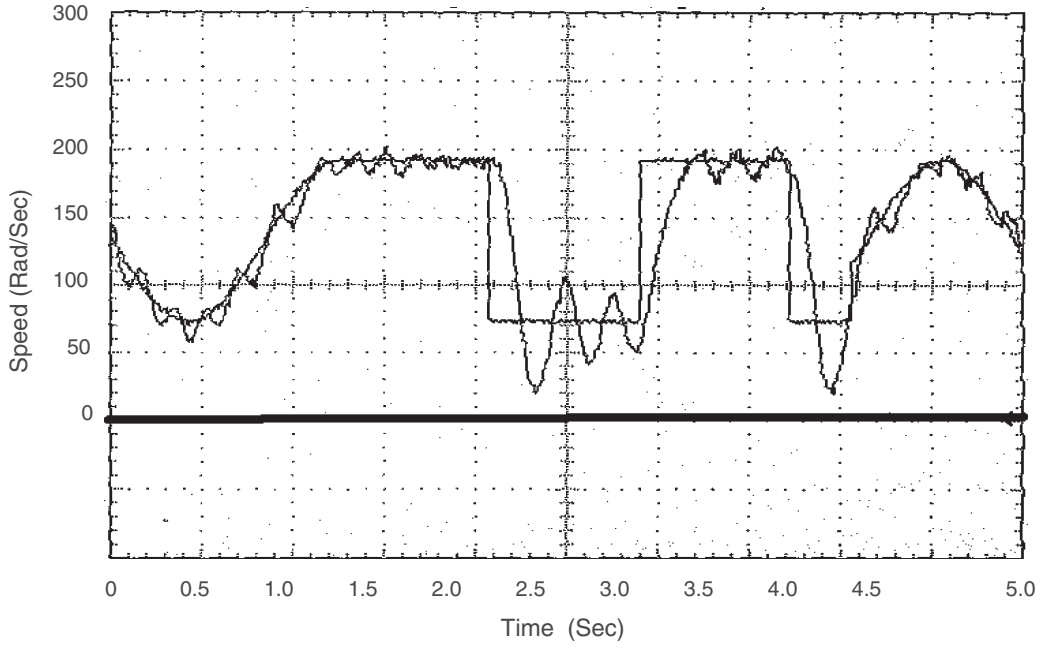

FIGURE 35.19 Bang–bang speed tracking under loading condition.

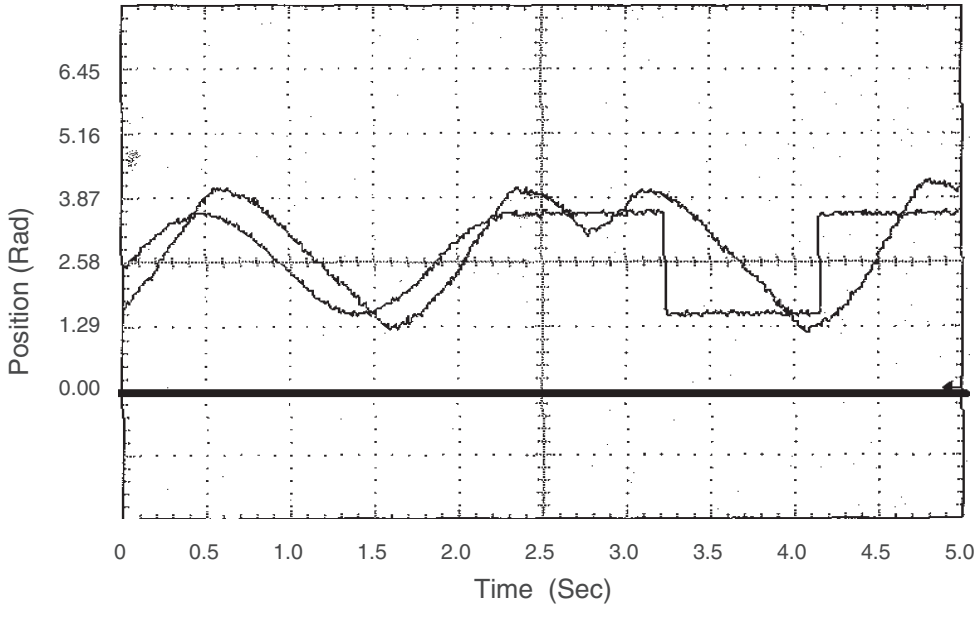

FIGURE 35.20 Bang–bang position tracking with load.

simplicity, good performance, and automation while utilizing a low cost hardware and software implementation. The test results indicate that the fuzzy logic-tracking controllers follow the trajectories successfully. Additionally, experimental results show that the effective smooth speed/position control and tracking of brushless DC (BLDC) motors can be achieved by the fuzzy logic controler (FLC), thus making it suitable for high performance motor drive applications. The controller design does not require explicit knowledge of the motor/load dynamics. This is a useful feature when dealing with parameter and load uncertainties. The advantage of using a fuzzy logic-tracking controller in this application is that it is well suited to the control of unknown or ill-defined non-linear dynamics.

Acknowledgments

The author would like to acknowledge the assistance of Mr. Daniel O. Ricketts, a graduate student at Howard University in clarifying these concepts and obtaining the experimental results. The author would also like to acknowledge the financial support by NASA Glenn Research Center, in the form of grant no. NAG3-2287 technically monitored by Mr. Donald F. Noga.

Further Reading

1. Ross, T. J., Fuzzy Logic with Engineering Applications, McGraw-Hill, 1995.
2. Mitra, S. and Pal. K. S., "Fuzzy Self-Organization, Inferencing, and Rule Generation," IEEE Trans. Syst., Man, Cybern.– Part A: Syst. Humans, Vol. 26, No. 5, pp. 608–619, September 1996.
3. Jamshidi, M., Vadiee, N. and Ross, T. Fuzzy Logic and Control, Prentice Hall, 1993.
4. Lee, C. C., "Fuzzy Logic in Control Systems: Fuzzy Logic Controller. Part I," IEEE Trans. Syst., Man, Cybern, Vol. 20, No. 2, pp. 404–418, March/April 1990.
5. Lee, J., "On Methods for Improving Performance of PI-Type Fuzzy Logic Controllers," IEEE Trans. Fuzzy Systems, Vol. 1, No. 4, November 1993.
6.
7. Lofti, A. and Tsoi, A. C. "Learning Fuzzy Inference Systems Using and Adaptive Membership Function Scheme," IEEE Trans. Syst., Man, Cybern., Vol. 26, No. 2, pp. 326–331, April 1996.
8. Lofti, A., Andersen, H. C., and Tsoi, A. C. "Matrix Formulation of Fuzzy Rule-Based Systems," IEEE Trans. Syst., Man, Cybern., Vol. 26 No. 2, pp. 332–339, April 1996.
9. Motorola, MC68HC11E9 Technical Data Manual, 1990.
10. Rubaai, A., Kotaru, R. and David Kankam, M. "A Continually Online-Trained Neural Network Controller for Brushless DC Motor Drives," IEEE Trans. Ind. Appl., Vol. 36, No. 2, pp. 475–483, March/April 2000.
11. Ross, T. J., Fuzzy Logic with Engineering Applications, McGraw-Hill, 1995.
12. Rubaai, A. and Kotaru, R. "Neural Net-Based Robust Controller Design for Brushless DC Motor Drives," IEEE Trans. Syst., Man, Cybern., Part C: Applications and Review, Vol. 29, No. 3, pp. 460–474, August 1999.
13. Allegro, UDN2936W-120 Technical Data Manual, 1993.
14. Marks II, R. J. (Editor), Fuzzy Logic Technology and Applications, IEEE Press, 1994.
15. Zadeh, L. "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes," IEEE Trans. Syst., Man, Cybern., Vol. 3, No. 1, pp. 28–44, January 1973.