

10-1-1992

# Novel Approach to Fuzzy Logic Controller Design for Systems With Deadzones

Jong-Hwan Kim

*Purdue University School of Electrical Engineering*

Jong-Hwan Park

*Korea Advanced Institute of Science and Technology. Department of Electrical Engineering,*

Seon-Woo Lee

*Korea Advanced Institute of Science and Technology. Department of Electrical Engineering,*

Edwin K. P. Chong

*Purdue University School of Electrical Engineering*

Follow this and additional works at: <http://docs.lib.purdue.edu/ecetr>

---

Kim, Jong-Hwan; Park, Jong-Hwan; Lee, Seon-Woo; and Chong, Edwin K. P., "Novel Approach to Fuzzy Logic Controller Design for Systems With Deadzones" (1992). *ECE Technical Reports*. Paper 271.

<http://docs.lib.purdue.edu/ecetr/271>

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact [epubs@purdue.edu](mailto:epubs@purdue.edu) for additional information.

A NOVEL APPROACH TO FUZZY  
LOGIC CONTROLLER DESIGN FOR  
SYSTEMS WITH DEADZONES

JONG-HWAN KIM  
JONG-HWAN PARK  
SEON-WOO LEE  
EDWIN K. P. CHONG

TR-EE 92-44  
OCTOBER 1992



SCHOOL OF ELECTRICAL ENGINEERING  
PURDUE UNIVERSITY  
WEST LAFAYETTE, INDIANA 47907-1285

# A Novel Approach to Fuzzy Logic Controller Design for Systems With Deadzones

Jong-Hwan Kim\*    Jong-Hwan Park\*    Seon-Woo Lee\*  
Edwin K. P. Chong<sup>†</sup>

## Abstract

Existing fuzzy control methods do not perform well when applied to systems containing nonlinearities arising from unknown deadzones. In particular, we show that a conventional fuzzy logic controller applied to a system with a **deadzone** suffers from poor transient performance and a large steady-state error. In this report, we propose a **novel** two-layered fuzzy logic controller for controlling systems with deadzones. The **two-layered** control structure consists of a fuzzy logic-based precompensator followed by a conventional fuzzy logic controller. Our proposed controller exhibits superior transient and steady-state performance compared to conventional **fuzzy** controllers. In addition, the controller is robust to variations in **deadzone nonlinearities**. We **illustrate** the effectiveness of our scheme using computer simulation **examples**.

---

\*Dept. of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), 373-1 Kusung-dong, Yuseong-gu, Taejeon-shi 305-701, Republic of Korea. The first author is currently on sabbatical at Purdue University.

<sup>†</sup>School of Electrical Engineering, Purdue University, 1285 Electrical Engineering Bldg., West Lafayette, IN 47907-1285.

# I Introduction

We propose a two-layered fuzzy logic-based controller for controlling systems with deadzones. Our two-layered structure consists of a fuzzy precompensator and a fuzzy controller. The two-layered structure is based on analyzing the source of large **steady-state** errors which arise when a conventional fuzzy controller is applied to a system with a deadzone. Our proposed scheme has good transient as well as steady-state **performance**, and is robust to variations in **deadzone** nonlinearities.

Many physical components in control systems contain nonsmooth nonlinearities, such as saturation, relays, hysteresis, and deadzones. Such nonlinearities are especially **common** in actuators used in practice, such as hydraulic servovalves. Furthermore, the nonlinearities in such systems are often unknown and vary with time. For example, a common source of nonlinearities arise from friction, which vary **with** temperature and **wear**, and may differ significantly between components which are mass produced. **Therefore** the study of methods to deal with nonsmooth **nonlinearities** has been of interest to control practitioners for some time. In this report, we consider only **deadzone** nonlinearities. Deadzones are of interest in their own right, and provide good models for many nonsmooth nonlinearities found in practice.

Several classical methods exist for controlling systems with nonsmooth **nonlinearities**, including sliding mode control [1], and dithering [2]. Motivated by limitations in these methods, such as chattering in sliding mode control, **Recker** et al. [3] proposed an adaptive control scheme for controlling systems with **deadzones**. In [3], full state **measurements** were assume to be available. More recently, **Tao and** Kokotovic [4] **considered** the more realistic situation where only a single output measurement is available. In practice, however, the transient performance of the adaptive control schemes above is limited.

Fuzzy logic-based controllers have received considerable interest in recent years (see for example [5], [6], [7], [8], [9]). Fuzzy-based methods are useful when precise mathematical formulations are infeasible. Moreover, fuzzy logic controllers often yield

superior results to conventional control approaches [7]. However, direct application of **conventional** fuzzy controllers to a system with deadzones results in **poor** transient and steady-state behavior, as we shall see in the next section. In particular, a steady-state error occurs when using a conventional fuzzy controller to a system with **deadzones**—the **size** of the steady-state error increases with the **deadzone** width. The steady-state error **arises** because conventional fuzzy controllers use only the output error and the **change** in output error as inputs to the controller. To eliminate the steady-state error, we **may** attempt to use a fuzzy controller **that** also incorporates the "integral" of the output error as an input to the controller. Such a controller was **considered** in [8]. However, even though the steady-state error is eliminated when applied to a system with deadzones, the transient performance is not satisfactory, as we shall see later.

In this report we propose a fuzzy logic-based scheme which does not suffer from the **deficiencies** mentioned above of conventional fuzzy controllers applied to systems with deadzones. The idea underlying our approach is based on analyzing the source of the **steady-state** error resulting in using a conventional fuzzy controller. Our control **scheme** consists of two "layers": a fuzzy precompensator, and a conventional fuzzy **controller**. We demonstrate that our controller has good transient as well as **steady-state** performance, and is robust to variations in **deadzone** nonlinearities.

The remainder of this report is organized as follows. In **Section II** we describe a system with a deadzone, and study the characteristics of a conventional fuzzy logic controller applied to the system. We show that the conventional **fuzzy** controller results in poor performance, and give an analysis of the source of steady-state errors. We **also** study the behavior of PID and fuzzy PID controllers. In **Section III** we propose our two-layered fuzzy logic controller. We describe the idea underlying our approach, and give a precise description of the controller. We also **provide** simulation plots to illustrate the behavior of our scheme. Finally we conclude in **Section IV**.

## II Characteristics of Conventional FLC

In this section we describe a conventional fuzzy logic controller (FLC), and study the behavior of the FLC applied to a system with a deadzone.

### II.1 Basic Control Structure

We consider the (discrete-time) system shown in Figure 1, which is a conventional FLC control system [8]. The transfer function  $P(z)$  represents the plant,  $D$  represents an actuator with deadzone,  $F[e(k), \Delta e(k)]$  represents a FLC control law,  $K_1$  is the feedforward gain,  $v(k)$  is the output of the controller,  $u(k)$  is the output of the actuator,  $y_m(k)$  is the reference input (command signal to be followed), and  $y_p(k)$  is the output of the plant. The characteristics of the actuator with deadzone  $D$  is described by the function

$$D[v] = \begin{cases} m(u - d), & \text{if } u > d \\ 0, & \text{if } -d \leq v \leq d \\ m(v + d), & \text{if } v < -d \end{cases}$$

where  $d, m > 0$ . Figure 2 illustrates the characteristics of the actuator with deadzone. The parameter  $2d$  specifies the width of the deadzone, while  $m$  represents the slope of the response outside the deadzone.

### 11.2 Fuzzy Logic Controller

We describe the FLC control law  $F[e(k), \Delta e(k)]$  as follows. The approach is based on standard fuzzy logic rules—for details on fuzzy logic controllers we refer the reader to [7]. We think of  $e(k)$  and  $\Delta e(k)$  as inputs to the controller, and  $F[e(k), \Delta e(k)]$  as the output. As we shall see later,  $e(k)$  is the output error  $y_m(k) - y_p(k)$ , and  $\Delta e(k)$  is the change in output error  $e(k) - e(k - 1)$ . Associated with the fuzzy control law is a collection of *linguistic values*

$$L = \{ \text{NB, NM, NS, ZO, PS, PM, PB} \}$$

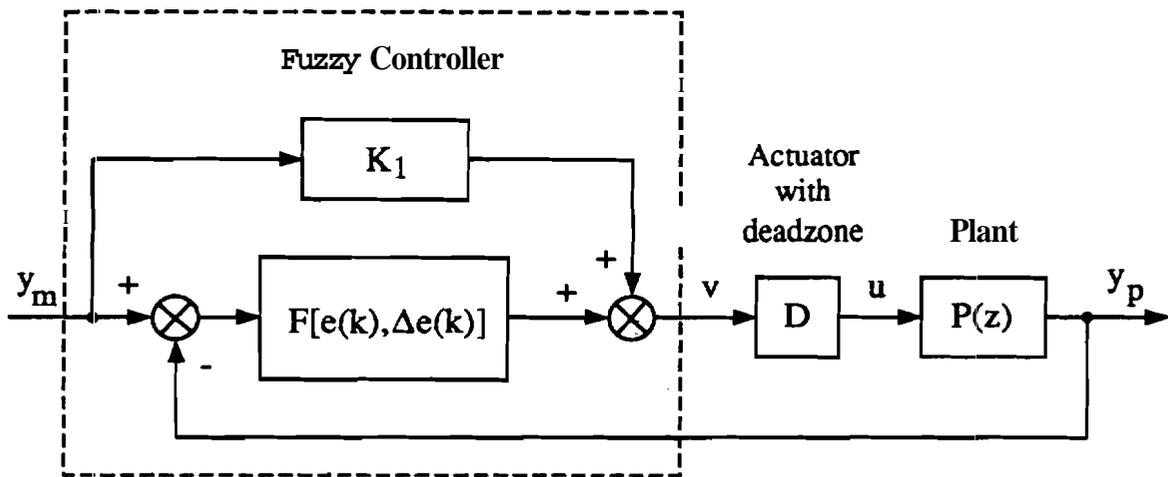


Figure 1: Conventional FLC system with deadzone

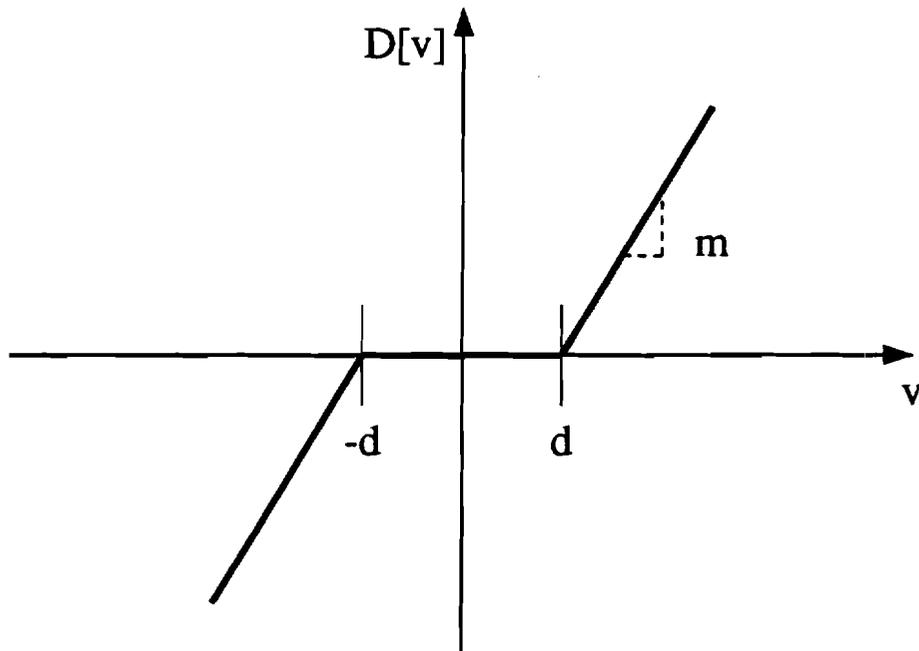


Figure 2: Characteristics of Actuator with deadzone

and a collection of membership *functions*

$$M = \{M_{NB}, M_{NM}, M_{NS}, M_{ZO}, M_{PS}, M_{PM}, M_{PB}\}$$

Each membership function is a map from the real line to the interval  $[0, 1]$ ; Figure 3 shows a plot of the membership functions. The "meaning" of each linguistic value should be clear from its mnemonic; for example, NB stands for "negative-big", NM stands for "negative-medium", NS stands for "negative-small", ZO stands for "zero", and likewise for the "positive" (P)linguistic-value.

The fuzzy control law consists of three stages: fuzzification, decision making fuzzy logic, and defuzzification. The process of fuzzification transforms the inputs  $e(k)$  and  $\Delta e(k)$  into the setting of linguistic values. Specifically, for each linguistic value  $l \in L$ , we assign a pair of numbers  $n_e(l)$  and  $n_{\Delta e}(l)$  to the inputs  $e(k)$  and  $\Delta e(k)$  via the associated membership function  $M_l$ , by

$$\begin{aligned} n_e(l) &= M_l(C_e e(k)) \\ n_{\Delta e}(l) &= M_l(C_{\Delta e} \Delta e(k)) \end{aligned}$$

where  $C_e$  and  $C_{\Delta e}$  are scale factors. The numbers  $n_e(l)$  and  $n_{\Delta e}(l)$ ,  $l \in L$ , are used in the fuzzy logic decision process, which we describe next.

Associated with the fuzzy logic decision process is a set of *fuzzy* rules  $R = \{R_1, R_2, \dots, R_r\}$ . Each  $R_i$ ,  $i = 1, \dots, r$ , is a triplet  $(l_e, l_{\Delta e}, l_w)$ , where  $l_e, l_{\Delta e}, l_w \in L$ . An example of a rule is the triplet  $(NS, PS, ZO)$ . Rules are often written in the form: "if  $e(k)$  is  $l_e$  and  $\Delta e(k)$  is  $l_{\Delta e}$ , then  $w$  is  $l_w$ " (here we think of  $w$  as the output of the fuzzy logic rule). For this conventional FLC, the rules are given in Table 1. This set of rules is fairly standard and well known; see for example [9]. In this case,  $r = 21$ , but in general we may have more or fewer number of rules. As is usual in fuzzy logic approaches, the rules were constructed based on expert experience. Each rule  $R_i = (l_e, l_{\Delta e}, l_w)$  takes a given pair  $e(k)$  and  $\Delta e(k)$  and assigns to it a function  $p_i(e(k), \Delta e(k), w)$ ,  $w \in [-1, 1]$ , as follows:

$$\begin{aligned} N_{min} &= \min(n_e(l_e), n_{\Delta e}(l_{\Delta e})) \\ p_i(e(k), \Delta e(k), w) &= \min(N_{min}, M_{l_w}(w)) \end{aligned}$$

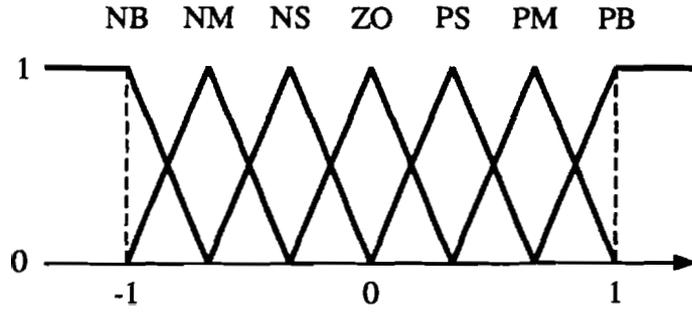


Figure 3: Membership Functions

We combine the functions  $p_i$ ,  $i = 1, \dots, r$  to get an overall function  $q$  by

$$q(e(k), \Delta e(k), w) = \max(p_1(e(k), \Delta e(k), w), \dots, p_r(e(k), \Delta e(k), w))$$

Finally, the defuzzification process maps the result of the fuzzy logic rule stage to a real number output  $F(e(k), \Delta e(k))$  by

$$F[e(k), \Delta e(k)] = C_F \frac{\int_{-1}^1 w q(e(k), \Delta e(k), w) dw}{\int_{-1}^1 q(e(k), \Delta e(k), w) dw}$$

where  $C_F$  is a scale factor. This method of defuzzification is called the Center of Area (COA) method, since the ratio in the right hand side of the above equation is simply the center of area of the function  $q(e(k), \Delta e(k), w)$  (as a function of  $w$ ).

### 11.3 Analysis of Steady-State System Behavior

We now study the steady-state behavior of the system controlled by the conventional FLC described in the previous section. We will show that in the presence of a **deadzone**, a steady-state error occurs.

The dynamics of overall system is described by the following equations:

$$\begin{aligned} e(k) &= y_m(k) - y_p(k) \\ \Delta e(k) &= e(k) - e(k-1) \\ v(k) &= K_1 y_m(k) + F[e(k), \Delta e(k)] \end{aligned}$$

		$e(k)$						
		NB	NM	NS	ZO	PS	PM	PB
$\Delta e(k)$	NB				NB	NS		
	NM				NM	NS		
	NS				NS	ZO		PM
	ZO	NB	NM	NS	ZO	PS	PM	PB
	PS	NM	NS	ZO	PS			
	PM				PM			
	PB			PM	PB			

Table 1: Fuzzy logic rules for conventional FLC

$$u(k) = D[v(k)]$$

$$y_p(k) = P(z)[u(k)]$$

Note that the equation  $y_p(k) = P(z)[u(k)]$  involves a slight abuse of notation; however, its **meaning** should be obvious. It turns out that  $F[0,0] = 0$ , and therefore if we **fix** the reference input  $y_m(k) = y_m$ , the steady-state actuator input is  $K_1 y_m$ .

Consider the case where there is no deadzone, i.e.,  $d = 0$ , and  $m = 1$ . In this case the plant output can be written as

$$y_p(k) = P(z)[K_1 y_m(k) + F[e(k), \Delta e(k)]]$$

Since  $e(k) = y_m(k) - y_p(k)$ , then the plant output can also be written as

$$y_p(k) = y_m(k) - e(k)$$

We **now fix**  $y_m(k) = y_m$ , and study the behavior of the system in steady-state. In this case, we can set  $\Delta e(k) = 0$  to get

$$y_p(k) = K_s [K_1 y_m + F[e(k), 0]] = y_m - e(k) \quad (1)$$

where  $K_s$  is the steady-state gain of  $P(z)$  (assumed stable), given by  $K_s = \lim_{z \rightarrow 1} (1 - z^{-1})P(z)$ . The steady-state error  $e_{ss}$  is then the solution to equation (1), that is,

$$K_s[K_1 y_m + F[e_{ss}, 0]] = y_m - e_{ss} \quad (2)$$

We **assume** that the controller is "well-tuned", so that  $K_1 = K_s^{-1}$ . Equation (2) then becomes

$$K_s F[e_{ss}, 0] = -e_{ss} \quad (3)$$

We do not have a closed form expression for the function  $F[\cdot, 0]$ . Nevertheless, it is **easy** to see from the description of the FLC in the previous section that  $F[\cdot, 0]$  is an **increasing odd** function, **as** illustrated in Figure 4. The graph of  $K_s F[\cdot, 0]$  in **Figure 4** was obtained by direct calculation via computer. We can solve equation (3) graphically—we simply plot the left and right hand sides of equation (1) on the same **graph**, **and** find the point where they intersect. As can be seen in Figure 4, the solution is  $e_{ss} = 0$ . Therefore, the steady-state error for a system without a **deadzone** is exactly zero.

We now consider the case where a **deadzone** is present, i.e.,  $d \neq 0$ , and  $m > 0$  is arbitrary. In this case, the steady-state output of the plant can be written as

$$y_p(k) = K_s D[K_1 y_m + F[e(k), 0]] = y_m - e(k)$$

Therefore, the steady-state error is the solution to the equation

$$K_s D[K_1 y_m + F[e(k), 0]] - y_m = -e_{ss} \quad (4)$$

The first term in the left hand side of (4) is illustrated in Figure 5(a). Once again we use a graphical approach to solve (4); see Figure 5(b). As we can see, the solution  $e_{ss}$  is no longer zero, but some nonzero number (with the same sign as  $y_m$ ; in Figure 5(b) we **have** assumed a positive  $y_m$ ). It is clear that the nonzero steady-state error is a direct result of the presence of the **deadzone** in the actuator. In the next section we illustrate this behavior via an example.

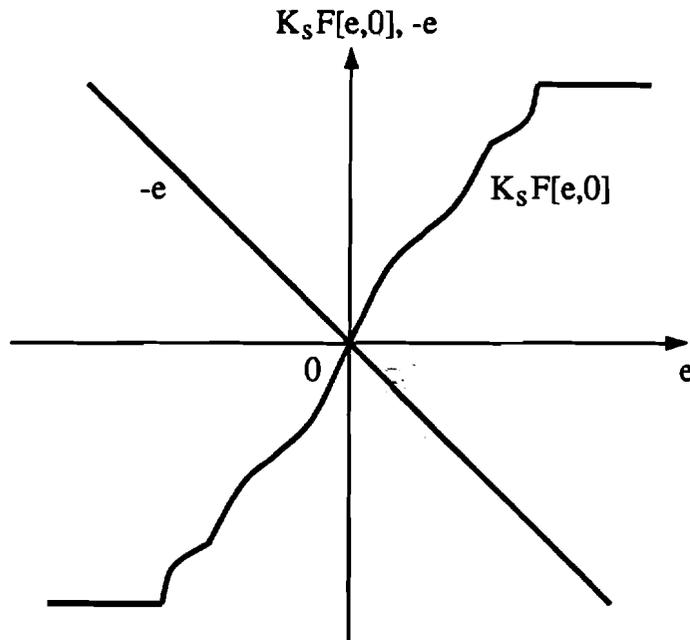


Figure 4: Graph of  $K_s F[e,0]$  and  $-e$

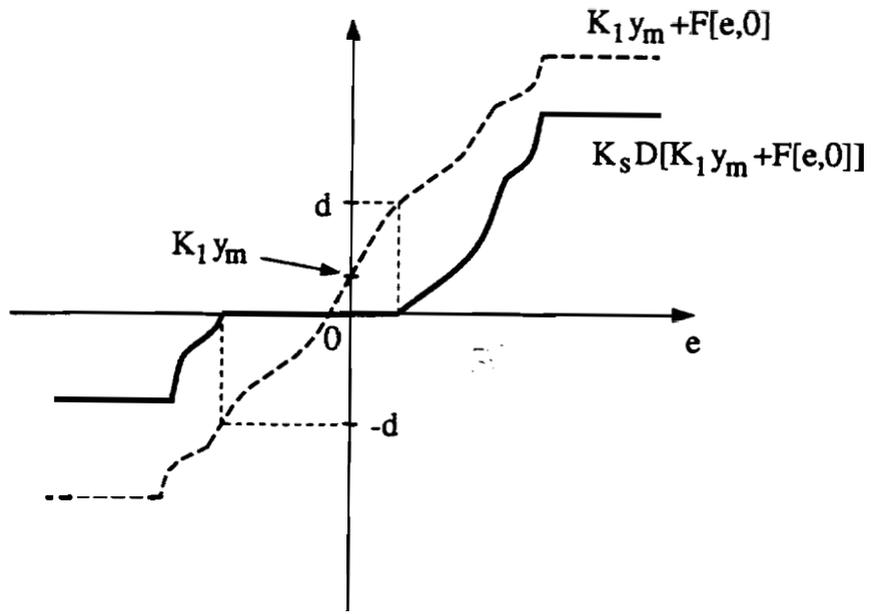
## 11.4 An Example

Consider a (continuous time) plant with transfer function

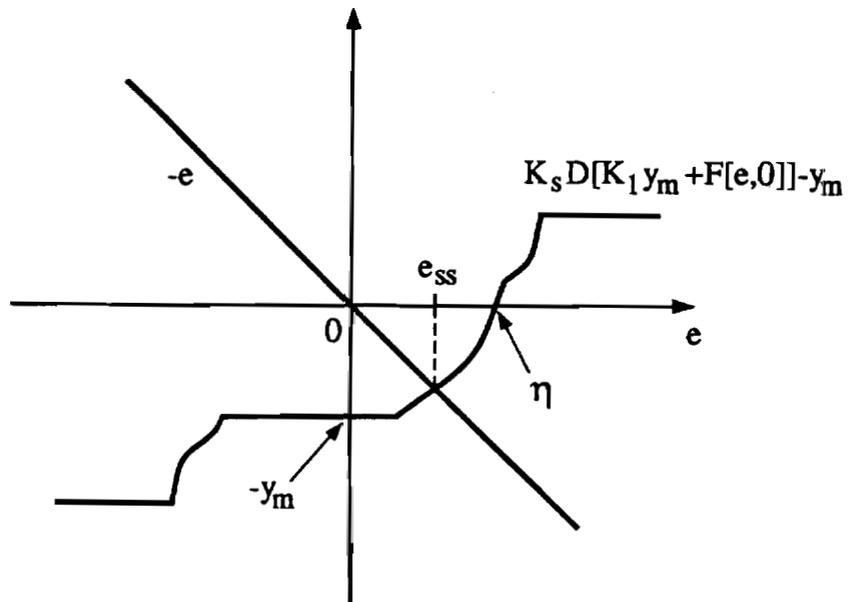
$$\frac{10}{s^2 + s + 1}$$

We apply the conventional FLC described before to the above plant, using the standard sample-and-hold approach, with a sampling time of 0.025 seconds. The scale factors used for the FLC are  $C_e = 1/y_m$ ,  $C_{\Delta e} = 9/y_m$ , and  $C_F = 5y_m$ . These values for the scale factors were chosen by experience. In this example, we set  $y_m = 1$ , and  $K_1 = 0.1$ .

Figure 6 shows output responses of the plant for three values of  $d$ : 0.0, 0.5, 1.0. In all cases we used  $m = 1$ . It is clear from Figure 6 that there is a relatively large steady-state error and overshoot when a deadzone is present. The steady-state error and overshoot increases with the the deadzone width.



(a)



(b)

Figure 5: Graphs of: (a)  $K_s D[K_1 y_m + F[e, 0]]$ ; (b)  $K_s D[K_1 y_m + F[e, 0]] - y_m$  and  $-e$

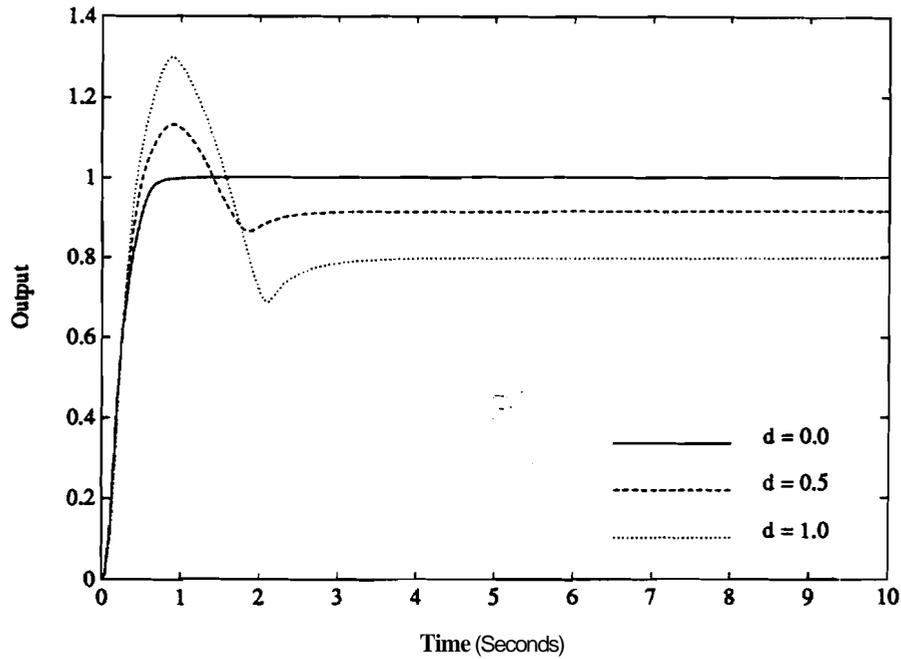


Figure 6: Output responses of plant with conventional FLC

## 11.5 PID and Fuzzy PID Controllers

We may argue that a steady-state error exists in the previous system because the controller uses only the output error and change of output error. It is well known that if we also include the "integral" of the error as an input to the controller, then steady-state errors can be eliminated. In this section we study the behavior of a PID controller and a fuzzy PID controller applied to the system with a deadzone. These controllers include not only the error and change of error, but also "integral" of error, as input.

Consider the control structure shown in Figure 7, which consists of a conventional PID ("proportional-integral-derivative") controller applied to the system with deadzone. The control law used is given by:

$$u(k) = u(k-1) + K_P \Delta e(k) + K_I e(k) + K_D (\Delta e(k) - \Delta e(k-1))$$

The above is the standard PID controller law, used widely in practice.

To observe the behavior of the system in Figure 7, we used the plant given in the

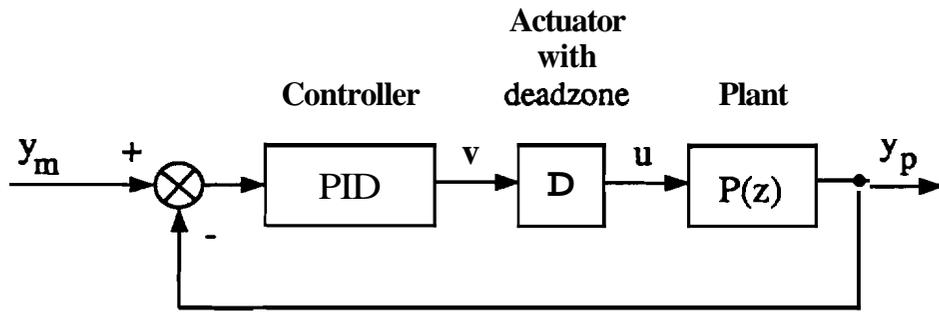


Figure 7: PID controller for-system with deadzone

previous example, with the following parameter values:  $K_P = 1.284$ ,  $K_I = 0.0325$ , and  $K_D = 46.8$ . As before, we used a sampling time of 0.025 seconds. The output responses are shown in Figure 8. As we can see, the steady-state error is eliminated. However, the transient response is sensitive to the **deadzone** width, and is increasingly poor as the **deadzone** width is increased.

We now consider a fuzzy-based scheme which is similar to the one considered in the last section, but which incorporates the "integral" of error as an input to the controller. We refer to the controller as a "fuzzy PID" controller. The scheme is discussed in detail in [8], and is illustrated in Figure 9. The only difference between the fuzzy PID scheme and the conventional FLC considered previously is that  $K_I = 0$ , and there is a "Fuzzy I" block in parallel with the "Fuzzy PD" block. The Fuzzy PD block is essentially identical to the conventional FLC described before (the only difference is in the set of rules used—49 rules were used here, these being taken from [8, Table 10]). The Fuzzy I block uses  $e(k)$  as the input. We refer the reader to [8, Table 7] for the fuzzy rules used in the Fuzzy I block. The **fuzzification** and **defuzzification** procedures used in the Fuzzy I block are the same as before, except with different scale factors—we denote the input scale factor by  $C_{ei}$  and the output scale factor by  $C_I$ .

We applied the Fuzzy PID controller to the same system as the previous example. We used the following internal variables:  $C_e = 1/y_m$ ,  $C_{\Delta e} = 11/y_m$ ,  $C_F = 8y_m$ ,

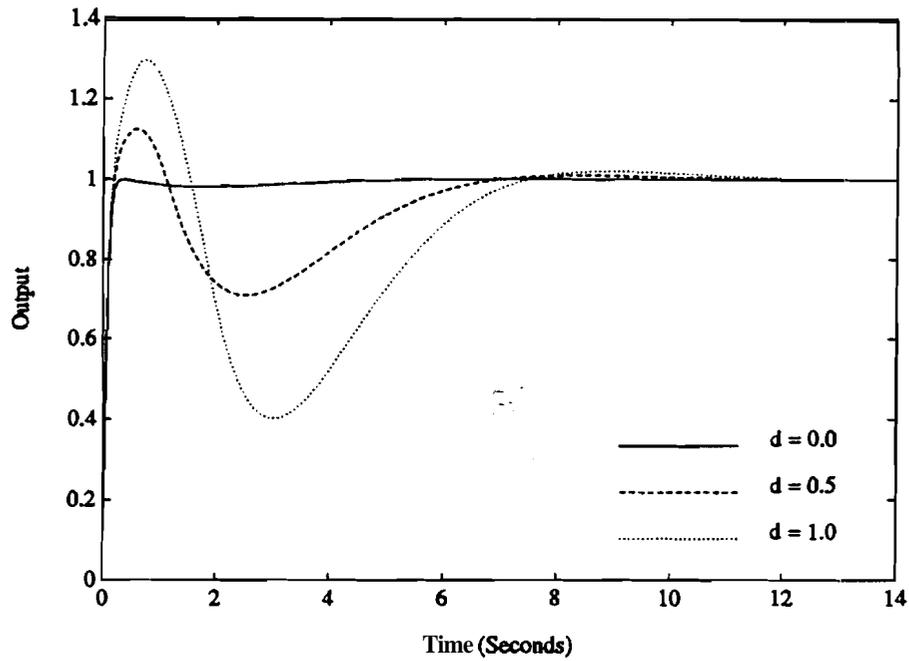


Figure 8: Output responses of plant with PID controller

$C_{ei} = 8/y_m$ ,  $C_I = 0.02y_m$ . As before,  $y_m = 1$ . Figure 10 shows output responses for the system with the Fuzzy PID controller. We see that the **steady-state error is eliminated**, but the transient performance with large **deadzone width** is still not **satisfactory**.

### III Two-Layered Fuzzy Logic Controller

In **this** section we describe a novel **two-layered** fuzzy logic controller. Our aim is to eliminate the steady-state error and improve the performance of the output response for FLC systems with deadzones. As we shall see, our proposed scheme is indeed insensitive to deadzones, and exhibits good transient and **steady-state** behavior.

#### 1 1 1 Basic Control Structure

We use a graphical approach to describe the idea underlying our proposed controller. Consider Figure 5(b), which illustrates the source of the steady-state error for the

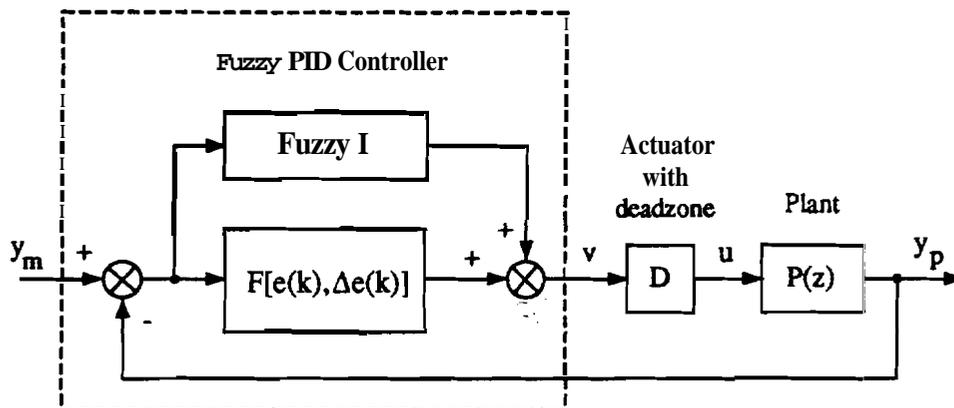


Figure 9: Fuzzy **PID** controller for system with deadzone

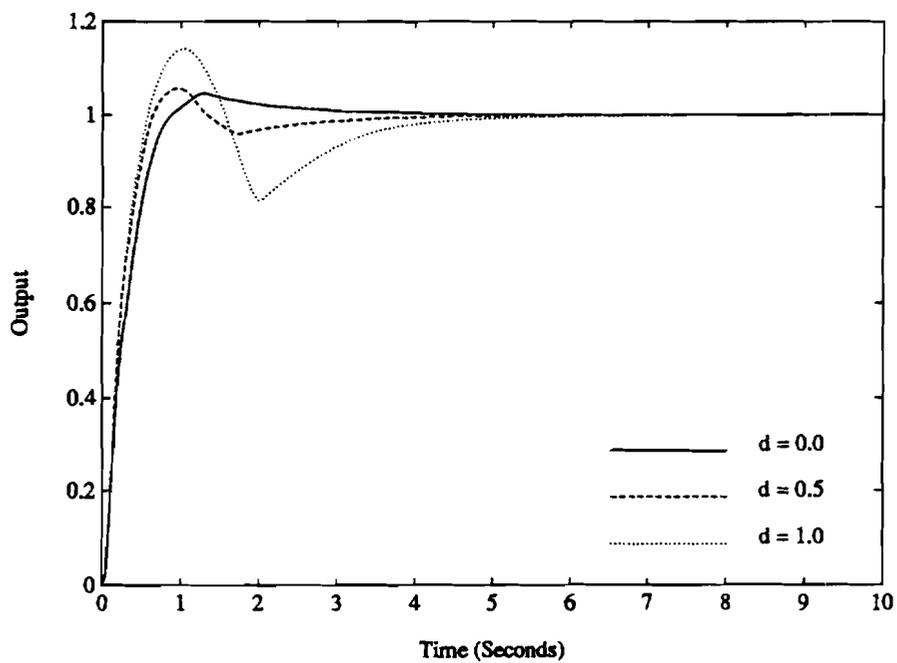


Figure 10: Output responses of plant with Fuzzy **PID** controller

conventional FLC system. Suppose we shift the graph of  $K_s D[K_1 y_m + F[e, 0]] - y_m$  to the left by an amount equal to  $\eta$  (the intersection point of the graph with the  $e$ -axis). Then, it is clear that the steady-state error (the point of intersection of the two graphs in Figure 5(b)) becomes zero. Shifting the graph of  $K_s D[K_1 y_m + F[e, 0]] - y_m$  to the left by an amount  $\eta$  is equivalent to adding  $\eta$  to  $e$ . In other words, the graph of  $K_s D[K_1 y_m + F[e + \eta, 0]] - y_m$  intersects the graph of  $-e$  at the origin. The key idea underlying our proposed controller is to shift the curve of  $K_s D[K_1 y_m + F[e + \eta, 0]] - y_m$  as described above so that the steady-state error is zero. Note that instead of adding  $\eta$  to  $e$  to shift the curve, we can achieve a similar effect by adding some other constant  $\mu$  to the reference input  $y$ . In our controller we use fuzzy logic rules to calculate the appropriate value of  $\mu$  to be added to the reference input. Notice that unlike in the conventional FLC case, the above argument does not depend on assuming that the controller is well-tuned to the steady-state gain  $K_s$ , i.e.,  $K_1$  need not be equal to  $K_s^{-1}$ , so long as the graph of  $K_s D[K_1 y_m + F[e + \eta, 0]] - y_m$  is shifted by the appropriate amount. We can treat  $K_1$  as an additional design parameter.

We now proceed to describe our proposed controller. First, we define the variables  $y'_m(k)$  and  $e'(k)$  as follows:

$$\begin{aligned} y'_m(k) &= y_m(k) + \mu(k) \\ e'(k) &= e(k) + \mu(k) \end{aligned}$$

where:  $\mu(k)$  is a compensating term which is generated using a fuzzy logic scheme, which we will describe below. The proposed control scheme is shown in Figure 11. As we can see, the controller consists of two "layers": a fuzzy precompensator, and a conventional FLC. Hence we refer to our scheme as a two-layered fuzzy logic controller. The error  $e(k)$ , change of error  $\Delta e(k)$ , and  $\mu(k-1)$  (previous compensating term) are inputs to the precompensator. The output of the precompensator is  $\mu(k)$ . The dynamics of overall system is then described by the following equations:

$$\begin{aligned} e(k) &= y_m(k) - y_p(k) \\ \Delta e(k) &= e(k) - e(k-1) \end{aligned}$$

$$\begin{aligned}
\mu(k) &= G[e(k), \Delta e(k), \mu(k-1)] \\
y'_m(k) &= y_m(k) + \mu(k) \\
e'(k) &= y'_m(k) - y_p(k) \\
\Delta e'(k) &= e'(k) - e'(k-1) \\
v(k) &= K_1 y'_m(k) + F[e'(k), \Delta e'(k)] \\
u(k) &= D[v(k)] \\
y_p(k) &= P(z)[u(k)].
\end{aligned}$$

In the next two sections we describe in detail the two layers of our proposed controller structure.

### III.2 First Layer: Fuzzy Precompensator

We now describe the first layer in our two-layered controller structure, which consists of the fuzzy logic-based precompensator. As before, our fuzzy precompensator makes use of a set of linguistic values. However, in addition to the previous set of linguistic values  $L$  and membership functions  $M$ , the precompensator also uses a new set of linguistic values  $L' = \{NE, ZE, PO\}$  and associated membership functions  $M' = \{M_{NE}, M_{ZE}, M_{PO}\}$ . The mnemonic  $NE$  stands for "negative",  $ZE$  stands for "zero", and  $PO$  stands for "positive". Figure 12 shows a plot of the membership functions in  $M'$ . The linguistic values in  $L'$  are used for the "input" variables of the precompensator, while the linguistic values in  $L$  are used for the "output".

As before, the fuzzy precompensator consists of three steps: fuzzification, decision making fuzzy logic, and defuzzification. For each  $l' \in L'$ , the fuzzification process for the precompensator assigns to each of the inputs  $e(k)$ ,  $\Delta e(k)$ , and  $\mu(k-1)$ , the numbers  $m_e(l')$ ,  $m_{\Delta e}(l')$  and  $m_\mu(l')$ , respectively, via

$$\begin{aligned}
m_e(l') &= M_{l'}(C'_e e(k)) \\
m_{\Delta e}(l') &= M_{l'}(C'_{\Delta e} \Delta e(k)) \\
m_\mu(l') &= M_{l'}(C'_\mu \mu(k-1))
\end{aligned}$$

where  $C'_e$ ,  $C'_{\Delta e}$ , and  $C'_\mu$  are scale factors. Associated with the decision making fuzzy logic stage of the precompensator are twenty-seven rules  $\{R'_1, \dots, R'_{27}\}$ , as shown in Table 2. In this case, each rule  $R'_i$  is a quadruplet  $(l'_e, l'_{\Delta e}, l'_\mu, l_\mu)$ , where  $l'_e, l'_{\Delta e}, l'_\mu \in L'$ , and  $l_\mu \in L$  (where  $L$  is the set of linguistic values used in the conventional FLC as described previously). As mentioned before, we usually express the rule as "if  $e(k)$  is  $l'_e$  and  $\Delta e(k)$  is  $l'_{\Delta e}$  and  $\mu(k-1)$  is  $l'_\mu$ , then  $\mu$  is  $l_\mu$ ". In this case, we think of  $\mu$  as the output of the rule. We emphasize that the "output linguistic value"  $l_\mu$  is in  $L$  (not  $L'$ ). For each rule  $R'_i = (l'_e, l'_{\Delta e}, l'_\mu, l_\mu)$ ,  $i = 1, \dots, 27$ , we compute the function  $p'_i(e(k), \Delta e(k), \mu(k-1), \mu)$ ,  $\mu \in [-1, 1]$ , as follows:

$$N'_{min} = \min(m_e(l'_e), m_{\Delta e}(l'_{\Delta e}), m_\mu(l'_\mu))$$

$$p'_i(e(k), \Delta e(k), \mu(k-1), \mu) = \min(N'_{min}, M_{l_\mu}(\mu))$$

where  $M_{l_\mu}$  is the membership function of  $l_\mu \in L$ , as shown in Figure 3. We combine the functions  $p'_i$ ,  $i = 1, \dots, 27$ , to get

$$q'(e(k), \Delta e(k), \mu(k-1), \mu)$$

$$= \max(p'_1(e(k), \Delta e(k), \mu(k-1), \mu), \dots, p'_{27}(e(k), \Delta e(k), \mu(k-1), \mu))$$

Finally, the defuzzification process for the precompensator gives us the real output  $G[e(k), \Delta e(k), \mu(k-1)]$  (using the COA method as before):

$$G[e(k), \Delta e(k), \mu(k-1)] = C_G \frac{\int_{-1}^1 \mu q'(e(k), \Delta e(k), \mu(k-1), \mu) d\mu}{\int_{-1}^1 q'(e(k), \Delta e(k), \mu(k-1), \mu) d\mu} + \mu(k-1)$$

where  $C_G$  is a scale factor. Note that we add  $\mu(k-1)$  to the computed and scaled center-of-error term.

### 111.3 Second Layer: Conventional FLC

The second layer of our controller structure consists of a conventional FLC, which is essentially identical to that described in Section II.2. The only difference in this case is that instead of using  $e(k)$  and  $\Delta e(k)$  as inputs to the FLC, we use  $e'(k)$  and  $\Delta e'(k)$ , where  $e'(k) = e(k) + \mu(k)$ ,  $\Delta e'(k) = e'(k) - e'(k-1)$ , and  $\mu(k) = G[e(k), \Delta e(k), \mu(k-1)]$ .

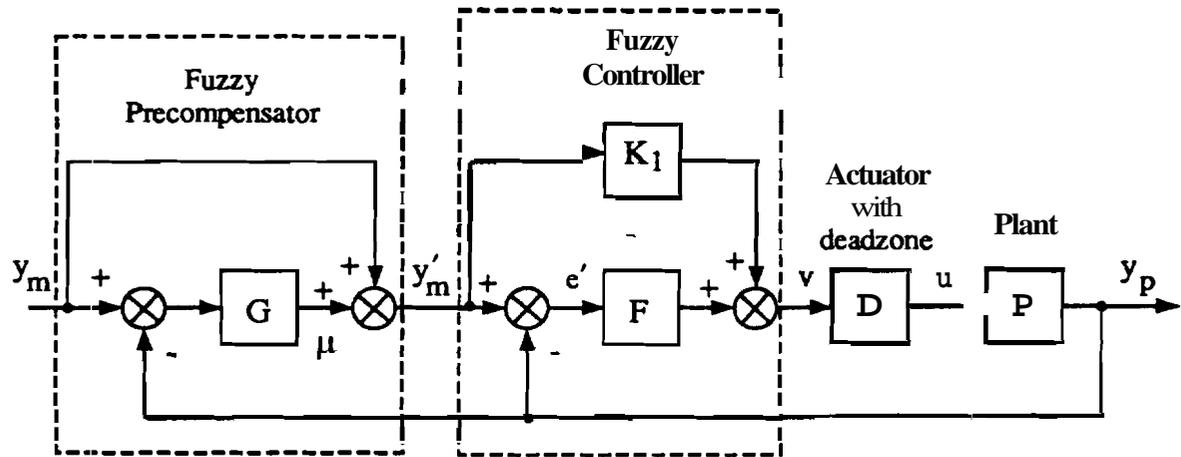


Figure 11: Proposed Two-Layered Fuzzy Logic Controller

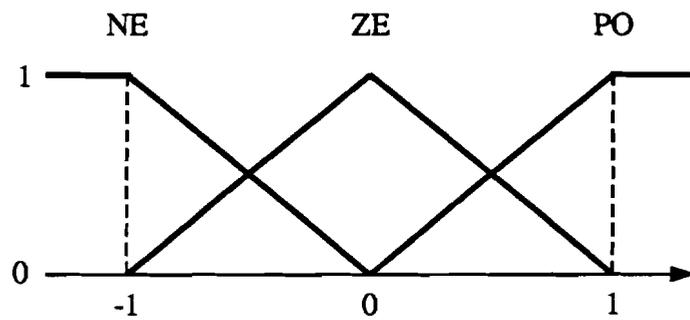


Figure 12: Membership Functions

IF			THEN
$e(k)$	$\Delta e(k)$	$\mu(k-1)$	$\mu(k)$
NE	NE	NE	NS
		ZE	ZO
		PO	ZO
	ZE	NE	PS
		ZE	ZO
		PO	NS
PO	NE	PM	
	ZE	PS	
	PO	ZO	
ZE	NE	NE	ZO
		ZE	NS
		PO	NS
	ZE	NE	ZO
		ZE	ZO
		PO	ZE
PO	NE	PS	
	ZE	PS	
	PO	ZO	
PO	NE	NE	PM
		ZE	PS
		PO	ZO
	ZE	NE	PM
		ZE	PS
		PO	ZO
PO	NE	PB	
	ZE	PS	
	PO	ZO	

**Table 2: Rules for the Fuzzy Precompensator**

1)] is the output of the precompensator. In particular, as indicated by the dynamics equations previously, the output of the FLC is given by

$$v(k) = K_1 y'_m(k) + F[e'(k), \Delta e'(k)]$$

where  $y'_m(k) = y_m(k) + \mu(k)$ .

### III.4 Example

We consider again the plant of Section 11.4. We now apply the proposed two-layered fuzzy logic controller to the plant; as before we use a sampling time of 0.025 seconds. The scale factors used in the second layer (conventional FLC) are as before, except with  $y_m$  replaced by  $y'_m$ , i.e.,  $C_e = 1/y'_m$ ,  $C_{\Delta e} = 9/y'_m$ , and  $C_F = 5y'_m$ . The scale factors used in the first layer (precompensator) are as follows:  $C'_e = 4.5/y_m$ ,  $C'_{\Delta e} = 49.5/y_m$ ,  $C'_\mu = 3/y_m$ ,  $C_G = 0.2y_m$ . In this example, we once again set  $y_m = 1$ , and  $K_1 = 0.1$ .

Figure 13(a) shows output responses of the plant for  $m = 1$  and three values of  $d$  (as before): 0.0, 0.5, 1.0. The output responses in Figure 13(a) show considerable improvement over those of Figure 6. Not only is the steady-state error reduced to virtually zero, but the transient response is also improved. In Figure 13(a), the "internal variables" (e.g., scale factors, membership functions) used were "tuned" for a deadzone width of  $d = 0$  and a slope of  $m = 1$ . Nevertheless, as we can see, the controller also performs well for deadzone widths of  $d = 0.5$  and 1.0. Therefore, we conclude that our controller is robust to variations in the deadzone width. In practice, we can use the same values of interval variables for a whole range of deadzone widths, without having to retune the values. However, as we can see in Figure 13(a), the transient response does deteriorate slightly as  $d$  increases. This deterioration can be eliminated if we readjust the internal variables for the particular  $d$ .

Figure 13(b) shows output responses of the plant for  $d = 0.5$  and three values of  $m$ : 2.0, 3.0, 6.0. In all three plots, the same values for the internal variables were used as before, except  $C_G = 3.5$  in this case. As we can see, the controller performs well in

all **three** cases. Hence we conclude that the controller is also robust to variations in slope. Naturally, the performance deteriorates as  $m$  increases, and the performance at a particular slope  $m$  will be better if the internal variables are specially tuned for that specific  $m$ .

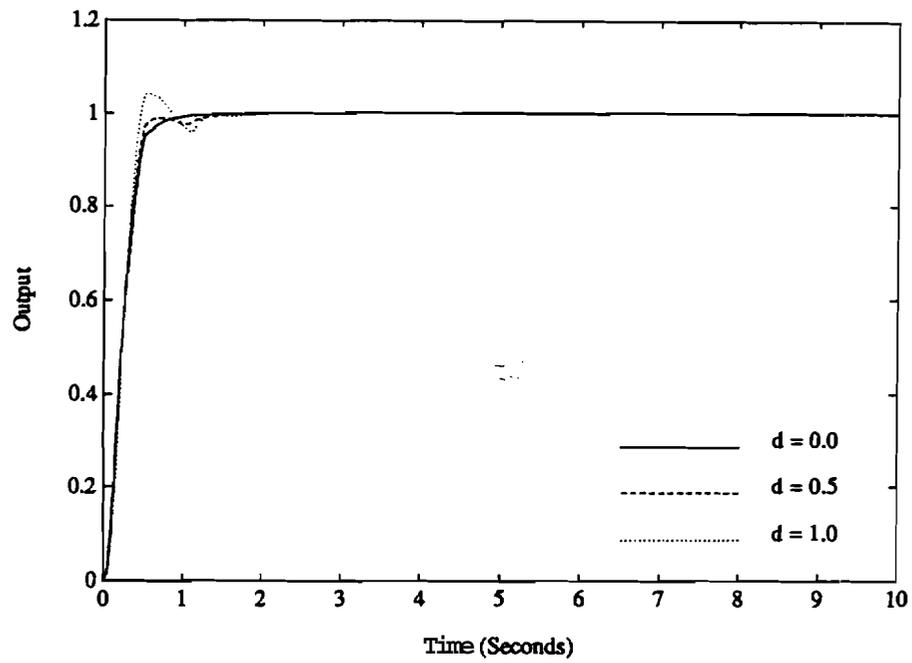
In the above examples we used  $K_1 = 0.1 = K_s^{-1}$ , which means that  $K_1$  is “well-tuned.” to the steady-state gain of the plant. Figure 14 show output responses of the plant with values of  $K_1$  which are not well-tuned; in Figure 14(a) we used  $K_1 = 0.5$  (5 times  $K_s^{-1}$ ), and in Figure 14(b) we used  $K_1 = 0.02$  (1/5 times  $K_s^{-1}$ ). We can see that the performance is relatively robust to the choice of  $K_1$ . Naturally, with fixed values of  $K_1$  and the internal variables, we expect the performance to deteriorate with increasing **deadzone** widths, as illustrated in Figure 14. The **performance** for large **deadzone** widths may be improved if we retune the internal variables.

## IV Conclusions

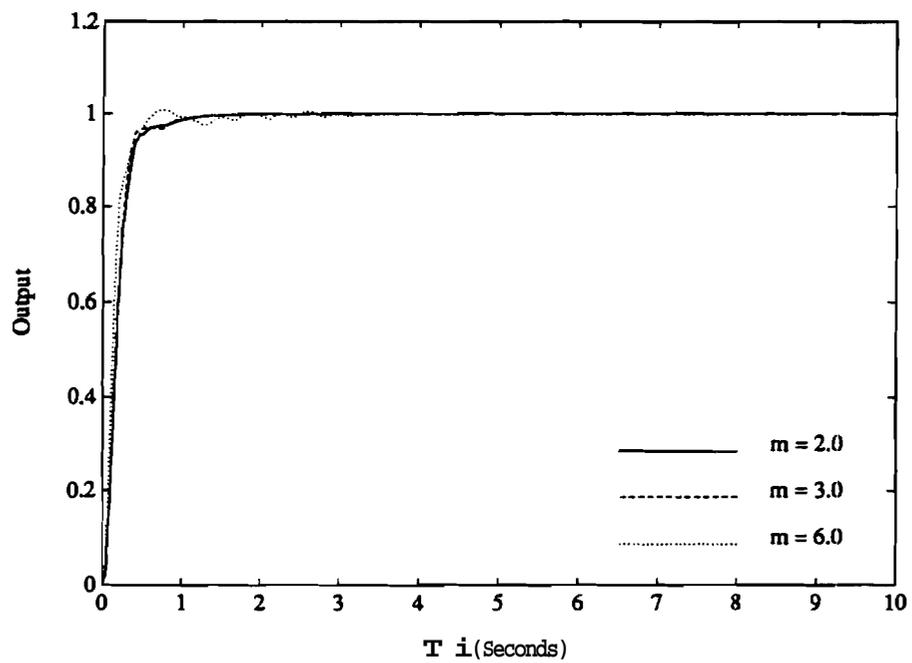
In this report, we proposed a two-layered fuzzy logic controller for systems with **deadzones**. Our controller consists of a fuzzy precompensator and a conventional FLC. The proposed controller has superior steady-state and transient performance, compared to a conventional FLC. An advantage of our present **approach** is that an **existing** conventional FLC can be easily modified into our control structure by **adding** a fuzzy precompensator, without having to retune the internal variables of the existing FLC. In addition, the two-layered control structure is robust to variations in the **deadzone** nonlinearities (width and slope), **as** well as the steady-state gain of the plant. We demonstrated the performance of our controller via **several** computer **simulation** examples.

## References

- [1] V. I. Utkin, *Sliding Modes and Their Application in Variable Structure Systems*.

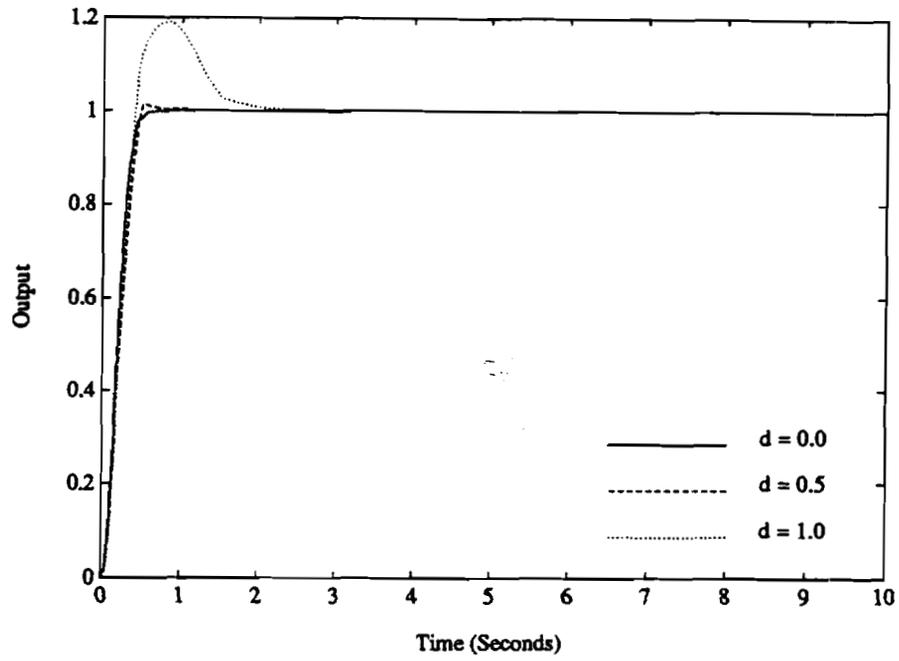


(a)

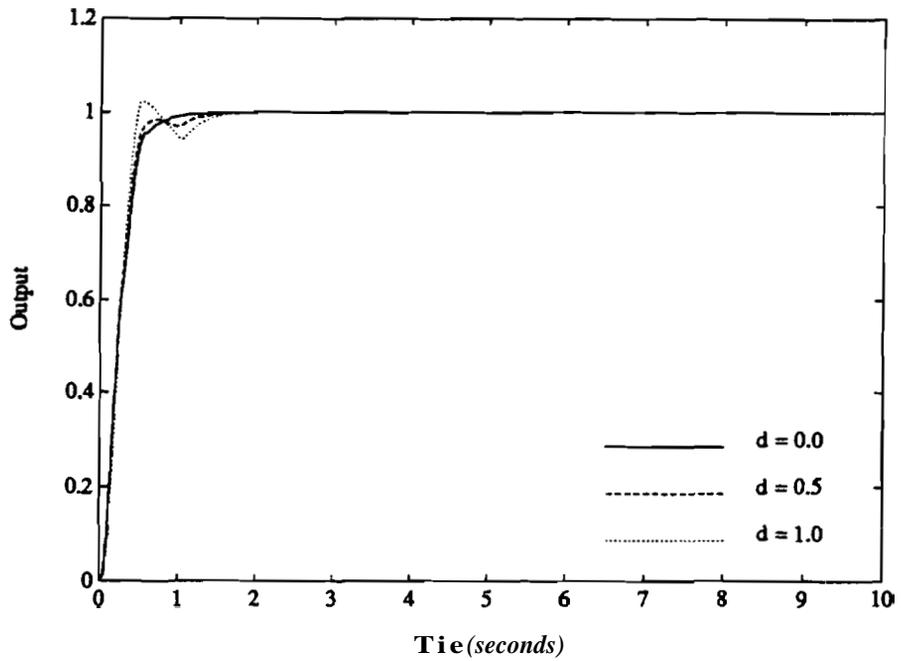


(b)

Figure 13: Output responses of plant with proposed FLC



(a)



(b)

Figure 14: Output responses of plant with proposed FLC

Moscow:Mir, 1978.

- [2] C. A. Desoer and S. M. Shahruz, "Stability of dithered **nonlinear** systems with **backlash** or hysteresis," *Inf. J. Control*, vol. 43, no. 4, pp. 1045–1060, 1986.
- [3] D. A. Recker, P. V. Kokotovic, D. Rhode and J. Winkelman, "Adaptive nonlinear control of systems containing a dead-zone," in *Proc. of the IEEE Conf. on Dec. and Contr.*, pp. 2111–2115, Brighton, U.K., Dec. 1991.
- [4] G. Tao and P. V. Kokotovic, "Adaptive control of Plants with **unknown** dead-zones," Report No. CCEC-91-1006, Univ. of Cal., Santa Barbara, U.S.A., Sept. 1991.
- [5] E. H. Mamdani and B. R. Gaines, "Fuzzy reasoning and *its Applications*" London:Academic. 1981.
- [6] Y. F. Li and C. C. Lau, "Development of fuzzy algorithms for servo systems," *IEEE Contr. Syst. Magazine*, vol. 9, pp. 65–72, 1989.
- [7] C. C. Lee, "Fuzzy Logic in Control **Systems:Fuzzy** Logic Controller—Part I,PartII," *IEEE Trans. on Sys. Man and Cyber.*, vol. 20, no. 2, pp. 404–435,1990.
- [8] D. P. Kwok, P. Tam, C. K. Li and P. Wang, "Linguistic PID controllers," in *Proc. of IFAC 11th Triennial World Congress*, Tallinn, USSR, vol. 4, pp. 205–210, Aug. 1990.
- [9] I. Kosko, *Neural Networks and Fuzzy Systems*, Englewood Cliff's, New Jersey: Prentice Hall, 1992.