A Robust Self-Tuning Scheme for PI- and PD-Type Fuzzy Controllers

Rajani K. Mudi and Nikhil R. Pal, *Member, IEEE*

***Abstract*—We propose a simple but robust model independent self-tuning scheme for fuzzy logic controllers (FLC's). Here, the output scaling factor (SF) is adjusted on-line by fuzzy rules according to the current trend of the controlled process. The rule-base for tuning the output SF is defined on error (e) and change of error (Δe) of the controlled variable using the most natural and unbiased membership functions (MF's). The proposed self-tuning technique is applied to both PI- and PD-type FLC's to conduct simulation analysis for a wide range of different linear and nonlinear second-order processes including a marginally stable system where even the well known Ziegler–Nichols tuned conventional PI or PID controllers fail to provide an acceptable performance due to excessively large overshoot. Performances of the proposed self-tuning FLC's are compared with those of their corresponding conventional FLC's in terms of several performance measures such as peak overshoot, settling time, rise time, integral absolute error (IAE) and integral-of-time-multiplied absolute error (ITAE), in addition to the responses due to step set-point change and load disturbance and, in each case, the proposed scheme shows a remarkably improved performance over its conventional counterpart.**

***Index Terms*—Fuzzy controller, scaling factors, self-tuning controller.**

I. Introduction

FUZZY logic controllers (FLC's) have been reported to be successfully used for a number of complex and nonlinear processes [1]. Sometimes FLC's are proved to be more robust [2] and their performances are less sensitive to parametric variations [3] than conventional controllers. A comprehensive review on the design and implementation of FLC's can be found in [3]–[5]. Different types of adaptive FLC's such as self-tuning and self-organizing controllers have also been developed [6]–[11] and implemented for various practical processes. Even equivalence between FLC's and conventional controllers have been established [12]–[14]. Recently many researchers are trying to achieve enhanced performance and increased robustness of FLC's, using neural networks and genetic algorithms in designing such controllers [15]–[18].

Among the various types proportional integral (PI), proportional derivative (PD), and proportional-integral derivative (PID) of FLC's, just like the widely used conventional PI controllers [19] in process control systems, PI-type FLC's are most common and practical followed by the PD-type FLC's. Because proportional (P) and integral (I) actions are combined in the proportional-integral (PI) controller to take advantages of the inherent stability of proportional controllers and the offset elimination ability of integral controllers. The performance of PI-type FLC's is known to be quite satisfactory for linear first-order systems. But like conventional PI-controllers, performance of PI-type FLC's for higher order systems, systems with integrating elements or large dead time, and also for nonlinear systems may be very poor due to large overshoot and excessive oscillation. Such systems may be ultimately uncontrollable [20]. For example, the first overshoot in the step response of a system with large dead time, may be too large to be acceptable for many applications. PD-type FLC's are suitable for a limited class of systems [21]. And they are not recommendable in presence of measurement noise and sudden load disturbances. PID-type FLC's are rarely used due to the difficulties associated with the generation of an efficient rule base and the tuning of its large number of parameters.

An FLC has a fixed set of control rules, usually derived from experts' knowledge. The membersip functions (MF's) of the associated input and output linguistic variables are generally predefined on a common universe of discourse. For the successful design of FLC's proper selection of input and output scaling factors (SF's) and/or tuning of the other controller parameters are crucial jobs, which in many cases are done through trial and error or based on some training data. Of the various tunable parameters, SF's have the highest priority due to their global effect on the control performance. However, relative importance of the input and output SF's to the performance of a fuzzy logic control system is yet to be fully established.

Unlike conventional control, which is based on mathematical model of a plant, a FLC usually embeds the intuition and experience of a human operator and sometimes those of designers and researchers. While controlling a plant, a skilled human operator manipulates the *process input* (i.e., controller output) based on e and Δe with a view to minimizing the error within the shortest possible time. Fuzzy logic control is a knowledge-based system. By analogy with the human operator, the output SF should be considered a very important parameter of the FLC since its function is similar to that of the controller gain. Moreover, it is directly related to the stability of the control system. So the output SF should be determined very carefully for the successful implementation of a FLC.

Manuscript recrived March 9, 1997; revised May 15, 1998. This work was supported by Prof. D. Patranabis, Department of Instrumentation Engineering, Jadavpur University, Calcutta, India, and the Department of Science and Technology, Government of India, under Grant III.5(21)/96-ET.

R. K. Mudi is with the Department of Instrumentation Engineering, Jadavpur University, Salt-Lake Campus, Calcutta, 700 091 India.

N. R. Pal is with the Machine Intelligence Unit, Indian Statistical Institute, Calcutta, 700 035 India.

Publisher Item Identifier S 1063-6706(98)08257-5.

Most of the practical processes under automatic control are nonlinear higher order systems and may have considerable dead time. Sometimes their parameters may be randomly changed with changes in ambient conditions or with time. Control action is unavoidably delayed in a process with dead time. For this reason, dead time is recognized as the most difficult dynamic element naturally occurring in physical systems [19]. Therefore, any useful technique of designing a control system must be capable of dealing with dead time. To have a satisfactory performance the controller output or process input should be a nonlinear function of e and Δe. FLC tries to incorporate this nonlinearity by a limited number of IF-THEN rules, which may not always be enough to produce a good approximation to the controller output required for the optimum performance. In such a situation, only static or fixed valued SF's and predefined MF's may not be sufficient to eliminate this drawback. To overcome this, a lot of research works on tuning of FLC's have been reported where either the input–output SF's or the definitions of fuzzy sets are tuned (on-line or off-line) to match the current plant characteristics [6], [7], [22]–[28].

He *et al.* [6] proposed a scheme for self-tuning of a conventional PID controller using fuzzy rules. The proportional sensitivity (K_p), integral time (T_i) and derivative time (T_d) are initially calculated using Ziegler–Nichols tuning formula [29]. These three parameters are then modified on-line by a single parameter, which is updated by a rule base defined on e and Δe. It is reported [6] that there is a considerable improvement in the overall performance of the controller over its conventional counterpart. Results in [6] shows a remarkable reduction in overshoots of second-order processes with dead time but at the cost of increased rise times. The self-tuning method of FLC's by Nomura *et al.* [22] is a well-known gradient decent technique to optimize both the fuzzy antecedent and crisp consequent parts. The controller [22] is tuned iteratively by minimizing the square error between the FLC output and the desired output given by the training data. This method simultaneously modifies the crisp consequent values and, centers and widths of triangular input fuzzy sets. This off-line tuning method may be very good for time-invariant control systems, but its applicability is limited due its dependency on the availability of a reliable set of training data. Zheng [23] suggested to tune the parameters of PI-type FLC's in order of their significance; that is, first parameters with a global effect and then ones with only local effect and, hence, given the maximum importance to the tuning of SF's. Zheng did not provide any algorithm for tuning of FLC's, but discussed various factors, their interaction, and their impact on the controller performance that should be considered while designing tuning algorithms for FLC's. Input and output SF's are recommended to be selected from the knowledge of conventional PI-controller parameters (K_p and T_i) if available, otherwise through trial and error. Simulation result in [23] with tuned MF's shows a marginal improvement in transient response of a second-order linear process where tuning resulted in asymmetric (triangle) MF's with unequal base for e. To be more specific, the width of MF's increased around $e = 0$. Such MF's contradicts the usual practice [3], [5], [30] where the MF's take narrow width and become more crowded near the origin to provide increased sensitivity around the steady state condition. Thus, the proposed MF's tuning scheme [23] cannot guarantee improved performance under load disturbance, which is a very important criteria for the performance evaluation of any control system [19].

The gain tuning method of Yoshida *et al.* [24] assumes all processes as *first-order* systems with dead time. The input and output SF's are calculated by some empirical relations involving process parameters. Good control performances for higher order systems cannot be ensured by this technique. Auto-tuning fuzzy controller of Hayashi [25] considers two tuning functions. From the approximate parameters of the identified plant model (first-order lag with dead time) the input and output SF's are calculated using the concept of Chien–Hrones–Reswick (CHR) tuning rules for a conventional PI controller. Then the crisp consequent parts are modified by fuzzy rules using overshoot and rise time to improve the control performance. Linear first-order plant models with dead time have also been considered in the auto-tuning scheme of Iwasaki and Morita [26]. Here, the parameters of an assumed plant model are iteratively revised through fuzzy inference using differences between the actual plant features (rise time and overshoot) and the plant model features. The overall performances of these controllers [24]–[26] will be dependent on the appropriateness of the assumed process model.

Palm [27] proposed to achieve the optimal adjustment in the input SF with the help of input–output cross-correlation function, though he assigned a higher priority to the tuning of output SF over that of input SF's. Here, the input data are assumed to follow a Gaussian distribution whose parameters are unknown. An optimal input SF is obtained by maximizing the cross-correlation function, which is a measure of the statistical dependence between input and output. Li and Gatland [28] also have given more emphasis on the tuning of input and output SF's than that of MF's or rule base. They basically suggested a trial and error method for tuning of input and output SF's for a fuzzy PID controller developed from two FLC's in parallel—one is a PI-type and the other is a PD-type. Maeda and Murakami [7] proposed fuzzy rule-based schemes for adjustment of input–output SF's as well as for tuning of control rules for Takagi–Sugeno (TS) model. The fuzzy rule-base for tuning has three sets of rules, based on three different performance measures: overshoot, rise time, and amplitude. After the tuning of SF's, the crisp consequent parts of the control rules are modified in each sampling time considering a fuzzy performance index and the deviation of the actual control response from a predefined target response.

With a view to eliminating the overshoot caused by the accumulation of control input in a fuzzy PI-type controller, Lee [20] proposed two augmented versions of the conventional fuzzy PI controller using resetting factors. The first of the two fuzzy controllers determines the resetting rate based on error and error rate, while the second one uses error and control input. The computation of the resetting factor is driven by a fuzzy rule base. The controller remarkably improves the transient response of a second-order linear system with integrating element. But the authors in [31] have clearly shown

with extensive simulation conducted on different types of second-order linear as well as nonlinear systems with and without integration that the controller in [20] with resetting action is almost similar to a conventional fuzzy PD controller.

The preceding discussion shows that many researchers have tried to improve the tuning methods of FLC's to make its design easy and faster. But unlike conventional controllers a standard and systematic method for the tuning of FLC's (PI, PD or PID) is yet to be developed. Sometimes nonfuzzy schemes are used for the tuning of fuzzy controllers [22], [24] while in other occasions fuzzy inference mechanisms are used for tuning of nonfuzzy controllers [6], [26]. Of course, there are many fuzzy controllers [7], [20], [25] tuned using fuzzy inference mechanisms. Moreover, most of the reported works on FLC tuning is limited only to the first-order linear systems with dead time. But it is very hard to have such simple and perfect models for practical processes that are generally nonlinear and higher order systems and sometimes have large dead time. *Thus, there is a need for a robust tuning scheme of FLC's, which would be applicable irrespective of the nature of the processes and the structure of the FLC's*. Designing a general tuning method for FLC's to obtain the optimal response is not an easy task because the computation of the optimal values of the tunable parameters (SF's, MF's, and rules) needs the required control objectives as well as the fixed model of the controller. Unfortunately, FLC's have no fixed structures like conventional PI, PD, and PID controllers because there is still no well-defined criteria for deciding on: 1) the shape of MF's; 2) the number of linguistic values; 3) the standard rule-base; and 4) the most appropriate inference mechanism and defuzzification strategy. Probably due to these reasons designing an optimal FLC analytically becomes very difficult.

These limitations of the conventional FLC's motivated us to investigate methods of tuning based on experts' knowledge rather than mathematical models. Our scheme is based on the fact that irrespective of the nature of the process to be controlled and the control policy adopted, a skilled human operator always tries to manipulate the process *input*, usually by adjusting the controller gain based on the current process states (generally e and Δe) to get the process "optimally" controlled. The exact manipulation strategy of an operator is quite complex in nature and possibly no mathematical model can replace it accurately. We propose a simple but robust *model independent* self-tuning scheme, where the controller gain is adjusted continuously with the help of fuzzy rules. Here, we have concentrated only on the tuning of output SF, considering that it is equivalent to the controller gain. Tuning of the output SF has been given the highest priority because of its strong influence on the performance and stability of the system. Palm [27] also has rightly pointed out this fact. In our scheme, the FLC is tuned on-line (while the controller is in operation) by dynamically adjusting its output SF by a gain updating factor (α). The value of α is determined from a rule base defined on e and Δe and derived from the knowledge of control engineering. Note that our scheme is significantly different from others [7], [22]–[28] as it actually tries to mimic an operator's strategy. In order to achieve the desired performance, our scheme generates subsequent corrective control actions based on the current process trend only, not from direct performance measures. The self-tuning mechanism is applied to both PI- and PD-type FLC's for simulation experiments with various types of linear as well as nonlinear processes that are generally encountered in process industries. The proposed FLC's are also used for marginally stable and unstable systems where well-known Ziegler–Nichols tuned PI or PID controllers exhibit very poor performance [32], [33]. The simulation results show that in each case the proposed scheme outperforms its conventional counterpart.

The rest of the paper is divided into three sections. In Section II, the proposed self-tuning FLC is described in detail mentioning different aspects of its design consideration, i.e., choice of MF's, selection of SF's, determination of rules and the self-tuning mechanism. The simulation results for various types of linear and nonlinear second-order processes, including an unstable system (system with nonminimum phase pole), are presented in Section III. Finally, we conclude with Section IV.

II. The Proposed Self-Tuning Fuzzy Controller

Here, we consider both PI- and PD-type FLC's. The gain (output SF) of these controllers are adjusted on-line according to the current states of the controlled processes, thereby making them self-tuning FLC's. Although, the characteristics of a PI- or PD-type FLC depends on both input and output SF's [14], [23] (i.e., for the best performance, simultaneous adjustment of both input and output SF's is more justified), our objective here is to adapt only the output SF for given input SF's to achieve better control performance. Observe that a self-tuning FLC is an adaptive controller but, there is no consensus in the literature on the terminology used in describing adaptive controllers. We call an FLC adaptive if any one of its tunable parameters (SF's, MF's, and rules) changes when the controller is being used, otherwise it is a nonadaptive or conventional FLC [5]. An adaptive FLC that fine tunes an already working controller by modifying either its MF's or SF's or, both of them [5], [7], [22], [24]–[26] is called a self-tuning FLC. On the other hand, when a FLC is tuned by automatically changing its rules then it is called a self-organizing FLC [8]–[11]. Since our proposed FLC is tuned by modifying the output SF of an existing FLC we describe it as a self-tuning FLC. The block diagram of the proposed self-tuning FLC is shown in Fig. 1. Though the control system shown in Fig. 1 is a PI-type FLC, the basic structure of the proposed self-tuning FLC shown by the dash-dot (-·-) line in Fig. 1 remains the same for both PI- and PD-type FLC's. In the case of a PI-type FLC, the actual value of the controller output (u) is obtained by

$$u(k) = u(k-1) + \Delta u(k). \tag{1}$$

In (1), k is the sampling instance and $\Delta u(k)$ is the incremental change in controller output. We emphasize here that this accumulation (1) of controller output takes place outside the FLC and is not reflected in the rules themselves. On the other hand, if the output of the FLC is u (not Δu) and there is no accumulation of controller output, then Fig. 1 is converted

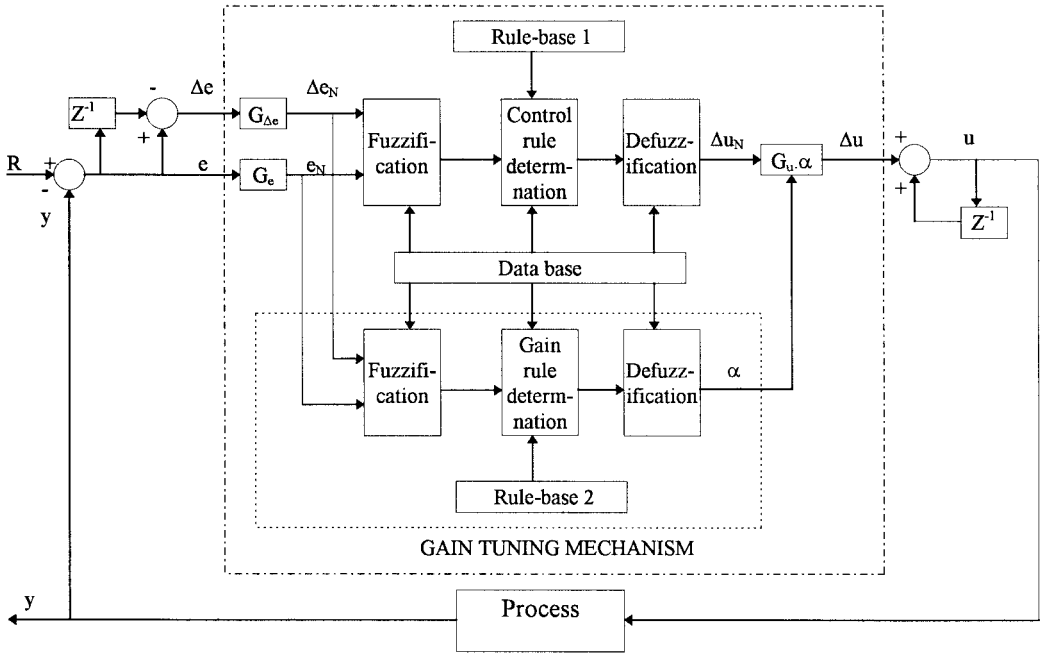


Fig. 1. Block diagram of the proposed self-tuning FLC.

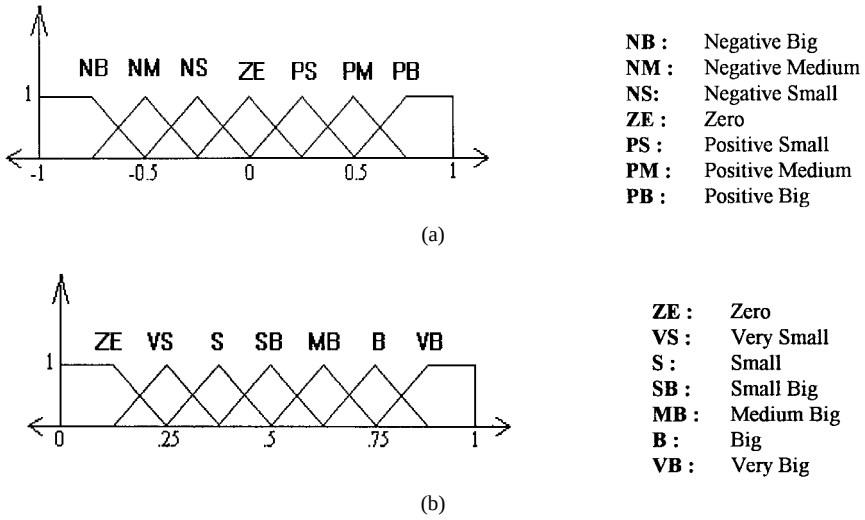


Fig. 2. Membership functions of (a) $E, \Delta E, \Delta U$, and U and (b) gain updating factor (α).

to a PD-type FLC. In this paper, PI- and PD-type conventional FLC's will be denoted by FPIC and FPDC, respectively, and their corresponding self-tuning FLC's will be denoted by STFPIC and STFPDC. Fig. 1 reveals that the output SF (gain) of the controller is modified by a self-tuning mechanism, which is shown by the dotted boundary. The detailed design considerations are discussed next.

A. Membership Functions

All membership functions (MF's) for: 1) controller inputs, i.e., error (e) and change of error (Δe) and 2) incremental change in controller output (Δu) for PI-type FLC or controller output (u) for PD-type FLC, are defined on the common interval [−1, 1]; whereas the MF's for the gain updating factor (α) is defined on [0, 1]. We use symmetric triangles (except the two MF's at the extreme ends) with equal base and 50% overlap with neighboring MF's as shown in Fig. 2. This is the most natural and unbiased choice for MF's. Though the MF's in Fig. 2(a) and (b) are shown separately for the shake of clear understanding, in actual implementation, only the MF's of Fig. 2(a) are sufficient. Fig. 2(b) can be obtained by translating Fig. 2(a) along the horizontal axis by an amount $+1$ and then mapping it on [0, 1] by the following:

$$y = 0.5(x + 1). \tag{2}$$

Here, x is any point in the closed interval $[-1, 1]$ on the horizontal axis of Fig. 2(a) and y is the corresponding point on Fig. 2(b). By this transformation MF's $NB, NM, NS, ZE, PS, PM,$ and PB of Fig. 2(a) are mapped to the MF's $ZE, VS, S, SB, MB, B,$ and VB,

respectively, of Fig. 2(b). Thus, the data-base of the FLC in Fig. 1 contains only the quantitative information about the MF's of Fig. 2(a). Note that the volume of data-base remains the same for both the conventional and proposed modified FLC's, though one more linguistic variable (α) is considered for the proposed FLC. In this context, we mention that there is only one fuzzification module in actual implementation. Because the rule-base for α is also described in terms of e and Δe—like the control rule-base. Fig. 1 shows two fuzzification modules for the ease of understanding.

B. Scaling Factors

The MF's for both scaled inputs (e_N and Δe_N) and output (Δu_N or u_N) of the controller have been defined on the common interval $[-1, 1]$. The values of the actual inputs e and Δe are mapped onto $[-1, 1]$ by the input SF's G_e and $G_{\Delta e}$, respectively. For conventional FLC's the controller output (Δu_N or u_N) is mapped onto the respective actual output (Δu or u) domain by the output SF G_u. On the other hand, the actual output of the self-tuning FLC is obtained by using the effective SF $(\alpha.G_u)$ as shown in Fig. 1. Selection of suitable values for $G_e, G_{\Delta e}$ and G_u are made based on the knowledge about the process to be controlled and sometimes through trial and error to achieve the best possible control performance. This is so because, unlike conventional nonfuzzy controllers to date, there is no well-defined method for good setting of SF's for FLC's. We propose to compute α on-line using a model independent fuzzy rule base defined in terms of e and Δe. The relationships between the SF's and the input and output variables of the self-tuning FLC are as follows:

$$e_N = G_e.e \tag{3}$$

$$\Delta e_N = G_{\Delta e}.\Delta e \tag{4}$$

$$\Delta u = (\alpha.G_u).\Delta u_N \quad \text{(for STFPIC)} \tag{5}$$

$$u = (\alpha.G_u).u_N \quad \text{(for STFPDC)} \tag{6}$$

C. The Rule Bases

The incremental change in controller output (Δu) for a fuzzy PI controller is determined by the rules of the form:

R_{PI}: If e is E and Δe is ΔE then Δu is ΔU.

The fuzzy PD controller, on the other hand, uses rules of the form:

R_{PD}: If e is E and Δe is ΔE then u is U.

The rule base for computing Δu is shown in Fig. 3(a). This is a very often used rule base designed with a two-dimensional phase plane in mind where the FLC drives the system into the so-called sliding mode [2], [3], [20], [23], [30]. Note that for the fuzzy PD controller we use the same rule-base in Fig. 3(a), but with the consequent fuzzy memberships defined on U, not on ΔU.

The gain updating factor (α) is calculated using fuzzy rules of the form:

R_α: If e is E and Δe is ΔE then α is $\boldsymbol{\alpha}$.

Δe / e	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NM	NS	NS	ZE
NM	NB	NM	NM	NM	NS	ZE	PS
NS	NB	NM	NS	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PS	PM	PB
PM	NS	ZE	PS	PM	PM	PM	PB
PB	ZE	PS	PS	PM	PB	PB	PB

(a)

Δe / e	NB	NM	NS	ZE	PS	PM	PB
NB	VB	VB	VB	B	SB	S	ZE
NM	VB	VB	B	B	MB	S	VS
NS	VB	MB	B	VB	VS	S	VS
ZE	S	SB	MB	ZE	MB	SB	S
PS	VS	S	VS	VB	B	MB	VB
PM	VS	S	MB	B	B	VB	VB
PB	ZE	S	SB	B	VB	VB	VB

(b)

Fig. 3. (a) Fuzzy rules for computation of Δu and u. (b) Fuzzy rules for computation of α.

With a view to improving the overall control performance, we use the rule base in Fig. 3(b) for computation of α. It [Fig. 3(b)] is designed in conjunction with the rule base in Fig. 3(a). Some of the important considerations that have been taken into account for determining the rules are as follows.

1) To make the controller produce a lower overshoot and reduce the settling time (but not at the cost of increased rise time) the controller gain is set at a small value when the error is big (it may be $+ve$ or $-ve$), but e and Δe are of opposite signs. For example, If e is PB and Δe is NS then α is VS or if e is NM and Δe is PM then α is S. To minimize the effects of delayed control action due to inherent process dead time or measuring lag such small gain is essential to maintain the controller performance within the acceptable limit, especially when the process dead time becomes considerably large. Observe that when the error is big but e and Δe are of the same sign (i.e., the process is now not only far away from the set point but also it is moving farther away from it), the gain should be made very large to prevent from further worsening the situation. This has been realized by rules of the form: IF e is PB and Δe is PS THEN α is VB or IF e is NM and Δe is NM THEN α is VB.
2) Depending on the process trend, there should be a wide variation of the gain around the set point (i.e., when e is small) to avoid large overshoot and undershoot. For example, overshoot will be reduced by the rule IF e is ZE and Δe is NM THEN α is B. This rule indicates that the process has just reached the set point but it is moving away upward from the set point rapidly. In this situation, large gain will prevent its upward motion more severely resulting in a smaller overshoot. Similarly, a large undershoot can be avoided using the rules of the

form: IF e is NS and Δe is PS THEN α is VS. This type of gain variation around the set point will also prevent excessive oscillation and as a result the convergence rate of the process to the set point will be increased. Note that unlike conventional FLC's, here the gain of the proposed controller around the set point may vary considerably depending on the trend of the controlled process. Such a variation further justifies the need for variable SF.

3) Practical processes or systems are often subjected to load disturbances. A good controller should provide regulation against changes in load; in other words, it should bring the system to the stable state within a short time in the event of load disturbance. This is accomplished by making the gain of the controller as high as possible. Hence, to improve the control performance under load disturbance, the gain should be sufficiently large around the steady-state condition. For example, IF e is PS and Δe is PM THEN α is B or IF e is NS and Δe is NM THEN α is B. Note that immediately after a large load disturbance, e may be small but Δe will be sufficiently large (they will be of same sign) and, in that case, α is needed to be large to increase the gain. At steady state (i.e., $e \approx 0$ and $\Delta e \approx 0$) controller gain should be very small (e.g., IF e is ZE and Δe is ZE THEN α is ZE) to avoid chattering problem around the set point.

Further modification of the rule base for α may be required, depending on the type of response the control system designer wishes to achieve. It is very important to note that the rule base for computation of α will always be dependent on the choice of the rule base for the controller. For example, the rule base in Fig. 3(b) is justified and defined for the controller rule base in Fig. 3(a). Any significant change in the controller rule base may call for changes in the rule base for α accordingly.

D. The Self-Tuning Mechanism

From (5) and (6) it is found that the effective gain of the self-tuning controller is $\alpha.G_u$, not simply G_u. The value of G_u is constant for a particular type of conventional FLC. But the gain of our self-tuning FLC does not remain fixed while the controller is in operation, rather it is modified in each sampling time by the gain updating factor α, depending on the trend of the controlled process output. The reason behind this on-line gain variation is to make the controller respond according to the desired performance specifications. We already explained how the desired variation in α can be achieved using the rule base in Fig. 3(b). Thus, the proposed controller is basically an adaptive feedback loop controller. The functional relationship of α can be viewed as

$$\alpha(k) = f(e(k), \Delta e(k)) \tag{7}$$

where f is a nonlinear function (computational algorithm) of e and Δe, which is described by the rule base shown in Fig. 3(b) and the associated inferencing scheme. The variation of α with e and Δe is shown in Fig. 4, which is seen to be highly nonlinear. Fig. 4 depicts the desirable characteristics of α as a function of e and Δe. For example, if error is positive big and change of error is negative big then the system is moving fast toward the set point and, hence, α should be kept very small to avoid possible large overshoot. Fig. 4 indeed reflects this. Fig. 4(b), a rotated version of Fig. 4(a) is provided for a better visual representation. The control surfaces (controller output versus e and Δe) for fixed gain (conventional) and variable gain (self-tuning) FLC's are depicted in Fig. 5(a) and (b), respectively. Careful inspection of these two figures reveals that the control surface of the self-tuning FLC is more nonlinear as well as smooth than that of its conventional counterpart. For example, observe the first and third quadrants of both figures. As far as real time implementation is concerned smoothness of the control surface is highly desirable due to the limited speed of the actuator response and to avoid the chattering of gears for the plant to be controlled. Fig. 5(a) also indicates that the limited number of IF-THEN rules using simple MF's and fixed valued SF's are not sufficient to produce the required nonlinear controller output for the desired control performance. To eliminate this shortcoming, one can increase the number of MF's and there by increasing the number of rules, but it will increase the design complexity. In the proposed self-tuning scheme the controller output [Fig. 5(b)] is generated by the continuous and nonlinear variation of α. The most important point to note is that α is *not* dependent in any way, on any process parameter. The value of α depends only on the instantaneous process states. Hence, the proposed self-tuning scheme is *model independent*.

The following steps can be used for tuning of the proposed controller.

Step 1: Tune the SF's of the self-tuning FLC without the gain tuning mechanism and assuming $\alpha = 1$ (i.e., conventional FLC) for a given process to achieve a reasonably good control performance. In doing so first, G_e should be selected in such a way that the error (e_N) almost covers the entire domain $[-1, 1]$ to make efficient use of the rule bases. Then $G_{\Delta e}$ and G_u are to be tuned to make the transient response of the system as good as possible. Since there is no existing well-defined method (like Ziegler–Nichols tuning formula for conventional nonfuzzy controllers) for the determination of SF's, suitable values of $G_{\Delta e}$ and G_u are to be selected from the knowledge of the process to be controlled and sometime through trial and error, which we have already mentioned. At the end of this step, we get a good controller without self-tuning and then this controller becomes the starting point (input) for the self-tuning controller in Step 2.

Step 2: Set the output SF (G_u) of the self-tuning FLC nearly three times greater than that obtained in Step 1 keeping the values of G_e and $G_{\Delta e}$ same as those of the conventional FLC. We remind that in this step, $\alpha \neq 1$, but is obtained from the rule base in Fig. 3(b). This nearly *three times* the enhancement of G_u for the self-tuning FLC is found *empirically* with an objective to maintain the same rise time as that of the conventional FLC. In all our simulations we used it *exactly three times*. Little deviation from this factor is found to bring only small changes in the responsiveness of the system. Observe that for the self-tuning FLC a large value of G_u (three

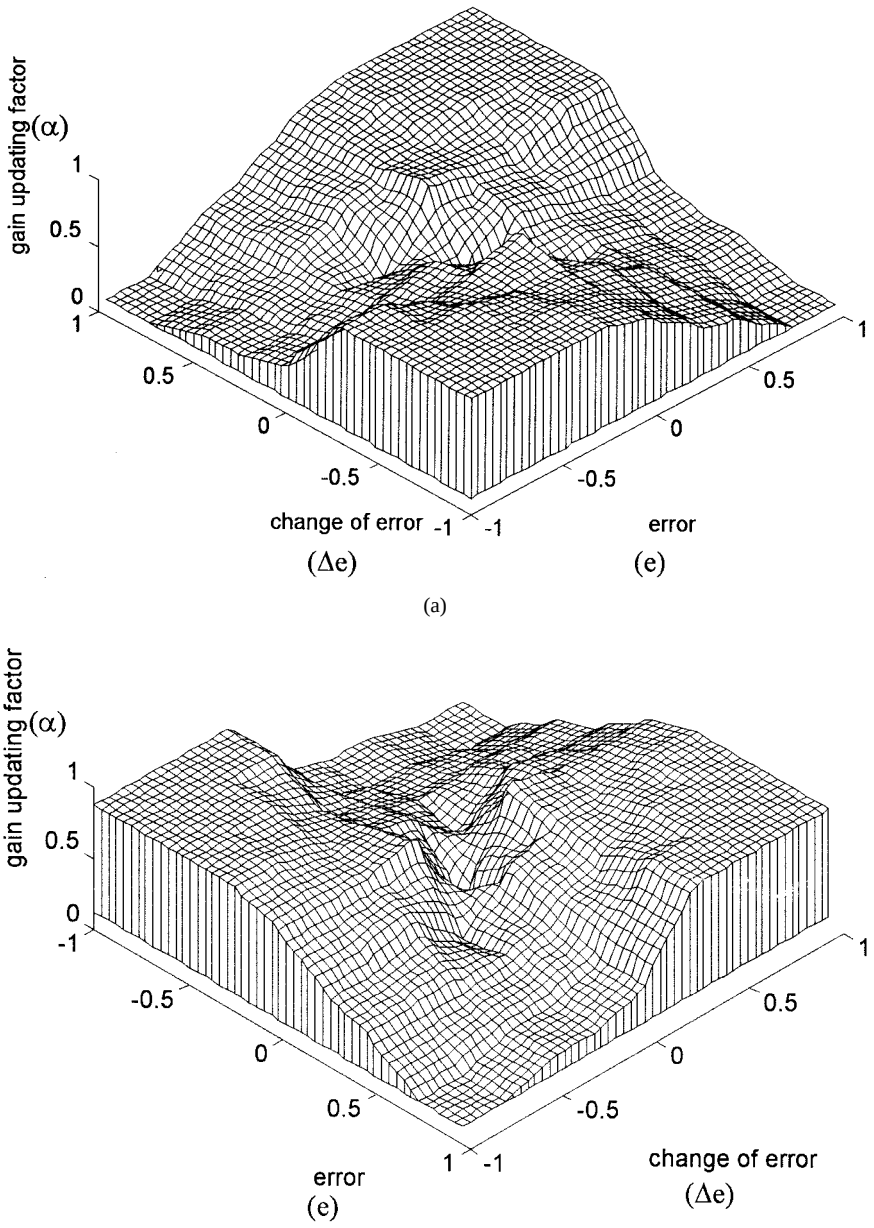


Fig. 4. (a) Variation of gain updating factor (α) with error (e) and change of error (Δe). (b) Variation of gain updating factor (α) with error (e) and change of error (Δe) [90° rotated from Fig. 4(a)].

times greater than conventional FLC) becomes permissible due to the factor α which always lies in (0,1]. But such a large value of G_u for the conventional FLC makes either the control performance unacceptable or the system uncontrollable.

Step 3: Fine tune the rules for α depending on the type of response wanted to achieve and based on the considerations described in Section II-C. For example if one wants to further reduce the overshoot at the cost of increased rise time then the value of α should be kept very small up to the medium values of e. This can be achieved by a rule of the form: IF e is PM and Δe is NS THEN α is VS [not S as shown in Fig. 3(b)]. However, for all our simulation results reported in the next section, we did not require any fine tuning (Step 3).

Observe that, in the presence of some training data, systematic methods like gradient descent may be developed for such tuning. In this study, we do NOT use any training data and do not tune any of the parameters based on data, rather, we use the most natural type of unbiased MF's. The rule-base for α is designed based on an intuitive analysis of the desired system performance and an often used control rule base [Fig. 3(a)]. We shall see in the next section that even with such rule bases, the proposed self-tuning scheme exhibits an excellent performance for widely different types of systems.

III. Results

In this section, we show the simulation results for some typical second-order linear as well as nonlinear processes using both of the proposed self-tuning FLC's. A second-order process with dead time is fairly common and also a useful

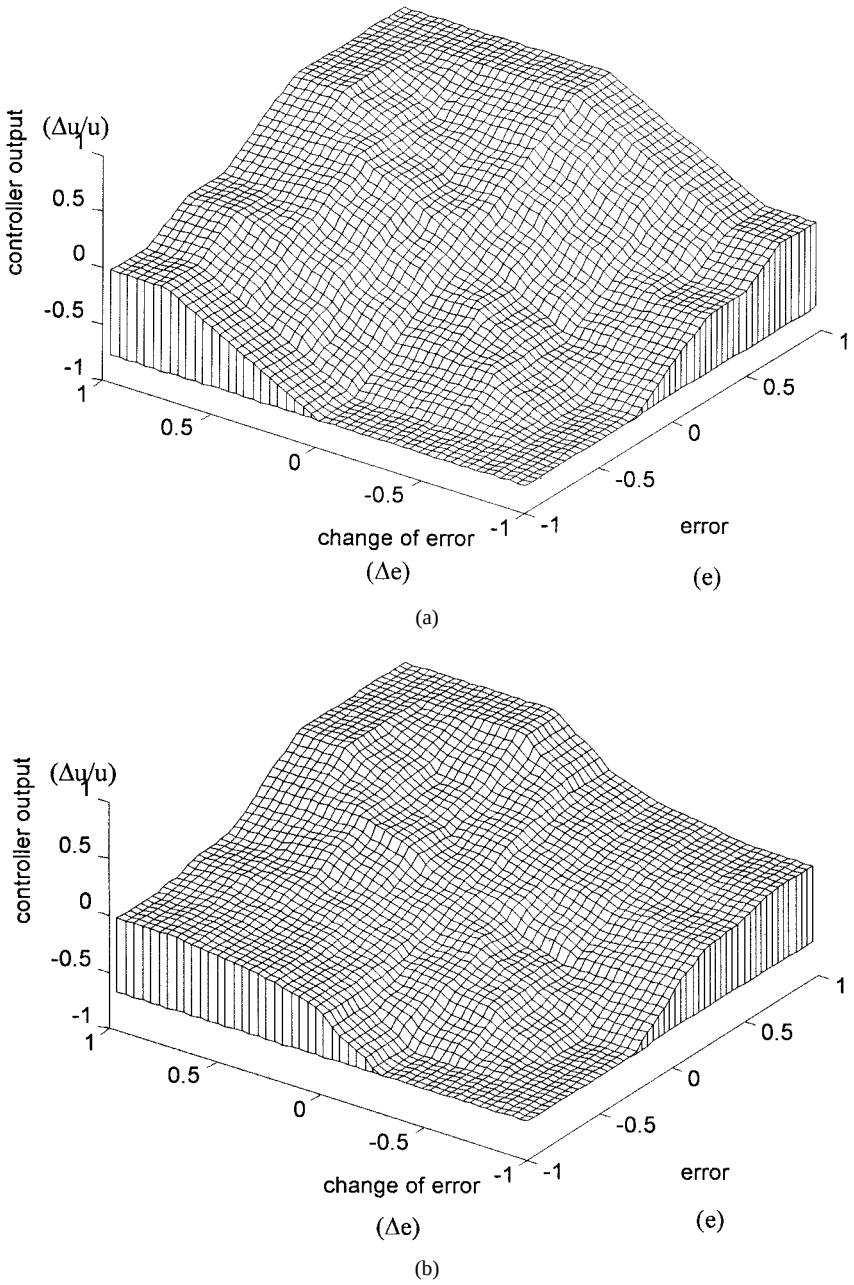


Fig. 5. (a) Control surface of the conventional FLC. (b) Control surface of the proposed self-tuning FLC.

model for many practical processes [29]. The performances of the two proposed FLC's (STFPIC and STFPDC) are compared with those of their corresponding conventional FLC's (FPIC and FPDC). For a clear comparison between the conventional and self-tuning FLC's several performance measures such as, peak overshoot $(\%OS)$, settling time (t_s), rise time (t_r), integral absolute error (IAE), and integral-of-time-multiplied absolute error (ITAE) [32] are used. The values of different performance indexes are provided in different tables for each process separately. Since peak overshoot and rise time usually conflict each other they may not be reduced simultaneously. If one of them is made smaller, the other tends to become larger. Unlike [6], [21], in the results to come readers will find that, rise times for both conventional and self-tuning FLC's are maintained almost at the same value but with a considerably reduced overshoot and much improved overall performance in case of self-tuning FLC's. The two integral criteria IAE and ITAE are considered because mere visual observations of response curves are not always enough to make a good comparison between different types of controllers. Large errors contribute heavily to IAE; on the other hand ITAE penalizes heavily errors that occur late in time. Thus, IAE and ITAE reflect the transient and steady-state characteristics of a control system, respectively. To make a complete study of the relative performances of the two types of FLC's (conventional and self-tuning), each process is tested with step set-point change as well as load disturbance. To establish the robustness of the proposed scheme we use the SAME rule bases (Fig. 3) and MF's (Fig. 2) for ALL processes with different values of dead time. In all cases, Mamdani type inferencing [8] and height method [5] of defuzzification are used. We have also used the center of sums method

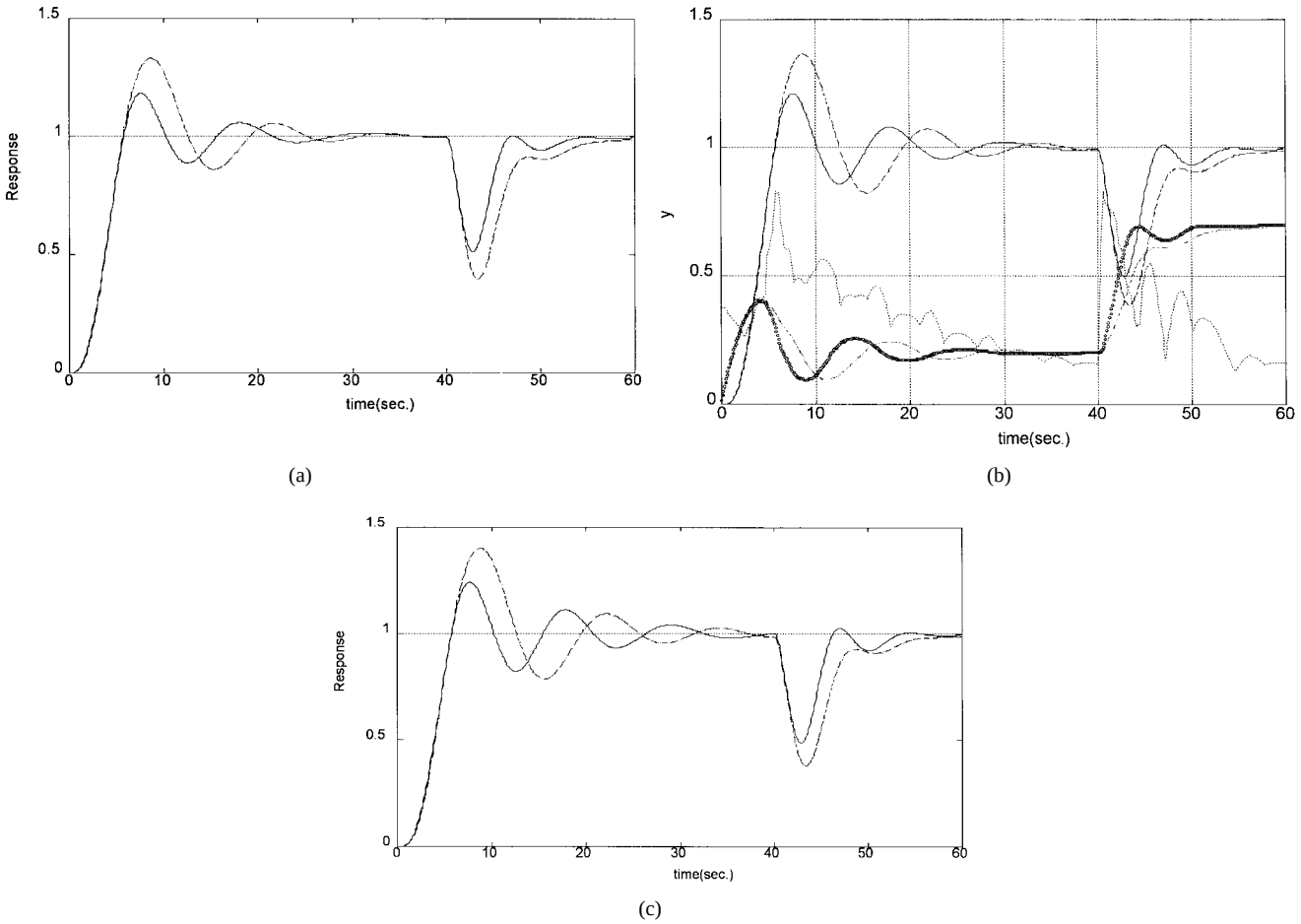


Fig. 6. (a) Responses of the second-order linear system in (8) with $L = 0.1$ for FPIC $(--)$ and STFPIC (—). (b) Responses of the second-order linear system in (8) with $L = 0.2$ for FPIC $(--)$ and STFPIC (—) with corresponding variation of α $(\cdots)$ and u $(-\cdot-)$ of FPIC, and u $(\circ\circ\circ\circ\circ)$ of STFPIC. (c) Responses of the second-order linear system in (8) with $L = 0.3$ for FPIC $(--)$ and STFPIC (—).

[5] of defuzzification (similar to the well-known center-of-gravity method) while conducting simulation analysis, but could not find any significant difference in control performance between these two different defuzzification methods. With a view to maintaining the simplicity of the controller and to avoid the extra computational burden, the height method of defuzzification has been used which results in a very simple but reliable and faster algorithm. For the numerical integration we use fourth-order Runge–Kutta method with an interval of 0.1 s. for the simulation of all processes. We now elaborate the performance analysis of different processes for PI-type (i.e., FPIC and STFPIC) as well as PD-type (i.e., FPDC and STFPDC) FLC's.

A. Performance Analysis for the PI-Type FLC's

1) Second-Order Linear Process: The process transfer function $G_p(s)$ is

$$G_p(s) = e^{-LS}/(s^2 + s + 0 \cdot 2). \tag{8}$$

For this process we consider four different values of dead time (L), i.e., $L = 0$, 0.1, 0.2, and 0.3 with $G_e = 0.9$ and $G_{\Delta e} = 13.5$. Fig. 6(a)–(c), respectively, show the responses of (8) for $L = 0.1$, 0.2, and 0.3 under STFPIC and FPIC due to both step set-point change and load disturbance applied at $t = 40$ s in each case. Various performance indexes for (8) with four different values of L are listed in Table I. Fig. 6(b) also includes a typical highly nonlinear variation of α for the process in (8) with $L = 0.2$ and the corresponding variation of the process input or controller output (u) for both STFPIC and FPIC. Fig. 6(b) clearly conforms to the desired variation of α (i.e., small value for big error, wide variation around the set point and large value at the event of load disturbance) as described earlier in Section II-C. Fig. 6(b) also shows the marked differences between the outputs of the two controllers. The control action of STFPIC is found to be more aggressive than that of FPIC, especially at the event of load disturbance (appearing at $t = 40$ s). As a result, STFPIC shows very good regulation against load variations compared to FPIC. Results [Fig. 6(a)–(c) and Table I] for (8) with large parametric variation (dead time) reveal the consistently better performance of STFPIC over FPIC.

2) Marginally Stable System: Let us consider a marginally stable system described by

$$G_p(s) = e^{-LS}/(s(s+1)). \tag{9}$$

This is a marginally stable system because one of its poles is at the origin $(s = 0)$ [32] and presence of dead time makes the system further difficult to control. Here also,

TABLE I
PERFORMANCE ANALYSIS FOR THE LINEAR SECOND-ORDER PROCESS IN (8)

L	G_u	FLC	%OS	t_s(sec.)	t_r(sec.)	IAE	ITAE
0	0.02	FPIC	29.98	23.6	5.5	8.052	163.80
	0.06	STFPIC	15.93	20.7	5.6	5.968	96.23
0.1	0.02	FPIC	33.18	28.4	5.5	8.461	169.43
	0.06	STFPIC	18.30	25.7	5.5	6.254	100.33
0.2	0.02	FPIC	36.55	29.7	5.5	8.933	176.42
	0.06	STFPIC	21.03	25.9	5.5	6.648	107.19
0.3	0.02	FPIC	40.12	35.2	5.5	9.475	185.10
	0.06	STFPIC	24.07	31.2	5.5	7.187	117.62

TABLE II
(a) PERFORMANCE ANALYSIS OF CONVENTIONAL CONTROLLERS (PI AND PID) AND FLCS (STFPIC AND FPIC) FOR THE MARGINALLY STABLE SYSTEM IN (9). (b) PERFORMANCE ANALYSIS FOR THE MARGINALLY STABLE SYSTEM IN (9)

L	k_p / G_u	FLC	%OS	t_s(sec.)	t_r(sec.)	IAE	ITAE
0.2	1.55	PI	99.04	42.6	1.6	10.329	134.78
	2.07	PID	75.41	8.7	1.4	2.885	7.93
	0.028	FPIC	60.86	32.7	4.6	8.330	92.89
	0.084	STFPIC	27.28	25.9	4.7	5.027	42.19
	0.50	PI	68.82	24.5	2.9	6.479	48.61
	0.70	PID	80.26	25.7	2.3	7.360	65.46

(a)

L	G_u	FLC	%OS	t_s(sec.)	t_r(sec.)	IAE	ITAE
0	0.05	FPIC	37.92	15.1	3.8	5.294	77.48
	0.15	STFPIC	11.80	12.9	3.9	3.731	42.44
0.1	0.035	FPIC	50.69	23.4	4.3	8.382	184.58
	0.105	STFPIC	20.34	19.0	4.4	5.142	88.89
0.3	0.022	FPIC	69.72	50.7	5.1	12.373	207.17
	0.066	STFPIC	33.21	34.9	5.1	6.374	73.49

(b)

four different values of L (i.e., 0, 0.1, 0.2, and 0.3) are used for the process in (9) with $G_e = 0.9$ and $G_{\Delta e} = 20$. Ziegler–Nichols tuning formula (empirically derived) is a well known scheme for determining good settings of PI and PID controllers for a wide range of common industrial processes [29]. But conventional controllers (PI or PID) cannot show good performance for marginally stable systems (systems with integration). Even Ziegler–Nichols-tuned PID controllers fail to provide a satisfactory performance for such systems due to excessively large overshoot—not acceptable in most cases [33]. As an illustration, Fig. 7(a) and (b) and Table II(a) show the comparative performance analysis of Ziegler–Nichols tuned PI ($K_p = 1.55$ and $T_i = 3.00$) and PID ($K_p = 2.07, T_i = 1.80$, and $T_d = 0.45$) controllers with STFPIC and FPIC for (9) with $L = 0.2$. Fig. 7(a) depicts an excellent performance of STFPIC over Ziegler-Nichols tuned PI controller; even conventional FLC (FPIC) is found to provide a reasonably good performance though the overshoot is still quite large. We see from Fig. 7(b) that even the PID (Ziegler–Nichols tuned) controller produces too large an overshoot (75%) to be accepted, although the other performance measures exhibit improvement over STFPIC. In such situations, by reducing only the gain (K_p) of PI or PID controllers, we cannot reduce the overshoot to an acceptable limit. This fact can be justified from Fig. 7(c) and Table II(a) where the gain (K_p) of PI and PID controllers is reduced to almost one-third of their respective Ziegler–Nichols tuned value. With this reduced gain, for PI the overshoot reduces to 69%, but for PID it increases from 75–80% which is still much above the overshoot for STFPIC. Moreover, with this reduced gain, STFPIC is better or comparable to the PID controller with respect to all performance indexes except rise time.

Response characteristics of this system under STFPIC and FPIC for $L = 0.1$ and 0.3 are, respectively, shown in Fig. 7(d) and (e). Fig. 7(d) represents the responses of (9) due to both step set point and impulse load disturbance introduced at $t = 47.5$ s whereas Fig. 7(e) displays the responses only for set point change. Comparison of performances between FPIC and STFPIC is provided in Table II(b). Here, STFPIC remarkably reduces the $\%OS$ and t_s in each case. Table II(b) and Fig. 7(d) and (e) both reveal that due to the self-tuning mechanism the performance of STFPIC remains within the acceptable limit even when such a difficult process is associated with comparatively large dead time. For example, corresponding to

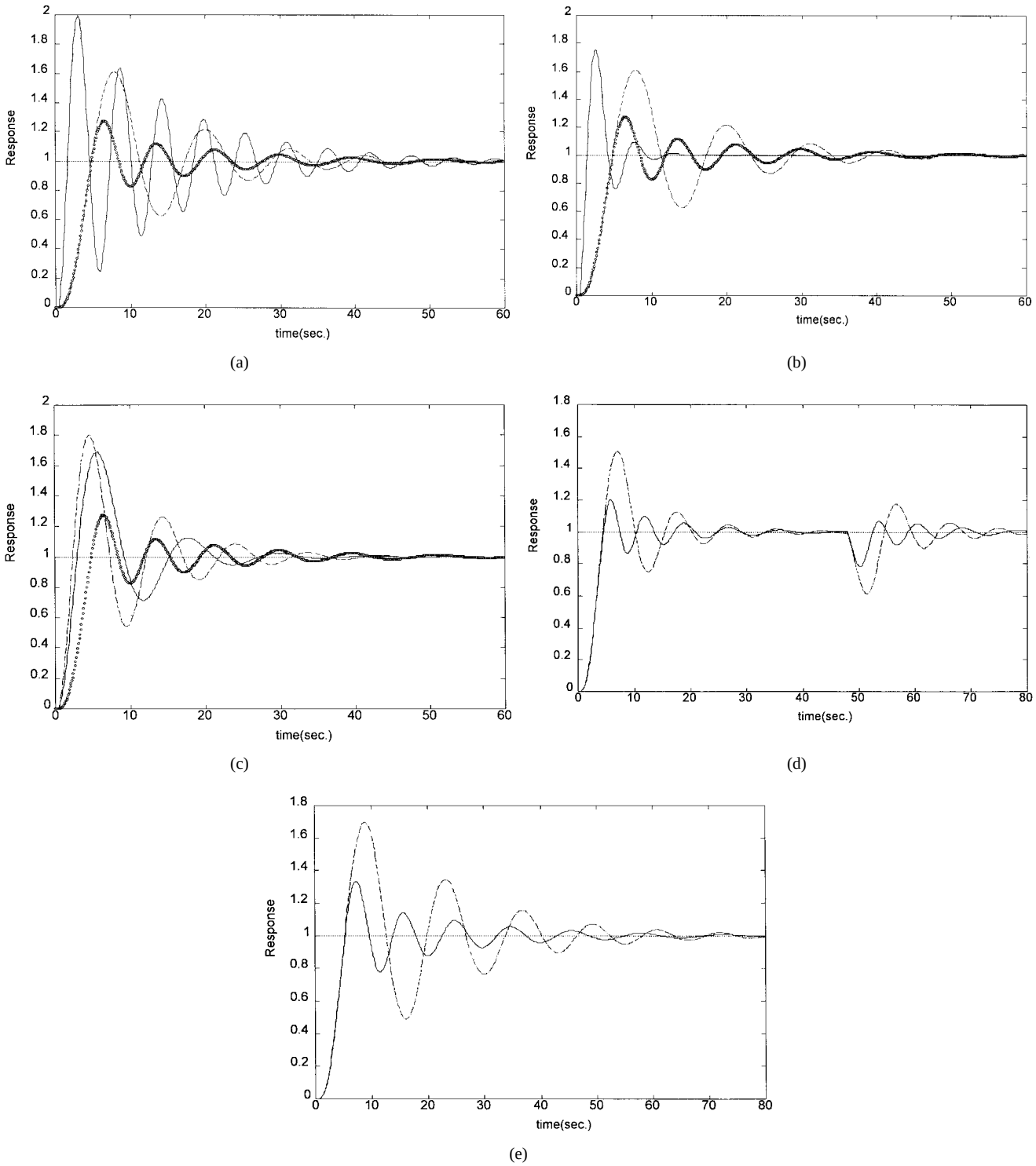


Fig. 7. (a) Responses of the marginally stable system in (9) with $L = 0.2$ for Ziegler–Nichols tuned PI (—), FPIC ($--$) and STFPIC ($\circ\ \circ\ \circ\ \circ\ \circ$). (b) Responses of the marginally stable system in (9) with $L = 0.2$ for Ziegler–Nichols tuned PID (—), FPIC ($--$) and STFPIC ($\circ\ \circ\ \circ\ \circ\ \circ$). (c) Responses of the marginally stable system in (9) with $L = 0.2$ for PI (—) and PID ($--$) with reduced gain (one third of their respective Ziegler–Nichols tuned value) and STFPIC ($\circ\ \circ\ \circ\ \circ\ \circ$). (d) Responses of the marginally stable system in (9) with $L = 0.1$ for FPIC ($--$) and STFPIC (—). (e) Responses of the marginally stable system in (9) with $L = 0.3$ for FPIC ($--$) and STFPIC (—).

$L = 0.3$, ITAE $= 207.17$ and $\%OS = 69.72$ for FPIC but for STFPIC they are 73.49 and 33.21, respectively [Table II(b)].

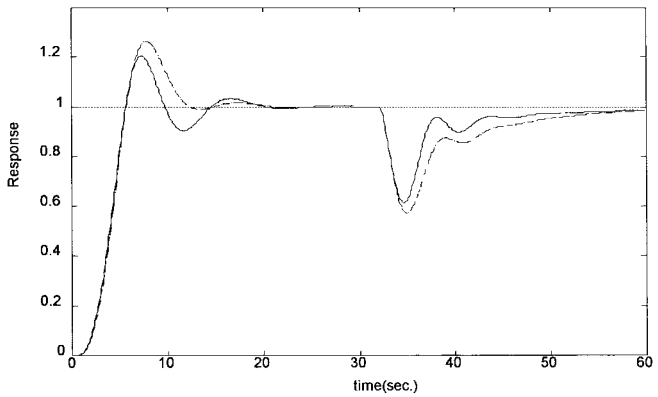
3) Second-Order Nonlinear Process: We have tested the performance of STFPIC for several nonlinear processes with dead time. Since the basic characteristics of the results are the same we report here only one of them. Consider the nonlinear process governed by

$$d^2y/dt^2 + dy/dt + 0 \cdot 25.y^2 = u(t - L). \tag{10}$$

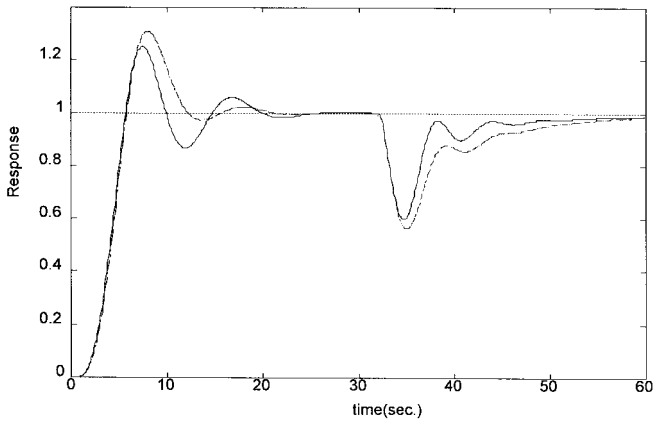
Responses of (10) due to set point as well as load disturbance for $L = 0.3$ and 0.5 are, respectively, shown in Fig. 8(a) and (b) with $G_e = 0.9$ and $G_{\Delta e} = 11$. Load

TABLE III
PERFORMANCE ANALYSIS FOR THE SECOND-ORDER NON-LINEAR PROCESS IN (10)

L	G_u	FLC	%OS	t_s(sec.)	t_r(sec.)	IAE	ITAE
0.1	0.018	FPIC	21.86	10.6	5.4	6.870	124.11
	0.054	STFPIC	16.17	12.7	5.4	5.863	89.51
0.3	0.018	FPIC	26.32	11.1	5.5	7.219	126.81
	0.054	STFPIC	20.37	13.3	5.5	6.175	90.27
0.5	0.018	FPIC	30.97	11.3	5.6	7.615	130.35
	0.054	STFPIC	25.22	14.80	5.5	6.609	92.96

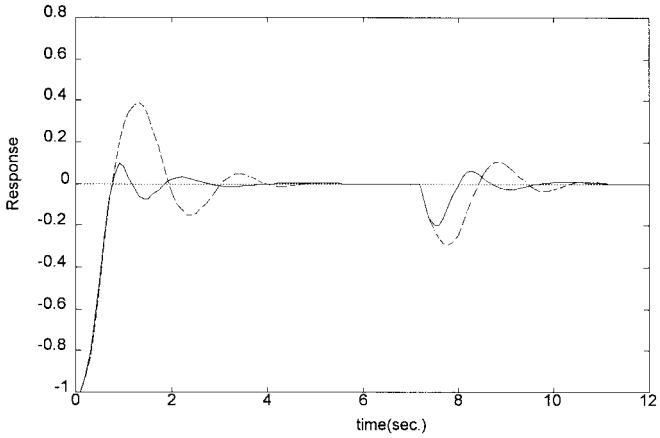


(a)

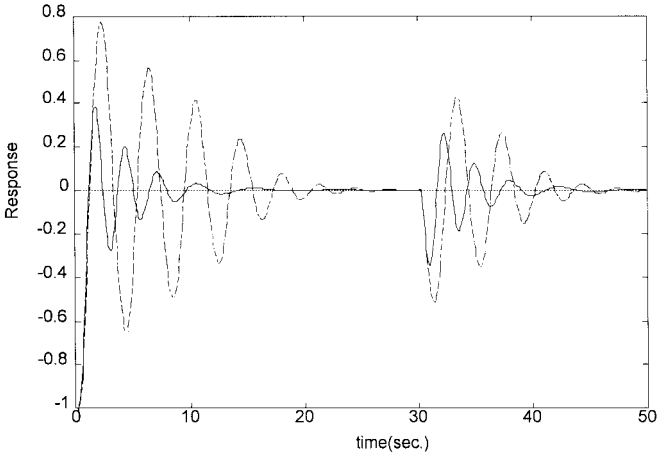


(b)

Fig. 8. (a) Responses of the second-order nonlinear system in (10) with $L = 0.3$ for FPIC $(--)$ and STFPIC (—). (b) Responses of the second-order nonlinear system in (10) with $L = 0.5$ for FPIC $(--)$ and STFPIC (—).



(a)



(b)

Fig. 9. (a) Responses of the nonminimum phase system in (11) with $L = 0$ for FPDC $(--)$ and STFPDC (—). (b) Responses of the nonminimum phase system in (11) with $L = 0.1$ for FPDC $(--)$ and STFPDC (—).

disturbances are applied at $t = 32$ s in both cases. Table III provides the quantitative performance analysis of STFPIC and FPIC for (10) with three different values of L. From the results, here also we find that the overall performance of STFPIC is always better for different values of L over its conventional counterpart (FPIC) though STFPIC produces little higher undershoot and t_s.

B. Performance Analysis for PD-Type FLC's

1) Nonminimum Phase System (Unstable System): We now consider an unstable system described by (11)

$$G_p(s) = -e^{-LS}/(s^2 - 1). \tag{11}$$

This system is unstable due to the presence of a nonminimum phase pole at $s = +1$. Actually, it's a simplified and linearized model of an inverted pendulum [34]. So PI-type controller is not applicable here since it introduces a pole at the origin $(s = 0)$. The PD-type controller introduces a zero in the left-half s plane. Therefore, by pole-zero cancellation the system may be stable under a PD controller [32]. Fig. 9(a) and (b) depict the responses of the system under STFPDC and FPDC (with $G_e = 0.9$ and $G_{\Delta e} = 2.7$) for $L = 0$ and 0.1, respectively. Impulse load disturbances appeared at $t = 7$ s in Fig. 9(a) while at $t = 30$ s in Fig. 9(b). Note

TABLE IV
PERFORMANCE ANALYSIS FOR THE NONMINIMUM PHASE SYSTEM (UNSTABLE) IN (11)

L	G_u	FLC	%OS	t_s(sec.)	t_r(sec.)	IAE	ITAE
0	9	FPDC	39.16	3.0	0.8	1.153	3.17
	27	STFPDC	9.89	1.7	0.8	0.680	1.36
0.1	4	FPDC	77.83	18.6	1.3	7.351	109.20
	12	STFPDC	38.64	7.7	1.2	2.64	35.03

TABLE V
PERFORMANCE ANLAYSIS FOR THE NONLINEAR PROCESS IN (12)

L	G_u	FLC	%OS	t_s(sec.)	t_r(sec.)	IAE	ITAE
0.3	0.7	FPDC	16.82	6.4	2.9	3.080	22.85
	2.1	STFPDC	3.35	3.0	3.0	2.189	10.38
0.5	0.7	FPDC	30.31	8.6	2.8	3.743	28.76
	2.1	STFPDC	15.49	7.6	2.8	2.352	18.03

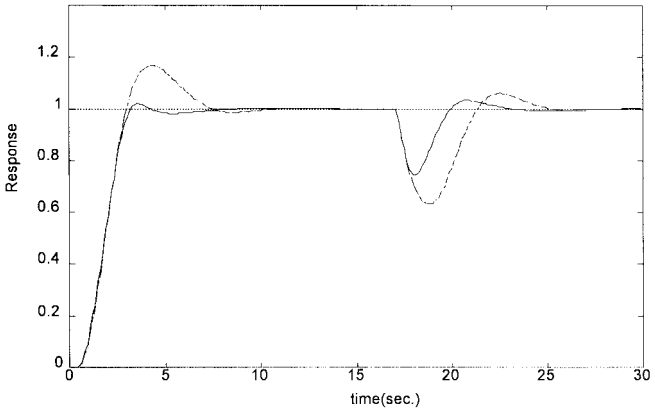
that, the performance of FPDC becomes unacceptable in the presence of a very small dead time $(L = 0.1)$, but in that situation too the performance of STFPDC is extremely good [Fig. 9(b)]. For further high values of L (say, $L = 0.3$), the system becomes uncontrollable even with the self-tuning FLC. Table IV shows the performance analysis of STFPDC and FPDC for (11). Fig. 9(a) and (b) and Table IV once again indicate a remarkably improved performance of STFPDC over FPDC both in transient and steady-state conditions.

2. Nonlinear Process: Several nonlinear processes are also used for the performance analysis of STFPDC. One such nonlinear process is described by the following:

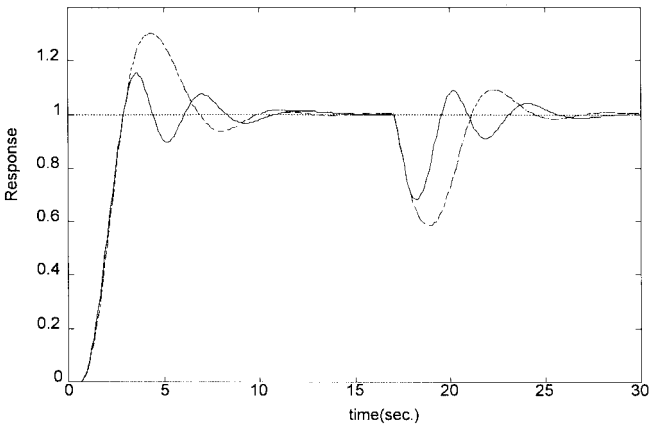
$$d^2y/dt^2 + 0 \cdot 3 \cdot y \cdot dy/dt = u(t - L). \tag{12}$$

The performances of STFPDC and FPDC (with $G_e = 0.9$ and $G_{\Delta e} = 11$) are tested for (12) even with a large dead time $(L = 0.5)$. Responses of (12) due to both step set point and impulse load disturbances (applied at $t = 17$ s) are presented in Fig. 10(a) and (b) for $L = 0.3$ and 0.5, respectively. Table V includes the various performance indexes. Again, we find an excellent performance of STFPDC compared to FPDC.

To summarize, the proposed scheme shows much improved performance over the conventional method for a wide variety of linear as well as nonlinear systems. Even in some cases it is seen to be more effective than the well-known Ziegler–Nichols tuning formula for conventional nonfuzzy controllers. Though, it is basically designed for time-invariant systems, it shows quite satisfactory performance for large parametric (dead time, the most difficult element among process parameters) variation; even when the FLC's remain fixed, i.e., without any change of their parameters including G_u as illustrated in Sections III-A.1, III-A.3, and III-B.2. This indicates that our scheme can also work for some processes with time varying dead time as long as its variation is not too large. To handle situations with large variation in dead-time special techniques like Smith predictor or analytical predictor (used in conventional control [29]) are yet to be developed for fuzzy control systems.



(a)



(b)

Fig. 10. (a) Responses of the second-order nonlinear system in (12) with $L = 0.3$ for FPDC $(--)$ and STFPDC (—). (b) Responses of the second-order nonlinear system in (12) with $L = 0.5$ for FPDC $(--)$ and STFPDC (—).

IV. CONCLUSION

We proposed a simple but robust model independent self-tuning scheme for FLC's. Here, the output SF, which may be considered equivalent to the controller gain was tuned on-line by fuzzy rules defined on e and Δe. The most important feature

of the proposed scheme is that it depends neither on the process being controlled nor on the controller used. Conceptually, this scheme differs from others in the literature as it attempts to implement the operator's strategy while running a plant. For example, some of the existing schemes attempt to attain a targeted levels for some of the performance indexes like overshoot and/or undershoot, while in the present case the objective is to mimic the operator's action which in turn is expected to result in the desired levels for various performance indexes. The proposed self-tuning scheme was applied to both PI- and PD-type FLC's for a wide range of different linear and nonlinear processes. Performances of self-tuning FLC's were also compared with those of their corresponding conventional FLC's with respect to several indexes such as peak overshoot, settling time, rise time, IAE, and ITAE, in addition to the responses due to set-point change and load disturbance and, in each case, the proposed scheme was found to outperform its conventional counterpart. Robustness of the proposed scheme was established by using the same rule bases and MF's for the simulation of all processes including even nonlinear and nonminimum phase processes with dead time. In order to further establish the effectiveness of the scheme we used the most natural and unbiased choices for MF's.

We have used two rule bases both defined on e and Δe. This raises a natural question: "Can we combine them?" Probably the answer is yes. One might think, this can be done by defining a different linguistic value for the controller output for each distinct combination of linguistic values for $u/\Delta u$ and α as demanded by Fig. 3(a) and (b). To make it clear, when e is, say, NM and Δe is PS then $u/\Delta u$ is NS [Fig. 3(a)] and α is S [Fig. 3(b)], so the pair (NS, S) gives a distinct combination. We can assume a rule of the form: IF e is NM and Δe is PS THEN $u/\Delta u$ is NSS where NSS is a new linguistic value. However, this approach will have several problems. These linguistic values may not have descent shapes such as triangle, etc. Their semantic interpretation as well as representation for implementation would be very difficult. Even if we use triangular membership functions, it may not be possible to use the most natural and unbiased choice of symmetric triangles with equal base and 50% overlap with neighboring MF's. Moreover, the highly nonlinear controller output is not only dependent on this (NS, S) combination, but also on its neighboring rules in both Fig. 3(a) and (b).

Another alternative may be to use system identification (SI) techniques through exploratory data analysis [35]–[40] when the controller outputs for different $e, \Delta e$ combinations are available. This actually needs data on the controller output which may be either supplied by an expert (which is difficult) or generated by our self-tuning scheme. Thus, a possible scheme may be as follows: first use our self-tuning controller to generate enough data and then use fuzzy clustering based schemes [35]–[40] for rule extraction. This will make the tuning scheme training data dependent. We plan to investigate this possibility in future.

However, the identification of the combined system would be much more difficult than that of two separate subsystems defined by the two rule bases. This can be easily illustrated as follows. Suppose the control surface obtained by the rule base in Fig. 3(a) is governed by $f_1(e, \Delta e) = a_1.e^2 + a_2.\Delta e^2 + a_3.e.\Delta e + a_4.e + a_5.\Delta e + a_6$ and the gain surface (Fig. 4) generated by the rule base in Fig. 3(b) is defined by $f_2(e, \Delta e) = b_1.e^2 + b_2.\Delta e^2 + b_3.e.\Delta e + b_4.e + b_5.\Delta e + b_6$. Hence, the combined control surface is defined by $g(e, \Delta e) = f_1(e, \Delta e) * f_2(e, \Delta e)$, which is a fourth-order polynomial of e and Δe. In this case identification of "g" requires estimation of only $6 + 6 = 12$ parameters, while if we directly want to identify "g" assuming a fourth-order polynomial (i.e., $g(e, \Delta e) = c_1.e^4 + c_2.\Delta e^4 + c_3.e^3.\Delta e + c_4.e.\Delta e^3 + c_5.e^2.\Delta e^2 + c_6.e^3 + c_7.\Delta e^3 + c_8.e^2.\Delta e + c_9.e.\Delta e^2 + c_{10}.e^2 + c_{11}.\Delta e^2 + c_{12}.e.\Delta e + c_{13}.e + c_{14}.\Delta e + c_{15}$) we have to identify 15 (i.e., more) parameters. Moreover, simultaneous estimation of a large number of parameters may increase the chance of getting stuck to some local minima as well as numerical instability.

The proposed self-tuning philosophy may possibly be applied for the tuning of input SF's or both input and output SF's simultaneously which may lead to achieve FLC's with more improved performances. Moreover, one may design a hybrid controller in which a fuzzy rule-based system will modulate the output of a nonfuzzy controller such as a Ziegler–Nichols tuned controller. The output modulation may be realized using a SF as done in the present case. But the same rule base [Fig. 3(b)] used for tuning the output SF of the fuzzy sliding mode controller may not be satisfactory in such cases because the control policies of conventional and fuzzy controllers are not identical. All these are currently under investigation.

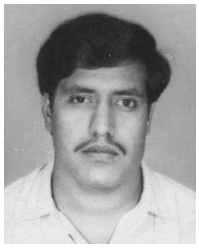
ACKNOWLEDGMENT

The authors would like to thank all referees for their valuable suggestions which helped us to improve this paper.

REFERENCES

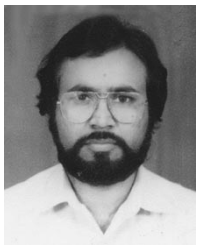
[1] M. Sugeno, *Industrial Applications of Fuzzy Control*. Amsterdam, The Netherlands: Elsevier, 1985.
[2] R. Palm, "Sliding mode fuzzy control," in *Proc. Fuzz IEEE*, San Diego, CA, 1992, pp. 519–526.
[3] C. J. Harris, C. G. Moore, and M. Brown, *Intelligent Control—Aspects of Fuzzy Logic and Neural Nets*. Singapore: World Scientific, 1993.
[4] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller—Parts I, II," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 404–435, Mar./Apr. 1990.
[5] D. Dirankov, H. Hellendorn, and M. Reinfrank, *An Introduction to Fuzzy Control*. New York: Springer-Verlag, 1993.
[6] S. Z. He, S. Tan, F. L. Xu, and P. Z. Wang, "Fuzzy self-tuning of PID controller," *Fuzzy Sets Syst.*, vol. 56, pp. 37–46, 1993.
[7] M. Maeda and S. Murakami, "A self-tuning fuzzy controller," *Fuzzy Sets Syst.*, vol. 51, pp. 29–40, 1992.
[8] T. J. Procyk and E. H. Mamdani, "A linguistic self-organizing process controller," *Automatica*, vol. 15, no. 1, pp. 53–65, 1979.
[9] S. Shao, "Fuzzy self-organizing control and its application for dynamical systems," *Fuzzy Sets Syst.*, vol. 26, pp. 151–164, 1988.
[10] Y. Park, U. Moon, and K. Y. Lee, "A self-organizing fuzzy logic controller for dynamic systems using a fuzzy auto-regressive moving average (FARMA) model," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 75–82, Feb. 1995.
[11] R. Tanscheit and E. M. Scharf, "Experiments with the use of a rule-based self-organizing controller for robotics applications," *Fuzzy Sets Syst.*, vol. 26, pp. 195–214, 1988.
[12] S. Galichet and L. Foulloy, "Fuzzy controllers: Synthesis and equivalences," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 140–148, May 1995.
[13] B. S. Moon, "Equivalence between fuzzy logic controllers and PI controllers for single input systems," *Fuzzy Sets Syst.*, vol. 69, pp. 105–113, 1995.

[14] H. Ying, W. Siler, and J. J. Buckley, "Fuzzy control theory: A nonlinear case," *Automatica*, vol. 26, no. 3, pp. 513–520, 1990.
[15] C. K. Chiang, H. Y. Chung, and J. J. Lin, "A self-learning fuzzy logic controller using genetic algorithms with reinforcements," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 460–467, Aug. 1997.
[16] C. L. Karr and E. J. Gentry, "Fuzzy control of PH using genetic algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 46–53, Feb. 1993.
[17] C. L. Chen and W. C. Chen, "Fuzzy controller design by using neural network techniques," *IEEE Trans. Fuzzy Syst.*, vol. 2, pp. 235–244, Aug. 1994.
[18] A. Homaifar and E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 129–139, May 1995.
[19] F. G. Shinskey, *Process Control Systems—Application, Design, and Tuning*. New York: McGraw-Hill, 1998.
[20] J. Lee, "On methods for improving performance of PI-type fuzzy logic controllers," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 298–301, Nov. 1993.
[21] H. A. Malki, H. Li, and G. Chen, "New design and stability analysis of fuzzy proportional-derivative control systems," *IEEE Trans. Fuzzy Syst.*, vol. 2, pp. 245–254, Nov. 1994.
[22] H. Nomura, I. Hayashi, and N. Wakami, "A self-tuning method of fuzzy control by decent method," in *Proc. Int. Fuzzy Syst. Assoc.*, Brussels, Belgium, 1991, pp. 155–158.
[23] L. Zheng, "A practical guide to tune of proportional and integral (PI) like fuzzy controllers," in *Proc. Fuzz IEEE*, San Diego, CA, Mar. 1992, pp. 633–641.
[24] M. Yoshida, Y. Tsutsumi, and T. Ishida, "Gain tuning method for design of fuzzy control systems," in *Proc. Int. Conf. Fuzzy Logic Neural Networks*, Fukuoka, Japan, 1990, pp. 405–408.
[25] S. Hayashi, "Auto-tuning fuzzy PI controller," in *Proc. Int. Fuzzy Syst. Assoc.*, Brussels, Belgium, 1991, pp. 41–44.
[26] T. Iwasaki and A. Morita, "Auto-tuning controller with fuzzy identification," in *Proc. Int. Conf. Fuzzy Logic Neural Networks*, Fukuoka, Japan, 1990, pp. 401–404.
[27] R. Palm, "Scaling of fuzzy controller using the cross-correlation," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 116–123, Feb. 1995.
[28] H. X. Li and H. B. Gatland, "Conventional fuzzy control and its enhancement," *IEEE Trans. Syst., Man, Cybern.*, vol. 26, no. 5, pp. 791–797, 1996.
[29] P. B. Deshpande and R. H. Ash, *Elements of Computer Process Control with Advanced Control Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.
[30] B. K. Bose, "Experts system, fuzzy logic, and neural network applications in power electronics and motion control," *Proc. IEEE*, vol. 82, pp. 1303–1323, Aug. 1994.
[31] R. Mudi and N. R. Pal, "A note on fuzzy PI-type controllers with resetting action," *Fuzzy Sets Syst.*, communication.
[32] K. Ogata, *Modern Control Engineering*. Englewood Cliffs, NJ: Prentice-Hall, 1970.
[33] K. J. Åström, C. C. Hang, P. Persson, and W. K. Ho, "Toward intelligent PID control," *Automatica*, vol. 28, no. 1, pp. 1–9, 1992.
[34] T. J. Ross, *Fuzzy Logic with Engineering Applications*. New York: McGraw-Hill, 1995.
[35] M. Sugeno and T. Yasukawa, "A fuzzy-logic based approach to qualitative medeling," *IEEE Trans. Fuzzy Syst.*, vol. 1, pp. 7–31, Feb. 1993.
[36] R. R. Yager and D. P. Filev, "Generation of fuzzy rules by mountain clustering," *J. Int. Fuzzy Syst.*, vol. 2, pp. 209–219, 1994.
[37] R. Babuska and U. Kaymak, "Application of compatible cluster merging to fuzzy modeling of multivariable systems," in *Proc. 2nd Eur. Congress Intell. Tech. Soft-Computing, (EUFIT'95)*, Aachen, Germany, 1995, pp. 565–569.
[38] T. A. Runkler and R. H. Palm, "Identification of nonlinear systems using regular fuzzy c-elliptotype clustering," *Proc. IEEE 5th Int. Conf. Fuzzy Syst.*, Piscataway, NJ, 1996, pp. 1026–1030.
[39] M. Delgado, A. F. Gómez-Skarmeta, and F. Martin, "A fuzzy clustering based rapid-prototyping for fuzzy modeling," *IEEE Trans. Fuzzy Syst.*, vol. 5, pp. 223–233, May 1997.
[40] N. R. Pal, K. Pal, J. C. Bezdek, and T. A. Runkler, "Some issues in system identification using clustering," *Proc. Int. Conf. Neural Network*, Houston, TX, June 1997, pp. 2524–2529.



Rajani K. Mudi received the B.Tech. and M.Tech. degrees in applied physics from the University of Calcutta, India, in 1990 and 1992, respectively.

Since 1992, he has been a Lecturer in the Department of Instrumentation Engineering, Jadavpur University, Calcutta, India. He was a Guest Lecturer in the Department of Applied Physics from 1993 to 1996 and the Department of Plastics and Rubber Technology at the University of Calcutta in from 1993 to 1995. His research interest includes intelligent instrumentation, fuzzy control and system identification.



Nikhil R. Pal (M'91) received the B.Sc. degree in physics and the Master of Business Management, from the University of Calcutta, India, in 1979 and 1982, respectively, and the M.Tech. (computer science) and the Ph.D. (computer science) degrees from the Indian Statistical Institute, Calcutta, in 1984 and 1991, respectively.

Currently, he is a Professor in the Machine Intelligence Unit of Indian Statistical Institute, Calcutta. From 1991 to 1993 and again from 1994 to 1994 he was with the Computer Science Department of the University of West Florida, Pensacola. He was a Faculty Guest of the University of Calcutta in 1991. His research interest includes image processing, pattern recognition, fuzzy sets theory, measures of uncertainty, neural networks, genetic algorithms, and fuzzy logic controllers. He is an associate editor of the *International Journal of Approximate Reasoning* and has been a Guest Editor of different special issues of *International Journal of Approximate Reasoning*, *International Journal of Uncertainty, Fuzziness and Knowledge Based Systems*, and *Fuzzy Sets and Systems*.

Dr. Pal is an Associate Editor of IEEE TRANSACTIONS ON FUZZY SYSTEMS.