A note on fuzzy PI-type controllers with resetting action

Rajani K. Mudi[a], Nikhil R. Pal[b,*]

[a] Department of Instrumentation Engineering, Jadavpur University, Salt-Lake Campus, Sector III, Block LB, Calcutta 700 091, India

[b] E & C.S.U., Indian Statistical Institute, 203, Barrackpore Trunk Road, Calcutta 700 035, India

Abstract

In this note we analyze the effect of the resetting operation in the fuzzy PI controller proposed by Lee (IEEE Trans. Fuzzy Systems 1(4) (1993) 298–301). We justify that the resetting operation can result in a steady-state error for different systems thereby destroying one of the important characteristics of PI-type controllers. We also exhibited through simulations with several second-order linear as well as non-linear systems with and without integration that the resetting operation makes the fuzzy PI controller behave more like a fuzzy PD controller, although for some systems the resetting operation can improve the transient response significantly.

Keywords: Fuzzy controller; Self-tuning; Resetting action; Error analysis

1. Introduction

Interests in fuzzy logic control has been growing very fast because of its simplicity and versatility. fuzzy logic controllers (FLC) have been reported to be successfully implemented in a number of complex and non-linear processes [21]. Sometimes FLCs are proved to be more robust than conventional controllers [18]. A comprehensive review of the FLC design and implementation can be found in [3,9,2]. Different types of adaptive FLCs such as self-tuning and self-organising controllers have also been reported in the literature [5,11,19]. Recently, many researchers are trying to achieve enhanced performance and increased robustness of FLCs, using neural network and genetic algorithms in designing such controllers [8,1,6].

The main objective behind any controller design is to regulate or maintain the desired output of the plant/system to be controlled. The designed controller is expected to change its output as often and as much as necessary to keep the controlled variable at the desired value, i.e., at the set-point. Failing to achieve this results in either an instability or a steady-state error, i.e., offset. Presence of an offset indicates the incapability of the controller to provide the required control action for achieving the desired plant output. Therefore, a good controller should at least guarantee system's stability as well as zero offset. Among the various types (PI, PD and PID types) of FLCs, just like the widely used conventional PI controller in process control systems, PI-type FLCs are most common and practical followed by PD-type

* Corresponding author. Tel.: +91-33-577-8085; fax: +91-33-577-6680.

E-mail address: nikhil@isical.ac.in (N.R. Pal).

FLCs. Because proportional (P) and integral (I) actions are combined in the proportional–integral (PI) controller to take advantages of inherent stability of proportional controllers and of elimination of offset by integral controllers. However, performance of PI-type FLC (FPIC) is known to be quite satisfactory for linear first-order systems. Like conventional PI controllers, performance of FPICs for higher-order systems and systems with large dead-time or transportation lag, and also for non-linear systems may be very poor due to large overshoot and excessive oscillation. And such systems may be ultimately uncontrollable [10]. PD-type FLCs (FPDC) are suitable for systems having integrating element in the feedforward path, where offset can be made zero [12], and they are not usually recommendable for systems without integration due to the presence of a large offset which is intolerable in most cases and very difficult to eliminate [14,13]. PID-type FLCs are rarely used due to the difficulties associated with the formulation of a comparatively larger rule-base and its tuning of more parameters.

A FLC has a fixed set of control rules usually derived from experts knowledge. The membership functions of the associated input and output linguistic variables are predefined on the respective universe of discourse. For the successful design of a FLC, proper selection of input and output scaling factors and/or tuning of the other controller parameters are critical jobs which in many cases are done through trial and error or based on some training data. A lot of research work on tuning of FLCs has been reported where either the input–output scaling factors or the definitions of fuzzy sets are automatically updated to match the current plant characteristics [23,16,4,22,15,7].

With a view to eliminating the overshoot caused by the accumulation of control input in a fuzzy PI controller (FPIC), Lee [10] has proposed two augmented versions of the conventional fuzzy PI controller using resetting factors (FPICR). The first of the two fuzzy controllers determines the resetting rate based on error (e) and error rate (Δe), while the second one uses error and controller output (u). The computation of the resetting factor is driven by a fuzzy rule-base. The controller remarkably improves the transient response of higher-order systems with integrating element. Here we analyze the effect of resetting action in the PI-type FLC and justify that such a controller (FPICR) essentially behaves like a fuzzy PD-type controller (FPDC). We establish it further through extensive simulation experiments on different higher-order linear as well as non-linear systems with and without integration.

2. Models of the fuzzy controllers

Here we consider the conventional fuzzy PI- and PD-type controllers. The conventional fuzzy PI controller (FPIC) is described by the equation

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

where in Eq. (1) k is the sampling instance and $\Delta u(k)$ is the incremental change in controller output, determined by fuzzy rules of the form

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

The fuzzy PD controller (FPDC), on the other hand uses rules of the form

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

Lee [10] augmented the fuzzy PI-type controller by a resetting factor $r; 0 \leqslant r \leqslant 1$ as

$$u(k+1) = (1 - (r(k))^p)u(k) + \Delta u(k).$$

Here p determines the non-linearity on the effect of r in the resetting operation. But the author in [10] neither provided the exact value of p used in simulation experiments nor recommended the suitable value/range of p for different types of systems. The model of the PI-type FLC proposed by Lee [10] is shown in Fig. 1. In Fig. 1, G_e, $G_{\Delta e}$ and G_u are the respective scaling factors for e, Δe and u. The resetting factor r is calculated using fuzzy rules of the form:

R_r: If e is E and Δe is ΔE then r is R.

In all cases Mamdani-type inferencing [19] and height method [2] of defuzzification are used. Lee [10] remarked that the controller performance is not much sensitive to the fuzzy partitions of the input–output space. Moreover, based on the results reported we could not reproduce the membership functions used by the author [10]. Hence we have used symmetric triangles with equal base and 50% overlap with neighboring membership function as shown in Fig. 2. This is the most natural and unbiased choice for membership functions. We use the same rule-base of Lee [10]

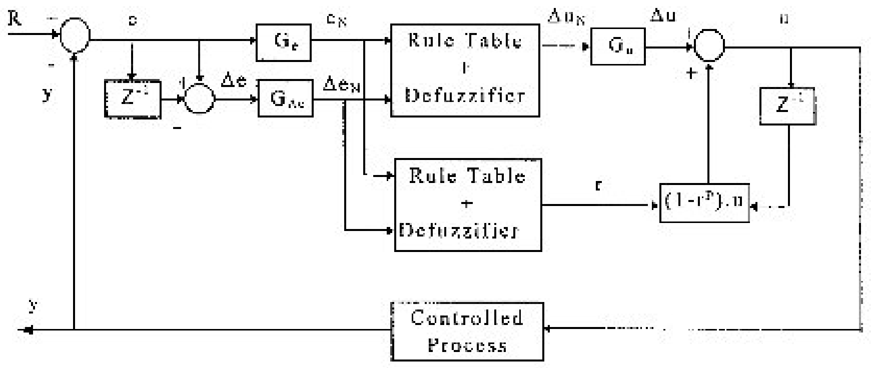
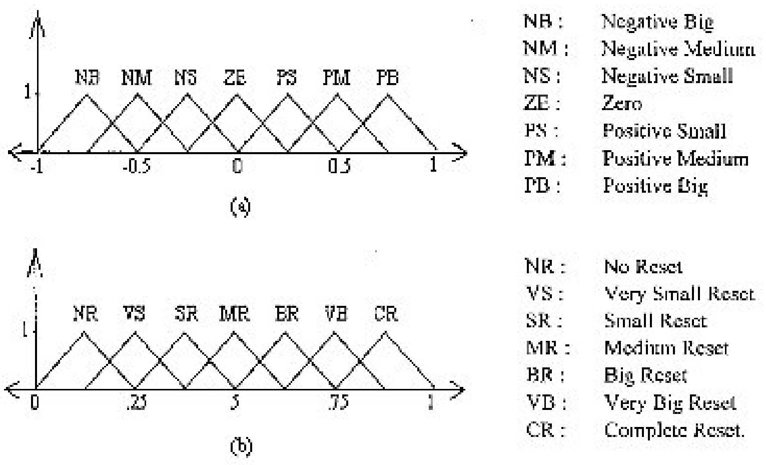


Fig. 1. Structure of the FLC proposed by Lee [10].



NB :	Negative Big
NM :	Negative Medium
NS :	Negative Small
ZE :	Zero
PS :	Positive Small
PM :	Positive Medium
PB :	Positive Big

NR :	No Reset
VS :	Very Small Reset
SR :	Small Reset
MR :	Medium Reset
BR :	Big Reset
VB :	Very Big Reset
CR :	Complete Reset.

Fig. 2. Membership functions of (a) e, Δe, u and Δu and (b) the resetting rate (r).

for computing Δu and r as shown in Figs. 3a and b, respectively. Note that for the FPDC we use the same rule-base in Fig. 3a where the consequent fuzzy memberships are defined on U, not on ΔU. Since Lee did not mention the exact/suitable value of p, we assume $p = 1$.

3. Steady-state error analysis

3.1. Qualitative analysis on steady-state error of FPICR [10]

The PI-type fuzzy controller [10] (FPICR) is described by

$$u(k+1) = (1 - (r(k))^p)u(k) + \Delta u(k).$$

This can be written as

$$u(k+1) = u(k) - [(r(k))^p u(k) - \Delta u(k)].$$

The value of r changes between 0 and 1, and value of Δu can vary over a certain range $[-a, a]$.

Now if

$$[(r(k))^p u(k) - \Delta u(k)] = 0$$

or

$$(r(k))^p u(k) = \Delta u(k) \qquad (2)$$

for some value of k then $u(k+1) = u(k)$.

Case 1: If condition (2) is satisfied when both e and Δe become zero, then $u(k+1) = u(k)$ will be the desired control input at steady state. In this situation no steady-state error (offset) will be present.

(a)

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

(b)

Δe / e	NB	NM	NS	ZE	PS	PM	PB
NB	NR	SR	BR	CR	BR	SR	NR
NM	NR	NR	MR	VB	MR	NR	NR
NS	NR	NR	VS	BR	VS	NR	NR
ZE	NR	NR	NR	NR	NR	NR	NR
PS	NR	NR	VS	BR	VS	NR	NR
PM	NR	NR	SR	VB	SR	NR	NR
PB	NR	VS	MR	CR	MR	VS	NR

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

Case 2: Suppose due to the resetting action after some time step k, $e(k)$ has a reasonable value but $\Delta e(k)$ is nearly zero. If, under this situation, condition (2) is satisfied then $u(k+1) = u(k)$. Since $\Delta e(k) \approx 0$, practically $e(k)$ will not change; i.e., $e(k+1) \approx e(k)$ and of course, $\Delta e(k+1) \approx \Delta e(k) \approx 0$. Since $e(k) \approx e(k+1)$ and $\Delta e(k+1) \approx \Delta e(k)$ the same set of rules will be fired at time step k and $(k+1)$, and hence $\Delta u(k+1) \approx \Delta u(k)$ and $r(k+1) \approx r(k)$. Therefore, $r(k+1)u(k+1) \approx \Delta u(k+1) \Rightarrow u(k+2) \approx u(k+1) \approx u(k)$. The process will thus maintain a steady-state error $e(k) = e(k+1) = e(k+2) = \cdots$. Thus we see that FPICR can maintain a steady-state error (offset) and this can happen with both step input and ramp input. In fact in the next section we are going to demonstrate that FPICR behaves like a FPDC both for ramp input and step input and this is true for a wide variety of processes. Since the basic structures of conventional controllers and their fuzzy versions are the same, the response characteristics of a given process under a particular type of fuzzy or conventional controllers are not expected to differ much. Hence, first we briefly analyze the steady-state behavior of conventional controllers.

3.2. Steady-state error analysis for linear systems

Consider a unity feedback control system as shown in Fig. 4. The open-loop transfer function of the system is $G(s) = G_c(s)G_p(s)$. Here $G_c(s)$ is the controller transfer function and $G_p(s)$ is the process or plant transfer function. For a PD control $G_c(s) = K_p(T_d s + 1)$ and for PI control $G_c(s) = (K_p/T_i s)(T_i s + 1)$, where K_p, T_d and T_i are, respectively, the proportional sensitivity, derivative time and integral time. Now $G(s)$ can be written as

$$G(s) = \frac{K}{s^q}\left(\frac{N(s)}{D(s)}\right) \quad \text{where } K \text{ is a constant},$$

$$N(s) = (z_1 s + 1)(z_2 s + 1)\ldots(z_m s + 1)$$

and

$$D(s) = (p_1 s + 1)(p_2 s + 1)\ldots(p_n s + 1).$$

Here $z_1, z_2, \ldots, z_m$ and $p_1, p_2, \ldots, p_n$ are the *zeros* and *poles* of $G(s)$, respectively. A system is called *type*-0, *type*-1, *type*-2,... if $q = 0, q = 1, q = 2, \ldots$, respectively. Therefore, a PI controller increases the systems' type by one but PD controller does not.

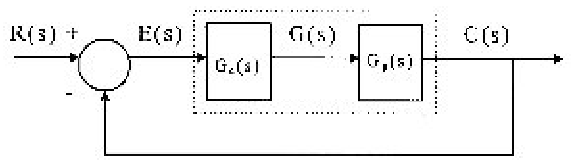


Fig. 4. Unity feedback control system.

Table 1
Steady-state error for different systems with different inputs

System \ Input	Step Input $r(t)=1$	Ramp input $r(t)=t$
Type-0	$1/[1+K]$	∞
Type-1	0	$1/K$
Type-2	0	0

Now the closed-loop transfer function of the unity feedback control system is

$$\frac{C(s)}{R(s)} = \frac{G(s)}{1+G(s)}$$

and

$$E(s) = R(s) - C(s)$$

or

$$\frac{E(s)}{R(s)} = \frac{1}{1+G(s)}.$$

Therefore,

$$E(s) = \frac{R(s)}{1+G(s)}.$$

From the *final value theorem* the steady-state error (e_{ss}) can be calculated as [17]

$$e_{ss} = \underset{s\to 0}{\mathrm{Lt}}\,[s.E(s)]$$

or

$$e_{ss} = \underset{s\to 0}{\mathrm{Lt}}\left[\frac{s.R(s)}{1+G(s)}\right].$$

For step input $R(s)=1/s$, and for unit ramp input $R(s)=1/s^2$. Now e_{ss} in terms of K for different types of systems can be calculated as shown in Table 1 [17].

4. Simulation analysis

Now we show simulation results for several second-order linear and non-linear systems with step as well as ramp input using FPIC, FPDC and FPICR in order to establish our hypothesis about the equivalent steady-state behavior of FPICR and FPDC. A second-order system is fairly common and also a useful model for many practical systems. In all figures (Figs. 5–13), responses for FPICR, FPDC and FPIC are, respectively, shown by *solid* (—), *dotted* (···) and *dashed* (--) lines, and the set point is shown by *dash-dot* (-·-) line. First we consider a few linear systems.

4.1. Linear systems

$$\text{(i)}\quad G_p(s) = \frac{1}{s(s+1)}. \tag{3}$$

The system described in (3) is associated with an integration and this is the same and only system as considered by Lee [10] while demonstrating the improved performance of fuzzy PI-type FLC with resetting action. In this case the open-loop transfer function for PD control will be $G(s)=K_p(T_d s+1)/s(s+1)$. This is a *type*-1 system, so for step input $e_{ss}=0$ and for ramp input $e_{ss}=1/K_p\neq 0$. Results obtained from FPICR and FPDC with step input are shown in Fig. 5. From these results it is seen that the performances of FPICR and FPDC are similar in nature.

For PI controller $G(s)=(K_p/T_i)(T_i s+1)/s^2(s+1)$ and this is a *type*-2 system. For this controller the steady-state error, e_{ss} should be zero (Table 1) for step response and this is indeed reflected in the result obtained by using FPIC as shown in Fig. 5. Fig. 5 shows that the performance of FPICR for step input is similar to that of FPDC. Note that the step response of FPICR is quite different from FPIC. Fig. 6 depicts the ramp responses corresponding to FPICR, FPDC and FPIC, respectively. Here again, the ramp responses of FPICR and FPDC are very similar, though there is a small difference in the values of the tracking errors. Moreover, the offset for FPIC, as expected, is zero, but for FPICR, like FPDC, there is a non-zero offset.

$$\text{(ii)}\quad G_p(S) = \frac{1}{s^2+s+k_1}, \tag{4}$$

where $k_1>0$ is a constant,

$$\begin{aligned} s^2+s+k_1 &= (s+p_1)(s+p_2) \\ &= (p_1 . p_2)\left(\frac{s}{p_1}+1\right)\left(\frac{s}{p_2}+1\right) \end{aligned}$$

when $p_1 . p_2 = k_1$, and $(p_1+p_2)=1$.

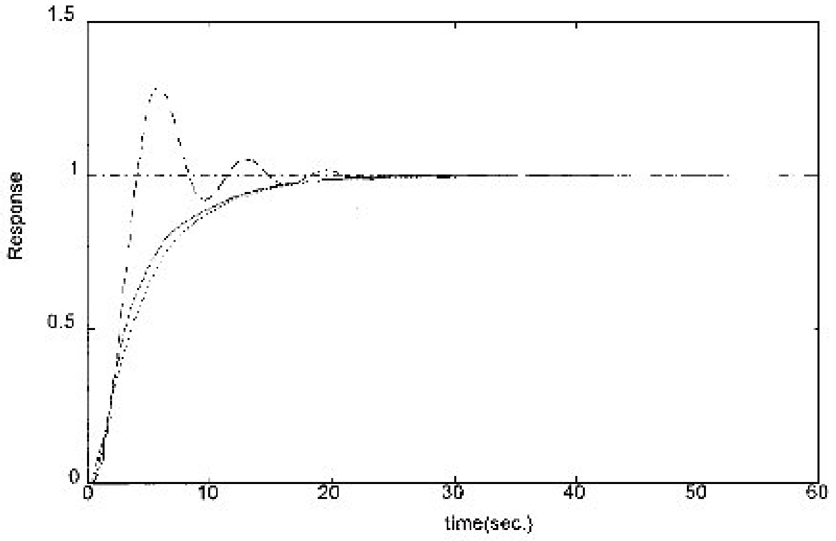


Fig. 5. Step responses of (3) for FPICR (—), FPDC (···) and FPIC (– –).

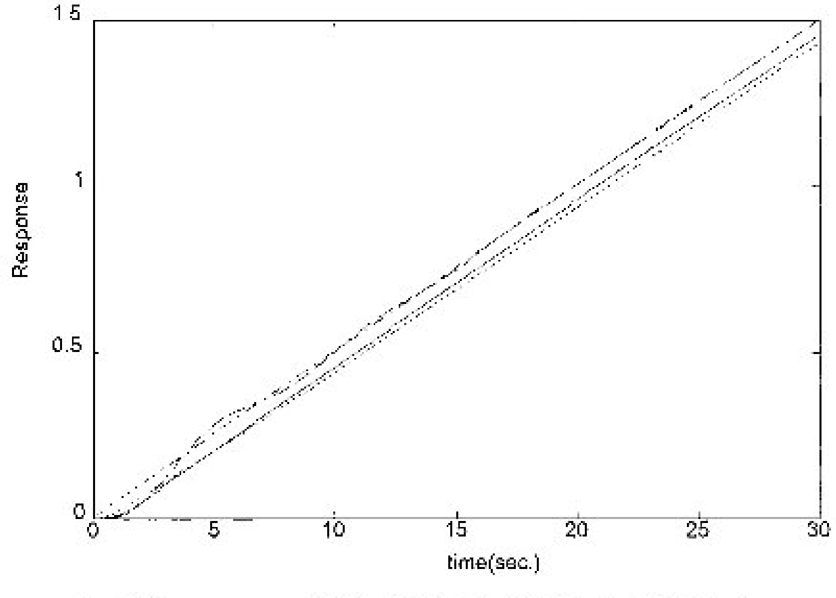


Fig. 6. Ramp responses of (3) for FPICR (—), FPDC (···) and FPIC (– –).

For a PD controller $G(s) = [(K_p/k_1)(T_d s + 1)/(s/p_1 + 1)(s/p_2 + 1)]$. This is a *type*-0 system, so for step input $e_{ss} = 1/(1+K)$ (Table 1), where $K = K_p/k_1$. On the other hand, for ramp input $e_{ss} = \infty$ (Table 1), which means e_{ss} will increase with time. Here as k_1 decreases e_{ss} decreases and when $k_1 = 0$, $e_{ss} = 0$, and in that situation the system becomes identical to that in (3).

For this system (without integration) we report results with $k_1 = 0.5$ and 0.2 with both ramp and step set point inputs. Fig. 7 displays the system's step responses with $k_1 = 0.5$ for three controllers. Here again the responses for FPICR and FPDC are similar in nature and in both cases there are some offsets though they are not equal. Fig. 8 shows the ramp responses of the same system ($k_1 = 0.5$). For both FPICR and FPDC

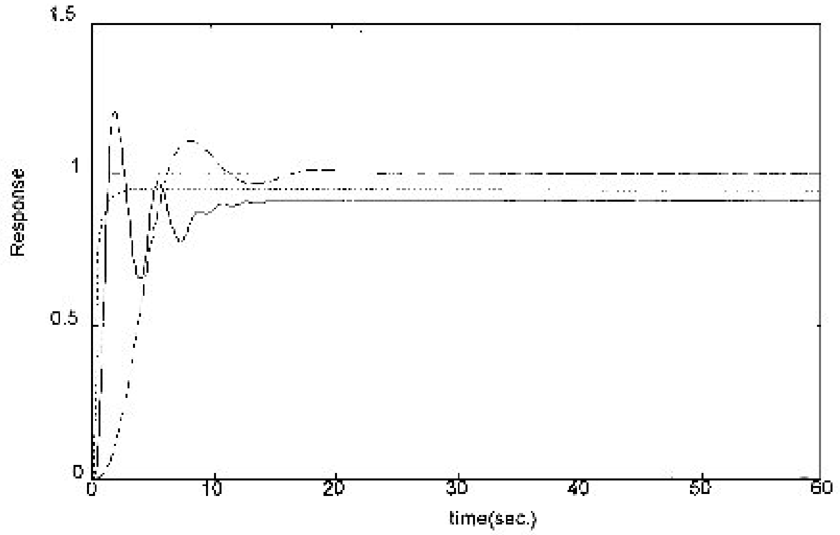


Fig. 7. Step responses of (4) with $k_1 = 0.5$ for FPICR (—), FPDC (· · ·) and FPIC (– –).

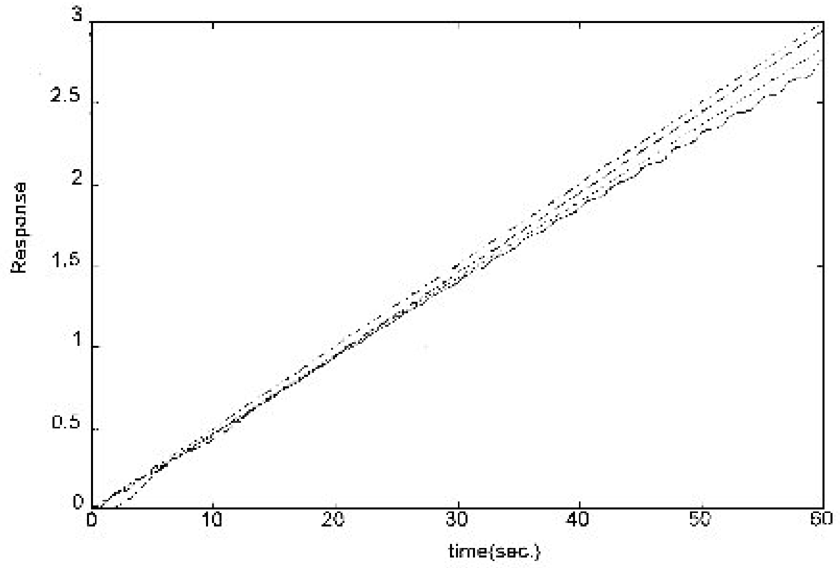


Fig. 8. Ramp responses of (4) with $k_1 = 0.5$ for FPICR (—), FPDC (· · ·) and FPIC (– –).

error is found to increase with time and in the limit it becomes infinite. For PI controller $G(s) = [(K/T_i)(T_i s + 1)/s(s/p_1 + 1)(s/p_2 + 1)]$ i.e., a *type*-1 system. As expected, for FPIC $e_{ss} = 0$ for step input (Fig. 7) and there is an offset for ramp input (Fig. 8).

Fig. 9 presents the step responses for the same system with $k_1 = 0.2$. The basic characteristics of the responses remain the same with those of $k_1 = 0.5$ but the offsets for FPDC and FPICR are smaller than that obtained with $k_1 = 0.5$. We have also experimented with other values of k_1 (like $k_1 = 1, 0.7$) and the results show that for this system the offset value increases with increasing k_1 for both FPICR and FPDC. Next we consider two non-linear systems.

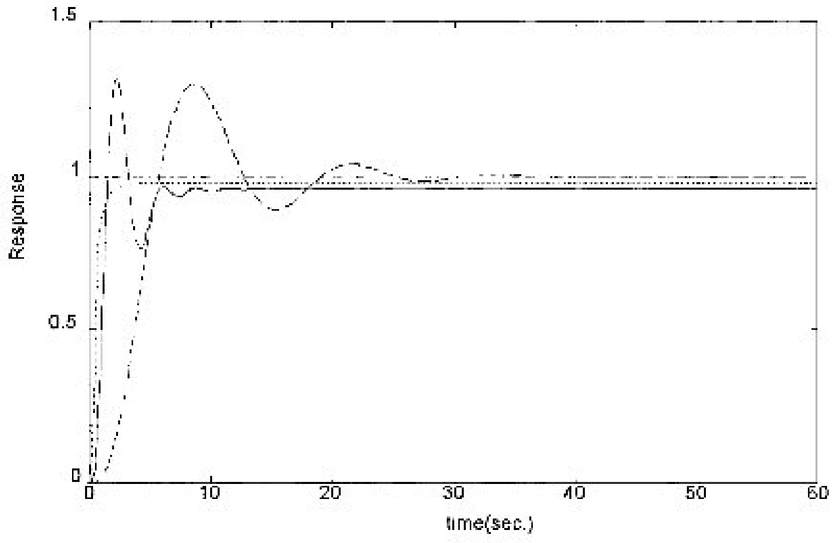


Fig. 9. Step responses of (4) with $k_1 = 0.2$ for FPICR (—), FPDC (· · ·) and FPIC (– –).

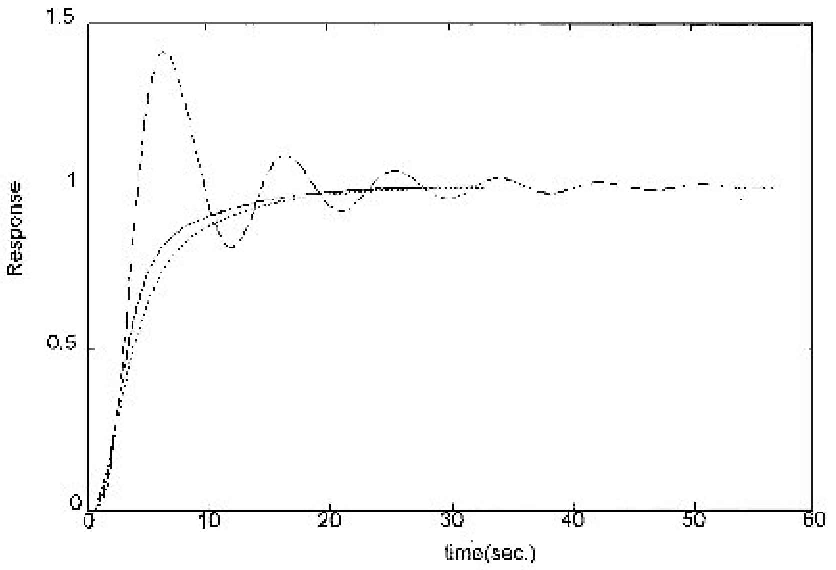


Fig. 10. Step responses of (5) for FPICR (—), FPDC (· · ·) and FPIC (– –).

4.2. Non-linear systems

We have examined the performances of FPICR for several non-linear second-order systems. Here we shall report only two of them. Since it is very difficult to get an exact analytical expression for the steady-state error (e_{ss}) of non-linear systems, we compare performances of the controllers with respect to step and ramp response characteristics only.

Figs. 10 and 11, respectively, show the step responses and ramp responses of the three controllers for the non-linear system with integration:

$$\frac{d^2y}{dt^2} + 0.6y\frac{dy}{dt} = u(t). \tag{5}$$

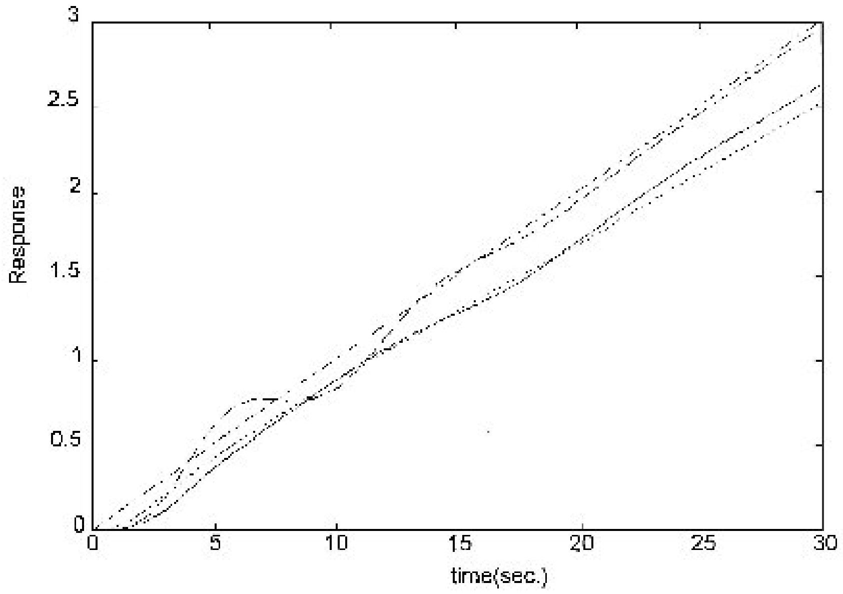


Fig. 11. Ramp responses of (5) for FPICR (—), FPDC (···) and FPIC (– –).

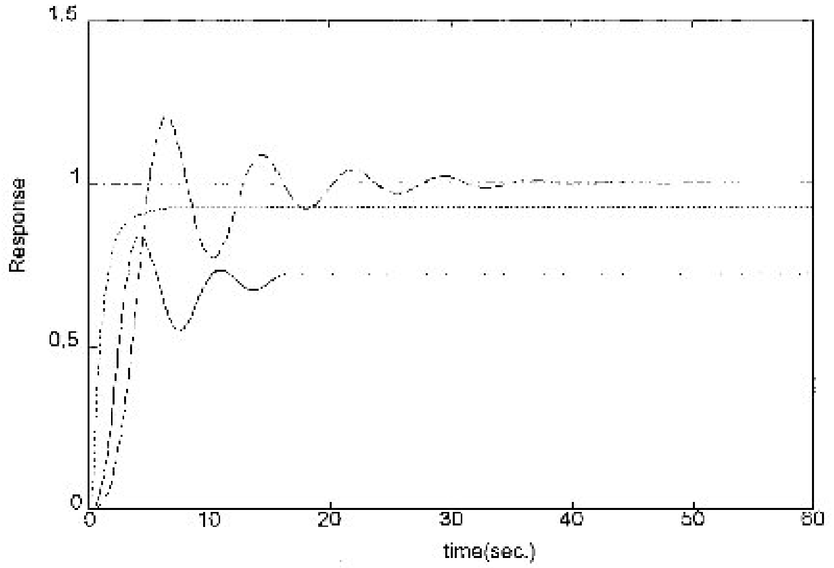


Fig. 12. Step responses of (6) for FPICR (—), FPDC (···) and FPIC (– –).

On the other hand, Figs. 12 and 13, respectively, represent the step responses and ramp responses for the non-linear system without integration:

$$\frac{d^2y}{dt^2} + 0.5\frac{dy}{dt} + 3y^2 = u(t). \tag{6}$$

From Figs. 10–13 once again we see that the performances of FPICR and FPDC are similar (though not identical) for non-linear systems too.

As mentioned earlier, the basic structures of various types of conventional controllers and their corresponding fuzzy controllers are the same, the response

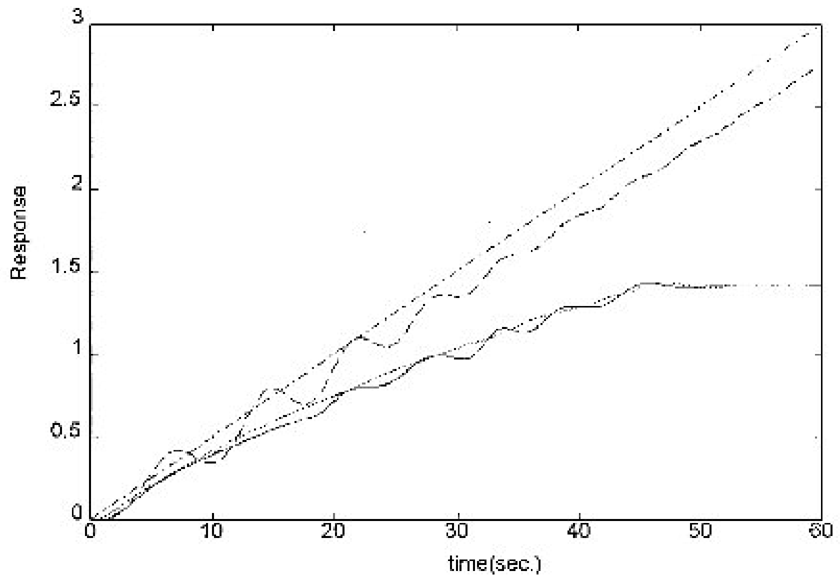


Fig. 13. Ramp responses of (6) for FPICR (—), FPDC (· · ·) and FPIC (– –).

Table 2
Summaries of steady-state errors for different systems under different controllers

Input / System	Step Input			Ramp Input		
	FPIC	FPDC	FPICR	FPIC	FPDC	FPICR
$\frac{1}{s(s-1)}$	0	0	0	0	Offset	Offset
$\frac{1}{s^2+s+0.5}$	0	Offset	Offset	Offset	∞	∞
$\frac{1}{s^2+s+0.2}$	0	Offset	Offset	Offset	∞	∞
$\frac{d^2y}{dt^2} - 0.6y\frac{dy}{dt} + u(t)$	0	0	0	Offset	∞	∞
$\frac{d^2y}{dt^2} + 0.5\frac{dy}{dt} + 3y^2 = u(t)$	0	Offset	Offset	∞	∞	∞

characteristics of a given system under a particular type of conventional and fuzzy controllers are not expected to differ much. And this fact is clearly reflected in the simulation results of various linear and non-linear systems under FPIC and FPDC. *But the FPICR of Lee* [10], *although presented as a PI-type fuzzy controller does not show the expected behavior. Rather it is found to behave like a fuzzy PD controller.*

Table 2 summarizes the results for different systems with three different types of fuzzy controllers. It clearly reveals that due to the resetting operation in FPICR, its behavior is more or less similar to that of FPDC. Although the resetting operation in FPICR can improve the transient response of some systems having integrating elements, it may lose a very important characteristics (zero offset for step response) of FPIC, hence, the basic purpose of using PI-type fuzzy controllers is lost.

5. Conclusions

We have analyzed the fuzzy PI controller with resetting operation proposed by Lee [10]. We have shown that the resetting operation augmented by Lee can destroy the desirable characteristics of PI-type controllers for some systems (systems without integration) though it can improve significantly the transient responses of some other systems (systems with integration). We have also demonstrated through simulations for both step and ramp responses of a wide variety of second-order linear as well as non-linear systems that the fuzzy PI-type controller with resetting action behaves more closely to a fuzzy PD-type controller.

Acknowledgement

N.R. Pal acknowledges the Department of Science and Technology, Govt. of India for partially supporting this work.

References

[1] C.L. Chen, W.C. Chen, Fuzzy controller design by using neural network techniques, IEEE Trans. Fuzzy Systems 2 (3) (1994) 235–244.

[2] D. Dirankov, H. Hellendoorn, M. Reinfrank, An Introduction to Fuzzy Control, Springer, New York, 1993.

[3] C.J. Harris, C.G. Moore, M. Brown, Intelligent Control-aspects of Fuzzy Logic and Neural Nets, World Scientific, Singapore, 1993.

[4] S. Hayashi, Auto-tuning fuzzy PI controller, Proc. IFSA'91, 1991, pp. 41–44.

[5] S.Z. He, S. Tan, F.L. Xu, P.Z. Wang, Fuzzy self-tuning of PID controller, Fuzzy Sets and Systems 56 (1993) 37–46.

[6] A. Homaifar, E. McCormick, Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms, IEEE Trans. Fuzzy Systems 3 (2) (1995) 129–139.

[7] C. Jung, C. Ham, A real-time tuning algorithm for fuzzy controller and its application for the nuclear power plant, Proc. 4th Internat. Conf. on Soft-computing — IIZUKA'96, Vol. 1, 1996, pp. 363–366.

[8] C.L. Karr, E.J. Gentry, Fuzzy control of PH using genetic algorithm, IEEE Trans. Fuzzy Systems 1 (1) (1993) 46–53.

[9] C.C. Lee, Fuzzy logic in control systems: fuzzy logic controller — Part I, II, IEEE Trans. Systems Man, Cybernet. 20 (2) (1990) 404–435.

[10] J. Lee, On methods for improving performance of PI-type fuzzy logic controllers, IEEE Trans. Fuzzy Systems 1 (4) (1993) 298–301.

[11] M. Maeda, S. Murakami, A self-tuning fuzzy controller, Fuzzy Sets and Systems 51 (1992) 29–40.

[12] H.A. Malki, H. Li, G. Chen, New design and stability analysis of fuzzy proportional derivative control systems, IEEE Trans. Fuzzy Systems 2 (4) (1994) 245–254.

[13] R.K. Mudi, K. Majumdar, N.R. Pal, D. Patranabis, A self-tuning PD-type fuzzy logic controller for zero-load processes, Proc. Internat. Conf. on Trends in Industrial Measurements and Automation TIMA-99, Chennai (India), 1999, pp. 337–346.

[14] R.K. Mudi, N.R. Pal, A self-tuning fuzzy PD controller, IETE J. Res. 44 (4&5) (1998) 177–189.

[15] R.K. Mudi, N.R. Pal, A self-tuning fuzzy PI controller, Fuzzy Sets and Systems 115 (2000) 327–338.

[16] H. Nomura, I. Hayashi, N. Wakami, A self-tuning method of fuzzy control by decent method, Proc. IFSA'91, 1991, pp. 155–158.

[17] K. Ogata, Modern Control Engineering, Prentice-Hall, Englewood Cliffs, NJ, 1970.

[18] R. Palm, Sliding mode fuzzy control, Proc. Fuzz IEEE, San Diego, 1992, pp. 519–526.

[19] T.J. Procyk, E.H. Mamdani, A linguistic self-organising process controller, Automatica 15 (1) (1979) 53–65.

[20] S. Shao, Fuzzy self-organising control and its application for dynamical systems, Fuzzy Sets and Systems 26 (1988) 151–164.

[21] M. Sugeno, Industrial Applications of Fuzzy Control, Elsevier Science, Amsterdam, 1985.

[22] K. Tanaka, M. Sano, A new tuning method of fuzzy controllers, Proc. IFSA'91, 1991, pp. 207–210.

[23] L. Zheng, A practical guide to tune of PI like fuzzy controller, Proc. Fuzz IEEE, 1992, pp. 633–640.