2006 IEEE International Conference on Systems, Man, and Cybernetics October 8-11, 2006, Taipei, Taiwan

Neuro-Fuzzy Implementation of a Self-Tuning Fuzzy Controller

R. K. Mudi, Chanchal Dey, and T. T. Lee, *Fellow, IEEE*

***Abstract*—A self-tuning fuzzy PI controller (STFPIC) is designed elsewhere, using parallelly operated two rule-bases; one control rule-base, and the other gain rule-base, each having 49 rules. The output scaling factor (SF) of STFPIC is modified online by a gain updating factor following an operator's strategy. STFPIC is found to provide significantly improved performance for a wide range of processes. This study is an attempt for neuro-fuzzy implementations of STFPIC with considerably lesser number of rules, which are complete and capable of realizing almost similar performance as that of STFPIC. We consider two different structures of the proposed neuro-fuzzy PI controller (NFPIC); called NFPIC-1 and NFPIC-2, having only 50 and 49 rules respectively against 98 original rules of STFPIC. NFPIC-1 is similar in structure to that of STFPIC with two parallel rule-bases, each having 25 rules, whereas, the structure of NFPIC-2 is same as that of a conventional fuzzy controller with a single rule-base. Effectiveness of the developed neuro-fuzzy controllers (NFPIC-1 and NFPIC-2) is demonstrated using second-order linear as well as nonlinear processes.**

I. Introduction

Fuzzy logic controllers (FLCs) are capable to provide satisfactory performances for both linear and nonlinear complex systems [1, 2]. Both fuzzy logic and neural networks are proved to be universal approximators. But in general fuzzy systems do not possess learning capability. On the other hand, neural networks (NN) can learn easily from environment. Neuro-fuzzy (NF) hybrid systems through the integration of these two complementary technologies have been successfully tested for many practical systems [3-6].

To make the conventional FLCs more powerful and robust, various types of self-tuning features are incorporated with them [7-10]. Mudi *et al.* [8] proposed a model independent scheme for constructing a self-tuning fuzzy PI controller (STFPIC). It describes a robust self-tuning scheme for FLC's, which would be applicable irrespective of the nature of the process and structure of the FLC [7]. In a continuous production process, a skilled human operator always tries to manipulate the controller output depending on the current process states *i.e.*, error (e) and change of error (Δe) to get the process optimally controlled. This gain manipulation strategy is too complex to implement mathematically. An effort has been made to incorporate the knowledge of a skilled operator with the help of 49 fuzzy rules [8]. These additional gain modifying rules are concurrently used with 49 control rules of STFPIC to make it a self-tuning controller with an overall improved performance. Attempts have been made through exploratory data analysis to extract a small set of new rules to realize similar performance as that of STFPIC [11, 12].

Keeping in mind the universal approximation property and learning power of back-propagation algorithm of neural networks, in this study, we try to develop NF models of STFPIC with a reduced number of rules, maintaining the same level of performance. We use the multi-layer feed-forward NN model and back propagation algorithm for the tuning of input-output membership functions (MFs) of the proposed neuro-fuzzy PI controller (NFPIC). We consider two different control structures with nearly 50% less rules than STFPIC. In the first case, NF models for control rule-base and gain rule-base are separately trained with only 25 rules each, and they are used in parallel like in STFPIC; we denote this controller as NFPIC-1. In the second case, NF model for the resultant control surface of STFPIC is realized by only 49 rules in place of the original 98 rules; we call it NFPIC-2. For performance comparison between developed controllers (*i.e.*, NFPIC-1 and NFPIC-2) and STFPIC different linear and nonlinear processes are tested under set point change and load variation with varying dead time. Simulation results show that NFPIC-1 and NFPIC-2 are capable of providing almost the same level of performance as that of STFPIC. Since, the present study is based on STFPIC, next we provide its brief description.

II. The STFPIC [8]

The simplified block diagram of the STFPIC is shown in Fig. 1a. The output SF of the controller is modified by a gain updating factor 'α', shown by the dotted boundary. Details of STFPIC are available in [8]. However, to make this study self-contained, various design aspects of STFPIC are briefly mentioned below.

Normalized MFs for inputs and output (*i.e.*, e_N, Δe_N, and Δu_N) of the controller (Fig. 1a) are defined on the common domain [-1, 1], whereas the MFs for α is defined on [0, 1]. Except at the two extreme ends, MFs are symmetric triangles with equal base and 50% overlap with neighboring MFs. The relationships between the SFs (G_e, $G_{\Delta e}$ and G_u) and input-output variables of the STFPIC are as follows:

This work is supported by National Science Council, Taiwan, R.O.C. under Grant No. NSC-94-2811-E-009-020.

R. K. Mudi is with the Electrical & Control Engineering Department, National Chiao-Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail : rkmudi@iee.jusl.ac.in).

Chanchal Dey is with the Department of Applied Physics, University of Calcutta, Kolkata-700009, India, (e-mail : cdaphy@caluniv.ac.in).

T. T. Lee is with the Electrical & Control Engineering Department, National Chiao-Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail : ttlee@cn.nctu.edu.tw).

$$e_N = G_e \times e,\ \Delta e_N = G_{\Delta e} \times \Delta e, \text{ and } \Delta u = (\alpha G_u) \times \Delta u_N. \quad (1)$$

The operation of a PI-type FLC is described by

$$u(k) = u(k\text{-}1) + \Delta u(k). \quad (2)$$

In Eqn. (2) k is the sampling instance and Δu is the incremental change in controller output, which is determined by the rules of the form, R_{PI} : If e is E and Δe is ΔE then Δu is ΔU. The rule-base for computing Δu is shown in Fig. 1b, defined on e and Δe. 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 α. The rule-base in Fig. 1c is used for the computation of α. This is designed in conjunction with the controller rule-base in Fig. 1b with a view to mimicking an operator's strategy, while running a plant. In STFPIC the required nonlinear controller output (Δu_{STFPIC}) is generated by modifying the output of a simple FLC (Δu_{FPIC}) with the updating factor α.

$$\text{i.e., } \Delta u_{STFPIC} \propto \alpha(\Delta u_{FPIC})$$

$$\text{or} \quad \Delta u_{STFPIC} = K\alpha(\Delta u_{FPIC}), \quad (3)$$

where K is the proportionality constant. Eqn. (3) indicates that the STFPIC is equivalent to a PI-type FLC (FPIC) with a dynamic gain.

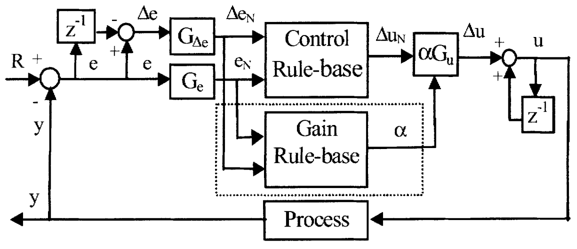


Figure 1a. Block diagram of the STFPIC

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

N=Negative, P=Positive, B=Big, M=Medium, S=Small, ZE=Zero

Figure 1b. Fuzzy rules for computation of Δu.

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

N=Negative, P=Positive, V=Very, B=Big, M=Medium, S=Small, ZE=Zero

Figure 1c. Fuzzy rules for computation of α.

III. Neuro-Fuzzy Implementation of STFPIC

The STFPIC uses total 98 rules in the two rule-bases; control rule-base, and gain rule-base with 49 rules each. Here, we utilize the universal approximation property of NN through its learning by back propagation algorithm to generate the nonlinear gain and control surfaces separately with 25 rules each. The corresponding NF controller, NFPIC-1 is shown in Fig. 2a. Also, a single NF model with 49 control rules is considered to realize the same resultant effect of the two rule-bases (Fig. 1a). Figure 2b shows its associated controller, NFPIC-2. These two NF controllers (*i.e.*, NFPIC-1 and NFPIC-2) are expected to provide similar performance as that of STFPIC.

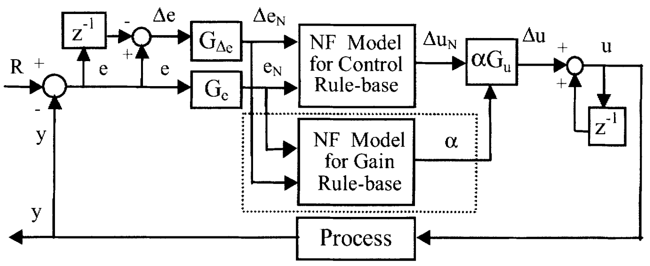


Figure 2a. Block diagram of the NFPIC-1

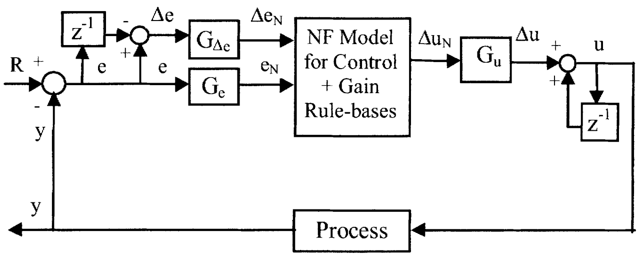


Figure 2b. Block diagram of the NFPIC-2

A. Data generation

Data set should cover the entire operating range of the system to be identified. If we generate the data by running a process in a closed loop then some specific rules will be used only. If we change the initial operating conditions then a new set of rules will be fired for the same process. In such cases identified system will lose its generalization property. To avoid this problem, we have generated data sets by uniform sampling of the entire input space. We have generated three data sets; $\{e, \Delta e, \Delta u\}$, $\{e, \Delta e, \alpha\}$, and $\{e, \Delta e, \alpha \times \Delta u\}$, each consists of 676 triplets. Where e and Δe are uniformly quantized within their normalized domain [-1, 1] as follows:

$$\forall i\,(0 \le i \le 25) \quad e = -1 + i \times 0.08$$
$$\text{and} \quad \Delta e = -1 + i \times 0.08. \quad (4)$$

The value of Δu or α for each pair of (e, Δe) is determined by the product inferencing and height method of defuzzification [13].

B. Neuro-fuzzy models and training

We use a five-layer NF structure as shown in Fig. 3, which is similar to that used in [3]. The two nodes in the first layer are the input nodes for two input variables e and Δe. During forward pass, two input nodes transmit the value of each (e, Δe) pair to the second layer. Each individual node of the second layer acts as a fuzzy set. The fuzzified values of the inputs (*i.e.*, e and Δe) or outputs of the second layer are fed to the third layer, which is the rule-node layer. Nodes in this layer represent the all possible fuzzy rules for each input pair (*i.e., e,* Δe). Links from this third layer to fourth layer perform the precondition matching of fuzzy rules. Nodes in the fourth layer perform fuzzy AND operation. After rule matching the output of the fourth layer is passed to the fifth layer or output node for defuzzification. The defuzzified output is obtained from the single output node of the fifth layer. Here connection weights between any two nodes of successive layers are unity. During training phase, the node in the fifth layer *i.e.,* the output node is used as the input node to feed the training samples as obtained from the relation (4). Detailed layer by layer operation of this neuro-fuzzy model is available in [3].

Initial bell-shaped MFs for NFPIC-1 are shown in Figs. 4a and 4b, and those for NFPIC-2 are shown in Fig. 4c. The following relation gives the definition of the membership function:

$$MF = \exp\left[-\frac{(x-\eta_i)^2}{\sigma_i^{\,2}}\right], \qquad (5)$$

where σ_i and η_i are respectively the initial width and center of the i^{th} MF for an input x (x may be any one of e, Δe, Δu and α). Initial MFs have more than 50% overlap with neighboring MFs as shown in Figs. 4. While designing NFPIC-1 and NFPIC-2, parameters of these MFs (*i.e.*, σ_i and η_i) are tuned through back-propagation algorithm similar to [3].

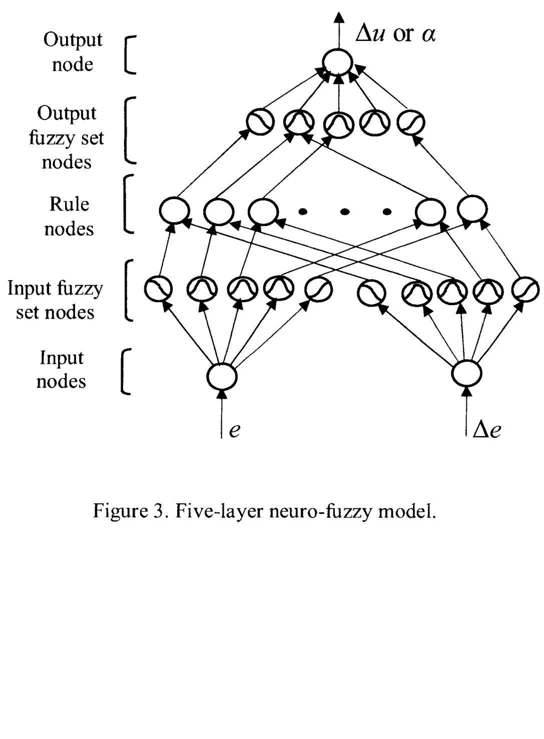
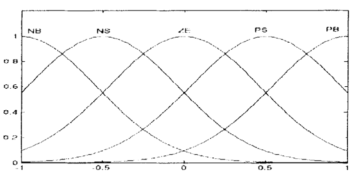
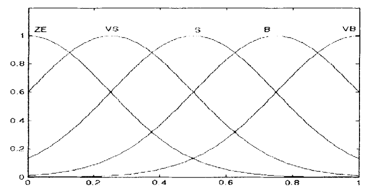


Figure 3. Five-layer neuro-fuzzy model.



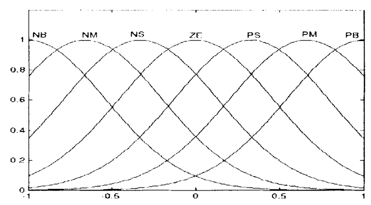
NB: Negative Big
NS: Negative Small
ZE: Zero
PS: Positive Small
PB: Positive Big

Figure 4a. Initial MFs for e, Δe and Δu of NFPIC-1.



ZE: Zero
VS: Very Small
S: Small
B: Big
VB: Very Big

Figure 4b. Initial MFs for α of NFPIC-1.



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

Figure 4c. Initial MFs for e, Δe and Δu of NFPIC-2.

To design the NFPIC-1 as shown in Fig. 2a, we train two sets of rules separately, each having 25 rules. One set of rule is trained to generate the control surface using five MFs {NB, NS, ZE, PS, PB} for each input-utput variables (*i.e.*, e, Δe and Δu) having same initial width ($\sigma = 0.65$) and center positions [-1.0, -0.5, 0.0, 0.5, 1.0] as shown in Fig. 4a. The other set of 25 rules is trained to realize the gain surface of STFPIC. In this case, the initial MFs for α, corresponding center positions and width are {ZE, VS, S, B, VB}, [0.0, 0.25, 0.5, 0.75, 1.0], and 0.35 respectively (Fig. 4b). Figures 5a and 5b depict the control surfaces of STFPIC and NFPIC-1 respectively. It appears that there is no significant difference between two surfaces, though NFPIC-1 uses almost 50% less rules than STFPIC. But, gain surfaces of STFPIC and NFPIC-1 as shown in Figs. 5c and 5d reveal some noticeable differences (though not significant) between them, specifically around steady state (*i.e.*, $e \approx 0$ and $\Delta e \approx 0$) . This indicates that highly nonlinear gain variation mechanism implemented through 49 rules in STFPIC is difficult to realize with a NF model using only 25 rules. Therefore, the close-loop response of NFPIC-1 is not expected to be very close to that of STFPIC. Figure 5e shows that the resultant control surface of STFPIC is smooth without any abruptness, unlike its gain surface (Fig. 5c). This property prompted us to develop NFPIC-2. In designing NFPIC-2, only 49 rules are trained to approximate the function of STFPIC having 98 rules. Here, seven equal width ($\sigma = 0.65$) MFs {NB, NM, NS, ZE, PS, PM, PB} with initial center positions [-1.0, -0.66, -0.33, 0.0, 0.33, 0.66, 1.0] are used (Fig. 4c). The control surface of NFPIC-2 as shown in Fig. 5f indicates its close resemblance with that of STFPIC (Fig. 5e), *i.e.,* NFPIC-2 will be expected to provide close-loop performance similar to that of STFPIC.

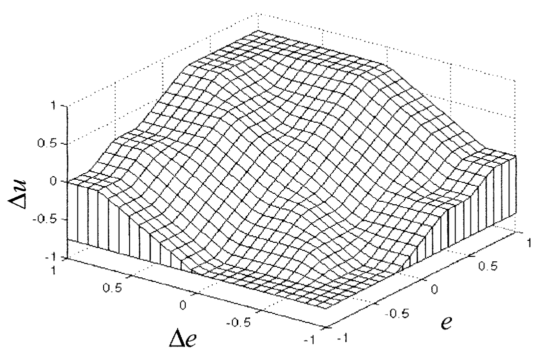


Figure 5a. Control surface of STFPIC (uses 49 rules).

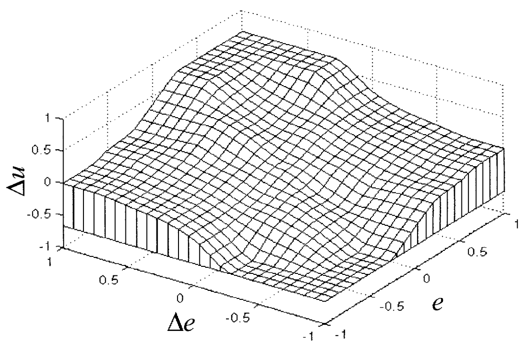


Figure 5e. Resultant control surface of STFPIC (uses total 98 rules; 49 control rules and 49 gain rules).

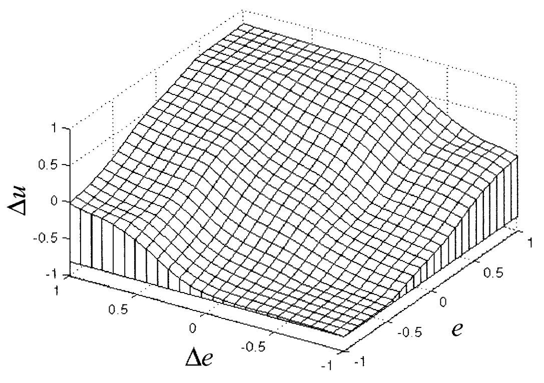


Figure 5b. Control surface of NFPIC-1 (trained by 25 rules).

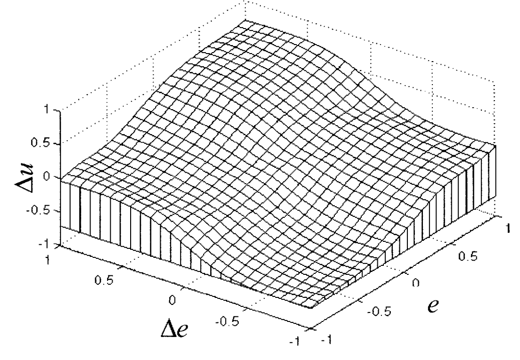


Figure 5f. Control surface of NFPIC-2 (trained by 49 rules).

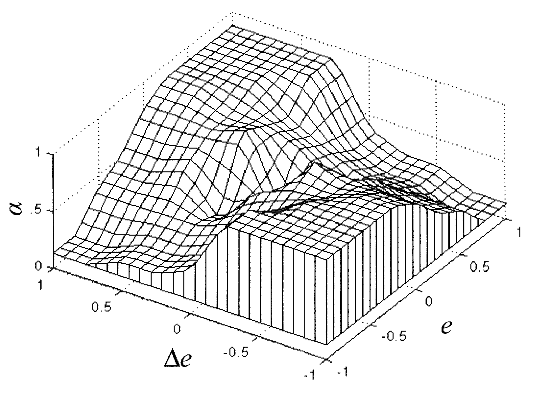


Figure 5c. Gain surface of STFPIC (uses 49 rules).

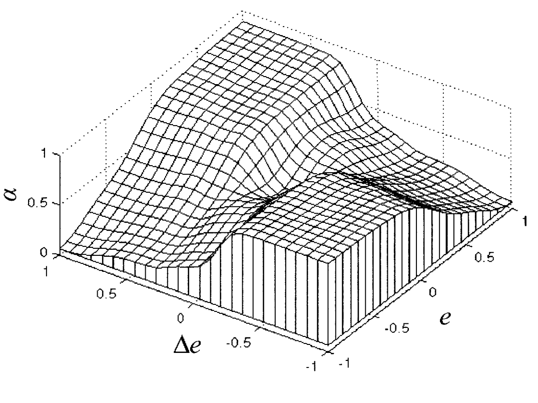


Figure 5d. Gain surface of NFPIC-1 (trained by 25 rules).

IV. Results

In order to investigate the close-loop performances of NFPIC-1 and NFPIC-2 having almost 50% reduced rules compared to STFPIC, we perform simulation study with second-order linear and nonlinear processes under set point change and load variation with varying dead time. We now present the performance analysis for individual processes.

A. Second-order linear process

$$\ddot{y} + \dot{y} + 0.2y = u(t - L) \qquad (6)$$

Response characteristics of this linear system (6) with L = 0.1s and 0.2s under NFPIC-1 and STFPIC are shown in Figs. 6a and 6b respectively. Figures 6c and 6d show the responses of (6) under NFPIC-2 and STFPIC. From the response curves (Figs. 6a-6d), it is found that both NFPIC-1 and NFPIC-2 exhibit similar performance as that of STFPIC. Although, NFPIC-2 follows STFPIC more closely compared to NFPIC-1. This fact can be justified from the comparisons of Fig. 6a with Fig. 6c, and Fig. 6b with Fig. 6d. The reason behind this difference between NFPIC-1 and NFPIC-2 is the presence of abruptness in the gain surface of STFPIC around steady state (Fig. 5c), which has been difficult to accurately model using only 25 rules (Fig. 5d) instead of 49 rules.

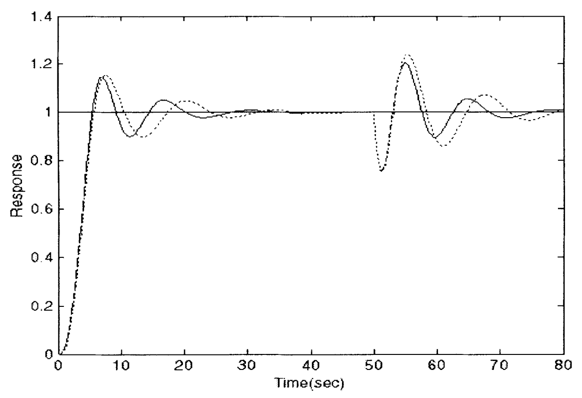


Figure 6a. Responses of (6) with $L = 0.1$s [—— STFPIC, ---- NFPIC-1].

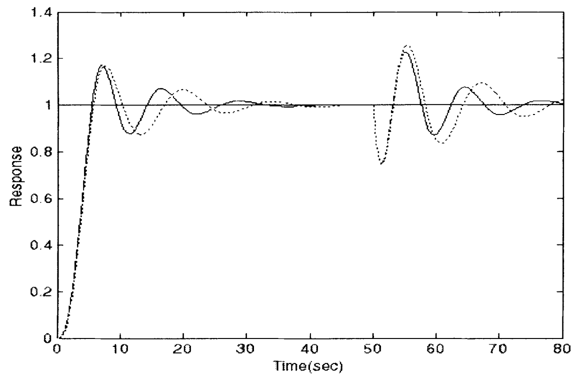


Figure 6b. Responses of (6) with $L = 0.2$s [—— STFPIC, ---- NFPIC-1].

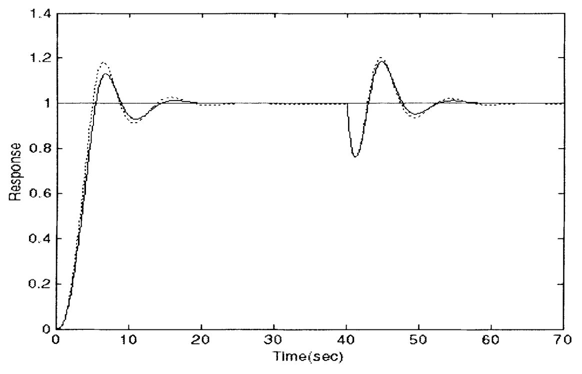


Figure 6c . Responses of (6) with $L = 0.1$s [—— STFPIC, ---- NFPIC-2].

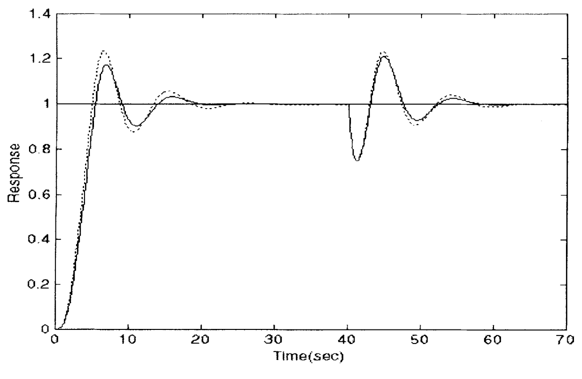


Figure 6d. Responses of (6) with $L = 0.2$s [—— STFPIC, ---- NFPIC-2].

B. Second-order nonlinear process

To establish the effectiveness of the proposed scheme, now we consider a nonlinear process described by

$$\ddot{y} + \dot{y} + 0.25y^2 = u(t-L)\,. \qquad (7)$$

This second order nonlinear process is tested with two different values of dead time. Figures 7a and 7b respectively show the response characteristics of (7) for $L = 0.1$s and 0.3s under NFPIC-1 and STFPIC. Responses due to NFPIC-2 and STFPIC are shown in the Figs. 7c and 7d. From results (Figs. 7a-7d), we observe that the overall performance of NFPIC-1 or NFPIC-2 is similar to that of STFPIC. As expected, like previous case, it is found that response characteristics of (7) under NFPIC-2 (Figs. 7c and 7d) more closely track those of STFPIC compared to NFPIC-1. For other linear and nonlinear processes, we have also found the similar results under different values of dead time.

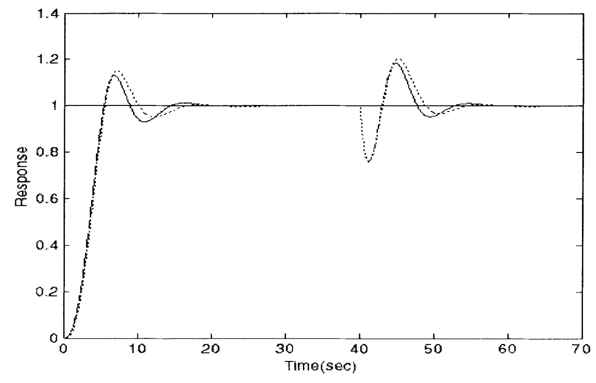


Figure 7a. Responses of (7) with $L = 0.1$s [—— STFPIC, ---- NFPIC-1].

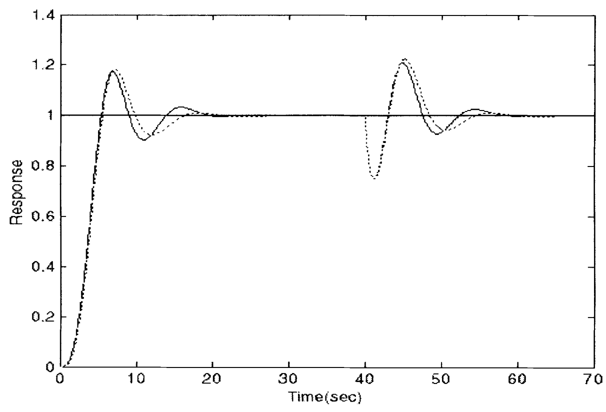


Figure 7b. Responses of (7) with $L = 0.3$s [—— STFPIC, ---- NFPIC-1].

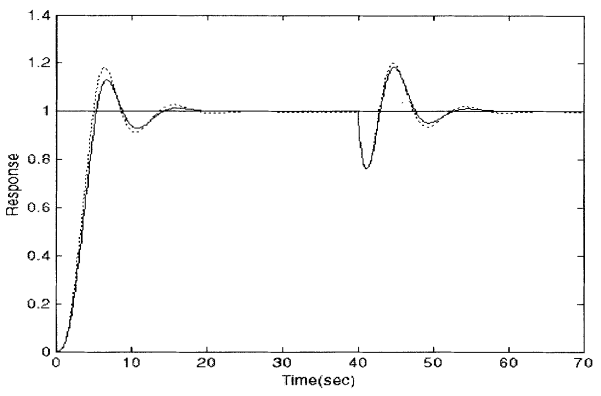


Figure 7c. Responses of (7) with $L = 0.1$s [—— STFPIC, ---- NFPIC-2].

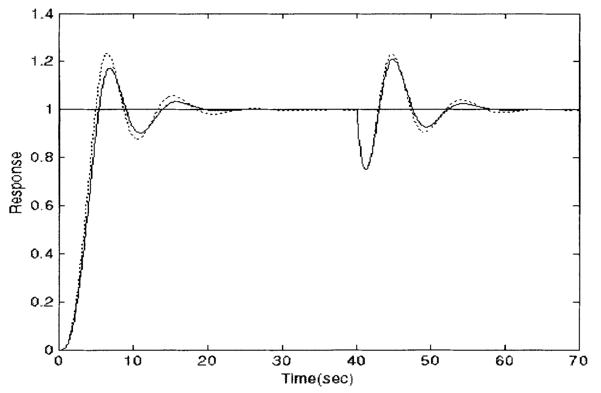


Figure 7d. Responses of (7) with $L = 0.3$s [—— STFPIC, ---- NFPIC-2].

V. Conclusion

We developed two neuro-fuzzy control structures, NFPIC-1 and NFPIC-2 with similar performance as that of a previously designed self-tuning fuzzy controller (STFPIC). NFPIC-1 and NFPIC-2 used almost 50% less rules than STFPIC. All the 98 rules of STFPIC were not equally important but it was difficult to identify which rules were to be removed and which were to be merged. With a view to realizing the similar performance of STFPIC by NF models with reduced number of rules, we utilized the learning capability of a multi-layer NN in designing NFPIC-1 and NFPIC-2. Here, widths and center positions of the input and output MFs were tuned by back propagation algorithm. Simulation experiments on different second-order linear and nonlinear processes with dead time clearly revealed that the overall performance of NFPIC-1 or NFPIC-2 is close to that of STFPIC, although NFPIC-2 is found to follow the STFPIC more closely compared to NFPIC-1. This study also revealed that it becomes difficult to accurately approximate a function having abruptness through a NF model with limited number of nodes in the hidden layer.

References

[1] M. Sugeno, *Industrial Applications of Fuzzy Control*, Amsterdam : Elsevier Science, 1985.

[2] Y. Zhao and E. G. Collins, "Fuzzy PI control design for an industrial weigh feeder", *IEEE Trans. on Fuzzy Syst.*, vol. 11, no. 3, pp. 311 - 319, 2003.

[3] C. T. Lin and C. S. Geroge Lee " Neural network based fuzzy logic control and decision system", *IEEE Trans. on Computers*, vol. 40, no. 12, 1991, pp. 1320 - 1336, 1991.

[4] T. Fukuda and T. Shibata, "Theory and applications of neural networks for industrial systems," *IEEE Trans. on Industrial Electronics*, vol. 39, no. 6, pp. 472-488, 1992.

[5] C. T. Lin, "A neural fuzzy control system with structure and parameter learning", *Fuzzy Sets and Systems.*, vol. 70, pp. 183 - 212, 1995.

[6] C. T. Lin, C. F. Juang, and C. P. Li, "Water bath temperature control with a neural fuzzy inference network", *Fuzzy Sets and Systems.*, vol. 111, pp. 285 - 306, 2000.

[7] R. K. Mudi and N. R. Pal, "A robust self tuning scheme for PI- and PD-type fuzzy controllers", *IEEE Trans. on Fuzzy Syst.*, vol. 7, no. 1, pp. 2 - 16, 1999.

[8] R. K. Mudi and N. R. Pal, "A self-tuning fuzzy PI controller," *Fuzzy Sets & Systems*, vol. 115, No. 2, pp. 327-338, 2000.

[9] C. Dey and R. K. Mudi, " Design of PI-type fuzzy controller with on-line membership function tuning", *Proc. International Conf. on Neural Information Processing, ICONIP – 2005*, Taipei, 2005.

[10] A. Chatterjee and K. Watanabe, " An adaptive fuzzy strategy for motion control of robot manipulators", *Soft Computing*, vol. 9, pp. 185 – 193, 2005.

[11] K. Pal, R. K. Mudi and N. R. Pal, "A new scheme for fuzzy rule based system identification and its application to self-tuning fuzzy controllers," *IEEE Trans. on Syst., Man, Cybern.*, vol. 32, no. 4, pp. 470-482, 2002.

[12] N. R. Pal, R. K. Mudi, K. Pal and D. Patranabis, "Rule extraction for self-tuning fuzzy controllers," *International Journal of Fuzzy Systems*, vol. 6, no. 2, pp. 71-80, 2004.

[13] D. Driankov, H. Hellendorn and M. Reinfrank, *An Introduction to Fuzzy Control*, NY: Springer-Verlag, 1993.