NEURO-FUZZY NETWORK BASED ADAPTIVE INTEGRATING COTNROL

C.W Chan

Department of Mechanical Engineering, The University of Hong Kong

Xiang-Jie Liu, Felipe Lara-Rosano

Centro de Instrumentos UNAM, Mexico City, D.F. 04510, Mexico

Abstract: A self-tuning neurofuzzy integrating controller is derived in this paper for offset eliminating purpose. CARIMA plant model is used and the control law produces integral control terms in a natural way. Neurofuzzy networks are chosen to implement the direct self-tuning nonlinear integrating controller. The performance of the self-tuning integrating neurofuzzy controller is illustrated by examples involving both linear and nonlinear systems . *Copyright©2002IFAC*

Keywords: Self-tuning control, neural networks, fuzzy control, nonlinear control

1. INTRODUCTION

In real-time process control, mostly there exist demand for the system to track the setpoint exactly. Usually this could not be achieved by "position" control. The integrating self-tuning controller, suggested by P.S. Tuffs(1985), aims at solving the offset problem in the control system. For the problem of offsets, integral action is incorporated in the controller. A common approach is to postulate a disturbance process which has stationary increments. CARIMA model

$$A(z^{-1})y(t) = z^{-k}B(z^{-1})u(t) + C(z^{-1})e(t)/\Delta \qquad (1)$$

is provided for disturbed process from which the integrating aspect of the controller is derived. $A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$ are polynomials in the backward-shift operator z^{-1}, k is a lower bound on the plant's dead time in sample intervals, u(t) and y(t) are the input and output of the plant, and e(t) is a sequence of independent random variables with common variance σ^2. Δ is the differencing operator $1-z^{-1}$.

With generalised minimum variance (GMV), integrating self-tuning control of offset eliminating methods based on a linear mathematics description of the process has been proved to be a very effective method. The linear model may be acceptable only in the case where the process is working around the operating point. As most industry process is highly nonlinear, non-minimum, uncertainty and load disturbance, nonlinear model should be used to describe the behavior of the process. Thus nonlinear control method should be introduced. Neural networks have been used in the adaptive control of nonlinear systems(Narendra, and Parthasarathy, 1990), involving both indirect (Bittanti, and Piroddi, 1994)and direct (Lightbody, and Irwin, 1995) method. Also fuzzy theory have been combined with adaptive control(Takagi, and Sugeno, 1985)which is aimed to solve the problem of uncertainty and thus introduce nonlinearity and human intelligence.

The combination of neural network and fuzzy control constitutes the neurofuzzy networks(Brown, and Harris, 1994), in which fuzzy rules could easily express the expert knowledge in linguistic form while neural networks posses the learning ability which could approximate nonlinear functions with arbitrary accuracy. Thus it greatly impulses the development of adaptive nonlinear control. Its ability of being trained on line using faster linear parameter estimation algorithms, and providing smooth transition between successive operating regimes make it suitable for the implementation of adaptive nonlinear control. A simplified recursive least squares method which is derived from the local change property of neurofuzzy networks is proposed (Chan*, et al.,* 2000).

The purpose of this paper is to propose an adaptive integrating controller and then to discuss a neurofuzzy network implementation. Simulation examples involving a linear and a nonlinear system respectively are presented to illustrate the performance of the self-tuning integrating neurofuzzy controllers.

2. INTEGRATING CONTROL FOR LINEAR AND NONLINEAR SYSTEMS

The motivation of the integrating control comes from the offset elimination capability of an integrator and thus to regulate the output of equation (1) to a constant set point. With a system pseudo-output $\phi(t+k) = Py(t+k) + Qu(t) - Rr(t)$, the cost function to be minimized is the variance of the pseudo-output.

Define E and G via the Diophantine equation:

$$PC = E\Delta A + z^{-k}G \qquad (2)$$

where the order of E and G are k-1 and $\max(n_a, n_p + n_c - k)$. n_a n_p and n_c are orders of A, P and C.

Multiply through the system equation (1) by E, substitute for EA from the Diophantine equation and adding $CQu(t) - CRr(t)$ to both side gives:

$$\begin{aligned} &C[Py(t+k) + Qu(t) - Rr(t)] \\ &= CQu(t) + BE\Delta u(t) + Gy(t) - CRr(t) + CEe(t+k) \end{aligned} \qquad (3)$$

which can be cast in the form:

$$\begin{aligned} \phi(t+k) = &\frac{1}{C}[Gy(t) + CQu(t) - CRr(t) + BE\Delta u(t)] \\ &+ Ee(t+k) \end{aligned} \qquad (4)$$

The error term Ee(t+k) is uncorrelated with the remainder of the right-hand side. The cost function is therefore minimized by setting the first term on the right-hand side to zero. Let $Q = Q'\Delta$, the integrating control law is :

$$(CQ' + BE)\Delta u(t) + Gy(t) - CRr(t) = 0 \qquad (5)$$

or

$$F\Delta u(t) + Gy(t) + Hr(t) = 0 \qquad (6)$$

with $\quad F = CQ' + BE \quad H = -CR$

Substituting equation (5) into CARIMA model yields the closed loop equation:

$$y(t) = \frac{z^{-k}BR}{\Delta AQ' + PB} r(t) + \frac{BE + Q'C}{\Delta AQ' + PB} e(t) \quad (7)$$

Note that if $Q' = 0$, the open loop zeroes are cancelled. For zero steady state tracking error(on average), equation (7) yields the condition

$$\left. \frac{BR}{\Delta AQ' + PB} \right|_{z=1} = 1 \qquad (8)$$

This is mostly satisfied by choosing $R = P(1)$ and $Q'(1) = 0$.

For the self-tuning purpose, rearranging (6) gives,

$$\Delta u(t) = -\overline{F}(z^{-1})\Delta u(t) - \overline{G}(z^{-1})y(t) - \overline{H}(z^{-1})r(t) \qquad (9)$$

where

$$\begin{aligned} \overline{F}(z^{-1}) &= F(z^{-1})/f_0 - 1 = \sum_{i=1}^{n_{\Delta u}} \overline{f}_i z^{-1} \\ \overline{G}(z^{-1}) &= G(z^{-1})/f_0 = \sum_{i=0}^{n_y} \overline{g}_i z^{-1} \\ \overline{H}(z^{-1}) &= H(z^{-1})/f_0 = \sum_{i=0}^{n_r} \overline{h}_i z^{-1} \end{aligned} \qquad (10)$$

f_0 is the leading constant in $F(z^{-1})$ and $n_y, n_{\Delta u}$ are the orders of the system. From (9) $\phi(t)$ can be expressed as:

$$\begin{aligned} \phi(t) = &[\Delta u(t-k) + \overline{F}\Delta u(t-k) + \overline{G}y(t-k) \\ &+ \overline{H}r(t-k)] + Ee(t+k) \end{aligned} \qquad (11)$$

In linear self-tuning integrating control, the parameters $\hat{\theta}^T(t) = [\hat{f}_0, \hat{f}_1, \cdots, \hat{g}_0, \hat{g}_1, \cdots, \hat{h}_1 \cdots]$ are updated from (11) using the RLS method by defining a data vector $x^T(t) = [\Delta u(t-k), \cdots, y(t-k), \cdots, r(t-k), \cdots]$. The integrating control law is computed on-line by (9) using the updated parameters.

The above linear self-tuning integrating control could be used for the following general nonlinear dynamical system :

$$\begin{aligned} y(t) = f[&y(t-1), \cdots, y(t-n'_y), \Delta u(t-k), \cdots, \\ &\Delta u(t-k-n'_{\Delta u}+1), e(t-1), \cdots, e(t-n'_e)] + e(t) \end{aligned} \qquad (12)$$

when a local linearized model similar to equation (1) is presented. $n'_y, n'_{\Delta u}, n'_e$ and k are respectively the orders and the time delay of the system, which are assumed known. Assuming that f[.] is a smooth nonlinear function such that a Taylor series expansion exists. A local linear model of (12) at an operating point O(t) can be given by the CARIMA model, though it is valid only in a small neighborhood about the operating point O(t). Thus neurofuzzy network is aimed at presenting satisfactorily operation over the full range of the operating points.

3. NEUROFUZZY NETWORK BASED ADAPTIVE INTEGRATING CONTROL

3.1 Neurofuzzy network based adaptive integrating control

The constitution of neurofuzzy network could be in fact a fuzzy controller realized by neural network(Fig.1). The network is initially designed to specify the shape(order) of each of the univariate basis functions, and this implicitly determines the number of basis functions mapped to for a particular network input. Thus the weights of the network were trained using linear parameter estimation methods from experimental data. Conceptually, the output of the network shown in Fig. 1 could be considered as a weighted sum of the outputs of several linear self-tuning incremental controllers designed at several specific operating points, which reflect the idea of nonlinear self-tuning integrating control. As the fuzzy sets in the neurofuzzy networks are distributed over the neighbourhood regions, control result obtained from a neurofuzzy network is generally smooth.

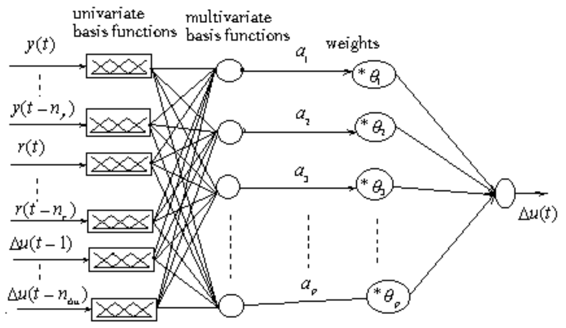

Fig. 1: Illustration of neurofuzzy network controller

It is obviously that the univariate basis function can be used to represent the fuzzy membership functions implementing the fuzzy linguistic terms. Therefore the neurofuzzy network shown in Figure 1 could be expressed as a set of fuzzy production rules :

R1: if $y(t)$ is positive small and $\cdots, y(t-n_y)$ is negative large
and $r(t)$ is positive medium and $\cdots, r(t-n_r)$ is positive medium
and $\Delta u(t-1)$ is negative medium and $\cdots \Delta u(t-n_{\Delta u})$ is negative large
then $\Delta u(t)$ is positive zero

....

Rp: if $y(t)$ is negative large and $\cdots, y(t-n_y)$ is negative medium
and $r(t)$ is positive medium and $\cdots, r(t-n_r)$ is positive medium
and $\Delta u(t-1)$ is negative zero and $\cdots \Delta u(t-n_{\Delta u})$ is negative large
then $\Delta u(t)$ is positive medium

The incremental control output is the synthesizing of these rules:

$$\Delta u(t) = a^T(x(t))\theta \tag{13}$$

$\theta = [\theta_1 \theta_2 \cdots \theta_p]$ is the weight vector of the network, x(t) is the input vector given by

$$\begin{aligned} x(t) = [&y(t), \cdots, y(t-n_y), r(t), \cdots, r(t-n_r), \\ &\Delta u(t-1), \cdots, \Delta u(t-n_{\Delta u})] \end{aligned} \tag{14}$$

a(x(t)) is the multivariate basis function obtained by the tensor products of the outputs of the univariate basis functions:

$$a_i(x(t)) = \prod_{k=1}^{n} \mu_{A_k^i}(x_k(t)) \tag{15}$$

where $n = n_y + n_r + n_{\Delta u} + 2$ is the dimension of the input vector. Thus the desirable properties of the univariate B-spline basis functions are all extended in a natural way to the multivariate basis functions. They are defined on hyperrectangles of size $(k_1 \times k_2 \times \cdots \times k_n)$ and therefore possess a bounded support. The output is positive inside this domain and zero outside. Fig.2 shows the multivariate basis function formed from two ,order 2, univariate basis function.

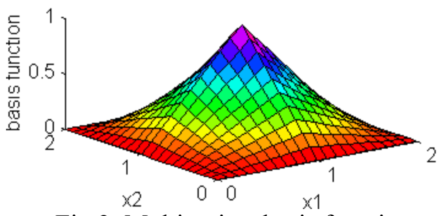

Fig.2 Multivariate basis function

The total number of weights in the network is:

$$p = (R_y + \rho_y)^{n_y+1}(R_r + \rho_r)^{n_r+1}(R_{\Delta u} + \rho_{\Delta u})^{n_{\Delta u}} \tag{16}$$

where $\rho_y, \rho_r, \rho_{\Delta u}$ are the respective order of the basis functions for y(t), r(t) and $\Delta u(t)$. R_y, R_r and $R_{\Delta u}$, the respective number of inner knots. Let the neurofuzzy network be denoted by NF(x(t)), the generalized system output given by (11) can now be rewritten as:

$$\phi(t) = [\Delta u(t-k) - NF(x(t-k))] + Ee(t) \tag{17}$$

Since the mean of e(t) is zero, the training target of the output of the neurofuzzy network is:

$$\psi(t) = \Delta u(t-k) - \phi(t) \qquad (18)$$

On-line training of the neurofuzzy by:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{P(t-1)a(x(t-k))[\psi(t) - a^T(x(t-k))\hat{\theta}(t-1)]}{1 + a^T(x(t-k))P(t-1)a(x(t-k))}$$

$$P(t) = P(t-1) - \frac{P(t-1)a(x(t-k))a^T(x(t-k))P(t-1)}{1 + a^T(x(t-k))P(t-1)a(x(t-k))} \qquad (19)$$

where $\hat{\theta}(t)$ is the estimate of θ at time t, and P(t), the estimate of the covariance matrix.

3.2 Relationships to previous controllers

For linear self-tuning integrating control method, using the decomposition of $G = 1 + \Delta G'$ in equation (6) and take $\delta F = 0$ $\delta G' = 0$ gives the linear PI controller(Tuffs,1985):

$$u(t) = \frac{g_0'}{f_0}[-y(t) + (CRr(t) - y(t)/g_0'\Delta] \qquad (20)$$

A simple extension to PID-like structure is obtained by choosing $\delta F = 1$ $\delta G' = 0$. While the controller is replaced by neurofuzzy network, it could really be nonlinear PI or PID controller. The derivation and results are given by Ying (1993), Liu (1997) and Liu (1999).

4. SIMULATION EXAMPLES

Example 1-Linear system:

Consider the following linear CARIMA model:

$$y(t) = 1.5y(t-1) - 0.7y(t-2) + u(t-1) + 0.5u(t-2) + [e(t) - 0.5e(t-1)]/\Delta + 0.4 \qquad (21)$$

where e(t)~N(0,1).

Let the generalized system output to be $\phi(t) = y(t) - r(t-1)$, which means that P=1, R=1 and Q=0. Solving the Diophantine equation acquires:

$$G = 2 - 2.2z^{-1} + 0.7z^{-2} \qquad H = -1 + 0.5z^{-1}$$
$$BE = 1 + 0.5z^{-1} \qquad CQ' = 0$$

The control law could be:

$$\Delta u(t) = -[2y(t) - 2.2y(t-1) + 0.7y(t-2) + 0.5\Delta u(t-1) - r(t) + 0.5r(t-1)] \qquad (22)$$

For self-tuning control, the control law could be expressed as:

$$\bar{f}_0\Delta u(t) - \bar{f}_1\Delta u(t-1) - \bar{g}_0 y(t) - \bar{g}_1 y(t-1) - \bar{g}_2 y(t-2) - \bar{h}_1 r(t-1) - r(t) = 0 \qquad (23)$$

Using RLS estimation:

$$\phi(t) + r(t-1) = \bar{f}_0\Delta u(t-1) - \bar{f}_1\Delta u(t-2) - \bar{g}_0 y(t-1) - \bar{g}_1 y(t-2) - \bar{g}_2 y(t-3) - \bar{h}_1 r(t-2) \qquad (24)$$

Fig.3 shows the system response of the integrating control under triangular wave setpoint and the parameter estimating process.

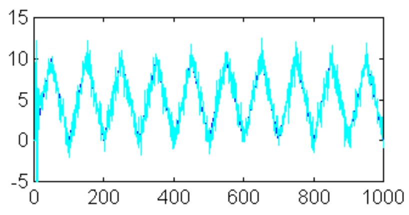

Fig3-a: Self tuning integrating control of linear model

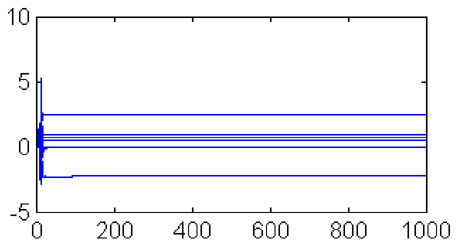

Fig3-b: Estimating of $\hat{f}_0, \hat{f}_1, \hat{g}_0, \hat{g}_1, \hat{g}_2, \hat{h}_1$

The self-tuning neurofuzzy controller is implemented with the same inputs. Two triangular basis functions are required for each input, giving $\rho_y = \rho_r = \rho_u = 2$, and $R_y = R_r = R_{\Delta u} = 0$. The number of weights of the neurofuzzy network p is: 2^6=64. The range of y(t) and r(t) is chosen to be between –5 and 10, whilst that for $\Delta u(t)$ is between –10 and 10. The weights $\hat{\theta}(0)$ are initially set to 0.1, and the covariance matrix P(0) to 100I, where I is an identity matrix. Fig.4 shows the system response and the estimating process.

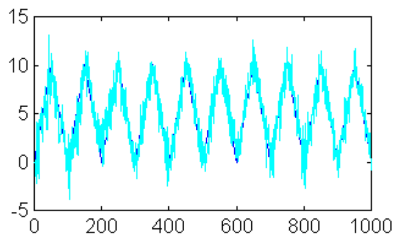

Fig4-a. Response with the self-tuning neurofuzzy controller

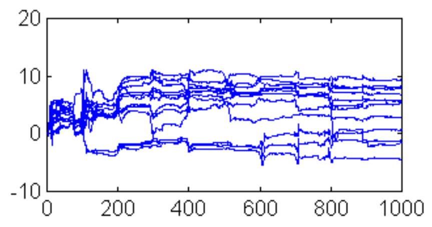

Fig4-b: Estimating of $,\hat{\theta}_1 \rightarrow \hat{\theta}_{16}$ (totally 64 coefficients)

Fig5 shows comparison of the accumulated cost function between self-tuning integrating controller and self-tuning integrating neurofuzzy controller from time t=230s to t=1000s. As can be seen, there is no obvious difference of the system response between these two methods.

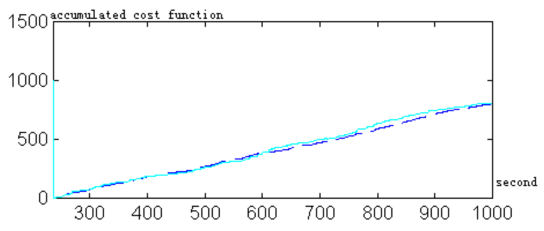

Fig.5 Comparison of the accumulated cost function between self-tuning integrating controller(solid line) and self-tuning integrating neurofuzzy controller (dashed line)

Example 2 Nonlinear model:

Consider the following nonlinear CARIMA model(Bittanti and Piroddi, 1994):

$$y(t) = 0.3y(t-1) + 0.6y(t-2) + [u(t-2)]^{1/3} + e(t)/\Delta \tag{25}$$

where e(t)~N(0,0.1).

Since $\phi(t) = y(t) - r(t-2)$, the linearized self-tuning integrating control law could be:

$$\begin{aligned}&\bar{f}_0\Delta u(t) - \bar{f}_1\Delta u(t-1) - \bar{g}_0 y(t) - \bar{g}_1 y(t-1)\\ &-\bar{g}_2 y(t-2) - r(t) = 0\end{aligned} \tag{26}$$

Fig 6 shows the system response under square wave by the self-tuning integrating controller. There exist obvious oscillations in the output. Nevertheless, the coefficients of the control law could reach convergence, which proves the correctness of the algorithm. It shows weak robustness possessing weak ability of overcoming noise disturbances.

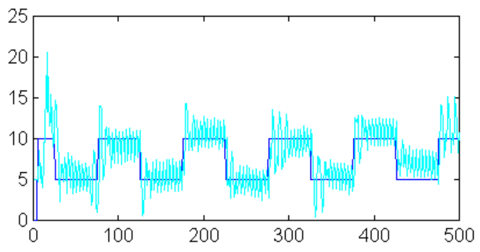

Fig.6-a : Response with linearized self-tuning integrating controller

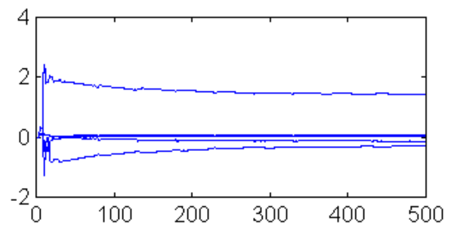

Fig.6-b: Estimating process of $\hat{f}_0, \hat{f}_1, \hat{g}_0, \hat{g}_1, \hat{g}_2$

The self-tuning neurofuzzy controller is implemented with the same inputs. Two triangular basis functions is used for each input. y(t) and r(t) are selected between 3 and 12, $\Delta u(t)$ is between – 112 and 117. The number of weights of the neurofuzzy network p is: 2^5=32. Fig.7 shows the good tracking of the set-point of the self-tuning neurofuzzy integrating controller and it weights estimating process. Fig.8 shows the comparison of the accumulated cost function between the two method. As can be seen, great improvement has been made by using neurofuzzy integrating controller. It shows better tracking ability and the accumulated cost function is much reduced. For nonlinear system, nonlinear controller could control nonlinear model better.

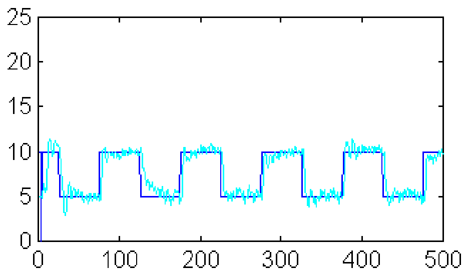

Fig.7-a Response with the self-tuning neurofuzzy controller

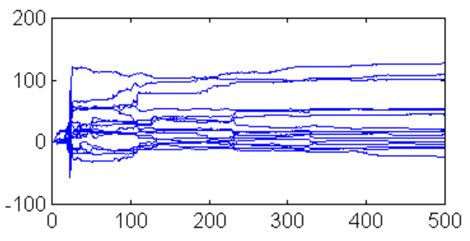

Fig.7-b: Estimating process of $,\hat{\theta}_1 \rightarrow \hat{\theta}_{16}$

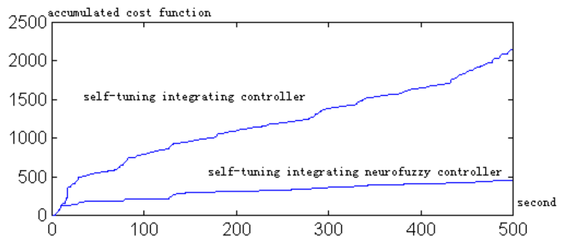

Fig.8 Comparison of the accumulated cost function

5. CONCLUSION

A self-tuning neurofuzzy integrating controller is derived following the approach in deriving self-tuning integrating controllers for linear systems while CARIMA model is introduced. Neurofuzzy networks are chosen to act as nonlinear self-tuning controller. From the simulation examples, the performance of the self-tuning neurofuzzy integrating controller for the linear system is similar to that of the self-tuning integrating controller, but is superior to the self-tuing integrating controller for the nonlinear system. Comparing with 'position' adaptive control, integrating adaptive control could:

(1) Eliminate offsets effectively because integral action is incorporated in their control laws. Besides, the parameter estimation is more robust because the estimated controller parameters need not to be re-tuned when sudden changes in offset are encountered.

(2) Reduce the computing burden for neurofuzzy network. There is one fewer input into the neurofuzzy network than that of 'position' control(if there is u(t),u(t-1),u(t-2)in the position control law, then there is only $\Delta u(t), \Delta u(t-1)$ in the integrating control law). For the univariate basis function chosen as example 1, there will be 2^7=128 weights to be estimated in the position control rather than 2^6=64 weights in the integrating control.

REFERENCE

Brown, M. and Harris, C. J. (1994). Neurofuzzy adaptive modelling and control. Englewood Cliffs, NJ:Prentice Hall.

Bittanti, S.and Piroddi, L.(1994). GMV technique for nonlinear control with neural networks. *IEE Proc.-Control Theory Appl.*, **141**(2), 57-69.

Chan, C. W., CHEUNG, K. C., and YEUNG, W. K.(2000). A computation-efficient on-line training algorithm for neurofuzzy networks. *Int. J. Sys. Sciences*, **31**(3), 297-306.

Lightbody, G.and Irwin,G.W. (1995). Direct neural model reference adaptive control. *IEE Proc.-Control Theory Appl.*, **142**(1), 31-43.

Narendra, K. S. and Parthasarathy, K. (1990). Identification and control of dynamical systems using neural networks. *IEEE Trans.Neural Networks*, **1**, 4-27.

Takagi, T. and Sugeno, M. (1985). Fuzzy identification of systems and its application to modelling and control. *IEEE Trans*.SMC, **15**(1), 116-132.

P.S. Tuffs,et.al.,(1985). Self-tuning control of offset: a unified approach. *Proc. IEE,* **132D**, 100-110.

Xiang-Jie Liu, Huan-Shui Zhang and Tian-You Chai. (1997). Structure analysis of three-dimensional fuzzy controller and its relationship to PID controller. Proc. the 36th IEEE Conference on Decision and Control (pp.3350-3351). San Diego, California USA.

Xiang-Jie Liu, Xiaoxin Zhou(1999). Structure Analysis of Fuzzy Controller with Gaussian Membership Function. Proc.the 14th IFAC World Congress. Volume **K** (pp.201-206).Beijing China.

Ying H.(1993). A nonlinear fuzzy controller with linear control rules is the sum of a global two-dimensional multilevel relay and a local nonlinear proportional-integral controller. *Automatica,* **29**, 499-503.