Available online at www.sciencedirect.com

ScienceDirect

Neurocomputing 70 (2006) 559–567

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Mold temperature control of a rubber injection-molding machine by TSK-type recurrent neural fuzzy network

Chia-Feng Juang[a,*], Shui-Tien Huang[a], Fun-Bin Duh[b]

[a]Department of Electrical Engineering, National Chung Hsing University, Taichung, 402 Taiwan, ROC
[b]Department of Electrical Engineering, Feng Chia University, Taichung, 407 Taiwan, ROC

Received 12 August 2005; received in revised form 18 October 2005; accepted 4 November 2005
Available online 17 February 2006
Communicated by J. Zhang

Abstract

Practical mold temperature control of a rubber injection-molding machine is studied in this paper. The controller used is a recurrent fuzzy network called Takagi–Sugeno–Kang (TSK)-type recurrent fuzzy network (TRFN), which is characterized by its recurrent structure, on-line structure and parameter learning. Due to the powerful learning ability of TRFN, a simple controller design scheme using direct inverse configuration is proposed. With recurrent structure in TRFN, no a priori knowledge of the molding machine order is required. The designed TRFN controller performs well even if the sampling interval is different from the original one used for training. The design of TRFN consists of off-line and on-line training. For off-line learning, structure and parameter of TRFN are learned, and the consequent part parameters are tuned by Kalman filter algorithm. On-line learning is performed to fine tune the consequent parameters of TRFN and achieve a better control performance with the use a simple gradient descent algorithm. Practical experiments and comparisons with other types of controllers demonstrate the performance of the proposed TRFN controller.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Recurrent fuzzy networks; Neural fuzzy networks; Fuzzy control; Direct inverse control; Temperature control

1. Introduction

For a rubber injection-molding machine to produce high quality products, precise temperature control of the mold during the production process is an important factor. The temperature of a mold depends not only on distribution of the heater but also on the controller, and in this paper, we focus on the latter. Recently, the idea of soft computing such as fuzzy logic (FL) and neural networks (NN) has attracted considerable attention for the realization of better and more efficient control of nonlinear plants. For the FL controller (FLC) design, conventionally, the selection of fuzzy if-then rules often relies on a substantial amount of heuristic observations to express proper knowledge of strategies, which usually requires a lot of design effort. For the NN, its nonlinear mapping and self-learning abilities have been the motivating factors for its use in developing adaptive control systems. However, slow convergence is the major disadvantage of the NN. Moreover, when it is trained on-line in order to adapt well to the environment variations, its global tuning property usually leads to over-tuning, which will degrade the performance of the controller. To overcome these disadvantages of FC and NN, and to keep the advantages of both, the technique of neural fuzzy network has been proposed [14]. Two categories of neural fuzzy network structures, feedforward [2,5,7,8,23] and recurrent [3,4,6,9,13,17,18,21,24,26], have been proposed to solve the problems with static and dynamic characteristics, respectively. For mold temperature control of a rubber injection-molding machine, we usually face the time-delay problem, so control by recurrent neural fuzzy network would be a better choice, as proposed in this paper.

Recurrent fuzzy networks are fuzzy networks with feedback connections in their structure. Many types of

*Corresponding author.
E-mail address: cfjuang@dragon.nchu.edu.tw (C.-F. Juang).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2005.11.003

recurrent fuzzy networks have been proposed [3,4,6, 9,13,17,18,21,24,26]. In [26], a recurrent neural-fuzzy network is proposed, where the consequence of a rule is a reduced linear model in AutoRegressive with eXogenous inputs (ARX) form. In [17], a dynamic fuzzy neural network is proposed, where the consequent parts of the fuzzy rules are recurrent neural networks with internal feedback and time delay synapses. In [13], a wavelet-based recurrent fuzzy neural network is proposed for dynamic system identification. Previously, we proposed two types of recurrent neural fuzzy networks, the recurrent self-organizing neural fuzzy inference network (RSONFIN) [9] and TSK-type recurrent fuzzy network (TRFN) [6]. In TRFN, the consequent part is of TSK-type, and its performance is shown to be better than RSONFIN, in which a fuzzy set is used in consequence. Compared with other recurrent fuzzy networks which also use TSK-type consequence [17,18,26], one major advantage of TRFN is that no a priori knowledge of the plant order is required, which eases the design process. In this paper, we use the TRFN structure as a temperature controller, and modify the parameter learning process of original TRFN to improve the learning performance.

Many control configurations for time-delayed plant control have been proposed, and one generally adopted approach is the generalized predictive control (GPC) [1]. Recently, fuzzy controller design based on GPC has been proposed [11,12,16] for nonlinear plant control. In the fuzzy controller, the consequence is of linear GPC form and is designed by linear GPC learning algorithm. The drawback of this model is that we should know in advance the order of input and output terms of the linear GPC model in the fuzzy consequence. Other controller design configurations based on neural learning approaches include the direct inverse, direct and indirect adaptive controls [19]. Among these, the direct inverse control configuration requires no emulation of the plant and is simpler for implementation [10,15,22]. The direct inverse control with feedforward multilayer neural networks has been proposed [10,22]. The direct inverse control by feedforward neural fuzzy network has also been proposed, where the learning and control performance is demonstrated to be better than the feedforward-neural network [15]. Owing to the feedforward network structure, proper network inputs are selected by trial and error [10,15,22]. For TRFN, due to its recurrent structure, the input selection process can be avoided. In this paper, we perform mold temperature control by TRFN-based direct inverse control configuration. Other generally adopted controller, including proportional-integrator (PI) and fuzzy controllers, are also compared.

The paper is organized as follows. Section 2 describes the molding machine system. In Section 3, structure of TRFN and TRFN-based off-line and on-line direct inverse control configurations are introduced. Section 4 depicts the experimental studies. Conclusions are presented in Section 5.

2. Description of the molding machine temperature control system

The entity of the rubber injection molding system presented in this paper is shown in Fig. 1. The mold is heated using both upper and lower heating plates simultaneously to achieve a desired temperature. The upper and lower plates are composed of six 220 V, 1200 W heaters connected in a triangular shape, as shown in Fig. 2. The mold itself weighs 450 Kg with a size of 60 cm × 60 cm × 40 cm. The heating plates are driven by solid-state relay (SSR) drivers, and temperatures are measured by LM335 sensors. The system is controlled by a PC with Intel Celeron 467 MHz CPU. The interface card

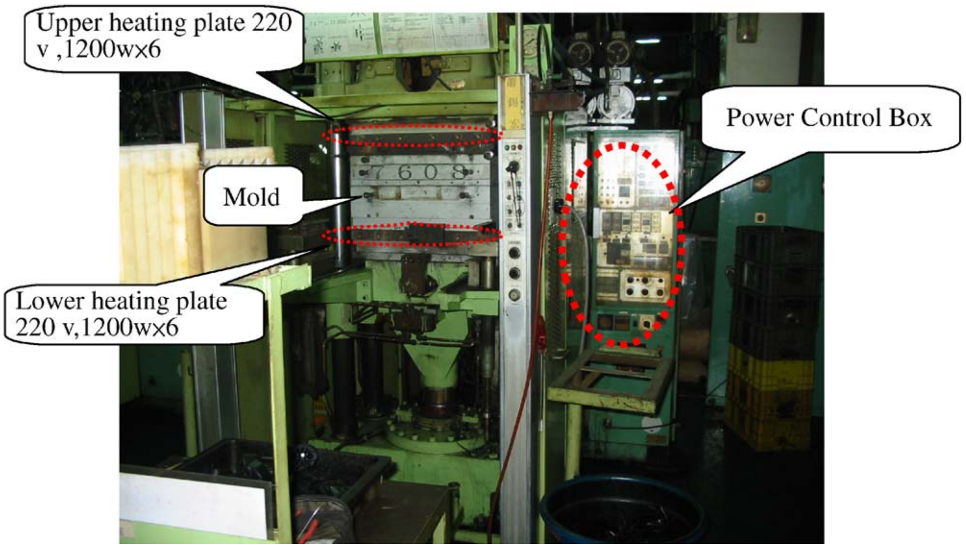

Upper heating plate 220 v ,1200w×6
Power Control Box
Mold
Lower heating plate 220 v,1200w×6

Fig. 1. Entity of the rubber injection molding machine.

is Flytech PB-PC010/C card with 12 bits A/D and D/A. The block diagram of the control system is shown in Fig. 3. The output of the TRFN controller is denoted by u, and the plates are heated by the following rules:

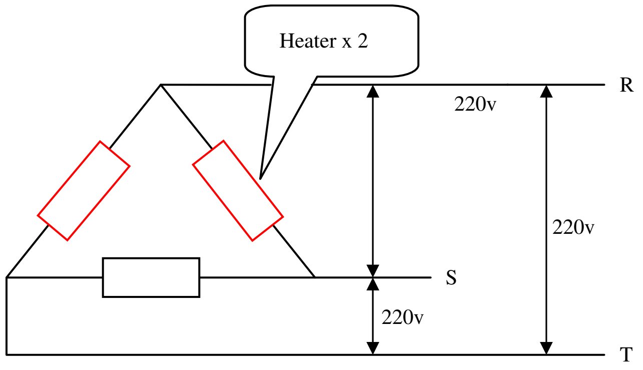


Fig. 2. Basic circuit showing the distribution of heaters.

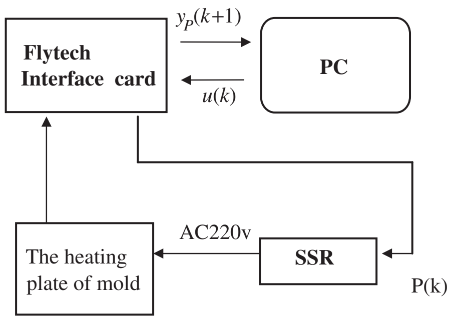


Fig. 3. Block diagram of the PC-based mold temperature control system.

IF $u(k) \leqslant 0$, THEN $P(k) = 0\%$ Heating plate voltage off
IF $u(k) \geqslant 5$, THEN $P(k) = 100\%$ Heating plate voltage on
IF $0 < u(k) < 5$, THEN $P(k) = u(k)/5$ Heating plate voltage on

where k denotes the sample number and $P(k)$ is the output of the interface card sent in the form of PWM. For a discrete-time control system, we must decide the sampling interval in advance. For the control system, the sampling period T_S is 45 s.

3. TRFN-based direct inverse control

The TRFN is a general connectionist model of a dynamic fuzzy inference system, which can find its best structure and parameters automatically. In this section, the structure of TRFN is introduced followed by the TRFN-based control configuration.

3.1. Structure of TRFN

The structure of TRFN is shown in Fig. 4. A network with two external inputs and a single output is considered here for convenience. This six-layered network realizes a recurrent fuzzy network of the following form:

Rule 1: IF $x_1(t)$ is A_{11} and $x_2(t)$ is A_{12} and $h_1(t)$ is G THEN $y(t+1)$ is $a_{10}+a_{11}x_1(t)+a_{12}x_2(t)+a_{13}h_1(t)$ and $h_1(t+1)$ is w_{11} and $h_2(t+1)$ is w_{21}
Rule 2: IF $x_1(t)$ is A_{21} and $x_2(t)$ is A_{22} and $h_2(t)$ is G

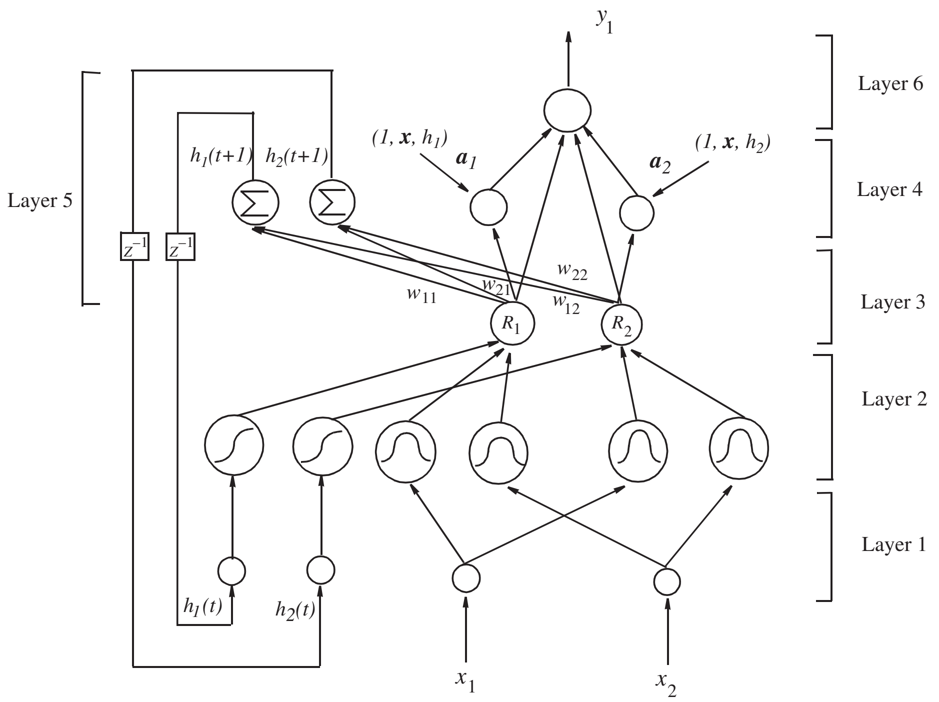


Fig. 4. Structure of the TRFN.

THEN $y(t+1)$ is $a_{20}+a_{21}x_1(t)+a_{22}x_2(\mathrm{t})+a_{23}h_2(\mathrm{t})$ and $h_1(t+1)$ is $w_{12}$and $h_2(\mathrm{t}+1)$ is w_{22}

where A and G are fuzzy sets, w and a are the consequent parameters for inference output h and y, respectively. The consequent part for the external output y is of TSK-type and is a linear combination of the external input variables x and internal variables h, plus a constant.

To give a clear understanding of the mathematical function of each node, we will describe functions of TRFN layer by layer. For notation convenience, the net input to the ith node in layer k is denoted by $u_i^{(k)}$ and the output value by $O_i^{(k)}$.

Layer 1: No function is performed in this layer. The node only transmits input values to layer 2.

Layer 2: Nodes in layer 2 act as membership functions. Two types of membership functions are used in this layer. For external input $u_j^{(2)}=x_j$, the following Gaussian membership function is used,

$$O_i^{(2)}=\exp\left\{-\frac{\left(u_j^{(2)}-m_{ij}\right)^2}{\sigma_{ij}^2}\right\} \text{ and } u_j^{(2)}=O_j^{(1)}, \tag{1}$$

where m_{ij} and σ_{ij} are, respectively, the center and the width of the Gaussian membership function of the ith term of the jth input variable x_j. For internal variable $u_i^{(2)}=h_i$, the following sigmoid membership function is used,

$$O_i^{(2)}=\frac{1}{1+\exp\left\{-u_i^{(2)}\right\}} \text{ and } u_i^{(2)}=O_i^{(5)}. \tag{2}$$

Layer 3: Each node in this layer calculates the firing strength of a rule by product operation. The function of each rule is

$$O_i^{(3)}=\prod_{j=1}^{n+1}O_j^{(2)} = \frac{1}{1+\exp\left\{-O_i^{(5)}\right\}}\exp\left\{-\left(\sum_{j=1}^{n}\frac{\left(O_j^{(1)}-m_{ij}\right)^2}{\sigma_{ij}^2}\right)\right\}, \tag{3}$$

where n is the number of external inputs.

Layer 4: Nodes in this layer perform a linear summation of the TSK-type consequence. The mathematical function of each node i is

$$O_i^{(4)}=\sum_{j=0}^{n+1}a_{ij}u_j^{(4)}=a_{io}+\sum_{j=1}^{n}a_{ij}x_j+a_{in+1}h_i, \tag{4}$$

where n is the number of external input variables, and a_{ij}, $j=0,\ldots,n+1$, are the parameters to be tuned.

Layer 5: The context node functions as a defuzzifier for the fuzzy rules with inference output h. The link weights represent the singleton values in the consequent part of the internal rules. The simple weighted sum is calculated in each node,

$$h_i=O_i^{(5)}=\sum_{j=1}^{r}O_j^{(3)}w_{ij}. \tag{5}$$

As in Fig. 4, the delayed value of h_i is fed back to layer 1 and acts as an input variable to the precondition part of a rule. Each rule has a corresponding internal variable h and is used to decide the influence degree of temporal history to the current rule.

Layer 6: The node in this layer computes the output signal y of the TRNFN. The output node together with links connected to it act as a defuzzifier. The mathematical function is

$$y=O^{(6)}=\frac{\sum_{j=1}^{r}O_j^{(3)}O_j^{(4)}}{\sum_{j=1}^{r}O_j^{(3)}}. \tag{6}$$

3.2. TRFN-based direct inverse control

The TRFN-based direct inverse control configuration is shown in Fig. 5. During control, the reference temperature, $y_{\mathrm{ref}}(k+1)$, current temperature, $y\mathrm{p}(k)$, and last control input, $u(k)$, are fed as TRFN inputs. In this configuration, our objective is to train the TRFN to emulate the inverse of the controlled plant input–output mapping. The selection of network inputs is an important task before network training. For the molding machine, the mold temperature depends not only on current temperature and control input but also on their past values. In addition, the dependence is related to the sampling interval. For TRFN, due to its recurrent property, only $y_{\mathrm{P}}(k)$ and $u(k)$ are used even when the sampling interval varies. Training consists of off-line and on-line learning phases, whose details are introduced in the following.

3.2.1. Off-line training

To obtain the training samples for off-line training, a sequence of random input signals $u(k)$, $0\leqslant u(k)\leqslant 5\mathrm{V}$, are injected directly into the plant, and then an open-loop input–output characteristic of the plant is obtained. The random inputs should be injected successively so that the temporal relations between input voltages and output

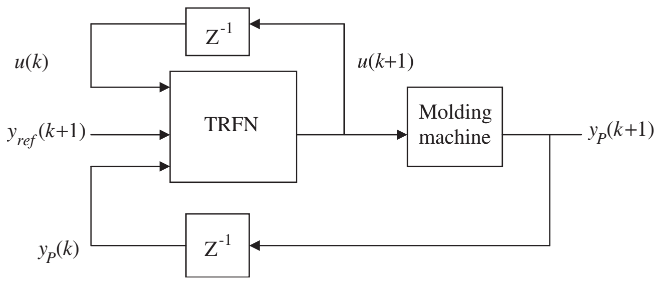


Fig. 5. TRFN-based direct inverse control.

temperature are learned. For the molding machine, the temperature increases as random input voltages are injected successively. The number of training data is selected to cover the entire reference temperature space of interest and 70 training data is selected in the experiment. Based on Fig. 5, to learn the inverse of the controlled plant, the values of $u(k)$, $y_P(k)$ and $y_P(k+1)$ are fed as TRFN inputs and the corresponding control signal $u(k+1)$as the desired output. Let the output of TRFN be denoted by $\hat{u}(k+1)$ during training. The TRFN is supervised to minimize the error cost function

$$\text{Error} = \sum_{k=0}^{69}[u(k+1) - \hat{u}(k+1)]^2.$$

To train TRFN, we can divide it into two subtasks: structure learning and parameter learning, which are both performed concurrently. That is, for each input data, the structure learning is performed followed by parameter learning. There are no rules initially in TRFN, and the objective of the structure learning is to decide the number of fuzzy rules, initial location of membership functions, and initial consequent parameters. Before structure learning, two parameters, F_{in} and β should be assigned in advance. The parameter F_{in} is a threshold that influences the total number of rules generated. A larger value of F_{in} will generate a larger number of rules. The parameter β decides the overlap degree between two rules (clusters). That is, it decides the initial width of each Gaussian fuzzy set. With the same value of F_{in}, a higher value of β means a larger initial width of each fuzzy set, and so fewer rules are generated. Different values of parameters may generate different initial locations of fuzzy rules. Although different locations are generated initially, the succeeding network parameter learning helps to find the optimal locations. The detailed structure learning of TRFN has been previously presented [17]. On the contrary, the objective of parameter learning is to tune the free parameters of the constructed network to an optimal extent. Since the desired output is a sequence of random variables which change abruptly, it is difficult to learn the input–output mapping. For this problem, one possible solution is to get a smoother output. For example, instead of using a random signal $u(k)$ exciting the unknown system to be modeled, a sequence of random steps of limited size is devised. According to our experiment, a smaller squared error is obtained from this training data, but the control performance is bad. The reason for this is that for the molding temperature control system, abrupt change of the control voltage is required as is seen in the following section. However, such abrupt variation is not learned in the training data with smoother output. Another solution is to improve the performance of the modeling network, which is the way adopted here. To improve the learning performance of TRFN, we modify the gradient descent-learning algorithm [17], and a new type of learning algorithm, the Kalman filter algorithm, is employed for consequent part parameter learning. Given $A\vec{x} = \vec{d}$, the least square solution $\vec{x}^*$ can be found by pseudoinverse as $\vec{x}^* = (A^{\mathrm{T}}A)^{-1}A^{\mathrm{T}}\vec{d}$. To compute $\vec{x}^*$, the Kalman filter algorithm provides an on-line computation way that computes $\vec{x}^*$ recursively. Detailed derivation of the algorithm can be found in [14]. With the Kalman filter algorithm, the parameter vector $\vec{a}$ in layer 4 is updated by

$$\vec{a}(k+1) = \vec{a}(k) + Q(k)\vec{\mu}(k+1)(\hat{u}(k+1) - u(k+1)),$$

$$Q(k+1) = \frac{1}{\lambda}\left[Q(k) - \frac{Q(k)\vec{\mu}^{\mathrm{T}}(k+1)\vec{\mu}(k+1)Q(k)}{\lambda + \vec{\mu}^{\mathrm{T}}(k+1)Q(k)\vec{\mu}(k+1)}\right], \quad (10)$$

where $0<\lambda\leqslant 1$ is the forgetting factor, $\vec{\mu}^{\mathrm{T}} = \left[O_1^{(3)}/\sum_{j=1}^{r}O_j^{(3)} \cdots O_r^{(3)}/\sum_{j=1}^{r}O_j^{(3)}\right]^{\mathrm{T}}$, and Q is the covariance matrix. The initial value of Q is set as $Q(0) = 10\cdot I$. The other free parameters, m_{ij} and σ_{ij} in layer 2 and w in layer 5 are tuned by a real time recurrent learning algorithm [25].

3.2.2. On-line learning

The objectives of on-line learning are to improve the control performance of a TRFN controller after off-line training and to deal with unpredictable changes in the control environment, which usually happens in many industrial control processes. For on-line learning, the direct inverse-modeling learning scheme shown in Fig. 6 is used. Parameter learning is performed alone during on-line learning. This is in contrast to off-line learning, where a whole network is learned with abrupt change in the desired output. For on-line learning, the network parameters are locally modified with smooth changes in the desired output value. Thus a simpler learning algorithm, the gradient descent algorithm, is adopted. At time step k, the TRFN controller generates a control signal $u(k+1)$ according to inputs $u(k)$, $y_P(k)$ and $y_{\text{ref}}(k+1)$, and it obtains a new temperature $y_p(k+1)$. Then we can define a training pattern with input $[u(k), y_P(k), y_P(k+1)]$ and desired output $u(k+1)$. The squared difference between TRFN output $\hat{u}(k+1)$ and desired output $u(k+1)$, denoted as

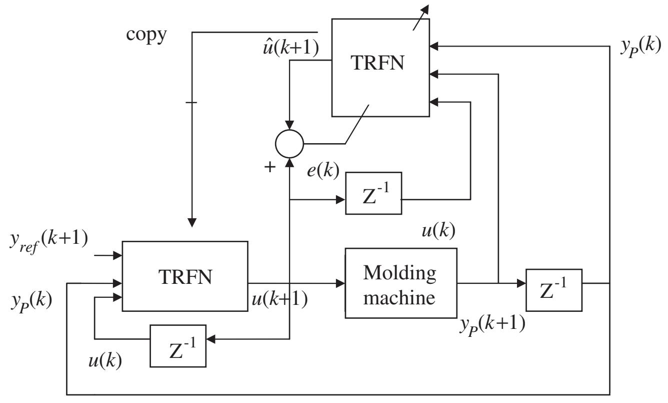


Fig. 6. On-line training of TRFN based on direct inverse control configuration.

$e(k) = (\hat{u}(k+1) - u(k+1))^2$, is minimized at each time step by the following gradient descent algorithm

$$a_{ij}(k+1) = a_{ij}(k) - \eta \frac{\partial e(k+1)}{\partial a_{ij}(k)}$$

$$\text{for } i = 1, \ldots, r, \; j = 0, \ldots, 4, \tag{11}$$

where

$$\frac{\partial e(k+1)}{\partial a_{ij}(k)} = (\hat{u}(k+1) - u(k+1)) \frac{O_i^{(3)} u_j^{(4)}}{\sum_{k=1}^{r(t)} O_k^{(3)}}. \tag{12}$$

For on-line learning, since the changes on the controller as well as the controlled plant are usually small, the tuning of the consequent part parameters is sufficient. In addition, tuning with only the consequent part parameters significantly reduces the on-line training time and is more applicable to real-time control requirements.

4. Experimental studies

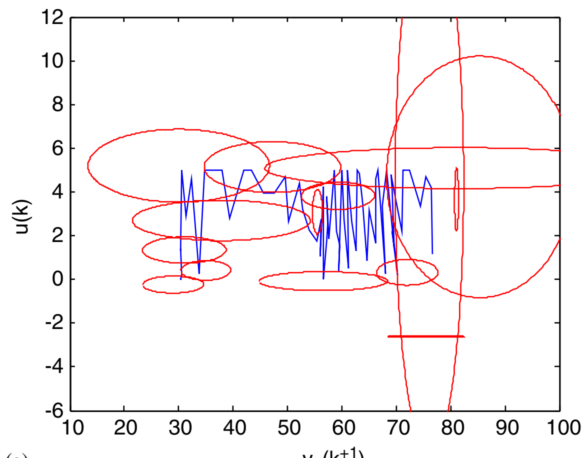
To train TRFN from the data collected off-line, the learning parameters for structure learning of TRFN are set as $F_{\text{in}} = 0.06$ and $\beta = 0.3$. For network parameter learning, the learning constant for Kalman filter algorithm is $\lambda = 0.999$, and the learning constant for real-time recurrent learning is 0.01. Fifteen rules are generated after 500 iterations of training. Distributions of the training data and generated fuzzy rules on two of the three input variables are shown in Fig. 7, where Figs. 7(a) and (b) show distributions on the inputs $[y_p(k+1), u(k)]$ and $[y_p(k), y_p(k+1)]$, respectively. In these figures, the contour of each cluster corresponds to firing strength of $\exp(-1)$, and all data are in fact covered by the clusters. The learned root-mean-squared-error (RMSE) is 0.255. The desired output and the actual output of TRFN are shown in Fig. 8. The control result by TRFN after off-line learning is shown in Fig. 9. Since only 70 pieces of data are used for training, which may not be representative for plant inverse modeling, the control results are not satisfactory. To further improve the performance, on-line learning is performed, and better control results are achieved, as shown in Fig. 9. To measure the control error quantitatively, we define the following sum of absolute error (SAE)

$$\text{SAE}(m,n) = \sum_{k=m}^{n} |y_{\text{ref}}(k+1) - y_{\text{P}}(k+1)|.$$

The SAE of the control result by TRFN is shown in Table 1.

To demonstrate the learning performance of TRFN, learning by RSONFIN [9] and TRFN-S [6] with consequent part parameters designed by gradient descent algorithm is performed. The generated numbers of rules in RSONFIN and TRFN-S are also set to be 15, respectively. The learned RMSEs of RSONFIN and TRFN-S are 1.39 and 1.15, respectively, which are larger than RMSE of the proposed TRFN, where the consequent parameters are trained by Kalman filter algorithm.

Next, the two generally adopted controllers, PI and fuzzy logic controllers, are compared. For the PI control, a


(a)

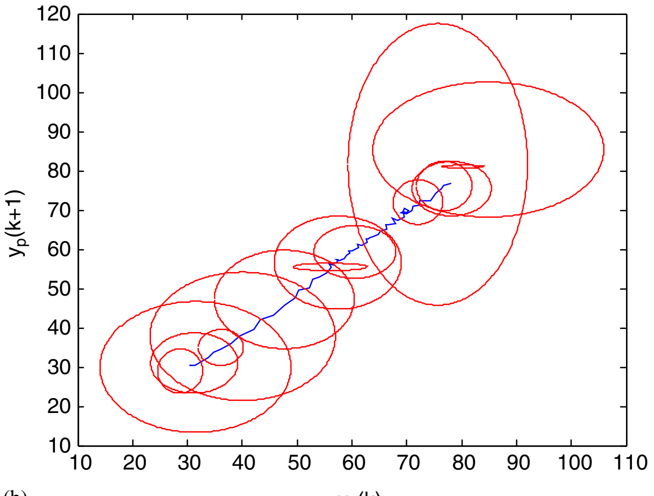

(b)

Fig. 7. Distributions of the training data and generated 15 fuzzy rules on input variables (a) $[y_p(k+1), u(k)]$; (b) $[y_p(k), y_p(k+1)]$.

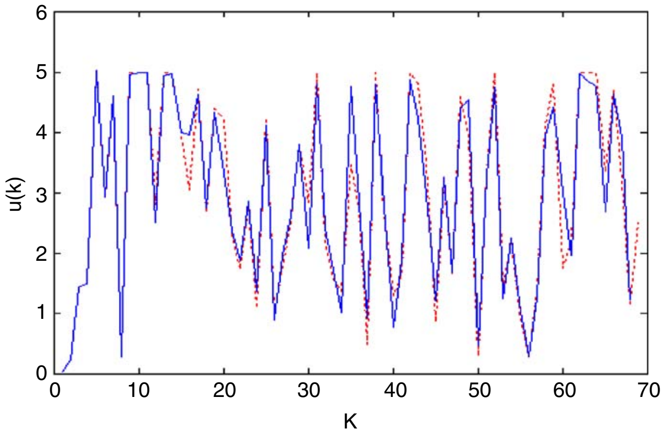


Fig. 8. Desired output (solid line) and actual output (dotted line) of TRFN.

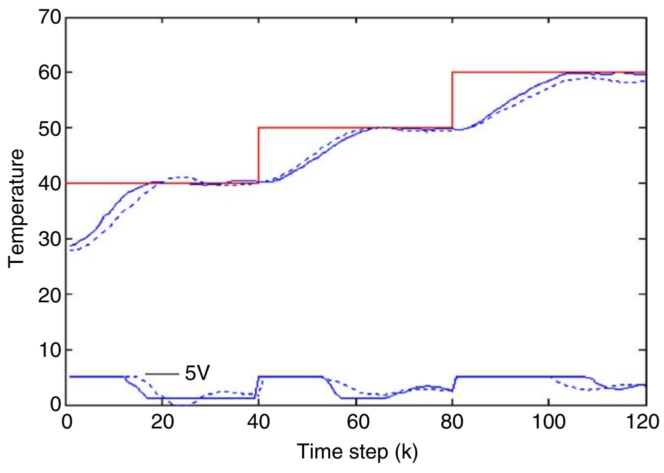


Fig. 9. Desired temperature and controlled temperature by TRFN after off-line leaning (dotted line) and on-line leaning (solid line).

Table 1
The sum of absolute error (SAE) between TRFN, PID, and FLC for different sampling intervals

	TRFN	PI	FLC
SAE(8,120) $T_S = 45\,s$	245.26	278.26	294.8
SAE(8,60) $T_S = 60\,s$	124.49	144.31	147.83
SAE(25,240) $T_S = 15\,s$	494.6	514.2	511.0

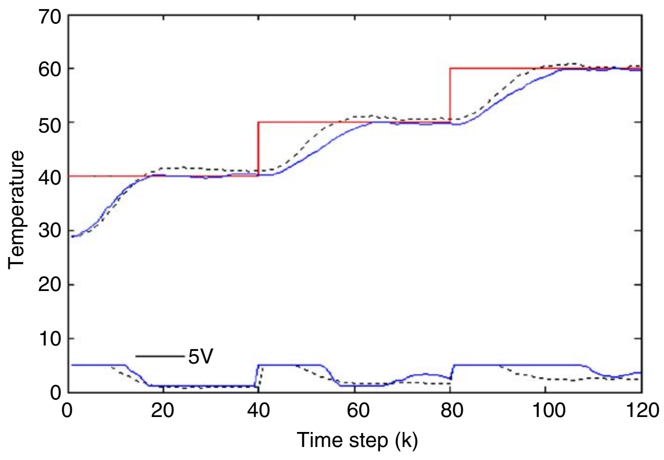


Fig. 10. Temperature control results obtained by TRFN (solid line) and PI controller (dotted line).

positional form discrete PI controller [20] is used, and the positional parameter K_P and integral parameter K_I that achieve the best performance after several trials are found to be 30 and 0.2, respectively. The control result is shown in Fig. 10, from which we find that the PI controller has larger overshoot, settling time, and steady state error than the TRFN controller. Especially, at reference temperature

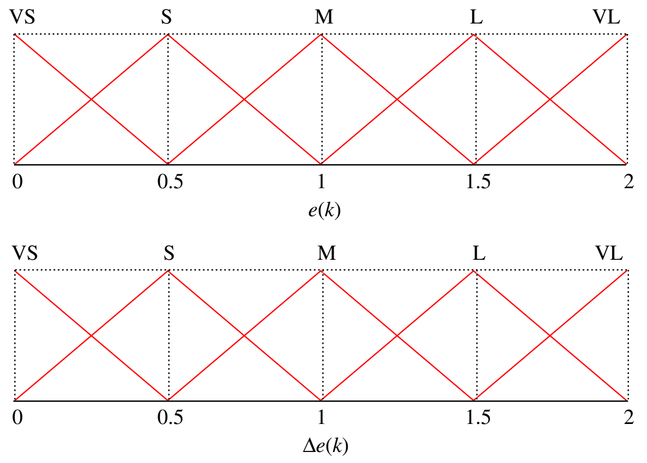


Fig. 11. Membership functions for fuzzy controller inputs $e(k)$ and $\Delta e(k)$.

Δe(k) / e(k)	VL	L	M	S	VS
VL	5	5	5	5	5
L	5	5	4	4	4
M	4	4	3	3	3
S	3	3	2	2	2
VS	2	1	1	1	0

Fig. 12. The consequent values for each fuzzy rule.

$40\,^{\circ}C$, the settling time of PI controller is about 15 min longer than that of TRFN controller, which may seriously influence the quality of the product. For quantity comparison, the SAE of PI controller is shown in Table 1, which shows that TRFN achieves a smaller SAE.

For the fuzzy logic control (FLC), we specify the input variables as the performance error $e(k)$, which is the error between the reference output and actual temperature of the water bath system, and the rate of change of the performance error $\Delta e(k)$. The output variable $u(k)$ is the voltage between 0 and 5 V. We partition the two input variables into five fuzzy sets, respectively, and there are 25 fuzzy rules in all. Triangular membership functions are chosen for the fuzzy sets of the two input fuzzy variables, as shown in Fig. 11. The output variable, $u(k)$, is quantified into six fuzzy singletons: 0.0, 1.0, 2.0, 3.0, 4.0 and 5.0 V. According to engineering judgment and tenths of trials, the consequent values of the 25 fuzzy rules that achieve the best performance are shown in Fig. 12. This manual design process is very time consuming as each trial takes 120 time steps of control, which corresponds to one and half an hour when sampling time is 45 s. The natural cooling of the molding machine requires more time. The whole design process takes several days of trials. The control results are shown in Fig. 13. Since the characteristic of the machine differs at different temperatures, it is difficult for the designed fuzzy controller to perform well at different reference temperatures. For this reason, additional trials considering the reaction of the designed FLC to disturbances and added delays are not experimented during the design process. Fig. 13 shows that the FLC has larger steady state error than the TRFN controller, and the settling time is hundreds of seconds longer. The control error of FLC is shown in Table 1, which shows that TRFN achieves smaller control error than FLC.

Finally, we test the recurrent property of TRFN in dealing with time delays. The sampling interval is increased to $T_S = 60$ s and decreased to $T_S = 15$ s, respectively. Using the controller parameters designed at $T_S = 45$ s, the SAEs of TRFN, PI, and FLC for these two new sampling intervals are shown as in Table 1. Although the parameters of the TRFN controller were obtained at $T_S = 45$, due to its recurrent property, we can still achieve good control performance without performing additional on-line learning. As a demonstration, control results of TRFN and FLC with $T_S = 60$ s and results of TRFN and PI with $T_S = 15$ s are shown in Figs. 14 and 15, respectively. The result in Fig. 14 shows that the steady state error by FLC is about 2 °C higher than that by TRFN for reference temperature 40 °C.

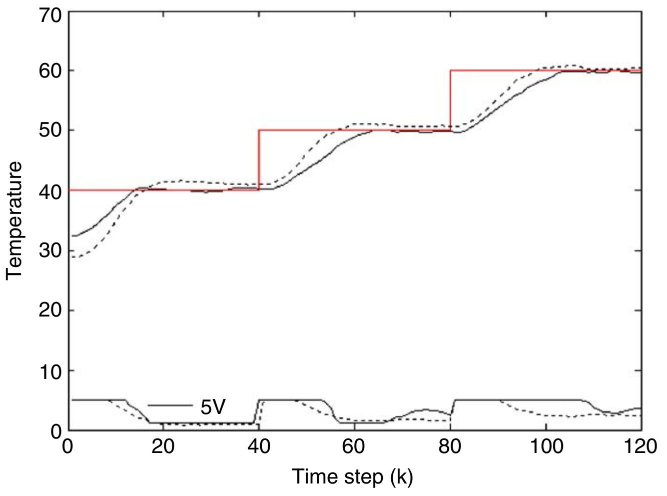


Fig. 13. Temperature control results obtained by TRFN (solid line) and FLC (dotted line).

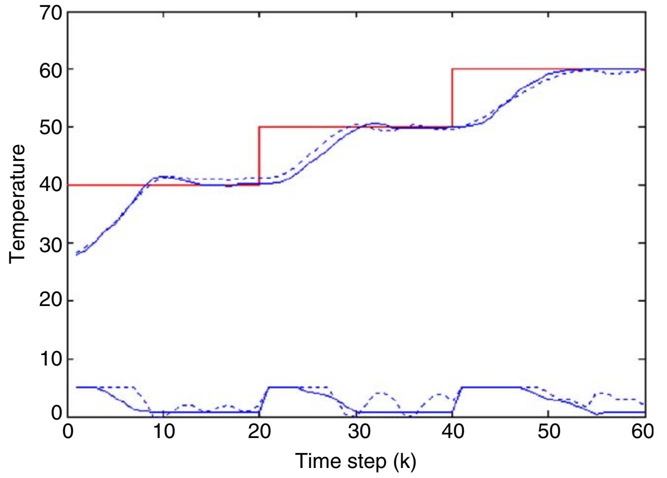


Fig. 14. Control results of TRFN (solid line) and FLC (dotted line) when T_S is increased from 45 to 60 s.

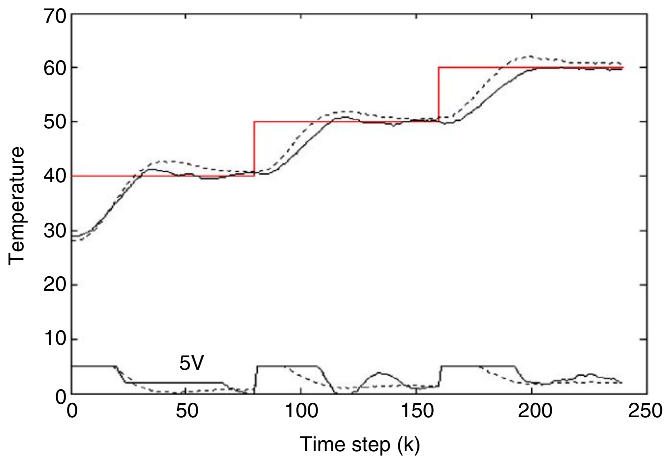


Fig. 15. Control results of TRFN (solid line) and PI (dotted line) when T_S is decreased from 45 to 15 s.

5. Conclusion

This paper proposes a TRFN controller for the temperature control of a rubber injection-molding machine. This time-delayed property in this machine complicates the control problem. We choose TRFN as the controller since it is characterized by high learning performance and recurrent structure. The experiments have shown that good performance is maintained with TRFN controller, even with the variation in the sampling interval. The training process of TRFN is simple, and the adopted direct inverse control configuration is easy to implement without identifying the machine. Application of the design process to other time-delayed plant control problems will be studied in future.

Acknowledgements

This work was supported by the National Science Council, Republic of China, under Grant number NSC 94-2213-E-005-014.

References

[1] D.W. Clarke, C. Mohtadi, P.S. Tuffs, Generalized predictive control part I, the basic algorithm, Automatica 23 (2) (1998) 137–148.

[2] Y. Frayman, L.P. Wang, A fuzzy neural approach to speed control of an elastic two-mass system, in: Proceedings of International Conference Computational Intelligence and Multimedia Applications, Queensland, 1997, pp. 341–345.

[3] Y. Frayman, L. Wang, A dynamically constructed fuzzy neural controller for direct model reference adaptive control of multi-input-multi-output nonlinear processes, Soft Comput. 6 (2002) 244–253.

[4] V. Gorrini, H. Bersini, Recurrent fuzzy systems, in: Proceedings of the IEEE International Conference on Fuzzy Systems, vol. 1, Orlando, FL, 1994, pp. 193–198.

[5] J.S. Jang, ANFIS: adaptive-network-based fuzzy inference system, IEEE Trans. Syst. Man Cybernet. 23 (3) (1993) 665–685.

[6] C.F. Juang, A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithms, IEEE Trans. Fuzzy Systems 10 (2) (2002) 155–170.

[7] C.F. Juang, An automatic building approach to special Takagi-Sugeno fuzzy network for unknown plant modeling and stable control, Asian J. Control 5 (2) (2003) 176–186.

[8] C.F. Juang, C.T. Lin, An on-line self-constructing neural fuzzy inference network and its applications, IEEE Trans. Fuzzy Systems 6 (1998) 12–32.

[9] C.F. Juang, C.T. Lin, A recurrent self-organizing neural fuzzy inference network, IEEE Trans. Neural Networks 10 (1999) 828–845.

[10] M. Khalid, S. Omatu, R. Yusof, MIMO furnace control with neural networks, IEEE Trans. Control Syst. Technol. 1 (1993) 238–245.

[11] J.H. Kim, D.T. College, A. Gun, G. Do, Fuzzy model based predictive control, in: IEEE International Conference on Fuzzy Systems, vol. 1, Anchorage, AK, 1998, pp. 405–409.

[12] X. Li, Z. Chen, Z. Yuan, Generalized predictive control using fuzzy neural networks in T–S model, in: Proceedings of the Third World Congress on Intelligent Control and Automation, vol. 2, 2000, pp. 885–889.

[13] C.J. Lin, C.C. Chin, Prediction and identification using wavelet-based recurrent fuzzy neural networks, IEEE Trans. Syst. Man Cybernet. B: Cybernet. 34 (5) (2004) 2144–2154.

[14] C.T. Lin, C.S.G. Lee, Neural Fuzzy Systems: A Neural-Fuzzy Synergism to Intelligent Systems, Prentice-Hall, Englewood Cliffs, NJ, 1996.

[15] C.T. Lin, C.F. Juang, C.P. Li, Temperature control with a neural fuzzy inference network, IEEE Trans. Syst. Man Cyber. C: Appl. Rev. 29 (3) (1999) 440–451.

[16] M. Mahfouf, M.F. Abbod, D.A. Linkens, Online elicitation of Mamdani-type fuzzy rules via TSK-based generalized predictive control, IEEE Trans. Syst. Man Cybernet. B: Cybernet. 33 (3) (2003) 465–475.

[17] P.A. Mastorocostas, J.B. Theocharis, A recurrent fuzzy-neural model for dynamic system identification, IEEE Trans. Syst. Man Cyber. B: Cybernet. 32 (2) (2002) 176–190.

[18] G.C. Mouzouri, J.M. Mendel, Dynamic nonsingleton fuzzy logic systems for nonlinear modeling, IEEE Trans. Fuzzy Systems 5 (2) (1997) 199–208.

[19] K.S. Narendra, K. Parthasarathy, Identification and control of dynamical systems using neural networks, IEEE Trans. Neural Network 1 (1) (1990) 4–27.

[20] C.L. Phillips, H.T. Nagle, Digital Control System Analysis and Design, Prentice-Hall, Englewood Cliffs, NJ, 1995.

[21] Surmann, M. Maniadakis, Learning feedforward and recurrent fuzzy systems: a genetic approach, J. Syst. Architecture 47 (2001) 649–662.

[22] J. Tanomaru, S. Omatu, Process control by on-line trained neural controllers, IEEE Trans. Ind. Electrons 39 (6) (1992) 511–521.

[23] W.L. Tung, C. Quek, GenSoFNN: a generic self-organizing fuzzy neural network, IEEE Trans. Neural Networks 13 (5) (2002) 1075–1086.

[24] Y.C. Wang, C.J. Chien, C.C. Teng, Direct adaptive iterative learning control of nonlinear systems using an output-recurrent fuzzy neural network, IEEE Trans. Syst. Man Cybernet. B: Cybernet. 34 (3) (2004) 1348–1359.

[25] R.J. Williams, D. Zipser, A learning algorithm for continually running recurrent network, Neural Comput. 1 (2) (1989) 270–280.

[26] J. Zhang, A.J. Morris, Recurrent neuro-fuzzy networks for nonlinear process modeling, IEEE Trans. Neural Networks 10 (2) (1999) 313–326.

Chia-Feng Juang received the B.S. and Ph.D. degrees in control engineering from the National Chiao-Tung University, Hsinchu, Taiwan, ROC, in 1993 and 1997, respectively. From 1997 to 1999, he was in military service. From 1999 to 2001, he was an Assistant Professor of the Department of Electrical Engineering at the Chung Chou Institute of Technology. In 2001, he joined the National Chung Hsing University, Taichung, Taiwan, ROC, where he is currently an Associate Professor of Electrical Engineering. His current research interests are computational intelligence, intelligent control, computer vision, speech signal processing, and FPGA chip design.

Shui-Tien Huang received the B.S. degree in Electrical Engineering from the National Chung-Hsing University, Taichung, Taiwan, ROC, in 2004. His current research interests are intelligent control and neural fuzzy network.

Fun-Bin Duh received the B.S degree in Electronic Engineering and M.S. degree in Automatic Control Engineering from the Feng-Chia University, Taichung, Taiwan, ROC, in 1981 and 1983, respectively, and the Ph.D. degree in the Department of Electrical and Control Engineering from the National Chiao-Tung University, Hsinchu, Taiwan, ROC, in 2003. Since 1983, he has been with the Chung-Shan Institute of Science and Technology, Taiwan. He is currently a part-time assistant Professor of Electronic Engineering at Feng-Chia University. His current research interests are digital signal processing, neural network, estimation theory, radar-tracking system, servo control system, and VHDL digital system design.