Published in IET Control Theory and Applications
Received on 16th August 2007
Revised on 29th March 2008
doi: 10.1049/iet-cta:20070315

ISSN 1751-8644

Robust fuzzy-neural sliding-mode controller design via network structure adaptation

P.-Z. Lin[1] *C.-F. Hsu*[2] *T.-T. Lee*[3] *C.-H. Wang*[1]

[1]*Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu 300, Taiwan, Republic of China*
[2]*Department of Electrical Engineering, Chung Hua University, Hsinchu 300, Taiwan, Republic of China*
[3]*Department of Electrical Engineering, National Taipei University of Technology, Taipei 106, Taiwan, Republic of China*
E-mail: fei@cn.nctu.edu.tw

Abstract: A robust fuzzy-neural sliding-mode control (RFSC) scheme for unknown nonlinear systems is proposed. The RFSC system is composed of a computation controller and a robust controller. The computation controller containing a self-structuring fuzzy-neural network (SFNN) identifier is the principle controller, and the robust controller is designed to achieve L_2 tracking performance. The SFNN identifier uses the structure- and parameter-learning phases to perform the estimation of the unknown system dynamics. The structure-learning phase consists of the growing of membership functions, the splitting of fuzzy rules and the pruning of fuzzy rules, and thus the SFNN identifier can avoid the time-consuming trial-and-error tuning procedure for determining the network structure of fuzzy neural network. Finally, the proposed RFSC system is applied to three nonlinear dynamic systems. The simulation results show that the proposed RFSC system can achieve favourable tracking performance by incorporating SFNN identifier, sliding-mode control and robust control techniques.

1 Introduction

The fuzzy-neural network (FNN) that incorporates the advantages of fuzzy inference and neuro-learning has been an interesting topic. The FNN possesses the merits of the low-level learning and computational power of neural network and the high-level human knowledge representation from fuzzy theory [1, 2]. Recently, the FNNs are increasingly receiving attention in solving the control problems [3–7]. For the FNN-based adaptive control approaches in [3–7], the structure of FNN should be determined in advance by trial-and-error. However, it is difficult to consider the balance between the rule number and the desired performance. If the number of fuzzy rules is chosen too large, the computational load is heavy so that they are not suitable for practical applications. If the number of fuzzy rules is chosen too small, the learning performance may not be good enough to achieve desired performance.

To solve the problem of structure determination in FNN approaches, much interest has been focussed on the self-structuring fuzzy-neural network (SFNN) approach [8–10]. The self-structuring approach demonstrates the property of automatically generating rules of FNN without the preliminary knowledge. In general, the mathematical descriptions of the existing rules can be expressed as a fuzzy cluster where each cluster corresponds to a fuzzy IF-THEN rule [10]. As usually seen in other self-structuring approaches, a new membership function is generated when a new input signal is far away from the current clusters, and an existing rule is cancelled when the fuzzy rule is insignificant.

Recently, the SFNNs also have been widely adopted for the control of complex dynamic systems owing to its good generalisation capability, structure adaptation and simple computation [11–14]. Some of them use the gradient descent method to derive the parameter learning algorithms [11, 14]; however, stability analysis has not been performed yet. Some of them use the Lyapunov function to derive the parameter-learning algorithms; however, the complex design procedure is not suitable for practical applications [12, 13]. Lin *et al.* use a similarity measure method to avoid a newly generated membership function being similar to the existing ones; however, the network structure would grow large, as

the input data has large variations [11, 14]. In [12, 13], an error reduction ratio with orthogonal-triangular (QR) decomposition is used to prune rules; however, the computational load is too heavy. In summary, a comparison between the proposed robust fuzzy-neutral sliding-mode control (RFSC) and [10, 14, 15] is made. In [10], an independent component analysis (ICA)-mixture-model-based self-constructing FNN is proposed. It is not suitable for controller design. In [14], a similarity measure method is used to avoid a newly generated membership function being similar to the existing ones. However, the computational load of pruning algorithm is too heavy. In [15], a self-structuring neural network is proposed; however, the proposed approach cannot avoid the structure of neural network growing endlessly. The proposed RFSC can grow the membership functions, split the fuzzy rules and prune the fuzzy rules of SFNN automatically. The structure of neural network can avoid growing endlessly and the computational load can be reduced.

In this study, a novel SFNN that includes the growing of membership functions, the splitting of fuzzy rules and the pruning of fuzzy rules is developed. The idea of these algorithms is given as follows. The growing method can be considered when the input values will be far away from the edge of existing membership functions as shown in [10]. The split algorithm [15] is performed when the weight updating rate of certain neuron is larger, since larger weight updating rate will cause a larger updating of certain weight values, and the precise approximation is difficult to capture. The pruning algorithm [16] will eliminate those rules that make a very small contribution to the neural network output. Because the error between the real function and the estimation function gradually converges to a small bound after several iterations, and some fuzzy rules that are generated in the transient period may be less or never used. Therefore the structure learning phase of the proposed SFNN possesses several techniques, including the growing of membership functions, the splitting of fuzzy rules and the pruning of fuzzy rules.

For the conventional FNN-based adaptive control approaches, the structure of FNN should be determined in advance by trial-and-error. By the proposed SFNN approach, this paper develops a novel FNN-based adaptive control approaches, which the structure of FNN can automatically generate fuzzy rules to avoid the time-consuming trial-and-error tuning procedure. This paper proposes a RFSC system, which is composed of a computational controller and a robust controller. The computation controller containing the SFNN identifier is the principle controller, and the robust controller with desired attenuation level is designed to achieve L_2 tracking performance. The structure-learning phase of the SFNN identifier possesses the ability of on-line generation and elimination of fuzzy rules to achieve optimal network structure, and the parameter-learning phase adjusts the parameters of membership functions and fuzzy rules to achieve favourable approximation performance. By the proposed structure adaptation, the annoyance will be solved where the balance between the rule number and the desired control performance is difficult to consider. To investigate the effectiveness of the proposed control scheme, the RFSC system is applied to three nonlinear dynamic systems. Simulation results demonstrate that the proposed RFSC scheme can achieve favourable tracking performance without unknown system dynamics. The SFNN identifier can estimate the system dynamics and achieve approximation performance in the presence of external disturbance in a real-time environment.

2 Sliding-mode control design

Consider the nth-order nonlinear dynamic system of the form

$$\begin{cases} x^{(n)} = f(\mathbf{x}) + u + d \\ y = x \end{cases} \tag{1}$$

where $\mathbf{x} = [x\, \dot{x} \cdots x^{(n-1)}]^{\mathrm{T}}$ is the state vector of the system, y is the output of system, $f(\mathbf{x})$ is the system dynamics, d is the external disturbance and u is the control effort. Assume that the system dynamics is well known, (1) can represent the nominal model of the nonlinear dynamic system

$$\begin{aligned} x^{(n)} &= f_n(\mathbf{x}) + u + d_n \\ y &= x \end{aligned} \tag{2}$$

where $f_n(\mathbf{x})$ is a mapping that represents the nominal behaviour of $f(\mathbf{x})$ and d_n is the nominal value of d. If uncertainties occur, that is, parameters of the system deviate from the nominal value, the controlled system can be modified as

$$\begin{aligned} x^{(n)} &= f_n(\mathbf{x}) + u + d_n + z \\ y &= x \end{aligned} \tag{3}$$

where z is the lumped uncertainty which is defined as $z = \Delta f(\mathbf{x}) + \Delta d$ with the assumption $|z| \leq Z$ in which Z is a given positive constant, $\Delta f(\mathbf{x})$ denotes the system uncertainties and Δd denotes the disturbance uncertainties. The control objective is to find a control law so that the system output y can track a command trajectory y_{c}. A tracking error and a sliding surface are defined as

$$e = y_{\mathrm{c}} - y \tag{4}$$

and

$$s = e^{(n-1)} + k_n e^{(n-2)} + \cdots + k_2 e + k_1 \int_0^t e\, d\tau \tag{5}$$

where k_i, $i = 1, 2, \ldots, n$ are the positive constants. The sliding-mode control law is given as [17]

$$u_{\mathrm{sc}} = u_{\mathrm{eq}} + u_{\mathrm{ht}} \tag{6}$$

where the equivalent controller u_{eq} is represented as

$$u_{eq} = -f_n(\mathbf{x}) - d_n + x_c^{(n)} + \mathbf{k}^T\mathbf{e} \quad (7)$$

in which $\mathbf{e} = [e\ \dot{e}\ \cdots\ e^{(n-1)}]^T$ and $\mathbf{k} = [k_1\ k_2\ \cdots\ k_n]^T$, and the hitting controller u_{ht} is designed to guarantee the system stability as

$$u_{ht} = Z\text{sgn}(s) \quad (8)$$

where sgn(·) is the sign function. Substituting (6)–(8) into (3) yields

$$e^{(n)} + k_n e^{(n-1)} + \cdots + k_2\dot{e} + k_1 e = -z - Z\text{sgn}(s) = \dot{s} \quad (9)$$

An important concept of sliding-mode control is to make the system satisfy the reaching condition and guarantee sliding condition. Consider the candidate Lyapunov function in the following form as

$$V_1 = \frac{1}{2}s^2 \quad (10)$$

Differentiating (10) with respect to time and using (9), we obtain

$$\begin{aligned}\dot{V}_1 &= s\dot{s} = -zs - Z|s| \le |z||s| - Z|s| \\ &= -(Z - |z|)|s| \le 0\end{aligned} \quad (11)$$

In summary, the sliding-mode controller in (6) can guarantee the stability in the sense of the Lyapunov theorem [17]. Because the system dynamics and the external disturbance may be unknown or perturbed, the sliding-mode control law u_{sc} cannot be implemented. Moreover, a large control gain Z is often required in order to minimise the reaching time from the initial state to the switching surface. The selection of the control gain Z, which is related to the magnitude of uncertainties, keeps the trajectory on the sliding surface [17]. Nevertheless, the parameter variations of the system are difficult to measure for practical applications. A conservative control law with a large control gain Z is usually considered, but the unnecessary jumping movement on the switching surface may yield and cause an outcome of a large amount of chattering. The chattering phenomena in control efforts will wear the bearing mechanism and excite unmodelled dynamics.

3 Robust fuzzy-neural sliding-mode control design

3.1 Description of SFNN

The structure of the SFNN has four layers of neural network: the input, the membership function, the rule and the output layers. Nodes at layer 1 are input nodes (linguistic nodes) that represent input linguistic vector. Nodes at layer 2 are term nodes which act as membership functions to represent the terms of the respective linguistic vector. Each node at layer 3 is a fuzzy rule. Layer 4 is the output layer, where the node in this layer is the output of SFNN. The interactions for those layers are given as follows.

Layer 1 – Input layer: For every node i, in this layer, the net input and the net output are represented as

$${}^1net_i = {}^1x_i \quad (12)$$

$${}^1y_i = {}^1f_i({}^1net_i) = {}^1net_i, \quad i = 1, 2, \ldots, L \quad (13)$$

where ${}^1x_i$ represents the ith input to the node of layer 1 and L is the total number of input variables. They mean that output equals input in this layer. This layer of SFNN just executes the transmission work.

Layer 2 – Membership layer: In this layer, each node performs a membership function and acts as a unit of memory. The bell-shaped function is adopted as the membership function. For the ith input, the corresponding net input and output of the jth node can be expressed as

$${}^2net_{ij} = -\frac{\left({}^2x_i - {}^2m_{ij}\right)^2}{\left({}^2\sigma_{ij}\right)^2} \quad (14)$$

$${}^2y_{ij} = {}^2f_{ij}\left({}^2net_{ij}\right) = \exp\left({}^2net_{ij}\right), \quad j = 1, 2, \ldots, M \quad (15)$$

where ${}^2m_{ij}$ is the mean, ${}^2\sigma_{ij}$ is the standard deviation and M is the total number of membership functions with respect to the respective input node. In this study, the input linguistic variable is the tracking error vector.

Layer 3 – Rule layer: Each node k in this layer is denoted by Π which multiplies the incoming signals and outputs the result of the product. For the kth rule node, the operation of the net input and output of this layer is presented as

$${}^3net_k = \prod {}^3w_{ij}\,{}^3x_{ij} \quad (16)$$

$${}^3y_k = {}^3f_k({}^3net_k) = {}^3net_k, \quad k = 1, 2, \ldots, N \quad (17)$$

where ${}^3x_{ij}$ represents the i, jth input to the kth node of layer 3, ${}^3w_{ij}$ between the membership and the rule layers are assumed as unity and N is the total number of fuzzy rules.

Layer 4 – Output layer: The single node o in this layer is labelled as Σ, which computes the overall output as the summation of all incoming signals. It executes the sum-of-weighting defuzzification. The description of the net input and output is expressed as

$${}^4net_o = \sum_k {}^4w_k\,{}^4x_k \quad (18)$$

$${}^4y_o = {}^4f_o({}^4net_o) = {}^4net_o \quad (19)$$

doi: 10.1049/iet-cta:20070315

where ${}^4w_k$ is the output action strength of the output associated with the kth rule, ${}^4x_k$ represents the kth input to the node of layer 4 and ${}^4y_o$ is the output of SFNN. For the FNN approaches in [3–7], the structure of the FNN is determined in advance by trial-and-error, but it is difficult to consider the balance between the rule number and the desired control performance. In this paper, the structure-learning algorithm, which is composed of the growing of membership functions, the splitting of fuzzy rules and the pruning of fuzzy rules, is proposed to solve this problem. The descriptions for those algorithms are given as follows.

For the method of the growing of membership functions, the concept, which decides whether to add a new node (membership function) in layer 2 and the associated fuzzy rule in layer 3, will be introduced. In order to construct fuzzy rules of the SFNN, a new rule is generated when a new input signal is far away from the current clusters. A cluster corresponds to a fuzzy IF-THEN rule. Each cluster in the product space of the input–output data represents a rule in the rule base. The firing strength of a rule can be represented as the degree that the incoming data ${}^1x_i$ belongs to the cluster. If the value of firing strength is too small, it represents that the input value is on the edge of range of the existing membership functions. Under this situation, the output will cause a bad output performance. Therefore a new node (membership function) should be added at this moment. The firing strength obtained from (17) is used as the degree measure [10]

$$\beta_k = {}^3y_k, \quad k = 1, 2, \ldots, N \tag{20}$$

where N is the number of the existing rules. Define the maximum degree $\beta_{\max}$ among β_k as

$$\beta_{\max} = \max_{1 \le k \le N} \beta_k \tag{21}$$

It can be observed that if $\beta_{\max} \le G_{\text{th}}$ where $G_{\text{th}} \in (0, 1)$ is a pre-given threshold, the incoming data is far away from the edge of range of the existing membership functions. Hence, a new membership function should be generated. The mean and the standard deviation of the new membership function and the corresponding weight are selected as follows

$$m_i^{\text{new}} = {}^1x_i \tag{22}$$

$$\sigma_i^{\text{new}} = \sigma_i \tag{23}$$

$$w^{\text{new}} = 0 \tag{24}$$

where ${}^1x_i$ is the new incoming data and σ_i is a pre-specified constant. If the unknown control system dynamics is complex, more membership functions can be created by choosing a large G_{th}.

Next, the split of the kth fuzzy rule is executed if the following condition is satisfied [15]

$$\frac{|{}^4\dot{\hat{w}}_k|}{\sum_{k=1}^{N} |{}^4\dot{\hat{w}}_k|} \ge S_{\text{th}}, \quad k = 1, 2, \ldots, N \tag{25}$$

where S_{th} denotes a split threshold value and the tuning law ${}^4\dot{\hat{w}}_k$ will be derived in the next subsection. On the left-hand side of (25), the denominator implies the total sum of the positive weight variations and the numerator represents the positive weight variations connecting to the kth node. The proposed split algorithm is derived from the observation that if the updating of the weight values is relatively large, the precise approximation is difficult to capture. If (25) is satisfied, a new neuron is duplicated to spread the relatively large variation of the weights. If the designer wants more neurons in the SFNN, because the nonlinear plant is complex, more neurons can be created by choosing a small S_{th}, and vice versa. The weight value connecting the k'th node is considered as

$${}^4w_{k'} = (1 - \alpha) \cdot {}^4w_k \tag{26}$$

where α is a positive constant. The original weight value connecting the kth node is changed as

$${}^4w_k = \alpha \cdot {}^4w_k \tag{27}$$

This method is induced from the fact that the weights connected to the newly created neuron will share the large variations of the weights. When the estimated parameters are all the optimal parameters in the training procedure, that is, the learning algorithm searches the optimal solutions, the tuning law ${}^4\dot{\hat{w}}_k$ will be equal to zero for all k. Note that if the condition satisfies ${}^4\dot{\hat{w}}_k = 0$ for all k, the operation of split does not be executed.

To avoid the endless growing of network structure and the overload computation, the algorithm of the pruning of fuzzy rules is developed to eliminate inappropriate fuzzy rules. When the rth firing strength β_r is smaller than a threshold value P_{th}, it means that the relationship becomes weak between the input and the rth rule. This fuzzy rule may be less or never used. Then, we will gradually reduce the value of the rth referring index when the rth firing strength β_r satisfies our setting condition continuously. The referring index is determined as [16]

$$I_r = \begin{cases} I_r^p \exp(-\tau), & \text{if } \beta_r < P_{th} \\ I_r^p, & \text{if } \beta_r \ge P_{th} \end{cases}, \quad r = 1, 2, \ldots, N \tag{28}$$

where I_r is the referring index of the rth rule and its initial value is 1, P_{th} is the pruning threshold value, τ is the decayed constant, and I_r^p denotes the most recent I_r. If $I_r \le I_{\text{th}}$ is satisfied, where I_{th} is another pre-given threshold the rth

fuzzy rule is pruned. Moreover, the computational load should also be decreased. If the computational load is the important issue for practical implement, we can choose a large P_{th} so that more fuzzy rules can be pruned. In summary, the flow chart of the structure-learning algorithm is shown in Fig. 1. The major contributions of using these algorithms for training SFNN structure are: (1) SFNN can be operated directly without spending much time on pre-determining membership functions and fuzzy rules; (2) the computational load can be reduced simultaneously.

3.2 Parameter-learning algorithm

Since the system dynamics $f(\mathbf{x})$ and the external disturbance d are unknown in practical applications, the equation of the nonlinear dynamic system in (1) can be rewritten as

$$\begin{cases} x^{(n)} = F(\mathbf{x}) + u \\ y = x \end{cases} \tag{29}$$

where $f(\mathbf{x})$ is called the lumped system dynamics and defined as $f(\mathbf{x}) = f(\mathbf{x}) + d$. As shown in Fig. 2, the proposed RFSC system is composed of a computational controller and a robust controller, that is

$$u_{\text{fsc}} = u_{\text{cc}} + u_{\text{rc}} \tag{30}$$

where the computational controller is designed as

$$\begin{aligned} u_{\text{cc}} &= -F(\mathbf{e}, \hat{\mathbf{\Theta}}) + x_c^{(n)} + \mathbf{k}^T\mathbf{e} \\ &= -\hat{F} + x_c^{(n)} + \mathbf{k}^T\mathbf{e} \end{aligned} \tag{31}$$

in which the SFNN identifier $F(\mathbf{e}, \hat{\mathbf{\Theta}}) = \hat{F}$ is used to on-line estimate the lumped system dynamics $F(\mathbf{x})$ in (29). $\mathbf{e}$ and $\hat{\mathbf{\Theta}} = [{}^4\hat{w}_k^2\ \hat{m}_{ij}^2\ \hat{\sigma}_{ij}]^T$ are the input vector and estimated parameter vector of the SFNN identifier, respectively. In the sliding-mode control approach, if the sliding condition is satisfied, the error terms will converge to zero as time approaches to infinite. Furthermore, bounds on s can be directly translated to bounds on the tracking error, because the sliding surface is composed of the error terms. In particular, once the system is on the surface, the system trajectories will remain on the surface and the tracking error tends exponentially to zero. Consequently, the stability of closed-loop system can be guaranteed [17] and Section 2 has the proof of stability. This motion is the same as minimising the error function if the error function is selected as the cost function. Hence, the sliding condition, which substitutes for the error function, becomes the cost function. The on-line learning algorithm is a gradient descent algorithm in the space of network parameters and aims to minimise $s\dot{s}$ for

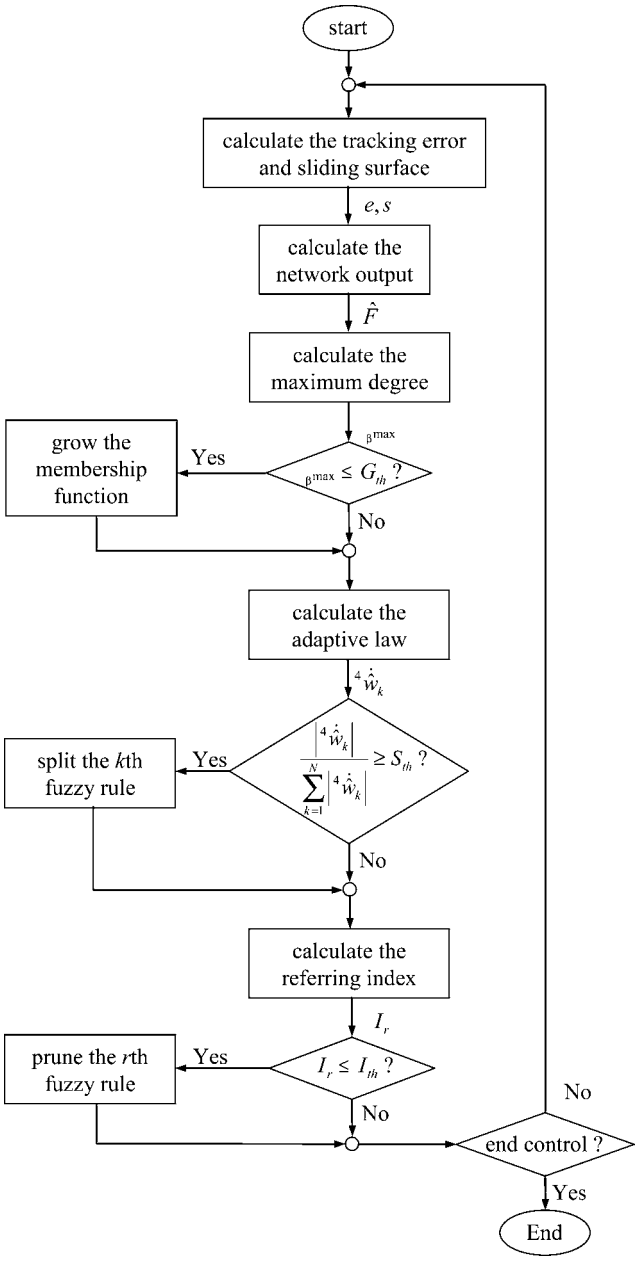

Figure 1 *Flow chart of the proposed structure adaptation algorithm*

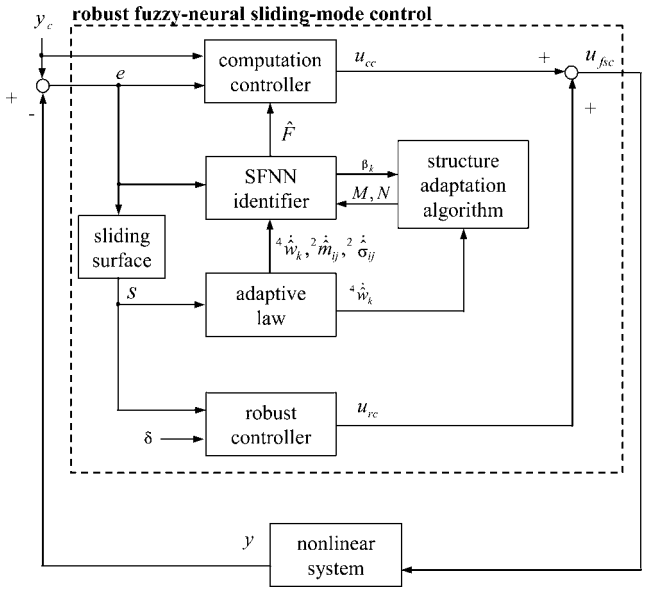

Figure 2 *Block diagram of RFSC system*

doi: 10.1049/iet-cta:20070315

achieving fast convergence of s. Substituting (30) into (29) and using (5) yield

$$e^{(n)} + k_n e^{(n-1)} + \cdots + k_2 \dot{e} + k_1 e = \hat{F} - F - u_{rc} = \dot{s} \quad (32)$$

Multiply both sides of (32) by s to yield

$$s\dot{s} = (\hat{F} - F - u_{rc})s \quad (33)$$

According to the gradient descent method [3], the adaptive law of weights in the output layer can be derived as

$$^4\dot{\hat{w}}_k = -\eta_w \frac{\partial s\dot{s}}{\partial(^4\hat{w}_k)} = -\eta_w \frac{\partial s\dot{s}}{\partial \hat{F}} \frac{\partial \hat{F}}{\partial(^4\hat{w}_k)} = -\eta_w \cdot s \cdot {}^4x_k \quad (34)$$

where the positive constant η_w is a learning rate. In order to easily derive the update laws of mean and variance, the parameter ξ is derived in advance as

$$\begin{aligned}\xi &= \frac{\partial s\dot{s}}{\partial \hat{F}} \frac{\partial \hat{F}}{\partial(^4net_o)} \frac{\partial(^4net_o)}{\partial(^3y_k)} \frac{\partial(^3y_k)}{\partial(^3net_k)} \frac{\partial(^3net_k)}{\partial(^2y_{ij})} \frac{\partial(^2y_{ij})}{\partial(^2net_{ij})} \\ &= s \cdot {}^4w_k \cdot {}^3y_k \end{aligned} \quad (35)$$

The update laws of means and variances, $^2m_{ij}$ and $^2\sigma_{ij}$, can be obtained by the gradient search algorithm [3] as

$$\begin{aligned}{}^2\dot{\hat{m}}_{ij} &= -\eta_m \frac{\partial s\dot{s}}{\partial(^2\hat{m}_{ij})} = -\eta_m \frac{\partial s\dot{s}}{\partial \hat{F}} \frac{\partial \hat{F}}{\partial(^4net_o)} \frac{\partial(^4net_o)}{\partial(^3y_k)} \frac{\partial(^3y_k)}{\partial(^3net_k)} \frac{\partial(^3net_k)}{\partial(^2y_{ij})} \\ &\times \frac{\partial(^2y_{ij})}{\partial(^2net_{ij})} \frac{\partial(^2net_{ij})}{\partial(^2\hat{m}_{ij})} = -\eta_m \xi \frac{2\left(^2x_i - {}^2\hat{m}_{ij}\right)}{\left(^2\hat{\sigma}_{ij}\right)^2} \end{aligned} \quad (36)$$

and

$$\begin{aligned}{}^2\dot{\hat{\sigma}}_{ij} &= -\eta_\sigma \frac{\partial s\dot{s}}{\partial(^2\hat{\sigma}_{ij})} \\ &= -\eta_\sigma \frac{\partial s\dot{s}}{\partial \hat{F}} \frac{\partial \hat{F}}{\partial(^4net_o)} \frac{\partial(^4net_o)}{\partial(^3y_k)} \frac{\partial(^3y_k)}{\partial(^3net_k)} \frac{\partial(^3net_k)}{\partial(^2y_{ij})} \\ &\times \frac{\partial(^2y_{ij})}{\partial(^2net_{ij})} \frac{\partial(^2net_{ij})}{\partial(^2\hat{\sigma}_{ij})} = -\eta_\sigma \xi \frac{2\left(^2x_i - {}^2\hat{m}_{ij}\right)^2}{\left(^2\hat{\sigma}_{ij}\right)^3} \end{aligned} \quad (37)$$

where η_m and η_σ are the learning rates with positive constants.

3.3 Robust controller design

Approximating linear or nonlinear mapping through learning is one of most useful property of a neural network. The relationship between $F(\mathbf{x})$ and $F(\mathbf{e}, \hat{\mathbf{\Theta}})$ is described as

$$F(\mathbf{x}) - F(\mathbf{e}, \hat{\mathbf{\Theta}}) = \varepsilon \quad (38)$$

where ε denotes the modelling error. Substituting (38) into (32), (32) can be rewritten as follows

$$\dot{s} = -\varepsilon - u_{rc} \quad (39)$$

If ε exists, a robust controller will be considered to satisfy a specified L_2 tracking performance [18–20]

$$\int_0^T s^2(t)\,dt \le s^2(0) + \delta^2 \int_0^T \varepsilon^2(t)\,dt \quad (40)$$

where δ is a prescribed attenuation constant. If the system starts with an initial condition $s(0) = 0$, the L_2 tracking performance in (40) can be rewritten as

$$\sup_{\varepsilon \in L_2[0,T]} \frac{\|s\|}{\|\varepsilon\|} \le \delta \quad (41)$$

where $\|s\|^2 = \int_0^T s^2(t)\,dt$ and $\|\varepsilon\|^2 = \int_0^T \varepsilon^2(t)\,dt$. If $\delta = \infty$, this is the case of minimum error tracking control without disturbance attenuation. Define a Lyapunov function as

$$V_2 = \frac{1}{2}s^2 \quad (42)$$

Differentiating (42) with respect to time and using (39), we obtain

$$\dot{V}_2 = s\dot{s} = s(-\varepsilon - u_{rc}) \quad (43)$$

The robust controller is designed as

$$u_{rc} = \frac{\delta^2 + 1}{2\delta^2} s \quad (44)$$

and thus (43) can be rewritten as

$$\begin{aligned}\dot{V}_2 &= s\left(-\varepsilon - \frac{\delta^2 + 1}{2\delta^2} s\right) \\ &= -s\varepsilon - \frac{s^2}{2} - \frac{s^2}{2\delta^2} = -\frac{s^2}{2} - \frac{1}{2}\left(2s\varepsilon + \frac{s^2}{\delta^2}\right) \\ &= -\frac{s^2}{2} - \frac{1}{2}\left(\frac{s^2}{\delta^2} + 2s\varepsilon + \varepsilon^2\delta^2\right) + \frac{1}{2}\varepsilon^2\delta^2 \\ &= -\frac{s^2}{2} - \frac{1}{2}\left(\frac{s}{\delta} + \varepsilon\delta\right)^2 + \frac{1}{2}\varepsilon^2\delta^2 \le -\frac{1}{2}s^2 + \frac{1}{2}\varepsilon^2\delta^2 \end{aligned} \quad (45)$$

Assume $\varepsilon \in L_2[0, T]$, $\forall T \in [0, \infty)$. Integrating the above equation from $t = 0$ to $t = T$ yields

$$V_2(T) - V_2(0) \le -\frac{1}{2}\int_0^T s^2 dt + \frac{1}{2}\delta^2 \int_0^T \varepsilon^2\,dt \quad (46)$$

Since $V_2(t) \geq 0$, we can rearrange (46) as

$$\frac{1}{2}\int_0^T s^2\,dt \leq V_2(0) + \frac{1}{2}\delta^2 \int_0^T \varepsilon^2\,dt \tag{47}$$

Using (42), this inequality is equivalent to inequality (40). Since $V_2(0)$ is finite, if the approximation error $\varepsilon \in L_2$, that is $\int_0^T \varepsilon^2(\tau)\,d\tau < \infty$, it implies that $\lim_{t\to\infty}|s| = 0$. Then, the RFSC can achieve L_2 tracking performance with attenuation of disturbances including approximation errors and external disturbances.

4 Simulation results

Chaotic dynamic system is a nonlinear deterministic system that displays complex, noisy-like and unpredictable behaviour. It can be observed in many nonlinear circuits and mechanical systems [21]. For control engineers, control of a chaotic dynamic system has become a significant research topic in physics, mathematics and engineering communities [22–25]. However, some of them cannot achieve favourable control performance [22–24] and some of them require overly complex design procedures [25]. In this section, the proposed RFSC scheme is applied to two chaotic dynamic systems and a simple nonlinear system to verify its effectiveness. It should be emphasised that parameters and network structure of the SFNN can be tuned on-line by the proposed algorithm. Moreover, the appropriate network structure will be obtained by the methods of the growing of membership functions, the splitting of fuzzy rules and the pruning of fuzzy rules during the process of training adjustable parameters.

Example 1: Consider a second-order chaotic system such as the Duffing's equation describing a special nonlinear circuit or a pendulum moving in a viscous medium as follows [22]

$$\ddot{x} = f(\mathbf{x}) + u \tag{48}$$

where $f(\mathbf{x}) = -p\dot{x} - p_1 x - p_2 x^3 + q\cos(\omega t)$ is the system dynamics, t is the time variable, ω is the frequency, u is the control force, and p, p_1, p_2 and q are real constants. Depending on the choice of these constants, the system is known that the solutions of (48) may exhibit periodic, that is it is almost periodic and chaotic behaviour. In order to observe the chaotic unpredictable behaviour, the open-loop system behaviour, that is, $u = 0$, is simulated with $p = 0.4$, $p_1 = -1.1$, $p_2 = 1.0$, $q = 1.95$ and $\omega = 1.8$. The phase plane plots with an initial condition point (0.5, 0) for $q = 1.95$ and $q = 7.00$ are shown in Figs. 3*a* and 3*b*, respectively. The time responses of the uncontrolled chaotic system with initial conditions (0.5, 0) and (0.6, 0) are shown in Figs. 3*c* and 3*d*, respectively. It is shown that the uncontrolled chaotic system has different trajectories for different q values and different initial points.

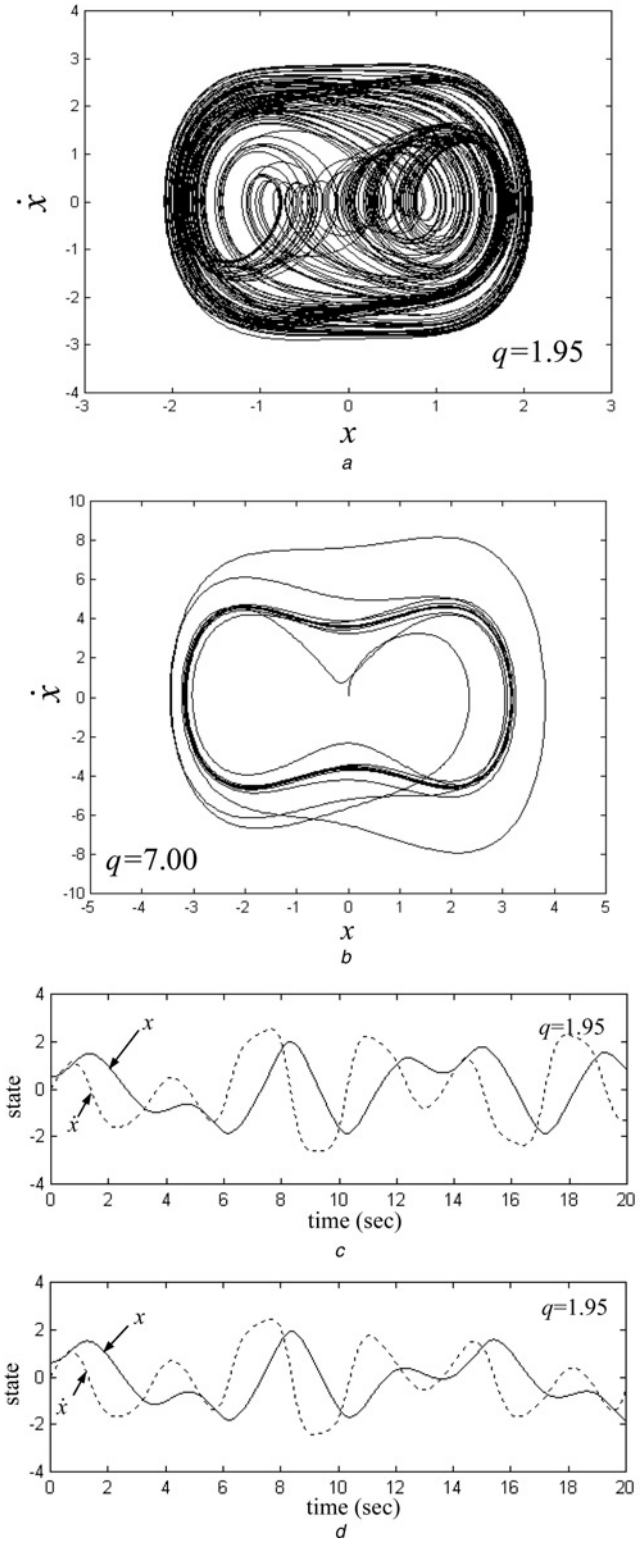

Figure 3 *Behaviour of uncontrolled chaotic system*

The parameters of RFSC are selected as $k_1 = 2$, $k_2 = 1$, $\eta_w = 40$, $\eta_m = \eta_\sigma = 2$, $G_{\text{th}} = 0.3$, $S_{\text{th}} = 0.5$, $I_{\text{th}} = 0.2$, $\alpha = 0.9$, $P_{\text{th}} = 0.15$, $\tau = 0.01$ and $\delta = 0.5$. All the gains in the proposed control system are chosen to achieve the best transient control performance considering the requirement of

IET Control Theory Appl., 2008, Vol. 2, No. 12, pp. 1054–1065
doi: 10.1049/iet-cta:20070315

stability and possible operating conditions. The simulation results of RFSC for $q = 1.95$ and $q = 7.00$ are shown in Figs. 4 and 5, respectively. The tracking responses of state x are shown in Figs. 4*a* and 5*a*; the tracking responses of state $\dot{x}$ are shown in Figs. 4*b* and 5*b*; the associated control efforts are shown in Figs. 4*c* and 5*c*; the number of fuzzy rules are shown in Figs. 4*d* and 5*d*; the responses of neural network

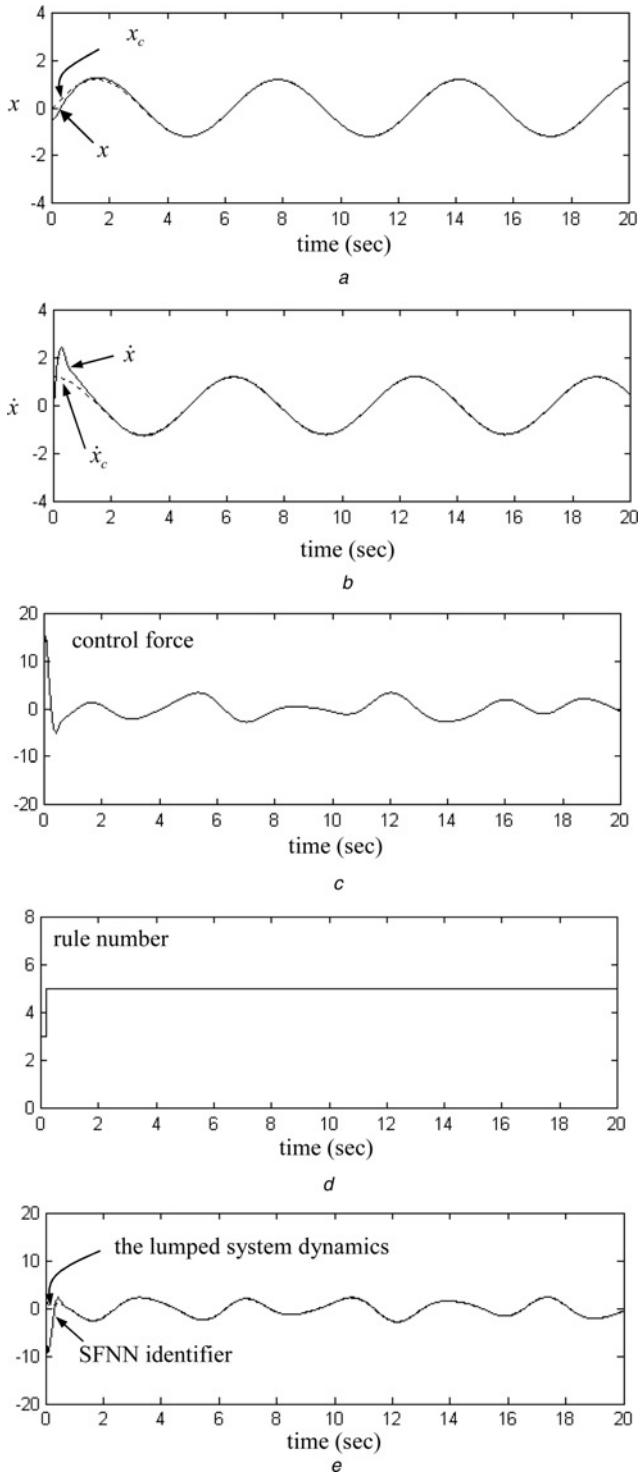

Figure 4 *Simulation results of RFSC system for q = 1.95*

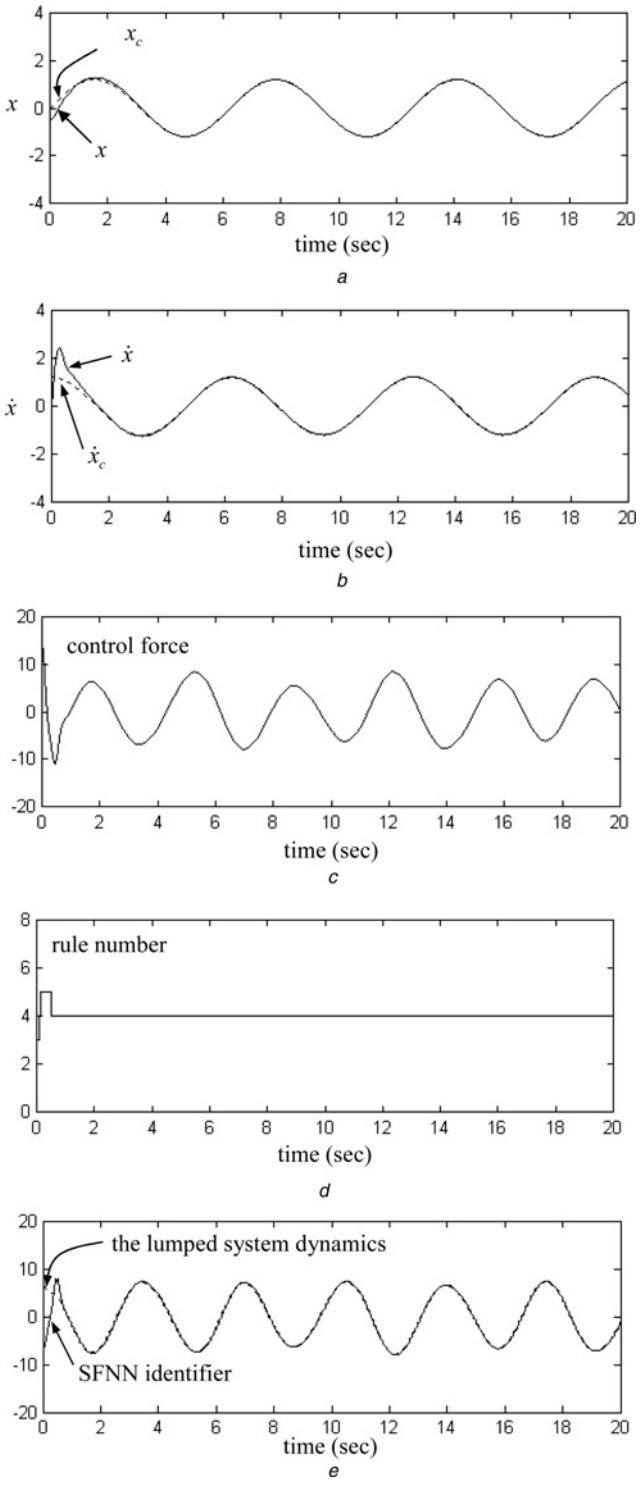

Figure 5 *Simulation results of RFSC system for q = 7.00*

are shown in Fig. 4*e* and 5*e*, respectively. The numbers of fuzzy rules for $q = 1.95$ and $q = 7.00$ are 5 and 4, respectively. It shows that the proposed network structure adaptation can grow membership functions, split fuzzy rules and prune fuzzy rules automatically for different nonlinear systems. From the simulation results, not only the perfect tracking responses can be achieved without any knowledge of system dynamics, but also the concise SFNN structure

can be obtained by applying the proposed self-structuring mechanism and the on-line learning algorithms.

Example 2: The third-order Chua's chaotic circuit, shown in Fig. 6, is a simple electronic system which consists of one linear resistor (R_1), two capacitors (C_1, C_2), one inductor (L_1), and one nonlinear resistor (λ). It owns very rich nonlinear dynamics such as chaos and bifurcations [23]. λ is defined as a cubic function as

$$\lambda(v_{C_1}) = av_{C_1} + cv_{C_1}^3 \quad (a < 0, c > 0) \tag{49}$$

The standard canonical form of Chua's circuit can be expressed as [24]

$$x^{(3)} = f(\mathbf{x}) + u + d \tag{50}$$

where $f(\mathbf{x}) = (14/1805)x - (168/9025)\dot{x} + (1/38)\ddot{x} - (2/45)((28/361)x + (7/95)\dot{x} + \ddot{x})^3$ is the system dynamic function, and a square-wave disturbance d is added at 10 sec with ± 2.0 amplitude and period 12 sec. The parameters of RFSC are selected as $k_1 = 3$, $k_2 = 3$, $k_3 = 1$, $\eta_w = 20$, $\eta_m = \eta_\sigma = 1$, $G_{\text{th}} = 0.3$, $S_{\text{th}} = 0.5$, $I_{\text{th}} = 0.2$, $\alpha = 0.9$, $P_{\text{th}} = 0.15$, $\tau = 0.01$ and $\delta = 0.5$. The choice of these parameters is also through some trails. To illustrate the effectiveness of RFSC design method, a comparison between the FNN-based adaptive control in [3] and SRFC systems is made. The parameters of the FNN-based adaptive control with three membership functions are selected as $k_1 = 3$, $k_2 = 3$, $k_3 = 1$, $\eta_w = 20$, $\eta_m = \eta_\sigma = 1$. First, the simulation results of the FNN-based adaptive control with fix-structure neural network are shown in Fig. 7. The tracking response of state x is shown in Fig. 7*a*; the tracking response of state $\dot{x}$ is shown in Fig. 7*b*; the tracking response of state $\ddot{x}$ is shown in Fig. 7*c*; the associated control effort is shown in Fig. 7*d*; the response of neural network is shown in Fig. 7*e*, respectively. The simulation results show that the favourable tracking performance can be achieved. However, the degenerate tracking responses are caused when the disturbance occurs. Then, the proposed RFSC with SFNN identifier is applied to the Chua's chaotic circuit again. The simulation results of the RFSC are shown in Fig. 8. The tracking response of state x is shown in Fig. 8*a*; the tracking response of state $\dot{x}$ is shown in Fig. 8*b*; the tracking response of state $\ddot{x}$ is shown in Fig. 8*c*; the associated control effort is shown in Fig. 8*d*; the number of fuzzy rule is shown in Fig. 8*e*; the response of neural network is shown in Fig. 8*f*. Comparing with Fig. 7, Fig. 8 shows that the proposed RFSC system can achieve better tracking performance even under disturbance. The number of the constructed fuzzy rules is 5, and the maximum of the constructed fuzzy rules is 7. The system dynamics with the external disturbance can be estimated well by the SFNN identifier. A concise network structure can be obtained by the proposed self-structuring method.

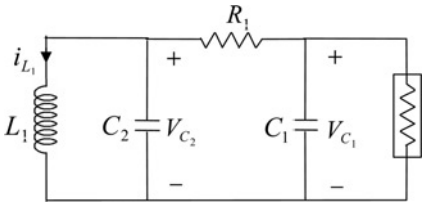

Figure 6 *Chua's chaotic circuit*

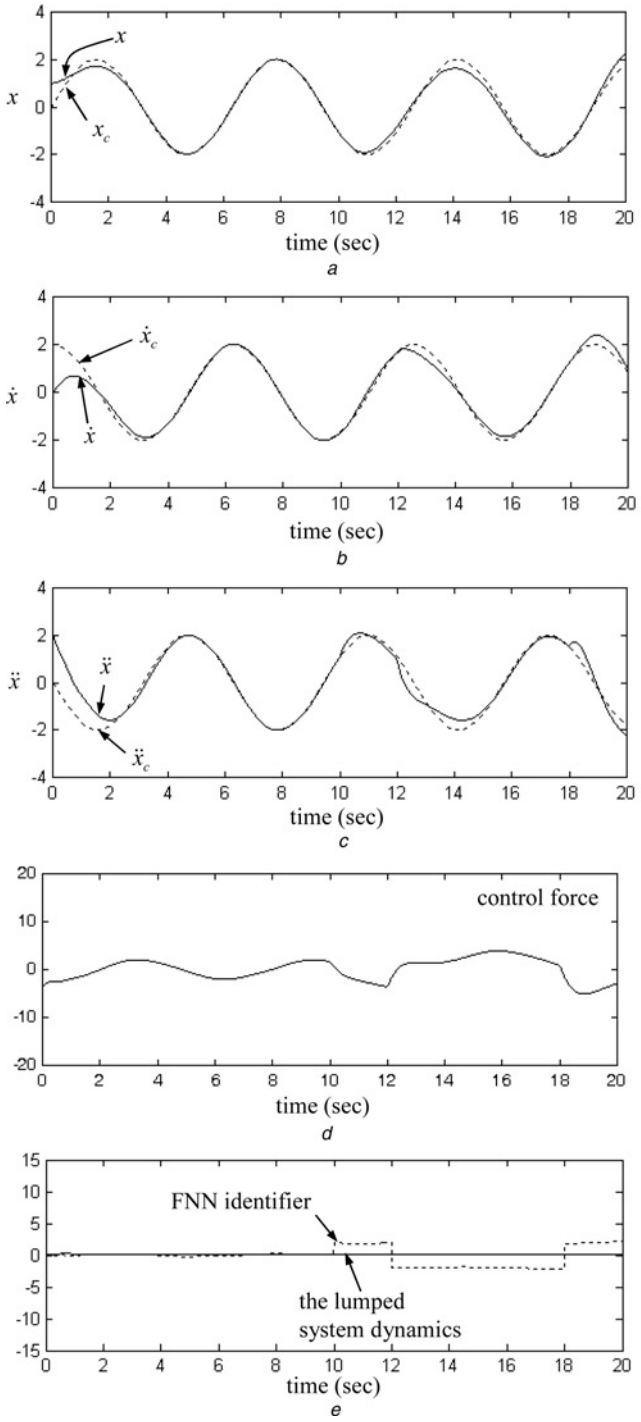

Figure 7 *Simulation results of fix-structuring FNN-based adaptive control system for Chua's chaotic circuit*

doi: 10.1049/iet-cta:20070315

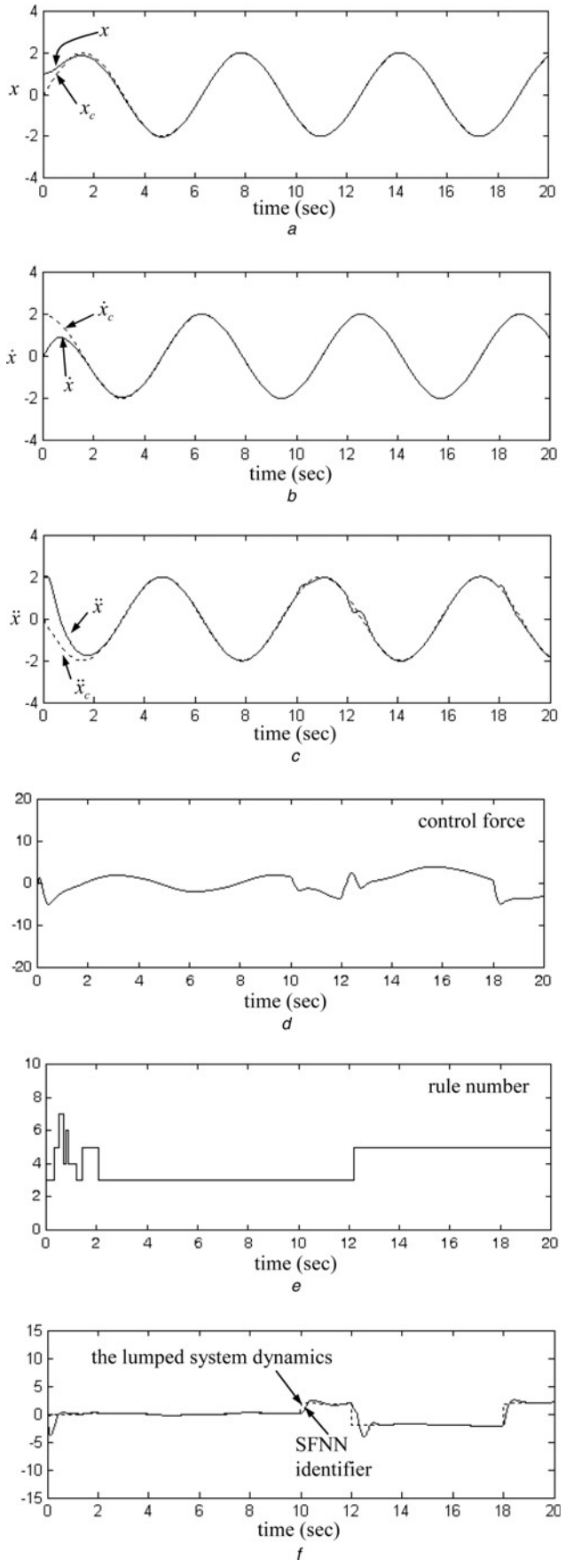

Figure 8 *Simulation results of RFSC system for Chua's chaotic circuit*

Example 3: Consider a plant as follows [26]

$$\dot{x} = f(\mathbf{x}) + u \tag{51}$$

where $f(\mathbf{x}) = (1 - e^{-x})/(1 + e^{-x})$ is the system dynamics, and u is the control force. The parameters of RFSC are selected as $k_1 = 1$, $\eta_w = 40$, $\eta_m = \eta_\sigma = 2$, $G_{\text{th}} = 0.3$, $S_{\text{th}} = 0.5$, $I_{\text{th}} = 0.2$, $\alpha = 0.9$, $P_{\text{th}} = 0.15$, $\tau = 0.01$ and $\delta = 0.5$. The choice of these parameters is also through some trails. The simulation results of RFSC are shown in Fig. 9. The tracking response of state x are shown in Fig. 9*a*; the associated control effort is shown in Fig. 9*b*; the number of fuzzy rules is shown in Fig. 9*c*; the response of neural network is shown in Fig. 9*d*. The number of the constructed fuzzy rules is 5. The results illustrate that the proposed RFSC scheme can achieve satisfied tracking performance for unknown nonlinear systems, and a concise

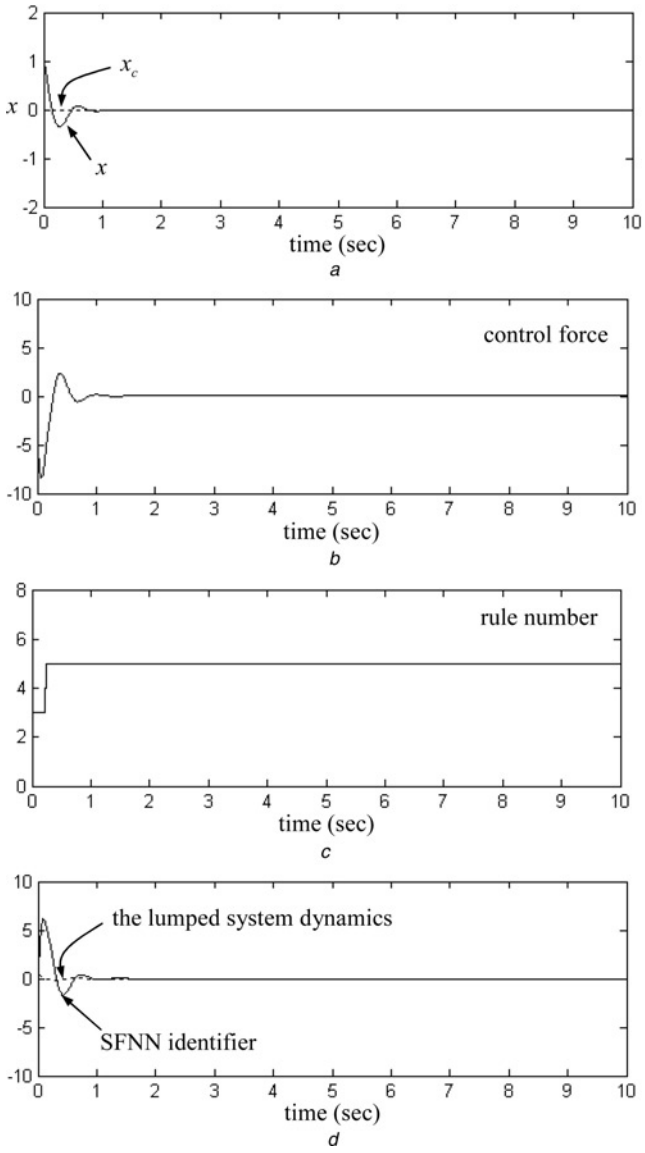

Figure 9 *Simulation results of RFSC system for nonlinear dynamic system*

network structure can be obtained by the proposed self-structuring method.

5 Conclusions

For the FNN-based adaptive control approaches, the structure of FNN should be determined in advance by trial-and-error. It is difficult to consider the balance between the rule number and the desired performance. Therefore this paper proposes a RFSC system with structure adaptation approach. The proposed RFSC system is composed of a computational controller and a robust controller. In the computational controller design, a SFNN with the structure- and parameter-learning is utilised to on-line estimate the unknown control dynamics equation. The robust controller with desired attenuation level is designed to achieve L_2 tracking performance. The structure-learning phase of SFNN possesses several techniques, including the growing of membership functions, the splitting of fuzzy rules and the pruning of fuzzy rules, to organise FNN automatically. The parameter-learning phase of SFNN adjusts the parameters of the membership functions and fuzzy rules based on the Lyapunov function to achieve favourable approximation performance.

The major contributions of this paper are: (1) the SFNN with the growing, splitting and pruning algorithm of the membership functions, and fuzzy rules has been developed to achieve favourable learning performance of network structure. (2) the RFSC, in which the gradient descent method is used to derive the on-line parameter-learning algorithms, has been derived and developed well.

6 Acknowledgments

The authors appreciate the partial financial support from the National Science Council of Republic of China under grant NSC 96-2218-E-216-001. The authors would like to express their gratitude to the reviewers for their valuable comments and suggestions.

7 References

[1] LIN C.T., LEE C.S.G.: 'Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems' (Prentice-Hall, 1996)

[2] SPOONER J.T., MAGGIORE M., ORDONEZ R., PASSINO K.M.: 'Stable adaptive control and estimation for nonlinear systems: neural and fuzzy approximator techniques' (John Wiley and Sons, 2002)

[3] LIN F.J., HWANG W.J., WAI R.J.: 'A supervisory fuzzy neural network control system for tracking periodic inputs', *IEEE Trans. Fuzzy Syst.*, 1999, **7**, (1), pp. 41–52

[4] WANG C.H., LIU H.L., LIN C.T.: 'Direct adaptive fuzzy-neural control with state observer and supervisory controller for unknown nonlinear dynamical systems', *IEEE Trans. Fuzzy Syst.*, 2002, **10**, (1), pp. 39–49

[5] LIN C.M., HSU C.F.: 'Supervisory recurrent fuzzy neural network control of wing rock for slender delta wings', *IEEE Trans. Fuzzy Syst.*, 2004, **12**, (5), pp. 733–742

[6] HSU C.F., LIN C.M.: 'Fuzzy-identification-based adaptive controller design via backstepping approach', *Fuzzy Sets Syst.*, 2005, **151**, (1), pp. 43–57

[7] LEU Y.G., WANG W.Y., LEE T.T.: 'Observer-based direct adaptive fuzzy-neural control for nonaffine nonlinear systems', *IEEE Trans. Neural Netw.*, 2005, **16**, (4), pp. 853–861

[8] HUANG G.B., SARATCHANDRAN P., SUNDARARAJAN N.: 'An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks', *IEEE Trans. Syst. Man Cybern. B, Cybern.*, 2004, **34**, (6), pp. 2284–2292

[9] LI C., LEE C.Y., CHENG K.H.: 'Pseudo-error-based self-organizing neuro-fuzzy system', *IEEE. Trans. Fuzzy Syst.*, 2004, **12**, (6), pp. 812–819

[10] LIN C.T., CHENG W.C., LIANG S.F.: 'An on-line ICA-mixture-model-based self-constructing fuzzy neural network', *IEEE Trans. Circuits Syst. I*, 2005, **52**, (1), pp. 207–221

[11] LIN F.J., LIN C.H., SHEN P.H.: 'Self-constructing fuzzy neural network speed controller for permanent-magnet synchronous motor drive', *IEEE Trans. Fuzzy Syst.*, 2001, **9**, (5), pp. 751–759

[12] WU S., ER M.J., GAO Y.: 'A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks', *IEEE Trans. Fuzzy Syst.*, 2001, **9**, (4), pp. 578–594

[13] GAO Y., ER M.J.: 'Online adaptive fuzzy neural identification and control of a class of MIMO nonlinear systems', *IEEE Trans. Fuzzy Syst.*, 2003, **11**, (4), pp. 462–477

[14] LIN F.J., LIN C.H.: 'A permanent-magnet synchronous motor servo drive using self-constructing fuzzy neural network controller', *IEEE Trans. Energy Convers.*, 2004, **19**, (1), pp. 66–72

[15] PARK J.H., HUH S.H., KIM S.H., SEO S.J., PARK G.T.: 'Direct adaptive controller for nonaffine nonlinear systems using self-structuring neural networks', *IEEE Trans. Neural Netw.*, 2005, **16**, (2), pp. 414–422

[16] HSU C.F.: 'Self-organizing adaptive fuzzy neural control for a class of nonlinear systems', *IEEE Trans. Neural Netw.*, 2007, **18**, (4), pp. 1232–1241

[17] SLOTINE J.J.E., LI W.P.: 'Applied nonlinear control' (Prentice Hall, 1991)

doi: 10.1049/iet-cta:20070315

[18] HSU C.F., LIN C.M., LEE T.T.: 'Wavelet adaptive backstepping control for a class of nonlinear systems', *IEEE Trans. Neural Netw.*, 2006, **17**, (5), pp. 1175–1183

[19] PENG Y.F., LIN C.M.: 'Intelligent motion control of linear ultrasonic motor with H^{∞} tracking performance', *IET Control Theory Appl.*, 2007, **1**, (1), pp. 9–17

[20] KUNG C.C., CHEN T.H.: 'H^{∞} tracking-based adaptive fuzzy sliding mode controller design for nonlinear systems', *IET Control Theory Appl.*, 2007, **1**, (1), pp. 82–89

[21] HERZEL H., BERRY D., TITZE I.R., SALEH M.: 'Analysis of vocal disorders with methods from nonlinear dynamics', *J. Speech Hear. Res.*, 1994, **37**, pp. 1108–1019

[22] LORIA A., PANTELEY E., NIJMEIJER H.: 'Control of the chaotic Duffing equation with uncertainty in all parameter', *IEEE Trans. Circuits Syst. I*, 1998, **45**, (6), pp. 1252–1255

[23] CHANG Y.C.: 'A robust tracking control for chaotic Chua's circuits via fuzzy approach', *IEEE Trans. Circuits Syst. I*, 2001, **48**, (7), pp. 889–895

[24] WANG C.H., LIN T.C., LEE T.T., LIU H.L.: 'Adaptive hybrid intelligent control for uncertain nonlinear dynamical systems', *IEEE Trans. Syst. Man Cybern.*, 2002, **32**, (5), pp. 583–597

[25] LORÍA A., ZAVALA-RÍO A.: 'Adaptive tracking control of chaotic systems with applications to synchronization', *IEEE Trans. Circuits Syst. I*, 2007, **54**, (9), pp. 2019–2029

[26] WANG L.X.: 'Adaptive fuzzy systems and control: design and stability analysis' (Prentice-Hall, 1994)