Proceedings of the 2012 International Conference on
Advanced Mechatronic Systems, Tokyo, Japan, September 18-21, 2012

Adaptive TSK-type Self-evolving Neural Control for Unknown Nonlinear Systems

Yu-Hsiung Lin and Chun-Fei Hsu, *Member, IEEE*

***Abstract*—In this paper, a real-time approximator using a TSK-type self-evolving neural network (TSNN) is studied. The learning algorithm of the proposed TSNN not only automatically online generates and prunes the hidden neurons but also online adjusts the network parameters. Then, an adaptive TSK-type self-evolving neural control (ATSNC) system which is composed of a neural controller and a smooth compensator is proposed. The neural controller uses a TSNN to approximate an ideal controller and the smooth compensator is designed to eliminate the effect of the approximation error introduced by the neural controller upon the system stability in the Lyapunov sense. Finally, the proposed ATSNC system is applied to a chaotic system to illustrate its effectiveness. It shows by the simulation results that a favorable control performance can be achieved by the proposed ATSNC scheme.**

I. Introduction

THE mathematical models of the controlled systems are difficult to develop accurately due to the lack of knowledge of some parameters or the presence of disturbances in the system. To attack this problem, intelligent control approaches such as neural mechanism are dynamic model-free control techniques. There have been considerable interests in exploring the applications of neural network to deal with the non-linearity and uncertainty of control systems [1]-[4]. The most important feature of these adaptive neural control schemes is the self-learning ability that neural networks are used to approximate arbitrary linear or nonlinear mappings through online learning algorithms without requiring preliminary offline tuning.

The neural-network-based adaptive control has represented an alternative design method for the control of unknown or uncertain nonlinear systems [5], [6]. The successful key element of neural-network-based control technique is the approximation ability, where the parameterized neural network can approximate the unknown controlled system dynamics or the ideal controller after learning. However, the learning algorithm only includes the parameter learning phase, and they have not considered the structure learning phase of the neural network.

To attack this problem determining the number of hidden neurons, several dynamic RBF networks which can vary its structure dynamically to keep the prescribed approximation accuracy are proposed [7]-[9]. Huang et al. [7] presents a generalized growing and pruning algorithm. The growing and pruning strategy is based on linking the required learning accuracy with the significance of the nearest. In [8], a structure learning algorithm for recurrent RBF networks is studied. It can delete some unimportant RBF nodes based on the recursive least square approach. In [9], a significance criterion and a growth criterion are applied to design the structure learning algorithm for a RBF network. Hsu proposed a self-constructing recurrent neural network with considering the significant of the hidden neuron [10].

Since the hidden neurons number is finite, the approximation error is inevitable when it is used to approximate an ideal controller or system uncertainties. To ensure the closed-loop control system's stability, a compensator should be designed to dispel the approximation error. The most frequently used compensator is a sliding-mode type control which requires the bound of the approximation error [11]. However, it will result in substantial chattering in the control effort. Wu et al. [12] presented a smooth compensator to guarantee system stable without occurring chattering phenomena. The tracking error can exponentially converge to a small neighborhood of the trajectory command. A H^{∞} tracking control theory is used to attenuate the effects of approximation error in [13], [14]. However, the control effort may lead to a large control signal.

This paper proposes a TSK-type self-evolving neural network (TSNN) which can vary its structure dynamically to keep the prescribed approximation accuracy with a simple computation. Then, an adaptive TSK-type self-evolving neural control (ATSNC) system which is composed of a neural controller and a smooth compensator is proposed. The online parameter learning algorithm of the TSNN, including the connective weights, the center and width of the Gaussian functions are derived using the back-propagation method. Some simulation results are carried out to investigate the effectiveness of the proposed ATSNC scheme.

II. Problem Statement

The model of many practical nonlinear systems can be expressed in the following form as

$$\ddot{x} = f(\mathbf{x}) + gu \tag{1}$$

where $\mathbf{x} = [x, \dot{x}]^T$ is the state vector of the system, $f(\mathbf{x})$ and g are the system dynamics and u is the control input. The

Manuscript received June 10, 2012.

Yu-Hsiung Lin is with the Department of Electrical Engineering, Chung Hua University, Hsinchu 300, Taiwan (e-mail: bear@chu.edu.tw).

Chun-Fei Hsu is with the Department of Electrical Engineering, Tamkang University, New Taipei City 25137, Taiwan (phone: 886-2-26215656 Ext. 2615; fax: 886-2-26209814; e-mail: fei@ee.tku.edu.tw).

control problem is to find a control law so the state x can track a command x_c closely. To determine the control law, define the tracking error as

$$e = x_c - x . \tag{2}$$

If the system dynamics are known, there exists an ideal controller [15]

$$u^* = \frac{1}{g}[-f(\mathbf{x}) + \ddot{x}_c + k_1\dot{e} + k_2 e] \tag{3}$$

where k_1 and k_2 are nonzero positive constants. Applying (3) into (1) yields

$$\ddot{e} + k_1\dot{e} + k_2 e = 0 . \tag{4}$$

If k_1 and k_2 are chosen to correspond to the coefficients of a Hurwitz polynomial, it implies that $\lim_{t\to\infty} e = 0$ [15]. Since the terms $f(\mathbf{x})$ and g in the system dynamics may be unknown or perturbed, the ideal controller cannot be precisely obtained.

Before designing the sliding-mode control, rewriting (1), the nominal model of the nonlinear dynamic system can be represented as follows

$$\ddot{x} = f_n(\mathbf{x}) + g_n u \tag{5}$$

where $f_n(\mathbf{x})$ and g_n represent the nominal behavior of $f(\mathbf{x})$ and g, respectively. If uncertainties occur, i.e., parameters of the system deviate from the nominal value, the controlled system can be modified as

$$\begin{aligned}\ddot{x} &= [f_n(\mathbf{x}) + \Delta f(\mathbf{x})] + [g_n + \Delta g(\mathbf{x})]u \\ &= f_n(\mathbf{x}) + g_n u + w(\mathbf{x})\end{aligned} \tag{6}$$

where $\Delta f(\mathbf{x})$ and Δg denote the system uncertainties and $w(\mathbf{x})$ is called the lumped uncertainty which is defined as $w(\mathbf{x}) = \Delta f(\mathbf{x}) + \Delta g(\mathbf{x})u$. The lumped uncertainty is assumed to be bounded, i.e., $|w(\mathbf{x})| \le W$ where W is a given positive constant. The sliding surface is defined as

$$s = \dot{e} + k_1 e + k_2 \int_0^t e(\tau)d\tau . \tag{7}$$

The sliding-mode control law is given as [15]

$$u_{sm} = u_{eq} + u_{ht} \tag{8}$$

where the equivalent controller u_{eq} is represented as

$$u_{eq} = \frac{1}{g_n}[-f_n(\mathbf{x}) + \ddot{x}_c + k_1\dot{e} + k_2 e] \tag{9}$$

and the hitting controller u_{ht} is designed as

$$u_{ht} = \frac{1}{g_n}[W \operatorname{sgn}(s)] \tag{10}$$

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

$$\ddot{e} + k_1\dot{e} + k_2 e = -w(\mathbf{x}) - W\operatorname{sgn}(s) = \dot{s} . \tag{11}$$

Consider the candidate Lyapunov function in the following form as

$$V_1 = \frac{1}{2}s^2 . \tag{12}$$

Differentiating (12) with respect to time and using (11), we can obtain

$$\begin{aligned}\dot{V}_1 &= s\dot{s} = -w(\mathbf{x})s - W|s| \\ &\le |w(\mathbf{x})||s| - W|s| \\ &= -(W - |w(\mathbf{x})|)|s| \le 0 .\end{aligned} \tag{13}$$

In summary, a sliding-mode controller can guarantee the stability in the sense of the Lyapunov theorem [15]. Since the control law uses a sign function, an undesirable chattering phenomenon is resulted in.

III. ATSNC System Design

The proposed ATSNC system as shown in Fig. 1 is composed of a neural controller and a smooth compensator, i.e.

$$u_{atc} = u_{nc} + u_{sc} \tag{14}$$

where the neural controller u_{nc} uses a TSNN to approximate an ideal controller and the smooth compensator u_{sc} is designed to eliminate the approximation error between the neural controller and ideal controller.

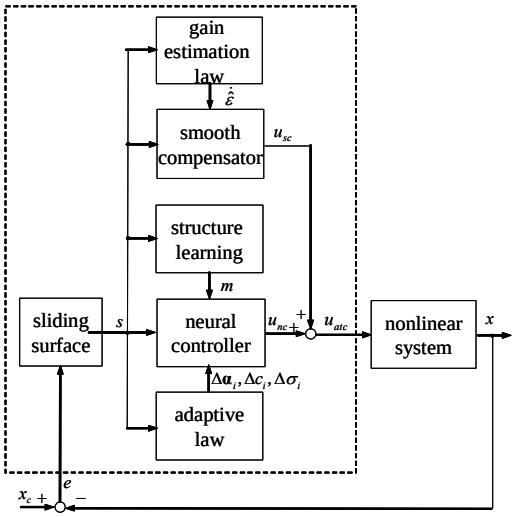


Fig.1. The block diagram of the ATSNC system

A. TSNN

The TSNN as shown in Fig. 2 includes the input layer, hidden layer, TSK layer and output layer. The signal propagation and the basic function in each space are introduced in the following.

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

Layer 2 - Hidden layer: In this layer, the Gaussian function is adopted as the radial basis function. The output of the *i*-th hidden neurons is designed as

$$h_i = \exp[-\frac{(s - c_i)^2}{\sigma_i^2}],\ i = 1,2,...,m \tag{15}$$

where m is the number of the hidden neurons, and c_i and σ_i denote the center and width of the Gaussian function, respectively.

Layer 3 - TSK layer: The TSK layer represents the linear combination function. Each node in this layer is denoted by

$$u_i = \alpha_{i0} + \alpha_{i1} s = \boldsymbol{\alpha}_i^T \boldsymbol{\xi} \quad (16)$$

where α_{i0} and α_{i1} are the parameters designed by the designer, $\boldsymbol{\alpha}_i = [\alpha_{i0}, \alpha_{i1}]^T$ and $\boldsymbol{\xi} = [1, s]^T$.

Layer 4 - Output layer: The single node computes the overall output as the summation of all incoming signals. The output of TSNN can be represented as

$$u_{nc} = \sum_{i=1}^{m} u_i h_i = \sum_{i=1}^{m} \boldsymbol{\alpha}_i^T \boldsymbol{\xi} h_i \,. \quad (17)$$

Determining an appropriate number of the hidden neurons m is an important issue because of the trade-off between the computation load and the learning performance.

To attack the problem of structure determination, this paper proposes an on-line structuring learning algorithm. The activation function h_i in (15) can be represented as the degree to which the incoming data belong to the existing hidden neurons. For an incoming data, find

$$K = \arg \max_{1 \le i \le m(t)} h_i \quad (18)$$

where $m(t)$ is the number of the existing hidden neurons at the time t. It implies that the K-th hidden neuron has the maximum firing weight. If $h_K \le G_{th}$ is satisfied, where $G_{th} \in (0,1)$ a pre-given threshold, then a new hidden neuron is generated. The initial parameters associated with the new hidden neuron are given by

$$c^{new} = s \quad (19)$$

$$\sigma^{new} = \sigma \quad (20)$$

$$\boldsymbol{\alpha}^{new} = \mathbf{0} \quad (21)$$

where σ is a pre-specified constant. To avoid the structure of neural network growing unboundedly, a pruning algorithm based on the density is studied in this paper, where the density is the number of times each hidden neuron is used in the algorithm. A density of the i-th hidden neuron is defined as

$$d_i(t+1) = \begin{cases} d_i(t) & , if\ h_i < \rho \\ d_i(t) + 1 & , if\ h_i \ge \rho \end{cases} \quad (22)$$

where the initial value of d_i is 0 in each time iterations and ρ is a designed constant. After some iterations, if $d_i \le D_{th}$ are satisfied, where D_{th} a pre-specified threshold, then the i-th hidden neuron is deleted.

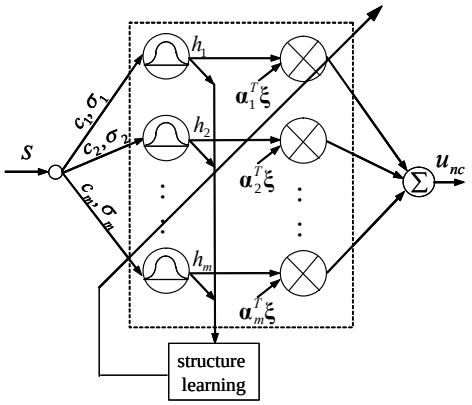


Fig. 2. Network structure of TSNN

B. Parameter learning

Substituting (14) into (1) and using (3), the error dynamic equation can be obtained as

$$\ddot{e} + k_1 \dot{e} + k_2 e = g(u^* - u_{nc} - u_{sc}) = \dot{s} \,. \quad (23)$$

Multiplying both sides of (23) by s gives

$$s\dot{s} = sg(u^* - u_{nc} - u_{sc}) \,. \quad (24)$$

According to the gradient descent method, the correction $\Delta\boldsymbol{\alpha}_i$ applied to the weight $\boldsymbol{\alpha}_i$ is given by [1]

$$\Delta\boldsymbol{\alpha}_i = -\eta_\alpha \frac{\partial s\dot{s}}{\partial \boldsymbol{\alpha}_i} = -\eta_\alpha \frac{\partial s\dot{s}}{\partial u_{nc}} \frac{\partial u_{nc}}{\partial \boldsymbol{\alpha}_i} = \eta_\alpha sgh_i\boldsymbol{\xi} \quad (25)$$

where η_α is the positive learning rate. The weight $\boldsymbol{\alpha}_i$ is updated according to the following equation

$$\boldsymbol{\alpha}_i(t+1) = \boldsymbol{\alpha}_i(t) + \Delta\boldsymbol{\alpha}_i(t) \,. \quad (26)$$

Moreover, the corrections Δc_i and $\Delta\sigma_i$ applied to the center and width, respectively, can be obtained as [1]

$$\Delta c_i = -\eta_c \frac{\partial s\dot{s}}{\partial c_i} = 2\eta_c sg\boldsymbol{\alpha}_i\boldsymbol{\xi} \frac{(s - c_i)}{\sigma_i^2} h_i \quad (27)$$

$$\Delta\sigma_i = -\eta_\sigma \frac{\partial s\dot{s}}{\partial \sigma_i} = 2\eta_\sigma sg\boldsymbol{\alpha}_i\boldsymbol{\xi} \frac{(s - c_i)^2}{\sigma_i^3} h_i \quad (28)$$

where η_c and η_σ are the positive learning rates. The center and width are updated as following

$$c_i(t+1) = c_i(t) + \Delta c_i(t) \quad (29)$$

$$\sigma_i(t+1) = \sigma_i(t) + \Delta\sigma_i(t) \,. \quad (30)$$

To deal with the unavailable system dynamics, g is rewritten as $|g|\mathrm{sgn}(g)$. Therefore, the update laws shown in (25), (27) and (28) can be rewritten as follows

$$\Delta\boldsymbol{\alpha}_i = \beta_\alpha sh_i\boldsymbol{\xi} \quad (31)$$

$$\Delta c_i = 2\beta_c s\boldsymbol{\alpha}_i\boldsymbol{\xi} \frac{(s - c_i)}{\sigma_i^2} h_i \quad (32)$$

$$\Delta\sigma_i = 2\beta_\sigma s\boldsymbol{\alpha}_i\boldsymbol{\xi} \frac{(s - c_i)^2}{\sigma_i^3} h_i \quad (33)$$

where the terms $\eta_\alpha|g|$, $\eta_c|g|$ and $\eta_\sigma|g|$ are absorbed into the new positive learning rates β_α, β_c and β_σ, respectively.

C. Stability analysis

There exists an approximation error between the ideal controller and the neural controller, i.e.

$$u^* - u_{nc} = \varepsilon \quad (34)$$

where ε denotes the approximation error; however, the approximation error cannot be easily measured in practical applications. Substituting (34) into (23) yields

$$\dot{s} = g(\varepsilon - u_{sc}) \,. \quad (35)$$

In this paper, the smooth compensator is designed as

$$u_{sc} = \hat{\varepsilon} + ks \quad (36)$$

where $\hat{\varepsilon}$ denotes the estimated value of the approximation error and k is a small positive constant. Using (36), (35) can be rewritten as

$$\dot{s} = g(\varepsilon - \hat{\varepsilon} - ks) = g(\tilde{\varepsilon} - ks) \quad (37)$$

where $\tilde{\varepsilon} = \varepsilon - \hat{\varepsilon}$. To guarantee the stability of the ADTRNC system, a Lyapunov function candidate is defined as

$$V_2 = \frac{1}{2}s^2 + \frac{g}{2\eta_\varepsilon}\tilde{\varepsilon}^2 \tag{38}$$

where η_ε is the positive learning rate. Differentiating (38) with respect to time and using (37), we can obtain that

$$\dot{V}_2 = s\dot{s} + \frac{g}{\eta_\varepsilon}\tilde{\varepsilon}\dot{\tilde{\varepsilon}} = \tilde{\varepsilon}g(s + \frac{1}{\eta_\varepsilon}\dot{\tilde{\varepsilon}}) - kgs^2 . \tag{39}$$

For achieving $\dot{V}_2 \le 0$, the error estimation law is designed as

$$\dot{\hat{\varepsilon}} = -\dot{\tilde{\varepsilon}} = \eta_\varepsilon s \tag{40}$$

then (39) can be rewritten as

$$\dot{V}_2 = -kgs^2 \le 0 . \tag{41}$$

As a result, the stability of the proposed ATSNC system can be guaranteed.

IV. Simulation Results

Chaotic systems have been studied and known to exhibit a complex dynamical behavior. This study considers a second-order chaotic system such as the Duffing's equation [16]-[18]

$$\ddot{x} = -p\dot{x} - p_1 x - p_2 x^3 + q\cos(\omega t) + u = f(\mathbf{x}) + u \tag{42}$$

where $\mathbf{x} = [x, \dot{x}]^T$ is the state vector, $f(\mathbf{x}) = -p\dot{x} - p_1 x - p_2 x^3 + q\cos(\omega t)$ is the system dynamic, u is the control effort, and p, p_1, p_2, q and ω are real constants. For observing the complex phenomena, the open-loop chaotic system behavior with $u = 0$ was simulated with $p = 0.4$, $p_1 = -1.1$, $p_2 = 1.0$ and $\omega = 1.8$. With an initial condition $(x, \dot{x}) = (0,0)$, the phase plane plots for $q = 2.1$ and $q = 7.0$ are shown in Figs. 3(a) and 3(b), respectively. It is shown that the uncontrolled chaotic system has different chaotic trajectories with different system parameters.

To show the advantage of the proposed TSNN, a dynamic RBF neural network (DRNN) is used to replace a TSNN. The proposed ATSNC system with DRNN is applied to the chaotic system. The simulation results of the ATSNC system with DRNN are shown in Figs. 4 and 5 for $q = 2.1$ and $q = 7.0$, respectively. The simulation results verify that the favorable tracking performance can be achieved but the convergence is slow.

Then, the proposed ATSNC system with TSNN is applied to the chaotic system again. The simulation results of the ATSNC system with TSNN are shown in Figs. 6 and 7 for $q = 2.1$ and $q = 7.0$, respectively. The simulation results verify that the favorable tracking performance and no chattering phenomena can be achieved by the proposed ATSNC system. Moreover, the network structure learning approach demonstrates the properties of generating or pruning the hidden neurons of the TSNN automatically.

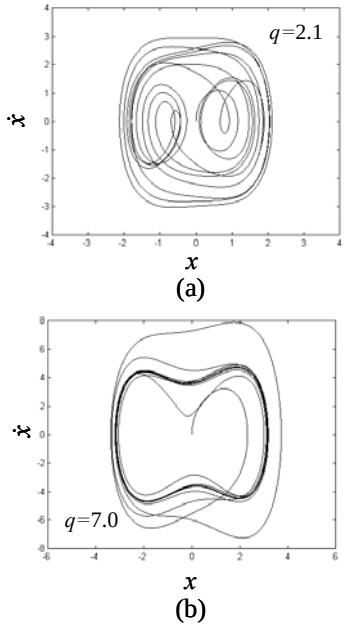


Fig. 3. The uncontrolled chaotic system

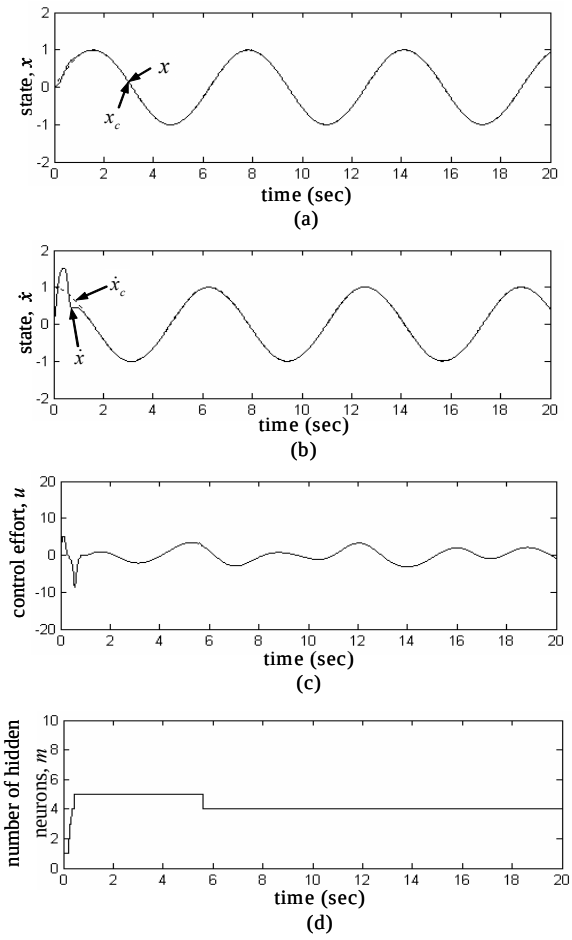


Fig. 4. Simulation results of the ATSNC system with DRNN for $q = 2.1$.

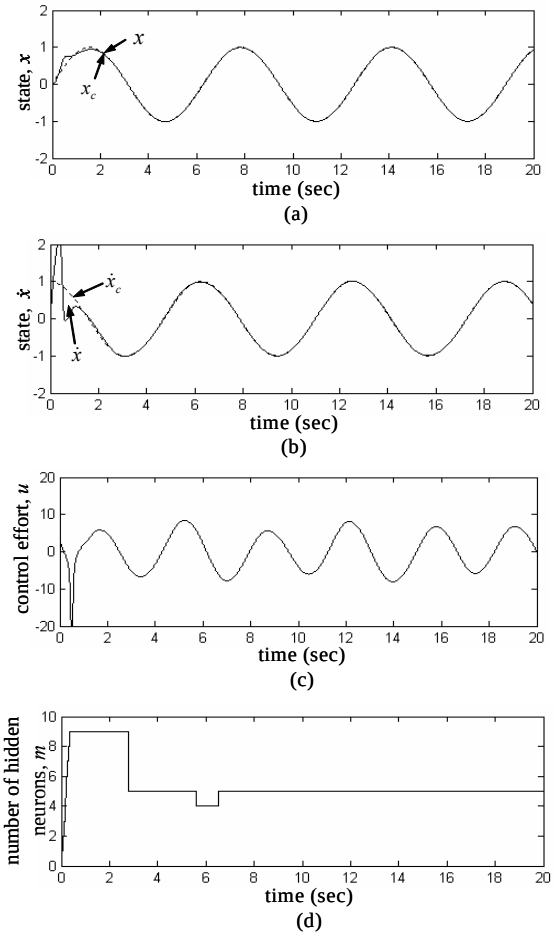
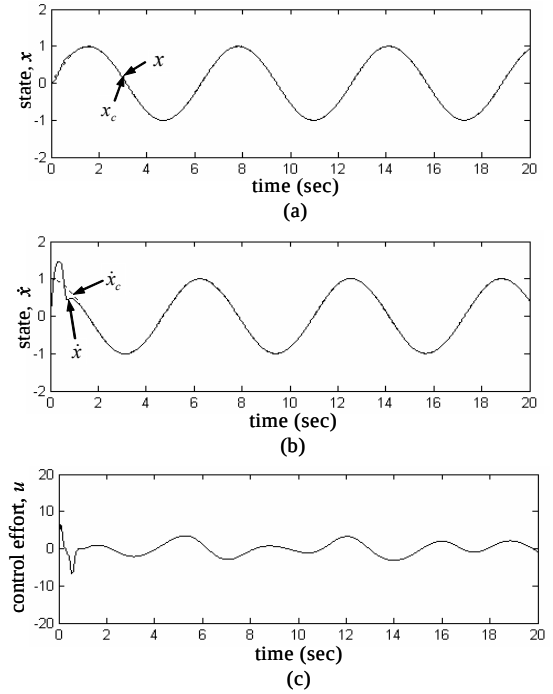


Fig. 5. Simulation results of the ATSNC system with DRNN for $q = 7.0$.



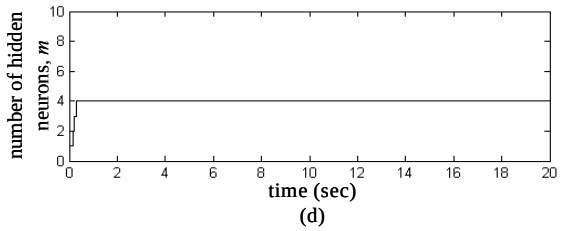


Fig. 6. Simulation results of the ATSNC system with TSNN for $q = 2.1$.

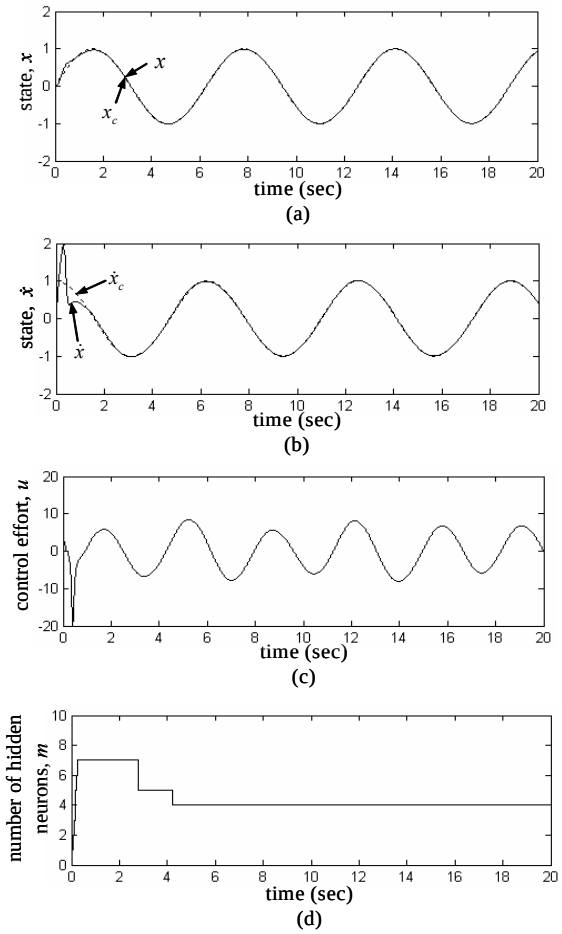


Fig. 7. Simulation results of the ATSNC system with TSNN for $q = 7.0$.

V. CONCLUSIONS

In this paper, an adaptive TSK-type self-evolving neural control (ATSNC) system has been successfully developed via a TSK-type self-evolving neural network (TSNN). Some simulation results verify that a favorable control performance can be achieved by the proposed ATSNC scheme after learning of the controller parameters. The comparison of control performance between with DRNN and with TSNN is summarized in Table1. It is seen that the proposed ATSNC system with TSNN can achieve better performance than its with DRNN. The major contributions of this study are: (i) the successful development of an online-trained TSNN which combines the merits of TSK-type neural network and RBF neural network; (ii) the successful application of the chaotic

control system to track a command trajectory with robust control performance.

Table 1. Performance comparison

Performance	ATSNC system with DRNN		ATSNC system with TSNN	
	$q = 2.1$	$q = 7.0$	$q = 2.1$	$q = 7.0$
maximum of tracking error	0.0945	0.2452	0.056	0.1405
mean of tracking error	0.0058	0.0112	0.0054	0.0092
standard deviation of tracking error	0.0130	0.0277	0.0114	0.0156

ACKNOWLEDGMENT

The authors appreciate the partial financial support from the National Science Council of Republic of China under the grant NSC 100-2628-E-032-003.

REFERENCES

[1] C. M. Lin, and C. F. Hsu, "Supervisory recurrent fuzzy neural network control of wing rock for slender delta wings," *IEEE Trans. Fuzzy Syst.*, vol. 12, pp. 733–742, Oct. 2004.

[2] Y. G. Leu, W. Y. Wang, and T. T. Lee, "Observer-based direct adaptive fuzzy-neural control for nonaffine nonlinear systems," *IEEE Trans. Neural Netw.*, vol. 16, pp. 853–861, Jul 2005.

[3] C. F. Hsu, K. H. Cheng, and T. T. Lee, "Robust wavelet-based adaptive neural controller design with a fuzzy compensator," *Neurocomputing*, vol. 73, pp. 423–431, Dec. 2009.

[4] C. H. Chiu, "The design and implementation of a wheeled inverted pendulum using an adaptive output recurrent cerebellar model articulation controller," *IEEE Trans Ind. Electron.*, vol. 57, pp. 1814–1822, May 2010.

[5] S. Wang, and D. L. Yu, "Adaptive RBF network for parameter estimation and stable air-fuel ratio control," *Neural Netw.*, vol. 21, pp. 102–112, Jan 2008.

[6] T. Zhao, "RBFN-based decentralized adaptive control of a class of large-scale non-affine nonlinear systems," *Neural Comput. Appl.*, vol. 17, pp. 357–364, Aug 2008.

[7] G. B. Huang, P. Saratchandran, and N. Sundararajan, "A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation," *IEEE Trans. Neural Netw.*, vol. 16, pp. 57-67, Jan 2005.

[8] C. S. Leung, and A.C. Tsoi, "Combined learning and pruning for recurrent radial basis function networks based on recursive least square algorithms," *Neural Comput. Appl.*, vol. 15, pp. 62–78, March 2006.

[9] M. Bortman, and M. Aladjem, "A growing and pruning method for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 20, pp. 1039–1045, June 2009.

[10] C. F. Hsu, "Intelligent position tracking control for LCM drive using stable online self-constructing recurrent neural network controller with bound architecture," *Control Engineering Practice*, vol. 17, pp. 714–722, June 2009.

[11] C. M. Lin, and C. F. Hsu, "Neural network hybrid control for antilock braking systems," *IEEE Trans. Neural Netw.*, vol. 14, pp. 351–359, March 2003.

[12] T. F. Wu, P. S. Tsai, F. R. Chang, and L. S. Wang, "Adaptive fuzzy CMAC control for a class of nonlinear system with smooth compensation," *IEE Proc. Control Theory Appl.*, vol. 153, pp. 647–657, Nov. 2006.

[13] W. Y. Wang, M. L. Chan, C. C. J. Hsu, and T. T. Lee, "H^{∞} tracking-based sliding mode control for uncertain nonlinear systems via an adaptive fuzzy-neural approach," *IEEE Trans Syst. Man Cybern. B Cybern.*, vol. 32, pp. 483–492, Aug 2002.

[14] Y. F. Peng, "Robust intelligent backstepping tracking control for uncertain nonlinear chaotic systems using H^{∞} control technique," *Chaos Solitons Fractals*, vol. 41, pp. 2081–2096, Aug 2009.

[15] J. J. E. Slotine, and W. P. Li, Applied nonlinear control, Prentice-Hall, Englewood Cliffs, 1991.

[16] G. Chen, and X. Dong, "On feedback control of chaotic continuous time systems," *IEEE Trans. Circuits Syst. I*, vol. 40, pp. 591–601, Sept 1993.

[17] C. F. Hsu, C. M. Chung, C. M. Lin, and C. Y. Hsu, "Adaptive CMAC neural control of chaotic systems with a PI-type learning algorithm," *Expert Systems with Appl.*, vol. 36, pp. 11836–11843, Nov. 2009.

[18] C. M. Lin, M. H. Lin, and Y. F. Peng, "Synchronization of chaotic gyro systems using recurrent wavelet CMAC," *Cybern. Syst.*, vol. 41, pp. 391–405, Oct. 2010.