USING CHAOS TO IMPROVE GENERALIZATION IN SMART NN CONTROL DESIGN [1]

Cong Wang [*,2] **Guanrong Chen** [*,3] **Shuzhi S. Ge** [**,4]

[*] *Department of Electronic Engineering*
City University of Hong Kong, Hong Kong SAR, P. R. China
[**] *Department of Electrical & Computer Engineering*
National University of Singapore, Singapore 117576

Abstract: In this paper, a smart NN control scheme is proposed. This scheme is designed such that the current control action can utilize the knowledge that the NN learned from the past control process. A chaotic signal is employed as the reference signal to improve the generalization ability of the NN in the training phase of the scheme, where the complex chaotic signal offers much more information for NN learning thereby significantly improving the efficiency of the NN generalization. Compared with most of the adaptive neural controllers, the smart neural controller (in the operational phase) is a static and low-order controller, and thus needs much less computational resources, and is more feasible in practical implementation. Simulation studies are included to demonstrate the effectiveness of the new control scheme.

Keywords: Neural network (NN), smart NN control, generalization, chaos

1. INTRODUCTION

Over the past decade, adaptive neural control of uncertain nonlinear systems has attracted considerable interest, and significant progress in this area has been achieved in both theoretical studies and practical applications (see, e.g., Lewis *et al.,* 1999; Ge *et al.,* 2001; and the references therein). In the adaptive neural control approaches, neural networks are mostly used as function approximators. One advantage of the adaptive neural control design is that both stability and control performance of the closed-loop can be guaranteed by suitably choosing the design parameters. However, it is noticed that in most existing adaptive neural control approaches, the neural networks used have to learn the uncertainties throughout each control process over and over again. In other words, the controller does not utilize the knowledge that the neural network has learned previously in the earlier control processes. All the weights of the NN have to be updated in each control process, where most elements of the control process, such as the plant, the reference signal, the initial conditions, and the control parameters are kept unchanged. Moreover, with a large number of neurons being updated simultaneously, adaptive NN controllers are in general very high-order dynamical controllers, and thus they are complicated and expensive for implementation in practice. Finally, the generalization ability of neural networks is not considered in such adaptive neural design. From a neural network learning point of view, oftentimes the purpose of using a neural network is to generalize, i.e., to process inputs not necessarily in the training set from which the NN can provide meaningful outputs. With all these issues considered, the adaptive neural controllers proposed thus far are believed to be "not smart," at least not as "smart" as desired, at least for the reason that many basic properties of NNs have not been explored and utilized. Therefore, "smarter" NN controllers are expected.

In this paper, we propose a smart neural control scheme that can utilize the knowledge that the NNs

[1] Supported by the Hong Kong RGC under the CERG Grant No. 9040579
[2] E-mail: cwang@ee.cityu.edu.hk
[3] Corresponding author: gchen@ee.cityu.edu.hk
[4] E-mail: elegesz@nus.edu.sg

learned from past control processes. The scheme is composed of two phases: the training phase and the operational phase, or what is called the generalization phase. In the training phase, the uncertain nonlinear system is controlled by an adaptive neural controller to track a desired reference signal. The employment of the adaptive neural controller guarantees the stability and control performance of the closed-loop system. For the facility of utilizing the knowledge learned in this training phase, RBF NNs, instead of MNNs, are used as function approximators due to its spatially localized learning capability. The weights of the NNs are tuned on-line to learn the unknown nonlinearities in the desired controller. In the operational phase, the same uncertain system is controlled by the smart neural controller, where the weights of the well-trained RBF NNs are fixed to some values obtained from the training phase. The tuning of the weights are completely turned off. Compared with the adaptive neural controller in the training phase, the neural controller in the operational phase becomes "smarter", since it exploits the knowledge the NNs learned from previous control process, and it is in character a static, low-order controller, which means that the implementation of such controller is much simplified. It will be shown that under the condition that the RBF NNs are well-trained in the training phase such that some kind of generalization ability of RBF NNs has been obtained (to be detailed in Section 2), the smart neural controller in the operational phase can accomplish the same tracking task, i.e., the output of the controlled system converges to a small neighborhood of the desired reference signal.

Since the tuning of the NN weights are turned off in the operational phase, the NNs may not be able to approximate the unknown nonlinearities correctly when the inputs to the NNs take different values as compared with those in the training phase. In this case, the possibly invalid of NN approximation may cause some extra problems, such as the demotion of control performance, and even more, the loss of the closed-loop stability. Therefore, for the applicability of the smart neural controller in the operational phase, it is essential for the RBF NNs to obtain certain level of generalization ability from the adaptive neural control process. It is well known that necessary conditions for good generalization include (Haykin, 1999): (i) the unknown function to be approximated is smooth enough; (ii) the training set is a sufficiently large and representative subset in the parameter space. To improve the generalization for smooth functions in case of a small training set, injecting artificial noise (jitter) into the inputs during training was proposed in (Holmstrom and Koistinen, 1992). Training with jitter works because, with similar inputs, the desired outputs will all be similar due to the smoothness of the functions to be approximated. However, it is usually difficult to determine the amount of jitter, since too much of it will obviously produce garbage, while too little of it will not have much effect (Holmstrom and Koistinen, 1992). Moreover, injecting artificial noise might damage the stability of the closed-loop system even if the noise is in an exponentially decaying disturbance form (Krstic et al. 1995). Thus, without taking particular action to cope with the artificial noise, using jitter to improve NN generalization is not an effective method in the training phase of the smart neural control scheme.

In the literature of adaptive neural control, periodic signals are commonly used as reference signals, and the convergence of the controlled system's output to a small neighborhood of a periodic reference signal is usually guaranteed. From the perspective of neural network training, the control task of tracking to a periodic reference signal will lead to a limited training set for the neural networks in the adaptive neural controller. The insufficiency of the training set cannot make the RBF NNs well trained to have "good generalization" ability.

To provide a larger training set for better generalization of RBF NNs, in this paper, we employ a chaotic reference signal in the training phase, on the basis of the well-known ergodicity of chaos. The ergodicity of chaos (see, e.g. Chen and Dong, 1998) implies that a chaotic trajectory has a dense set of periodic orbits of different periods, and it passes arbitrarily closely by any spatial point within the bounded chaotic attractor of the system. Utilizing this ergodicity of chaos in the training phase, all the inputs to the NNs in the adaptive neural controller become non-periodic. Thus, compared with training with periodic reference signals, the seemingly simple employment of chaotic reference signals can provide a much larger training set for better generalization of RBF NNs. Moreover, the stability and control performance of the closed-loop system can be more easily guaranteed than training with jitter. In the operational phase to follow, the well-trained neurocontroller will be able to track any of the periodic or non-periodic trajectories nearby the chaotic attractor, with good generalization ability.

To summarize, by resolving some problems in the current adaptive neural control research, the smart neural control scheme can learn from the past experience, and finish the same control task in a "smarter" way. In this sense, we make feasible a class of neural control methods that can act in a way similar to the control process of human in learning to accomplish some complicated control tasks. We also reveal that chaos can be quite beneficial to engineering design and applications if appropriately utilized (Chen and Dong, 1998). Simulation studies are conducted to verify the effectiveness of the scheme. The idea of the smart neural control design can be extended to many classes of uncertain nonlinear systems, and is expected to be applicable to a wide variety of industrial applications.

2. SMART NN CONTROL DESIGN

In this section, we present a "smart" neural control scheme. Since RBF NNs (instead of MNNs) are used as approximation models in the scheme, the reason for choosing RBF networks is firstly explained in the following subsection.

2.1 *RBF NNs*

The RBF NNs belong to the class of local approximators where each basis function can only locally affect the network output. In other words, RBF NNs store information locally in a transparent fashion. The adaptation in one part of the input space does not affect knowledge stored in a different area, i.e., they have spatially localized learning capability. As will be shown later, it is the local property of RBF NNs that makes it simple to utilize the "experience" gained from the past control processes.

In the following, we first give a brief introduction to the RBF NNs. For a continuous function $f(Z): R^q \to R$, it has been shown (see, e.g., Haykin, 1999) that a RBF NN $W^T S(Z)$ can be used to approximate $f(Z)$ over a compact set $\Omega_Z \subset R^q$ to arbitrary any accuracy, i.e.,

$$f(Z) = W^{*T} S(Z) + \epsilon, \ \forall Z \in \Omega_Z \tag{1}$$

where the input vector $Z \in \Omega \subset R^q$, the weight vector $W = [w_1, w_2, \cdots, w_l]^T \in R^l$, W^* represents the ideal constant weights, and ϵ is the approximation error; $S(Z) = [s_1(Z), \cdots, s_l(Z)]^T$, with $s_i(Z)$ being the following Gaussian functions:

$$s_i(Z) = \exp\left[\frac{-(Z-\mu_i)^T(Z-\mu_i)}{\eta_i^2}\right], \quad i = 1, 2, ..., l$$

where $\mu_i = [\mu_{i1}, \mu_{i2}, \cdots, \mu_{iq}]^T$ is the center of the receptive field and η_i is the width of the Gaussian function.

2.2 *Training Phase*

To illustrate the basic ideas of smart neural control scheme, we begin with a simple second-order nonlinear system in the following Brunovsky form:

$$\begin{cases} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x_1, x_2) + g(x_1, x_2)u \\ y &= x_1 \end{cases} \tag{2}$$

where $\bar{x} = [x_1, x_2]^T \in R^2$, $u \in R$, $y \in R$ are the state variables, system input and output, respectively and $f(x_1, x_2)$ and $g(x_1, x_2)$ are both unknown smooth nonlinear functions.

The control objective is to design an adaptive neural controller for system (2) such that (i) all the signals in the closed-loop remain semi-globally uniformly ultimately bounded, and (ii) the output y follows a desired trajectory y_d generated from the following reference model:

$$\begin{aligned} \dot{x}_{di} &= f_{di}(x_d) \\ y_d &= x_{d1} \end{aligned} \tag{3}$$

where $x_d = [x_{d1}, x_{d2}, \cdots, x_{dm}]^T \in R^m$ are the states, $y_d \in R$ is the system output, $f_{di}(\cdot), i = 1, 2, \cdots, m$ are known smooth nonlinear functions. Assume that the states of the reference model remain uniformly bounded, i.e., $x_d \in \Omega_d$, $\forall t \geq 0$.

Assumption 1. The sign of $g(x_1, x_2)$ is known, and there exist constants $g_1 \geq g_0 > 0$ such that $g_1 \geq |g(x_1, x_2)| \geq g_0 \ \forall \bar{x} \in \Omega \subset R^2$.

Using an RBF NN, the adaptive neural controller can be obtained from (Ge *et al.*, 2001)

$$u = -z_1 - c_2 z_2 - \hat{W}^T S(Z) \tag{4}$$

where $z_1 = x_1 - y_d$, $z_2 = x_2 - \alpha_1$, $Z = [x_1, x_2, \dot{\alpha_1}]^T$ with $\alpha_1 = -c_1 z_1 + \dot{x_{d1}}$, $c_1, c_2 > 0$ are design parameters, and $\dot{\alpha}_1 = \frac{\partial \alpha_1}{\partial x_1}\dot{x}_1 + \left(\frac{\partial \alpha_1}{\partial x_d}\right)^T \dot{x}_d = -c_1 x_2 + \sum_{i=1}^m \frac{\partial \alpha_1}{\partial x_{di}} f_{di}$ is an intermediate variable which is computable. The RBF NN $\widehat{W}^T S(Z)$ is used to approximate the unknown function $h(Z) = \frac{1}{g(x_1,x_2)}(f(x_1, x_2) - \dot{\alpha_1})$, where $\widehat{W}$ is the estimate of W^*, and is updated via

$$\dot{\widehat{W}} = \dot{\widetilde{W}} = \Gamma[S(Z)z_2 - \sigma \widehat{W}] \tag{5}$$

where $\widetilde{W} = \widehat{W} - W^*$, $\sigma > 0$ is a small constant, and $\Gamma = \Gamma^T > 0$.

Lemma 1. Consider the closed-loop system consisting of the plant (2), the reference model (3), the controller (4), and the NN weight updating law (5). Assume there exists a sufficiently large compact set $\Omega_Z \in R^3$ such that $Z \in \Omega_Z$ for all $t \geq 0$. Then, for bounded initial conditions in Ω_Z, all signals in the closed-loop system remain bounded, and the output tracking error $y(t) - y_d(t)$ converges to a small neighborhood around zero by appropriately choosing design parameters.

Proof: See (Ge *et al.* 2001). $\diamondsuit$

Apart from the convergence of $y(t)$ to a small neighborhood of $y_d(t)$, it can also be proven that $\widehat{W} - W^*$ converges to a neighborhood of zero. However, due to the exitance of the NN approximation errors, it is clear that the exact convergence of $\widehat{W}$ to W^* will not be achieved. Due to the local property of the RBF NN, specifically, the property that $|s_i(Z)|$ become very small for sufficiently large values of radius $|Z - \mu_i|$, the convergence of $\widehat{W}$ to a neighborhood of W^* implies that

$$h(Z) = W^{*T}S(Z) + \epsilon = \widehat{W}^T S(Z) - \widetilde{W}^T S(Z) + \epsilon$$
$$= \widehat{W}^T S(Z) + \epsilon_1, \ \forall Z \in \Omega_{Z^1} \quad (6)$$

where $\epsilon_1 = \epsilon - \widetilde{W}^T S(Z)$ is the practical approximation error in the training phase, with possibly $|\epsilon_1| > |\epsilon|$, Ω_{Z^1} denotes the region in which the NN input Z visits during the training phase, with $\Omega_{Z^1} \subseteq \Omega_Z \subset R^3$.

Remark 1. From the adaptation law (5), it can be seen that for the neurons whose centers are far away from the trajectories of inputs Z, $|S(Z)|$ will become small due to the local property of the RBF NN. In this case, the weights of these neurons will converge to zero because of the term $-\sigma\widehat{W}$ in Eq. (5). This means that only the neurons whose centers are close to the trajectories of input Z will be activated and updated in the training phase.

2.3 *Operational Phase*

The control task in this phase is to achieve tracking of output y to the same or similar reference signal y_d (still generated from the reference model (3)), under the restriction that the tuning of the neural weights is turned off. To fulfill this task, we still exploit the local property of RBF NNs. The weight of each neuron in the NNs is chosen as the value which is related to the maximum value of the neural weight in the training phase. All of the neural weights are then fixed to the chosen values and do not need to be updated again.

The proposed smart neural controller is chosen as

$$u = -z_1 - c_2 z_2 - \overline{W}^T S(Z) \quad (7)$$

where $z_1 = x_1 - y_d$, $z_2 = x_2 - \alpha_1$, $Z = [x_1, x_2, \dot{\alpha}_1]^T$, $\alpha_1 = -c_1 z_1 + \dot{x}_{d1}$, and the NN weights $\overline{W} = [\bar{w}_1, \cdots, \bar{w}_l]^T \in R^l$ are chosen as

$$\bar{w}_i = \lambda \ \hat{w}_i(t^i_{max}), \ i = 1, \cdots, l \quad (8)$$

where $t^i_{max} = \{t \in [0, \infty) | \max_{t\in[T_0,T_1]} |\hat{w}_i(t)|\}$, $0 < \lambda < 1$ is a constant. $\widehat{W} = [\hat{w}_1, \cdots, \hat{w}_l]^T \in R^l$ are obtained from the training phase, $[T_0, T_1]$ with $T_1 > T_0 > 0$ represents a piece of time segment within the training phase after the transient process.

Proof: The selection of $\overline{W} = [\bar{w}_1, \cdots, \bar{w}_l]^T$ in (8) is based on the localized feature of RBF networks. With the spatially localized basis function $S(Z)$, the adaptation law (5) makes the response of each neuron in the RBF NN only sensitive to local regions of input space. Thus, it is intuitively understandable that in the training phase, the ith neuron's weight $\hat{w}_i(t)$ reaches its maximum value only when the input trajectory passes the region which is closest to the receptive field center of this neuron. By choosing $\overline{W} = [\bar{w}_1, \cdots, \bar{w}_l]^T$ according to (8), the unknown nonlinearity $h(Z)$ can be approximated by $\overline{W}^T S(Z)$ to an accuracy as good as using $\widehat{W}^T S(Z)$, i.e.,

$$h(Z) = W^{*T}S(Z) + \epsilon = \widehat{W}^T S(Z) + \epsilon_1$$
$$= \overline{W}^T S(Z) + \epsilon_2, \ \forall Z \in \Omega_{Z^1} \quad (9)$$

where $\epsilon_2 > 0$ is the approximation error in the operational phase, and $|\epsilon_1| - |\epsilon_2|$ is a small value. Moreover, due to the generalization ability of NNs, it can be reasonably expected that the smooth function $h(Z)$ can still be approximated in the region nearby Ω_{Z^1}, i.e.,

$$h(Z) = \overline{W}^T S(Z) + \epsilon_3, \ \forall Z \in \Omega_{Z^2} \quad (10)$$

where Ω_{Z^2} denotes the region which is close to Ω_{Z^1}, and $\epsilon_3 > 0$ is the approximation error in this region with $|\epsilon_2| - |\epsilon_3|$ being a small value. Note that it is not necessary to achieve accurate approximation using $\overline{W}^T S(Z)$ in both Ω_{Z^1} and Ω_{Z^2}, since what we seek is to exploit $\widehat{W}(t)$ effectively such that the same tracking task can be accomplished.

Since the smooth unknown function $h(Z)$ can be approximated by $\overline{W}^T S(Z)$ to the accuracy ϵ_2 (in (9)) and ϵ_3 (in (10)) which are close to ϵ_1, following the similar procedure in the proof of Lemma 1, the tracking error $y - y_d$ can be proven to converge to a neighborhood of zero, which is as small as using $\widehat{W}^T S(Z)$ in the training phase. $\diamondsuit$

Remark 2. The above result can be extended to many other classes of uncertain nonlinear systems such as those discussed in (Lewis *et al.*, 1999; Ge *et al.,* 2001).

2.4 *Simulation Studies*

To verify and test the smart NN control scheme, the following van der Pol oscillator system (see, e.g. Nicolis and Prigogine, 1989) is taken as the plant for control:

$$\begin{aligned} \dot{x}_1 &= x_2 \quad (11) \\ \dot{x}_2 &= -x_1 + \beta(1 - x_1^2)x_2 + (1.5 + 0.3\cos(x_1))u \\ y &= x_1 \end{aligned}$$

where $\beta > 0$ is a system parameter ($\beta = 0.7$ in this paper), the smooth functions $f(x_1, x_2) = -x_1 + \beta(1 - x_1^2)x_2$ and $g(x_1, x_2) = 1.5 + 0.3\cos(x_1)$ are assumed to be unknown to the controller u. The desired trajectory y_d is generated from the following Brusselator model (Nicolis and Prigogine, 1989):

$$\begin{aligned} \dot{x}_{d1} &= A - (B+1)x_{d1} + x_{d1}^2 x_{d2} \\ \dot{x}_{d2} &= Bx_{d1} - x_{d1}^2 x_{d2} \quad (12) \\ y_d &= x_{d1} \end{aligned}$$

where x_{d1} and x_{d2} are system states, $A, B > 0$ are system parameters. As shown in (Nicolis and

Prigogine, 1989), the phase-plane trajectories of the Brusselator model approach a limit cycle when $B > A^2 + 1$ (the solid line in Fig. 1, with $A = 0.4$, $B = 1.2$). The phase-plane trajectories becomes a chaotic attractor when A is modulated by the harmonic law: $A = A_0 + a\cos(\omega t)$ (the dashdotted line in Fig. 1, with $A_0 = 0.4$, $B = 1.2$, $a = 0.05$ and $\omega = 0.81$).

Firstly, the periodic signal is employed as the reference signal for training the RBF NN during the training phase. The adaptive neural controller (4) is used to control the uncertain system (12). The weights of the NN are updated online according to Eq. (5), so as to learn the unknown nonlinearity $h(Z) = (\frac{1}{g(x_1,x_2)}(f(x_1,x_2) - \dot{\alpha_1})$ in the desired control.

In the following operational phase, system (12) is controlled by the designed smart neural controller (7), where the tuning of the weights are set off and the NN weights are chosen according to Eq. (8), with $\lambda = 0.75$.

In the simulation, the RBF NN, $\widehat{W}^T S(Z)$, contains 125 nodes (i.e., $l = 125$), with centers μ_i $(i = 1, \cdots, l)$ evenly spaced on $[0, 1.2] \times [-0.6, 0.6] \times [-0.6, 0.6]$, and with widths $\eta_i = 0.3$ $(i = 1, \cdots, l)$. The design parameters of the above controller are $c_1 = 0.7$, $c_2 = 0.7$, $\Gamma = diag\{3.0\}$, $\sigma = 0.2$. The initial weights $\widehat{W}(0) = 0.0$, the initial conditions $[x_1(0), x_2(0)]^T = [0.5, 0.2]^T$ and $[x_{d1}(0), x_{d2}(0)]^T = [0.2, 0.3]^T$.

From Figs. 2-3, we can see that the output of the system converges to a small neighborhood of the target periodic signal. The tracking performance shown in Fig. 3 becomes worse as compared with that shown in Fig. 2. This is due to the poor generalization ability of the NN when trained with a periodic signal, and this will be improved by using a chaotic reference signal in the smart neural control scheme, as to be further discussed in next section.

3. CHAOS IMPROVED NN GENERALIZATION

For the applicability of the smart neural controller in the operational phase, it is essential to make both Ω_{Z^1} and Ω_{Z^2} as large as possible, so that the NN approximation of $h(Z)$ is still valid. For this purpose, in this section, we employ a chaotic reference signal in the training phase based on the ergodicity of chaos. In the training phase, when tracking to the chaotic reference signal is achieved, all the inputs to the NN of the adaptive controller become non-periodic. The tracking to a chaotic reference signal will make much more neurons in the RBF networks being activated and updated, and thus can provide a much larger training set than using a periodic reference signal. The stability of the closed-loop system is not damaged compared with using scattering jitter. With this learning, in the operational phase to follow, the smart neural controller should become more capable of tracking any given periodic or non-periodic trajectory nearby the chaotic attractor, thanks to the good generalization ability of the well-trained RBF NNs in the controller.

To verify the effectiveness of this method, the chaotic signal generated from the Brusselator is employed in the training phase in this simulation. Figure 4 shows the simulation results when the controller (4) is applied to system (12) for tracking the chaotic signal y_d (shown in Fig. 1). In the operational phase, system (12) is controlled to track the periodic signal again by using the smart neural controller (7). Recall from Fig. 3 that this periodic signal was not tracked satisfactorily by the same controller trained by a periodic signal. In comparison, we can see, from the result shown in Fig. 5, that fairly good tracking performance is obtained by the controller after training by the chaotic signal. A detailed comparison of tracking errors of the controller by using these two different training signals is shown in Fig. 6.

4. CONCLUSIONS

In this paper, a smart NN control scheme has been designed. The smart NN controller needs much less computational resources to complete the same task as compared with adaptive neural controllers. The generalization property of NNs is thus effectively exploited and improved by using chaotic reference signals. Simulation results have shown that appropriately utilizing chaos is indeed beneficial to engineering design, and this potential of chaos applications in control systems engineering should be further explored.

5. REFERENCES

Chen, G. and Dong, X. (1998). *From Chaos to Order: Methodologies, Perspectives and Applications*, World Scientific Pub. Co., Singapore.

Ge, S. S., C.C. Hang, T.H. Lee and T. Zhang (2001). *Stable Adaptive Neural Network Control*, Kluwer Academic Publishers, Norwell, USA.

Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*, 2nd Ed. Prentice-Hall, New Jersey.

Holmstrom, L. and P. Koistinen (1992). Using additive noise in back-propagation training, *IEEE Trans. Neural Networks*, vol. 3, no. 1, pp. 24-38.

Lewis, F. L., S. Jagannathan and A. Yeildirek (1999). *Neural Network Control of Robot Manipulators and Nonlinear Systems*, Taylor & Francis, London.

Nicolis, G. and I. Prigogine (1989). *Exploring complexity*, W. H. Freeman and Company, New York.

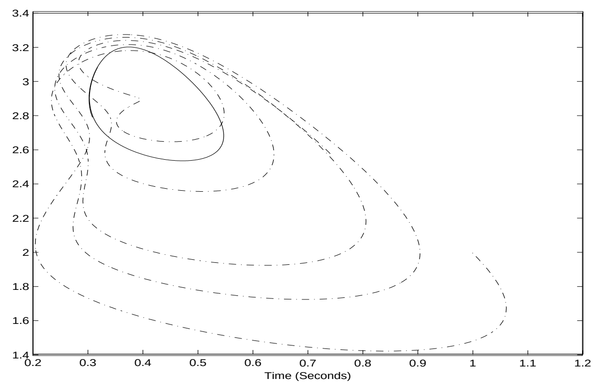

Fig. 1. Phase-plane trajectories of the Brusselator (limit cycle–solid line, chaotic attractor–dashdotted line).

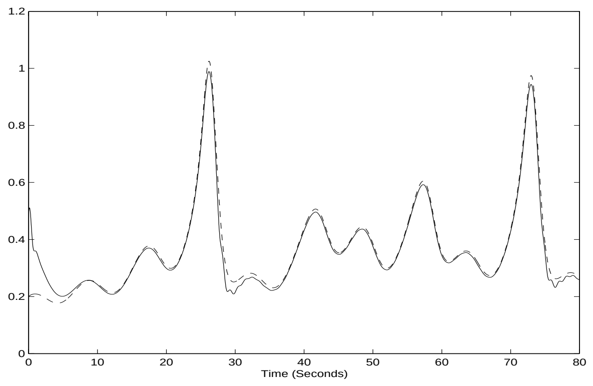

Fig. 4. Tracking a chaotic signal–training phase (y: solid line, y_d: dashed line).

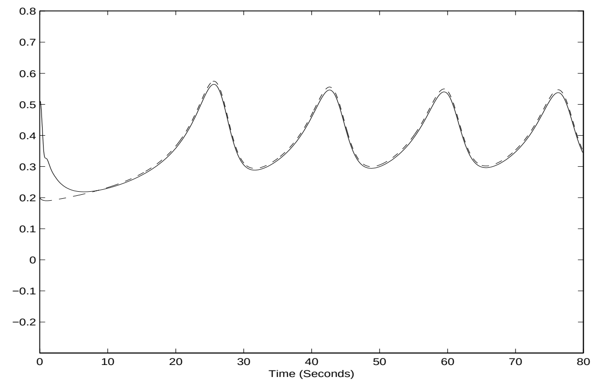

Fig. 2. Tracking a periodic signal–training phase (y: solid line, y_d: dashed line).

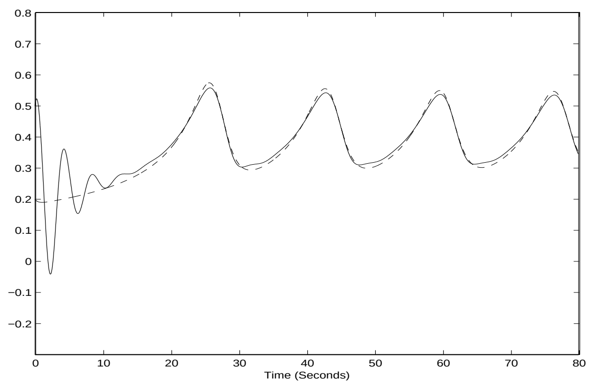

Fig. 5. Tracking the periodic signal again–operational phase (y: solid line, y_d: dashed line).

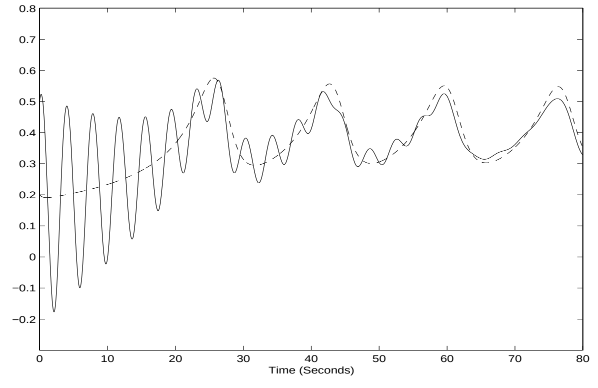

Fig. 3. Tracking a periodic signal–operational phase (y: solid line, y_d: dashed line).

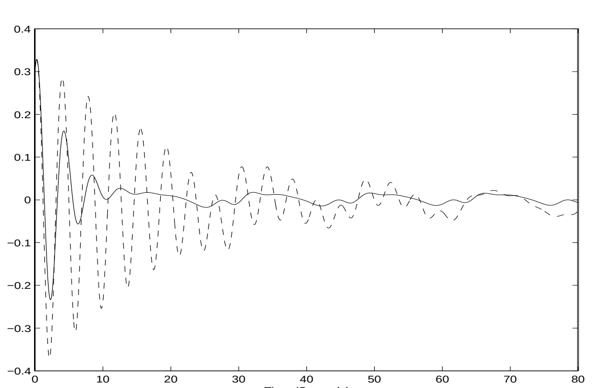

Fig. 6. Tracking errors—operational phase (training with chaotic signal: solid line, training with periodic signal: dashed line).