INTERNATIONAL JOURNAL OF ADAPTIVE CONTROL AND SIGNAL PROCESSING
Int. J. Adapt. Control Signal Process. 2003; **17**: 467–488 (DOI: 10.1002/acs.757)

Smart neural control of uncertain non-linear systems

Cong Wang[1,†], Guanrong Chen[1,*,‡] and Shuzhi S. Ge[2,§]

[1] *Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, People's Republic of China*
[2] *Department of Electrical & Computer Engineering, National University of Singapore, Singapore 117576, Singapore*

SUMMARY

In this paper, we present a 'smart' neural control scheme for uncertain non-linear systems using the localized radical basis function (RBF) networks. This scheme is designed such that the current control action can utilize the knowledge that the NN learned from the past control process. Compared with most existing adaptive neural controllers, which are in general very-high-order dynamic controllers due to the simultaneous adaptation of a large number of neural weights, the smart RBF neural controller is a static and low-order one, and thus is more computationally feasible in practical design and implementation. To improve the generalization ability of the RBF networks, which plays an important role in the smart neural control scheme, chaotic reference signals are employed in the training phase of the scheme, where the complex chaotic signals offer richer information for NN learning due to the ergodicity of chaos. The proposed neural control scheme can act 'smartly' in the operational phase after the RBF networks have been well-trained in the training phase, in a way similar to the process that humans accomplish some complicated control tasks easily after the 'neuro-controllers' in their brains have been well-trained previously. The smart neural control scheme also provides a strong motivation for the current research on chaos generation. Simulation studies are included to demonstrate the effectiveness of the new control scheme. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: neural networks; smart neural control; localized RBF networks; generalization; chaos

1. INTRODUCTION

The past decade has witnessed considerable interest in the area of adaptive neural control for uncertain non-linear systems (see, e.g. Reference [3–5] and the cited references, particularly for a survey of recent development therein). In the literature of adaptive neural control, neural networks are mostly used as function approximators. The unknown non-linearities are

*Correspondence to: Guanrong Chen, Department of Electronic Engineering, City University of Hong Kong, Hong Kong SAR, People's Republic of China.
† E-mail: cwang@ee.cityu.edu.hk
‡ E-mail: gchen@ee.cityu.edu.hk
§ E-mail: elegesz@nus.edu.sg

Contract/grant sponsor: Hong Kong Research Grants Council; contract/grant numbers: CityU 1018/OIE and 1004/02E

 Accepted 29 May 2003

parametrized by linearly or non-linearly parameterized neural networks, such as radial basis function (RBF) neural networks and multilayer neural networks (MNNs). Important features of adaptive neural control include: (i) it is based on the Lyapunov stability theory, (ii) stability and performance of the closed-loop control system can be readily determined, (iii) neural network (NN) weights are tuned on-line, using Lyapunov synthesis method, rather than optimization techniques. To date, adaptive neural design has been considered for several classes of uncertain non-linear systems, including non-linear systems in the Brunovsky form [4–10], non-linear systems in the strict-feedback and pure-feedback forms [11–15], and some classes of MIMO non-linear systems [2]. The recent progress in this area of research indicates that adaptive neural control is especially suitable for controlling highly uncertain, non-linear, and complex systems.

While much efforts have been continuously devoted to extending adaptive neural design for more general non-linear systems, the practical issue concerning the implementation of adaptive neural controllers in engineering applications has not been fully addressed. There exist several fundamental problems about implementing an adaptive neural controller, which needs further investigation. Firstly, in most existing adaptive neural control approaches, the employed neural networks have to learn the uncertainties in each control cycle throughout the entire process, over and over again. In other words, the controller does not utilize much of the knowledge that the neural networks have learned previously in the earlier control cycles. All the weights of the NNs have to be updated in each control cycle in the process, even all the elements of the control system, such as the plant, the reference signal, the initial conditions, and the control parameters are kept unchanged. Secondly, when applying a neural network in a closed-loop feedback system, even a static NN becomes a dynamic one [1]. With a large number of neurons being simultaneously updated, adaptive NN controllers are generally very-high-order dynamical controllers, and thus they are complicated and expensive for analog implementation in practice. Thirdly, the generalization ability, which is an important characteristic of neural networks, is usually not considered in such adaptive neural control design. Note that from a neural network learning point of view, often the purpose of using a neural network is to generalize, namely, to process inputs not necessarily in the training set, from which the NNs can provide meaningful out-puts. With all these issues considered, the adaptive neural controllers proposed thus far are believed to be 'not so smart', at least not as 'smart' as desired, 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 neural control scheme that can resolve some of the aforementioned problems. The proposed scheme is composed of two phases. In the first phase, the uncertain non-linear system is controlled by a conventional adaptive neural controller, to track a desired reference signal. The employment of the adaptive neural controller guarantees the stability of the closed-loop system as well as the output tracking to the desired reference signal (see, e.g. Reference [1, 2] and the references therein). This phase can be regarded as the training phase, because the neural weights of the NNs are tuned on-line to learn the unknown non-linearities in the controller. To facilitate and utilize the knowledge learned in this training phase, RBF networks, instead of MNNs, are used as function approximators on the basis of their good spatially localized learning capability. In the second phase, regarded as the operational phase, the same uncertain nonlinear system is controlled by the 'smart' neural controller, which has a similar form as the adaptive neural controller in the training phase, but with the difference that the weights of the RBF networks are fixed to some values obtained from the training phase, and the tuning of the neural weights are turned off. This operational phase is called the

generalization phase. Compared with the adaptive neural controller in the training phase, the neural controller in the operational phase becomes 'smarter', since it exploits the knowledge that the NNs learned from the previous control cycle or process, and it is by nature a static, low-order controller. Therefore, the implementation of such a controller is much simplified. As a result, with possibly smaller size hardware realization and less power consumption, the smart neural controller is more feasible for practical applications. It will be shown that under the condition that the RBF networks are well-trained in the training phase such that some kind of generalization ability of RBF networks 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 neighbourhood 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 non-linearities correctly, when the inputs to the NNs take different values as compared with those in the training phase. In this case, the possibly invalid NN approximation may cause some unexpected problems, such as demotion of control performance and loss of 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.

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 neighbourhood 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 network used in the adaptive neural controller The insufficiency of the training set cannot make the RBF NNs well-trained so as to have a 'good generalization' ability. It is well known that necessary conditions for good generalization for neural networks include [16]: (i) the unknown function to be approximated is smooth enough and; (ii) the training set is sufficiently large and representative.

To improve the network generalization ability for smooth functions in case of a small training set, injecting artificial noise (called 'jitter') into the inputs during training was proposed in [17]. The basic idea of training with jitter is that, 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 [8]. Moreover, training with jitter might cause new problems in neural control design. Specifically, since neural control of a non-linear system is considered, injecting artificial noise might damage the stability of the closed-loop system, or drive the system states to escape to infinity in a finite time, even if the noise is exponentially decaying [18]. Thus, without taking particular actions to cope with the artificial noise, using jitter to improve NN generalization is not an effective way for closed-loop neural control in general.

To provide a larger training set for better generalization of RBF NNs, in this paper, we employ a chaotic reference signal instead of using jitter in the training phase, on the basis of the well-known ergodicity of chaos. Over the past four decades, chaotic behaviours have been found in many simple nonlinear dynamical systems with significant engineering implications [19]. The ergodicity of chaos implies that a chaotic trajectory has a dense set of periodic orbits of different periods, and it passes arbitrarily closely nearby any spatial point within the bounded chaotic attractor of the system. Utilizing this ergodicity of chaos in the training phase, the information for training is much richer when the inputs to the NNs in the adaptive neural controller become non-periodic. Therefore, better generalization for RBF NNs may be achieved as compared to

the training with periodic reference signals. Moreover, the stability and control performance of the closed-loop system can be more easily guaranteed than the training with jitter. In the operational phase to follow, the well-trained neuro-controller will be able to track any periodic or non-periodic trajectories nearby the chaotic attractor, with a good generalization ability. It will be shown that the richer the chaotic reference signals, the better the training of the RBF NNs, and consequently, the better the control performance of the smart neural controller. This reveals that chaos can be quite beneficial to engineering design and applications if it is appropriately utilized [19].

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 (such as riding a bicycle or a sailboard). The idea of the smart neural control design can be extended to many classes of uncertain non-linear systems, including strict-feedback systems, pure-feedback systems, and nonlinear MIMO systems and is expected to be applicable to a wide variety of industrial applications.

The rest of the paper is organized as follows: The smart neural control scheme is presented in Section 2 for a simple second-order non-linear system. Section 3 considers the improvement of the generalization ability of RBF NNs by using chaotic reference signals. Simulation results performed on an illustrative example are demonstrated through Sections 2 and 3, to show the effectiveness of the approach. Section 4 presents an extension of the smart neural control to uncertain strict-feedback systems. Section 5 concludes the paper.

2. SMART NEURAL CONTROLLER DESIGN

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

2.1. RBF networks

In adaptive neural control design, RBF networks and MNNs are usually used as tools for modelling non-linear functions because of their capabilities in function approximation. The reason we prefer an RBF NN in the smart neural control scheme is that the RBF networks belong to the class of local approximators, where each basis function can only locally affect the network output. The RBF networks store information locally in a transparent fashion, in the sense that the adaptation in one part of the input space does not significantly affect the knowledge stored in a different area, i.e. they have spatially localized learning capability. As will be shown below, it is the local property of RBF NNs that makes it simple to utilize the 'knowledge' or 'experience' gained from the past control processes.

In the following, we first give a brief introduction to the RBF networks. The RBF networks can be considered as a two-layer network in which the hidden layer performs a fixed non-linear transformation with no adjustable parameters, i.e. the input space is mapped into a new space. The output layer then combines the outputs in the latter space linearly. Therefore, they belong to a class of linearly parameterized networks, and can be describedin the following

form:

$$f_{nn}(Z) = \sum_{i=1}^{L} w_i s_i(Z) = W^{\mathrm{T}} S(Z) \tag{1}$$

where $Z \in \Omega_Z \subset R^q$ is the input vector, Ω_Z is a compact set, $W = [w_1, w_2, \ldots, w_L]^{\mathrm{T}} \in R^L$ is the weight vector, $L > 1$ is the NN node number, and $S(Z) = [s_1(Z), \ldots, s_L(Z)]^{\mathrm{T}}$, with $s_i(\cdot)$ being the radial basis functions. Commonly used RBFs are the Gaussian functions, which have the form

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

where $\mu_i = [\mu_{i1}, \mu_{i2}, \ldots, \mu_{iq}]^{\mathrm{T}}$, is the centre of the receptive field and η_i is the width of the Gaussian function. The radial basis functions can also be chosen in Hardy's multiquadric form [20]

$$s_i(Z) = \sqrt{\sigma_i^2 + (Z-\mu_i)^{\mathrm{T}}(Z-\mu_i)} \tag{3}$$

or Inverse Hardy's multiquadric form [9]

$$s_i(Z) = \frac{1}{\sqrt{\sigma_i^2 + (Z-\mu_i)^{\mathrm{T}}(Z-\mu_i)}} \tag{4}$$

Both Gaussian function and inverse multiquadric function belong to the localized basis functions, in the sense that $s_i(Z) \to 0$ as $\|Z\| \to \infty$. The Hardy's multiquadric function is non-local in that $s_i(Z) \to \infty$ as $\|Z\| \to \infty$.

For a continuous function $f(Z) : \Omega_Z \to R$, it has been shown (see, e.g. Reference [16]) that, when the node number L is sufficiently large, an RBF network, $W^{\mathrm{T}} S(Z)$, can be used to approximate $f(Z)$ over the compact set Ω_Z with arbitrary accuracy $\epsilon^* > 0$, i.e.

$$\max_{Z \in \Omega_z} \left| f(Z) - W^* S(Z) \right| < \epsilon^* \tag{5}$$

or

$$f(Z) = W^{*\mathrm{T}} S(Z) + \epsilon, \quad \forall Z \in \Omega_Z \tag{6}$$

where $|\epsilon| < \epsilon^*$. Moreover, it was shown Reference [5] that Gaussian RBF networks $W^{\mathrm{T}} S(Z)$, with a finite number of Gaussian nodes centred on a regular lattice in Ω_Z, and with fixed variances, can uniformly approximate a smooth function $f(Z) : \Omega_Z \to R$ to any chosen tolerance ϵ^* according to (6).

2.2. Training phase

To illustrate the basic ideas of the proposed smart neural control scheme, a simple second-order non-linear system in the following Brunovsky form is discussed:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = f(x_1, x_2) + g(x_1, x_2)u \\ y = x_1 \end{cases} \tag{7}$$

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

Basically, there are two classes of adaptive neural control approaches. The first one is the indirect adaptive neural control (see, e.g. Reference [9, 11]), where NNs are employed to approximate the unknown non-linearities in the system dynamics. Another one is the direct approach [2, 12, 13], where the unknown non-linearities in the desired controller are approximated by NNs. For uncertain nonlinear systems with unknown affine terms (e.g.,with unknown $g(x_1, x_2)$), the direct approach has some advantages with respect to the indirect approach. Firstly, it is simpler to analyse and implement. In general, the stability proof for indirect adaptive approach tends to be much more algebraically involved than the proof of the direct one. Secondly, the direct approach does not require parameter projection, which is generally employed to keep the approximations of the affine terms bounded away from zero in the indirect approach. Thus, due to these advantages, a direct adaptive neural controller is employed in the training phase of the smart neural control scheme.

The control objective is to design an adaptive neural controller for system (7) such that

(i) all the signals in the closed-loop remain uniformly ultimately bounded;
(ii) the output y follows a desired trajectory y_d generated from the following reference model:

$$\begin{aligned} \dot{x}_{di} &= f_{di}(\bar{x}_d) \\ y_d &= x_{d1} \end{aligned} \tag{8}$$

where $\bar{x}_d = [x_{d1}, x_{d2}]^{\mathrm{T}} \in R^2$ are the states, $y_d \in R$ is the system output, and $f_{di}(\cdot), i = 1, 2$ are known smooth nonlinear functions.

Assumption 1

The states of the reference model remain uniformly bounded on a compact set Ω_d, i.e. $\bar{x}_d \in \Omega d$, $\forall t \geq 0$.

Assumption 2

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

The above assumption implies that $g(x_1,x_2)$ is either strictly positive or strictly negative. Without losing generality, it is assumed that $g(x_1, x_2) > g_0 > 0$, $\forall \bar{x} \in \Omega \in R^2$.

For the control of the nonlinear system (7), a direct adaptive neural controller using an RBF NN can be obtained from [2] as

$$u = -z_1 - c_2 z_2 - \hat{W}^{\mathrm{T}} S(Z) \tag{9}$$

where $z_1 = x_1 - y_d$, $z_2 = x_2 - \alpha_1$, $Z = [x_1, x_2, \dot{\alpha}_1]^{\mathrm{T}}$ with $\alpha_1 = -c_1 z_1 + \dot{x}_{d1}, c_1, c_2 > 0$ are design parameters, and $\dot{\alpha}_1 = \left(\partial \alpha_1 / \partial x_1\right) \dot{x}_1 + \left(\partial \alpha_1 / \partial x_d\right)^{\mathrm{T}} \dot{x}_d = c_1 x_2 + \sum_{i=1}^{2} \left(\partial \alpha_1 / \partial x_{di} f_{di}\right)$ is an intermediate variable which is computable. The RBF network, $\hat{W}^{\mathrm{T}} \mathrm{S}(\mathrm{Z})$, is used to approximate the unknown function $h(Z) = 1/g(x_1, x_2)\, f(x_1, x_2) - \dot{\alpha}_1)$, where W is an estimate of W^*, and is updated via

$$\dot{\hat{W}} = \dot{\tilde{W}} = \Gamma S(Z) z_2 - \sigma \hat{W} \tag{10}$$

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

Lemma 1 (Ge and Wang [14])

Consider the closed-loop system consisting of the plant (7), the reference model (8), the controller (9), and the NN weight updating law (10). Then, for approprate initial conditions in a sufficiently large compact set $\Omega_Z \in R^3$, all signals in the closed-loop system remain bounded, and the output tracking error $y(t) - y_d(t)$ converges to a small neighbourhood around zero by appropriately choosing design parameters.

Remark 1

To date, adaptive neural control has been considered for much more general non-linear systems in the literature. The purpose of starting with the simple nonlinear system (7) is for a clear demonstration of the problems in adaptive neural control. In particular, we can see from the control law (9) and the adaptation law (10) that the neural weights $\hat{W}$ have to be updated throughout the control process repeatedly even for the same control task, and the adaptive neural controller (consisting of (9) and (10)) is in fact a very-high-order controller. These problems will be resolved by the smart neural control to be designed in the next subsection.

Since the knowledge learned in previous adaptive neural control process is to be exploited by the smart neural controller in the following operational phase, it is necessary to investigate what the NNs have learned in the training phase. It is clear that accurate convergence of $\hat{W}$ to W^* generally cannot be achieved unless the PE (persistent exciting) condition is satisfied [5, 10]. Since $\tilde{W} = \hat{W} - W^*$, one has

$$\begin{aligned} h(Z) &= W^{*\mathrm{T}} S(Z) + \epsilon = \hat{W}^{\mathrm{T}} S(Z) - \tilde{W}^{\mathrm{T}} S(Z) + \epsilon \\ &= \hat{W}^{\mathrm{T}} S(Z) + \epsilon_1, \quad \forall Z \in \Omega_{Z^1} \end{aligned} \tag{11}$$

where $\epsilon_1 = \epsilon - \tilde{W}^{\mathrm{T}} S(Z)$ is the practical approximation error in the training phase, with most possibly $|\epsilon_1| > |\epsilon|$, and 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 2

Note that $|\epsilon_1| > |\epsilon|$ implies that $\hat{W}^{\mathrm{T}} S(Z)$ might not be able to approximate the unknown non-linearity $h(Z)$ to the accuracy ϵ in adaptive neural control. This can be regarded as one feature of adaptive neural control, i.e. as long as the control objective (especially, the tracking task) is fulfilled, it is neither important nor necessary to achieve accurate approximation of the uncertainties. This feature is also compatible with one feature of human control process: even if the neuro-controllers in human brain have learned how to control, human usually cannot tell whether the learned knowledge coincides accurately with the true knowledge. On the other hand, by appropriately choosing the design parameters, some knowledge of the uncertainties has been learned by the RBF networks, and it is the inexplicit knowledge in $\hat{W}$ that needs to exploit in the following operational phase.

Remark 3

Notice also that the non-necessity of accurate NN approximation does not mean the insignificance of NN approximation, and of the training of NNs. Due to the property of the

localized RBF network, specifically, the property that $|s_i(Z)|$ become very small for sufficiently large values of radius $||Z - \mu_i||$ (see (2)), one can obtain from the adaptation law (10) that for the neurons whose centres are far away from the trajectories of inputs Z, the term $\Gamma S(Z)z_2$ in (10) will become very small. In this case, the weights of these neurons will converge to a small neighborhood of zero because of the term $-\sigma\hat{W}$ in Equation (10). This means that only the neurons whose centres are close to the trajectories of the input Z will be activated and updated in the training phase.

Remark 4

Since it is common to employ periodic reference signals in the literature of adaptive neural control, the convergence of the system's output to a small neighbourhood of a periodic reference signal will only make a limited number of neurons updated in the training phase. This kind of insufficient training cannot make the RBF networks to obtain 'good generalization' ability. This problem will be reduced in Section 4 by using chaotic reference signals.

2.3. Operational phase

To resolve the afore-mentioned problems in adaptive neural control design, a smart neural controller is presented here for the same non-linear system (7) in the following operational phase. The control task is to achieve tracking of the output y to the same or a similar reference signal y_d (still generated from the reference model (8)), under the restriction that the tuning of the neural weights is turned off. To fulfil this task, the knowledge learned in the training phase must be effectively utilized in the operational phase. By exploiting the local property of RBF networks, the weight of each neuron in the NNs is chosen (by a heuristic technique) as the value which is related to the maximum value of the neural weight in the training phase. All 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

$$u = -z_1 - c_2 z_2 - \bar{W}^{\mathrm{T}} S(Z) \tag{12}$$

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

$$\bar{w}_i = \lambda \hat{w}_i(t^i_{\max}), \quad i = 1, \ldots, L \tag{13}$$

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, $\hat{W} = [\hat{w}_1, \ldots, \hat{w}_L]^{\mathrm{T}} \in R^L$ 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.

The following theorem shows the stability and control performance of the closed-loop system.

Theorem 1

Consider the closed-loop system consisting of the plant (7), the reference model (8), and the smart neural controller (12) with the RBF NN weights given by Equation (13). Assume that (i) in the training phase, the RBF network $\hat{W}^{\mathrm{T}} S(Z)$ is well-trained such that Ω_{Z_1} is large enough, and (ii) in the operational phase, the input Z to the RBF network $\bar{W}^{\mathrm{T}} S(Z)$ remains within or

near Ω_{Z_1} for all $t \geqslant 0$. Then, the smart neural controller can accomplish the same tracking task as in the training phase, i.e. the output $y(t)$ of the controlled system converges to a small neighbourhood of the desired reference signal $y_d(t)$.

Proof. The selection of $\bar{W} = [\bar{w}_1, \ldots, \bar{w}_L]^T$ in (13) is based on the feature of the localized RBF network. With the spatially localized basis function $S(Z)$, the adaptation law (10) makes the response of each neuron in the RBF NN sensitive only to local regions of the input space. Thus in the training phase, the ith neuron's weight $\hat{w}_i(t)$ will most probably reach its maximum value when the input trajectory passes by the region close to the receptive field centre of this neuron. By choosing $\bar{W} = [\bar{w}_1, \ldots, \bar{w}_L]^T$ according to (13), the unknown nonlinearity $h(Z)$ can be approximated by $\bar{W}^T S(Z)$ to an accuracy as good as by using $\hat{W}^T S(Z)$, i.e.

$$\begin{aligned} h(Z) &= W^{*T} S(Z) + \epsilon = \hat{W}^T S(Z) + \epsilon_1 \\ &= \bar{W}^T S(Z) + \epsilon_2, \quad \forall Z \in \Omega_{Z^1} \end{aligned} \tag{14}$$

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 a region nearby Ω_{Z^1}, i.e.

$$h(Z) = \bar{W}^T S(Z) + \epsilon_3, \quad \forall Z \in \Omega_{Z^2} \tag{15}$$

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 require achieving accurate approximation using $\bar{W}^T S(Z)$ in both Ω_{Z^1} and Ω_{Z^2}, since the purpose here is to exploit $\hat{W}(t)$ effectively such that the same tracking task can be accomplished.

Since the unknown but smooth unknown function $h(Z)$ can be approximated by $\bar{W}^T S(Z)$ to the accuracy ϵ_2 (in (14)) and ϵ_3 (in (15)), which are both close to ϵ_1, following a similar procedure in the proof of Lemma 1, the tracking error $y - y_d$ can be proven to converge to a neighbourhood of zero, which is as small as that achieved by using $\hat{W}^T S(Z)$ in the training phase. □

Remark 5

Note that the stability result obtained in the operational phase is 'local', since the output tracking can only be achieved under the assumption that the NN input Z remains within $\Omega_{Z^1} \cup \Omega_{Z^2}$, where the NN approximation of $h(Z)$ can be guaranteed. To achieve global results, an extra non-adaptive component can be combined into the control strategy, e.g. using sliding model control as proposed in Reference [5].

Remark 6

The benefits of the smart neural controller thus obtained include:

- The knowledge the RBF networks learned from the training phase is utilized. The turn-off of the neural weights tuning makes the smart neural controller a static low-order controller, which leads to much simpler hardware implementation. The first two problems in adaptive neural control design (mentioned in the Introduction Section) are resolved.
- The omission of the NN tuning process makes the smart neural controller capable of manipulating the same uncertain non-linear system with fewer computational resources

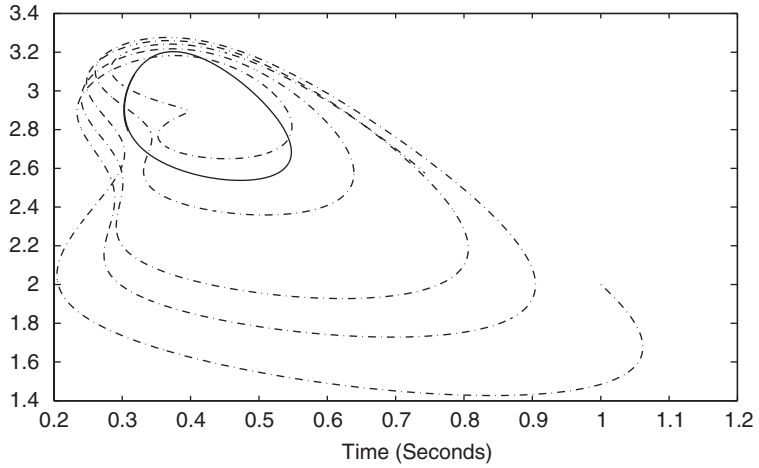

Figure 1. Phase-plane trajectories of the Brusselator (limit cycle: solid line, chaotic attractor: dashdotted line).

and less power consumption. This is similar to the process that when humans have learned how to control some complicated motion tasks, they can do the same job 'easily,' even to the extent without conscious thoughts. In this sense, the smart neural controller is compatible with the 'autonomic' control ability of humans, i.e. it can perform the same (or similar) control tasks in a 'sub-conscious' status with great ease.

2.4. Simulation studies

To verify and test the smart neural control scheme described above, the following van der Pol oscillator system [21] is taken as the plant for control:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_1 + \beta(1 - x_1^2)x_2 + (1.5 + 0.3\cos(x_1))u \\ y &= x_1 \end{aligned} \tag{16}$$

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 [21]:

$$\begin{aligned} \dot{x}_{d1} &= A - (B + 1)x_{d1} + x_{d1}^2 x_{d2} \\ \dot{x}_{d2} &= -Bx_{d1} - x_{d1}^2 x_{d2} \\ y_d &= x_{d1} \end{aligned} \tag{17}$$

where x_{d1} and x_{d2} are system states, A, $B > 0$ are system parameters. As shown in Reference [21], the phase-plane trajectory of the Brusselator model approaches a limit cycle when $B > A^2 + 1$ (the solid line in Figure 1, with $A = 0.4$ and $B = 1.2$). The phase-plane trajectory becomes a chaotic attractor when A is modulated by the harmonic oscillation: $A = A_0 + a\cos(\omega t)$ (the dashdotted line in Figure 1, with $A_0 = 0.4$ and $B = 1.2$, $a = 0.05$ and $\omega = 0.81$).

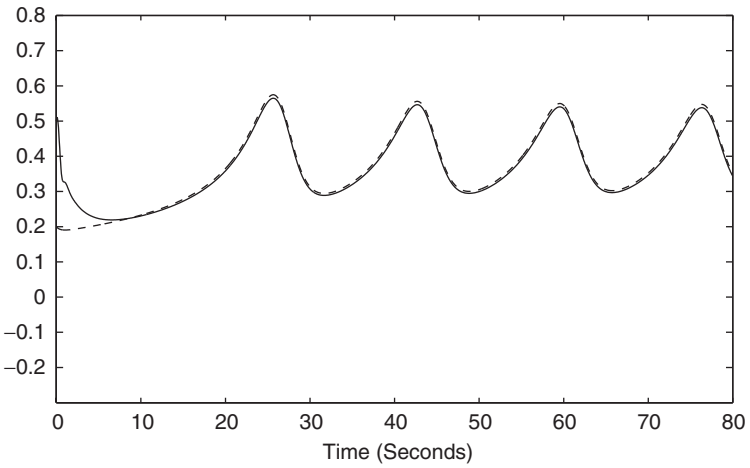

Figure 2. Tracking a periodic signal-training phase (y: solid line, y_d: dashed line).

Firstly, the periodic signal is employed as the reference signal for training the RBF NN during the training phase. The adaptive neural controller (9) is used to control the uncertain system (17). The weights of the NN are updated online according to Equation (10), so as to learn the unknown nonlinearity $h(Z) = 1/g(x_1,x_2)(f(x_1,x_2) - \dot{\alpha}_1$ in the desired control.

In the following operational phase, system (17) is controlled by the designed smart neural controller (12), where the tuning of the weights are set off and the NN weights are chosen according to Equation (13), with $\lambda = 0.75$.

In the simulation, the RBF NN, $\hat{W}^{\mathrm{T}}S(Z)$ contains 125 nodes (i.e. $L = 125$), with centres $\mu_i (i = 1,\ldots,L)$ evenly spaced on $[0,1.2] \times [-0.6,0.6] \times [-0.6,0.6]$, and with widths $\eta_i = 0.3$ $(i = 1,\ldots,L)$. The design parameters of the above controller are $c_1 = 0.7$, $c_2 = 0.7$, $\Gamma = \mathrm{diag}\{3.0\}$, $\sigma = 0.2$ The initial weights $\hat{W}(0) = 0.0$, the initial conditions $[x_1(0),\ x_2(0)]^{\mathrm{T}} = [0.5,\ 0.2]^{\mathrm{T}}$ and $[x_{\mathrm{d}1}(0),\ x_{\mathrm{d}2}(0)]^{\mathrm{T}} = [0.2,\ 0.3]^{\mathrm{T}}$. Note that most of the design parameters are chosen in a way similar to those in typical adaptive neural control design (see, e.g. Reference [4]).

From Figures 2 and 3, one can see that the output of the system converges to a small neighbourhood of the target periodic signal. The tracking performance shown in Figure 3 becomes worse as compared with that shown in Figure 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 developed smart neural control scheme, as to be further discussed in the next section.

3. IMPROVING NN GENERALIZATION BY CHAOS

In the operational phase of the smart neural control scheme, since the inputs to the NN might take different values as compared with the training phase, the generalization ability of the NN plays an important role in the smart controller design.

To achieve good generalization ability, the training set must be a sufficiently large and representative subset. In common practice, periodic signals are often employed as reference signals in adaptive neural control. The convergence of the system's output to a small

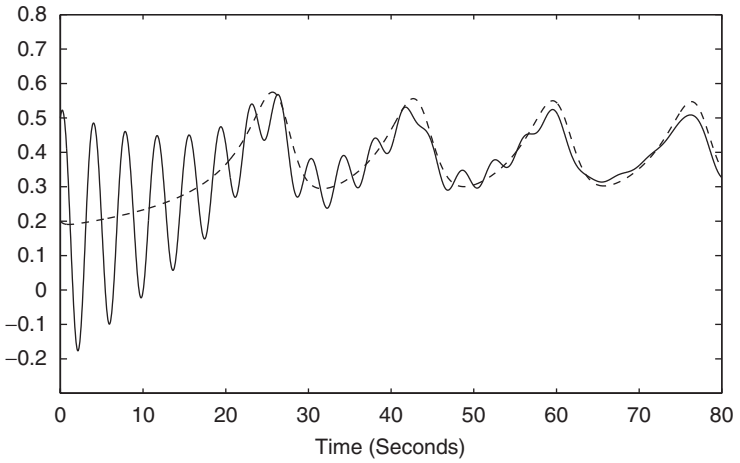

Figure 3. Tracking a periodic signal-operational phase (y: solid line, y_d: dashed line).

neighbourhood of a periodic reference signal will only make a limited number of neurons (whose centres are close to the periodic trajectory of input Z) being updated in the training phase. This kind of training cannot make the regions Ω_{Z^1} and Ω_{Z^2} large enough, and therefore cannot lead to a good NN generalization capability and a good control performance.

In the literature, injecting artificial noise (jitter) to the inputs of NNs has been proposed [17]. The aim is to improve the generalization ability, since for the training of static NNs, injecting noise could make more neurons being updated, and thus expand the input space. In the closed-loop control framework, however, the injected noise (even in an exponentially decaying form) may damage the closed-loop stability [18]. Thus, without taking extra actions to cope with the noise, using jitter to improve NN generalization is not an effective method.

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 $h(Z)$ (as described in Equations (14) and (15)) is still valid. To produce larger such regions, in this section, a chaotic reference signal is employed in the training phase based on the ergodicity of chaos. The ergodicity of chaos implies that a chaotic trajectory has a dense set of periodic orbits of different periods, and it passes arbitrarily closely nearby any spatial point within the chaotic attractor of the system. In other words, the complex chaotic signal will offer much richer information for NN learning than using only periodic reference signals. 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, and in fact dense within a compact set. The tracking to a chaotic reference signal will make 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. Moreover, the stability of the closed-loop system is not damaged on the contrary to the use of scattering jitter. With this learning, in the operational phase to follow, the smart neural controller should become more capable of tracking a given periodic or non-periodic trajectory nearby the chaotic attractor, thanks to the good generalization ability of the well-trained RBF network 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 (9) is applied to system (17) for tracking the chaotic signal y_d (shown in Figure 1).

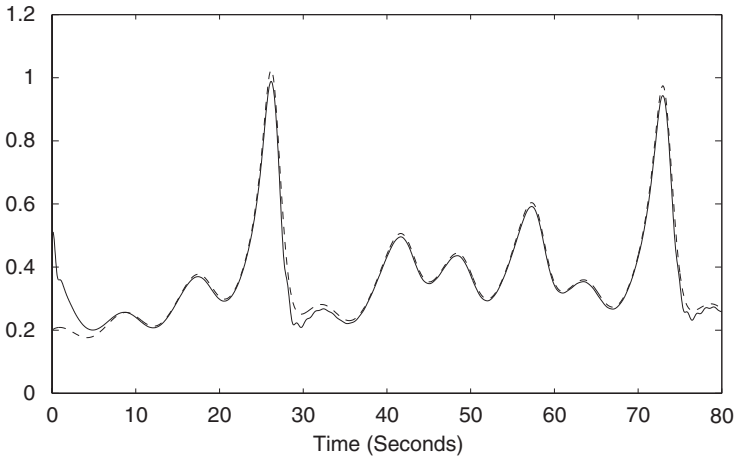

Figure 4. Tracking a chaotic signal-training phase (y: solid line, y_d: dashed line).

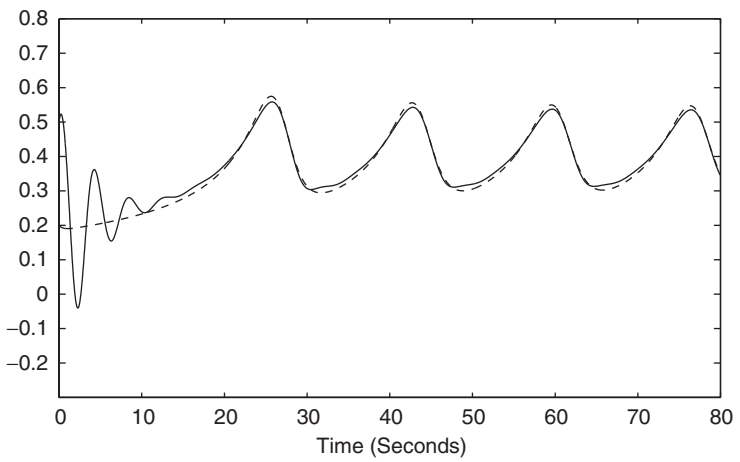

Figure 5. Tracking the periodic signal again—operational phase (y: solid line, y_d: dashed line).

In the operational phase, the weights of the RBF NN are selected according to Equation (13), and system (17) is controlled to track the periodic signal again by using the smart neural controller (12). Recall from Figure 3 that this periodic signal was not tracked satisfactorily by the same controller trained by a periodic signal. In comparison, one can see, from the result shown in Figure 5, that fairly good tracking performance is obtained by the controller after training using the chaotic signal. A detailed comparison of tracking errors of the controller by using these two different training signals is shown in Figure 6.

Remark 7

Tracking to chaotic reference signals provides a feasible method for effective training of the RBF networks in the training phase. It is comprehensible that the richer the chaotic reference signals, the more neurons being activated and updated, and consequently, the better the training

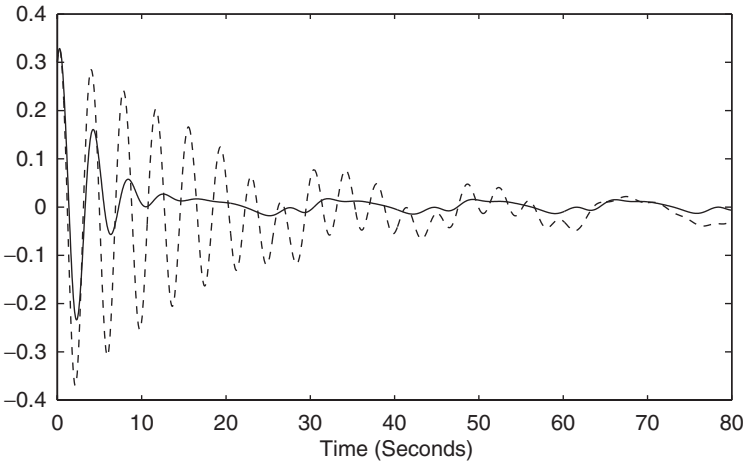

Figure 6. Tracking errors—operational phase (training with chaotic signal: solid line; training with periodic signal: dashed line).

of RBF networks and the better the control performance of the smart neural controllers. Therefore, for a given target periodic signal, it would be quite beneficial if chaotic signals could be produced around this periodic signal, so that regions Ω_{Z^1} and Ω_{Z^2} could be largely expanded. In this sense, the research on smart neural control provides a strong motivation for the current research on chaos generation, or what is called 'chaotification' [19], particularly near a periodic signal [22].

4. EXTENSION TO UNCERTAIN STRICT-FEEDBACK SYSTEMS

We have shown that for the uncertain non-linear system (7), the proposed smart neural controller can utilize the knowledge that the NN learned from the past control process. This advance can be further extended to some other classes of uncertain non-linear systems, such as uncertain strict-feedback systems, uncertain pure-feedback systems, and uncertain MIMO systems, as discussed in Reference [2]. In this section, the smart neural control for a class of uncertain strict-feedback systems is discussed. The extensions to other classes of non-linear systems can be carried out by following some procedures similar to that outlined in this section.

4.1. Training phase

A direct adaptive NN control scheme was proposed in Reference [14] for a class of uncertain strict-feedback systems in the following form:

$$\begin{aligned} \dot{x}_i &= f_i(\bar{x}_i) + g_i(\bar{x}_i)x_{i+1}, \quad 1 \leq i \leq n-1 \\ \dot{x}_n &= f_n(\bar{x}_n) + g_n(\bar{x}_{n-1})u, \quad n \geq 2 \\ y &= x_1 \end{aligned} \tag{18}$$

where $\bar{x}_i = [x_1, \ldots, x_i]^{\mathrm{T}} \in R^i$, $i = 1, \ldots, n$, $u \in R$, $y \in R$ are state variables, system input and output, respectively, and $f_i(\cdot)$ and $g_i(\cdot)$, $i = 1, \ldots, n$, are unknown but smooth functions. The

reference model is in the form (8), with $\bar{x}_d = [x_{d1}, x_{d2}, \ldots, x_{dm}]^{\mathrm{T}} \in R^m$ are the states, $y_d \in R$ is the system output, and $f_{di}(\cdot)$, $i = 1, \ldots, m$ are known smooth non-linear functions.

For a control objective similar to those stated in Section 2.2, the following assumptions were made in Reference [14] for the affine terms $g_i(\cdot)$, $i = 1, \ldots, n$.

Assumption 3

The signs of $g_i(\cdot)$ are known, and there exist constants $g_{i1} \geq g_{i0} > 0$ such that $g_{i1} \geq |g_i(\cdot)| \geq g_{i0}$, $i = 1, \ldots, n$, $\forall \bar{x}_n \in \Omega \subset R^n$, where Ω is a compact set.

Assumption 4

There exist constants $g_{id} > 0$ such that $|\dot{g}_i(\cdot)| \leq g_{id}$, $\forall \bar{x}_n \in \Omega \subset R^n$.

Now, consider the closed-loop system consisting of the plant (18), the reference model (8), the adaptive NN controller

$$u = -z_{n-1} - c_n z_n - \hat{W}_n^T S_n(Z_n) \tag{19}$$

and the neural weights adaptation laws

$$\dot{\hat{W}}_i = \Gamma_i [S_i(Z_i)_{zi} - \sigma_i \hat{W}_i], \quad i = 1, \ldots, n \tag{20}$$

with co-ordinate transformation

$$z_1 = x_1 - x_{d1} \tag{21}$$

$$z_i = x_i - \alpha_{i-1} \tag{22}$$

and virtual controls

$$\alpha_1 = -c_1 z_1 - \hat{W}_1^{\mathrm{T}} S_1(Z_1) \tag{23}$$

$$\alpha_i = -z_{i-1} - c_i z_i - \hat{W}_i^T S_i(Z_i), \quad i = 2, \ldots, n-1 \tag{24}$$

where $c_i > 0$ are the control gains, $\Gamma_i = \Gamma_i^{\mathrm{T}} > 0$ are adaptation gain matrices, $\sigma_i > 0$ are small constants, and the NN inputs are selected as

$$Z_1 = [x_1, \dot{x}_{d1}]^{\mathrm{T}} \subset R^2 \tag{25}$$

$$Z_i = \left[\bar{x}_i^{\mathrm{T}}, \frac{\partial \alpha_{i-1}}{\partial x_1}, \ldots, \frac{\partial \alpha_{i-1}}{\partial x_{i-1}}, \phi_{i-1} \right]^{\mathrm{T}} \subset R^{2i}, \quad i = 2, \ldots, n \tag{26}$$

with

$$\partial \alpha_{i-1} / \partial_{zi}, \ldots, \partial \alpha_i - 1 / \partial x_i \quad \text{and} \quad \phi_{i-1} = \frac{\partial \alpha_{i-1}}{\partial x_d} \dot{x}_d + \sum_{k=1}^{i-1} \frac{\partial \alpha_{i-1}}{\partial \hat{W}_k} \left[\Gamma_k (S_k(Z_k) Z_k - \sigma_k \hat{W}_k) \right]$$

being intermediate variables.

The following result was obtained in Reference [14] for controlling the uncertain strict-feedback system (18).

Lemma 2 (Ge and Wang [14])

Consider the closed-loop system consisting of the plant (18), the reference model (8), the controller (19), and the NN weight updating law (20). Then, for corre-sponding initial

conditions $Z(0)$ within some sufficiently large compact sets $\Omega_i \in R^{2i}$, all signals in the closed-loop system remain uniformly bounded, and the output tracking error $y(t)-y_{d1}(t)$ converges to a small neighbourhood around zero by appropriately choosing design parameters.

4.2. Operational phase

In the training phase of smart neural control of system (18), the neural networks $\hat{W}_i^{\mathrm{T}} S_i(Z_i)$ are update online to learn the unknown non-linearities in the adaptive NN control design. One feature of this adaptive NN control scheme is that intermediate variables $\partial\alpha_{i-1}/\partial x_1, \ldots, \partial\alpha_{i-1}/\partial x_1$, and ϕ_{i-1} are introduced to keep the number of inputs to the RBF NN minimal, so that the possible 'curse of dimensionality' problem [16] caused by a large number of NN inputs can be avoided. However, when considering smart neural control of uncertain strict-feedback system (18), these intermediate variables may generate some problems in the operational phase. Especially, since the intermediate variables are functions of the neural weights $\hat{W}_i$, when all the neural weights are fixed to some certain values (denoted as $\bar{W}_i$), it is possible that these intermediate variables may become much different as compared with those computed by using $\hat{W}_i$.

To resolve this problem, one may still rely on the assumption that the RBF NNs are well-trained in the training phase. In other words, it is assumed that $\Omega_{Z^1} \cup \Omega_{Z^2}$ is large enough as well as dense enough. Note that Ω_{Z^1} represents the region to which the NN inputs belong during the training phase, while Ω_{Z^2} denotes the region around Ω_{Z^1} where the NN approximation is still valid. Under such an assumption, the NN approximation of the unknown non-linearities remains to be satisfied for all $Z \in \Omega_{Z^1} \cup \Omega_{Z^2}$. That is, for the uncertainties $h_i(Z_i)$ in the desired virtual controls α_i^* and the desired control u^*, the following equality holds:

$$\begin{aligned} h_i(Z_i) &= W_i^{*\mathrm{T}} S_i(Z_i) + \epsilon_i \\ &= W_i^{\mathrm{T}} S_i(Z_i) + \epsilon_{i1} = \bar{W}_i^{\mathrm{T}} S_i(Z_i) + \epsilon_{i2}, \forall Z_i \in \Omega_{Z_i^1} \\ &= \bar{W}_i^{\mathrm{T}} S_i(Z_i) + \epsilon_{i3}, \quad \forall Z_i \in \Omega_{Z_i^2} \end{aligned} \tag{27}$$

where $\epsilon_{i1}, \epsilon_{i2}, \epsilon_{i3}$ are the approximation errors in the training and operational phases, respectively, and $|\epsilon_{i1}|-|\epsilon_{i2}|$ and $|\epsilon_{i2}|-|\epsilon_{i3}|$ are small values.

Now, consider the closed-loop system consisting of the plant (18), the reference model (8), the controller

$$u = -z_{n-1} - c_n z_n - \bar{W}_n^{\mathrm{T}} S_n(Z_n) \tag{28}$$

and the virtual controls

$$\alpha_1 = -c_1 z_1 - \bar{W}_1^{\mathrm{T}} S_1(Z_1) \tag{29}$$

$$\alpha_i = -z_{i-1} - c_i z_i - \bar{W}_i^{T} S_i(Z_i), \quad i = 2, \ldots, n-1 \tag{30}$$

with the RBF NN weights $\bar{W}_i = [\bar{w}_{i1}, \ldots, \bar{w}_{iLi}]^{\mathrm{T}}$ being given by

$$\bar{w}_{ij} = \lambda \hat{w}_{ij}(t_{\max}^{ij}), \quad j = 1, \ldots, L_i, \quad i = 1, \ldots, n \tag{31}$$

where $t_{\max}^{ij} = \left\{ t \in [0, \infty) | \max_{t \in [T_0, T_1]} |\hat{w}_{ij}(t)| \right\}$, $[T_0, T_1]$ with $T_1 > T_0 > 0$ representing a piece of time segment within the training phase after the transient process, and $0 < \lambda < 1$ is a constant.

The following theorem shows the control performance of the closed-loop system.

Theorem 2

Assume that (i) in the training phase, the RBF NNs are well-trained such that Ω_{Z^1} is large enough, and (ii) in the operational phase, the inputs to RBF NNs remain to be within $\Omega_{Z^1} \cup \Omega_{Z^2}$ for all $t \geq 0$. Then, the convergence of tracking error $y(t)$–$y_d(t)$ to a neighbourhood around zero can still be achieved by appropriately choosing design parameters.

Proof. The essential idea to prove Theorem 2 is similar to that for proving Theorem 1, although the procedure needs to be modified correspondingly. A detailed proof is given in Appendix A.

Remark 8

Note that in the smart neural control scheme, the neural weights adaptation law (31) provides only one possible method to determine the weights $\bar{W}_i$. Other methods are possible and will be further studied in the near future to achieve even more efficient adaptive learning.

Remark 9

The proposed neural control scheme can be extended to other classes of uncertain nonlinear systems discussed in References [1, 2], and applied to some industrial applications, such as robot control, due to its computational feasibility and simple implementation. With the new control scheme, the robot may act as 'smartly' as humans to accomplish some complicated control tasks.

5. CONCLUSIONS

In this paper, a smart neural control scheme has been designed, which can effectively utilize the knowledge that the RBF NNs learned previously from the past control cycles throughout the entire process. Compared with most existing adaptive neural controllers, which are in general very-high-order dynamic controllers due to the simultaneous adaptation of a large number of neural weights, the smart neural controller is a static low-order one, and is more computationally feasible for practical implementation. Chaotic reference signals are employed in the training phase of the scheme, which yields much better training results to improve the generalization ability of the RBF NNs as compared to the typical training using periodic signals, thanks to the ergodicity of chaos. Simulation results have shown convincing improvements by comparison, and verified the applicability of the proposed smart neural controller to effective control of some general uncertain nonlinear systems.

Finally, it should be pointed out that the developed smart neural control scheme also provides a strong motivation for the current research on chaos generation toward controller design for engineering applications. It is believed that the potential of chaos in control systems design and applications is promising thereafter should be further explored.

ACKNOWLEDGEMENTS

The Research was supported by the Hong Kong Research Grants Council under the CERG Grants CityU 1018/01E and 1004/02E.

APPENDIX A. PROOF OF THEOREM 2

In the smart neural controller design, direct adaptive NN design is combined with the backstepping method. At each recursive step i ($1 \leq i \leq n$) in the backstepping design, the desired virtual control α_i^* and the desired control $u^* = \alpha_n^*$ are first shown to exist, which possess some good stabilizing properties. The desired virtual control α_i^* ($i = 1, \ldots, n$) contains uncertainties $f_i(\cdot)$ and $g_i(\cdot)$ ($i = 1, \ldots, n$), and thus cannot be implemented in practice. The virtual control α_i and the practical control u are constructed by using RBF NNs, $\bar{W}_i^{\mathrm{T}} S_i(Z_i)$, to approximate the unknown parts in the desired virtual control α_i^* and the desired control u^*, where $\bar{W}_i$ denote the neural weights obtained from W_i in the training phase The tuning of the neural weights are turned off in the operational phase.

Step 1: The derivative of $z_1 = x_1 - x_{d1}$ is

$$\dot{z}_1 = f_1(x_1) + g_1(x_1)x_2 - \dot{x}_{d1}$$

Consider x_2 as a virtual control input. There exists a desired feedback control

$$\alpha_1^* = -c_1 z_1 - \frac{1}{g_1}[f_1 - \dot{x}_{d1}] \tag{A1}$$

where $c_1 > 0$ is a design constant.

Since $f_1(x_1)$ and $g_1(x_1)$ are unknown smooth functions, the desired feedback control α_1^* cannot be implemented in practice. From Equation (A1), it can be seen that the unknown part $(1/g_1(x_1))(f_1(x_1) - \dot{x}_{d1})$ is a smooth function of x_1 and $\dot{x}_{d1}$. Denote

$$h_1(Z_1) \triangleq \frac{1}{g_1(x_1)}(f_1(x_1) - \dot{x}_{d1}) = W_1^{\mathrm{T}} S_1(Z_1) + \epsilon_1 \tag{A2}$$

where $Z_1 \triangleq [x_1, \dot{x}_{d1}]^{\mathrm{T}} \subset R^2$. An RBF NN is used to approximate the unknown $h_1(Z_1)$, with W_1^* being the ideal constant weight, and $|\epsilon_1| \leq \epsilon_1^*$ being the approximation error with constant $\epsilon_1^* > 0$.

Similar to the analysis in the proof of Theorem 1, an RBF NN, $\bar{W}_1^{\mathrm{T}} S_1(Z_1)$, can be employed in the operational phase to approximate $h_1(Z_1)$ as

$$\begin{aligned} h_1(Z_1) &= W_1^{*\mathrm{T}} S_1(Z_1) + \epsilon_1 \\ &= \hat{W}_1^{\mathrm{T}} S_1(Z_1) + \epsilon_{11} = \bar{W}_1^{\mathrm{T}} S_1(Z_1) + \epsilon_{12} \forall Z_1 \in \Omega_{Z_1^1} \\ &= \bar{W}_1^{\mathrm{T}} S_1(Z_1) + \epsilon_{13},\ \forall Z_1 \in \Omega_{Z_1^2} \end{aligned} \tag{A3}$$

with ϵ_{11} being the approximation error in the training phase, and ϵ_{12} and ϵ_{13} being the approximation error in the operational phase, $|\epsilon_{11}| - |\epsilon_{12}|$ and $|\epsilon_{12}| - |\epsilon_{13}|$ being small values.

Choosing the virtual control

$$\alpha_1 = -c_1 z_1 - \bar{W}_1^{\mathrm{T}} S_1(Z_1) \tag{A4}$$

where $\bar{W}_1$ is given by Equation (31), to exploit the knowledge or experience stored in the weight $\hat{W}_1$. Then,

$$\begin{aligned} \dot{z}_1 &= f_1(x_1) + g_1(x_1)(z_2 + \alpha_1) - \dot{x}_{d1} \\ &= g_1(x_1)\left[z_2 - c_1 z_1 - \bar{W}_1^{\mathrm{T}} S_1(Z_1) + h_1(Z_1)\right] \\ &= g_1(x_1)(z_2 - c_1 z_1 + \epsilon_{12}) \end{aligned} \tag{A5}$$

Consider the following Lyapunov function candidate:

$$V_1 = \frac{1}{2g_1(x_1)} z_1^2 \tag{A6}$$

The derivative of V_1 is

$$\begin{aligned}\dot{V}_1 &= \frac{z_1 \dot{z}_1}{g_1} - \frac{\dot{g}_1 z_1^2}{2g_1^2} \\ &= \frac{z_1}{g_1}[g_1(z_2 - c_1 z_1 + \epsilon_{12})] - \frac{\dot{g}_1}{2g_1^2} z_1^2 \\ &= z_1 z_2 - c_1 z_1^2 - \frac{\dot{g}_1}{2g_1^2} z_1^2 + z_1 \epsilon_{12}\end{aligned} \tag{A7}$$

Let $c_1 = c_{10} + c_{11}$, with c_{10} and $c_{11} > 0$. Then, Equation (A7) becomes

$$\dot{V}_1 = z_1 z_2 - \left(c_{10} + \tfrac{\dot{g}_1}{2g_1^2}\right) z_1^2 - c_{11} z_1^2 - c_{11} z_1^2 + z_1 \epsilon_{12} \tag{A8}$$

By completing of square, one has

$$-c_{11} z_1^2 + z_1 \epsilon_{12} \le -c_{11} z_1^2 + z_1 |\epsilon_{12}| \le \frac{\epsilon_{12}^2}{4c_{11}} \le \frac{\epsilon_{12}^{*2}}{4c_{11}} \tag{A9}$$

Because $-\left(c_{10} + \dot{g}_1/2g_1^2\right) z_1^2 \le -\left(c_{10} - g_{1d}/2g_{10}^2\right) z_1^2$, by choosing c_{10} such that $c_{10}^* := c_{10} - g_{1d}/2g_{10}^2 > 0$, one gets the following inequality:

$$\dot{V}_1 < z_1 z_2 - c_{10}^* z_1^2 + \frac{\epsilon_{12}^{*2}}{4c_{11}} \tag{A10}$$

where the coupling term $z_1 z_2$ will be cancelled in the next step.

Step i ($2 \le i \le n$): The derivative of $z_i = x_i - \alpha_{i-1}$ is $\dot{z}_i = f_i(\bar{x}_i) + g_i(\bar{x}_i) x_{i+1} - \dot{\alpha}_{i-1}$ ($\dot{x}_{n+1} = u$). By viewing x_{i+1} as a control input for stabilizing the $(z_1, \ldots, z_i)$-subsystem, there exists a desired feedback control $\alpha_i^* = x_{i+1} (\alpha_n^* = u^*)$, in the form

$$\alpha_i^* = -z_{i-1} - c_i z_i - \frac{1}{g_i(\bar{x}_i)} (f_i(\bar{x}_i) - \dot{\alpha}_{i-1}) \tag{A11}$$

where

$$\dot{\alpha}_{i-1} = \sum_{k=1}^{i-1} \frac{\partial \alpha_{i-1}}{\partial x_k} (g_k(\bar{x}_k) x_{k+1} + f_k(\bar{x}_k)) + \phi_{i-1} \tag{A12}$$

with

$$\phi_{i-1} = \sum_{k=1}^{i-1} \frac{\partial \alpha_{i-1}}{\partial x_d} \dot{x}_d + \sum_{k=1}^{i-1} \frac{\partial \alpha_{i-1}}{\partial W_k} \left[\Gamma_k \left(S_k(Z_k) z_k - \sigma_k \hat{W}_k\right)\right]$$

which is computable.

Let

$$h_i(Z_i) := \frac{1}{g_i(\bar{x}_i)} (f_i(\bar{x}_i) - \dot{\alpha}_{i-1}) \tag{A13}$$

denote the unknown part in α_i^* (A11), with

$$Z_i \triangleq \left[\bar{x}_i^{\mathrm{T}}, \frac{\partial \alpha_{i-1}}{\partial x_1}, \ldots, \frac{\partial \alpha_{i-1}}{\partial x_{i-1}}, \phi_{i-1}\right]^{\mathrm{T}} \subset R^{2i} \tag{A14}$$

Note that the number of inputs to the NN is kept minimal by the introduction of the intermediate variables $\partial\alpha_{i-1}/\partial x_1, \ldots, \partial\alpha_{i-1}/\partial x_{i-1}, \phi_{i-1}$. Similar to Step 1, by employing an RBF NN, $\bar{W}_i^{\mathrm{T}} S_i(Z_i)$, to approximate $h_i(Z_i)$, one has

$$\begin{aligned} h_i(Z_i) &= W_i^{*\mathrm{T}} S_i(Z_i) + \epsilon_i = \hat{W}_i^{\mathrm{T}} S_i(Z_i) + \epsilon_{i1} \\ &= \bar{W}_i^{\mathrm{T}} S_i(Z_i) + \epsilon_{i2}, \quad \forall Z_i \in \Omega_{Z_i^2} \end{aligned} \tag{A15}$$

where $\epsilon_{i1}, \epsilon_{i2} > 0$ are the approximation errors in the training and operational phases, respectively, and $|\epsilon_{i1} - \epsilon_{i2}|$ is a small value.

Define the error variable $z_{i+1} = x_{i+1} - \alpha_i$ and choose the virtual control

$$\alpha_i = -z_{i-1} - c_i z_i - \bar{W}_i^{\mathrm{T}} S_i(Z_i) \tag{A16}$$

Then,

$$\begin{aligned} \dot{z}_i &= f_i(\bar{x}_i) + g_i(\bar{x}_i)(z_{i+1} + \alpha_i) - \dot{\alpha}_{i-1} \\ &= g_i[Z_{i+1} - Z_{i-1} - c_i z_i + \epsilon_{i2}], \quad i = 2, \ldots, n-1 \end{aligned} \tag{A17}$$

and

$$\begin{aligned} \dot{z}_n &= f_n(\bar{x}_i) + g_n(\bar{x}_{n-1})u - \dot{\alpha}_{n-1} \\ &= g_n[-Z_{n-1} - c_n z_n + \epsilon_{n2}] \end{aligned} \tag{A18}$$

Consider the Lyapunov function candidate:

$$V_i = V_{i-1} + \frac{1}{2g_i(\bar{x}_i)} z_i^2 \tag{A19}$$

By using (A10) and (A17), with straightforward derivation similar to those employed in Step 1, the derivative of V_i is found to satisfy

$$\dot{V}_i < z_i z_{i+1} - \sum_{k=1}^{i} c_{k0}^* z_k^2 + \sum_{k=1}^{i} \frac{\epsilon_{k2}^{*2}}{4c_{k1}}, \quad i = 2, \ldots, n-1 \tag{A20}$$

and

$$\dot{V}_i < -\sum_{k=1}^{n} c_{k0}^* z_k^2 + \sum_{k=1}^{n} \frac{\epsilon_{k2}^{*2}}{4c_{k1}} \tag{A21}$$

where c_{i0} is chosen such that $c_{i0}^* : \triangleq c_{i0} - g_{id}/2g_{10}^2 > 0$.

Let $\delta \triangleq \sum_{k=1}^{n} \epsilon_{k2}^{*2}/4c_{k1}$. If one chooses c_{k0}^* such that $c_{k0}^i \geq \gamma/2g_{k0}$, i.e. chooses c_{k0} such that $c_{k0} > \gamma/2g_{k0} + gkd/2g_{k0}^2$, $k = 1, \ldots, n$, where γ is a positive constant, then from (A21) the

following inequality is obtained:

$$\begin{aligned}\dot{V}_n &< -\sum_{k=1}^{n} c_{k0}^{*} z_k^2 + \delta \\ &\leq -\sum_{k=1}^{n} \frac{\gamma}{2g_{k0}} z_k^2 + \delta \\ &\leq -\gamma \left[\sum_{k=1}^{n} \frac{1}{2g_k} z_k^2 \right] + \delta \\ &\leq -\gamma V_n + \delta \end{aligned} \tag{A22}$$

From (A22), the so-called Boundedness Theorem (i.e. Theorem 2.14 in Reference [23]) implies that all z_i ($i = 1, \ldots, n$) are uniformly bounded. Since $z_1 = x_1 - x_{d1}$ and x_{d1} are bounded, x_1 is also bounded. From $z_i = x_i - \alpha_{i-1}$, $i = 2, \ldots, n$, and the definitions of virtual controls α_i (A4)(A16), it follows that x_i, $i = 2, \ldots, n$, remain bounded. Using (28), one can conclude that control u is also bounded. Thus, all the signals in the closed-loop system remain bounded.

Let $\rho \triangleq \delta/\gamma > 0$. Then (A22) satisfies

$$0 \leq V_n(t) < \rho + (V_n(0) - \rho)\exp(-\gamma t) \tag{A23}$$

From (A23), one has

$$\begin{aligned}\sum_{k=1}^{n} \frac{1}{2g_k} z_k^2 &< \rho + (V_n(0) - \rho)\exp(-\gamma t) \\ &< \rho + V_n(0)\exp(-\gamma t) \end{aligned} \tag{A24}$$

Let $g^{*} = \max_{1 \leq i \leq n}\{g_{i1}\}$. Then,

$$\frac{1}{2g^{*}} \sum_{k=1}^{n} z_k^2 \leq \sum_{k=1}^{n} \frac{1}{2g^{*}} z_k^2 < \rho + V_n(0)\exp(-\gamma t) \tag{A25}$$

that is,

$$\sum_{k=1}^{n} z_k^2 < 2g^{*}\rho + 2_g^{*} V_n(0)\exp(-\gamma t) \tag{A26}$$

which implies that given $\mu > \sqrt{2g^{*}\rho}$, there will exist a $T > 0$ such that for all $t \geq T$, the tracking error satisfies

$$|z_1(t)| = |x_1(t) - x_{d1}(t)| = |y(t) - y_d(t)| < \mu \tag{A27}$$

where μ is the size of a small residual set which depends on the NN approximation errors ϵ_{i2} and the controller parameters c_i, $i = 1, \ldots, n$. It is then easily seen that the increase in the control gains c_i and NN node number L_j will result in a better tracking performance. □

REFERENCES

1. Lewis FL, Jagannathan S, Yeildirek A. *Neural Network Control of Robot Manipulators and Nonlinear Systems*. Taylor & Francis: London, 1999.
2. Ge SS, Hang CC, Lee TH, Zhang T. *Stable Adaptive Neural Network Control*. Kluwer Academic: Norwell, USA, 2001.

3. Narendra KS, Lewis FL (Ed.) Special issue on neural network feedback control. *Automatica*, 2001; **37**:8.
4. Polycarpou MM, Ioannou PA. Modeling, identification and stable adaptive control of continuous-time nonlinear dynamical systems using neural networks. *Proceedings of American Control Conference*, Boston, MA, USA, 1992; 36–40.
5. Sanner RM, Slotine JJE. Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks* 1992; **3**:837–863.
6. Rovithakis GA, Christodoulou MA. Adaptive control of unknown plants using dynamical neural networks. *IEEE Transanctions on Systems, Man and Cybernetics* 1994; **24**:400–412.
7. Lewis FL, Yesildirek A, Liu K. Multilayer neural-net robot controller with guaranteed tracking performance. *IEEE Transanctions on Neural Networks* 1996; **7**:388–398.
8. Spooner JT, Passino KM. Stable adaptive control using fuzzy systems and neural networks. *IEEE Transactions on Fuzzy Systems* 1996; **4**:339–359.
9. Polycarpou MM. Stable adaptive neural control scheme for nonlinear systems. *IEEE Transactions on Automatic Control* 1996; **41**:447–451.
10. Farrell J. Stability and approximator convergence in nonparametric nonlinear adaptive control. *IEEE Transactions on Neural Networks* 1998; **9**:1008–1020.
11. Polycarpou MM, Mears MJ. Stable adaptive tracking of uncertain systems using non-linearly parametrized on-line approximators. *International Journal of Control* 1998; **70**:363–384.
12. Kwan C, Lewis FL. Robust backstepping control of non-linear systems using neural networks. *IEEE Transactions on Systems, Man and Cybernetics Part A* 2000; **30**:753–766.
13. Zhang YP, Peng Y, Jiang ZP. Stable neural controller design for unknown non-linear systems using backstepping. *IEEE Transactions on Neural Networks* 2000; **11**:1347–1359.
14. Ge SS, Wang C. Direct adaptive NN control of a class of nonlinear systems. *IEEE Transactions on Neural Networks* 2002; **13**:214–221.
15. Ge SS, Wang C. Adaptive NN control of uncertain nonlinear pure-feedback systems. *Automatica* 2002; **38**:671–682.
16. Haykin S. *Neural Networks: A Comprehensive Foundation*. (2nd edn). Prentice-Hall: Englewood Cliffs, New Jersey, 1999.
17. Holmstrom L, Koistinen P. Using additive noise in back-propagation training. *IEEE Transactions on Neural Networks* 1992; **3**:24–38.
18. Krstić M, Kanellakopoulos I, Kokotović P. *Nonlinear and Adaptive Control Design*. Wiley: New York, 1995.
19. Chen G, Dong X. *From Chaos to Order: Methodologies, Perspectives and Applications*. World Scientific: Singapore, 1998.
20. Micchelli CA. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation* 1986; **2**:11–22.
21. Nicolis G, Prigogine I. *Exploring Complexity*. W. H. Freeman and Company: New York, 1989.
22. Chen G, Yang L. Chaotifying a continuous-time system near a stable limit cycle. *Chaos, Solitons & Fractals* 2003; **15**:245–254.
23. Qu Z. *Robust Control of Nonlinear Uncertain Systems*. Wiley: New York, 1998.

