Design of a Neuro-Controller for Multivariable Nonlinear Time-Varying Systems

H. Al-Duwaish
King Fahd University of Petroleum & Minerals
Department of Electrical Engineering
P.O.Box 677
Saudi Arabia
hduwaish@kfupm.edu.sa

S.Z. Rizvi
King Fahd University of Petroleum & Minerals
Department of Electrical Engineering
P.O.Box 76
Saudi Arabia
srizvi@kfupm.edu.sa

Abstract: This paper presents a controller design method for multi-input multi-output (MIMO) nonlinear time-varying systems using Radial Basis Funtion (RBF) neural network. The developed neuro-controller generates optimal control signals abiding by constraints, if any, on the control signal or on the system output. The proposed controller does not require an explicit knowledge of the states of the system or any apriori knowledge of the structure of nonlinearity of the system. Time based variations in system parameters as well as system nonlinearities are successfully compensated by the neural network. Simulation results for nonlinear time-varying systems are included at the end and controller performance is analyzed.

Key–Words: Neural Network, Radial Basis Functions, Optimization, Multivariable, Constraints.

1 Introduction

Nonlinear adaptive control has been a subject of immense interest among researchers. Almost every real life process has nonlinear behavior. At the same time, systems in real life are prone to change in parameters due to several internal and external factors affecting the process. Hence, robust controller design for nonlinear systems has become a vital requirement in modern control theory and practice. The nonlinearity in systems are most often beyond the limit of linear controllers. This has propelled a lot of nonlinear control techniques to surface over the past two decades like nonlinear PID control. The extended version of nonlinear PID control constitutes sliding mode control which defines an error surface and tries to drive the error to zero.

Amongst several techniques that have surfaced in the recent past to cater to the challenging problem of nonlinear control, artificial neural networks (ANN) have emerged as an efficient class of machines capable of learning complex nonlinear functions. The use of neural networks in controller design is therefore a natural choice. Neural networks have been used for pattern recognition, function approximation, time-series prediction and classification problems for quite some time [1]. The ability of neural networks to map complex input output relationships make them ideal for compensating plant nonlinearities and hence make them ideal for controller design problem.

Several techniques involving neural network have surfaced in the past decade for nonlinear adaptive control. However, in many adaptive control approaches, it is well understood that there exists a necessary assumption that the controlled system has to be a minimum phase system [2, 3, 4], i.e. the zero dynamics of the system must be stable. Several nonlinear adaptive control techniques like those of Chen and Narendra [5], Fu and Chai [6] and Wang and Huang [7] have based their controller on this assumption. Many other works on nonlinear control like those of Wang and Huang [7], Hayakawa et al. [8], and Petre et al. [10] require state-feedback. While state-feedback poses no harm to controller performance, it requires state measurement at every sampling time using sensors, which can increase the implementation cost if the system has significant number of states. As compared to state-feedback, an output-feedback scheme can be less expensive owing to the fact that controlled outputs are a smaller subset of system states in most cases. The controller in [8] also requires a process model derived from mass and energy balance equations, thus requiring a rigorous model. Complete or partial knowledge of the model of the system is a requirement for many such neuro-control schemes [9]. Zhuo et al. [11] also propose an adaptive neural network control scheme for systems containing non-smooth nonlinearities in the actuator device. Though the authors make no assumptions on the unknown system param-

eters and nonlinearities, the control scheme like Wang and Huang [7] is limited to single-input single-output (SISO) systems.

In this paper, a neural network based controller for nonlinear time-varying systems is proposed. The proposed controller utilizes a feedback signal of the controlled outputs rather than all the states, and does not require a rigorous time-varying model. The proposed controller utilizes RBF neural network as compared to multi-layer feed-forward (MF) neural network or Cellular neural network (CNN) used in several control schemes [12, 7, 13, 14]. RBF networks, as described in future sections of the paper, consist of only one hidden layer and hence have significantly reduced learning time as compared to multi-layer perceptrons (MLP)s [1, 15]. RBF has been employed in the literature for controlling specific time-invariant systems like mobile robots as well [16].

A primitive version of the proposed controller for linear time-varying systems was presented in [17]. This work seeks to successfully extend the work in [17] to systems having nonlinearities and having multiple inputs and outputs.

This paper is arranged as follows. Section 2 takes a look at the proposed design. Section 3 seeks to generalize the derived weight update equations for MIMO systems. Once, neural network training scheme and update equations are successfully derived, a detailed look at the performance of the controller under different conditions is analyzed in section 4. Throughout this paper, the following convention for notations has been used. Variables in lower case represent scalar quantities. Lower case bold variables represent vector quantities. Upper case bold variables represent matrices. The only exception to this convention is in the choice of a more conventional J for the cost function.

2 Controller design

Consider a single-input single-output time-varying nonlinear system given by

$$\begin{aligned} \mathbf{x}(t+1) &= f(\mathbf{x}(t),t) + g(u(t),t) \\ y(t) &= h(\mathbf{x}(t),t). \end{aligned} \tag{1}$$

It is required that the process outputs follow a desired reference $r(t)$. The nonlinear time-varying process is approximated by a linear time-invariant (LTI) model using offline identification.

$$\begin{aligned} \mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}u(t) \\ \hat{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}u(t). \end{aligned} \tag{2}$$

As will be shown with the aid of simulation results, the robustness of the RBF network allows us to control the desired system without any knowledge of the model of the plant based on physical laws. Instead, a rough estimate of the system acquired using initial identification provides a good enough learning platform for the neural network and unlike conventional adaptive control approaches, the neural network does not require updating the model. The controller consisting of an RBF nerual network is given by

$$v(t) = \mathbf{w}\boldsymbol{\phi}^T(t), \tag{3}$$

where $\mathbf{w} \in \Re^q$ defines synaptic weights for the RBF output layer and $\boldsymbol{\phi}(t)$ is the basis function vector which is given by

$$\boldsymbol{\phi}(t) = [\phi(\|\mathbf{r}(t) - \mathbf{c}_1\|) \cdots \phi(\|\mathbf{r}(t) - \mathbf{c}_q\|)]. \tag{4}$$

In the above equation, q denotes the number of neurons in the hidden layer, $\mathbf{c}_i$ is the center vector for the i^{th} neuron of that layer, ϕ is the radial basis function, and $\|.\|$ denotes norm. Further, to cope with constraints, if any, on the control signal, an arrangement has to be sought. Constraints in many practical cases are on the magnitude of the control signals, or on the rate of change of control signals.

$$\begin{aligned} u_{min} \leq u(t) \leq u_{max}, \\ \Delta u_{min} \leq \Delta u(t) \leq \Delta u_{max}. \end{aligned} \tag{5}$$

To meet this constraint the output of the RBF network is transformed by a tangent-sigmoid activation function forming the constrained control signal

$$\begin{aligned} u_j(t) &= \alpha \frac{e^{\,kv_j(t)} - 1}{e^{\,kv_j(t)} + 1} \\ &= \alpha \frac{e^{\,k\mathbf{w}_j\boldsymbol{\phi}(t)} - 1}{e^{\,k\mathbf{w}_j\boldsymbol{\phi}(t)} + 1}, \end{aligned} \tag{6}$$

where $\alpha = |u_{min}| = |u_{max}|$ denotes the upper and lower limits of the constraints and k is used to adjust the slope of the linear part of tangent-sigmoid function. Figure 1 shows the tangent-sigmoid constraint function with different values of k.

The difference between the reference $r(t)$ and the process output $\hat{y}(t)$ gives the error $e(t)$. In order to update the controller, the weights of the RBF network are trained in the negative direction of the derivative of cost function J which is given by

$$J = e^2(t). \tag{7}$$

The weight update equation is given by (8) as

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \eta \frac{\partial J}{\partial \mathbf{w}}. \tag{8}$$

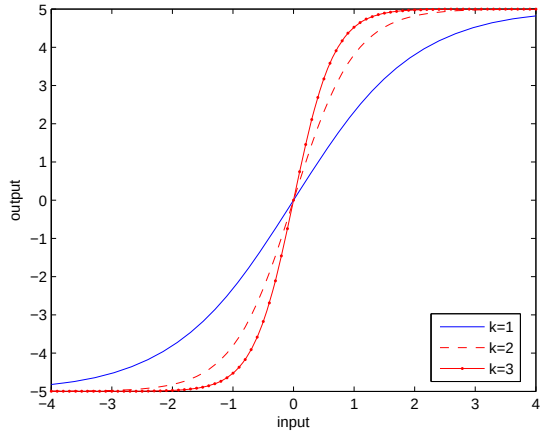

Figure 1: Tangent-Sigmoid function of equation (6) with $\alpha = 5$

Now finding the partial derivative of I w.r.t $\mathbf{w}$

$$\begin{aligned}\frac{\partial J}{\partial \mathbf{w}_j} &= 2e^2(t)\frac{\partial}{\partial \mathbf{w}}e(t)\\ &= 2e^2(t)\frac{\partial}{\partial \mathbf{w}}\left(r(t)-\hat{y}(t)\right)\\ &= -2e^2(t)\frac{\partial}{\partial \mathbf{w}}\hat{y}(t)\end{aligned}$$

Since $y\hat{(t)}$ is defined in terms of a state space model given in equation (2), the above equation can be re-written as

$$\frac{\partial J}{\partial \mathbf{w}_j} = -2e^2(t)\frac{\partial}{\partial \mathbf{w}}\left(\mathbf{C}\mathbf{x}(t)+\mathbf{D}u(t)\right),$$

which can be written as

$$\frac{\partial J}{\partial \mathbf{w}_j} = -2e^2(t)\frac{\partial}{\partial \mathbf{w}}(\mathbf{C}\{\mathbf{A}\mathbf{x}(t-1)+\mathbf{B}u(t-1)\} +\mathbf{D}u(t)).$$

The terms independent of $\mathbf{w}$ would vanish[1],

$$\frac{\partial J}{\partial \mathbf{w}} = -2e^2(t)\left(\frac{\partial \mathbf{C}\mathbf{B}u(t-1)}{\partial \mathbf{w}}+\frac{\partial \mathbf{D}u(t)}{\partial \mathbf{w}}\right). \quad (9)$$

[1]Since $\mathbf{x}(t-1)$ depends on $u(t-2)$ which in turn is a function of $\mathbf{w}$, the dependence of state vector $\mathbf{x}(t-1)$ on the weights $\mathbf{w}$ of the neural network is acknowledged. However the term for derivative of $\mathbf{C}\mathbf{A}\mathbf{x}(t-1)$ w.r.t $\mathbf{w}$ is deliberately neglected since expansion of $\mathbf{x}(t)$ into past state terms $\mathbf{A}\mathbf{x}(t-n)+\mathbf{B}u(t-n)$ for $n \geq 2$ does not yield significant improvement on controller result.

Finding the derivative of tangent-sigmoid function

$$\begin{aligned}\frac{\partial u(t)}{\partial \mathbf{w}} &= \alpha\frac{\partial}{\partial \mathbf{w}}\frac{e^{k\mathbf{w}\boldsymbol{\phi}(t)}-1}{e^{k\mathbf{w}\boldsymbol{\phi}(t)}+1}\\ &= \alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{w}\boldsymbol{\phi}(t)}}{(e^{k\mathbf{w}\boldsymbol{\phi}(t)}+1)^2}. \quad (10)\end{aligned}$$

Equation (9) can now be written as

$$\frac{\partial J}{\partial \mathbf{w}} = -2e^2(t)\left(\frac{\mathbf{C}\mathbf{B}\partial u(t-1)}{\partial \mathbf{w}}+\frac{\mathbf{D}\partial u(t)}{\partial \mathbf{w}}\right)$$

$$= -4\alpha e(t)k\left(\frac{\mathbf{C}\mathbf{B}\phi(t-1)e^{k\mathbf{w}\phi(t-1)}}{(e^{k\mathbf{w}\phi(t-1)}+1)^2} + \frac{\mathbf{D}\phi(t)e^{k\mathbf{w}\phi(t)}}{(e^{k\mathbf{w}\phi(t)}+1)^2}\right) \quad (11)$$

Thus, the final weight update equation is given by

$$\mathbf{w}(k+1) = \mathbf{w}(k) - 4\eta\alpha e(t)k\left(\frac{\mathbf{D}\phi(t)e^{k\mathbf{w}\phi(t)}}{(e^{k\mathbf{w}\phi(t)}+1)^2} + \frac{\mathbf{C}\mathbf{B}\phi(t-1)e^{k\mathbf{w}\phi(t-1)}}{(e^{k\mathbf{w}\phi(t-1)}+1)^2}\right). \quad (12)$$

3 Generalization to MIMO systems

Having derived a weight update equation for the proposed controller, in this section, we seek to extend the proposed controller to MIMO nonlinear time-varying systems. Consider therefore, a multi-input multi-output nonlinear process having p inputs and m outputs as shown in figure 2. It is required that the process outputs follow a desired set of reference points $\mathbf{r}(t) = [r_1(t)\cdots r_m(t)]$. The linear time-invariant approximation, as in the SISO case, will be obtained using offline identification.

$$\begin{aligned}\mathbf{x}(t+1) &= \mathbf{A}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t)\\ \hat{\mathbf{y}}(t) &= \mathbf{C}\mathbf{x}(t)+\mathbf{D}\mathbf{u}(t). \quad (13)\end{aligned}$$

Notice that the bold variables $\mathbf{u}(t)$ and $\hat{\mathbf{y}}(t)$ indicate an array of multiple inputs and outputs at every time instant. The RBF network controller will generate control signals $\mathbf{u}(t)$. Of the p RBF outputs, a vector of synaptic weights $\mathbf{w}_j$ will be updated for each RBF output. The j^{th} output of the RBF network is given by

$$v_j(t) = \mathbf{w}_j\boldsymbol{\phi}^T(t), \quad (14)$$

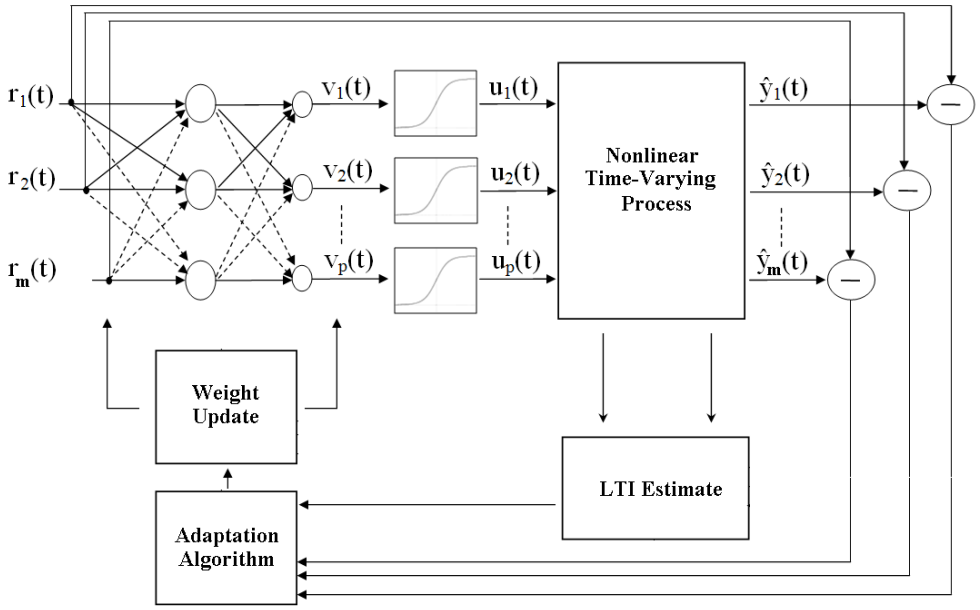

Figure 2: Proposed Neuro-Controller for nonlinear time-varying systems

where $\mathbf{w}_j$ is the vector for weights of j^{th} RBF output, given by

$$\mathbf{w}_j = [w_{1j} \cdots w_{qj}], \tag{15}$$

Keeping the same constraint functions, each RBF output $v_j(t)$ is mapped into constrained control signal $u_j(t)$. The MIMO nonlinear system generates outputs $\hat{\mathbf{y}}(t)$. The difference between the reference $\mathbf{r}(t)$ and the process output $\hat{\mathbf{y}}(t)$ gives the error $\mathbf{e}(t) = [e_1(t) \cdots e_m(t)]^T$. The cost function J for multivariable system would be given by

$$J = \mathbf{e}^T(t)\mathbf{e}(t). \tag{16}$$

The weight update equation for j^{th} control signal $u_j(t)$ is given in equation (17) as

$$\mathbf{w}_j(k+1) = \mathbf{w}_j(k) - \eta \frac{\partial J}{\partial \mathbf{w}_j}. \tag{17}$$

Now finding the partial derivative of J w.r.t $\mathbf{w}_j$

$$\begin{aligned}
\frac{\partial J}{\partial \mathbf{w}_j} &= 2\mathbf{e}^T(t)\frac{\partial}{\partial \mathbf{w}_j}\mathbf{e}(t) \\
&= 2\mathbf{e}^T(t)\frac{\partial}{\partial \mathbf{w}_j}\left(\mathbf{r}(t) - \hat{\mathbf{y}}(t)\right) \\
&= -2\mathbf{e}^T(t)\frac{\partial}{\partial \mathbf{w}_j}\hat{\mathbf{y}}(t) \\
&= -2\mathbf{e}^T(t)\frac{\partial}{\partial \mathbf{w}_j}[\hat{y}_1(t \cdots \hat{y}_m(t))] \\
&= -2\mathbf{e}^T(t)\frac{\partial}{\partial \mathbf{w}_j}\left(\mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)\right),
\end{aligned}$$

Proceeding as before, the expression for partial derivative of cost function can be written as

$$\frac{\partial I}{\partial \mathbf{w}_j} = -2\mathbf{e}^T(t)\frac{\partial}{\partial \mathbf{w}_j}(\mathbf{C}\{\mathbf{A}\mathbf{x}(t-1) + \mathbf{B}\mathbf{u}(t-1)\} + \mathbf{D}\mathbf{u}(t)).$$

Once again, the dependence of the state vector $\mathbf{x}(t-1)$ on the weights $\mathbf{w}_j$ is neglected for reasons mentioned in the previous footnote. The equation for partial derivative of J therefore becomes

$$\frac{\partial J}{\partial \mathbf{w}_j} = -2\mathbf{e}^T(t)\left(\frac{\mathbf{C}\mathbf{B}\partial \mathbf{u}(t-1)}{\partial \mathbf{w}_j} + \frac{\mathbf{D}\partial \mathbf{u}(t)}{\partial \mathbf{w}_j}\right) \tag{18}$$

Now since $\mathbf{C}$ and $\mathbf{B}$ are not single row or single column vectors, the expression is expanded as

$$\frac{\partial J}{\partial \mathbf{w}_j} = -2\mathbf{e}^T(t)\left(\begin{bmatrix} \psi_{11} \cdots \psi_{1p} \\ \vdots \quad \ddots \quad \vdots \\ \psi_{m1} \cdots \psi_{mp} \end{bmatrix}\begin{bmatrix} \frac{\partial}{\partial \mathbf{w}_j}u_1(t-1) \\ \vdots \\ \frac{\partial}{\partial \mathbf{w}_j}u_p(t-1) \end{bmatrix}\right.$$

$$+\left[\begin{array}{ccc} d_{11}\cdots d_{1p} \\ \vdots \quad \ddots \quad \vdots \\ d_{m1}\cdots d_{mp} \end{array}\right]\left[\begin{array}{c} \frac{\partial}{\partial \mathbf{w}_j}u_1(t) \\ \vdots \\ \frac{\partial}{\partial \mathbf{w}_j}u_p(t) \end{array}\right]\Bigg), \quad (19)$$

where $\mathbf{\Psi} \epsilon \Re^{m\times p}$ is the product of $\mathbf{C} \epsilon \Re^{m\times n}$ and $\mathbf{B} \epsilon \Re^{n\times p}$. The derivative of all terms except $u_j(t-1)$ and $u_j(t)$ would vanish. Substituting $\frac{\partial u_j(t)}{\partial \mathbf{w}_j}$ from the derivative equation 10,

$$\frac{\partial J}{\partial \mathbf{w}_j} = -2\mathbf{e}^T(t)\left(\left[\begin{array}{c} \psi_{1j}\frac{\partial}{\partial \mathbf{w}_j}u_j(t-1) \\ \vdots \\ \psi_{mj}\frac{\partial}{\partial \mathbf{w}_j}u_j(t-1) \end{array}\right] + \left[\begin{array}{c} d_{1j}\frac{\partial}{\partial \mathbf{w}_j}u_j(t) \\ \vdots \\ d_{mj}\frac{\partial}{\partial \mathbf{w}_j}u_j(t) \end{array}\right]\right) \quad (20)$$

$$= -2\mathbf{e}^T(t)\left(\left[\begin{array}{c} \psi_{1j}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{w}_j\boldsymbol{\phi}(t-1)}}{(e^{k\mathbf{w}_j\boldsymbol{\phi}(t-1)}+1)^2} \\ \vdots \\ \psi_{mj}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{w}_j\boldsymbol{\phi}(t-1)}}{(e^{k\mathbf{w}_j\boldsymbol{\phi}(t-1)}+1)^2} \end{array}\right] + \left[\begin{array}{c} d_{1j}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{w}_j\boldsymbol{\phi}(t)}}{(e^{k\mathbf{w}_j\boldsymbol{\phi}(t)}+1)^2} \\ \vdots \\ d_{mj}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{w}_j\boldsymbol{\phi}(t)}}{(e^{k\mathbf{w}_j\boldsymbol{\phi}(t)}+1)^2} \end{array}\right]\right) \quad (21)$$

Equation (24) on the next page gives the final weight update equation for MIMO nonlinear time-invariant systems, where $e_l(t)$ corresponds to error at the l^{th} output, ψ_{lj} is the element at the l^{th} row and j^{th} column of the matrix $\mathbf{\Psi}$, η is the learning rate of the RBF neural network, $\mathbf{w}_j$ is the vector for the weights of j^{th} RBF output, m is the number of outputs of the process, and $\boldsymbol{\phi}(t)$ is the basis function vector.

4 Simulation Results

A two input two output time-varying Hammerstein type nonlinear system is considered.

$$\mathbf{x}(t+1) = \left[\begin{array}{ccc} -0.0096 & -0.133 & -0.0692+e^{-t} \\ -0.082+e^{-t} & -0.0714+e^{-t} & 0.0852 \\ 0.0294 & 0.1624+e^{-t} & 0.1254 \end{array}\right]\mathbf{x}(t) + \left[\begin{array}{cc} 4.8+e^{-t} & 0.5+e^{-t} \\ 1.93 & 0.8 \\ 1.21 & 1 \end{array}\right]\mathbf{v}(t),$$

$$\mathbf{y}(t) = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right]\mathbf{x}(t),$$

where the input $\mathbf{v}(t)$ to the linear subsystem is a nonlinear function of the input $\mathbf{u}(t)$, and is given by

$$\left[\begin{array}{c} v_1(t) \\ v_2(t) \end{array}\right] = \left[\begin{array}{c} u_1(t)u_2^2(t) \\ tan[u_2(t)] \end{array}\right]$$

The control objective is to keep the two outputs at a desired reference trajectory. An RBF network with only six neurons is initialized with random synaptic weights. Centers are chosen around the desired setpoints and are fixed. A learning rate of 10^{-3} is chosen. Spread of the Gaussian function is chosen as the mean value of distance between consecutive centers. A control signal constraint of ± 2 is enforced, hence, α is taken as 2 and k is taken as 1. These parameters provide excellent trajectory tracking. Figures 3 and 4 show output tracking of the two outputs. The tracking error converges to a measure of 10^{-25}. Tracking error is shown in figure 5 while control signals are shown in figures 6 and 7. As seen from the figures, the controller has damped the system sufficiently and there are very small overshoots in the transient response. It can be seen from figures 6 and 7 that the conroller is abiding by the constraints on the control signal and the value of $|u_{1,2}(t)| \leq 2$.

4.1 Performance in the presence of noise

To analyze the performance of the controller in the presence of noise, normally distributed additive random noise with a variance of 0.01 is added at the output. The results are shown in figures 8, 9, 10, 11 and 12, and the controller performs well in noisy environment as well. Control signals abide by the constraint as before.

4.2 Robustness

The measure of robustness of a controller can be gauged from its performance keeping in mind the fact that the controlled system is constantly changing. A robust controller will quickly adapt to changes in system parameters. The controller proves to be robust enough to control the system without needing an update in the linearized estimate of the system required in Self Tuning Regulators (STR) and other adaptive control methods. To test the performance of the controller for greater variation in system parameters, the system given in the above example is modified such that the rate of decay of the exponential term in the system is decreased. This can be translated into the fact that variation in system parameters is increased

$$\frac{\partial J}{\partial \mathbf{w}_j} = -2[e_1(t)\cdots e_m(t)] \begin{bmatrix} \psi_{1j}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t-1)}}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t-1)}}+1)^2} + d_{1j}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t)}}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t)}}+1)^2} \\ \vdots \\ \psi_{mj}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t-1)}}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t-1)}}+1)^2} + d_{mj}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t)}}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}_{(t)}}+1)^2} \end{bmatrix} \tag{22}$$

$$= 2e_1(t)\left(\psi_{1j}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{W}_j\boldsymbol{\phi}(t-1)}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}(t-1)}+1)^2} + d_{1j}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{W}_j\boldsymbol{\phi}(t)}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}(t)}+1)^2}\right)$$

$$\cdots - 2e_m(t)\left(\psi_{mj}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{W}_j\boldsymbol{\phi}(t-1)}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}(t-1)}+1)^2} + d_{mj}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{W}_j\boldsymbol{\phi}(t)}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}(t)}+1)^2}\right). \tag{23}$$

Finally, the weight update equation for j^{th} control signal $u_j(t)$ becomes

$$\mathbf{w}_j(k+1) = \mathbf{w}_j(k) + 2\eta\sum_{l=1}^{m} e_l(t)\left(\psi_{lj}\alpha\frac{2k\boldsymbol{\phi}(t-1)e^{k\mathbf{W}_j\boldsymbol{\phi}(t-1)}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}(t-1)}+1)^2} + d_{lj}\alpha\frac{2k\boldsymbol{\phi}(t)e^{k\mathbf{W}_j\boldsymbol{\phi}(t)}}{(e^{k\mathbf{W}_j\boldsymbol{\phi}(t)}+1)^2}\right). \tag{24}$$

for a longer period of time and the system will take longer to cancel the effect of the exponential terms. The modified system is given as

$$\mathbf{x}(t+1) = \begin{bmatrix} -0.0096 & -0.133 & -0.0692+e^{\frac{-t}{2.5}} \\ -0.082+e^{\frac{-t}{2.5}} & -0.0714+e^{\frac{-t}{2.5}} & 0.0852 \\ 0.0294 & 0.1624+e^{\frac{-t}{2.5}} & 0.1254 \end{bmatrix}$$

$$\mathbf{x}(t) + \begin{bmatrix} 4.8+e^{\frac{-t}{2.5}} & 0.5+e^{\frac{-t}{2.5}} \\ 1.93 & 0.8 \\ 1.21 & 1 \end{bmatrix}\mathbf{v}(t),$$

$$\mathbf{y}(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\mathbf{x}(t).$$

The nonlinearity at the input is kept unchanged. The modified system will have more time variation for a longer period of time as compared to the unmodified system. Hence, controller performance for smaller values of time t will be crucial. This is done to gauge the performance of the controller in the presence of greater variation in system parameters. The proposed controller is used to control the modified system. Constraints on the control signal, number of neurons, spread of the Gaussian function are all kept unchanged. The performance of the controller can be seen from figures 13 and 14. Control efforts and convergence of error can be seen in figures 15, 16, and 17 respectively.

4.3 Response to disturbance

A step disturbance of magnitude 2 is introduced from discrete time $200 \leq t \leq 208$. The response of the controller is shown in figures 18 and 19. The controller performs well and steers the system output back to the desired output.

5 Final Remarks

The proposed neural network controller provides excellent control results for trajectory tracking of nonlinear time-varying multivariable systems and abides by constraints on the control signal or the output. The proposed controller forces the system output to converge to the reference trajectory. The derived adaptation algorithm provides excellent learning for the neural network. It is noteworthy that the developed controller is robust enough to control the system without requiring an update in the initial linearized estimate of the system, as required in Self Tuning Regulators (STR) and other adaptive control methods. Besides, although it is in the knowledge of the authors that linear time-invariant systems are not a subject of discussion in this paper, it is noteworthy that when dealing with linear time-invariant systems, most adaptive controllers require a condition of stability on the zero dynamics of the system. This condition, known as the

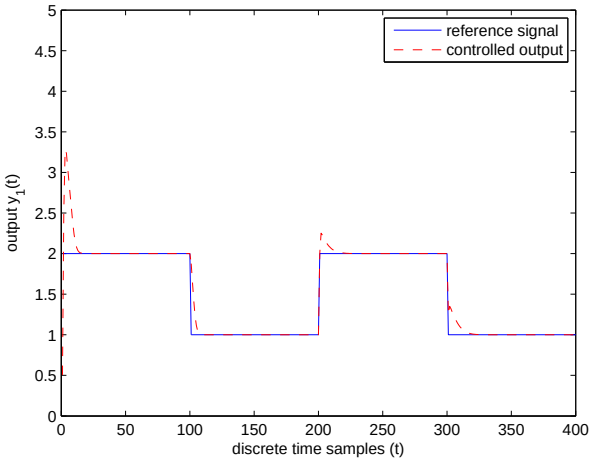

Figure 3: Output trajectory tracking (first output $y_1(t)$).

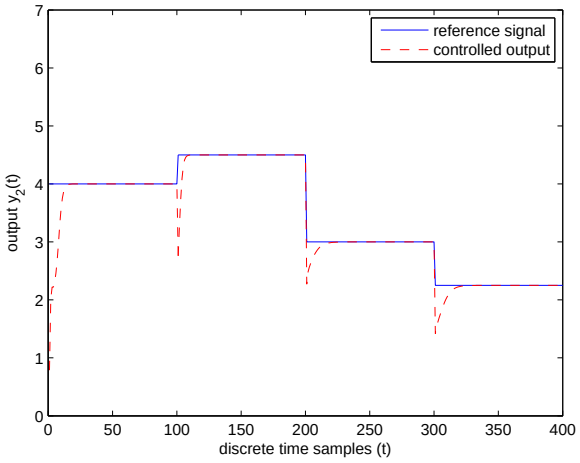

Figure 4: Output trajectory tracking (second output $y_2(t)$).

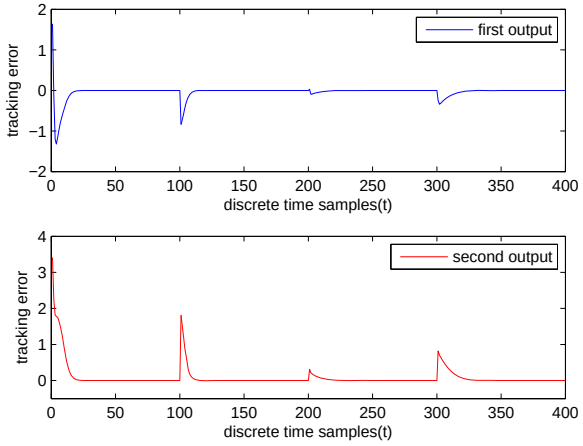

Figure 5: Output trajectory tracking error.

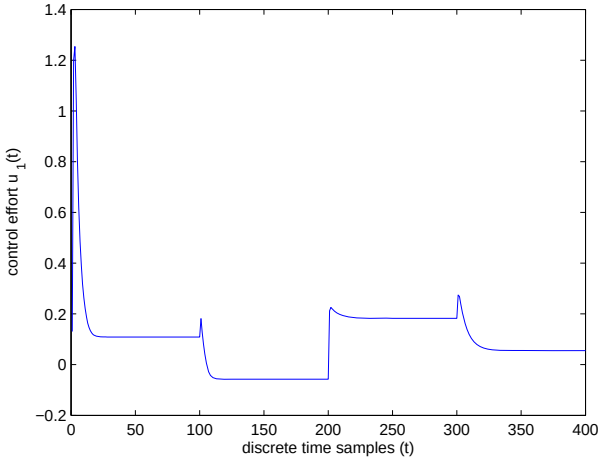

Figure 6: Control effort $u_1(t)$.

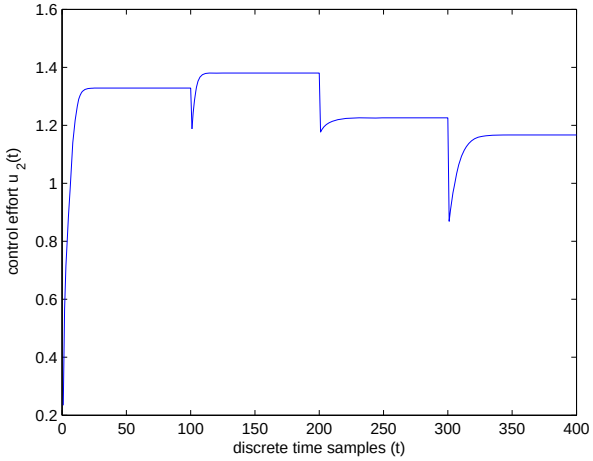

Figure 7: Control effort $u_2(t)$.

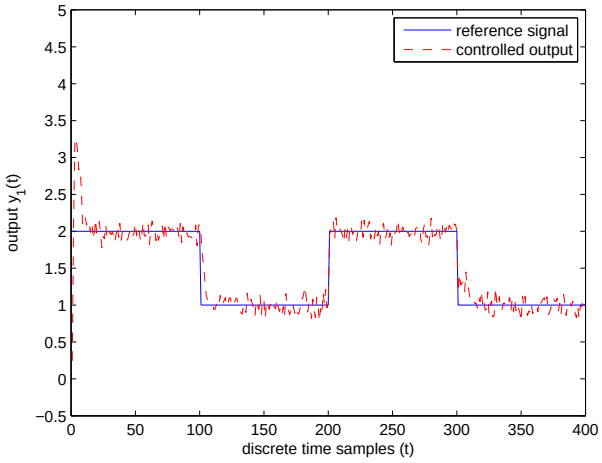

Figure 8: Output trajectory tracking in the presence of noise (first output $y_1(t)$).

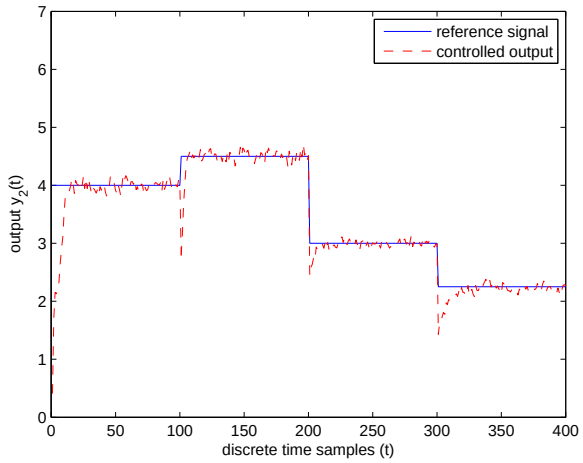

Figure 9: Output trajectory tracking in the presence of noise (second output $y_2(t)$).

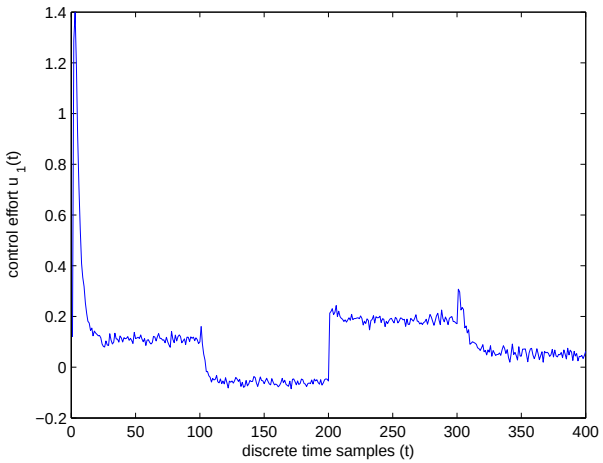

Figure 10: Control effort $u_1(t)$ in the presence of output noise.

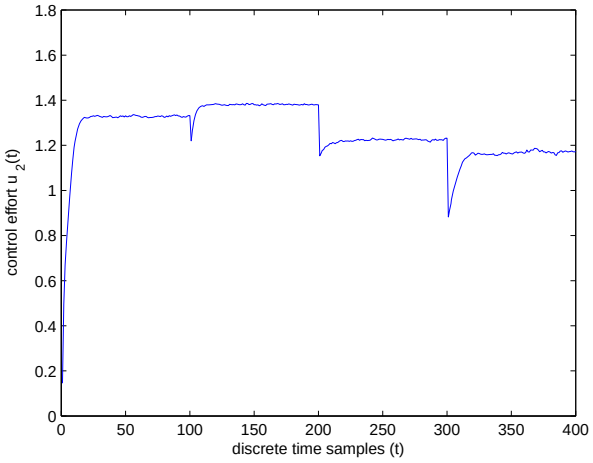

Figure 11: Control effort $u_2(t)$ in the presence of output noise.

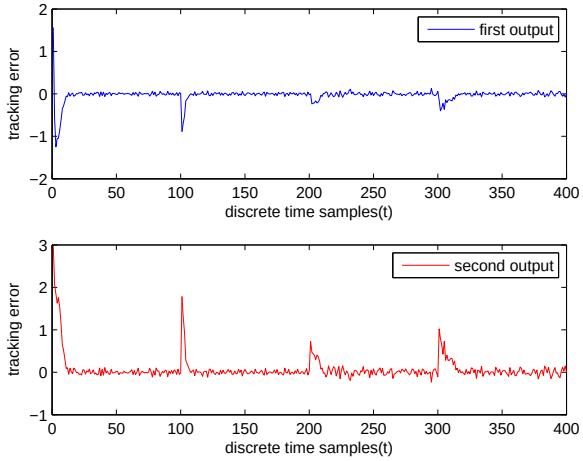

Figure 12: Trajectory tracking error in noisy environment.

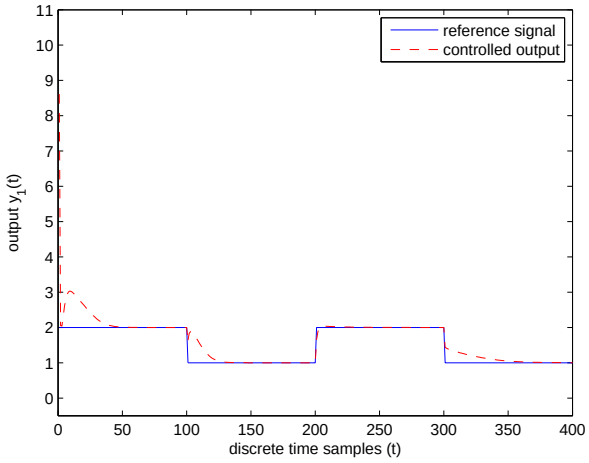

Figure 13: Output trajectory tracking for modified system with increased parameter variation (first output $y_1(t)$).

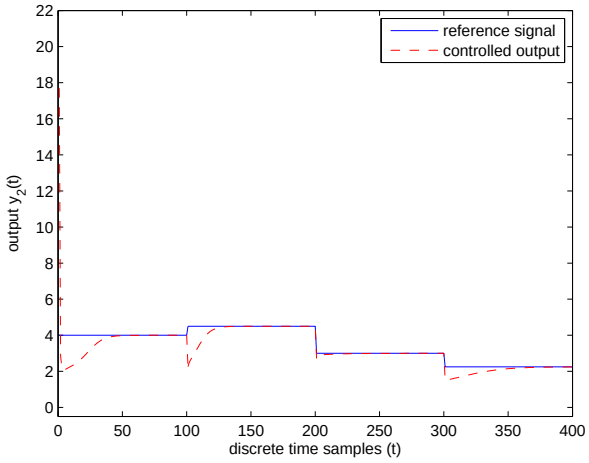

Figure 14: Output trajectory tracking for modified system with increased parameter variation (second output $y_2(t)$).

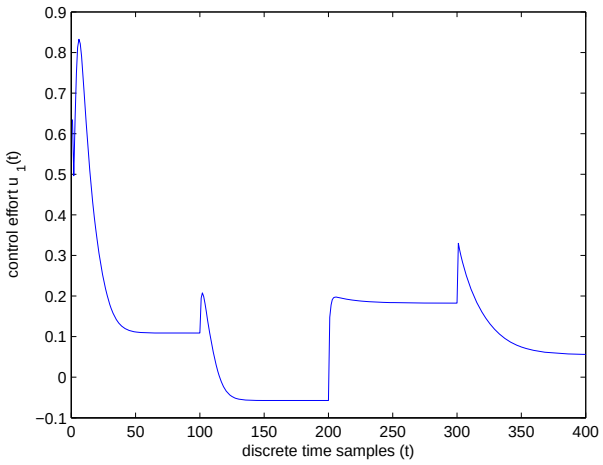

Figure 15: Control effort $u_1(t)$ for modified system with increased parameter variation.

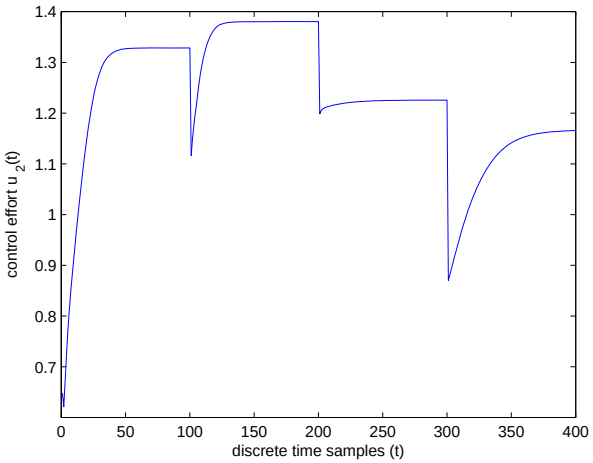

Figure 16: Control effort $u_2(t)$ for modified system with increased parameter variation.

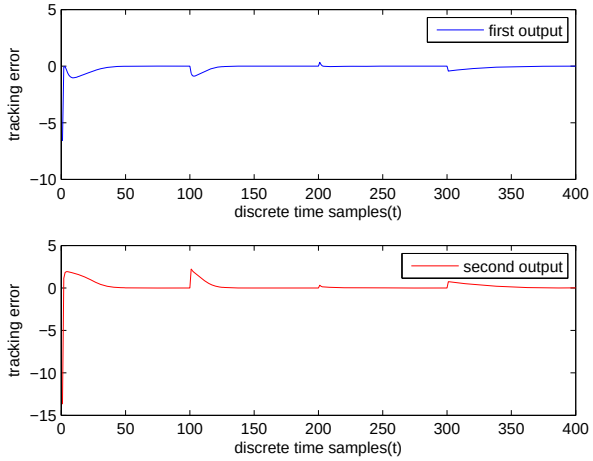

Figure 17: Output trajectory tracking error for modified system with increased parameter variation.

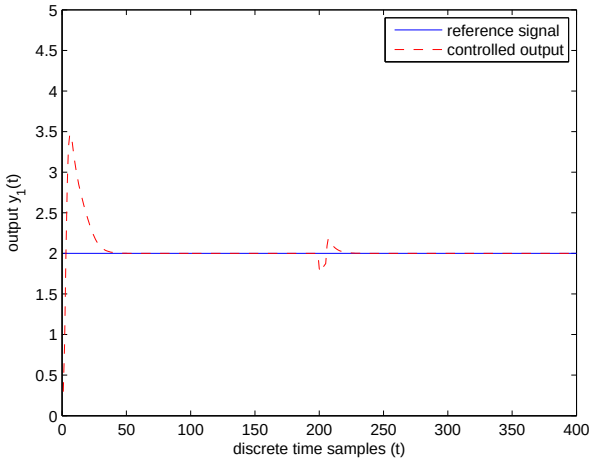

Figure 18: Effect of disturbance on output $y_1(t)$.

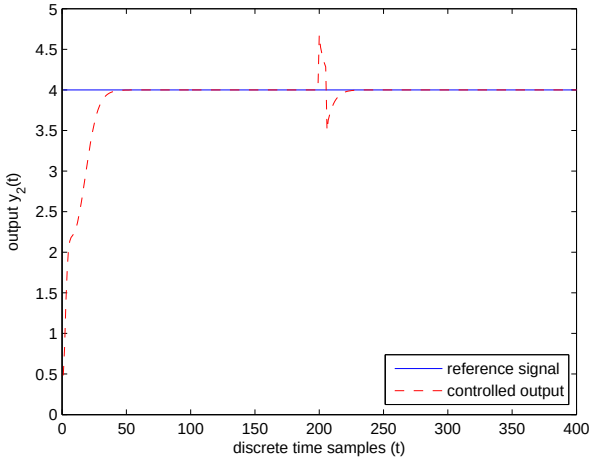

Figure 19: Effect of disturbance on output $y_2(t)$.

condition of minimum-phase system, is also relaxed for the proposed controller if applied to linear time-invariant systems. The universal learning capability of the RBF network makes it ideal for estimating the nonlinearity as well as for compensating the change in system parameters.

Acknowledgments

The authors would like to acknowledge the support of King Fahd University of Petroleum & Minerals, Dhahran, Saudi Arabia.

References:

[1] Haykin, S., *Neural Networks-A Comprehensive Foundation*, Prentice-Hall, 2nd Edition, 1999.

[2] Yahagi, T., "Theory of Digital Signal Signal Processing," Corona, Tokyo, 1985.

[3] Isermann, R., Lachmann, K.H., and Matko, D., "Adaptive Control Systems," Prentice-Hall, U.K., 1994.

[4] Schwarz, H., "Changing the unstable zero dynamics of nonlinear systems via parallel compensation," UKACC International Conference on Control '96, vol. 2, pp. 1226-1231, 1996.

[5] Chen, L., and Narendra, K.S., "Nonlinear adaptive control using neural networks and multiple models," *Automatica*, 2001, vol. 37, pp.1245-1255.

[6] Fu, Y., and Chai, T., "Nonlinear multivariable adaptive control using multiple models and neural networks," *Automatica*, 2007, vol. 43, pp. 1101-1110.

[7] Wang, D., Huang, J., "Adaptive neural network control for a class of uncertain nonlinear systems in pure-feedback form," *Automatica*, 2002, vol. 38, pp. 1365-1372.

[8] Hayakawa, T., Haddad, W., Hovakimyan, N., and Chellaboina V., "Neural Network Adaptive Control for Nonlinear Nonnegative Dynamical Systems," *IEEE Trans. Neural Networks*, 2005, vol. 16(2), pp. 399-413.

[9] Belkheiri, M., and Boudjema, F., "Function approximation based augmented backstepping control for an induction machine," *WSEAS Transactions on Systems and Control*, vol. 2(9), 2007, pp. 450-457.

[10] Petre, E., Selisteanu, D., and Sendrescu, D., "Neural Networks based Adaptive Control for a class of Time Varying Nonlinear Processes," *IEEE Int'l Conf. on Control, Automation and Systems*, 2008, Korea, p..1355-1360.

[11] Zhou, J., Er, M.J., and Zurada, J.M., "Adaptive neural network control of uncertain nonlinear systems with nonsmooth actuator nonlinearities," *Neurocomputing*, 2007, vol. 70, pp. 1062-1070.

[12] Lu, J., and Yahagi, T., "Application of Neural Networks to Nonlinear Adaptive Control Systems," *Proc. of IEEE-ICSP*, 2000, pp. 252-257.

[13] Hu, T., Zhu, J., Hu, C., and Sun, Z., "Neural Networks Direct Adaptive Control for a Class of MIMO Uncertain Nonlinear Systems," *IEEE Mid-summerWorkshop on Soft Computing in Industrial Applications*, 2005, Helsinki, Finland.

[14] Tiponut, V., Gacsadi, A., Caleanu, C., Gavrilut, I., "Neural network guided robot collectivity: an experimental setup," *Proceedings of the 7th WSEAS International Conference on Neural Networks*, 2006, Cavtat, Croatia, pp.41-46.

[15] Dinh, B.H., Dunnigan, M.W., and Reay, D.S., "A practical approach for position control of a robotic manipulator using a radial basis function network and a simple vision system," *WSEAS Transactions on Systems and Control*, 2008, vol. 3(4), pp. 289-298.

[16] Bayar, G., Konukseven, E.I., and Koku, A.B., "Control of a differentially driven mobile robot using radial basis function based neural networks," *WSEAS Transactions on Systems and Control*, 2008, vol. 3(12), pp. 1002-1013.

[17] Al-Duwaish, H., "Neural Networks Controller for Time-Varying Systems," *Proceedings of the 12th WSEAS Conf. on Automatic Control, Modeling, and Simulation, ACMOS 2010, Catania, Italy*, 2010, pp. 99-103.