Intelligent Control and Automation, **2011, 2, 176-181**
doi:10.4236/ica.2011.23021 Published Online August 2011 (http://www.SciRP.org/journal/ica)

Identification and Adaptive Control of Dynamic Nonlinear Systems Using Sigmoid Diagonal Recurrent Neural Network

Tarek Aboueldahab[1], Mahumod Fakhreldin[2]
[1]*Cairo Metro Company, Ministry of Transport, Cairo, Egypt*
[2]*Computers and Systems Department, Electronic Research Institute, Cairo, Egypt*
E-mail: *heshoaboda@hotmail.com, mafakhr@mcit.gov.eg*
Received April 28, 2011; *revised May* 12, 2011; *accepted May* 19, 2011

Abstract

The goal of this paper is to introduce a new neural network architecture called Sigmoid Diagonal Recurrent Neural Network (SDRNN) to be used in the adaptive control of nonlinear dynamical systems. This is done by adding a sigmoid weight victor in the hidden layer neurons to adapt of the shape of the sigmoid function making their outputs not restricted to the sigmoid function output. Also, we introduce a dynamic back propagation learning algorithm to train the new proposed network parameters. The simulation results showed that the (SDRNN) is more efficient and accurate than the DRNN in both the identification and adaptive control of nonlinear dynamical systems.

Keywords: Sigmoid Diagonal Recurrent Neural Networks, Dynamic Back Propagation, Dynamic Nonlinear Systems, Adaptive Control

1. Introduction

The remarkable learning capability of neural networks is leading to their wide application in identification and adaptive control of nonlinear dynamical systems [1,2-5,6] and the tracking accuracy depends on neural networks structure, which should be chosen properly [7-13].

Feedforward Neural Network (FNN) [4] is a static mapping and can not reflect the dynamics of the nonlinear systems without using Tapped Delay Lines (TDL) [7,9]. Fully connected Recurrent Neural Network (RNN) [9,14,15] contains interlink between neurons to reflect this dynamics but it suffers both structure complexity and the poor performance accuracy [9,15]. Based on Locally Recurrent Globally Feedforward network architectures (LRGF) many researchers focused in (DRNN) which doesn't contain interlink between hidden layer neurons leading to the network structure complexity reduction [15,8-10].

However, in all these architectures, the hidden layer neurons output restricted to the sigmoid function output which represents a major disadvantage in the network behavior and significantly reduces its performance accuracy. Therefore, a new architecture called Sigmoid Diagonal Recurrent Neural Network (SDRNN) based on the hidden layer sigmoid weight and its associated dynamic back propagation learning algorithm is proposed. Simulation results show that SDRNN is more suited than the (DRNN) for identification and adaptive control of nonlinear dynamical systems [9].

This paper is organized as follows: Section II presents some background concerning application of neural networks in adaptive control, Section III introduce the new architecture and its associated dynamical learning algorithm for adaptation of the sigmoid weight. Simulation results are shown in Section IV, and finally, conclusion and future work is shown in Section V.

2. Neural Network in Nonlinear System Identification and Control

In the identification stage of the adaptive control of nonlinear dynamical system, a neural network identifier model for the system to be controlled is developed. Then, this identifier is used to represent the system while training the neural network controller weights in the control

stage [6,8,10,12].

2.1. Identification

If a set of data (measurements) can be carried out on a nonlinear dynamic system, an identifier could be derived whose dynamic behavior should be as close as possible to this system. The identifier model is selected based on whether all the system states or only its output are measured [1,3,4,9].

The state space representation of a nonlinear system is given by the following equation:

$$x(k+1)=\Phi\left(x(k),u(k)\right) \text{ and } y(k)=\Gamma\left(x(k),u(k)\right) \quad (1)$$

where: $u(k)\in R^m$ the input to the system, $x(k)\in R^n$ the states of the system, $y(k)\in R^p$ is the outputs of the system, the nonlinear mappings functions $\Phi: R^n\times R^m\to R^n$ and $\Gamma: R^n\to R^p$ are dynamic and smooth [4,9].

Usually, all the system states are not measured for process representation, so the Input/Output representation which is also called Nonlinear Autoregressive Moving Average (NARMA) representation given by the following equation is used instead [4,6,9]

$$y(k+d)=\psi\left(y(k),y(k-1),\cdots,y(k-n+1),\right.$$
$$\left.u(k),\cdots,u(k-m+1)\right) \quad (2)$$

where d is the relative degree (or equivalent delay) of the system and it is assumed that both the order of the system n and relative degree are specified while the nonlinear function $\Psi(\cdot)$ is unknown.

In general, any discrete time dynamical nonlinear system can be represented using NARMA—Output Error Model representation shown in **Figure 1** and has the following forms:

$$y_m(k+d)=\hat{\Psi}\left(y_m(k),y_m(k-1),\cdots,\right.$$
$$y_m(k-n+1),u(k),u(k-1),\cdots,$$
$$\left.u(k-m+1),W_\Psi(k)\right) \quad (3)$$

It is crucial to note that, the present identifier output $y_m(k)$ represented by the nonlinear mapping function $\overset{\Lambda}{\psi}(\cdot)$ is a dynamic mapping function because it depends on its past outputs. Thus, the feed forward neural network can not be used to represent this mapping function and the recurrent neural network which is a dynamic non linear mapping is used instead. [9].

2.2. Control

According to both inverse function theorem and implicit function theorem [4,5,9], if the state vector $x(k)$ of the nonlinear system given by equation (1) is accessible, then the system output $y(k)$ can make exact tracking of a general output $y^*(k)$ using a control low given by the following equation

$$u(k)=\varphi\left(x(k),y^*(k+d)\right) \quad (4)$$

If the nonlinear dynamical system given by equations (1-2) is observable, then the control law $u(k)$ can also be represented in terms of its past values: $u(k-1),\cdots u(k-m+1)$, as well as previous dynamic system outputs $y(k),\cdots,y(k-n+1)$ and $y^*(k+d)$. If $r^*(k)$ represents the information that is needed at the instant k to implement the control law (*i.e.*, $r^*(k)=y^*(k+d)$) so the above equation can be re-written as follows [1,3-5,9]:

$$u(k)=\varphi\left(y(k),y(k-1),\cdots,y(k-n+1),\cdots,\right.$$
$$\left.u(k),u(k-1),\cdots,u(k-m+1),r^*(k)\right) \quad (5)$$

As this control law is represented by the nonlinear dynamical mapping function $\varphi(\cdot)$, so a neural network controller model represented by the nonlinear dynamical mapping function $\varphi(\cdot)$ can be used to achieve the control law and can be written as:

$$u(k)=\overset{\Lambda}{\varphi}\left(y(k),y(k-1),\cdots,y(k-n+1),\cdots,u(k),\right.$$
$$\left.u(k-1),\cdots,u(k-m+1),r^*(k),W_\varphi(k)\right) \quad (6)$$

and $W_\varphi(k)$ is the set of parameters of the neural network controller.

As the neural network controller output $u(k)$ depends on its past outputs, thus, the recurrent neural network controller is used to evaluate the controller parameters [9].

The structure of the closed loop nonlinear predictive controller consisting of the nonlinear system, the nonlinear identifier and the nonlinear controller is shown in **Figure 2.**

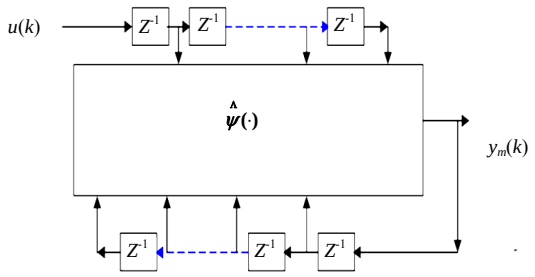

Figure 1. NARMA—output error model.

3. The Sigmoid Diagonal Recurrent Neural Network(SDRNN)

In this section, our proposed architecture and its associ-

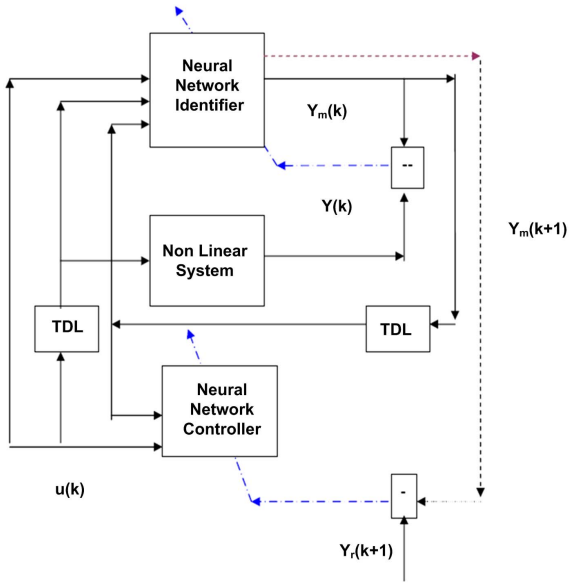

Figure 2. The structure of the closed loop nonlinear predictive controller.

ated modified dynamic back propagation learning algorithm to learn the new added sigmoid weight vector are presented

Define the following to obtain the mathematical model of the proposed neural network architecture:

n_i , n_h , n_o are the number of neurons in input, hidden, and output layers respectively.

W^I is the input weight matrix connecting between input layer and the hidden layer, W^S , W^D are the sigmoid weight vector and the diagonal weight vector of the hidden layer, and W^O is the output weight matrix connecting between hidden layer and the output layer

Assume, at sample k, the input to the i^{th} input neuron in the input layer is $I(k)$, so the output of the neural network can be calculated as follows:

The net input to the j^{th} sigmoid neuron in the hidden layer can be calculated as follows

$$Hin_j(k) = W_j^D \times H_j(k-1) + \sum_{i=1}^{n_i} W_{ij}^I \times I_i(k) \qquad (7)$$

And it's output can be calculated as

$$H_j(k) = \frac{f\left(Hin_j(k) \times W_j^s\right)}{W_j^s} \qquad (8)$$

The output of the m^{th} neuron in the output layer can be written as follows

$$Y_m(k) = \sum_{j=1}^{n_h} W_{jm}^o \times H_j(k) \qquad (9)$$

For the standard Diagonal Recurrent Neural Network, the sigmoid weight vector $W^S(k)$ is set to be one and for the normal Feed Forward Neural Networks the diagonal vector $W^D(k)$ is set to be zero.

Learning Algorithm

Given the structure of the network described by Equations (7)-(9) and applying the gradient descent method [9, 16] to update the network weights the partial derivatives of the neural network predictor output $Y_m(k)$ with respect to network weights are given by

$$\frac{\partial Y_m(k)}{\partial W_{jm}^O} = H_j(k) \qquad (10a)$$

$$\frac{\partial Y_m(k)}{\partial W_j^S} = \frac{W_{jm}^O}{W_J^S} \times \beta(k) \qquad (10b)$$

$$\frac{\partial Y_m(k)}{\partial W_j^D} = W_{jm}^O \times P_j(k) \qquad (10c)$$

$$\frac{\partial Y_m(k)}{\partial W_{ij}^I} = W_{jm}^O \times Q_{ij}(k) \qquad (10d)$$

where:

$$\beta(k) = \frac{\partial f\left(Hin_j(k) \times W_j^S\right)}{\partial W_j^S} - \frac{f\left(Hin_j(k) \times W_j^S\right)}{W_j^S}$$

$$P_j(k) = \mu_j(k) \times \sum_{j=1}^{n_h} \left[H_j(k-1) + W_{P_j}^D \times_{P_j} (k-1) \right] \qquad (11a)$$

$$Q_{ij}(k) = \mu_j(k) \times \left[I_i(k) + \sum_{j=1}^{n_h} W_j^D \times Q_{ij}(k-C) \right] \qquad (11b)$$

and

$$\mu_j(k) = \frac{1}{W_j^s} \times \frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial Hin_j(k)},$$

$$P_j(0) = 0, \quad Q_{ij}(0) = 0$$

From Equation (9), the partial derivatives of the neural network predictor output $Y_m(k)$ represented by the nonlinear mapping function $\psi\left(I(k), W^{\psi}\right)$ with respect to output weight matrix W_{jm}^O is:

$$\frac{\partial Y_m(k)}{\partial W_{jm}^O} = H_j(k)$$

which lead to (10a).

Also, the partial derivative with respect to sigmoid weight vector W_j^S is

$$\frac{\partial Y_m(k)}{\partial W_j^s} = W_{jm}^O \times \frac{\partial H_j(k)}{\partial W_j^s}$$

And from Equations (7) and (8),

$$\frac{\partial H_j(k)}{\partial W_j^s} = \frac{1}{W_j^s} \times \frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial W_j^s} - f\left(Hin_j(k) \times W_j^s\right) \times \frac{\partial}{\partial W_j^s}\left[\frac{1}{W_j^s}\right]$$

$$\frac{\partial H_j(k)}{\partial W_j^s} = \frac{1}{W_j^s} \times \left[\frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial W_j^s} - \frac{f\left(Hin_j(k) \times W_j^s\right)}{W_j^s}\right]$$

which lead to (10b).

From Equation (9), the partial derivatives with respect to diagonal weight vector W_j^D is

$$\frac{\partial Y_m(k)}{\partial W_j^D} = W_{jm}^O \times \frac{\partial H_j(k)}{\partial W_j^D} = W_{jm}^O \times P_j(k)$$

and from Equation (8)

$$\frac{\partial H_j(k)}{\partial W_j^D} = \frac{1}{W_j^s} \times \frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial W_j^D}$$

$$\frac{\partial H_j(k)}{\partial W_j^D} = \frac{1}{W_j^s} \times \frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial Hin_j(k)} \times \frac{\partial Hin_j(k)}{\partial W_j^D}$$

and from Equation (7)

$$\frac{\partial Hin_j(k)}{\partial W_j^D} = H_j(k-1) - W_j^D \times \frac{\partial H_j(k-1)}{\partial W_j^D}$$

which leads to (10c), (11a).

Also, the partial derivative with respect to input weight matrix W_{ij}^I is

$$\frac{\partial Y_m(k)}{\partial W_{ij}^I} = W_{jm}^O \times \frac{\partial H_j(k)}{\partial W_{ij}^I} = W_{jm}^O \times Q_{ij}(k)$$

and from Equation (8)

$$\frac{\partial H_j(k)}{\partial W_{ij}^I} = \frac{1}{W_j^s} \times \frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial W_{ij}^I}$$

$$\frac{\partial H_j(k)}{\partial W_{ij}^I} = \frac{1}{W_j^s} \times \frac{\partial f\left(Hin_j(k) \times W_j^s\right)}{\partial Hin_j(k)} \times \frac{\partial Hin_j(k)}{\partial W_{ij}^I}$$

and from Equation (7)

$$\frac{\partial Hin_j(k)}{\partial W_{ij}^I} = I_i(k) - W_j^D \times \frac{\partial H_j(k-1)}{\partial W_{ij}^I}$$

which leads to (10d), (11b).

The full proof of this lemma is given in details in reference number [9].

4. Results and Discussion

In this section, extensive experimentation is carried out in an attempt to demonstrate the performance of the SDRNN architecture and compare it to the DRNN architecture in nonlinear system identification and in adaptive control of nonlinear dynamical systems. As a measure to test the performance, we use the Mean Square Error (MSE) criteria between the actual nonlinear system output and the neural network output [8-10]. It is worth mentioning that in our proposed network, there is no need to use momentum term or learning rate adaptation [8, 10] because the training is done using the adaptation of the sigmoid function shape leading to reduction of the learning algorithm complexity [9].

Example 1: (Nonlinear system identification)

A benchmark problem is employed, the identification of a dynamical system. The example is taken from [7,9], where the nonlinear system to be identified is governed by the following difference equation

$$Y(k+1) = \frac{Y(k) \times Y(k-1) \times Y(k-2) \times u(k-1) \times \left[Y(k-2)-1\right]}{1+Y^2(k-1)+Y^2(k-2)} + \frac{u(k)}{1+Y^2(k-1)+Y^2(k-2)} \quad (12)$$

As it can be seen, the current output of the plant $Y(k+1)$ depends on three previous outputs $Y(k), Y(k-1), Y(k-2)$ and two previous inputs $u(k), u(k-1)$. A NARMA-Output Error Model given by Equation (3) is considered for identification of this nonlinear dynamical system. The neural network identifier output $Y_m(k+1)$ is the approximation of the nonlinear system output $Y(k+1)$. Thus the input to the neural network identifier are the three previous identifier outputs *i.e.* $(Y_m(k), Y_m(k-1), Y_m(k-2))$ and the two previous inputs $u(k), u(k-1)$ and the size of the neural network identifier is '5−8−1' (5 input units, 8 hidden units, and one output unit).

In order to comply with previous results reported in the literature, a new training data containing 200 batches of 900 patterns is generated. For each data batch, the input $u(k)$ is an independent and identically distributed uniform in the range between $\{-1, 1\}$ while the testing data set is composed of 1000 samples with a signal described by

$$u(k) = \begin{cases} \sin(\pi \times k/25) & k < 250 \\ 1 & 250 < k < 500 \\ -1 & 500 < k < 750 \\ 0.3 \times \sin(\pi \times k/25) + 0.1 \times \sin(\pi \times k/32) & \\ +0.6 \times \sin(\pi \times k/10) & 750 < k < 1000 \end{cases} \quad (13)$$

As a measure to test the identification performance,

the MSE using the standard DRNN is 0.0017 while using our proposed SDRNN, it is reduced to be 1.1049e-004. The nonlinear system output, the SDRNN identifier output and the DRNN identifier output are shown in **Figure 3.**

The solution of the problem without using the sigmoid weight vector as similar to the neural network identifier schemes in [2, 8-10], the system performance is very poor because the restriction to the output hidden layer neurons. While using our proposed architecture, this restriction is avoided due to the adaptation of the sigmoid function shape.

Example 2: (Rigid non-minimum phase model) [7,12,13]

The nonlinear system is given by the following difference equation

$$Y(k)=\frac{Y(k-1)}{1+Y^2(k-1)}+u(k-1)+5\times u(k-2) \qquad (14)$$

And the reference model is giving by the following dynamical difference equation [7,10,12]:

$$r(k)=\sin(2\times\pi\times k/25)+\sin(2\times\pi\times k/10)$$

and (15)

$$Y_r(k)=0.6\times Y_r(k-1)+r(k)$$

It can be seen that the current system output $Y(k+1)$ depends on its previous output $Y(k)$ and two previous inputs $u(k)$ and $u(k-1)$. Thus, the NARMA-Output Error Model given by Equation (3) is considered for system identification and the neural network identifier inputs are the previous identifier output $Y_m(k)$ and the two previous inputs and the size of the neural network identifier is 3-6-1' (3 input units, 6 hidden units, and one output unit).

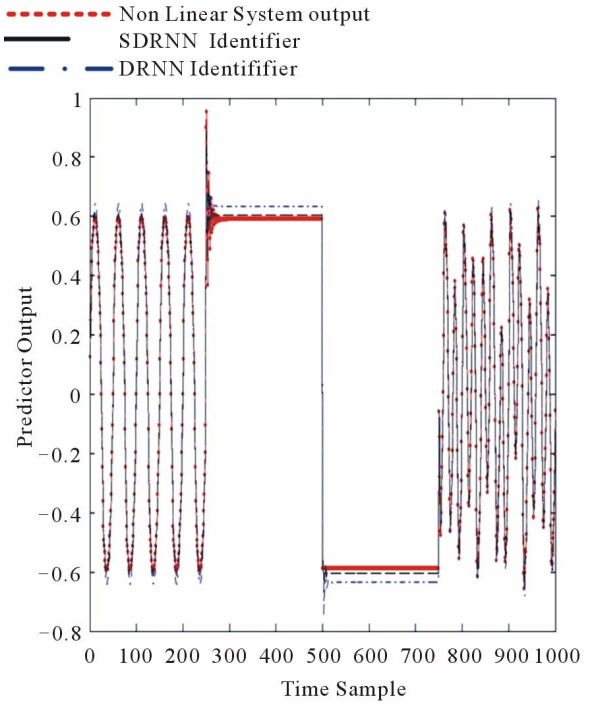

Figure 3. The nonlinear system output, the SDRNN identifier output and the DRNN identifier output.

Consequently, in the nonlinear adaptive control phase, the current neural network controller output $u(k)$ depends on it's previous output $u(k-1)$, reference model input $r(k)$ and reference output $Y_r(k)$ beside the previous neural network identifier output $Y_m(k)$. A NARMA-Output Error Model given by equation (3) is considered for the neural network controller and the size of this controller is 4-7-1' (4 input units, 7 hidden units, and one output unit).

After 10000 iterations for training identifier weights with uniformly distributed random signal, both identifier and controller start closed-loop control. The MSE for 100 samples using DRNN is 0.3127 while using our proposed SDRNN it enhances to be 0.0141.

The final diagonal and sigmoid weights in both the standard network controller and the proposed controller are shown in **Table 1.**

The reference model output, the SDRNN controller output and the DRNN controller output are shown in **Figure 4**.

From **Table 1** and **Figure 4**, it is obvious that the values of the diagonal weights in the DRNN controller is located within a wide range between 16.4877 (neuron number 4) and -45.6087 (neuron number 7) while in the SDRNN it is located is located in a narrow range between 0.2047 (neuron number 5) and -2.7992 (neuron number 6). This great difference is due to the existence of the sigmoid weight vector adapting the shape of the sigmoid function enabling the neural network controller output to get the appropriate values that can efficiently reduce the MSE between the actual reference t model and the nonlinear system output.

Table 1. Diagonal and sigmoid weights of neural network controllers.

Hidden Layer Neuron Number	Diagonal Weight		Sigmoid Weight
	Standard Network	Proposed Network	Proposed Network
1	–6.0076	0.1645	-0.0002
2	–10.7195	-1.9830	0.6095
3	–20.0180	-0.9809	0.0007
4	16.4877	-0.3553	0.0005
5	–19.3239	0.2047	0.1742
6	–15.1893	-2.7992	1.2101
7	–45.6087	-0.1767	0.7825

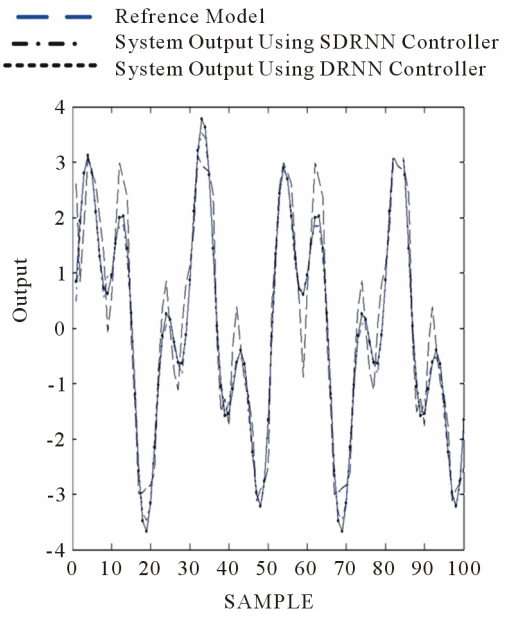

Figure 4. Model output, nonlinear dynamical system output using standard neural network architecture and using proposed architecture.

5. Conclusions and Future Work

We have presented a new neural network architecture called based on the adaptation of the shape of the sigmoid weight of the hidden layer neurons and have introduced its corresponding dynamic back propagation learning algorithm. This architecture is applied in both identification and adaptive control of nonlinear dynamical systems and gives better results than the standard DRNN For the future work, it is suggested that this architecture will be extended to be used in multivariable nonlinear system identification and adaptive control as well as other practical neural networks applications such as pattern recognition and time series prediction

6. References

[1] A. U. Levin, and K. S. Narendra, "Control of Nonlinear Dynamical Systems Using Neural Networks—Part II: Observability, Identification and Control," *IEEE Transactions on Neural Networks*, Vol. 7, No. 1, 1996, pp. 30-42. doi:10.1109/72.478390

[2] C. C. Ku and K. Y. Lee, "Diagonal Recurrent Neural Networks for Dynamic System Control," *IEEE Transactions on Neural Networks*, Vol. 6, No. 1, 1995, pp. 144-156. doi:10.1109/72.363441

[3] G. L. Plett, "Adaptive Inverse Control of Linear and Nonlinear Systems Using Dynamic Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 14, No.2, 2003, pp. 360-376. doi:10.1109/TNN.2003.809412

[4] K. S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 1, No. 1, 1990, pp. 4-27. doi:10.1109/72.80202

[5] L. Chen and K. S. Narendra, "Nonlinear Adaptive Control Using Neural Networks and Multiple Models," *Proceedings of the 2000 American Control Conference*, Chicago, 2002, pp. 4199-4203.

[6] R. Zhan and J. Wan "Neural Network-Aided Adaptive Unscented Kalman Filter for Nonlinear State Estimation," *IEEE Signal Processing Letters*, Vol. 13, No. 7, 2006, pp. 445-448. doi:10.1109/LSP.2006.871854

[7] A. S. Poznyak, W. Yu, E. N. Sanchez and J. P. Perez, "Nonlinear Adaptive Trajectory Tracking Using Dynamic Neural Networks," *IEEE Transactions on Neural Networks*, Vol. 10, No. 6, 1999, pp. 1402-1411. doi:10.1109/72.809085

[8] P. A. Mastorocostas, "A Constrained Optimization Algorithm for Training Locally Recurrent Globally Feedforward Neural Networks," *Proceedings of International Joint Conference on Neural Networks*, Montreal, 31 July 4 August 2005, pp.717-722.

[9] A. Tarek, "Improved Design of Nonlinear Controllers Using Recurrent Neural Networks," Master Dissertation, Cairo University, 1997.

[10] Xiang Li, Z. Q. Chen and Z. Z. Yuan, "Simple Recurrent Neural Network-Based Adaptive Predictive Control for Nonilnear Systems," *Asian Journal of Control*, Vol. 4, No. 2, June 2002, pp. 231-239.

[11] N. Kumar , V. Panwar, N. Sukavanam, S. P. Sharma and J. H. Borm, "Neural Network-Based Nonlinear Tracking Control of Kinematically Redundant Robot Manipulators," *Mathematical and Computer Modelling*, Vol. 53, No. 9-10, 2011, pp. 1889-1901. doi:10.1016/j.mcm.2011.01.014

[12] J. Pedro and O. Dahunsi, "Neural Network Based Feedback Linearization Control of a Servo-Hydraulic Vehicle Suspension System," *International Journal of Applied Mathematics and Computer Science*, Vol. 21, No. 1, 2011, pp. 137-147. doi:10.2478/v10006-011-0010-5

[13] A. Thammano and P. Ruxpakawong, "Nonlinear Dynamic System Identification Using Recurrent Neural Network with Multi-Segment Piecewise-Linear Connection Weight," *Memetic Computing*, Vol. 2, No. 4, 2010, pp. 273-282. doi:10.1007/s12293-010-0042-7

[14] A. C Tsoi and A. D. Back, "Locally Recurrent Globally Feedforward Networks: A Critical Review of Architectures," *IEEE Transaction Neural Networks*, Vol. 5, No. 2, 1994, pp. 229-239. doi:10.1109/72.279187

[15] T. Rashid, B. Q. Huang and T. Kechadi, "Auto-Regressive Recurrent Neural Network Approach for Electricity Load Forecasting," *International Journal of Computational Intelligence*, Vol. 3, No. 1, 2007, pp.66-71.

[16] B. A. Pearlmutter, "Gradient Calculations for Dynamic Recurrent Neural Networks: A Survey," *IEEE Transactions on Neural Networks*, Vol. 6, No. 5, 1995, pp. 1212-1228. doi:10.1109/72.410363

