Journal of KSNVE, Vol. 9, No. 4, pp. 849~855, 1999.

〈Original Paper〉

Control Signal Reconstruction of Non-Linear Systems with Noise Using Neural Networks

신경망을 이용한 비선형 잡음계의 제어신호 복원

Younghwan An*
안 영 환

(Received April 30, 1999 : Accepted June 1, 1999)

Key Words : Neural Approximator(신경 근사치), Neural Reconstructor(신경 복원기), Neural Network Simulator(신경망 시뮬레이터)

ABSTRACT

Neural Networks have shown potential to become an attractive alternative to classic methods for identification and control of non-linear dynamic systems. The purpose of this paper is to present an application of neural networks, that is a neural reconstruction of the input signal of a non-linear unknown system. This basic methodology could be used for practical purposes in several engineering fields. Clearly applications of the proposed scheme can be of interest for physical systems where a complete network of sensors measuring system inputs is not available. It should also be emphasized that the application of the reconstruction scheme is of little or no interest when the analyzed system works and operates at nominal conditions. In fact, only when failures and/or system anomalies occur, leading to performance degradation and/or system shutdown, the application of this scheme is of interest. The paper presents the results of the methodology applied to unknown non-linear dynamic systems and the robustness of the scheme to white and colored system noise was evaluated.

요 약

신경망은 비선형 동적 시스템의 식별 및 제어에 대한 기존 방법의 매력적인 대체방법으로서 가능성을 보여주었다. 이 논문의 목적은 신경망의 응용, 즉 미지의 비선형 시스템의 입력 신호에 대한 신경 복원을 제시하고 있다. 이 기본 방법론은 여러 공학분야에서 실질적인 용도로 쓰일 수 있으며, 분명히, 이 제시된 기법의 응용은 시스템 입력을 측정하는 완전한 감지기망이 가능하지 않는 물리적 시스템에 중요할 수 있다. 또한 이 복원기법의 응용은 시스템이 정상적으로 작동할 시에는 중요하지 않지만, 성능저하 또는 시스템 중단을 야기하는 고장 혹은 시스템 이상을 일으킬 시에는 중요한 역할을 할 수 있다. 이 논문에서는 미지의 비선형 동적 시스템에 이 방법을 적용한 결과를 제시하고 있으며, 백색/채색 시스템 잡음에 대한 이 기법의 강인성이 평가되었다.

* 정회원, 인하대학교 기계항공자동화 공학부

1. Introduction

Basic concepts of Neural Network (NN) theory were introduced in the 40's and advances were made in the structure and application of networks throughout 50's, 60's, and 70's. However, it is only in the last decade that applications of NNs to all engineering fields have populated in the technical literature. During the initial stage of this growth NNs have been applied to pattern recognition and classification. Within a later trend, NNs have been proposed for identification and control of dynamic systems of both linear and non-linear nature. A partial list of problems for which the potential of NNs are appealing would include automatic target recognition, speech recognition, pattern identification and/or classification applications as well as the design of fault tolerant systems[1,2].

Recently this attention toward neural networks, due to their capabilities, has perhaps caused a lack of attention toward a clear theoretical analysis and understanding of these capabilities. As a consequence, the alternative use of neural approaches for classic control and/or identification problems has understandably sparkled debates within the technical community. T. Kerr[3] has recently provided a well done, inspiring, and documented critique of the use of neural schemes for control and estimation purposes. However, he also acknowledges that there are a few areas where there has been a good symbiosis between NNs and classic control and estimation theory, mainly the use of Lyapunov functions to establish stability/convergence in neural learning as well as the use of basic or extended Kalman filtering in lieu of the Back Propagation algorithm (BPA) to achieve a substantial level of learning for a NN with a smaller number of iterations.

The purpose of the paper is very precise and consists in proposing a new use of NNs for reconstructing a control signal for non-linear dynamic systems. A typical example of the application of this scheme could be for aircraft crash investigation where anomalies in control surface actuators are believed to be a factor. This reconstruction scheme could provide valuable assistance since most flying commercial aircraft are equipped with flight data recorders which do not record the actual deflections of control surfaces. Other potential uses for the proposed scheme may include investigations of large structure failures, failures for nuclear and chemical plants, malfunctions for complex propulsive systems and, in general, reconstruction of failures and/or malfunctions of systems with complex input/output mapping. The proposed neural scheme is applied to a SISO non-linear system and the effect of the noise on the scheme is also evaluated.

The paper is organized as it follows. The next section reviews basic characteristics of NNs, which makes them attractive for adaptive control and estimation purposes. Another section describes with details the considered problem and shows its application to a non-linear system. A final section concludes the paper.

2. Capabilities of Neural Networks as Approximators

A multi-layer neural network (MNN) has been used as approximators and/or controllers to model and/or control complex non-linear systems due to their approximation and adaptation capabilities. Within a larger picture NNs can be considered as a class of approximators. Other types of approximators include polynomials, spline functions, rational functions, as well as a different class of NNs, known as radial basis function NNs. An important effort toward creating a unified approximation theory which included all these different classes was recently undertaken[4,5]. The critical need for such a unified approach can be understood if one considers the virtually infinite number of degrees of freedom in the selection of the approximation structure. Furthermore, a systematic procedure for creating non-linear approximators as well as stable learning schemes using Lyapunov's theory was introduced.

The purpose of this paper is to suggest a novel application rather than highlighting the performance of a specific class of NN algorithms for approximating a system. In this study, feed-forward MNNs with supervised learning through a gradient-based algorithm are used. Since this specific application involves off-line data processing, on-line learning neural properties toward real-time performance are not of primary concern. Nevertheless, even for off-line training, slow learning, especially for large order systems, and local minimum points are typical problems of the Back-Propagation algorithm (BPA) the most used algorithm for MNNs and its extensions, which include the Recurrent Back-Propagation, the Back-Propagation-Through-Time, and the Dynamic Back-Propagation algorithms.

To try to solve these problems, several alternatives to the BPA have been proposed by many researchers. Most of the efforts have been concentrated on introducing different activation functions or particular procedures for selecting the initial weights. Within this stream of investigations, an approach based on the heterogeneous network has been used in this study. In heterogeneous network each neuron in the hidden and output layer has its own capability of updating some new parameters taking advantage of the increased mapping and adaptation performance. Specifically, in an heterogeneous NN each neuron is able to change its output range (upper and lower bounds: U, L) and the slope of the sigmoid activation function (temperature : T). The non-linear activation function is given by :

$$f(x_{i,j},U_{i,j},L_{i,j},T_{i,j})=\frac{U_{i,j}-L_{i,j}}{1+e^{-x_{i,j}/T_{i,j}}}+L_{i,j} \quad (1)$$

where i, j are the indices for the generic neurons of the hidden and output layers, and 'x' is the argument of the classic sigmoid function of the BPA. This function is implemented at each processing element of the hidden layer(s) and output layer. The gradient is then calculated as in the BPA. The only difference is that the gradient descent is found with respect to each of the independent variables 'x', 'U', 'L', and 'T', as opposed to 'x' only. The associated algorithm is called Extended Back-Propagation algorithm and is shown a substantial performance improvement with respect to the BPA[6].

3. Application to a Non-Linear System and Simulation Results

K. S. Narendra and K. Parthasarathy[7] presented an important study highlighting the applicability of NNs for identification and control of non-linear systems. Consider now a non-linear SISO dynamic system; depending on the origin of the non-linearities it can be modeled using one of the following models:

Model #1

$$y(k+1)=f[y(k),....,y(k-n)]+\sum_{i=0}^{m}b_i u(k-i) \quad (2)$$

Model #2

$$y(k+1)=\sum_{i=0}^{n}a_i y(k-i)+g[u(k),....,u(k-m)] \quad (3)$$

Model #3

$$y(k+1)=f[y(k),...,y(k-n)]+g[u(k),...,u(k-m)] \quad (4)$$

Model #4

$$y(k+1)=f[y(k),....,y(k-n),u(k),....,u(k-m)] \quad (5)$$

It should be recalled that the general systems described by these models are assumed to be BIBO (bounded input, bounded output). There have been a large variety of studies focusing on the observability, and controllability for these non-linear systems. Despite these efforts, solutions to these problems on the same line as the ones for the linear systems have yet to be formulated.

For the purpose of introducing the application of this study, a Model #4-type non-linear SISO system is considered. The objective is to reconstruct $u(.)$ for a given sequence $y(.)$ assuming that the mathematical formulation of $f(.)$ is unknown. The problem is not original and several solutions have been proposed ranging from the Dynamic Inversion approach to different applications of well-known minimization algorithms. Here the proposed methodology consists in providing a neural approximator for the function $f(.)$ followed by a neural reconstruction of $u(.)$ for the given sequence $y(.)$.

Assuming that the non-linear system is stable and BIBO, the neural identifier will also need to be BIBO and stable. Another assumption is that the neural identifier will have to be series-parallel type meaning that the actual system output $y(k)$, rather than its estimate, is used as input by the identification scheme during and after the training. The other type would be a parallel-type identifier, meaning that the estimate of $y(k)$ from the neural identifier is obtained using as input estimates of $y(k)$ at previous time steps. The selection of a series-parallel mode instead of a parallel model is due to the fact that, since the system is assumed to be BIBO and stable, all the input to the NNs are bounded. However, once the neural identifier has been sufficiently trained and the estimation error has converged to a sufficiently low asymptotic value, the series-parallel mode could be replaced by the parallel working mode.

Under these assumptions, consider the following non-linear model #4-type system[7]:

$$y(k+1) = \frac{y(k)y(k-1)y(k-2)u(k-1)[y(k-2)-1]+u(k)}{1+y(k-1)^2+y(k-2)^2} \tag{6}$$

Initially the system is assumed to be unknown and noise-free. However, experimental data for $u(.)$ and $y(.)$ are available for the development of a neural identifier, which for our purposes we call "neural network simulator" (NNS). The output of the NNS is:

$$\hat{y}(k+1) = f[y(k), y(k-1), y(k-2), u(k), u(k-1)] \tag{7}$$

The development of the NNS involves two different phases, that is an initial training phase followed by a testing phase. During the first phase the selected NNS, working in a series-parallel mode, is trained for 300,000 steps with $u(k)$ being a random number uniformly distributed between [-1.5,1.5] leading to $y(k)$ within a [-1.5,1.5] range. Throughout this phase, the learning is monitored by running short simulation tests every 2,000 steps for a total of 150 test points. In other words, the training with the EBPA is halted every 2,000 steps, the associated numerical weights of the NNS are frozen , and a short simulation is conducted (100 steps) - with $u(k)$ still being uniformly distributed between [-1.5,1.5]. During these tests the mean and the variance of the simulation error are evaluated using:

$$M_{SE} = \sum_{i=1}^{N=100} \frac{[\hat{y}(i)-y(i)]}{N} = \sum_{i=1}^{N=100} \frac{e_y(i)}{N} \tag{8}$$

$$V_{SE} = \sum_{i=1}^{N=100} \frac{[e_y(i)-M_{SE}]^2}{N} \tag{9}$$

After each test, the neural learning is resumed allowing the numerical weights of the NNS to be updated using the EBPA. The statistical results for the training of the NNS are shown in Fig. 1, for which the selection of a suitable threshold for the statistical parameters of the estimation error is totally arbitrary. Fig. 2 shows a comparison of the actual system output, $y(.)$, and the output of the NNS (following the 300,000 step training) for a generic 500 step simulation given an arbitrary sequence $u(.)$. It should be emphasized that the training of an accurate NNS is only a necessary, but not sufficient, condition for an accurate reconstruction.

Following the NNS development, the next step consists in the introduction of the reconstruction scheme. Toward this purpose a Neural Network Reconstructor (NNR) is introduced. For the SISO case described above the input to the NNR is given by the NNS output: the NNR output - after I' iterations - is the estimate of the control signal u(k) which, at the time instant k', minimizes the difference between the NNS output (that is, estimate of $y(k+1)$) and the actual system output, $y(k+1)$.

It should be clear that the NNR and the NNS operate in different modes. While the NNS has been previously trained with an on-line temporal learning, the NNR operates in an iterative mode (within the same time step) within a temporal frame (for sequential time steps). In other words, the reconstruction scheme proceeds to the next time step only when the control signal $u(k)$ minimizing the quadratic difference between $y(k+1)$ and the NNS output is found by the NNR after 'l' iterations. This number of iterations depends on a user defined numerical threshold for the quadratic difference. A block diagram of the coupling between the NNS and the NNR - which ultimately provides the reconstruction scheme - is shown in Fig. 3.

The following statistical parameters for the reconstruction error were introduced and calculated for quantifying the reconstruction performance:

$$M_{RE}=\sum_{i=1}^{N}\frac{[u_{NNR}(i)-u(i)]}{N}=\sum_{i=1}^{N}e_u(i) \tag{10}$$

$$V_{RE}=\sum_{i=1}^{N}\frac{[e_u(i)-M_{RE}]^2}{N} \tag{11}$$

For testing the NNR the actual control sequence $u(.)$ was known. Clearly, in a practical application of the proposed scheme the actual control $u(.)$ would not be available and it is actually the output of the scheme. The only condition posed on the selection of $u(.)$ is that its magnitude is within [-1.5, 1.5], matching the control sequence range used to train the NNS. Fig. 4 shows a

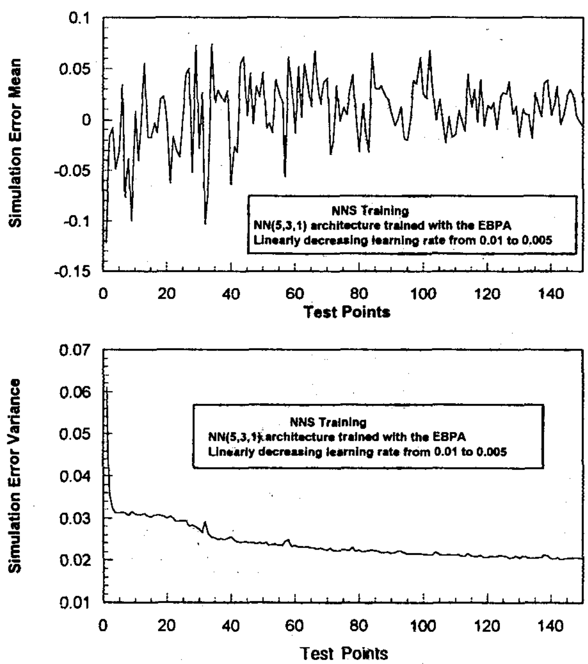

Fig. 1 Simulation error mean and variance versus test points

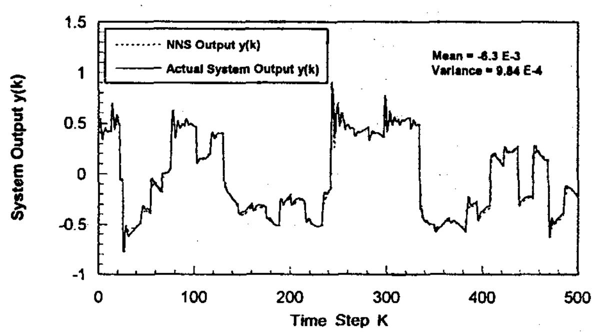

Fig. 2 Comparison of NNS output and actual system output

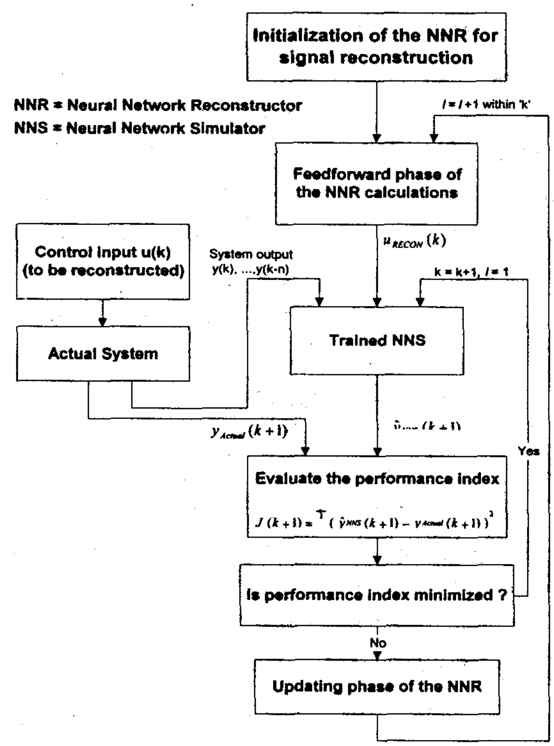

Fig. 3 Block diagram of the overall reconstruction scheme

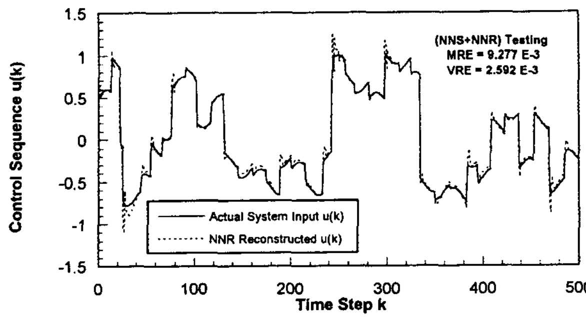

Fig. 4 Comparison of Reconstructed and actual control sequence

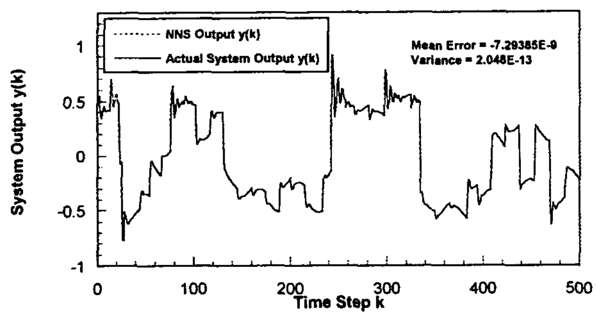

Fig. 5 Comparison of NNS output with reconstructed sequence and actual system output

comparison between the reconstructed sequence $u_{NNR}(.)$ - and the actual $u(.)$. The results are very encouraging. While this outcome is clearly desirable, the important issue of the uniqueness of the solution of the reconstruction $u_{NNR}(.)$ is not addressed here. The final test of the reconstruction is then performed by giving $u_{NNR}(.)$ as input to the NNS to generate $y_{NNS-Recon}(.)$ and comparing it with the actual $y(.)$. The results are shown in Fig. 5, showing a virtual overlap of the two responses since the effect of noise was not considered.

The next goal of the study for the SISO system consisted in verifying the applicability of the proposed scheme to a system with a certain level of noise. Of particular interest was the effect of the noise on a NNS trained with noise-free data. For this purpose consider :

$$y(k+1)=\frac{y(k)y(k-1)y(k-2)u(k-1)[y(k-2)-1]+u(k)}{1+y(k-1)^2+y(k-2)^2}+e(k) \tag{12}$$

with the following conditions :

Condition #1 : $e(k)$ is a white noise following a gaussian distribution with known mean and variance;

Condition #2 : $e(k)$ is a colored noise modeled by :

$$e(k)=a_1\, e(k-1)+e_w(k) \tag{13}$$

where $a_1=0.5$ while $e_w(.)$ is a white noise following a gaussian distribution with known mean and variance.

Using the NNS trained with noise-free data, the (NNS + NNR) reconstruction scheme was tested with data with different S/N ratios for both white and colored noise. The results are shown in Figs. 6~7 and Figs. 8~9 for condition #1 and #2, respectively. The noise study provided the following conclusions. First, the NNS and the NNR are overall robust to system noise with the color of the noise having little or no effect on such robustness. Ultimately the amount of robustness depends on the S/N ratio, as expected. Further-

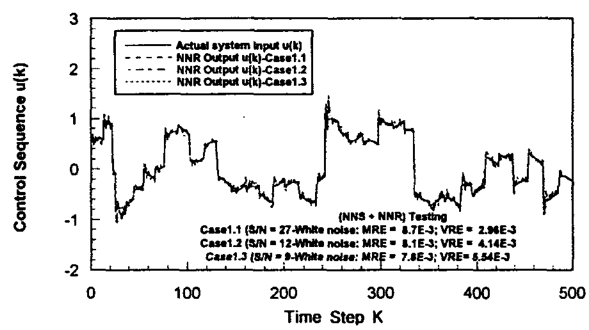

Fig. 6 Comparison of reconstructed and actual control sequence for different noise levels-white noise

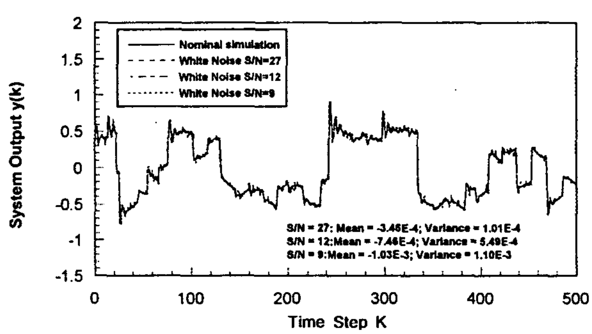

Fig. 7 Comparison of NNS output using reconstructed sequence and actual system output for different noise levels-white noise

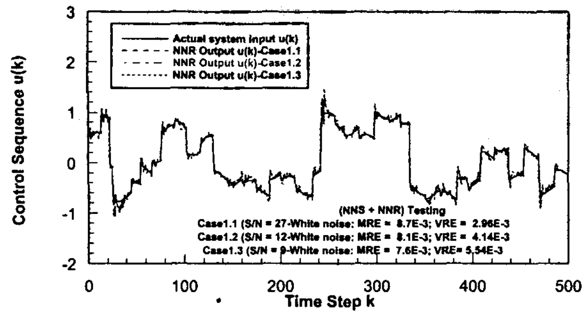

Fig. 8 Comparison of reconstructed and actual control sequence for different levels-colored noise

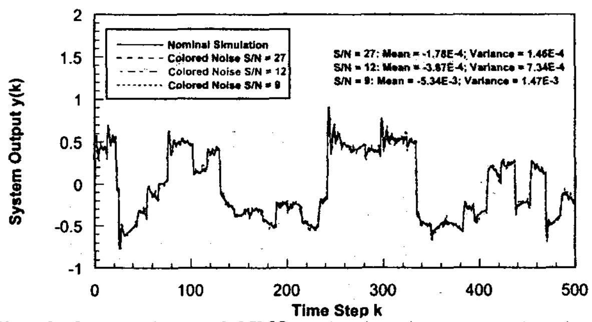

Fig. 9 Comparison of NNS output using reconstructed sequence and actual system output for different noise levels-colored noise

more, very little effect on the noise robustness was observed for a longer training for the NNS.

4. Conclusions

The contribution of this paper is to present a neural network based scheme for reconstructing the control sequence $u(.)$ from given measured $y(.)$, an unknown SISO system. The scheme consists in the coupling of a neural simulator (NNS), previously trained in series-parallel temporal learning mode with experimentally available $u(.)$ and $y(.)$, with an iterative neural reconstructor (NNR) which ultimately provides a reconstruction of $u(.)$ minimizing the difference between the actual $y(.)$ and the $y(.)$ from the NNS.

The scheme has been applied to BIBO SISO systems. The robustness of the scheme to white and colored system noise has been evaluated. The performance of the approach is overall very satisfactory, with the robustness to system noise ultimately being a function of the S/N ratio, as it is for any estimation and/or control scheme. Issues not addressed within this context are a quantification of performance degradation associated with eventual local minimum problems for the NNS within the reconstruction scheme, as well as ways of dealing with the non unique nature of the reconstructed $u(.)$ sequence. Although no claims are here made on the convergence and stability of the neural learning, it could be invoked that these critical performances can be assessed using Lyapunov's stability theory.

References

(1) L. Fausett, 1994, "Fundamentals of Neural Networks Architectures, Algorithms, and Applications", Prentice Hall, Englewood Cliffs, NJ.

(2) P.K. Simpson, 1990, "Artificial Neural Systems: Foundations, Paradigms, Applications, and Implementations", Pergamon Press, NY.

(3) T. Kerr, 1998, "Critique of Some Neural Network Architectures and Claims for Control and Estimation", IEEE Transactions of Aerospace and Electronic Systems, Vol. 34, No. 2, pp. 406~419.

(4) M. Polycarpou, 1998, "On-Line Approximators for Nonlinear system Identification : A Unified Approach", Control and Dynamic Systems Series edited by Prof. C.T. Leondes, Volume 7 of Neural Network Systems Techniques and Applications , Academic Press.

(5) M.Polycarpou, A.T.Vemuri, 1995, "Learning Methodology for Failure Detection and Accommodation", IEEE Control Systems Magazine, Vol. 15, No. 3, pp. 16~24.

(6) C.L. Chen, R.S. Nutter, 1992, "An Extended Back Propagation Learning by Using Heterogenous Processing Units", Proceedings of the International Joint Control Conference on Neural Networks (IJCNN '92), Baltimore, Md, Vol. 3, pp. 988~993.

(7) K.S. Narendra, K. Partasarathy, 1990, "Identification and Control of Dynamical Systems Using Neural Networks", IEEE Transactions on Neural Networks, Vol. 1, pp.4~27.