Control of Nonlinear System with a Disturbance Using Multilayer Neural Networks

Hong Seok Seong

Abstract: The mathematical solutions of the stability convergence are important problems in system control. In this paper, such problems are analyzed and resolved for system control using multilayer neural networks. We describe an algorithm to control an unknown nonlinear system with a disturbance, using a multilayer neural network. We include a disturbance among the modeling error, and the weight update rules of multilayer neural network are derived to satisfy Lyapunov stability. The overall control system is based upon the feedback linearization method. The weights of the neural network used to approximate a nonlinear function are updated by rules derived in this paper . The proposed control algorithm is verified through computer simulation. That is, as the weights of neural network are updated at every sampling time, we show that the output error become finite within a relatively short time.

Keywords: multilayer neural network, nonlinear system control, Lyapunov stability

I. Introduction

The use of adaptive, Lyapunov and variable structure control techniques are well known methods for the control of nonlinear systems. In particular, the feedback linearization method is a popular method for designing controllers for nonlinear systems. Unfortunately, the exact dynamic equation of the system must be known for the use of these methods [1]-[3].

The mathematical solutions of the stability convergence are important problems in system control. To solve these problems, we must know the dynamic equation of the system which often is not known exactly. For this reason, the dynamic equation of the system must be approximated. A neural network can approximate an arbitrary function through learning, and achieve parallel processing and fault tolerance with ease. That is, we can approximate the dynamic equation of an inexactly known system by using a neural network. For control problems, the structure which is used most often in such cases is the multilayer neural network using the error backpropagation algorithm. The multilayer neural network can approximate a nonlinear function to any desired degree of accuracy [4]. A neural network based controller is proposed in this paper to control an inexactly known system [5]-[7].

Most control algorithms using neural networks employ the gradient descent method to update the weights of neural network, but this update rule does not always guarantee the stability of the system. Recently, the stability of control systems using neural network has been studied. Yesildirek simulated the control of a robot manipulator by using a weight update rule obtained by using the dynamic equation of the robot manipulator and the Lyapunov function [8]. Jin showed that the error between the desired output and the system output is finite. In this case, he used neural networks such as the RBF(Radial Basis Function) and CMAC(Cerebellar Model Articulation Controller) [9]. Jin used a modified backpropagation algorithm having a dead-zone function for the weight update rule of the neural network, and showed that the output error converges to zero. From this he showed that the global closed system was stable [10][11]. Renders proved the input and output stability of the nonlinear system using the RBF neural network and Lyapunov function [12]. Seong derived a weight update rule of the neural network on the basis of the feedback linearization method, and showed that the Lyapunov stability of system was guaranteed by using this rule [13].

In the case of a system with a disturbance, many robust controls have been studied. In this paper, we derive the weight update rules of a multilayer neural network for a nonlinear system with a disturbance. We show that a control system which consists of this neuro-controller satisfies Lyapunov stability. The overall control system is based upon the feedback linearization method. At the control system, the nonlinear function is implemented with a 3-layer neural network and its weights are updated by using the derived update rules. The structure of this paper is as follows: in section II we describe the feedback linearization method to control the nonlinear system. In section III we discuss the multilayer neural network and its weight update rules. In section IV we describe how to implement the controller described in section II using neural network. In section V we explain the global structure of the controlled system and derive the weight update rules of the neural network which satisfy Lyapunov stability. In section VI we verify the performance of the proposed control algorithm through computer simulation. In section VII we provide our conclusions.

II. Nonlinear system

In this section, we define the assumptions made about the system and the standard equation of the nonlinear system, and describe the feedback linearization method used to control the nonlinear system. Firstly, we describe the following assumptions about the system.

Assumption 1: The nonlinear functions $f(\mathbf{x})$ and $g(\mathbf{x})$ are finite, and $g(\mathbf{x})$ is not zero $\left(g(\mathbf{x}) \neq 0\right)$.

Assumption 2: The desired states of the system and their derivatives are finite.

Assumption 3: The states of the system can be measured.

Assumption 4: The disturbance is bounded $\left(d(t) \leq d_u\right)$.

The nonlinear system which has a single input/output is

Manuscript received: Feb. 24, 2000., Accepted: Sep. 4, 2000.

Hong Seok Seong: Dept. of Electronics, Bucheon College

described as follows

$$y^{(n)} = f(y,y^{(1)},\Lambda\ ,y^{(n-1)}) + g(y,y^{(1)},\Lambda\ ,y^{(n-1)})u \quad (1)$$

where y is the system output and u is the system input, and $f(\cdot)$ and $g(\cdot)$ are the nonlinear functions. Let us define the state variable vector $\boldsymbol{x}$ as follows

$$\mathbf{x}^T = (x_1,x_2,\Lambda\ ,x_n) = (y,y^{(1)},\Lambda\ ,y^{(n-1)}) . \quad (2)$$

The nonlinear system (1) is expressed in state space using of the above state variables as follows

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 \\ &\text{M} \\ \dot{x}_n &= f(x_1,\Lambda\ ,x_n) + g(x_1,\Lambda\ ,x_n)u \\ y &= x_1 \end{aligned} \quad (3)$$

Now, define the tracking error as $e = y_d - y$. Here, y_d is the desired system output and y is the actual system output. If the nonlinear functions $f(\cdot)$ and $g(\cdot)$ of the nonlinear system are known exactly, the control input u which is made to follow the desired output can be expressed as follows

$$u = \frac{-f(\mathbf{x}) + y_d^{(n)} + \mathbf{k}^T\mathbf{e}}{g(\mathbf{x})} \quad (4)$$

where $\mathbf{k}^T = (k_1,\Lambda\ ,k_n)$ is the vector selected arbitrarily, at this time $h(s) = s^n + k_n s^{n-1} + \Lambda + k_1$ becomes the Hurwitz equation, and $\mathbf{e}$ is the tracking error vector defined as follows

$$\mathbf{e}^T = (e,e^{(1)}\Lambda\ ,e^{(n-1)}) = (y_d - y, y_d^{(1)} - y^{(1)},\Lambda\ ,y_d^{(n-1)} - y^{(n-1)}) . \quad (5)$$

If we substitute the control input (4) into the function of the nonlinear system (1), the following error equation is obtained.

$$e^{(n)} + k_n e^{(n-1)} + \Lambda + k_1 e = 0 . \quad (6)$$

Therefore, because $\mathbf{k}$ is the vector which satisfies the Hurwitz condition, we know that the $\lim_{t\to\infty} e(t)$ is zero. That is, if (4) is used as the control input, we know that the output of the nonlinear system follows the desired system output.

III. Multilayer neural network

A given nonlinear function can be approximated by a neural network to any desired degree of accuracy. We can describe this fact as follows[4].

Theorem 1: The neural network $\hat{f}(\mathbf{x},\mathbf{V}_f^*,\mathbf{w}_f^*)$ and $\hat{g}(\mathbf{x},\mathbf{V}_f^*,\mathbf{w}_f^*)$ with the optimal weight $\mathbf{V}_f^*,\mathbf{w}_f^*,\mathbf{V}_g^*,\mathbf{w}_g^*$ is able to approximate the continuous functions $f(\mathbf{x})$ and $g(\mathbf{x})$ within arbitrary degree $\varepsilon_f,\varepsilon_g$ in a compact area. That is,

$$\begin{aligned} &\max\left|\hat{f}(\mathbf{x},\mathbf{V}_f^*,\mathbf{w}_f^*) - f(\mathbf{x})\right| \le \varepsilon_f \\ &\max\left|\hat{g}(\mathbf{x},\mathbf{V}_f^*,\mathbf{w}_f^*) - g(\mathbf{x})\right| \le \varepsilon_g,\ \textit{all}\ \mathbf{x}\in C \end{aligned}$$

where C is the compact area with the finite order.

Using this theory, the nonlinear function of the nonlinear system is implemented as

$$\hat{f} = \mathbf{w}_f^T\mathbf{s}_f(\mathbf{V}_f^T\mathbf{x}), \hat{g} = \mathbf{w}_g^T\mathbf{s}_g(\mathbf{V}_g^T\mathbf{x}) \quad (7)$$

where $\mathbf{V}_f,\mathbf{V}_g$ are the hidden layer weight matrixes in $\Re^{n_h\times n_i}$ $(n_h\times n_i)$ of the multilayer neural networks used to approximate the nonlinear functions $f(\cdot),g(\cdot)$ respectively, and $\mathbf{w}_f,\mathbf{w}_g$ are the output layer weight vector in $\Re^{n_h}$ $(n_h\times 1)$, n_i is the number of the inputs in the input layer, and n_h is the number of the neurons in the hidden layer. $\hat{\mathbf{s}}(\mathbf{V}^T\mathbf{x})$ means $\left[s(\mathbf{v}_1^T\mathbf{x}),\Lambda\ ,s(\mathbf{v}_{n_h}^T\mathbf{x})\right]^T$ and $\mathbf{v}_i^T$ means $\left[v_{1i},\Lambda\ ,v_{n_i i}\right]$.

The mapping $\mathbf{s}(\cdot):\Re^{n_h}\to\Re^{n_h}$ is the vector valued activation function and each element of $\mathbf{s}(\cdot)$ is denoted as $s(\cdot)$ which is defined as follows:

$$s(x) = \frac{1}{1+e^{-x}} . \quad (8)$$

The weights of the neural network are updated as defined in (9) and (10). First, the weights of the hidden layer are updated as

$$\begin{aligned} \dot{\mathbf{V}}_f &= -\mathbf{e}^T\mathbf{P}\mathbf{b}\mathbf{x}\mathbf{w}_f^T\mathbf{D}_{s_f} + \|\mathbf{e}\|\mathbf{V}_f \\ \dot{\mathbf{V}}_g &= -\mathbf{e}^T\mathbf{P}\mathbf{b}u\mathbf{x}\mathbf{w}_g^T\mathbf{D}_{s_g} + \|\mathbf{e}\|\mathbf{V}_g \end{aligned} \quad (9)$$

where $\mathbf{D}_s$ is the diagonal matrix which has the derivative of the sigmoid function in diagonal elements. That is, $\mathbf{D}_s \equiv diag\left[s_1 s_2\Lambda\ s_{n_h}\right]$. The derivative s of the sigmoid function is the derivative of the output of the neuron and is expressed as $s = o(1-o)$, where o is the output of a neuron. The weights of the output layer are updated as

$$\begin{aligned} \dot{\mathbf{w}}_f &= -\mathbf{e}^T\mathbf{P}\mathbf{b}\left(\hat{\mathbf{s}}_f - \mathbf{D}_{s_f}\mathbf{V}_f^T\mathbf{x}\right) + \|\mathbf{e}\|\mathbf{w}_f \\ \dot{\mathbf{w}}_g &= -\mathbf{e}^T\mathbf{P}\mathbf{b}\,u\left(\hat{\mathbf{s}}_g - \mathbf{D}_{s_g}\mathbf{V}_g^T\mathbf{x}\right) + \|\mathbf{e}\|\mathbf{w}_g . \end{aligned} \quad (10)$$

This weight update rules are derived for the proof of Lyapunov stability. The weight update rules are derived in the section V.

IV. Controller using multilayer neural network

In section II, if the nonlinear function of the nonlinear system which is expressed as (1) is known exactly, we know that (4) can be used as the control input for the case of non-disturbance. However, in a real environment, we do not know the exact nonlinear function of the nonlinear system, and there

is often a disturbance present. A nonlinear system which has a disturbance can be described as follows

$$y^{(n)} = f(y, y^{(1)}, \Lambda, y^{(n-1)}) + g(y, y^{(1)}, \Lambda, y^{(n-1)})u + d \qquad (11)$$

where d is the disturbance. Therefore, an unknown nonlinear function is implemented by using the multilayer neural network based on the theory defined in section III and the control input u defined in (12). A disturbance shall be treated as the modeling error. Therefore, by using the control input in (12) and the weight update rules of the neural network in (9) and (10), we will show in section V that the output error and the weight errors become finite.

$$u = \frac{-\hat{f}(\mathbf{x}) + y_d^{(n)} + \mathbf{k}^T \mathbf{e}}{\hat{g}(\mathbf{x})} \ . \qquad (12)$$

Here, $\hat{f}(\cdot)$ and $\hat{g}(\cdot)$ are the outputs of the multilayer neural network approximating the nonlinear function. To obtain the error equation, substitute the control input (12) into the system equation (11), which gives

$$e^{(n)} = y^{(n)} - y_d^{(n)} = \mathbf{k}^T \mathbf{e} + \left\{ f(\mathbf{x}) - \hat{f}(\mathbf{x}) \right\} + \left\{ g(\mathbf{x}) - \hat{g}(\mathbf{x}) \right\} u + d \ . \quad (13)$$

If we express the error equation (13) by the means of tracking error vector (5), we get the following error state equation

$$\dot{\mathbf{e}} = \mathbf{A}\mathbf{e} + \mathbf{b}\left[\left\{ f(\mathbf{x}) - \hat{f}(\mathbf{x}) \right\} + \left\{ g(\mathbf{x}) - \hat{g}(\mathbf{x}) \right\} u + d \right] \qquad (14)$$

where $\mathbf{A}, \mathbf{b}$ is as

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 & \Lambda & 0 \\ 0 & 0 & 1 & \Lambda & 0 \\ \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} & \mathrm{M} \\ 0 & 0 & 0 & \Lambda & 1 \\ -k_1 & -k_2 & -k_3 & \Lambda & -k_n \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \mathrm{M} \\ 1 \end{pmatrix}. \qquad (15)$$

V. Stability of system

In this section, we derive the weight update rules of the neural network ((9), (10)). During the derivation of the rules, we will show that the output error is bounded.

Subsequently in order to use the stability theory, the following is described [14].

Theorem 2: Consider the differential equation

$$\dot{x} = f(x)$$

Let D be a bounded neighborhood of the origin and let D^c be its complement. If $V(x)$ is a scalar function with continuous partial derivatives in D^c and satisfies the following conditions, the solution of the given differential equation is bounded for all $t \geq 0$.

i) $V(x) > 0 \forall x \in D^c$

ii) $\dot{V}(x) \leq 0 \quad \forall x \in D^c$

iii) $\lim_{\|x\| \to \infty} V(x) \to \infty$

Fig. 1 is the global block diagram of the closed loop control system. We use 2-multilayer neural networks for the nonlinear functions $f(\cdot)$ and $g(\cdot)$.

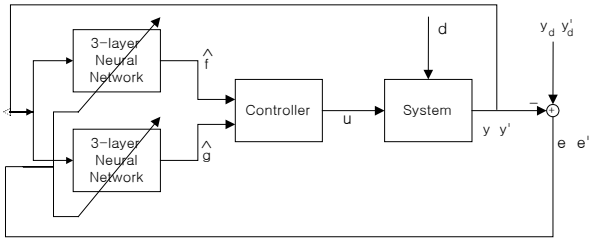

Fig. 1. The block diagram of the control system.

The following is the characteristic of the sigmoid function $s(x)$, and we use it when we derive the weight update rules of the neural network.

Lemma 1: The sigmoid function $s(x)$ has the following characteristic.

1) The sigmoid function is strictly increasing.

2) $\frac{ds}{dx} > 0$

3) $\tilde{s}(x_1, x_2) = s(x_1) - s(x_2) = \left.\frac{ds}{dx}\right|_{x=z} \tilde{x}$

where $\tilde{x} = x_1 - x_2$ and $x = z \in (x_1, x_2)$

Proof: 1) It is self-evident by the following definition of the sigmoid function.

$$s(x) = \frac{1}{1 + e^{-x}} \qquad (16)$$

2) By 1), the sigmoid function is always strictly increasing and its first derivative is therefore positive. 3) $s(x)$ defined as (16) is continuous in all x, and has a continuous first derivative. Therefore, by the mean value theorem,

$$\tilde{s}(x_1, x_2) = s(x_1) - s(x_2) = \left.\frac{ds}{dx}\right|_{x=z} (x_1 - x_2)$$

where z is a point between x_1 and x_2. Consequently,

$$\tilde{s}(x_1, x_2) = \left.\frac{ds}{dx}\right|_{x=z} \tilde{x} \ .$$

We will now describe the following fact about the norm of the input of the neural network and we will use it when we derive the weight update rules of the neural network. γ

Fact 1: The norm of the input $\mathbf{x}$ of the neural network satisfies the following inequality.

$$\|\mathbf{x}\| \leq c_1 \|\mathbf{y}_d\| + c_2 \|\mathbf{e}\| \qquad (17)$$

The c_1 and c_2 are $c_1 \geq 1$ and $c_2 \geq 1$.

Proof:

$$\begin{aligned}\|x\| &= \sqrt{x_1^2 + x_2^2 + \mathrm{L} + x_{n_i}^2} \\ &= \sqrt{\left(y_d - e\right)^2 + \left(y_d^{(1)} - e^{(1)}\right)^2 + \mathrm{L} + \left(y_d^{(n_i-1)} - e^{(n_i-1)}\right)^2} \\ &= \sqrt{\left(\|\mathbf{y}_d\|^2 + \|\mathbf{e}\|^2 - 2\left(y_d e + y_d^{(1)} e^{(1)} + \mathrm{L} + y_d^{(n_i-1)} e^{(n_i-1)}\right)\right)} \\ &\le c_1\|\mathbf{y}_d\| + c_2\|\mathbf{e}\| .\end{aligned}$$

Now, We can define the following theory. γ

Theorem 3: Under assumptions 1 to 4 given in section II, if the weights of the multilayer neural network are updated by (9) and (10), the control input (12) makes the output error and the weight errors of the neural network finite.

Proof: First of all, we can modify the previous error equation (14) as follows

$$\dot{\mathbf{e}} = \mathbf{A}\mathbf{e} + \mathbf{b}\left[\left\{ f^{*}(\mathbf{x}) - f(\mathbf{x})\right\} + \left\{ g^{*}(\mathbf{x}) - \hat{g}(\mathbf{x})\right\} u\right] + \mathbf{b}\varepsilon + \mathbf{b}d \quad (18)$$

where $\hat{f}^*(\cdot)$ and $\hat{g}^*(\cdot)$ are the values of the nonlinear functions using the optimal weights of the multilayer neural network, $\hat{f}(\cdot)$ and $\hat{g}(\cdot)$ are the values of the nonlinear functions using the current weights of the multilayer neural network, and ε is defined as (19). In the equation (19), $f(\cdot)$ and $g(\cdot)$ are the real values of the nonlinear function, and ε is the difference between the real value of the nonlinear function and the value of the nonlinear function using the optimal weight of the multilayer neural network. We call this the modeling error of the nonlinear function due to the neural network.

$$\varepsilon = \left\{ f(\mathbf{x}) - \hat{f}^*(\mathbf{x})\right\} + \left\{ g(\mathbf{x}) - \hat{g}^*(\mathbf{x})\right\} u . \quad (19)$$

Now, let us define the following Lyapunov function

$$V = \frac{1}{2}\mathbf{e}^T\mathbf{P}\mathbf{e} + \frac{1}{2}\tilde{\mathbf{w}}_f^T\tilde{\mathbf{w}}_f + \frac{1}{2}\tilde{\mathbf{w}}_g^T\tilde{\mathbf{w}}_g + \frac{1}{2}tr\left[\tilde{\mathbf{V}}_f^T\tilde{\mathbf{V}}_f\right] + \frac{1}{2}tr\left[\tilde{\mathbf{V}}_g^T\tilde{\mathbf{V}}_g\right] \quad (20)$$

where $\tilde{\mathbf{w}}$ is the difference between the optimal weight and the current weight of the output layer in the multilayer neural network $\left(\tilde{\mathbf{w}} = \mathbf{w}^* - \mathbf{w}\right)$, and $\tilde{\mathbf{V}}$ is he difference between the optimal weight and the current weight of the hidden layer in the multilayer neural network $\left(\tilde{\mathbf{V}} = \mathbf{V}^* - \mathbf{V}\right)$.

Differentiating (20) gives

$$\dot{V} = \frac{1}{2}\dot{\mathbf{e}}^T\mathbf{P}\mathbf{e} + \frac{1}{2}\mathbf{e}^T\mathbf{P}\dot{\mathbf{e}} + \tilde{\mathbf{w}}_f^T\dot{\tilde{\mathbf{w}}}_f + \tilde{\mathbf{w}}_g^T\dot{\tilde{\mathbf{w}}}_g + tr\left[\tilde{\mathbf{V}}_f^T\dot{\tilde{\mathbf{V}}}_f\right] + tr\left[\tilde{\mathbf{V}}_g^T\dot{\tilde{\mathbf{V}}}_g\right] . \quad (21)$$

Substituting the error equation (18) for (21) gives

$$\begin{aligned}\dot{V} &= \tfrac{1}{2}\mathbf{e}^T\left(\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A}\right)\mathbf{e} + \mathbf{e}^T\mathbf{P}\mathbf{b}\left(\varepsilon + d\right) + \mathbf{e}^T\mathbf{P}\mathbf{b}\left\{\left(f^* - \hat{f}\right) + \left(g^* - \hat{g}\right)u\right\} \\ &\quad + \dot{\tilde{\mathbf{w}}}_f^T\dot{\tilde{\mathbf{w}}}_f + \dot{\tilde{\mathbf{w}}}_g^T\dot{\tilde{\mathbf{w}}}_g + tr\left[\dot{\tilde{\mathbf{V}}}_f^T\dot{\tilde{\mathbf{V}}}_f\right] + tr\left[\dot{\tilde{\mathbf{V}}}_g^T\dot{\tilde{\mathbf{V}}}_g\right] .\end{aligned} \quad (22)$$

Define a Lyapunov equation as follows

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} = -\mathbf{Q} . \quad (23)$$

where $\mathbf{P}$ and $\mathbf{Q}$ are the positive definite symmetric matrix. In (22), $\hat{f}^* - \hat{f}$ can be modified as follows.

$$\begin{aligned}\hat{f}^* - \hat{f} &= \mathbf{w}_f^{*T}\mathbf{s}_f^* - \mathbf{w}_f^T\mathbf{s}_f \\ &= \mathbf{w}_f^{*T}\mathbf{s}_f^* - \mathbf{w}_f^T\hat{\mathbf{s}}_f + \mathbf{w}_f^{*T}\hat{\mathbf{s}}_f - \mathbf{w}_f^{*T}\hat{\mathbf{s}}_f \\ &= \left(\mathbf{w}_f^* - \mathbf{w}_f\right)^T\hat{\mathbf{s}}_f + \mathbf{w}_f^{*T}\left(\mathbf{s}_f^* - \hat{\mathbf{s}}_f\right) \\ &= \tilde{\mathbf{w}}_f^T\hat{\mathbf{s}}_f + \tilde{\mathbf{w}}_f^T\tilde{\mathbf{s}}_f + \mathbf{w}_f^T\tilde{\mathbf{s}}_f\end{aligned} . \quad (24)$$

In order to modify $\tilde{s}$, define the variables.

$$\begin{aligned}&y_i = \mathbf{v}_i^T\mathbf{x}, y_i^* = \mathbf{v}_i^{*T}\mathbf{x} \\ &v_i = \left(v_{1i}v_{2i}\Lambda\, v_{ni}\right)^T, i = 1,\Lambda, n_h\end{aligned} \quad (25)$$

$\tilde{s}$ is modified by lemma (3) as follows

$$\begin{aligned}\tilde{s}_i &= \hat{s}_i^*(y_i^*) - \hat{s}_i(y_i) \\ &= \left.\frac{d\hat{s}_i}{dy}\right|_{y=z_i}\left(\mathbf{v}_i^{*T}\mathbf{x} - \mathbf{v}_i^T\mathbf{x}\right) \\ &= \left.\frac{d\hat{s}_i}{dy}\right|_{y=z_i}\tilde{\mathbf{v}}_i^T\mathbf{x}\end{aligned} \quad (26)$$

where z_i is a value between y_i and y_i^* and $\tilde{\mathbf{s}}(\mathbf{V}^T\mathbf{x})$ is modified as follows

$$\tilde{\mathbf{s}}(\mathbf{V}^T\mathbf{x}) = \mathbf{D}_{s'}\tilde{\mathbf{V}}^T\mathbf{x} \quad (27)$$

where $\mathbf{D}_{s'} = diag\left[s'_1\, s'_2\Lambda\,\Lambda\, s'_{n_h}\right]$.

Substitute (27) for (24) gives

$$\begin{aligned}\hat{f}^* - \hat{f} &= \tilde{\mathbf{w}}_f^T\hat{\mathbf{s}}_f + \mathbf{w}_f^T\mathbf{D}_{s'_f}\tilde{\mathbf{V}}_f^T\mathbf{x} + \mathbf{w}_f^T\mathbf{D}_{s'_f}\tilde{\mathbf{V}}_f^T\mathbf{x} \\ &= \tilde{\mathbf{w}}_f^T\hat{\mathbf{s}}_f + \tilde{\mathbf{w}}_f^T\mathbf{D}_{s'_f}\left(\mathbf{V}_f^* - \mathbf{V}_f\right)^T\mathbf{x} + \mathbf{w}_f^T\mathbf{D}_{s'_f}\tilde{\mathbf{V}}_f^T\mathbf{x} \\ &= \tilde{\mathbf{w}}_f^T\left(\hat{\mathbf{s}}_f - \mathbf{D}_{s'_f}\mathbf{V}_f^T\mathbf{x}\right) + \tilde{\mathbf{w}}_f^T\mathbf{D}_{s'_f}\mathbf{V}_f^{*T}\mathbf{x} + \mathbf{w}_f^T\mathbf{D}_{s'_f}\tilde{\mathbf{V}}_f^T\mathbf{x} .\end{aligned} \quad (28)$$

$\hat{g}^* - \hat{g}$ can be modified similarly.

$$\hat{g}^* - \hat{g} = \tilde{\mathbf{w}}_g^T\left(\hat{\mathbf{s}}_g - \mathbf{D}_{s'_g}\mathbf{V}_g^T\mathbf{x}\right) + \tilde{\mathbf{w}}_g^T\mathbf{D}_{s'_g}\mathbf{V}_g^{*T}\mathbf{x} + \mathbf{w}_g^T\mathbf{D}_{s'_g}\tilde{\mathbf{V}}_g^T\mathbf{x} \quad (29)$$

Substituting (28) and (29) for $\dot{V}$ (22) gives

$$\begin{aligned}\dot{V} &= -\frac{1}{2}\mathbf{e}^T\mathbf{Q}\mathbf{e} + \mathbf{e}^T\mathbf{P}\mathbf{b}\left(\varepsilon + d\right) \\ &\quad + \tilde{\mathbf{w}}_f^T\left\{\dot{\tilde{\mathbf{w}}}_f + \mathbf{e}^T\mathbf{P}\mathbf{b}\left(\hat{\mathbf{s}}_f - \mathbf{D}_{s'_f}\mathbf{V}_f^T\mathbf{x}\right)\right\} \\ &\quad + \tilde{\mathbf{w}}_g^T\left\{\dot{\tilde{\mathbf{w}}}_g + \mathbf{e}^T\mathbf{P}\mathbf{b}u\left(\hat{\mathbf{s}}_g - \mathbf{D}_{s'_g}\mathbf{V}_g^T\mathbf{x}\right)\right\} \\ &\quad + tr\left\{\tilde{\mathbf{V}}_f^T\left(\dot{\tilde{\mathbf{V}}}_f + \mathbf{e}^T\mathbf{P}\mathbf{b}\mathbf{x}\mathbf{w}_f^T\mathbf{D}_{s'_f}\right)\right\} \\ &\quad + tr\left\{\tilde{\mathbf{V}}_g^T\left(\dot{\tilde{\mathbf{V}}}_g + \mathbf{e}^T\mathbf{P}\mathbf{b}u\mathbf{x}\mathbf{w}_g^T\mathbf{D}_{s'_g}\right)\right\} \\ &\quad + \mathbf{e}^T\mathbf{P}\mathbf{b}\tilde{\mathbf{w}}_f^T\mathbf{D}_{s'_f}\mathbf{V}_f^{*T}\mathbf{x} + \mathbf{e}^T\mathbf{P}\mathbf{b}u\tilde{\mathbf{w}}_g^T\mathbf{D}_{s'_g}\mathbf{V}_g^{*T}\mathbf{x} .\end{aligned} \quad (30)$$

We substitute the following weight update rules of the neural network for (30).

$$\begin{aligned}
\dot{\tilde{\mathbf{w}}}_f &= -\mathbf{e}^T\mathbf{P}\mathbf{b}\left(\hat{\mathbf{s}}_f - \mathbf{D}_{s_f}\mathbf{V}_f^T\mathbf{x}\right) + \|\mathbf{e}\|\mathbf{w}_f \\
\dot{\tilde{\mathbf{w}}}_g &= -\mathbf{e}^T\mathbf{P}\mathbf{b}u\left(\hat{\mathbf{s}}_g - \mathbf{D}_{s_g}\mathbf{V}_g^T\mathbf{x}\right) + \|\mathbf{e}\|\mathbf{w}_g \\
\dot{\tilde{\mathbf{V}}}_f &= -\mathbf{e}^T\mathbf{P}\mathbf{b}\mathbf{x}\mathbf{w}_f^T\mathbf{D}_{s_f} + \|\mathbf{e}\|\mathbf{V}_f \\
\dot{\tilde{\mathbf{V}}}_g &= -\mathbf{e}^T\mathbf{P}\mathbf{b}u\mathbf{x}\mathbf{w}_g^T\mathbf{D}_{s_g} + \|\mathbf{e}\|\mathbf{V}_g
\end{aligned} \tag{31}$$

and obtain

$$\begin{aligned}
\dot{V} &= -\frac{1}{2}\mathbf{e}^T\mathbf{Q}\mathbf{e} + \mathbf{e}^T\mathbf{P}\mathbf{b}(\varepsilon + d) + \|\mathbf{e}\|\,\tilde{\mathbf{w}}_f^T\mathbf{w}_f + \|\mathbf{e}\|\,\tilde{\mathbf{w}}_g^T\mathbf{w}_g \\
&+ \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_f^T\mathbf{V}_f\right\} + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_g^T\mathbf{V}_g\right\} \\
&+ \mathbf{e}^T\mathbf{P}\mathbf{b}\tilde{\mathbf{w}}_f^T\mathbf{D}_{s'_f}\mathbf{V}_f^{*T}\mathbf{x} + \mathbf{e}^T\mathbf{P}\mathbf{b}u\tilde{\mathbf{w}}_g^T\mathbf{D}_{s'_g}\mathbf{V}_g^{*T}\mathbf{x}\ .
\end{aligned} \tag{32}$$

$\|\cdot\|$ means the norm of vector. For notational convenience, define the matrix of the output layer weights for two nonlinear function as

$$\mathbf{R} \equiv \begin{pmatrix} \mathbf{w}_f & \mathbf{0} \\ \mathbf{0} & \mathbf{w}_g \end{pmatrix} \tag{33}$$

where $\mathbf{0}$ is a zero vector. Substituting matrix $\mathbf{R}$ for (32) gives

$$\begin{aligned}
\dot{V} &= -\frac{1}{2}\mathbf{e}^T\mathbf{Q}\mathbf{e} + \mathbf{e}^T\mathbf{P}\mathbf{b}(\varepsilon + d) + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{R}}^T\mathbf{R}\right\} \\
&+ \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_f^T\mathbf{V}_f\right\} + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_g^T\mathbf{V}_g\right\} \\
&+ \mathbf{e}^T\mathbf{P}\mathbf{b}\tilde{\mathbf{w}}_f^T\mathbf{D}_{s_f}\mathbf{V}_f^{*T}\mathbf{x} + \mathbf{e}^T\mathbf{P}\mathbf{b}u\tilde{\mathbf{w}}_g^T\mathbf{D}_{s_g}\mathbf{V}_g^{*T}\mathbf{x}\ .
\end{aligned} \tag{34}$$

This inequality is satisfied as follows from (34).

$$\begin{aligned}
\dot{V} &\le -\frac{1}{2}\|\mathbf{Q}\|\|\mathbf{e}\|^2 + \|\mathbf{e}\|\|\mathbf{P}\mathbf{b}\|(\varepsilon_u + d_u) + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{R}}^T\mathbf{R}\right\} \\
&+ \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_f^T\mathbf{V}_f\right\} + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_g^T\mathbf{V}_g\right\} + \|\mathbf{e}\|\|\mathbf{P}\mathbf{b}\|\|\tilde{\mathbf{w}}_f\|\left\|\mathbf{D}_{s_f}\right\|\left\|\mathbf{V}_f^*\right\|\|\mathbf{x}\| \\
&+ \|\mathbf{e}\|\|\mathbf{P}\mathbf{b}\|\|u\|\|\tilde{\mathbf{w}}_g\|\left\|\mathbf{D}_{s_g}\right\|\left\|\mathbf{V}_g^*\right\|\|\mathbf{x}\|
\end{aligned} \tag{35}$$

where $\|\mathbf{Q}\|$ means $\lambda_{\min}(\mathbf{Q})$ and $|\varepsilon| \le \varepsilon_u$. And assumption 4 is used.

From (35) and the previous Fact (17), the following inequalities can be obtained.

$$\begin{aligned}
&\|\mathbf{e}\|\|\mathbf{P}\mathbf{b}\|\|\tilde{\mathbf{w}}_f\|\left\|\mathbf{D}_{s_f}\right\|\left\|\mathbf{V}_f^*\right\|\|\mathbf{x}\| \le c_3\|\mathbf{e}\|\|\tilde{\mathbf{w}}_f\|\left(c_1\|\mathbf{y}_d\| + c_2\|\mathbf{e}\|\right) \\
&\le c_4\|\tilde{\mathbf{w}}_f\|\|\mathbf{e}\| + c_5\|\tilde{\mathbf{w}}_f\|\|\mathbf{e}\|^2
\end{aligned} \tag{36}$$

$$\|\mathbf{e}\|\|\mathbf{P}\mathbf{b}\|\|u\|\|\tilde{\mathbf{w}}_g\|\left\|\mathbf{D}_{s_g}\right\|\left\|\mathbf{V}_g^*\right\|\|\mathbf{x}\| \le c_4\|\tilde{\mathbf{w}}_g\|u\|\|\mathbf{e}\| + c_5\|\tilde{\mathbf{w}}_f\|u\|\|\mathbf{e}\|^2 \tag{37}$$

Substituting (36), (37) for (35) results in

$$\begin{aligned}
\dot{V} &\le -\frac{1}{2}\|\mathbf{Q}\|\|\mathbf{e}\|^2 + \|\mathbf{e}\|\|\mathbf{P}\mathbf{b}\|(\varepsilon_u + d_u) + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{R}}^T\left(\mathbf{R}^* - \tilde{\mathbf{R}}\right)\right\} \\
&+ \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_f^T\left(\mathbf{V}_f^* - \tilde{\mathbf{V}}_f\right)\right\} + \|\mathbf{e}\|tr\left\{\tilde{\mathbf{V}}_g^T\left(\mathbf{V}_g^* - \tilde{\mathbf{V}}_g\right)\right\} \\
&+ c_4\|\tilde{\mathbf{w}}_f\|\|\mathbf{e}\| + c_5\|\tilde{\mathbf{w}}_f\|\|\mathbf{e}\|^2 + c_4|u|\|\tilde{\mathbf{w}}_g\|\|\mathbf{e}\| + c_5|u|\|\tilde{\mathbf{w}}_g\|\|\mathbf{e}\|^2\ .
\end{aligned} \tag{38}$$

(38) is modified as follows

$$\begin{aligned}
\dot{V} \le\ & -\|\mathbf{e}\|\left[\left\{\frac{1}{2}\|\mathbf{Q}\| - c_5\left(\|\tilde{\mathbf{w}}_f\| + |u|\|\tilde{\mathbf{w}}_g\|\right)\right\}\|\mathbf{e}\|\right. \\
&\left. -\left\{\|\mathbf{P}\mathbf{b}\|(\varepsilon_u + d_u) + c_4\left(\|\tilde{\mathbf{w}}_f\| + |u|\|\tilde{\mathbf{w}}_g\|\right)\right\}\right] \\
&- \|\mathbf{e}\|\left\{\|\tilde{\mathbf{R}}\|\left(\|\tilde{\mathbf{R}}\| - \|\mathbf{R}^*\|\right) + \|\tilde{\mathbf{V}}_f\|\left(\|\tilde{\mathbf{V}}_f\| - \|\mathbf{V}_f^*\|\right) + \|\tilde{\mathbf{V}}_g\|\left(\|\tilde{\mathbf{V}}_g\| - \|\mathbf{V}_g^*\|\right)\right\}\ .
\end{aligned} \tag{39}$$

From (39), we can define the region $D\left\{\left(\mathbf{e}, \tilde{\mathbf{V}}_f, \tilde{\mathbf{V}}_g, \tilde{\mathbf{R}}\right)\right\}$ including the origin as follows

$$\begin{aligned}
D\Bigg\{&\left(\mathbf{e}, \tilde{\mathbf{V}}_f, \tilde{\mathbf{V}}_g, \tilde{\mathbf{R}}\right)\|\mathbf{e}\| \le \frac{\|\mathbf{P}\mathbf{b}\|(\varepsilon_u + d_u) + c_4\left(\|\tilde{\mathbf{w}}_f\| + |u|\|\tilde{\mathbf{w}}_g\|\right)}{\frac{1}{2}\|\mathbf{Q}\| - c_5\left(\|\tilde{\mathbf{w}}_f\| + |u|\|\tilde{\mathbf{w}}_g\|\right)}, \\
&\|\tilde{\mathbf{R}}\| \le \|\mathbf{R}^*\|, \|\tilde{\mathbf{V}}_f\| \le \|\mathbf{V}_f^*\|, \|\tilde{\mathbf{V}}_g\| \le \|\mathbf{V}_g^*\|\Bigg\}.
\end{aligned} \tag{40}$$

Therefore, $\dot{V} \le 0$ at the complement D^c of the region $D\left\{\left(\mathbf{e}, \tilde{\mathbf{V}}_f, \tilde{\mathbf{V}}_g, \tilde{\mathbf{R}}\right)\right\}$, and by the previous described stability theory, because the solutions of the differential equation (18) and (31) are finite, the output error and the weight errors of the neural network are bounded.

VI. Simulation

The proposed algorithm was analyzed for two single-input and single output (SISO) systems. For the computer simulations, we used the inverted pendulum system and the one-link manipulator as SISO systems because we can know their dynamic equations more exactly.

1. Inverted pendulum system

The dynamic equation of inverted pendulum system is as follows[15]

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{g\sin x_1 - \dfrac{mlx_2^2\cos x_1\sin x_1}{m_c + m}}{l\left(\dfrac{4}{3} - \dfrac{m\cos^2 x_1}{m_c + m}\right)} + \frac{\dfrac{\cos x_1}{m_c + m}}{l\left(\dfrac{4}{3} - \dfrac{m\cos^2 x_1}{m_c + m}\right)}u \\
y &= x_1
\end{aligned} \tag{41}$$

where $g = 9.8m/s^2$, $m_c = 1kg$, $m = 0.1kg$, and $l = 0.5m$. We used two 3-layer neural networks to let $f(\cdot)$ and $g(\cdot)$ learn respectively. Because the order of system is two, the number of neural network inputs is two. We used 20 neurons in the hidden layer. The parameters of the controller were set as follows: $k^T = [6.0, 8.0]$, and the disturbance $d(t) = 0.05\cos t$. The initial value of the weights in the neural network was set as a random value between -0.01 and 0.01, and the learning

rate of neural network at 0.4. The sampling time of controller was 10 msec. The initial values of system state were $\left(-\frac{\pi}{60},0\right)$. We used the fourth order Runge-Kutta method for the differential equation. The desired output of system was $y_d = \frac{\pi}{30}\sin\left(\frac{2\pi}{6}t\right)$.

Fig. 2 shows the output of system and the output error. Fig. 3 shows the derivative of system output and its error. Fig. 4 shows the control input.

Fig 2 shows that there was a small output error until about 10 sec, but after 10 sec this output error become insignificant, and the system output followed the desired system output.

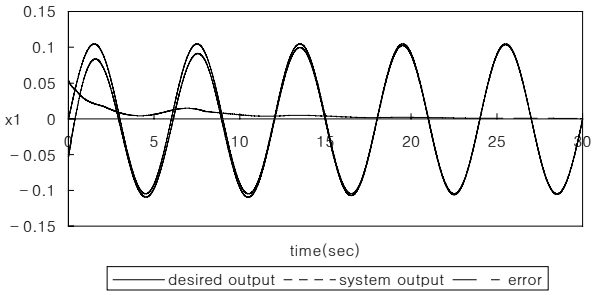

Fig. 2. The desired output, the system output and the error.

Similarly, the velocity of system output followed the desired velocity of system output (in fig 3). At the same time, we can see in fig 4 that the used control input is uniform.

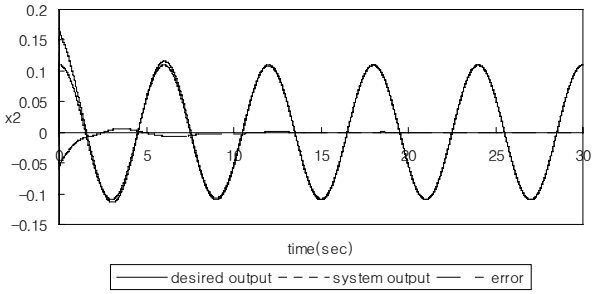

Fig. 3. The Desired output derivative, the system output derivative and the error.

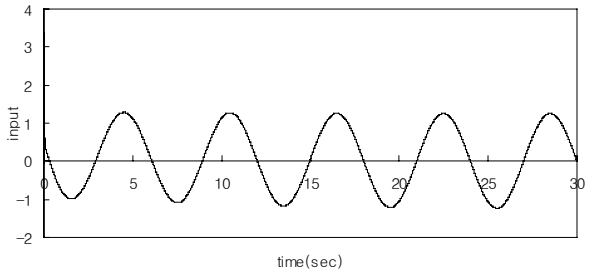

Fig. 4. The control input.

2. One link manipulator

The dynamic equation of a one-link manipulator is as follows [10]

$$T(t)=ml^2\ddot{\theta}(t)+v\dot{\theta}(t)+mgl\cos\theta(t) \qquad (42)$$

where the length l, mass m, friction coefficient v, and gravity constant g were respectively $l=1m$, $m=2.0kg$, $v=1.0kg\,m^2/s$, and $g=9.8kgm/s^2$. The conditions for the multilayer neural network and the controller parameter were same as the conditions described in section VI-1. The disturbance was $d(t)=\cos t$ and the initial values of system state $(0,0)$. The desired output of system was $y_d = 10\sin(0.5\pi t)$. Fig. 5 shows the output of system and the output error. Initially, there was a small output error, but this output error gradually decreased.

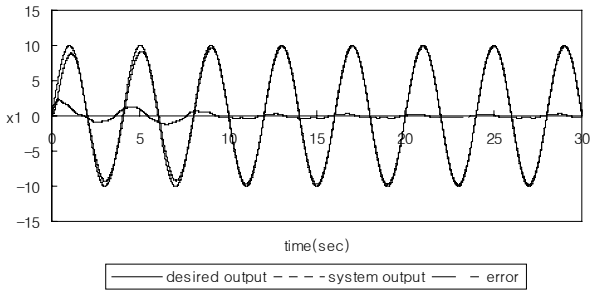

Fig. 5. The desired output, the system output and the error.

From the simulation results, we see that if we control the system using the proposed algorithm, the system output follows the desired system output.

VII. Conclusion

Until now, many researchers have used neural networks for system control. However, the mathematical solution of the stability convergence, which is an important problem in system control, received little attention. In this paper, such problems were analyzed and resolved for system control using multilayer neural networks. We derived the weight update rules of a neural network and proved that the output error and the weight error became finite. We verified the performance of the proposed algorithm through computer simulations. We also considered a nonlinear system with a disturbance. As the weights of neural network were updated at every sampling time, we showed that the output error became finite within a relatively short time. We expect that the proposed algorithm will be able to solve control problems for nonlinear systems for which the dynamic equation is not known exactly, and for which a disturbance is present.

As a further study, it is suggested that the control of MIMO systems using multilayer neural networks with multi-outputs be studied, considering the stability of the system at the same time. In the case of multilayer neural network with two or more hidden layers, the weight update rules guaranteeing the system stability have to be studied, as well.

References

[1] AIsidori, *Nonlinear Control Systems: An Introduction*, Springer-Verlag Berlin, Heidelberg, 2ed, 1989.

[2] J-J. E. Slotine and W. Li, *Applied Nonlinear Control, Prentice Hall International*, Erglewood Cliffs, NJ, 1991.

[3] M. Vidaysagar, *Nonlinear Systems Analysis*, Prentice Hall

International, Erglewood Cliffs, NJ, 1993.
[4] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-366, 1989.
[5] D. Psaltis, A. Sideris, and A. A. Yamamura, "A multilayered neural network controller," *IEEE Control System Magazine*, vol. 8, no. 2, pp. 17-21, Apr., 1988.
[6] F. C. Chen, "Back-propagation neural networks for nonlinear self tuning adaptive control," *IEEE Control System Magazine*, vol. 10, no. 2, pp. 44-48, Apr., 1990.
[7] F. C. Chen and H. K. Khalil, "Adaptive control of nonlinear systems using neural networks-a dead-zone approach," *Proceedings of American Control Conference*, vol. 1, pp. 667-672, 1991.
[8] A. Yesildirek and F. L. Lewis, "A neural net controller for robots with hebbian tuning and guaranteed tracking," *Proceedings of the American Control Conference*, vol. 4, pp. 2784-2789, Jun., 1995.
[9] Y. Jin, G. Pipe, and A. Winfield, "Stable neural control of discrete systems," *Proc. of the 1993 International Symposium of Intelligent Control*, pp. 110-115, Aug., 1993.
[10] L. Jin, P. N. Nikiforuk, and M. M. Gupta, "Adaptive tracking of SISO nonlinear systems using multilayered neural networks," *Proc of the American Control Conference*, vol. 1, pp. 56-60, 1992.
[11] L. Jin, P. N. Nikiforuk, and M. M. Gupta, "Direct adaptive output tracking control using multilayered neural networks", *IEE Proceedings-D*, vol. 140, no. 6, pp. 393-398. Nov., 1993.
[12] J. M. Renders, M. Saerens, and H. Bersini, "Adaptive neurocontrol of MIMO systems based on stability theory", *IEEE International Conference on Neural Network*, vol. 4, pp. 2476-2481, 1994.
[13] H. S. Seong and K. H. Lee, "The nonlinear system control using the neural network guaranteeing Lyapunov stability", *Journal of the Institute of Control, Automation, and System*, vol. 2 no. 3, pp. 142-147, 1996.
[14] K. S. Narendra and A. M. Annaswamy, Stable Adaptive Systems, Prentice Hall, Erglewood Cliffs, NJ, 1989.
[15] Li-in Wang, Adaptive Fuzzy Systems and Control: Design and Stability Analysis, Prentice Hall, Erglewood Cliffs, NJ, 1994.

Hong Seok Seong
Was born in Korea on March 26, 1963. He received the B.S., M.S., and Ph. D. degrees in electronics engineering from Sogang University in 1986, 1988, and 1997, respectively. Since joining ETRI in 1988, he had developed the optical communication system, self-tuning controller, and IMT-2000. Since March 2000, he has been with Bucheon College as full time instructor. His research interests include adaptive control, intelligent control, and neural network.