'94 KACC (1994. 10. 17 ~ 20)

Nonlinear System Control by use of Neural Networks

Ping Zhang* Yoshiyuki Sankai** Michio Ohta**
* Graduate School in Engineering, University of Tsukuba
**Institute of Engineering Mechanics, University of Tsukuba
Tsukuba 305, Japan

Abstract. An adaptive learning control scheme by use of multilayer neural networks for compensating for uncertainties in nonlinear dynamic system is examined. Multilayer neural networks are introduced to map the uncertainties in nonlinear dynamics and perform nonlinear state feedback. Parameters of neural networks are adjusted by conventional back-propagation algorithms modified with the projection operation. Effectiveness of the proposed scheme for tracking control are demonstrated through computer simulations.

Keywords. Nonlinear System, Neural Networks, Uncertainty, Adaptive Learning Control

1. Introduction

Multilayer neural networks have been widely used for controlling nonlinear systems because they can approximate arbitrary nonlinear functions, and have the ability of parallel distributed processing which can be easily implemented by hardware. Various control methods have been proposed for neural networks. For example, a neural control model using inverse dynamics for manipulators has been first developed by Kawato et al(1987). Identification and control problems of complex nonlinear dynamic systems using neural networks have been studied extensively by Narendra and Parthasarathy(1990), Polycarpou and Ioannou(1992). A direct adaptive control method for nonlinear system by gaussian neural networks has been proposed by Sanner and Slotine(1992). Neural networks also have been used to adaptive control problems of MIMO nonlinear dynamic systems such as manipulators by Liu and Chen(1993). In the case of adaptive control, the unknown the nonlinearities are modeled by the multilayer neural networks through off-line identification. After the identification, the neural networks are used to make the nonlinear feedback. The parameters of neural networks are adaptively adjusted so as to minimize the tracking control.

When neural networks are used to identify the nonlinear dynamical system, a basic assumption is that the dynamic system is stable so as to construct the stable identifying structure so far. However, some actual nonlinear dynamical systems such as robot manipulators perhaps don't satisfy this assumption. When system is unstable, the control is needed to be made for assuring the stability of system. In this case, difficulties for identification will be increased because the property of input signals is limited. In order to avoid the difficulties to identify the unstable system using neural networks, we have proposed an adaptive learning control method using neural networks to compensating for uncertainties in manipulator(1994). In this paper, we have attempted to make some improvements to our method and extend this method to more general nonlinear system.

2. Neural Networks

Artificial neural networks consist of many interconnected simple processing elements, which have a number of inputs and a single output. The output of each element is determined a nonlinear function of weighted sum of all inputs. The nonlinear function is usually called activation function which is non-decreasing such as sigmoid function or hyperbolic tangent function. One of the most striking feature of a multilayer feedforward neural network is its ability to approximate any arbitrary continuous nonlinear function uniformly on the compact sets with desired accuracy. This property has been rigorously proved by Funahashi(1989) and Hornik(1989). When neural networks are used to approximate the nonlinear function, it is necessary to adjust the interconnected weights and threshold value of each element by learning. A typical learning is called the supervised learning or learning with a teacher. The back-propagation(BP) algorithm is a method of adjusting the parameters of neural networks in the supervised learning, which has been discussed in detail elsewhere in the literature[8]. In this work, it will be briefly described in the following.

Suppose $f_i(x, w)$ is the ith output of two layer feedforward neural network with n neurons in the output layer , p neurons in the hidden layer, and m input patterns. Then it can be expressed as

$$f_i(x,w) = s_i^2\Big(\sum_{j=1}^{p} w_{ij}^2 s_j^1\Big(\sum_{k=1}^{m} w_{jk}^1 x_k + v_j^1\Big) + v_i^2\Big) \quad (i = 1, \ldots, n) \tag{1}$$

where w is adjustable parameter vector of neural network, represented the weights w_{ij}^2, w_{jk}^1 and threshold values v_i^2, v_j^1, $s_i^j(\cdot)$ is the activation function, such as hyperbolic tangent

function

$$s_i^j(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}} \tag{2}$$

The superscripts represent the number of the corresponding layer of neural networks. Suppose the teaching data is $u_t(t = 1, ..., n)$, the instantaneous cost function is defined as

$$J(t, w) = \frac{1}{2}[f(x_t, w) - u_t]^2 \tag{3}$$

The parameter update is given by

$$w(t+1) = w(t) - \alpha \frac{\partial J}{\partial w} \tag{4}$$

where α is the learning rate, usually chosen as a small positive constant. The gradient of cost function $J(w)$ with respected to the parameter w is determined by the following.

For the hidden-to-output connections:

$$\begin{aligned} \frac{\partial J(t)}{\partial w_{ij}^2} &= \delta_i^2 s_j^1, \qquad \frac{\partial J(t)}{\partial v_i^2} = \delta_i^2 \\ \delta_i^2 &= 2[f_i(t, w) - u_t] f_i' \end{aligned} \tag{5}$$

For the input-to-hidden connections:

$$\begin{aligned} \frac{\partial J(t)}{\partial w_{jk}^1} &= \delta_j^1 x_k, \qquad \frac{\partial J(t)}{\partial v_j^1} = \delta_j^1 \\ \delta_j^1 &= s'^1_j \sum_{i=1}^{n} \delta_i^2 w_{ij}^2 \end{aligned} \tag{6}$$

The equation(5),(6) usually called δ rule or back-propagation algorithm.

3. Adaptive Learning Control Method

Consider the MIMO nonlinear dynamical system with p inputs and p outputs described by

$$\begin{aligned} \dot{x}_{11} &= f_1(x) + g_{11}(x)u_1 + \ldots + g_{1p}(x)u_p \\ &\vdots \\ \dot{x}_{p1} &= f_p(x) + g_{p1}(x)u_1 + \ldots + g_{pp}(x)u_p \end{aligned} \tag{7}$$

where $u \in R^p$ is the vector of a control input, $f_i(x), g_{ij}(x) \in R^m (i = 1, \ldots, p; j = 1, \ldots, p)$ are smooth vector.The state variables satisfy the following relations:

$$\begin{aligned} \dot{x}_{1r} &= x_{1(r-1)} \\ \dot{x}_{1(r-1)} &= x_{1(r-2)} \\ &\vdots \\ \dot{x}_{12} &= x_{11} \\ &\vdots \\ \dot{x}_{pr} &= x_{p(r-1)} \\ \dot{x}_{p(r-1)} &= x_{p(r-2)} \\ &\vdots \\ \dot{x}_{p2} &= x_{p1} \end{aligned} \tag{8}$$

So the state vector is $x \in R^m$ and $m = p \times r$. Define

$$g(x) = \begin{bmatrix} g_{11}(x) & \ldots & g_{1p}(x) \\ \vdots & & \vdots \\ g_{p1}(x) & \ldots & g_{pp}(x) \end{bmatrix} \tag{9}$$

Then the equation(7) can be represented as

$$\begin{bmatrix} \dot{x}_{11} \\ \vdots \\ \dot{x}_{p1} \end{bmatrix} = \begin{bmatrix} f_1(x) \\ \vdots \\ f_p(x) \end{bmatrix} + g(x) \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \tag{10}$$

Assume $g(x)$ is bounded away from singularity, i.e. the inversion of $g(x)$ exist and has bounded norm over a compact set $S \in R^m$.

If $f_i(x), g_{ij}(x)$ are known and above assumption is satisfied, the control law can be defined as

$$\begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} = g^{-1}(x) \left(\begin{bmatrix} r_1 \\ \vdots \\ r_p \end{bmatrix} - \begin{bmatrix} f_1(x) \\ \vdots \\ f_p(x) \end{bmatrix} \right) \tag{11}$$

Then the linearized system becomes

$$\begin{bmatrix} \dot{x}_{11} \\ \vdots \\ \dot{x}_{p1} \end{bmatrix} = \begin{bmatrix} r_1 \\ \vdots \\ r_p \end{bmatrix} \tag{12}$$

Let the reference control inputs $r_1, ..., r_p$ be defined as

$$\begin{aligned} r_1 &= x_{m1r}^r + \alpha_{11}(x_{m1r}^{r-1} - x_{1r}^{r-1}) \\ &\quad + \ldots + \alpha_{1(r-1)}(x_{m1r} - x_{1r}) \\ &\vdots \\ r_p &= x_{mpr}^r + \alpha_{p1}(x_{mpr}^{r-1} - x_{pr}^{r-1}) \\ &\quad + \ldots + \alpha_{p(r-1)}(x_{mpr} - x_{pr}) \end{aligned} \tag{13}$$

where $x_{m1r}, \ldots, x_{mpr}$ are the reference trajectories. Define

$$\begin{aligned} e_1 &= (x_{m1r} - x_{1r}) \\ &\vdots \\ e_p &= (x_{mpr} - x_{pr}) \end{aligned} \tag{14}$$

then the following error dynamical equations can be obtained by

$$\begin{aligned} e_1^r + \alpha_{11} e_1^{r-1} + \ldots + \alpha_{1(r-1)} &= 0 \\ &\vdots \\ e_p^r + \alpha_{p1} e_1^{r-1} + \ldots + \alpha_{p(r-1)} &= 0 \end{aligned} \tag{15}$$

It is clear that the $e_1, \ldots, e_p$ will be approach zero if the coefficients $\alpha_{ij} (i = 1, ..., p; j = 1, ..., r-1)$ are chosen such that all polynomials in (15) are Hurwitz.

If the nominal dynamics $f_0(x), g_0(x)$ of the nonlinear system are known and there exist only the uncertainties $\Delta f(x), \Delta g(x)$, we can use the multilayer neural networks $\hat{f}_1(x, w_1), \ldots, \hat{f}_p(x, w_p)$ and $\hat{g}_{11}(x, v_{11}), \ldots, \hat{g}_{pp}(x, v_{pp})$ to approximate the nonlinear functions $\Delta f_1(x), \ldots, \Delta f_p(x)$ and $\Delta g_{11}(x), \ldots, \Delta g_{pp}(x)$. Here $w_1, \ldots, w_p$, and $v_{11}, \ldots, v_{pp}$ denote the parameters(weights,threshold value) of each neural network respectively. Therefore the system can be modeled

as

$$\begin{bmatrix} \hat{\dot{x}}_{11} \\ \vdots \\ \hat{\dot{x}}_{p1} \end{bmatrix} = \begin{bmatrix} f_{01}(x, w_1) \\ \vdots \\ \hat{f}_{0p}(x, w_p) \end{bmatrix} + \begin{bmatrix} \hat{f}_1(x, w_1) \\ \vdots \\ \hat{f}_p(x, w_p) \end{bmatrix} + [g_0(x) + \hat{g}(x, v)] \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \quad (16)$$

The control law can be constructed by

$$\begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} = [g_0(x) + \hat{g}(x, v)]^{-1} \left(\begin{bmatrix} r_1 \\ \vdots \\ r_p \end{bmatrix} - \begin{bmatrix} f_{01}(x, w_1) \\ \vdots \\ f_{0p}(x, w_p) \end{bmatrix} - \begin{bmatrix} \hat{f}_1(x, w_1) \\ \vdots \\ \hat{f}_p(x, w_p) \end{bmatrix} \right) \quad (17)$$

where

$$\hat{g}(x, v) = \begin{bmatrix} \hat{g}_{11}(x, v_{11}) & \cdots & \hat{g}_{1p}(x, v_{1p}) \\ \vdots & & \vdots \\ \hat{g}_{p1}(x, v_{p1}) & \cdots & \hat{g}_{pp}(x, v_{pp}) \end{bmatrix} \quad (18)$$

From the equation(17), through the simple computation, the system(10) will become

$$\begin{bmatrix} \dot{x}_{11} \\ \vdots \\ \dot{x}_{p1} \end{bmatrix} = \begin{bmatrix} f_1(x) - \hat{f}_1(x, w_1) \\ \vdots \\ f_p(x) - \hat{f}_p(x, w_p) \end{bmatrix} + [g(x) - \hat{g}(x, v)] \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} + \begin{bmatrix} r_1 \\ \vdots \\ r_p \end{bmatrix} \quad (19)$$

By using the definition(13) of reference control inputs, the equation(19) can be written by

$$\begin{bmatrix} e_1^r + \alpha_{11} e_1^{r-1} + \ldots + \alpha_{1(r-1)} \\ \vdots \\ e_p^r + \alpha_{p1} e_1^{r-1} + \ldots + \alpha_{p(r-1)} \end{bmatrix} = \begin{bmatrix} \hat{f}_1(x, w_1) - f_1(x) \\ \vdots \\ \hat{f}_p(x, w_p) - f_p(x) \end{bmatrix} + [\hat{g}(x, v) - g(x)] \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \quad (20)$$

The right side of equation(20) represent the modeling error.It can be rewritten as

$$\begin{bmatrix} \hat{\dot{x}}_{11} - \dot{x}_{11} \\ \vdots \\ \hat{\dot{x}}_{p1} - \dot{x}_{p1} \end{bmatrix} = \begin{bmatrix} \hat{f}_1(x, w_1) - f_1(x) \\ \vdots \\ \hat{f}_p(x, w_p) - f_p(x) \end{bmatrix} + [\hat{g}(x, v) - g(x)] \begin{bmatrix} u_1 \\ \vdots \\ u_p \end{bmatrix} \quad (21)$$

It is clear that when the modeling error is very small or approaches zero the tracking error $e_1, \ldots, e_p$ will approach zero. In order to let the modeling error be minimized by adjusting the parameters of neural networks, a instantaneous cost function which shall be minimized with respect to the parameters of neural networks is defined as

$$J(w, v) = \frac{1}{2}[(\hat{x}_{11}(k) - x_{11}(k))^2 + \ldots + (\hat{x}_{p1}(k) - x_{p1}(k))^2]. \quad (22)$$

It should note that the computation of instantaneous error utilizes the serial-parallel identified model[6]. According to the back-propagation algorithm modified by projection operation, the parameters of neural networks are adjusted by

$$\begin{aligned} w(k+1) &= p(k); && if \quad |p(k)| < w_b \\ w(k+1) &= w_b \frac{p(k)}{|p(k)|}; && if \quad |p(k)| \geq w_b \\ v(k+1) &= z(k); && if \quad |z(k)| < v_b \\ v(k+1) &= v_b \frac{z(k)}{|z(k)|}; && if \quad |z(k)| \geq v_b \\ p(k) &:= w(k) - \alpha \frac{\partial J(w)}{\partial w} \\ z(k) &:= v(k) - \alpha \frac{\partial J(v)}{\partial v} \end{aligned} \quad (23)$$

where α is learning step size,usually chosen as a small positive constant, and w_b, v_b are the radium of parameters of neural networks used to constrain the size of parameters. Because the learning control is implemented online and the modeling error is a kind of disturbance, the stability of the whole adaptive learning control system can not be guaranteed apriori. The bigger modeling error may be cause the bigger adjustments of neural networks so that is is possible to damage the neural networks. This phenomenon is called *parameter drift*[7]. In order to prevent the *parameter drift* phenomenon, the projection operation is used to modify the back-propagation algorithm.

Because the control inputs are the function of the parameters of neural networks, The computation of the gradient of cost function $J(w, v)$ with respect to parameters of neural networks is usually very complex. In order to avoid the complexity, we utilized the concept of pattern learning to simplify the computation of gradient of $J(w, v)$. The control inputs and the state of system at each sampling time are viewed as a group of input pattern. When this input pattern are fed to system and model simultaneously, the outputs of system and model will be viewed a group of output pattern and used to adjusting the parameters of model. The adjusted nonlinear functions in identifying model of neural networks are used to form the nonlinear feedback and to construct the new controller at the next sampling time. Based on this consideration, the gradient of cost function $J(w)$ with respect to parameters of neural networks is computed by

$$\begin{aligned} \frac{\partial J(w)}{\partial w_i} &= [\hat{x}_{i1}(k) - x_{i1}(k)] \frac{\partial f_i(x, w_i)}{\partial w_i} \\ \frac{\partial J(w)}{\partial v_{jk}} &= [\hat{x}_{j1}(k) - x_{j1}(k)] u_k \frac{\partial g_{jk}(x, w_i)}{\partial v_{jk}} \end{aligned} \quad (24)$$

4. Simulations

Utilizing the above learning method, tracking control problems of two-link manipulator are examined by computer

simulations. the dynamics of two-link manipulator is described by

$$\left(\begin{bmatrix} m_{011}(q) & m_{012}(q) \\ m_{021}(q) & m_{022}(q) \end{bmatrix} + \begin{bmatrix} \Delta m_{11}(q) & \Delta m_{12}(q) \\ \Delta m_{21}(q) & \Delta m_{22}(q) \end{bmatrix}\right) \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} d_{01}(q,\dot{q}) \\ d_{02}(q,\dot{q}) \end{bmatrix} + \begin{bmatrix} \Delta d_1(q,\dot{q}) \\ \Delta d_2(q,\dot{q}) \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \quad (25)$$

where $m_{0ij}(q)$ is an known elements of inertia matrix, $\Delta m_{ij}(q)$ is unknown elements of inertia matrix, $d_{0i}(q,\dot{q})$ and $\Delta d_i(q,\dot{q})$ represent the known and unknown elements respectively of centrifugal and Coriolis force, viscous and coulomb friction force and gravity force. τ_i is the joint actuator torque. These elements are in detail given by

$$\begin{aligned}
m_{011}(q) &= I_{01} + I_{02} + m_{02}l_1^2 + 2m_{02}l_1h_{02}\cos(q_2) \\
m_{012}(q) &= m_{021}(q) = -I_{02} - m_{02}l_1h_{02}\cos(q_2) \\
m_{022}(q) &= I_{02} \\
d_{01}(q,\dot{q}) &= -2m_{02}l_1h_{02}\sin(q_2)\dot{q}_1\dot{q}_2 \\
&\quad +m_{02}l_1h_{02}\sin(q_2)\dot{q}_2^2 + h_{01}\cos(q_1)m_{01}g \\
&\quad +[l_1\cos(q_1) + h_{02}\cos(q_1 - q_2)]m_{02}g \\
d_{02}(q,\dot{q}) &= m_{02}l_1h_{02}\sin(q_2)\dot{q}_2^2 \\
&\quad -h_{02}\cos(q_1 - q_2)m_{02}g
\end{aligned} \quad (26)$$

and

$$\begin{aligned}
\Delta m_{11}(q) &= \Delta I_1 + \Delta I_2 + \Delta m_2 l_1^2 \\
&\quad +2l_1(m_{02}\Delta h_2 + \Delta m_2 h_2)\cos(q_2) \\
\Delta m_{12}(q) &= \Delta m_{21}(q) \\
&= -\Delta I_2 - l_1(m_{02}\Delta h_2 + \Delta m_2 h_2)\cos(q_2) \\
\Delta m_{22}(q) &= \Delta I_2 \\
\Delta d_1(q,\dot{q}) &= -2l_1(m_{02}\Delta h_2 + \Delta m_2 h_2)\sin(q_2)\dot{q}_1\dot{q}_2 \\
&\quad +l_1(m_{02}\Delta h_2 + \Delta m_2 h_2)\sin(q_2)\dot{q}_2^2 + v_1\dot{q}_1 \\
&\quad +c_1\mathrm{sgn}(\dot{q}_1) + (m_{01}\Delta h_1 + \Delta m_1 h_1)g\cos(q_1) \\
&\quad +[l_1\cos(q_1) + h_2\cos(q_1 - q_2)]\Delta m_2 g \\
&\quad +\Delta h_2\cos(q_1 - q_2)m_{02}g \\
\Delta d_2(q,\dot{q}) &= l_1(m_{02}\Delta h_2 + \Delta m_2 h_2)\sin(q_2)\dot{q}_2^2 \\
&\quad +v_2\dot{q}_2 + c_2\mathrm{sgn}(\dot{q}_2) \\
&\quad -(m_{02}\Delta h_2 + \Delta m_2 h_2)g\cos(q_1 - q_2)
\end{aligned} \quad (27)$$

The parameters used in simulation are shown in table 1.

Reference tracking signals are given by

$$\begin{aligned}
q_{d1} &= \sin(2\pi t/4) + \cos(2\pi t/8) \\
q_{d2} &= \cos(2\pi t/4) + \sin(2\pi t/8).
\end{aligned} \quad (28)$$

The constant in the equation (15) are designed to generate second order underdamped system. Six multilayer neural networks are used to map the uncertain nonlinear dynamics due to friction force and changes of the inertia . Neural networks have the structures of three layers:

$$\begin{aligned}
\Delta f_{n1}, \Delta f_{n2} &: \mathcal{N}_{4,20,10,1} \\
\Delta g_{n11} \sim \Delta g_{n22} &: \mathcal{N}_{1,20,20,1}
\end{aligned}$$

parameters of each neural network are initialized with random values distributed uniformly in the interval [-1, 1]. The output layers of neural networks f_n and g_n are designed to be linear .The linear combining coefficients in the output layer of neural networks f_n and g_n are designed to be 6 and 10 respectively.Learning step size is 0.005. Sampling period for simulation is 0.01 second. Each period of tracking control is designed as 16 second. The parameter adjustment of neural networks in controller is delayed by one sampling time in comparison with the parameter adjustment of neural networks in the dynamical model. After the learning procedure has been implemented by 100 times, simulation results of tracking control of joint1 and joint2 are shown in Fig.1 and 2. The records of modeling error of joint1 and joint2 during the learning procedure of the last 100 times are shown in the Fig.3.

Table 1:

m_{01}	6	kg
m_{02}	1.8	kg
l_1	0.25	m
l_2	0.15	m
h_{01}	0.06	m
h_{02}	0.085	m
Δm_1	0.5	kg
Δm_2	1.0	kg
Δh_1	0.01	m
Δh_2	0.04	m
Δv_1	0.015	Nms/rad
Δv_2	0.028	Nms/rad
Δc_1	0.02	Nm
Δc_2	0.045	Nm

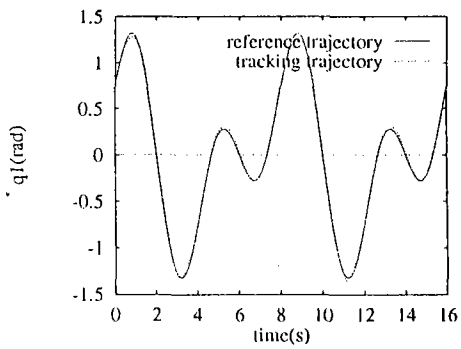

Figure 1: angle displacement of joint1

5. Conclusions

A adaptive learning control method to compensate for the uncertainties in the nonlinear dynamical system has been examined. The multilayer neural networks and the nominal dynamics of the nonlinear dynamical system are used to construct the model of nonlinear dynamical system which contains the uncertainties. The modelling error is minimized by

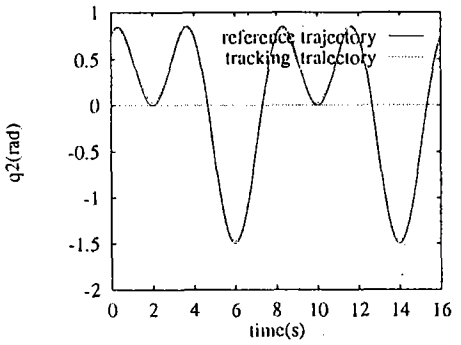

Figure 2: angle displacement of joint2

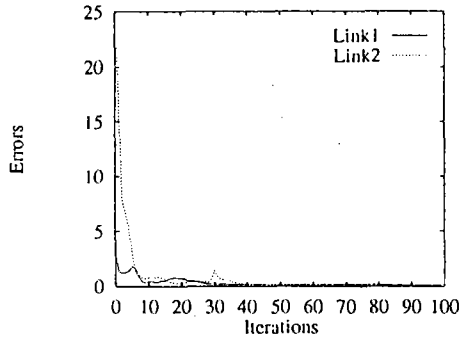

Figure 3: tracking error

the online learning. The neural networks are used to perform the nonlinear feedback so as to force the tracking error to approach the zero. Computer simulations for a two-link manipulator have shown positive results of tracking control when there exist the uncertainties in the manipulator's dynamics.

References

[1] Funahashi,K. (1989) *On the approximate realization of continuous mapping by neural networks.* Neural Networks, 2, 183-192.

[2] Graig,J.J. (1986). *Introduction to robotics: Mechanics & Control.* Addison-Wesley, London.

[3] Hornik,K.,Stinchcombe,M., and White,H. (1989). *Multilayer feedforward networks are universal approximators.* Neural Networks, 2, 359-366.

[4] Kawato,M., Furukawa,K., and Suzuki,R. (1987). *A hierarchical neural network model for control and learning of voluntary movement.* Biological Cybernetics, 57, 169-185.

[5] Liu,C.C., and Chen,F.C (1993). *Adaptive control of nonlinear continuous-time systems using neural networks-general relative degree and MIMO cases.* Int.J.Control, 58(2), 317-335.

[6] Narendra,K.S., and Parthasarathy,K. (1990). *Identification and control of dynamical systems using neural networks.* IEEE Tran.on Neural Networks, 1(1), 4-27.

[7] Polycarpou,M.M., and Ioannou,P. (1992). *Modeling, identification and stable adaptive control of continuous-time nonlinear dynamical system using neural networks.* Proc. American Control Conf., 2, 36-40.

[8] Rumlhart,D. Hinton,D., and Williams,G. (1986). *"Learning internal representations by error propagation,"* in Parallel Distributed Processing, 1, MA: The MIT Press.

[9] Zhang,P., Sankai,Y., and Ohta,M. (1994). *Compensation of uncertainties in manipulators by use of neural networks.* Proc. First Asian Control Conf., 3, 555-558.