TA09 11:20

Proceedings of the 34th
Conference on Decision & Control
New Orleans, LA - December 1995

Robust Adaptive Control of Robots Using Neural Network: Global Tracking Stability

C. M. Kwan *, D. M. Dawson **, and F. L. Lewis *

* Automation and Robotics Research Institute
The University of Texas at Arlington
7300 Jack Newell Blvd. S
Fort Worth, Texas 76118

** Department of Electrical and Computer Engineering
Center for Advanced Manufacturing
Clemson University
Clemson, SC 29634.

Research supported by NSF grant IRI-9216545

Abstract

A desired compensation adaptive law-based neural network (DCAL-NN) controller is proposed for the robust position control of rigid-link robots. The NN is used to approximate a highly nonlinear function. The controller can guarantee the global asymptotic stability of tracking errors and boundedness of NN weights. In addition, the NN weights here are tuned on-line, with no off-line learning phase required. When compared with standard adaptive robot controllers, we do not require persistent excitation conditions, linearity in the parameters, or lengthy and tedious preliminary analysis to determine a regression matrix. The controller can be regarded as a universal reusable controller because the same controller can be applied to any type of rigid robots without any modifications.

1. Introduction

Recently, many researchers have begun to focus on implementation issues such as the reduction of on-line computations [1] [11] [15] [19]. For example, it is well known that one of the disadvantages of the adaptive robot controllers is that the regression matrix used as feedforward compensation must be determined by extensive preliminary analysis, and evaluated on-line to compute certain nonlinear functions. Moreover, some terms in the robot dynamics are not necessarily linear in the parameters (e.g. friction is actually described by complex nonlinear functions). Such concerns caused people to rethink the previous theoretical development of robot controllers.

To eliminate the need for on-line evaluation of the regression matrix, Sadegh and Horowitz [19] proposed the desired compensation adaptive law (DCAL). The idea is to make use of the desired joint trajectory and replace the actual joint angles, velocities with the desired joint angles, velocities in the evaluation of the regression matrix. Therefore, since the desired trajectory is usually given, the regression matrix can be evaluated off-line and hence on-line computational demand can be greatly reduced. Although this scheme is quite simple, some tedious and lengthy preliminary dynamical analysis is still needed to get the regression matrix. This preliminary effort can be quite cumbersome for the case of robots with multiple degrees of freedom.

Neural network (NN) can be used for approximation of nonlinear systems, for classification of signals, and for associative memories. For control engineers, the approximation capability of NN is usually used for system identification [3] [8], or indirect "identification-based" control [2] [13] [14] [16]. However there is very little about the use of NN in direct closed-loop controllers that yield guaranteed performance.

Problems that remain to be addressed in NN research include ad hoc controller structures and the inability to guarantee satisfactory performance of the system in terms of small tracking errors and bounded NN weights. Uncertainty on how to initialize the NN weights leads to the necessity for "preliminary off-line tuning" [5]. Some of these problems have by now been addressed [11] [12] [17] [18].

In this paper, we propose a new robust robot control scheme by combining the theory of Neural Network (NN) and DCAL. The main advantage is that the NN controller is the same irrespective of the kind of rigid-link robot under control. Hence it can be termed a universal reusable controller since no preliminary dynamical analysis, which can be quite cumbersome for robots with multiple degrees of freedom, is needed. We will also confront some of the deficiencies of NN mentioned earlier for the full nonlinear 3-layer NN with arbitrary activation functions (as long as the function satisfies an approximation property, and it and its derivatives are bounded). Our NN weights here are tuned on-line, with no off-line learning phase required. Most importantly, we can guarantee the global asymptotic stability of joint position tracking error and the boundedness of NN weight updates. When compared with standard adaptive robot controllers, we do not require persistent excitation conditions, linearity in the parameters, and the evaluation of regression matrix. We note that all the NN controllers referred to above only guarantee, at best, local stability.

The paper is organized as follows. In Section 2 we will review some basics of neural networks, the rigid robot model and its properties. Then in Section 3, we will introduce our DCAL-NN controller. A detailed 2-link robot motion tracking example will be presented in Section 4 to illustrate our design. Finally conclusions will be included in Section 5.

2. Mathematical Preliminaries

Let R denote the real numbers, R^n the real n-vectors, $R^{m\times n}$ the real mxn matrices. Let S be a compact simply connected set of R^n. With map $f: S \rightarrow R^m$, define $C^m(S)$ the space such that f is continuous. We denote by $\|\bullet\|$ any suitable vector norm. When it is required to be specific we denote the p-norm by $\|\bullet\|_p$. The supremum norm of f(x) (over S) is defined as

$$\sup \|f(x)\|, \quad f: S \rightarrow R^m, \quad \forall x \in S. \qquad (2.1)$$

Given $A = [a_{ij}]$, $B \in R^{m \times n}$ the Frobenius norm is defined by

$$\|A\|_F^2 = \mathrm{tr}(A^T A) = \sum_{i,j} a_{ij}^2 \qquad (2.2)$$

with tr(•) the trace. The associated inner product is $<A, B>_F = \mathrm{tr}(A^T B)$. The Frobenius norm is compatible with the 2-norm so that $\|Ax\|_2 \le \|A\|_F \|x\|_2$, with $A(t) \in R^{m \times n}$ and $x \in R^n$.

2.1 Neural Networks

Given an input vector x in R^{N_1}, a three-layer neural net (NN) has an output given by

$$y_i = \sum_{j=1}^{N_2} \left[w_{ij} \sigma\left[\sum_{k=1}^{N_1} v_{jk} x_k + \theta_{vj} \right] + \theta_{wi} \right], \qquad i = 1, \dots, N_3 \qquad (2.3)$$

with $\sigma(\bullet)$ the activation function, v_{jk} the first-to-second layer interconnection weights, and w_{ij} the second-to-third layer interconnection weights. θ_{vm}, θ_{wm}, m = 1, 2,..., are called the threshold offsets and the number of neurons in layer l is N_l, with N_2 the number of hidden-layer neurons. A three-layer neural network is shown in Fig. 2. 1.

The NN equation may be conveniently expressed in matrix format by defining $x = [x_0\ x_1 \dots x_{N_1}]^T$, $y = [y_1\ y_2 \dots y_{N_3}]^T$, and weight matrices $W^T = [w_{ij}]$, $V^T = [v_{jk}]$. Including $x_0 = 1$ in x allows one to include the threshold vector $[\theta_{v1}\ \ \theta_{v2}\ \dots\ \theta_{vN_1}]$ as the first column of V^T. Hence the NN outputs can be compactly written as

$$y = W^T \sigma(V^T x) \qquad (2.4)$$

where, if $z = [z_1\ \ z_2\ \dots]^T$ is a vector we define $\sigma(z) = [\sigma(z_1)\ \ \sigma(z_2) \dots]^T$. Including 1 as a first term in the vector $\sigma(V^T x)$ allows one to incorporate the thresholds θ_{wj} as the first column of W^T. Any tuning of W and V then includes tuning of the thresholds as well as the weights.

A general function $f(x) \in C^m(S)$, $x(t) \in R^n$, can be written as

$$f(x) = W^T \sigma(V^T x) + \varepsilon(x) \qquad (2.5)$$

with $N_1 = n$, $N_3 = m$, and $\varepsilon(x)$ a *NN functional reconstruction error* vector.

2.2 Robot Model and Its Properties

We will assume that the robot is an n-link, serially connected, rigid-link revolute robot the following dynamic model [10]

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + T_d = \tau \qquad (2.7)$$

with $q, \dot{q}, \ddot{q} \in R^n$ denoting the link position, velocity, and acceleration vectors, respectively, $M(q) \in R^{n \times n}$ the inertia matrix, $V_m(q,\dot{q}) \in R^n$ the centripetal-Coriolis matrix, $G(q) \in R^n$ the gravity vector, $F \in R^n$ the friction terms, $T_d \in R^n$ the additive bounded disturbance, $\tau \in R^n$ the torque input vector. The rigid dynamics (2.7) has the following properties [4]. Note that linearity in the parameters assumption is not needed here.

Property 1: Boundedness of the inertia matrix

The inertia matrix M(q) is symmetric and positive definite, and satisfies the following inequalities

$$m_1 \|y\|^2 \le y^T M(q) y \le m_2 \|y\|^2, \qquad \forall y \in R^n \qquad (2.8)$$

where m_1 and m_2 are known positive constants, and $\|\bullet\|$ denotes the standard Euclidean norm.

Property 2: Skew symmetry

The inertia and centripetal-Coriolis matrices have the following property

$$y^T\left(\frac{1}{2}\dot{M}(q) - V_m(q,\dot{q})\right)y = 0, \qquad \forall y \in R^n \qquad (2.9)$$

where $\dot{M}(q)$ is the time derivative of the inertia matrix.

Property 3: Neural Network Approximator

We use a three layer NN to approximate a highly nonlinear robot function, i.e.

$$f_d(q_d, \dot{q}_d, \ddot{q}_d) = W^T \sigma\left(V^T x\right) + \varepsilon(x), \qquad (2.10)$$

where

$$f_d(q_d, \dot{q}_d, \ddot{q}_d) = M(q_d)\ddot{q}_d + V_m(q_d,\dot{q}_d)\dot{q}_d + G(q_d) + F(\dot{q}_d) \qquad (2.11)$$

$$x = \left[q_d^T\ \ \dot{q}_d^T\ \ \ddot{q}_d^T\right]^T, \qquad (2.12)$$

and W, V are constants. Throughout this paper, we will assume the desired trajectory, denoted by $q_d, \dot{q}_d, \ddot{q}_d \in R^n$, is bounded which implies that we are guaranteed that

$$\|\varepsilon(x)\| \le \varepsilon_b, \qquad (2.13)$$

where ε_b is a positive constant.

3. DCAL-NN Controller

The control objective is to develop a link position tracking controller for the robot dynamics given by (2.7) based on inexact knowledge of manipulator dynamics. To accomplish this purpose we first define the link position tracking error $e(t) \in R^n$ as

$$e = q_d - q \qquad (3.1)$$

where $q_d(t) \in R^n$ denotes the desired link position trajectory. In addition, we also define a filtered tracking error as

$$r = \dot{e} + \Lambda e \qquad (3.2)$$

where $\Lambda \in R^{n \times n}$ is a diagonal, positive definite control gain matrix. Using (3.2) and (2.7), we can derive the equation

$$M\dot{r} = f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d) - V_m r + T_d - \tau, \qquad (3.3)$$

where the highly nonlinear robot function $f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d)$ is defined by

$$f(\bullet) = M(q)(\ddot{q}_d + \Lambda\dot{e}) + V_m(q,\dot{q})(\dot{q}_d + \Lambda e) + G(q) + F(\dot{q}). \qquad (3.4)$$

3.1 Bounding Assumptions and Facts

For notational convenience we define the matrix of all weights as

$$Z = \mathrm{diag}\{W,V\} \qquad (3.5)$$

Also define

$$\widetilde{V} = V - \widehat{V}, \quad \widetilde{W} = W - \widehat{W}, \text{ and } \widetilde{Z} = Z - \widehat{Z} \qquad (3.6)$$

with $\widehat{V}$, $\widehat{W}$ the estimates of the ideal weight values as provided by the weight tuning algorithms (to be discussed).

Two standard assumptions, which are quite common in the robotics literature [10] and neural networks [11], are stated next. Two useful facts will be listed, which can be easily proved using these assumptions.

Assumption 1: The ideal weights are bounded by positive constants V_M, W_M, and Z_M so that

$$\|V\|_F \le V_M,\ \|W\|_F \le W_M, \text{ or } \|Z\|_F \le Z_M.$$

Assumption 2: The desired trajectory is bounded, i.e.

$$\|x\| \le Q_d$$

where x is defined in (2.12) and Q_d is positive scalar constant.

The hidden-layer output error for a given x is given by

$$\tilde{\sigma} = \sigma - \hat{\sigma} = \sigma(V^T x) - \sigma(\hat{V}^T x). \qquad (3.7)$$

The Taylor series expansion of σ in (3.7) for a given x may be written as

$$\sigma(V^T x) = \sigma(\hat{V}^T x) + \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \quad (3.8)$$

with $\sigma' = \left|\frac{d\sigma(z)}{dz}\right|_{z=\hat{z}}$ and $O(z)^2$ denoting higher order terms. Denoting $\hat{\sigma}' = \sigma'(\hat{V}^T x)$, we have

$$\tilde{\sigma} = \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2.$$

Noting that

$$O(\tilde{V}^T x)^2 = [\sigma(V^T x) - \sigma(\hat{V}^T x)] - \hat{\sigma}'\tilde{V}^T x, \quad (3.9)$$

we can then show the following fact for the higher-order term $O(\bullet)^2$.

Fact 1: For sigmoid and tanh activation functions, the higher-order term in the Taylor series are bounded by

$$\|O(\tilde{V}^T x)^2\| \le c_1 + c_2\|\tilde{V}\|_F \qquad (3.10)$$

where c_i are computable positive constants. The proof is given in an Appendix.

3.2 Controller Structure and Error Dynamics

Let a NN be given such that (2.10) holds with $\|\varepsilon(x)\| < \varepsilon_N$ where ε_N is a positive constant. According to the well known NN approximation property [6] and [9], since x is bounded, (2.10) always holds globally. We select the control input for (3.3) as

$$\tau = \hat{W}^T \sigma(\hat{V}^T x) + k_v r + k_p e + v_R \qquad (3.11)$$

with $\hat{W}$, $\hat{V}$ the dynamic estimate of W, V. Vectors e and r are defined by (2.10), (3.1), (3.2), respectively. $k_v \in R^{n\times n}$, $k_p \in R^{n\times n}$ are diagonal, positive definite gain matrices. The nonlinear term $v_R(t)$ to be defined later is used to compensate for error functions that provides robustness in the face of higher order term in the Taylor series, functional reconstruction error and bounded disturbance T_d. The controller structure is shown in Fig. 3.1.

Substituting controller (3.11) into (3.3) gives the closed-loop dynamics

$$M\dot{r} = f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d) - (k_v + V_m)r - \hat{W}^T\sigma(\hat{V}^T x) - k_p e - v_R + T_d.$$

Adding and subtracting $f_d(q_d,\dot{q}_d,\ddot{q}_d)$ defined in (2.11) into the above expression yields

$$M\dot{r} = f(\bullet) - f_d(\bullet) + f_d(\bullet) - (k_v + V_m)r - \hat{W}^T\sigma(\hat{V}^T x) - k_p e - v_R + T_d.$$

Now using the NN approximation property (2.10) gives

$$M\dot{r} = f(\bullet) - f_d(\bullet) + W^T\sigma(V^T x) + \varepsilon(x) - (k_v + V_m)r - \hat{W}^T\sigma(\hat{V}^T x) + T_d - k_p e - v_R \qquad (3.12)$$

Expanding $\sigma(V^T x)$ in (3.12) into Taylor series and manipulating the resulting expression yields

$$M\dot{r} = -(k_v + V_m)r - k_p e + \tilde{W}^T\hat{\sigma} + \hat{W}^T\hat{\sigma}'\tilde{V}^T x + w - v_R, \qquad (3.13)$$

where

$$w = f(\bullet) - f_d(\bullet) + \tilde{W}^T\hat{\sigma}'\tilde{V}^T x + W^T O(\tilde{V}^T x)^2 + T_d + \varepsilon(x). \qquad (3.14)$$

Fact 2: The disturbance term w(t) in (3.14) is bounded according to

$$\|w(t)\| \le \rho(z,\hat{W},\hat{V}), \qquad (3.15)$$

where the positive scalar function $\rho(\bullet)$ is defined by

$$\rho(z,\hat{W},\hat{V}) = \zeta_0 + \zeta_1\|z\| + \zeta_2\|z\|^2 + \zeta_3\|\hat{W}\|_F + \zeta_4\|\hat{V}\|_F + \zeta_5\|\hat{V}\|_F\|\hat{W}\|_F = S\varphi \qquad (3.16)$$

ζ_i's are positive bounding constants that depend on the desired trajectory, the physical properties of the robot (i.e., link masses, link lengths, friction coefficients, etc.), the disturbance bound, the error bound, and on the control gain matrix Λ of (3.2). The vectors z, S, φ used in (3.15) and (3.16) are defined as follows

$$z = [e^T \ \ r^T],$$

$$S = \left[1 \ \ \|z\| \ \ \|z\|^2 \ \ \|\hat{W}\|_F \ \ \|\hat{V}\|_F \ \ \|\hat{W}\|_F\|\hat{V}\|_F\right],$$

$$\varphi = [\zeta_0 \ \ \zeta_1 \ \ \zeta_2 \ \ \zeta_3 \ \ \zeta_4 \ \ \zeta_5]^T.$$

This fact is proven in an Appendix by using results in [10] and noting the fact that w(t) in (3.15) depends on $\hat{W}$, $\hat{V}$, and their product.

The robustifying term v_R in the controller (3.11) is given by

$$v_R = \frac{r(S\hat{\varphi})^2}{(S\hat{\varphi})\|r\| + \delta}, \qquad (3.17)$$

where

$$\dot{\delta} = -\gamma\delta, \quad \delta(0) = \text{design constant} > 0,\ \gamma > 0.$$

The parameter tuning law for φ is chosen as

$$\dot{\hat{\varphi}} = \Gamma S^T\|r\| = -\dot{\tilde{\varphi}}, \qquad (3.18)$$

where Γ is a symmetric and positive definite matrix and $\tilde{\varphi} = \varphi - \hat{\varphi}$. The stability analysis will be given next.

3.3 Weight Updates for NN

Theorem

Let the desired trajectory be bounded (Assumption 2). Let the control input be given by (3.11), (3.17), (3.18) and NN weight tuning provided by

$$\dot{\hat{W}} = F\hat{\sigma}r^T, \qquad (3.19a)$$

$$\dot{\hat{V}} = Gx(\hat{\sigma}'^T\hat{W}r)^T \qquad (3.19b)$$

with F, G some constant symmetric and positive definite matrices. Then the tracking errors e, $\dot{e}$ go to zero and the weight estimates $\hat{W}$, $\hat{V}$ are bounded.

Proof:

Consider the following Lyapunov function candidate

$$L = \frac{1}{2}r^T M r + \frac{1}{2}e^T k_p e + \frac{1}{2}\mathrm{tr}(\tilde{W}^T F^{-1}\tilde{W}) + \frac{1}{2}\mathrm{tr}(\tilde{V}^T G^{-1}\tilde{V}) + \frac{1}{2}L_R, \quad (3.20)$$

where

$$L_R = \frac{1}{2}\tilde{\varphi}^T\Gamma^{-1}\tilde{\varphi} + \frac{1}{2}\frac{\delta}{\gamma}.$$

Differentiating (3.20) and using (3.13) yields

$$\dot{L} \le -z^TQz + \dot{L}_1 + \dot{L}_2,$$

where

$$z = [e^T \quad r^T]^T,$$

$$Q = \begin{bmatrix} k_p\Lambda & -\frac{k_p}{2} \\ -\frac{k_p}{2} & k_v \end{bmatrix} > 0 \text{ for suitable choices of } k_p, k_v,$$

$$\dot{L}_1 = \|r\|S\hat{\varphi} - r^Tv_R - \delta, \qquad (3.21a)$$

$$\dot{L}_2 = tr[\tilde{W}^TF^{-1}\dot{\tilde{W}} + \tilde{W}^T\hat{\sigma}r^T] + tr[\tilde{V}^TG^{-1}\dot{\tilde{V}} + \tilde{V}^Tx(\hat{\sigma}'^T\hat{W}r)^T] \qquad (3.21b)$$

Using (3.17) in (3.21a) yields

$$\dot{L}_1 = \|r\|S\hat{\varphi} - \frac{r^Tr(S\hat{\varphi})^2}{(S\hat{\varphi})\|r\| + \delta} - \delta$$

$$\le \frac{\delta\|r\|S\hat{\varphi}}{(S\hat{\varphi})\|r\| + \delta} - \delta$$

$$\le 0.$$

Substituting the weight update rules (3.19) into (3.21b) gives

$$\dot{L}_2 = 0. \qquad (3.22)$$

Combining the above results, we have

$$\dot{L} \le -z^TQz, \quad Q > 0 \qquad (3.23)$$

(3.23) implies L, z, e, r, $\hat{W}$, $\hat{V}$, and $\hat{\varphi}$ are all bounded. From the form of (3.23), it is easy to show that z(t) is square integrable; hence e and r are square integrable. From the closed-loop error system (3.3), it is also easy to show that $\dot{r}$ is bounded; therefore, r is uniformly continuous. From Barbalat's Lemma [10], we know that e and $\dot{e}$ both go to zero asymptotically.

Q.E.D.

It should be emphasized that the controller (3.11) can be applied to any revolute rigid-ink robots without any prior knowledge about the system such as masses and lengths of every link. This is in sharp contrast to adaptive control approach which requires very tedious preliminary dynamical analysis of a given robot system to get the regression matrix. If a new robot comes in, the whole procedure has to start all over again. This can be quite a formidable task for robots with multiple degrees of freedom.

Note that the problem of <u>net weight initialization</u> occurring in other approaches in the literature does not arise. In fact, selecting the initial weights $\hat{W}(0)$, $\hat{V}(0)$ as zero takes the NN out of the circuit and leaves only the outer tracking loop in Fig. 3.1. The PD term k_vr and k_pe in (3.11) can stabilize the robot arm on an interim basis until the NN begins to learn. This also means that there is <u>no off-line learning phase</u> for this NN controller. Our results in the Theorem shows that there will not be any stability problem in the transient phase.

There is, however, no assurance for $\hat{W}$, $\hat{V}$ to converge to the true W, V. This phenomenon is reminiscent of similar things in adaptive control where no parameter convergence is assured unless certain signals in the system are persistently exciting.

4. Example

Consider a simple 2-link manipulator shown in Fig. 4.1. The model for this robot system can be described in the form of (2.1) with

$$M(q) = \begin{bmatrix} a + b\cos(q_2) & c + \frac{b}{2}\cos(q_2) \\ c + \frac{b}{2}\cos(q_2) & c \end{bmatrix},$$

$$V_m(q,\dot{q}) = \begin{bmatrix} -b\sin(q_2)\dot{q}_2 & -\frac{1}{2}b\sin(q_2)\dot{q}_2 \\ \frac{1}{2}b\sin(q_2)\dot{q}_1 & 0 \end{bmatrix},$$

$$G(q) = \begin{bmatrix} d\cos(q_1) + e\cos(q_2) \\ e\cos(q_1 + q_2) \end{bmatrix}, \qquad (4.1)$$

$$a = l_2^2m_2 + l_1^2(m_1 + m_2), \qquad b = 2l_1l_2m_2,$$
$$c = l_2^2m_2, \quad d = (m_1 + m_2)l_1g_0, \qquad e = m_2l_2g_0.$$

Parameters a, b, c, d, and e are unknown constants. The parameter values, which correspond to the first two links of a PUMA-560 robot [7], are l_1 = 0.432 m, l_2 = 0.432m, m_1 = 15.61 kg, m_2 = 11.36 kg, g_0 = 10 m/s^2. The desired joint trajectories are defined as $q_{d1}(t) = \sin(t)$, $q_{d2}(t) = \cos(t)$.

We divide our studies into three parts:

(i) PD controller

The controller is given by $\tau = k_vr + k_pe$ with k_v = diag{10,10}, k_p = diag{40,40}. The simulation results are shown in Fig. 4.2. The results are not very encouraging since there exist large steady-state errors in both links.

(ii) Standard DCAL controller

Following the design steps of DCAL [10], we arrive at the following controller $\tau = Y_d\hat{\theta} + k_vr + k_pe + v_R$ with $Y_d\theta = f_d(\bullet)$ defined by (2.11) and $\hat{\theta} \in R^5$ the vector of unknown parameters a, b, c, d, e defined in (4.1). Gains k_p, k_v are the same as the PD controller. The robustifying term v_R is defined by

$$v_R = \frac{r(S\hat{\varphi})^2}{(S\hat{\varphi})\|r\| + \delta},$$

where

$$S = [1 \;\; \|z\| \;\; \|z\|^2], \; z = [e^T \;\; r^T], \; \varphi = [\zeta_0 \;\; \zeta_1 \;\; \zeta_2]^T,$$

$$\dot{\delta} = -\gamma\delta, \quad \delta(0) = \text{design constant} > 0, \gamma > 0.$$

The parameter tuning law for φ is chosen as

$$\dot{\hat{\varphi}} = \Gamma S^T\|r\| = -\dot{\tilde{\varphi}},$$

where Γ is a symmetric and positive definite matrix and $\tilde{\varphi} = \varphi - \hat{\varphi}$. The vector of estimates of the unknown parameters $\hat{\theta}$ is tuned by the following law [10]

$$\dot{\hat{\theta}} = \Gamma_1Y_d^Tr$$

where Γ_1 is a symmetric and positive definite matrix. For the simulations, we choose $\Gamma = \Gamma_1 = 5$ I, $\gamma = 0.1$, $\delta(0) = 30$. All initial conditions of $\hat{\theta}$, $\hat{\varphi}$ are set to zero. The results are shown in Figs. 4.3 and 4.4. Fig. 4.4 shows results where a term in $\hat{\theta}$, i.e. $\hat{e}$, is missing. This simulates the case of unmodeled dynamics. It can be seen that the performance in Fig. 4.4 is not very good since there exist steady-state errors in both links. This demonstrates that the DCAL controller is not robust to unmodeled dynamics since the performance in Fig. 4.4 is worse than that of Fig. 4.3.

(iii) DCAL-NN

The controller is given by (3.11), (3.17) and (3.18). The parameters are $\Gamma = 5\,I$, $\gamma = 0.1$, $\delta(0) = 30$, $F = G = 0.5\,I$. k_p, k_v are the same as the PD controller. All initial conditions of $\widehat{W}$, $\widehat{V}$, $\widehat{\varphi}$ are set to zero. There are ten units in the hidden layer. Although more units could be used, our results show that 10 units are quite enough in this example. The results are shown in Fig. 4.5. It can be seen that the convergence is not only slightly faster than DCAL but also the controller is very robust since we do not know any prior dynamics of the robot.

5. Conclusion

We have presented a DCAL-NN robust controller to the motion control of rigid-link robots. The purpose of NN is to approximate a highly nonlinear robot function. The method does not require the robot dynamics to be exactly known. It can be viewed as a <u>universal reusable controller</u> since it can be applied to any type of rigid-link robots without any modifications. Compared with adaptive control, we do not need linearity in the parameters assumption, persistently excitation conditions, and the explicit expression of regression matrix. Our weight tuning schemes <u>do not require off-line "training phase"</u>. Error in position is <u>globally</u> asymptotically stable and NN weights are guaranteed to be bounded.

Appendix

Proof of Fact 1:

Since $\sigma(\cdot)$ and $\widehat{\sigma}'(\cdot)$ in (3.9) are both bounded, then we have

$$\|O(\widetilde{V}^T x)^2\| \le c_1 + \alpha\|\widetilde{V}^T x\| \qquad c_1 > 0,\ \alpha > 0.$$
$$\le c_1 + \alpha\|\widetilde{V}\|_F Q_d$$

where Q_d is defined in Assumption 2. Defining $c_2 = \alpha Q_d$ yields the fact.

Proof of Fact 2:

The first two terms in (3.14) can be shown to be bounded by [10]

$$\|f(\cdot) - f_d(\cdot)\| \le \alpha_0 + \alpha_1\|z\| + \alpha_2\|z\|^2$$

where α_i's are positive scalar constants. Noting the fact that $\widehat{\sigma}'(\cdot)$ is bounded, the third term in (3.14) can be bounded by

$$\alpha_3\|\widetilde{W}\|_F\|\widetilde{V}\|_F Q_d, \qquad \alpha_3 > 0 \quad \text{(Assumption 2)}$$
$$\le \alpha_3(\|W\|_F + \|\widehat{W}\|_F)(\|V\|_F + \|\widehat{V}\|_F)Q_d$$
$$\le \alpha_3(W_M + \|\widehat{W}\|_F)(V_M + \|\widehat{V}\|_F)Q_d. \quad \text{(Assumption 1)}$$

Using Fact 1, the fourth term in (3.14) can be similarly bounded by

$$\alpha_4 + \alpha_5\|\widehat{V}\|_F$$

with $\alpha_4 > 0$, $\alpha_5 > 0$. The fifth and sixth terms are bounded by constants. Thus, combining the above results and manipulating some of the terms yields

$$\rho(z,\widehat{W},\widehat{V}) = \zeta_0 + \zeta_1\|z\| + \zeta_2\|z\|^2 + \zeta_3\|\widehat{W}\|_F + \zeta_4\|\widehat{V}\|_F + \zeta_5\|\widehat{V}\|_F\,\|\widehat{W}\|_F = S\varphi$$

where ζ_i's are positive scalar constants.

References

[1] C. Abdallah, D. Dawson, P. Dorato, M. Jamishidi, "Survey of the robust control robots," *IEEE Control Systems Magazine*, vol. 11, no. 2, pp. 24-30, 1991.

[2] F.C. Chen and H.K. Khalil, "Adaptive control of nonlinear systems using neural networks," *Int. J. Control*, vol. 55, no. 6, pp.1299-1317, 1992.

[3] S.R. Chu and R. Shoureshi, "Neural-based adaptive nonlinear system identification," *Intelligent Control Systems*, DSC-vol. 45, ASME Winter Meeting, 1992.

[4] J. J. Craig, *Adaptive Control of Mechanical Manipulators*, New York: Wiley, 1986.

[5] X. Cui and K.G. Shin, "Direct control and coordination using neural networks," *IEEE Trans. Systems, Man, and Cybernetics*, vol.23, no.3, 1993.

[6] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303-314, 1989.

[7] J. Flores, "Position and Force Control of Flexible and Rigid Manipulators," Ph.D. Dissertation, The University of Texas at Arlington, December, 1991.

[8] B. Horn, D. Hush, and C. Abdallah, "The state space recurrent neural network for robot identification," *Advanced Control Issues for Robot Manipulators,* DSC-vol. 39, ASME Winter Annual Meeting, 1992.

[9] K. Hornik, M. Stinchombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359-366, 1989.

[10] F.L. Lewis, C. Abdallah and D. Dawson, *Control of Robot Manipulators*, MacMillan Company, 1993.

[11] F.L. Lewis, K. Liu, and A. Yesilkdirik, "Neural net robot controller with guaranteed tracking performance," *Proc. IEEE Int. Symp. Intelligent Control*, pp. 225-231, 1993.

[12] F.L. Lewis, K. Liu and A. Yesildirek, "Multilayer Neural Net Robot Controller With Guaranteed Tracking Performance," *IEEE Trans. Neural Networks*, 1995.

[13] K.S. Narendra, "Adaptive control using neural networks," *Neural Networks for Control*, pp. 115142, MIT Press, 1991.

[14] K.S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks, " *IEEE Trans. Neural Networks*, vol.1, pp. 4-27, 1990.

[15] R. Ortega and M. Spong, "Adaptive motion control of rigid robots: a tutorial," *IEEE Conf. Decision and Control*, pp. 1575-1584, 1988.

[16] T. Ozaki, T. Suzuki, T. Furuhashi, S. Okuma, and Y. Ushikawa, "Trajectory control of robotic manipulators," *IEEE Trans. Ind. Electronics*, vol. 38, pp. 195-202, 1991.

[17] M.M. Polycarpou and P.A. Ioannou, "Identification and control using neural network models: design and stability analysis," Tech. Report 91-09-01, Dept. Elect. Eng. Sys., U. of Southern California, September, 1991.

[18] N. Sadegh, "Nonlinear identification and control via neural networks," *Control Systems with Inexact Dynamic Models*, DSC-vol.33, ASME Winter Annual Meeting, 1991.

[19] N. Sadegh and R. Horowitz, "Stability and robustness analysis of a class of adaptive controllers for robotic manipulators," *Int. J. Robot. Res.*, vol.9, pp. 74-92, 1990.

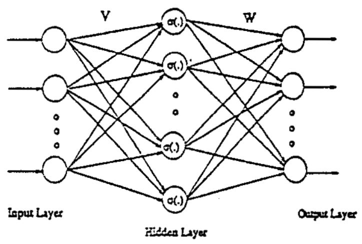

Fig. 2.1 Three Layer Neural Network.

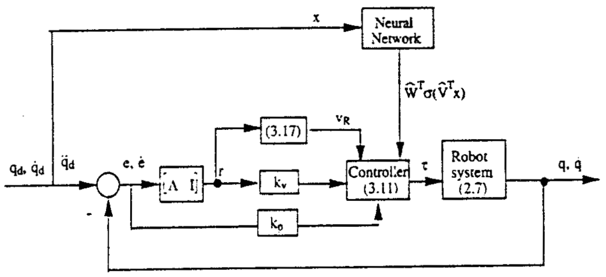

Fig. 3.1 DCAL-NN Controller Structure.

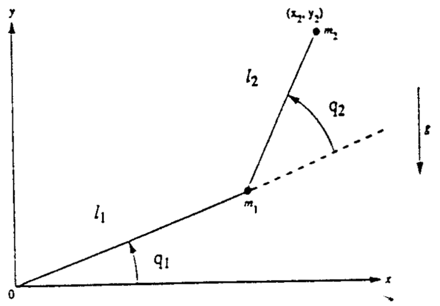

Fig. 4.1 A two-link manipulator.

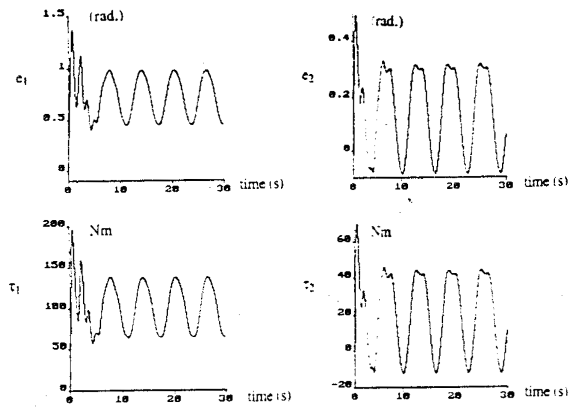

Fig. 4.2 Performance of PD controller.

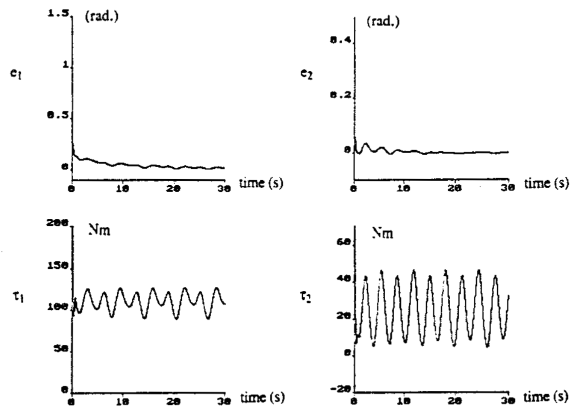

Fig. 4.3 Performance of DCAL controller.

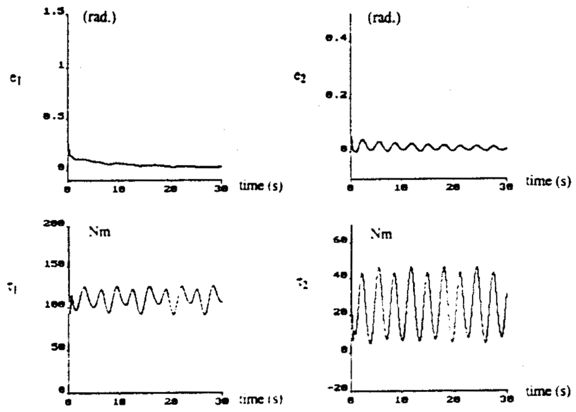

Fig. 4.4 Performance of DCAL controller with unmodelled dynamics.

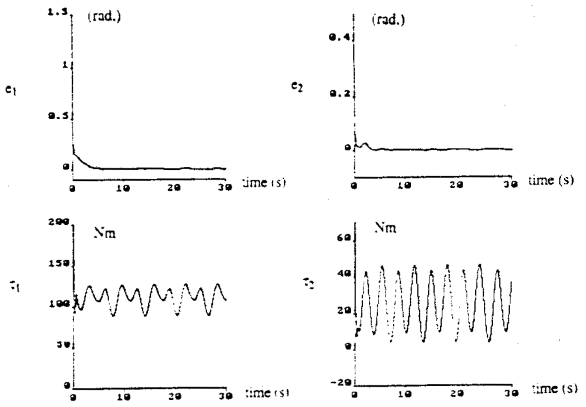

Fig. 4.5 Performance of DCAL-NN controller.