Stability Properties of Artificial Neural Network Based Robotic Controllers *

M. Kemal Cılız
Electrical Engineering Department, Boğaziçi University
Bebek, Istanbul 80815 Turkey

April 24, 1992

Abstract

Robotic manipulator control with *unknown or uncertain* dynamics has been an important research topic in the last decade. Without a parametric model of robot dynamics, learning control techniques are still the most effective methods for repeated trajectory following tasks. In this class of controllers, neurologically inspired algorithms have been gaining much attention in recent years. Although these techniques were shown to work effectively in simulation experiments, coupled and nonlinear nature of parameter update dynamics makes an effective mathematical analysis difficult. This paper investigates the *convergence* properties of an artificial neural network based learning controller. The results obtained reflect the local stability properties of the closed loop nonlinear system dynamics.

1. Introduction

In recent years, Artificial Neural Networks (ANN) have emerged as a viable alternative for various engineering problems of nonlinear nature. Within this new trend, many researchers have attempted to apply neurologically inspired algorithms for manipulator control. The main underlying assumption in these applications is the efficient capability of multi-layer ANNs to approximate multivariable functions. With the increasing interest in this area, IEEE Control Systems Magazine had come up with three issues which covered special sections on neural networks for control systems [1]. The interested researcher can find an exhaustive list of the previous work on the subject in [2].

The present paper discusses the stability properties of an ANN based trajectory following controller. The controller algorithm in question was originally proposed by the author in [3, 2] and successfully tested for trajectory following tasks through simulation experiments. The controller is basically similar to the adaptive inverse dynamics control algorithm of Craig et al. [4].

However instead of using an explicit parametric model, the controller utilizes *generic multi-layer ANNs* to adaptively approximate the manipulator dynamics over a specified region of the state space for a given desired trajectory. This generic neural network structure can be viewed as a nonlinear extension of a deterministic auto-regressive model which is commonly used in model matching problems for linear systems.

2. Controller Architecture

In this section we introduce the controller structure and then write the closed loop system dynamics for the proposed ANN based controller.

*This work was in part sponsored by Westinghouse Education Foundation under contract No:3597262.

Consider the vector representation of an n link rigid manipulator dynamics, given as

$$\tau = M(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{v}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) \tag{1}$$

where τ is the $n\times 1$ vector of joint torques, and $\mathbf{q}$ is the $n\times 1$ vector of joint positions. The matrix $M(\mathbf{q})$ is the $n\times n$ positive definite "inertia matrix". $\mathbf{v}(\mathbf{q},\dot{\mathbf{q}})$ is $n\times 1$ vector function representing centrifugal and Coriolis effects, and finally $n\times 1$ vector function $\mathbf{g}(\mathbf{q})$ represents torques due to gravity. The derivation of (1) can be found in common reference texts [5]. Equation (1) can be put in a more compact form as ,

$$\tau = M(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) \tag{2}$$

The control designer's *task* is to devise a controller such that the manipulator tracks a given desired trajectory $(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)$ as closely as possible.

If exact manipulator dynamics is available, then the *control*

$$\tau = M(\mathbf{q})(K_v\dot{\mathbf{e}} + K_p\mathbf{e} + \ddot{\mathbf{q}}_d) + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) \tag{3}$$

will result in an error equation of the form,

$$\ddot{\mathbf{e}} + K_v\dot{\mathbf{e}} + K_p\mathbf{e} = 0 \tag{4}$$

due to the cancellation of nonlinear terms, where $\mathbf{e} = \mathbf{q}_d - \mathbf{q}$ and $\dot{\mathbf{e}} = \dot{\mathbf{q}}_d - \dot{\mathbf{q}}$. K_v and K_p are the diagonal matrices of *velocity* and *position* servo feedback gains, respectively. Note that the above error equation is a decoupled one due to the diagonal nature of the constant matrices K_p and K_v. Therefore adjusting these gain matrices properly, tracking errors can be effectively forced to zero. If the dynamic model of the manipulator is not available to the designer, generally learning control techniques are used to generate the necessary feedforward torques.

Here we propose the use of a generic ANN to model the inverse dynamic structure of each joint. For an n-link manipulator, inverse system dynamics given in (2) can be defined by a nonlinear transformation

$$R^{-1} : \mathcal{R}^{3n} \to \mathcal{R}^n \tag{5}$$

which maps the $3n$ dimensional output space of the system (position, velocity and acceleration vectors) to the n-dimensional input space (joint torque vector τ). Such a nonlinear transformation can be effectively modeled by ANNs as discussed by Funahashi [6], Cybenko [7] and Hornik, Stinchcombe and White [8].

Mathematical Setup:

Inverse dynamics which is represented by the nonlinear transformation R^{-1} in (5), can be decomposed into n transformations for each joint, namely

$$\begin{aligned} \tau &= R^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) && (6)\\ &= \begin{bmatrix} r_1^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \\ \vdots \\ r_n^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \end{bmatrix} && (7)\end{aligned}$$

where each $r_i^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}})$, $i = 1,\ldots,n$ defines the inverse dynamics of the corresponding joint, that is,

$$r_i^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) : \mathcal{R}^{3n} \to \mathcal{R}, \text{ with } i = 1,\ldots,n$$

Each entry $r_i^{-1}(\cdot)$ of the vector function R^{-1} can be modeled by a multilayer ANN such that the overall system's inverse dynamics model is represented by,

$$\tau = \hat{R}^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) = \begin{bmatrix} \hat{r}_1^{-1}(\mathbf{z}(t)) \\ \vdots \\ \hat{r}_n^{-1}(\mathbf{z}(t)) \end{bmatrix} = \begin{bmatrix} N_1(\mathbf{z},\mathbf{p}_1) \\ \vdots \\ N_n(\mathbf{z},\mathbf{p}_n) \end{bmatrix} \tag{8}$$

where $(\hat{\cdot})$ denotes the *estimated* models, and $N_i(\cdot)$, $i = 1, \ldots, n$ represents the output of each ANN model that is used to realize the nonlinear mapping $r_i^{-1}(\cdot)$. $\mathbf{z}(t)$ is an augmented state vector of the robot dynamics,

$$[\mathbf{z}(t) = (\mathbf{q}^T(t), \dot{\mathbf{q}}^T(t), \ddot{\mathbf{q}}^T(t))^T \in \mathcal{R}^{3n}$$

which denotes the time dependent input vector of the inverse dynamics and $\mathbf{p}_i$ is the *vector* of all adjustable weights of the ith ANN model.

Here we assume that a *three-layer ANN* with "k" inputs, "m" hidden layer units and one output unit is used to model a joint's inverse dynamics. Then this model can be explicitly written as,

$$\hat{r}_i^{-1}(\mathbf{z}(t)) = N_i(\mathbf{z}(t), \mathbf{w}_i(t), H_i(t)) = \mathbf{w}_i^T \Upsilon(H_i \mathbf{z}) \tag{9}$$

where $\mathbf{w}_i(t) \in \mathcal{R}^m$ is the adaptive output layer weight (parameter) vector, $H_i(t) \in \mathcal{R}^{m \times k}$ is the hidden layer adaptive weight matrix of the "i"th ANN model. $\mathbf{z}(t) \in \mathcal{R}^k$ with $k = 3n$ is the input vector as defined before. In the rest of the text, time argument of these vectors will sometimes be dropped for notational simplicity. The vector function $\Upsilon(\cdot) \in \mathcal{R}^m$ is defined as,

$$\Upsilon(\cdot) = (g_1(\cdot), \ldots, g_m(\cdot))^T$$

where $g_i(\cdot) \in \mathcal{R}$ is by definition a monotone increasing function which is taken as a *sigmoid function* in this case, based on the justification given by the theorems in [6, 8]. Hence $g_i(x) = \frac{1}{(1+e^{-x})}$ and it is bounded as $0 \leq g_i(x) \leq 1$.

We next define a vector $\mathbf{v}_i = vec(H_i) \in \mathcal{R}^{mk}$ which represents the hidden layer adaptive weight matrix H_i of the ANN model in vectoral form. $vec(\cdot)$ operator gives a vector which is obtained by stacking the columns of its matrix argument. Based on this new parameter vector, the argument of vector function $\Upsilon(\cdot)$ can be written as

$$H_i \mathbf{z} = \Phi \mathbf{v}_i \tag{10}$$

where

$$\mathbf{v}_i = \{H_{i_{11}}, H_{i_{21}}, \ldots, H_{i_{m1}}, \ldots, H_{i_{1k}}, \ldots, H_{i_{mk}}\}^T$$

and $\Phi \in \mathcal{R}^{m \times mk}$ is a matrix which can be considered as the modified input of the ANN model and is defined as,

$$\Phi = \begin{pmatrix} z_1 & 0 & \ldots & 0 & \ldots\ldots & z_k & 0 & \ldots & 0 \\ 0 & z_1 & \ldots & 0 & \ldots\ldots & 0 & z_k & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ldots\ldots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & z_1 & \ldots\ldots & 0 & 0 & \ldots & z_k \end{pmatrix} \tag{11}$$

where $z_i \in \mathcal{R}$, $i = 1, \ldots, k$ are the elements of the input vector $\mathbf{z} \in \mathcal{R}^k$. Hence (9) can now be written as,

$$\hat{r}_i^{-1}(\mathbf{z}, \mathbf{w}_i, \mathbf{v}_i) = N_i(\mathbf{z}, \mathbf{w}_i, \mathbf{v}_i) = \mathbf{w}_i^T \Upsilon(\Phi \mathbf{v}_i) \tag{12}$$

Next we define a control law which consists of an *adaptive feedforward compensator* and a *feedback* signal as,

$$\tau = \hat{R}^{-1}(\mathbf{z}) + \ddot{\mathbf{e}} + K_v \dot{\mathbf{e}} + K_p \mathbf{e} = \mathrm{N}(\mathbf{z}) + \ddot{\mathbf{e}} + K_v \dot{\mathbf{e}} + K_p \mathbf{e} \tag{13}$$

where $K_v \in \mathcal{R}^{n \times n}$ and $K_p \in \mathcal{R}^{n \times n}$ are the diagonal gain matrices with entries k_v and k_p, respectively.

$$\hat{R}^{-1}(\mathbf{z}) = \mathrm{N}(\mathbf{z}) = \{N_1, \ldots, N_n\}^T \in \mathcal{R}^n$$

is the robot's dynamic model estimate which consists of "n" individual ANN models, each representing one joint's inverse dynamics.

3. Closed Loop System Dynamics

In this section we generate the closed loop system's dynamic equations and demonstrate the difficulty of a global stability analysis. With the control vector given in (13), system's error dynamics can be written by substituting (13) in (2),

$$\hat{R}^{-1}(\mathbf{z}) + \ddot{\mathbf{e}} + K_v\dot{\mathbf{e}} + K_p\mathbf{e} = \underbrace{M(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}})}_{R^{-1}(\mathbf{z})} \tag{14}$$

$$\ddot{\mathbf{e}} + K_v\dot{\mathbf{e}} + K_p\mathbf{e} = \tilde{R}(\mathbf{z}) \tag{15}$$

where $\tilde{R}^{-1}(\mathbf{z}) = \tilde{R}^{-1}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \in \mathcal{R}^n$ denotes the error between the actual inverse dynamics R^{-1} and the estimated model $\hat{R}^{-1}$, and can be explicitly written as,

$$\tilde{R}^{-1}(\mathbf{z}) = \begin{bmatrix} \tilde{r}_1^{-1}(\mathbf{z},\mathbf{p}_1) \\ \vdots \\ \tilde{r}_n^{-1}(\mathbf{z},\mathbf{p}_n) \end{bmatrix} \tag{16}$$

where

$$\tilde{r}_i^{-1}(\mathbf{z},\mathbf{p}_i) = r_i^{-1}(\mathbf{z}) - N_i(\mathbf{z},\mathbf{p}_1) = r_1^{-1}(\mathbf{z}) - \hat{r}_1^{-1}(\mathbf{z})$$

denotes the error in inverse dynamic modeling for each joint, and $\mathbf{p}_i = \{\mathbf{w}_i^T, \mathbf{v}_i^T\}^T \in \mathcal{R}^{m+mk}$ is the adaptive weight vector of the corresponding ("i"th) ANN model.

Using (15) and (16) the error dynamics (with diagonal K_p and K_v matrices) for each joint can be written as follows,

$$\ddot{e}_i + k_v\dot{e}_i + k_p e_i = \tilde{r}_i^{-1}(\mathbf{z},\mathbf{p}_i), \text{ for } i = 1,\ldots,n. \tag{17}$$

where e_i, $\dot{e}_i$ and $\ddot{e}_i$ denote the position, velocity and acceleration errors at joint i, respectively, k_p and k_v are the individual servo gains, respectively. Based on the existence theorems given in [6, 8], we can assume that there exists a three layer ANN structure (which is defined in (12)) that closely approximates the joint's inverse dynamics. That is,

$$r_i^{-1}(\mathbf{z}) = \mathbf{w}_{i0}^T\Upsilon(\Phi\mathbf{v}_{i0}) \tag{18}$$

where $\mathbf{w}_{i0}$ and $\mathbf{v}_{i0}$ represent the *desired* output layer and hidden layer weights that generate the *desired mapping*, r_i^{-1}. In the rest of the analysis the subscript "i"is dropped for the brevity of the presentation. Using the desired mapping given in (18), the error in the approximation of the inverse dynamics of a joint can be written as,

$$\tilde{r}^{-1}(\mathbf{z},\mathbf{p}) = \underbrace{\mathbf{w}_0^T\Upsilon(\Phi\mathbf{v}_0)}_{r^{-1}} - \underbrace{\mathbf{w}^T\Upsilon(\Phi\mathbf{v})}_{\hat{r}^{-1}} \tag{19}$$

Using this residual dynamic representation, we can analyze the error equation given in (17) which in fact represents a stable linear system with a nonlinear forcing function. Then the error dynamics in (17) can be written as,

$$e(t) = H(s)\tilde{r}^{-1}(\mathbf{z},\mathbf{p}) \tag{20}$$

where $H(s)$ is a strictly positive real (SPR) transfer function, due to the positive gain terms k_p and k_v in (17). Based on this error dynamic model, to update the parameter estimates $\mathbf{p}$, a simple *gradient update* algorithm can be written as,

$$\dot{\mathbf{p}} = -\Gamma\frac{\partial\tilde{r}^{-1}(\mathbf{z},\mathbf{p})}{\partial\mathbf{p}}e(t) \tag{21}$$

where $\Gamma \in \mathcal{R}^{(m+mk)\times(m+mk)}$ is a diagonal gain matrix which includes the learning rates as its diagonal entries. The state space representation of (17) is given by,

$$\begin{aligned} \dot{\mathbf{x}} &= A\mathbf{x} + B\tilde{r}^{-1}(\mathbf{z},\mathbf{p}) && (22) \\ e &= \mathbf{c}\mathbf{x} && (23) \end{aligned}$$

where $\mathbf{x} = (e, \dot{e})^T \in \mathcal{R}^{2\times 2}$ is the state vector, and A, B and and $\mathbf{c}$ denote the minimal state space representation due to strictly positive real $H(s)$. Since $H(s)$ is an SPR transfer function, based on the Kalman-Yakubovich lemma [9], for a given symmetric positive definite (p.d.) matrix Q, there exists another symmetric positive definite matrix P which satisfies the Lyapunov equation,

$$A^T P + PA = -Q \tag{24}$$

and the output relation,

$$\mathbf{c} = B^T P \tag{25}$$

Replacing the output vector $\mathbf{c}$ in (21) by (25), the gradient update law take sthe form,

$$\begin{aligned} \dot{\mathbf{p}} &= -\Gamma \frac{\partial \tilde{r}^{-1}(\mathbf{z},\mathbf{p})}{\partial \mathbf{p}} B^T P\mathbf{x} && (26) \\ \dot{\mathbf{p}} &= \Gamma \frac{\partial N(\mathbf{z},\mathbf{p})}{\partial \mathbf{p}} B^T P\mathbf{x} && (27) \end{aligned}$$

where $N(\mathbf{z},\mathbf{p})$ is the output the ANN model. Due to the nonlinear parametric dependence of the term $N(\mathbf{z},\mathbf{p})$, the computation of the partial derivative term in (27) requires the so-called *backpropagation* algorithm. Note that the closed loop adaptive system represented by (22) and (27) defines a *coupled nonlinear* system of differential equations and this makes a global convergence and stability analysis of the closed loop system difficult. However local properties of the system dynamics can be studied through the use of linearization techniques. Linearization dictates that, subject to smoothness of the nonlinear oprators in (22) and (27), one can constitute a linearized system whose stability properties are identical to the local stability properties of (22) and (27).

4. Stability and Convergence Analysis

In order investigate the local stability and convergence properties of the closed loop system, the system dynamics is linearized around the desired system state. Let $\mathbf{p}_*$ and $\mathbf{x}_*$ denote the desired values of the parameter vector $\mathbf{p}$ and the state vector $\mathbf{x}$, respectively. That is $\mathbf{x}_* = 0$ and $\mathbf{p}_* = (\mathbf{w}_0^T, \mathbf{v}_0^T)^T$. This basically corresponds to a condition where the system operates in the vicinity of the desired trajectories $(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)$, and the ANN model parameters are close to their desired values. With this choice of the nominal signals, the perturbation vectors become,

$$\tilde{\mathbf{x}} = \mathbf{x}_* - \mathbf{x} = -\mathbf{x}$$

which is actually the tracking error vector and

$$\tilde{\mathbf{p}} = \mathbf{p}_* - \mathbf{p}$$

which is the parameter error vector. With this set up, we first linearize (22) around $\mathbf{x}_* = 0$ and $\mathbf{p}_*$ as follows,

$$\dot{\tilde{\mathbf{x}}} = \left.\frac{\partial(A\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}_*} \tilde{\mathbf{x}} - B \left.\frac{\partial(\mathbf{w}^T \Upsilon(\Phi\mathbf{v}))}{\partial \mathbf{w}}\right|_{\mathbf{z}_*,\mathbf{w}_*,\mathbf{v}_*} \tilde{\mathbf{w}} - B \left.\frac{\partial(\mathbf{w}^T \Upsilon(\Phi\mathbf{v}))}{\partial \mathbf{v}}\right|_{\mathbf{z}_*,\mathbf{w}_*,\mathbf{v}_*} \tilde{\mathbf{v}} \tag{28}$$

where $\mathbf{z}_* = (\mathbf{q}_d{}^T, \dot{\mathbf{q}}_d{}^T, \ddot{\mathbf{q}}_d{}^T)$ basically corresponds to the desired (nominal) state $\mathbf{x}_* = 0$. Evaluating the partials, we get

$$\dot{\mathbf{x}} = A\mathbf{x} + B\Upsilon^T(\Phi_*\mathbf{v}_*)\tilde{\mathbf{w}} + B\mathbf{w}_*^T J_*\Phi_*\tilde{\mathbf{v}} \tag{29}$$

where $J_* \in \mathcal{R}^{m\times m}$ is a diagonal Jacobian matrix evaluated at the nominal values,

$$J_* = \begin{pmatrix} g_1'|_{\mathbf{z}_*,\mathbf{v}_*} & 0 & \dots & 0 \\ 0 & g_2'|_{\mathbf{z}_*,\mathbf{v}_*} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & g_m'|_{\mathbf{z}_*,\mathbf{v}_*} \end{pmatrix} \tag{30}$$

and Φ_* is the vector function (defined in (11)) which is evaluated at $\mathbf{z}_*$. Combining $\tilde{\mathbf{v}}$ and $\tilde{\mathbf{w}}$ as the *parameter error vector*,

$$\tilde{\mathbf{p}} = (\tilde{\mathbf{w}}^T, \tilde{\mathbf{v}}^T)^T \in \mathcal{R}^{(m+mk)}$$

equation (29) can be written as

$$\dot{\mathbf{x}} = A\mathbf{x} + B\Psi_*(\mathbf{z}_*, \mathbf{w}_*, \mathbf{v}_*)\tilde{\mathbf{p}} \tag{31}$$

where

$$\Psi_* = (\Upsilon_*^T(\Phi_*\mathbf{v}_*), \mathbf{w}_*^T J_*\Phi_*) \in \mathcal{R}^{1\times(m+mk)} \tag{32}$$

can be considered as the *linearized regressor vector* of the error dynamics, and Using the same arguments and linearizing (27) explicitly for $\mathbf{w}$ and $\mathbf{v}$, we get,

$$\dot{\tilde{\mathbf{w}}} = \Gamma_1\Upsilon(\Phi_*\mathbf{v}_*)B^T P\tilde{\mathbf{x}} \tag{33}$$
$$\dot{\tilde{\mathbf{v}}} = \Gamma_2\Phi_*^T J_*\mathbf{w}_* B^T P\tilde{\mathbf{x}} \tag{34}$$

where Φ_* and J_* are as defined previously and $\Gamma_1 \in \mathcal{R}^{mk\times mk}$ and $\Gamma_2 \in \mathcal{R}^{k\times k}$ are diagonal matrices which are in fact the partitions of the gain matrix Γ. Combining equations (33) and (34), linearized update equation for the parameter vector $\mathbf{p}$ can be written as follows

$$\dot{\tilde{\mathbf{p}}} = -\Gamma\Psi_*^T B^T P\mathbf{x}, \quad \text{since } \tilde{\mathbf{x}} = -\mathbf{x} \tag{35}$$

where Ψ_* is as defined in (32). Equations the (31) and (35) constitute the linearized closed loop system dynamics. Based on this linearized model we can now state the following theorem [10].

Theorem 1:
Given the linearized system dynamics in (31) and the update law in (35), the tracking error vector $\mathbf{x}$ *satisfies,*

$$\mathbf{x} \to 0 \text{ as } t \to \infty$$

with all signals remaining bounded.

The proof of the theorem is given in [10]. The above result ensures the convergence of the tracking error vector $\mathbf{x}$ in the vicinity of a nominal solution. However note that the above theorem does not guarantee the convergence of the parameter error vector $\tilde{\mathbf{p}}$, although it shows *its boundedness.*

5. Convergence of Weight Estimates

In order to investigate the convergence properties of the parameter error vector $\tilde{\mathbf{p}}$, let's write the linearized closed system dynamics using (31) and (35),

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\tilde{\mathbf{p}}} \end{bmatrix} = \begin{bmatrix} A & B\Psi_* \\ -\Gamma\Psi_*^T B^T P & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \tilde{\mathbf{p}} \end{bmatrix} \tag{36}$$

Equations similar to (36) arise in most adaptive control applications, and its asymptotic stability has been studied by several researchers [11, 12, 4]. Note that equation (36) defines a linear time-varying system and its asymptotic stability determines the convergence of the parameter error vector $\tilde{\mathbf{p}}$. The system given in (36) is asymptotically stable, if the transfer function

$$B^T P(sI - A)^{-1} B \tag{37}$$

is strictly positive real and if there exists positive constants β, γ and T such that

$$\beta I \succeq \int_{t_0}^{t_0+T} \Psi_*^T \Psi_* dt \succeq \gamma I \tag{38}$$

holds for all t_0 with $I \in \mathcal{R}^{(m+mk)\times(m+mk)}$ being the identity matrix [11]. The sign $\succeq$ defines the positive semi-definite ordering of the matrices. The transfer function defined in (37) is strictly positive real for the above case as discussed in Section 4. Therefore if condition (38) is satisfied, then $\tilde{\mathbf{p}}$ converges asymptotically. Equation (38) which is usually referred to as the persistent excitation (p.e.) condition, states that Ψ_* must vary sufficiently over the interval T such that the entire $(m + mk)$ dimensional space (i.e. the parameter space of the ANN model) is spanned to ensure the convergence of $\tilde{\mathbf{p}}$.

In equation (38), Ψ_* is bounded as seen from (32). Then, left hand inequality holds directly and the p.e. condition can be written as,

$$\int_{t_0}^{t_0+T} \Psi_*^T \Psi_* dt \succeq \gamma I \tag{39}$$

The outer product matrix in the integral equation given in (39) can be explicitly written as a partitioned matrix as,

$$\Psi_*^T \Psi_* = \left(\begin{array}{c|c} \Upsilon_*(\Phi_* \mathbf{v}_*)\Upsilon_*^T(\Phi_* \mathbf{v}_*) & \Upsilon(\Phi_* \mathbf{v}_*)\mathbf{w}_*^T J_* \Phi_* \\ \hline \Phi_*^T J_*^T \mathbf{w}_* \Upsilon_*^T(\Phi_* \mathbf{v}_*) & \Phi_*^T J_*^T \mathbf{w}_* \mathbf{w}_*^T J_* \Phi_* \end{array} \right) \tag{40}$$

Note that the above matrix is related to the desired trajectory vector $\mathbf{z}_* = (\mathbf{q}_d{}^T, \dot{\mathbf{q}}_d{}^T, \ddot{\mathbf{q}}_d{}^T)^T$ through Φ_* and J_*, with $\mathbf{z}_*$ acting as their arguments. Due to the nonlinear nature of the above matrix integral equation, it is rather difficult and impractical to generate persistently exciting trajectories from that equation. One method to utilize the condition given in (39) is to generate desired trajectories based on our engineering knowledge and then to test their p.e. condition. We are currently studying the structure of (39) in order to derive more explicit p.e. conditions on $\mathbf{z}_*$. In general, generation of persistently exciting trajectories is an important research topic [11]. Next we present some simulation results and demonstrate the error and parameter convergence properties of the proposed controller architecture.

6. Simulation Results

The architecture proposed in Section 2 was tested on a two link manipulator model using simulation methods [2]. As a test for the adaptation properties of the controller, the end effector mass is changed while the manipulator is closely following a prescribed trajectory. For this test, the second link mass is changed from its nominal value of 8 kgs. to 16 kgs. at the "2" second mark. The position error profiles due to this sudden change in manipulator dynamics are shown in Figure (1). As shown in the figure, the controller effectively reduces the sudden jumps in the position errors and brings the errors down approximately to their previous levels in about 1-1.5 seconds. In order to demonstrate the changes in the manipulator's inverse dynamics due to the end effector mass change and the ANN model's ability to track these changes, the torque profiles are monitored during the adaptation test (Figure (2)).

Desired torque profiles corresponding to this change and the torque profiles *generated by the ANN models* of each joint are plotted in Figure (2). The observed torque profiles converge to their desired levels by the end of the trajectory. More simulation results are given in [2].

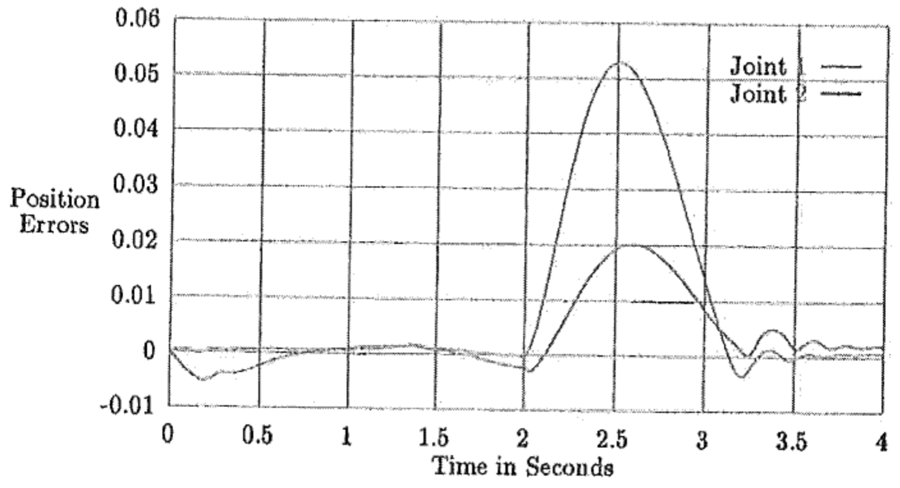

Figure 1: Position errors for the adaptation experiment. End effector mass is changed from 8 kgs. to 16 kgs.

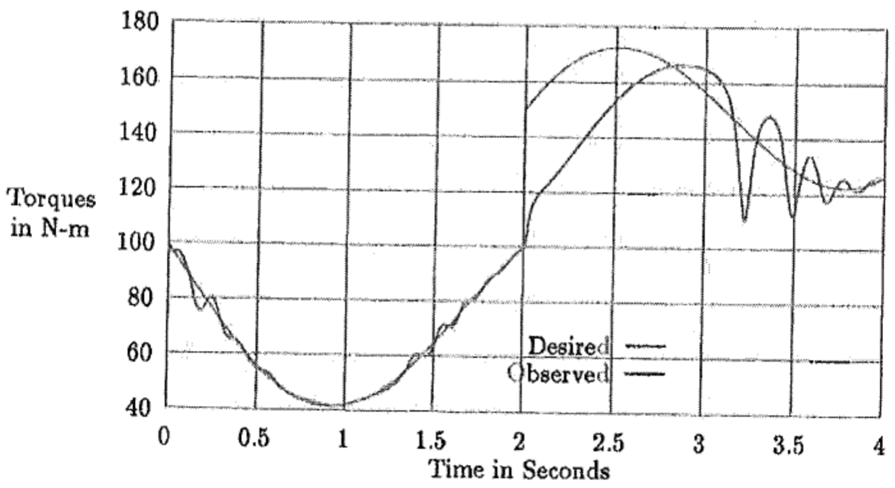

Figure 2: Desired and observed torque profiles when the end effector mass is changed from its nominal value of 8 kgs. to 16 kgs.

7. Conclusions

Neurologically inspired robotic controllers have been receiving much attention in recent years due to their effective learning capabilities. There are many reports on successful simulation results, however there is not yet a well defined mathematical analysis for their stability and convergence. This is due to the nonlinear nature of parameter update dynamics which makes use of the well known backpropagation algorithm. This paper investigates the local stability and convergence properties of an ANN based robotic controller which was previously proposed by the author. Simulation experiments demonstrate the convergence properties of the controller architecture.

References

[1] "IEEE Control Systems Magazine," April 1988-1989-1990. Special Section on Neural Networks.

[2] K. Cılız, *Artificial Neural Network Based Control of Nonlinear Systems With Application to Robotic Manipulators.* PhD thesis, Syracuse University, Electrical Engineering Department, 1990.

[3] K. Cılız and C. Işık, "Trajectory learning control of robotic manipulators using Neural Networks," in *IEEE International Symposium on Intelligent Control*, (Fairfax, VA), pp. 536–540, 1990.

[4] J. J. Craig, P. Hsu, and S. S. Sastry, "Adaptive control of mechanical manipulators," *International Journal of Robotics Research*, vol. 6, no. 2, pp. 16–28, 1987.

[5] A. J. Koivo, *Fundamentals for Control of Robotic Manipulators.* New York: John Wiley & Sons, 1989.

[6] K. C. Funahashi, "On the approximate realization of continuous mappings by Neural Networks," *Neural Networks*, vol. 2, pp. 183–192, 1989.

[7] G. Cybenko, "Approximations by superpositions of a sigmoidal function," tech. rep., University of Illinois, Center for Supercomputer Research and Development, Urbana, IL, 1989.

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

[9] C. A. Desoer and M. Vidyasagar, *Feedback Systems : Input-Output Properties.* New York, NY: Academic Press, 1975.

[10] K. Cılız and C. Işık, "Stability and convergence of neurologic model based robotic controllers," in *IEEE International Conference on Robotics and Automation*, (Nice, France), May 1992.

[11] K. S. Narendra and A. M. Annaswamy, *Stable Adaptive Systems.* Englewood Cliffs, N.J.: Prentice Hall, 1989.

[12] K. S. Narendra and A. P. Morgan, "On the uniform asymptotic stability of certain linear nonautonomous differential equations," *SIAM Jour. Cont. and Opt.*, no. 15, pp. 5–24, 1977.