Control of Robot Manipulators Using CMAC Neural Networks

S. Commuri
CGN & Associates
421 S. W. Washington Str.
Peoria, IL 61602.

S. Jagannathan
Automated Analysis Corp.
423 S. W. Washington Str.
Peoria, IL 61602.

F. L. Lewis
Auto. and Robotics Res. Inst.
7300 Jack Newell Blvd South
Fort Worth, TX 76118.

Abstract:

A comprehensive treatment of Cerebellar Model Articulation Controller (CMAC) Neural Network (NN) for the control of robot manipulators is presented. The structure and localized learning properties of CMAC NN is exploited to design efficient controllers for nonlinear systems belong to a given useful class. Continuous-time implementation of these controllers are systematically examined. Novel weight update schemes are derived and the closed-loop stability of the controller and the system is rigorously proved. These weight update schemes are shown to be nonstandard modifications of adaptive techniques prevalent in the literature. Finally, the validity of these techniques are demonstrated through numerical studies.

Keywords: Robot Control, CMAC NN Controllers, Nonlinear Adaptive Control,

1.Introduction

In recent years, learning-based control has emerged as an alternative to adaptive control. Important in this class of controllers are the neural networks where the learning is accomplished by tuning the weights of the NN. Rigorous research in NN for control applications is being pursued by several researchers [7-8,14]. While NNs have been popular in the literature, in practice, the global nature of the learning has adverse effect on the learning rate that is achievable. To circumvent this, CMAC NN has been proposed in the literature [1,2].

The functional approximation properties of feedforward NN [4] are basic in their application to control. In feedforward neural networks [7-8,14], since all the weights are updated during each learning cycle, the learning is essentially global in nature and slow. On the other hand, the CMAC NNs utilize the information on local NN structure and thus, the learning is faster with a good function approximation. In addition, this function approximation generated by these CMAC NNs is insensitive to the order of presentation of the training data. Furthermore, the attractive feature of NNs in general is that their design is model-free in the sense that NNs can be constructed for complex ill-defined processes without specific knowledge of the underlying model or the detailed dynamical characteristics.

To take advantage of the local CMAC NN structure, CMAC NNs are proposed for closed-loop control of complex dynamical systems [3,11-12,16]. The CMAC is a perceptron-like associative memory network with overlapping receptive fields that computes a nonlinear function over a domain of interest. The contents of these memory locations are referred to as the weights and the output of the CMAC NN is a linear combination of these weights [2].

While the advantages of using CMAC NN over conventional NN in many closed-loop applications are well known in literature [11-12], to our knowledge, there has been no study of using CMAC NNs for robot arm control in continuous-time which guarantee stability and bounded weight estimates. In [12], although real-time dynamic control of an industrial manipulator is presented using a CMAC NN, no convergence or stability analysis is provided. Therefore in [5] and [16], convergence analysis of a CMAC NN is detailed. However, Lyapunov-based analysis is not discussed in both papers and in [12] and [16] initial learning phase is also required for the CMAC controller.

The remainder of the paper is organized as follows. Section 2 gives a background on CMAC NNs and their approximation properties using multidimensional receptive fields. A brief outline of the control of robotic systems in continuous-time is presented in Section 3. A novel adaptive law is developed and the stability of the closed-loop system with the CMAC NN in continuous-time is rigorously proved in Section 4. Simulation results are presented in Section 5 and the conclusions and highlights of the paper are summarized in Section 6.

2. Background

Let R denote the real numbers, R^n denote the real n-vectors, R^{mxn} the real man matrices. Let S be a compact simply connected set of R^n. With maps $f:S \to R^k$, define $C^k(S)$ as the space such that f is continuous. We denote $\| \cdot \|$ any suitable vector norm. Given a matrix $A = [a_{ij}]$, $A \in R^{nxm}$ the Frobenius norm is defined by

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

with tr () the trace operation. The associated inner product is $<A,B>_F = tr(A^T B)$. The Frobenius norm $\|A\|_F$ which is denoted by $\| \cdot \|$ throughout this paper until unless specified explicitly, is nothing but the vector 2-norm over the space defined by stacking the matrix columns into a vector, so that it is compatible with the vector 2-norm, that is $\|Ax\| \le \|A\| . \|x\|$.

In order to formulate the controller, the following stability notion is needed. Consider the nonlinear system given by

$$\dot{x}(t) = f(x(t), u(t)),\ y(t) = h(x(t)),$$

where x(t) is a state vector, u(t) is the input vector and y(t) is the output vector. The solution is uniformly ultimately bounded (UUB) if for all $x(k_0) = x_0$, there exists $\varepsilon > 0$ and a number $T(\varepsilon, x_0)$ such that $\|x(t)\| \le \varepsilon$ for all $t \ge t_0 + T$.

2.1 Back Ground on CMAC Neural Networks

In Section 4 and 5, the controller assumes that a mechanism for reconstructing a nonlinear function f(.) is available for the case of continuous-time controller design and f(.) and g(.) for the case of discrete-time controller design. Now we show that CMAC neural networks are extremely convenient for approximation-based closed-loop control. Specifically we confront the problem of multi-input multi-output CMAC using simple receptive-field functions that are easily computable (n-dimensional linear splines). We consider that CMACs with linear splines possess a universal approximation property that can be used to design a controller that guarantees stability for a class of systems without explicit knowledge of the nonlinearity.

Fig. 2.1 shows a typical application of a CMAC neural network where the CMAC is used to manufacture a continuous function $h(\underline{x}) = [h_1(\underline{x}), h_2(\underline{x}), \ldots, h_m(\underline{x})]^T$, where $\underline{x} \in R^n$, and $h: R^n \to R^m$. The nonlinear function $h(\underline{x})$ produced by the CMAC is composed of two primary functions

$$R: X \Rightarrow A$$
$$P: A \Rightarrow Y \qquad (2.1)$$

where X is the continuous n-dimensional input space, A is an N_A-dimensional association space, and Y is the m-dimensional output space. The function R(.) is fixed and maps each point in the input space onto a association vector $\alpha = R(\underline{x})$ in the association space A. The function $P(\alpha)$ computes an output $h \in Y$ by projecting the association vector determined by $R(\underline{x})$ onto a vector of adjustable weights w such that

$$h = P(\alpha) = w^T \alpha. \qquad (2.2)$$

$R(\underline{x})$ in equation (2.2) is the multi-dimensional receptive field function which assigns an activation value to each point in the input space X.

2.2 Function Approximation of CMAC

The next result is of major importance as it shows these approximations provided by the CMAC neural network can be made arbitrarily accurate. It provides the design which shows explicitly how to design a CMAC NN for a prescribed approximation accuracy.

Theorem 2.2 : The function estimate $\hat{f}(\underline{x})$ provided by a CMAC uniformly approximates any C^1-continuous function $f(\underline{x}): R^n \to R^m \in F_L$ on $\Omega \subset R^n$. Specifically, given any $\mathcal{F}_L$ and $\varepsilon > 0$ the maximum partition size δ can be chosen such that

$$\left\| f(\underline{x}) - \hat{f}(\underline{x}) \right\| \le \varepsilon,\ \forall f(.) \in F_L \qquad (2.3)$$

where

$$\delta = \frac{\varepsilon}{mL}, \qquad (2.4)$$

Proof : See [3]. ∎

2.3 Implementation Properties of CMAC

The output equation of the CMAC can be represented as a function from R^n to R^m and expressed in vector notation as

$$h(\underline{x}) = w^T \Gamma(\underline{x}) \qquad (2.6)$$

where w is a matrix containing the set of weights, and Γ is a vector of the receptive field activation values. The definition of w and Γ is not unique, though $w^T\Gamma$ is equal to the right-hand side of equation (2.6).

Corollary 2.3: Any C^1 – function $f(.) \in F_L$ can be expressed as

$$f(\underline{x}) = w^T \Gamma(\underline{x}) + \varepsilon, \quad \forall \underline{x} \in U \tag{2.7}$$

where ε is the function estimation error and $\|\varepsilon\| \le \varepsilon_N$, with ε_N a known bound. ∎

The advantage of equation (2.6) can be immediately seen in light of Corollary 2.3. For any $\underline{x}$, since only 2^n receptive fields are active, only 2^n weights need to be adjusted to get exact function reconstruction using equation (2.7).

3. Dynamics of a Robot Manipulator

The dynamics of an n-link robot manipulator may be expressed in the Lagrange form as [15]

$$M(q)\ddot{q} + V_m(q,\dot{q})q + G(q) + F(\dot{q}) + d = \tau \tag{3.1}$$

with $q(t) \in R^n$ the joint variable vector, $M(q)$ the inertia matrix, $V_m(q,\dot{q})$ the coriolis/centripetal vector, and $F(\dot{q})$ the friction component. Bounded unknown disturbances are denoted by d(t) and τ is the control torque. It is assumed that d(t) is an unknown disturbance with a known upper bound d_M so that $\|d\| \le d_M$. The control problem is then, to design a control input τ such that the joint angles $q(t)$ track a desired trajectory $q_d(t)$.

The robot dynamics in (3.1) satisfy some important properties that simplify the control problem.

Property 1:

(a) The inertia matrix $M(q)$ is symmetric, positive-definite, and bounded so that $\mu_1 I \le M(q) \le \mu_2 I$ for all $q(t)$.

(b) The matrix $\dot{M} - 2V_m(q,\dot{q})$ is skew symmetric.

(c) $\|V_m\dot{q}\| \le v_B\|\dot{q}\|^2$ for constants v_B.

$\|F(\dot{q})\| \le v_B\|\dot{q}\| + k_B$ for constants v_B, k_B.

$$\|G(q)\| \le g_B \tag{3.2}$$

Given a desired arm trajectory $q_d(t)$, the tracking error is given by

$$e(t) = q(t) - q_d(t), \tag{3.3}$$

and the filtered tracking error is defined as

$$r = \dot{e} + \Lambda e \tag{3.4}$$

where $\Lambda = \Lambda^T > 0$. Differentiating $r(t)$ and using (3.1), the arm dynamics may be written in terms of the filtered tracking error as

$$M\dot{r} = -V_m r - d + f + \tau \tag{3.5}$$

where the nonlinear robot function is

$$f(x) \equiv -M(q)(\ddot{q}_d - \Lambda\dot{e}) - V_m(q,\dot{q})[\dot{q}_d - \Lambda e] - G(q) - F(\dot{q}) \tag{3.6}$$

and

$$x = \left[e^T \dot{e}^T \; q_d^T \; \dot{q}_d^T \; \ddot{q}_d^T\right]^T. \tag{3.7}$$

Define now a control input torque as

$$\tau = -k_v r - \hat{f} \tag{3.8}$$

where $\hat{f}(x)$ is an estimate of $f(x)$ and the gain matrix $k_v = k_v^T > 0$. The closed-loop system under this control input (3.8) becomes

$$M\dot{r} = -(k_v + V_m)r + \tilde{f} - d, \tag{3.9}$$

where the functional estimation error is given by

$$\tilde{f} = f - \hat{f}. \tag{3.10}$$

4. Adaptive-CMAC Control of a Manipulator

In the implementation of the controller (3.8), it is assumed that an estimate $\hat{f}(.)$ of the function $f(.)$ is available. According to (2.7), any nonlinear function can be approximated to any required degree of accuracy using a CMAC NN. The output of the CMAC NN is given as

$$\hat{f}(x) = \hat{W}^T\phi(x) \tag{4.1}$$

where $\hat{W}$ is a matrix of control representative values and $\phi(.)$ is the vector of membership values based on the multi-dimensional MFs. The carets (^) denote the estimates of the ideal parameter values that are provided by the tuning algorithms soon to be presented. The proposed control scheme is shown in Fig. 3.1. Note that the structure has a nonlinear CMAC NN inner loop plus a linear outer tracking loop. For such a network to ensure small tracking error in closed-loop control, the weights W associated with the network should be known. Since the weights are unknown in control applications, it is necessary to learn the nonlinear function on-line. Learning laws will now be derived that ensure stability of the overall filtered tracking error system (3.9).

Theorem 4.1: For the system in (3.1) let the inputs be selected as in (3.8). Further, let the estimate of the nonlinearity $\hat{f}(.)$ be manufactured by a *CMAC NN*. Let the weight estimate values of the CMAC NN controller be tuned on-line by the following update laws :

$$\dot{\hat{W}} = F\phi(x)r^T - \kappa F\|r\|\hat{W}, \tag{4.2}$$

with $F = F^T > 0$ a constant design matrix, and $\kappa > 0$ a design parameter. Then for large enough outer-loop gains k_v, the filtered tracking error $r(t)$, and the weight estimates are UUB. Further, the filtered tracking error can be

made arbitrarily small by proper selection of the feedback gains k_v.

Proof : See [3]. ∎

Let U be a compact set in R^n and $f(.) \in F_L$. Select the partitions π_i along each input dimension x_i such that the approximation property (2.7) holds. A positive definite Lyapunov function is selected. The derivative of the Lyapunov function is guaranteed to be negative as long as

$$\|r\| > \frac{\kappa \frac{W_{max}^2}{4} + (d_M + \varepsilon_N)}{k_{v_{min}}} = b_r, \qquad (4.3)$$

or

$$\|\tilde{w}\| > \frac{w_{max}}{2} + \sqrt{\frac{w_{max}^2}{4} + \frac{(d_M + \varepsilon_N)}{\kappa}} = b_w. \qquad (4.4)$$

Therefore, for all initial conditions satisfying

$\{\underline{x} : \|\underline{x}(t_0)\| \le \max\{b_r, r(t_0)\} + d_M\} \equiv U_0 \subset U$, the filtered tracking error r is non-increasing and the solution to equations (4.3),(4.4) evolves entirely within U where the approximation property (equation (2.7)) holds. Now as long as the filtered tracking error and the error in the weight estimates are bounded by the terms in equations (4.3) and (4.4), the Lyapunov function is decreasing. That is, the weight estimates and the filtered tracking error attain ultimate bounds given in equations (4.3),(4.4) and the solution to equations (3.1) evolves within U where $f(.)$ is Lipschitz. But the terms on the left-hand-side in equation (3.4) represent an exponentially stable linear system driven by the filtered tracking error r. This proves that the tracking error is also ultimately bounded. Therefore, for any initial condition within U_0 the system of equations (3.1) evolves entirely within U and attains the ultimate bounds given in equations (4.4),(4.5) thereby proving that the closed-loop system is UUB. ∎

Remarks: The first term in equations (4.2) is a gradient term while the second term is like $\varepsilon - \text{mod}$ term popular in adaptive control literature [13]. It is seen that the only equilibrium point for equation (4.2) is when r = 0, that is, when the CMAC is able to manufacture the nonlinearity exactly. Therefore, at equilibrium $\hat{W} = W$ if an exact reconstruction is possible. The rate of convergence of the weights depends on the filtered tracking error and is larger for larger values of the error. This guarantees stability of the closed-loop system while the CMAC is learning the dynamics of the system. The second term is necessary to overcome the requirement of *persistency of excitation* condition [13] for the convergence of the weights, and ensures robustness in the closed-loop system.

5. Simulation Results

As an example, the controller proposed in Sections 4 through 6 is tested on a two-link robot arm [26]. The joint variable is $q = [q_1, q_2]^T$. The arm parameters are taken as $l_1 = l_2 = 1\text{m}$, $m_1 = 1\text{kg}$, $m_2 = 2.3\text{kg}$. The desired trajectory was selected as $q_{d_1}(t) = 0.3\sin t$, for the case of joint 1 and $q_{d_2}(t) = 0.3\cos t$ for the joint 2. The input space was partitioned in a grid of size 0.25 and receptive fields are selected to cover the input space $\{[-0.5,0.5] \times [-0.5,0.5]\}$ along each input dimension. The initial conditions for both the states x_1 and x_2 and the CMAC NN weight estimates are taken to be zero. It is seen that although 625 weights are needed to define the outputs, only (2×2^4) weights are updated at any given instant. The controller parameters were selected as $k_v = \text{diag}\{5,5\}$, $\Lambda = \text{diag}\{5,5\}$ and the diagonal elements of the design matrix F are taken to be 10 with $\kappa = -2$.

The controller is designed assuming the dynamics of the robot arm are unknown. Figure 5.1(a) and (b) present the actual and desired response of the CMAC NN controller. It is seen from Figs 5.1 (a) and (b) that after a short initial learning phase, the system is able to track the desired trajectories satisfactorily. Fig. 5.2 presents the response of the PD controller without the CMAC NNs. From these figures, it is to be noted that the controller functions effectively in the presence of unknown dynamics. Furthermore, unlike conventional adaptive controllers, the CMAC NN does not require the computation of a regression vector nor does it make any linearity assumptions on the system parameters.

6. Conclusions

A comprehensive treatment of Cerebellar Model Articulation Controller (CMAC) Neural Network (NN) for the control of robot manipulators is presented. The structure and localized learning properties of

CMAC NN is exploited to design efficient controllers for nonlinear systems belong to a given useful class. Continuous-time implementation of these controllers are systematically examined.

The properties of the dynamics of the robot arm are employed to show the stability of the closed-loop system. The computation of the regression matrix is not performed and persistency of excitation is not required. Finally, the theoretical conclusions have been validated through empirical studies.

References

[1] J. S. Albus, J, "A New Approach to Manipulator Control : The Cerebellar Model Articulation Controller equations (CMAC)", Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control, vol. 97, series G, no. 3, pp. 220-227, 1975.

[2] Brown, M. and C.J. Harris, Neurofuzzy Adaptive Modeling and Control. Prentice Hall, Englewood Cliffs, New Jersey, 1994.

[3] S. Commuri, S. Jagannathan and F.L. Lewis, "CMAC Neural Network Control of Robot Manipulators", To appear in Journal of Robotic Systems, 1997.

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

[5] D. Ellison, "On the Convergence of the Multidimensional Albus Perceptron", The International Journal of Robotic Research, vol.10, no. 4, pp. 338-357, 1991.

[6] G. C. Goodwin and K. S. Sin, Adaptive Filtering, Prediction, and Control, Prentice-Hall Inc., Englewood Cliffs, NJ, 1984.

[7] S. Jagannathan and F. L. Lewis, "Multilayer discrete-time NN Controller with guaranteed performance", IEEE Trans. on Neural Networks, vol.7, no.1, pp.107-129, 1996.

[8] S. Jagannathan, "Discrete-Time Adaptive Control of Feedback Linearizable Nonlinear Systems Using Neural Networks", Proc. of the IEEE Conf. on Neural Networks, vol.4, June 1996.

[9] I. Kanellakopoulos, P. V. Kokotovic, and A. S. Morse, "Systematic design of adaptive controllers for feedback linearizable systems", IEEE Trans. on Automatic Control, vol.36, pp.1241-1253, 1991.

[10] I. Kanellakopoulos, "A discrete-time adaptive nonlinear system", IEEE Trans. on Automatic Control, vol.35, no.4, pp.416-424, 1994.

[11] L. G. Kraft and D.P. Campagna, "A Summary Comparison of CMAC Neural Network and Traditional Adaptive Control Systems", Handbook of Intelligent Control : Neural, Fuzzy, and Adaptive Approaches, pp. 143-169, van Nostrand Reinhold Publishing Company, New York, 1992.

[12] W. T. Miller, R. P. Hewes, F. H. Glanz and L. G. Kraft, "Real-Time Dynamic Control of an Industrial Manipulator Using a Neural-Network-Based Learning Controller", IEEE Trans. on Robotics and Automation, vol.6, no.1, pp.1-9, 1990.

[13] K. S. Narendra and A.M. Annaswamy, "A New Adaptive Law for Robust Adaptation Without Persistent Adaptation", IEEE Transactions on Automatic Control, vol. AC-32, no. 2, pp. 134-145, 1987.

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

[15] J-J. Slotine and W. Li, Applied Nonlinear Control, Prentice-Hall Inc., Englewood Cliffs, NJ, 1991.

[16] Y. Wong and A. Sideris, "Learning convergence in the Cerebellar Model Articulation Controller", IEEE Trans. on Neural Networks, vol.3, no.1, pp.115-121, 1992.

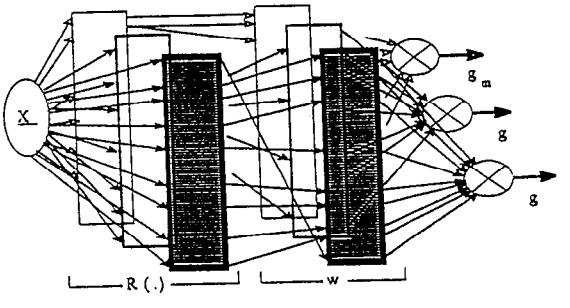

Fig. 2.1: A CMAC Neural Network.

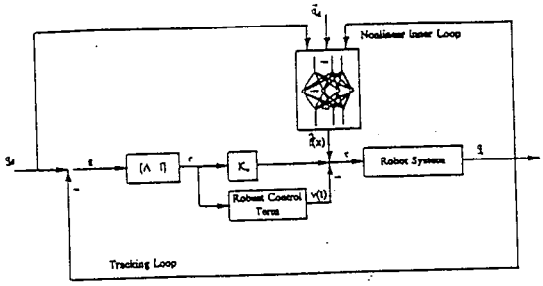

Fig. 3.1: The Continuous-Time CMAC NN Controller Structure.

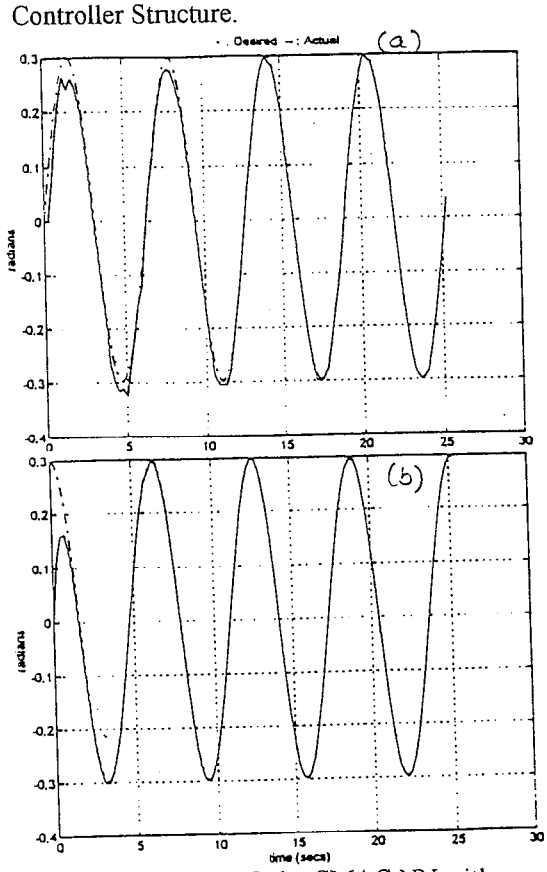

Fig. 5.1: Response of the CMAC NN with a PD controller. (a) Joint angle 1. (2) Joint angle 2.

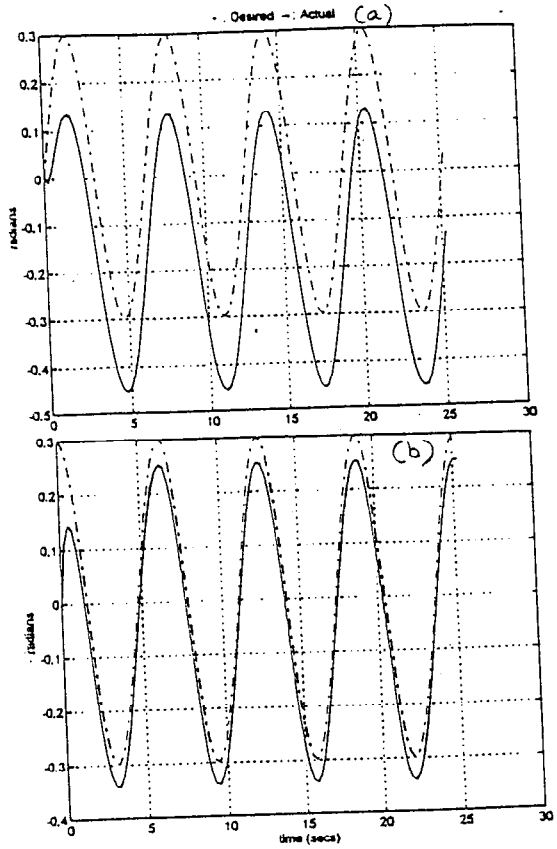

Fig. 5.2: Response of the PD controller without CMAC NN. (a) Joint angle 1. (b) Joint angle 2.