Journal of Applied Sciences

ISSN 1812-5654

science alert

ANSInet
an open access publisher
http://ansinet.com

Journal of Applied Sciences 6 (10): 2275-2280, 2006
ISSN 1812-5654

Identification and Trajectory Control of a Manipulator Arm Using a Neuro-Fuzzy Technique

[1]F. Arbaoui, [2]M.L.Saidi, [2]S. Kermiche and [2]H.A. Abbassi
[1]Department of Electrotechnics, Faculty of Sciences and Engineering,
20 Août 55 University B.P. 26 (21000), Route d'El-Hadaeik, Skikda, Algeria
[2]Department of Electronics, Faculty of Engineering Sciences,
Badji Mokhtar University, B.P.12 (23000), Annaba, Algeria

Abstract: Neural Network (NN) and Fuzzy Inference System (FIS) had been successfully employed in many engineering applications and were adopted in identification and designing controllers for robotic manipulators, the former because of its model-free feature and the other for its high flexibility. In this study, we focus on the application of a neuro-fuzzy technique to bring certain advantages over neural networks and fuzzy logic control for identification and tracking control of a robot manipulator which is a complicated multivariable nonlinear dynamical system. Neural network has the ability to learn by adjusting the interconnections between layers while fuzzy inference system is a computing framework based on the concept of fuzzy sets, fuzzy if-then rules and fuzzy reasoning. A Fuzzy Logic Controller (FLC) is combined with the neural network plant model trained on-line by the backpropagation algorithm using an adaptive learning rate.

Key words: Neural network, on-line identification, fuzzy controller, robotic tracking control

INTRODUCTION

Robotic manipulators are complicated nonlinear dynamical systems, time varying with inherent unmodeled dynamics, structured uncertainties caused by imprecision in the manipulator link proprieties and unforeseen loads and unstructured one, such as nonlinear friction, disturbances and the high-frequency part of the dynamics (Sun *et al*., 2001). Design of ideal controllers for such systems is one of the most challenging tasks in control theory today, especially when manipulators are asked to move very quickly while maintaining good dynamic performance (Arai *et al*., 1994).

In general, the control problem consists of obtaining dynamic models of the robotic system and using these models to determine control laws or strategies to achieve the desired system response and performance. A conventional approach to solve the robotic control problem is to use the computed torque algorithm (Er *et al*., 1997).

The main problems with conventional control in robotics control are (Peng and Won, 2002): need for mathematics model of the plant to be controlled, inability to face payload variation and disturbances and existence of plant uncertainties or sudden change of plant arameters leading up to inaccurate controller.

The emergence of softcomputing or computational intelligence technology inspired by biological and human intelligence is one of the most exciting and important fields in engineering. There are several approaches in configuring an intelligent control system such as neural networks which have the distinct learning and adaptive capabilities, fuzzy logic control theory, which can emulate human thinking and neuro-fuzzy control which possess certain advantages over neural networks and fuzzy logic control.

The using of FLC for controlling a robot manipulator is justified from the following reasons: the dynamics of robot is modeled by nonlinear and coupled differential equations and FLC gives high flexibility for its many degrees of freedom (shape and number of membership functions, aggregation methods, fuzzification and defuzzification methods, etc.). Fuzzy systems are suitable for uncertain and approximate reasoning, especially for the system with a mathematical model that is difficult to derive (Er *et al*., 1997).

The universal approximation capabilities of multilayer NN make it a popular choice for modeling nonlinear systems. It has been shown (Hornik *et al*., 1989) that a neural network with one hidden layer with an arbitrarily large number of neurons in the hidden layer can approximate any continuous function over a compact

Corresponding Author: F. Arbaoui, Department of Electronics, Faculty of Engineering Sciences, Badji Mokhtar University, B.P. 12 (23000), Annaba, Algeria

subset of $\mathbb{R}^n$. An identification of a robot dynamics with feedforward NN using supervised learning is adopted.

DYNAMIC MODEL OF ROBOT MANIPULATOR AND ACTUATORS

A robotic manipulator with n Degree Of Freedom (nDOF) can be modeled as a set of n rigid bodies connected in series with one end fixed to the ground and the other end free (Spong and Vidyasagar 1989). The bodies are jointed together with revolute or prismatic joints. A torque actuator acts at each joint. The dynamic equation of the manipulator using Lagrange formalism is given by:

$$\tau = D(q)\ddot{q} + V(q,\dot{q}) + F(q,\dot{q}) + G(q) + \tau_d \qquad (1)$$

where D (q) $\epsilon\mathbb{R}^{n\times n}$ is the inertia matrix (symmetric and positive definite), V (q, $\dot{q}$)ϵ $\mathbb{R}^n$ is the centrifugal and Coriolis vector, F (q, $\dot{q}$)ϵ $\mathbb{R}^n$ is the vector of viscous frictions and G (q) $\in \mathbb{R}^n$ is the gravity vector. $q \in \mathbb{R}^n, \tau \in \mathbb{R}^n$ and $\tau_d \in \mathbb{R}^n$ are generalised coordinates, applied joint torques and disturbance respectively.

The dynamic equation of a 2DOF planar robot manipulator (two links joined by rotary joints) shown in Fig. 1 is derived by using Euler-Lagrange method as follows:

$$\begin{bmatrix} D_{11}(q_2) & D_{12}(q_2) \\ D_{21}(q_2) & D_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} h(q_2)\dot{q}_2 & h(q_2)(\dot{q}_1 + \dot{q}_2) \\ -h(q_2)\dot{q}_1 & 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} g_1(q_1,q_2)g \\ g_2(q_1,q_2)g \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \qquad (2)$$

where,

$$D_{11}(q_2) = I_{l_{c_1}} + m_{l_1} l_{c_1}^2 + k_{r_1}^2 I_{m_1} + I_{l_{c_2}} + m_{l_2}(l_1^2 + l_{c_2}^2 + 2 l_1 l_{c_2} c_2) + I_{m_2} + m_{m_2} l_1^2,$$

$$D_{12}(q_2) = D_{21}(q_2) = I_{l_{c_2}} + m_{l_2}(l_{c_2}^2 + l_1 l_{c_2} c_2) + k_{r_2} I_{m_2}$$

$$D_{22}(q_2) = I_{l_{c_2}} + m_{l_2} + k_{r_2}^2 I_{m_1}$$

$$h(q_2) = -m_{l_2} l_1 l_{c_2} s_2$$

$$g_1(q_1,q_2) = (m_{l_2} l_{c_1} + m_{m_2} l_1 + m_{l_2} l_1) g c_1 + m_{l_2} c_{12},\ g_2(q_1,q_2) = m_{l_2} l_{c_2} g c_{12}$$

$$c_i = \cos(q_i),\ s_i = \sin(q_i),\ c_{ij} = \cos(q_i + q_j),$$

$$s_{ij} = \sin(q_i + q_j) \text{ and } g = 9.81$$

where l_1, l_2 denotes the length of the two links, l_{c_1} , l_{c_2} are the distances of the centers of mass of the two links from the respective joint axes, m_{l_1}, m_{l_2} are the masses of two links m_{l_1} ,m_{l_2} are the masses of the rotors of the two joints motors. The moments of inertia with respect to the axes of the two rotors and the moments of inertia relative to the centers of mass of the two links are denoted by I_{m_1} ,I_{m_2} and $I_{l_{c_1}}$,$I_{l_{c_2}}$ respectively. Also, it is assumed that the motors are located on the joints axes with centers of mass located at the origins of the respective frames.

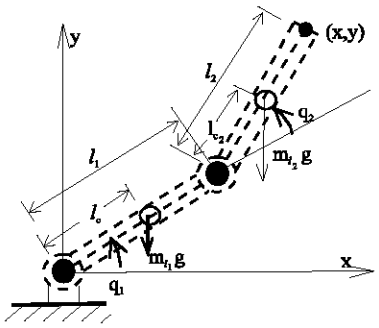

Fig. 1: 2DOF planar robot manipulator

DIRECT AND INVERSE KINEMATICS

Direct kinematics problem describes the end effector position and orientation as a function of the joint variables of the mechanical structure with respect to a reference frame. The result of direct kinematics function for a two planar manipulator is expressed by:

$$T(q) = \begin{bmatrix} 0 & s_{12} & c_{12} & a_1 c_1 + a_2 c_{21} \\ 0 & -c_{12} & s_{12} & a_1 s_1 + a_2 s_{21} \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3)$$

On the contrary, the inverse kinematics algorithm consists of the determination of joint variables corresponding to a given end effector. The solution of the inverse kinematics problem for two link planar manipulator (Fig. 1) is:

$$q_1 = \arccos(r/2) + \arccos(x/r),\ q_2 = -2\arccos(r/2) \qquad (4)$$

$$r = \sqrt{x^2 + y^2} \qquad (5)$$

TRAJECTORY PLANNER

The trajectory planner computes a function q_d (t) that completely specifies the motion of the robot as it traverses the path. We consider point to point motion to plan a trajectory from q (t_0) to q (t_f), i.e., the path is

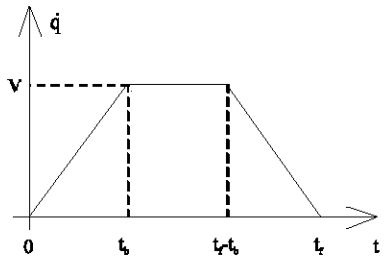

Fig. 2: Velocity shape

specified by its initial and final configurations. This type of motion is suitable for materials transfer tasks when the workspace is clear of obstacles. The problem here is to find a trajectory that connects an initial to a final configuration while satisfying other specified constraints at the endpoints (velocity and acceleration constraints). Since the control action on the manipulator is carried out in the joint space, an inverse kinematics algorithm is used to reconstruct the time sequence of joint variables corresponding to the above sequence in the operational space (Eq. 4 and 5). To generate suitable joint space trajectories we use the "Linear Segments with Parabolic Blends" method (LSPB). We specify the desired trajectory in three parts (Eq. 6). The first part from time t_0 to time t_b is a quadratic polynomial. This results in a linear "ramp" velocity (Fig. 2). At time t_b (blend time), the trajectory switches to a linear function. This corresponds to a constant velocity. Finally, at time t_f-t_b the trajectory switches once again, this time to a quadratic polynomial so that the velocity is linear. For convenience we suppose that $t_0 = 0$ and $\dot{q}(t_f) = \dot{q}(0) = 0$ (i.e., robot must start and end with zero velocity).The complete LSPB trajectory is given by (Spong and Vidyasagar, 1989):

$$q(t) = \begin{cases} q_0 + \frac{a}{2}t^2 & 0 < t \le t_b \\ \frac{q_f + q_0 - Vt_f}{2} + Vt & t_b < t \le t_f - t_b \\ q_f - \frac{at_f^2}{2} + at_f t - \frac{a}{2}t^2 & t_f - t_b < t \le t_f \end{cases} \quad (6)$$

where q_0 and q_f are an initial and final joint angle values and t_f is transition between two intermediate points.

CONTROL DESIGN

We can distinguish two main components in the designed system (Fig. 3): A Fuzzy Logic Controller (FLC) and a neural network identification structure of the plant. The aim of the motion is to find the generalized joint torques that allow execution of a motion q(t) so that

$$q(t) = q_d(t) \quad (7)$$

as close as possible, where $q_d(t)$ denotes the vector of desired joint trajectory variables. Tracking control is needed to make each joint track a desired trajectory. The output of the FLC, which is used for joint position control, is applied to the actuator (a dc servomotor). The outputs of the control system are measured joint angles or velocity. An on-line identification of robot manipulator using neural networks is achieved.

Design of the conventional fuzzy logic controller: The fuzzy controller (Lee, 1990) is shown in Fig. 4. In this study the Mamdani method of inference is used with two control inputs x_i (i=1, 2) and one output y_1. Control inputs are error position of the links and their derivations (e_1, de_1/dt, e_2, de_2/dt) and the outputs are the joint torque derivatives $d\tau_1$,$dt/d\tau 1/dt$. Fuzzy controller contains seven fuzzy rules obtained as follows:

If (e is NS) or (de is NS) then (dτ is PS)
If (e is PS) or (de is PS) then (dτ is NS)
If (e is Z) and (de is Z) then (dτ is Z)
If (e is NB) or (de is NB) then (dτ is PB)
If (e is PB) or (de is PB) then (dτ is NB)
If (de is PVB) then (dτ is NVB)
If (de is NVB) then (dτ is PVB)

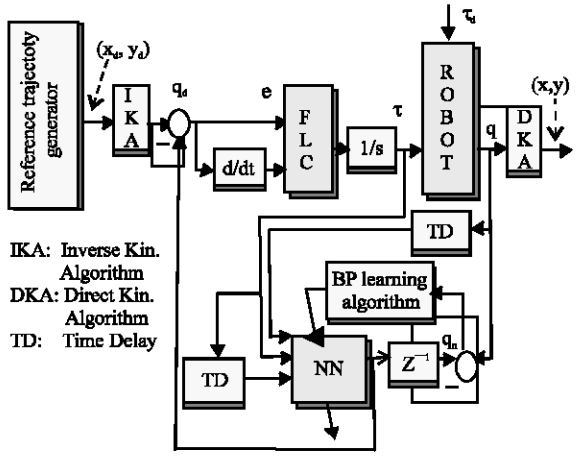

Fig. 3: Fuzzy control and on-line neural network identification system

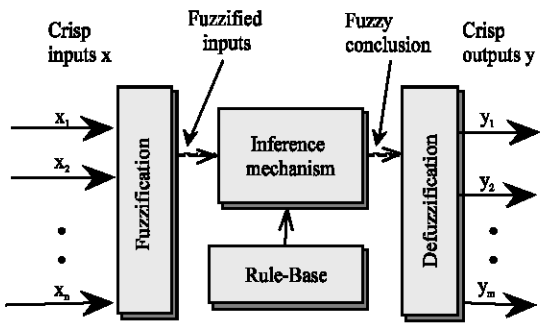

Fig. 4: The fuzzy logic controller architecture

Shapes of membership functions of input variables and output variable are depicted in Fig. 5-7, respectively.

The proposed fuzzy sets presented here are NB: Negative Big, NS: Negative Small, Z: Zero, PS: Positive Small, PB: Positive Big. Fuzzy sets NVB (Negative Very Big) and PVB (Positive Very Big) are introduced to eliminate too big changes of joint angles and torques.

On-line identification using Neural Network: On-line system identification methods used are mostly based on recursive implementation of off-line methods for models that are linear in the parameters such as least squares. In order to relax the linearity in the parameter assumption, neural networks (NNs) are being widely employed for system identification, since these networks learn complex mappings from a set of examples. NNs present an excellent candidate for modeling nonlinear systems. A feedforward NN with one hidden layer using supervised learning was trained for this purpose. The well-known backpropagation algorithm with adaptive learning rate (Jacobs, 1988) is employed to train the network for updating synaptic weights. The supervised learning is used for identifying process models from input/output data. The identification procedure with NNs is used to reproduce the behavior of the robot manipulator and the actuators. If we suppose that the process is described by the following non-linear discrete time difference equation:

$$q(t) = f[q(t-1),..., q(t-n); \tau(t-1),..., \tau(t-m)] \quad (8)$$

The process output q(t) at time t depends both on its past n values q (t-i) (i = 1,…,n) as well as the past m values of the input τ(t-j) (j=0,…,m). The series-parallel identification model (Fig. 8) corresponding to the process represented in Eq. 8 has the following form (Narendra and Parthasarathy 1990):

$$q_n(w, t) = f_w[q(t-1),..., q(t-n); \tau(t), \tau(t-1),...\tau(t-m)], m \le n \quad (9)$$

The training process for neural network nonparametric modeling can be expressed uniformly as the minimization of an error measure, typically sum-squared error, between the neural network output and the process output. If the sampled process data are collected over a period (0,T), the cost function J(w,t) is the following:

$$J(w, t) = \sum_{t=0}^{t}[q(t) - q_n((w, t)]^2 \quad (10)$$

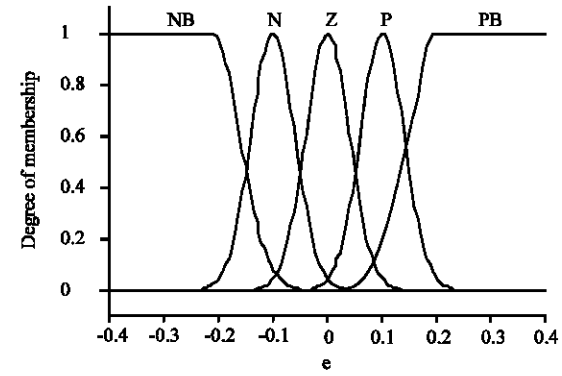

Fig. 5: Membership functions for input *e*

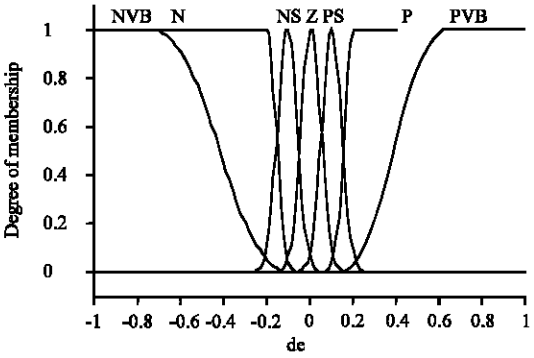

Fig. 6: Membership functions for input *de*

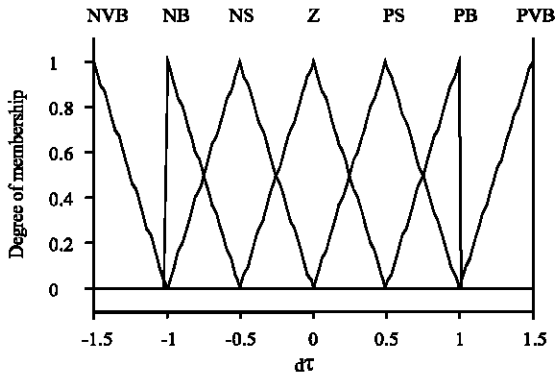

Fig. 7: Membership functions for output *dτ*

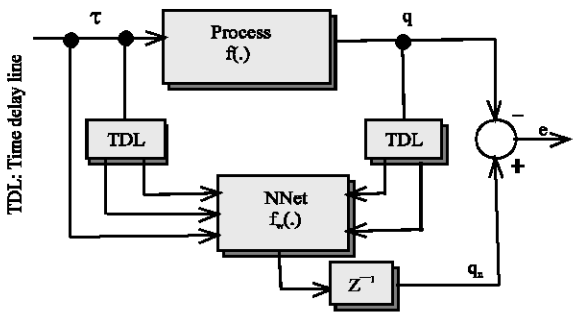

Fig. 8: Series-parallel identification model

The minimization is carried out with backpropagation algorithm through time. The relative factor $\chi(t)$ defined as follows:

$$\chi(t)=\frac{\Delta J(w,t)}{J(w,t)}=\frac{J(w,t)-J(w,t-1)}{J(w,t)} \quad (11)$$

is employed to adjust learning rate term η at each iteration according the following formula:

$$\eta(t+1)=\begin{cases}\eta(t)[1+\upsilon e^{-x(t)}] \text{for } x^{(t)}<1\\ \eta(t)[1-\upsilon e^{-x(t)}] \text{for } x^{(t)}\geq 0\end{cases},\upsilon\varepsilon[0,1] \quad (12)$$

SIMULATION RESULTS

Let's consider the 2DOF planar manipulator shown in Fig. 1. The parameters of the robot manipulator and joint actuators are taken as:

$l_1=l_2=l$ [m], $l_{c_1}=l_{c_2}=0.5$[m], $m_{l_1}=m_{l_2}=50$ [kg]

$I_{l_{c_1}}=I_{l_{c_2}}=10$ [kg.m^2], $m_{m_2}=m_{m_2}=5$ [kg] ,

$I_{m_1}=I_{m_2}=0.01$ [kg.m^2], $k_{r_1}=k_{r_2}=100$.

The desired trajectory in operational space is given by way point (Fig. 9). Actual robotic trajectory is presented in the same figure with a solid line. Transition between two intermediate way points is governed by trapezoidal joint velocity profile. A quite satisfactory control result is obtained, although only seven rules are used to design the fuzzy control law. The proposed fuzzy controller is able to track well the path.

The identification of the robot manipulator is performed in on-line mode during the fuzzy control of

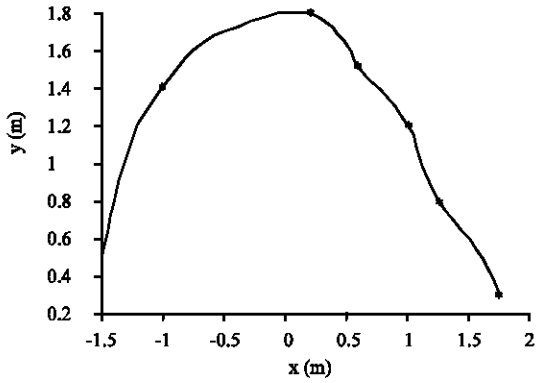

Fig. 9: Desired (*) and actual (-) trajectory in the Cartesian space

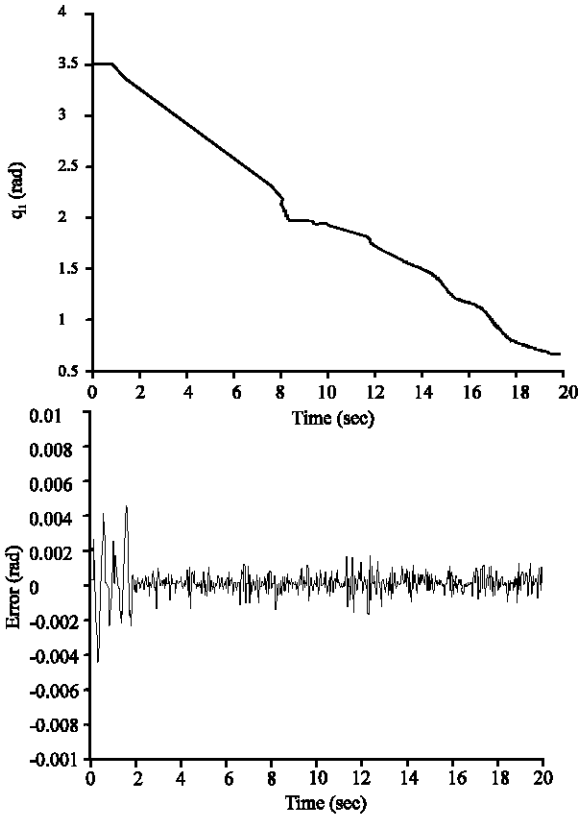

Fig. 10: Desired (solid line) and actual (dotted line) values of the joint angle q_1 and error between them

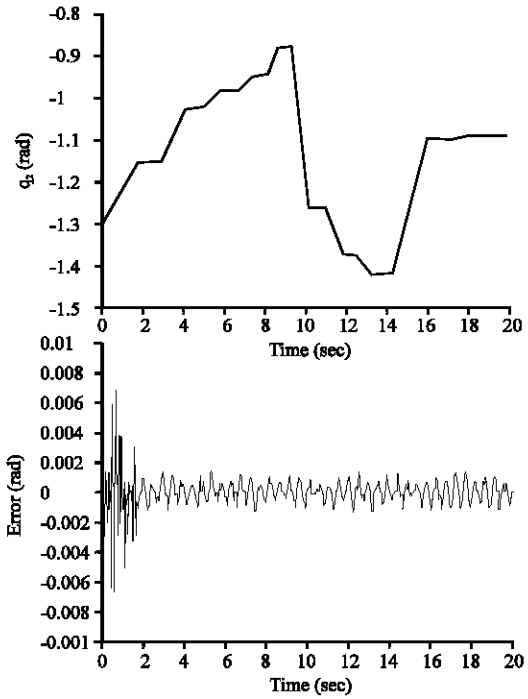

Fig. 11: Desired (solid line) and actual (dotted line) values of the joint angleq_2 and error between them

manipulator. Neural network contains 10 sigmoid neurons in hidden layer and 4 linear neurons in its output layer. The proposed algorithm started from the same initial learning rate $\eta = 0.03$ for both Layers (hidden and output). The best results were obtained with $\upsilon = 0.8$.

Comparison between actual and reference trajectory for both joints are shown in Fig. 10 and 11. The errors tracking have bigger values at the beginning. This is due to initialization of the neural network with random number values. However, after short time (about 2 s) the neural network output follows the robot trajectory with small error.

CONCLUSIONS

This study has demonstrated the applications of neural network and fuzzy logic system for the identification and control of a robot manipulator. Fuzzy logic controller was used for robot position control and neural network was proposed for on-line identification of the robotic manipulator dynamics during the motion control. On-line parameter training is derived using the gradient descent method with adaptive learning rate. The effectiveness of the proposed hybrid identification and control scheme has been confirmed by simulated results. The simulation results show that the designed fuzzy controller and neural network are able to provide satisfactory performance for both trajectory tracking and identification capabilities. The neuro-fuzzy control is an attractive and useful tool in robotics field.

REFERENCES

Arai, H., K. Tanic and S. Tachi, 1994. Path tracking of a manipulator considering torque saturation. IEEE Trans. Indus. Electron., 40: 25-31.

Er, M.J., S.M. Yap, P.W. Yeaw and F.L. Luo, 1997. A review of neural-fuzzy controllers for robotic manipulators. Industry Applications Conference, 1997. Thirty-Second IAS Annual Meeting, AS '97, Conference Record of the 1997 IEEE., 2: 812-819.

Hornik, K., M. Stinchcombe and H. White, 1989. Multilayer feedforward networks are universal approximator. Neural Networks, 2: 359-366.

Jacobs, R.A., 1988. Increased rate of convergence through learning rate adaptation. Neural Networks, 1: 295-307.

Lee, C.C., 1990, Fuzzy logic in control systems: Fuzzy logic-part I and II. IEEE Trans. Syst. Man. Cybern., 20: 404-435.

Narendra, K.S. and K. Parthasarathy, 1990. Identification and control of dynamical systems using neural networks. IEEE Trans. Neural Networks, 1: 4-27.

Peng, L. and P.Y. Won, 2002. Neural-fuzzy control system for robotic manipulators. IEEE Control Sys., 22: 53-63.

Spong, M.W. and M. Vidyasagar, 1989. Robot Dynamics and Control. John Wiley.

Sun, F., Z. Sun and P.Y. Woo, 2001. Neural network-based adaptive controller design of robotic manipulators with an observer. IEEE Trans. Neural Networks, 12: 54-67.