DECENTRALIZED NEURAL BACKSTEPPING CONTROL FOR AN INDUSTRIAL PA10-7CE ROBOT ARM

R. Garcia-Hernandez[1], E. N. Sanchez[2], M. A. Llama[3] and J. A. Ruz-Hernandez[1]

[1]*Facultad de Ingenieria, Universidad Autonoma del Carmen, Av. 56 No. 4, Cd. del Carmen, Campeche, Mexico*

[2]*Centro de Investigacion y de Estudios Avanzados del IPN, Unidad Guadalajara, Guadalajara, Jalisco, Mexico*

[3]*Division de Estudios de Posgrado, Instituto Tecnologico de la Laguna, Torreon, Coahuila, Mexico*

Keywords: High-order neural network, Extended Kalman filter, Backstepping, Trajectory tracking, Robot arm.

Abstract: This paper presents a discrete-time decentralized control strategy for trajectory tracking of a seven degrees of freedom (DOF) robot arm. A high order neural network (HONN) is used to approximate a decentralized control law designed by the backstepping technique as applied to a block strict feedback form (BSFF). The neural network learning is performed online by extended Kalman filter. The local controller for each joint use only local angular position and velocity measurements. The feasibility of the proposed scheme is illustrated via simulation.

SCIENCE AND TECHNOLOGY PUBLICATIONS

1 INTRODUCTION

Nowadays, industrial robots have gained wide popularity as essential components in the construction of automated systems. Reduction of manufacturing costs, increase of productivity, improvement of product quality standards, and the possibility of eliminating harmful of repetitive tasks for human operators represent the main factors that have determined the spread of the robotic technology in the manufacturing industry. Industrial robots are suitable for applications where high precision, repeatability and tracking accuracy are required.

In this context, a variety of control schemes have been proposed in order to guarantee efficient trajectory tracking and stability (Sanchez and Ricalde, 2003), (Santibañez et al., 2005). Fast advance in computational technology offers new ways for implementing control algorithms within the approach of a centralized control design.However, there is a great challenge to obtain an efficient control for this class of systems, due to its highly nonlinear complex dynamics, the presence of strong interconnections, parameters difficult to determine, and unmodeled dynamics. Considering only the most important terms, the mathematical model obtained requires control algorithms with great number of mathematical operations, which affect the feasibility of real-time implementations.

On the other hand, within the area of control systems theory, for more than three decades, an alternative approach has been developed considering a global system as a set of interconnected subsystems, for which it is possible to design independent controllers, considering only local variables to each subsystem: the so called decentralized control (Huang et al., 2003). Decentralized control has been applied in robotics, mainly in cooperative multiple mobile robots and robot manipulators, where it is natural to consider each mobile robot or each part of the manipulator as a subsystem of the whole system. For robot manipulators each joint and the respective link is considered as a subsystem in order to develop local controllers, which just consider local angular position and angular velocity measurements, and compensate the interconnection effects, usually assumed as disturbances. The resulting controllers are easy to implement for real-time applications (Liu, 1999).

In (Ni and Er, 2000), a decentralized control of robot manipulators is developed, decoupling the dynamic model of the manipulator in a set of linear subsystems with uncertainties; simulation results for a robot of two joints are shown. In (Karakasoglu et al., 1993), an approach of decentralized neural identification and control for robots manipulators is presented using models in discrete-time. In (Safaric and Rodic, 2000), a decentralized control for robot manipulators is reported; it is based on the estimation of each joint dynamics, using feedforward neural networks.

In recent literature about adaptive and robust control, numerous approaches have been proposed for

Garcia Hernandez R., N. Sanchez E., A. Llama M. and A. Ruz-Hernandez J..
DECENTRALIZED NEURAL BACKSTEPPING CONTROL FOR AN INDUSTRIAL PA10-7CE ROBOT ARM.
DOI: 10.5220/0003684300820089
In *Proceedings of the International Conference on Neural Computation Theory and Applications* (NCTA-2011), pages 82-89
ISBN: 978-989-8425-84-3
Copyright © 2011 SCITEPRESS (Science and Technology Publications, Lda.)

the design of nonlinear control systems. Among these, adaptive backstepping constitutes a major design methodology (Krstic et al., 1995). The idea behind the backstepping approach is that some appropriate functions of state variables are selected recursively as virtual control inputs for lower dimension subsystems of the overall system. Each backstepping stage results in a new virtual control design from the preceding stages; when the procedure ends, a feedback design for the true control input results, which achieves the original design objective.

In this paper, the authors propose a decentralized approach in order to design a suitable controller for each subsystem. Afterwards, each local controller is approximated by a high order neural network (HONN) (Ge et al., 2004). The neural network (NN) training is performed on-line by means of an extended Kalman filter (EKF) (Alanis et al., 2007), and the controllers are designed for each joint, using only local angular position and velocity measurements. Simulations for the proposed control scheme using a Mitsubishi PA10-7CE robot arm are presented.

2 DISCRETE-TIME DECENTRALIZED SYSTEMS

Let consider a class of discrete-time nonlinear perturbed and interconnected system which can be presented in the block strict feedback form (BSFF) (Krstic et al., 1995) consisting of r blocks

$$\begin{aligned}
\chi_i^1(k+1) &= f_i^1(\chi_i^1) + B_i^1(\chi_i^1)\chi_i^2 + \Gamma_{i\ell}^1 \\
\chi_i^2(k+1) &= f_i^2(\chi_i^1,\chi_i^2) + B_i^2(\chi_i^1,\chi_i^2)\chi_i^3 + \Gamma_{i\ell}^2 \\
&\vdots \\
\chi_i^{r-1}(k+1) &= f_i^{r-1}(\chi_i^1,\chi_i^2,\ldots,\chi_i^{r-1}) \\
&\quad + B_i^{r-1}(\chi_i^1,\chi_i^2,\ldots,\chi_i^{r-1})\chi_i^r + \Gamma_{i\ell}^{r-1} \\
\chi_i^r(k+1) &= f_i^r(\chi_i) + B_i^r(\chi_i)u_i + \Gamma_{i\ell}^r
\end{aligned} \tag{1}$$

where $\chi_i \in \Re^{n_i}$, $\chi_i = \left[\chi_i^{1\top}\ \chi_i^{2\top}\ \ldots\ \chi_i^{r\top}\right]^\top$ and $\chi_i^j \in \Re^{n_{ij}\times 1}$, $\chi_i^j = \left[\chi_{i1}^j\ \chi_{i2}^j\ \ldots\ \chi_{il}^j\right]^\top$, $i = 1,\ldots,N$; $j = 1,\ldots,r$; $l = 1,\ldots,n_{ij}$; N is the number of subsystems, $u_i \in \Re^{m_i}$ is the input vector, the rank of $B_i^j = n_{ij}$, $\sum_{j=1}^r n_{ij} = n_i$, $\forall \chi_i^j \in D_{\chi_i^j} \subset \Re^{n_{ij}}$. We assume that f_i^j, B_i^j and Γ_i^j are smooth and bounded functions, $f_i^j(0) = 0$ and $B_i^j(0) = 0$. The integers $n_{i1} \le n_{i2} \le \cdots \le n_{ij} \le m_i$ define the different subsystem structures. The interconnection terms are given by

$$\begin{aligned}
\Gamma_{i\ell}^1 &= \sum_{\ell=1,\ \ell\neq i}^{N} \gamma_{i\ell}^1(\chi_\ell^1) \\
\Gamma_{i\ell}^2 &= \sum_{\ell=1,\ \ell\neq i}^{N} \gamma_{i\ell}^2(\chi_\ell^1,\chi_\ell^2) \\
&\vdots \\
\Gamma_{i\ell}^{r-1} &= \sum_{\ell=1,\ \ell\neq i}^{N} \gamma_{i\ell}^{r-1}(\chi_\ell^1,\chi_\ell^2,\ldots,\chi_\ell^{r-1}) \\
\Gamma_{i\ell}^r &= \sum_{\ell=1,\ \ell\neq i}^{N} \gamma_{i\ell}^r(\chi_\ell)
\end{aligned} \tag{2}$$

where χ_ℓ represents the state vector of the ℓ-th subsystem with $1 \le \ell \le N$ and $\ell \neq i$.

Interconnection terms (2) reflect the interaction between the i-th subsystem and the other ones.

3 HIGH-ORDER NEURAL NETWORKS

3.1 Discrete-time HONN

Let consider the HONN described by

$$\begin{aligned}
\phi(w,z) &= w^\top S(z) \\
S(z) &= [s_1^\top(z), s_2^\top(z), \cdots, s_m^\top(z)] \\
s_i(z) &= \left[\prod_{j\in I_1}[s(z_j)]^{d_j(i_1)} \cdots \prod_{j\in I_m}[s(z_j)]^{d_j(i_m)}\right]^\top \\
i &= 1,2,\cdots,L
\end{aligned} \tag{3}$$

where $z = [z_1, z_2, \cdots, z_p]^\top \in \Omega_z \subset \Re^p$, p is a positive integer which denotes the number of external inputs, L denotes the neural network node number, $\phi \in \Re^m$, $\{I_1, I_2, \cdots, I_L\}$ is a collection of not ordered subsets of $\{1,2,\cdots,p\}$, $S(z) \in \Re^{L\times m}$, $d_j(i_j)$ is a nonnegative integer, $w \in \Re^L$ is an adjustable synaptic weight vector, and $s(z_j)$ is chosen as the hyperbolic tangent function:

$$s(z_j) = \frac{e^{z_j} - e^{-z_j}}{e^{z_j} + e^{-z_j}} \tag{4}$$

For a desired function $u^* \in \Re^m$, assume that there exists an ideal weight vector $w^* \in \Re^L$ such that the smooth function vector $u^*(z)$ can be approximated by an ideal neural network on a compact subset $\Omega_z \subset \Re^q$

$$u^*(z) = w^{*\top} S(z) + \varepsilon_z \tag{5}$$

where $\varepsilon_z \subset \Re^m$ is the bounded neural network approximation error vector; note that $\|\varepsilon_z\|$ can be reduced by increasing the number of the adjustable weights. The ideal weight vector w^* is an artificial quantity required only for analytical purposes (Ge et al., 2004), (Rovithakis and Christodoulou, 2000). In general, it

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

is assumed that there exists an unknown but constant weight vector w^*, whose estimate is $w \in \Re^L$. Hence, it is possible to define:

$$\tilde{w}(k) = w(k) - w^* \tag{6}$$

as the weight estimation error.

3.2 EKF Training Algorithm

It is known that Kalman filtering (KF) estimates the state of a linear system with additive state and output white noises (Song and Grizzle, 1995). For KF-based neural network training, the network weights become the states to be estimated. In this case, the error between the neural network output and the measured plant output can be considered as additive white noise. Due to the fact that neural network mapping is nonlinear, an EKF-type is required.

The training goal is to find the optimal weight values which minimize the prediction error. We use a EKF-based training algorithm described by:

$$\begin{aligned} K_i^j(k) &= P_i^j(k)H_i^j(k)M_i^j(k) \\ w_i^j(k+1) &= w_i^j(k) + \eta_i^j K_i^j(k)e_i^j(k) \\ P_i^j(k+1) &= P_i^j(k) - K_i^j(k)H_i^{jT}(k)P_i^j(k) + Q_i^j(k) \end{aligned} \tag{7}$$

with

$$M_i^j(k) = [R_i^j(k) + H_i^{jT}(k)P_i^j(k)H_i^j(k)]^{-1} \tag{8}$$

where $P \in \Re^{L\times L}$ is the prediction error covariance matrix, $w \in \Re^L$ is the weight (state) vector, η is the rate learning parameter such that $0 \leq \eta \leq 1$, L is the respective number of neural network weights, $x \in \Re^m$ is the measured plant state, $\hat{x} \in \Re^m$ is the neural network output, $K \in \Re^{L\times m}$ is the Kalman gain matrix, $Q \in \Re^{L\times L}$ is the state noise associated covariance matrix, $R \in \Re^{m\times m}$ is the measurement noise associated covariance matrix, and $H \in \Re^{L\times m}$ is a matrix, for which each entry (H_{ij}) is the derivative of one of the neural network output $(\hat{x}_i)$, with respect to one neural network weight (w_j), as follows

$$H_{ij}(k) = \left[\frac{\partial \hat{x}_i(k)}{\partial w_j(k)}\right] \tag{9}$$

where $i = 1,\ldots,m$ and $j = 1,\ldots,L$. Usually P and Q are initialized as diagonal matrices, with entries $P(0)$ and $Q(0)$, respectively. It is important to remark that $H(k)$, $K(k)$, and $P(k)$ for the EKF are bounded (Song and Grizzle, 1995).

4 CONTROLLER DESIGN

Once the system in the BSFF is defined, we apply the well-known backstepping technique (Krstic et al., 1995). We can define the desired virtual controls $(\alpha_i^{j*}(k), i = 1,\ldots,N; j = 1,\ldots,r-1)$ and the ideal practical control $(u^*(k))$ as follows:

$$\begin{aligned} \alpha_i^{1*}(k) &\triangleq x_i^2(k) = \varphi_i^1(\bar{x}_i^1(k), x_{id}(k+r)) \\ \alpha_i^{2*}(k) &\triangleq x_i^3(k) = \varphi_i^2(\bar{x}_i^2(k), \alpha_i^{1*}(k)) \\ &\vdots \\ \alpha_i^{r-1*}(k) &\triangleq x_i^r(k) = \varphi_i^{r-1}(\bar{x}_i^{r-1}(k), \alpha_i^{r-2*}(k)) \\ u_i^*(k) &= \varphi_i^r(x_i(k), \alpha_i^{r-1*}(k)) \\ \chi_i(k) &= x_i^1(k) \end{aligned} \tag{10}$$

where $\varphi_i^j(\cdot)$ with $1 \leq j \leq r$ are nonlinear smooth functions. It is obvious that the desired virtual controls $\alpha_i^*(k)$ and the ideal control $u_i^*(k)$ will drive the output $\chi_i(k)$ to track the desired signal $x_{id}(k)$. Let us approximate the virtual controls and practical control by the following HONN:

$$\begin{aligned} \alpha_i^j(k) &= w_i^{j\top} S_i^j(z_i^j(k)) \\ u_i(k) &= w_i^{r\top} S_i^r(z_i^r(k)), \quad j = 1,\cdots,r-1 \end{aligned} \tag{11}$$

with

$$\begin{aligned} z_i^1(k) &= [x_i^1(k), x_{id}^1(k+r)]^\top \\ z_i^j(k) &= [\bar{x}_i^j(k), \alpha_i^{j-1}(k)]^\top, \quad j = 1,\cdots,r-1 \\ z_i^r(k) &= [x_i(k), \alpha_i^{r-1}(k)]^\top \end{aligned}$$

where $w_i^j \in \Re^{L_j}$ are the estimates of ideal constant weights w_i^{j*} and $S_i^j \in \Re^{L_j\times n_j}$ with $j = 1,\ldots,r$. Define the weight estimation error as

$$\tilde{w}_i^j(k) = w_i^j(k) - w_i^{j*}. \tag{12}$$

Then, the corresponding weights updating laws are defined as

$$w_i^j(k+1) = w_i^j(k) + \eta_i^j K_i^j(k)e_i^j(k) \tag{13}$$

with

$$\begin{aligned} K_i^j(k) &= P_i^j(k)H_i^j(k)M_i^j(k) \\ M_i^j(k) &= [R_i^j(k) + H_i^{j\top}(k)P_i^j(k)H_i^j(k)]^{-1} \\ P_i^j(k+1) &= P_i^j(k) - K_i^j(k)H_i^{j\top}(k)P_i^j(k) + Q_i^j(k) \end{aligned} \tag{14}$$

$$H_i^j(k) = \left[\frac{\partial \hat{\upsilon}_i^j(k)}{\partial w_i^j(k)}\right] \tag{15}$$

and

$$e_i^j(k) = \upsilon_i^j(k) - \hat{\upsilon}_i^j(k) \tag{16}$$

where $\upsilon_i^j(k) \in \Re^{n_j}$ is the desired signal and $\hat{\upsilon}_i^j(k) \in$

$\Re^{n_j}$ is the HONN function approximation defined, respectively as follows

$$\begin{aligned} \upsilon_i^1(k) &= x_{i\mathrm{d}}^1(k) \\ \upsilon_i^2(k) &= x_i^2(k) \\ &\vdots \\ \upsilon_i^r(k) &= x_i^r(k) \end{aligned} \tag{17}$$

and

$$\begin{aligned} \hat{\upsilon}_i^1(k) &= \chi_i^1(k) \\ \hat{\upsilon}_i^2(k) &= \alpha_i^1(k) \\ &\vdots \\ \hat{\upsilon}_i^r(k) &= \alpha_i^{r-1}(k) \end{aligned} \tag{18}$$

$e_i^j(k)$ denotes the error at each step as

$$\begin{aligned} e_i^1(k) &= x_{i\mathrm{d}}^1(k) - \chi_i^1(k) \\ e_i^2(k) &= x_i^2(k) - \alpha_i^1(k) \\ &\vdots \\ e_i^r(k) &= x_i^r(k) - \alpha_i^{r-1}(k). \end{aligned} \tag{19}$$

The whole proposed neural backstepping control scheme is shown in Fig. 1.

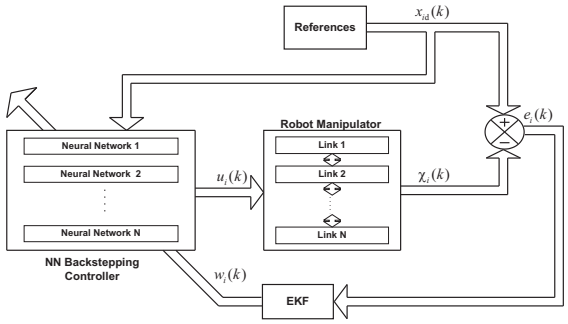

Figure 1: Decentralized neural backstepping control scheme.

5 SEVEN DOF MITSUBISHI PA10-7CE ROBOT ARM

5.1 Robot Description

The Mitsubishi PA10-7CE arm is an industrial robot manipulator which completely changes the vision of conventional industrial robots. Its name is an acronym of Portable General-Purpose Intelligent Arm. There exist two versions (Higuchi et al., 2003): the PA10-6C and the PA10-7C, where the suffix digit indicates the number of degrees of freedom of the arm. This work focuses on the study of the PA10-7CE model, which is the enhanced version of the PA10-7C. The PA10 arm is an open architecture robot; it means that it possesses:

- A hierarchical structure with several control levels.
- Communication between levels, via standard interfaces.
- An open general purpose interface in the higher level.

This scheme allows the user to focus on the programming of the tasks at the PA10 system higher level, without regarding on the operation of the lower levels. The programming can be performed using a high level language, such as Visual BASIC or Visual C++, from a PC with Windows operating system. The PA10 robot is currently the open architecture robot more employed for research (Jamisola et al., 2004), (Kennedy and Desai, 2003).The PA10 system is composed of four sections or levels, which conform a hierarchical structure:

Level 4: Operation control section (OCS); formed by the PC and the teaching pendant.

Level 3: Motion control section (MCS); formed by the motion control and optical boards.

Level 2: Servo drives.

Level 1: Robot arm.

Figure 2 shows the PA10-7CE robot arm. The PA10 robot is a 7-DOF redundant manipulator with revolute joints. Figure 3 shows a diagram of the PA10 arm, indicating the positive rotation direction and the respective names of each of the joints..

5.2 Control Objective

The decentralized discrete-time model for a seven DOF robot arm can be represented as follows

$$\begin{aligned} \chi_i^1(k+1) &= f_i^1(\chi_i^1) + B_i^1(\chi_i^1)\chi_i^2 + \Gamma_i^1 \\ \chi_i^2(k+1) &= f_i^2(\chi_i^1, \chi_i^2) + B_i^2(\chi_i^1, \chi_i^2)u_i(k) + \Gamma_i^2 \end{aligned} \tag{20}$$

where $i = 1, \ldots, 7$; $\chi_i^1(k)$ are the angular positions, $\chi_i^2(k)$ are the angular velocities, $u_i(k)$ represents the applied torque to i-th joint respectively. $f_i^j(\cdot)$ and $B_i^j(\cdot)$ depend only on the local variables and Γ_i^j are the interconnection effects.

Let define the following states:

$$x^1(k) = \left[\begin{array}{ccccccc} \chi_1^1 & \chi_2^1 & \chi_3^1 & \chi_4^1 & \chi_5^1 & \chi_6^1 & \chi_7^1 \end{array} \right]^\top$$

$$x^2(k) = \left[\begin{array}{ccccccc} \chi_1^2 & \chi_2^2 & \chi_3^2 & \chi_4^2 & \chi_5^2 & \chi_6^2 & \chi_7^2 \end{array} \right]^\top$$

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

Figure 2: Mitsubishi PA10-7CE robot arm.

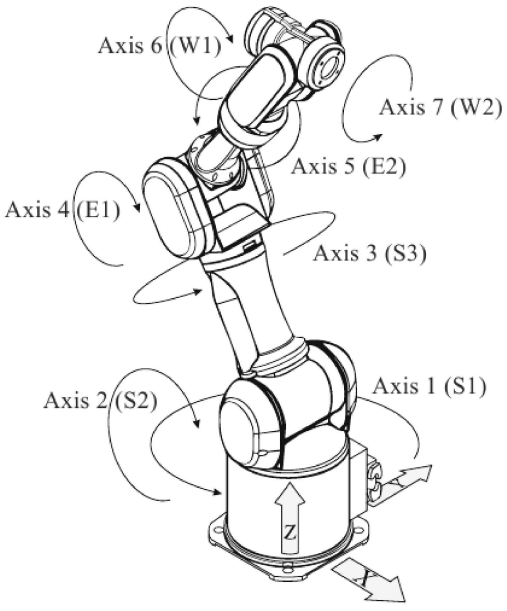

Figure 3: Mitsubishi PA10-7CE robot axes.

$$u(k) = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 \end{bmatrix}^\top$$
$$x_{\mathrm{d}}^1(k) = \begin{bmatrix} x_{1\mathrm{d}}^1 & x_{2\mathrm{d}}^1 & x_{3\mathrm{d}}^1 & x_{4\mathrm{d}}^1 & x_{5\mathrm{d}}^1 & x_{6\mathrm{d}}^1 & x_{7\mathrm{d}}^1 \end{bmatrix}^\top$$
$$\chi_i^1(k) = x_i^1(k) \tag{21}$$

where $x_{1\mathrm{d}}^1(k)$ to $x_{7\mathrm{d}}^1(k)$ are the desired trajectory signals. The control objective is to drive the output $\chi_i^1(k)$ to track the reference $x_{i\mathrm{d}}^1(k)$. Using (21), the system (20) can be represented in the block strict feedback form as

$$\begin{aligned} x_i^1(k+1) &= f_i^1(x_i^1(k)) + g_i^1(x_i^1(k))x_i^2(k) \\ x_i^2(k+1) &= f_i^2(\overline{x}_i^2(k)) + g_i^2(\overline{x}_i^2(k))u_i(k) \end{aligned} \tag{22}$$

where $\overline{x}_i^2(k) = \begin{bmatrix} x_i^1(k) & x_i^2(k) \end{bmatrix}^\top$, $i = 1,\ldots,7$, $f_i^1(x_i^1(k))$, $g_i^1(x_i^1(k))$, $f_i^2(\overline{x}_i^2(k))$ and $g_i^2(\overline{x}_i^2(k))$ are assumed to be unknown. To this end, we use a HONN to approximate the desired virtual controls and the ideal practical control described as

$$\begin{aligned} \alpha_i^{1*}(k) \triangleq x_i^2(k) &= \varphi_i^1(x_i^1(k), x_{i\mathrm{d}}^1(k+2)) \\ u_i^*(k) &= \varphi_i^1(x_i^1(k), x_i^2(k), \alpha_i^{1*}(k)) \\ \chi_i^1(k) &= x_i^1(k). \end{aligned} \tag{23}$$

The HONN proposed for this application is as follows:

$$\begin{aligned} \alpha_i^{1*}(k) &= w_i^{1\top} S_i^1(z_i^1(k)) \\ u_i(k) &= w_i^{2\top} S_i^2(z_i^2(k)) \end{aligned} \tag{24}$$

with

$$\begin{aligned} z_i^1(k) &= [x_i^1(k), x_{i\mathrm{d}}^1(k+2)] \\ z_i^2(k) &= [x_i^1(k), x_i^2(k), \alpha_i^1(k)]. \end{aligned} \tag{25}$$

The weights are updated using the EKF (13) - (19) with $i = 1,2$ and

$$\begin{aligned} e_i^1(k) &= x_{i\mathrm{d}}^1(k) - \chi_i^1(k) \\ e_i^2(k) &= x_i^2(k) - \alpha_i^1(k). \end{aligned} \tag{26}$$

The training is performed on-line using a series-parallel configuration. All the neural network states are initialized in a random way.

6 SIMULATION RESULTS

For simulation, we select the following discrete-time trajectories (Ramirez, 2008)

$$\begin{aligned} x_{1\mathrm{d}}^1(k) &= c_1(1 - e^{d_1 kT^3})\sin(\omega_1 kT)[\mathrm{rad}] \\ x_{2\mathrm{d}}^1(k) &= c_2(1 - e^{d_2 kT^3})\sin(\omega_2 kT)[\mathrm{rad}] \\ x_{3\mathrm{d}}^1(k) &= c_3(1 - e^{d_3 kT^3})\sin(\omega_3 kT)[\mathrm{rad}] \\ x_{4\mathrm{d}}^1(k) &= c_4(1 - e^{d_4 kT^3})\sin(\omega_4 kT)[\mathrm{rad}] \\ x_{5\mathrm{d}}^1(k) &= c_5(1 - e^{d_5 kT^3})\sin(\omega_5 kT)[\mathrm{rad}] \\ x_{6\mathrm{d}}^1(k) &= c_6(1 - e^{d_6 kT^3})\sin(\omega_6 kT)[\mathrm{rad}] \\ x_{7\mathrm{d}}^1(k) &= c_7(1 - e^{d_7 kT^3})\sin(\omega_7 kT)[\mathrm{rad}] \end{aligned} \tag{27}$$

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

Table 1: Parameters for desired trajectories.

i-th Joint	c_i	d_i	ω_i
1	$\pi/2$	0.001	0.285 rad/s
2	$\pi/3$	0.001	0.435 rad/s
3	$\pi/2$	0.01	0.555 rad/s
4	$\pi/3$	0.01	0.645 rad/s
5	$\pi/2$	0.01	0.345 rad/s
6	$\pi/3$	0.01	0.615 rad/s
7	$\pi/2$	0.01	0.465 rad/s

Table 2: Maximum torques.

Joint	Max Torque
1	232 N-m
2	232 N-m
3	100 N-m
4	100 N-m
5	14.5 N-m
6	14.5 N-m
7	14.5 N-m

the selected parameters c, d and ω for desired trajectories of each joint are shown in Table 1. The sampling time is selected as $T = 1$ millisecond.

These selected trajectories (27) incorporate a sinusoidal term to evaluate the performance in presence of relatively fast periodic signals, for which the nonlinearities of the robot dynamics are really important.

Simulation results for trajectory tracking using the decentralized neural backstepping control (DNBS) scheme are shown in Figs. 4 to Fig. 10. The initial conditions for the plant are different that those of the desired trajectory. According to these figures, the tracking errors for all joints present a good behavior and remain bounded as shown in Fig. 11.

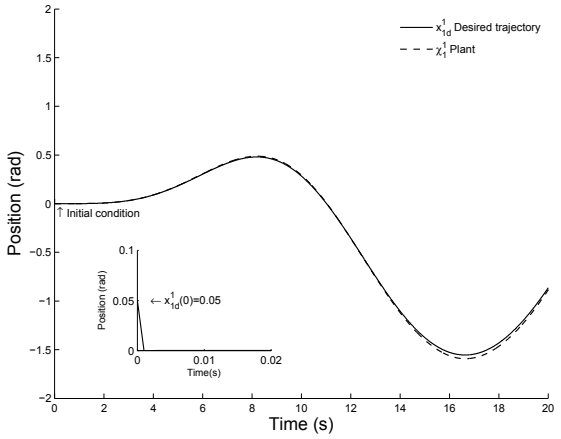

Figure 4: Trajectory tracking for joint 1 $x^1_{1\mathrm{d}}(k)$ (solid line) and $\chi^1_1(k)$ (dashed line).

The applied torques to each joint are always inside of the prescribed limits given by the actuators manufacturer (see Table 2); that is, their absolute values are smaller than the bounds $\tau_1^{\max}$ to $\tau_7^{\max}$, respectively.

7 CONCLUSIONS

In this paper a decentralized neural control scheme based on the backstepping technique is presented. The control law for each joint is approximated by a high order neural network. The training of each neural network is performed on-line using an extended Kalman filter. Simulations results for trajectory tracking using a seven DOF PA10-7CE Mitsubishi robot arm show the effectiveness of the proposed control scheme.

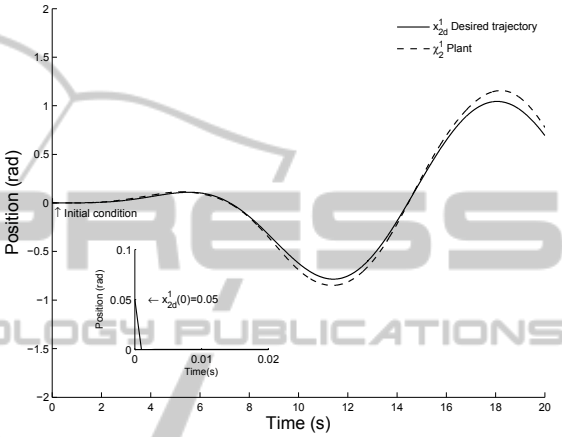

Figure 5: Trajectory tracking for joint 2 $x^1_{2\mathrm{d}}(k)$ (solid line) and $\chi^1_2(k)$ (dashed line).

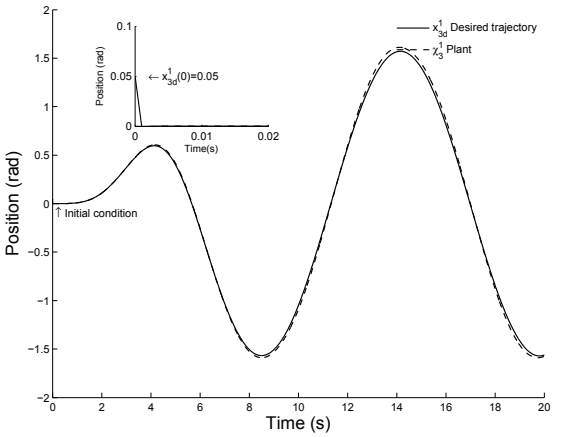

Figure 6: Trajectory tracking for joint 3 $x^1_{3\mathrm{d}}(k)$ (solid line) and $\chi^1_3(k)$ (dashed line).

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

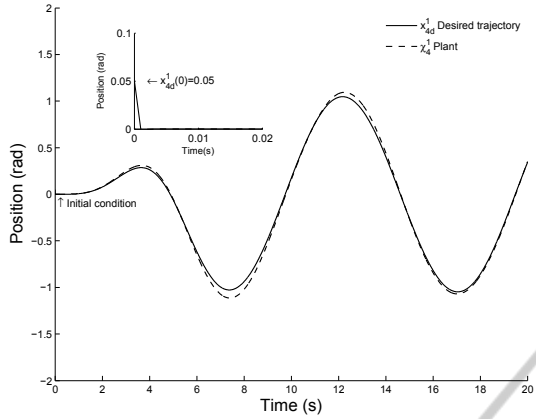

Figure 7: Trajectory tracking for joint 4 $x^1_{4d}(k)$ (solid line) and $\chi^1_4(k)$ (dashed line).

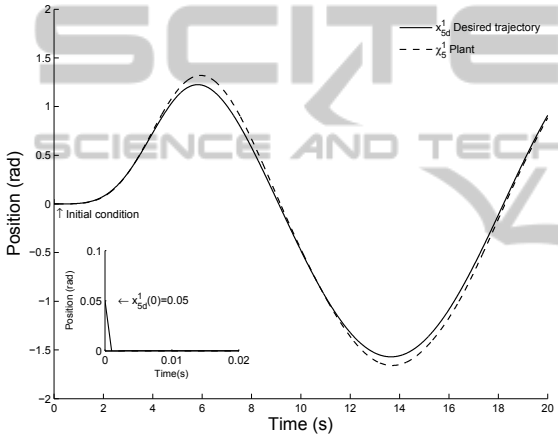

Figure 8: Trajectory tracking for joint 5 $x^1_{5d}(k)$ (solid line) and $\chi^1_5(k)$ (dashed line).

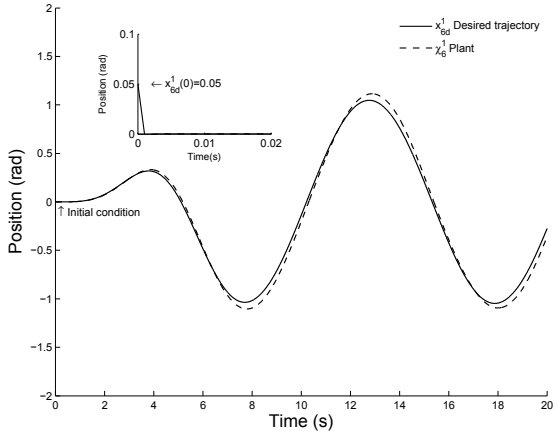

Figure 9: Trajectory tracking for joint 6 $x^1_{6d}(k)$ (solid line) and $\chi^1_6(k)$ (dashed line).

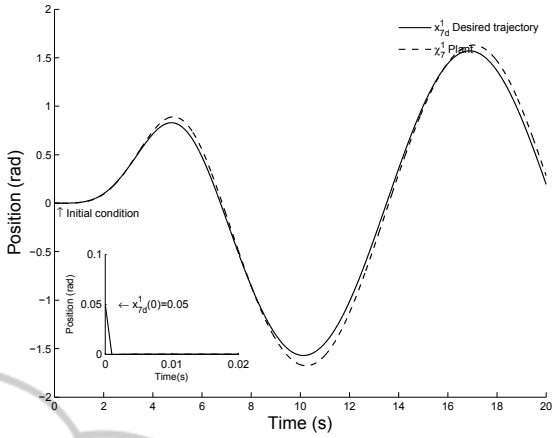

Figure 10: Trajectory tracking for joint 7 $x^1_{7d}(k)$ (solid line) and $\chi^1_7(k)$ (dashed line).

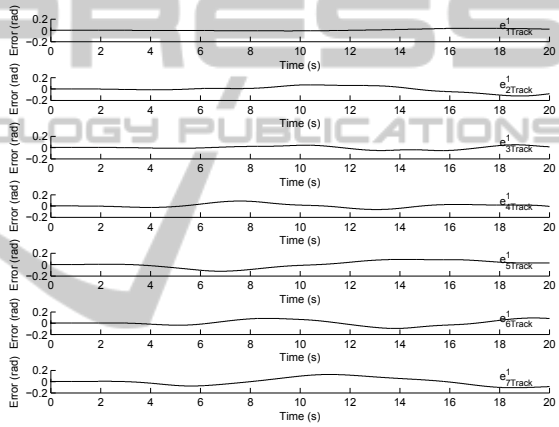

Figure 11: Tracking errors for joints 1 to 7.

ACKNOWLEDGEMENTS

The first author thanks to Universidad Autonoma del Carmen (UNACAR) and the Programa de Mejoramiento del Profesorado (PROMEP-MEXICO) for supporting this research.

REFERENCES

Alanis, A. Y., Sanchez, E. N., and Loukianov, A. G. (2007). Discrete-time adaptive backstepping nonlinear control via high-order neural networks. *IEEE Transactions on Neural Networks*, 18(4):1185–1195.

Ge, S. S., Zhang, J., and Lee, T. H. (2004). Adaptive neural network control for a class of MIMO nonlinear systems with disturbances in discrete-time. *IEEE Transactions on Systems, Man, and Cybernetics Part B*, 34(4):1630–1645.

Higuchi, M., Kawamura, T., Kaikogi, T., Murata, T., and

Kawaguchi, M. (2003). Mitsubihi clean room robot. Tecnical review, Mitsubishi Heavy Industries, Ltd.

Huang, S., Tan, K. K., and Lee, T. H. (2003). Decentralized control design for large-scale systems with strong interconnections using neural networks. *IEEE Transactions on Automatic Control*, 48(5):805–810.

Jamisola, R. S., Maciejewski, A. A., and Roberts, R. G. (2004). Failure-tolerant path planning for the PA-10 robot operating amongst obstacles. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 4995–5000, New Orleans, LA, USA.

Karakasoglu, A., Sudharsanan, S. I., and Sundareshan, M. K. (1993). Identification and decentralized adaptive control using dynamical neural networks with application to robotic manipulators. *IEEE Transactions on Neural Networks*, 4(6):919–930.

Kennedy, C. W. and Desai, J. P. (2003). Force feedback using vision. In *Proceedings of IEEE International Conference on Advanced Robotics*, Coímbra, Portugal.

Krstic, M., Kanellakopoulos, I., and Kokotovic, P. (1995). *Nonlinear and Adaptive Control Design*. John Wiley & Sons, Inc, New York, USA.

Liu, M. (1999). Decentralized control of robot manipulators: nonlinear and adaptive approaches. *IEEE Transactions on Automatic Control*, 44(2):357–363.

Ni, M. L. and Er, M. J. (2000). Decentralized control of robot manipulators with coupling and uncertainties. In *Proceedings of the American Control Conference*, pages 3326–3330, Chicago, Illinois, USA.

Ramirez, C. (2008). Dynamic modeling and torque-mode control of the Mitsubishi PA10-7CE robot. Master dissertation (in spanish), Instituto Tecnológico de la Laguna, Torreón, Coahuila, Mexico.

Rovithakis, G. A. and Christodoulou, M. A. (2000). *Adaptive Control with Recurrent High-Order Neural Networks*. Springer, London, U.K.

Safaric, R. and Rodic, J. (2000). Decentralized neural-network sliding-mode robot controller. In *Proceedings of 26th Annual Conference on the IEEE Industrial Electronics Society*, pages 906–911, Nagoya, Aichi, Japan.

Sanchez, E. N. and Ricalde, L. J. (2003). Trajectory tracking via adaptive recurrent neural control with input saturation. In *Proceedings of International Joint Conference on Neural Networks*, pages 359–364, Portland, Oregon, USA.

Santibañez, V., Kelly, R., and Llama, M. A. (2005). A novel global asymptotic stable set-point fuzzy controller with bounded torques for robot manipulators. *IEEE Transactions on Fuzzy Systems*, 13(3):362–372.

Song, Y. and Grizzle, J. W. (1995). The extended Kalman filter as local asymptotic observer for discrete-time nonlinear systems. *Journal of Mathematical Systems, Estimation and Control*, 5(1):59–78.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS