Hindawi Publishing Corporation
ISRN Robotics
Volume 2013, Article ID 173703, 11 pages
http://dx.doi.org/10.5402/2013/173703

Research Article

Comparative Study between Robust Control of Robotic Manipulators by Static and Dynamic Neural Networks

Nadya Ghrab[1] and Hichem Kallel[2]

[1] *National Institute of Applied Science and Technology (INSAT), Northern Urban Center Mailbox 676, 1080 Tunis, Tunisia*
[2] *Department of Physics and Electrical Engineering, National Institute of Applied Science and Technology (INSAT), Tunisia*

Correspondence should be addressed to Hichem Kallel; golden.k@gnet.tn

Received 28 January 2013; Accepted 20 March 2013

Academic Editors: A. Bechar, A. Sabanovic, R. Safaric, K. Terashima, and C.-C. Tsai

A comparative study between static and dynamic neural networks for robotic systems control is considered. So, two approaches of neural robot control were selected, exposed, and compared. One uses a static neural network; the other uses a dynamic neural network. Both compensate the nonlinear modeling and uncertainties of robotic systems. The first approach is direct; it approximates the nonlinearities and uncertainties by a static neural network. The second approach is indirect; it uses a dynamic neural network for the identification of the robot state. The neural network weight tuning algorithms, for the two approaches, are developed based on Lyapunov theory. Simulation results show that the system response, equipped by dynamic neural network controller, has better tracking performance, has faster response time, and is more reliable to face disturbances and robotic uncertainties.

1. Introduction

Several orders of neural robot control approaches have been proposed in the literature. These approaches are classified into two main classes: direct and indirect neural controls. If it requires prior identification of the controlled process model, it is called indirect control; otherwise it is called direct control. For the direct one, many architectures of control are mentioned in the literature [1–5]. For the second class, we cite neural control via dynamic neural network [6, 7], Model Reference Adaptive Control (MRAC) [8–10], Internal Model Control (IMC) [11–13], and predictive neural control [14, 15]. Both of these control classes are robust thanks to their ability to overcome the nonlinearities and uncertainties in the robot dynamics.

In this paper, the aim is to compare the performance of static neural networks to dynamic neural networks in robotic systems control. For this, two types of control, from the already mentioned, are selected, presented, and tested for a two-link robot. One uses a static neural network; the other uses a dynamic neural network. The first approach is a direct neural control for improvement of a classic controller proportional derivative (PD), proposed by Lewis [1]; it manages to approximate the nonlinearities and uncertainties in the robot dynamics by a static neural network. The second approach is an indirect neural control via a high-order dynamic neural network, proposed by Sanchez et al. [7], which manages to use a dynamic neural network for a dynamic identification of the robot state. Based on simulation results, a comparative study between these two approaches is presented using different performance criteria.

The rest of this paper is organized as follows. Section 2 presents the dynamic model of the robot manipulator. Section 3 describes the direct neural control proposed by Lewis [1]. Section 4 describes the indirect neural control proposed by Sanchez et al. [7]. Section 5 is intended for the simulation results, and a comparative study between the two approaches is mentioned in Sections 3 and 4. And finally, Section 6 draws conclusion and sums up the whole paper.

2. Dynamic Model of the Robot Manipulator

In this section, the dynamic model of the robot manipulator is presented. The equation of the robot dynamics is

$$J(\theta)\ddot{\theta} + h(\theta,\dot{\theta})\dot{\theta} + G(\theta) + F(\dot{\theta}) + \tau_d = \tau. \tag{1}$$

$\theta, \dot{\theta}, \ddot{\theta} \in \mathbb{R}^n$ denote the joint angle, the joint velocity, and the joint acceleration; $J(\theta) \in \mathbb{R}^{n\times n}$ denote the inertia matrix; $h(\theta, \dot{\theta}) \in \mathbb{R}^{n\times n}$ denote the Centrifugal and Coriolis force matrix; $G(\theta) \in \mathbb{R}^n$ denote the gravitational force vector; $F(\dot{\theta}) \in \mathbb{R}^n$ the friction term such as $F(\dot{\theta}) = F_v\dot{\theta} + F_c(\dot{\theta})$ where $F_c(\dot{\theta}) \in \mathbb{R}^n$ is the coulomb parameter; $F_v \in \mathbb{R}^{n\times n}$ the viscous parameter; $\tau_d(t) \in \mathbb{R}^n$ represents disturbances; and $\tau(t) \in \mathbb{R}^n$ is the torque vector.

3. Direct Neural Controller via Static Neural Network

In this section, the approach of direct neural control for improvement of a classic controller proportional-derivative (PD), proposed by Lewis [1], is briefly presented. This approach manages to approximate the nonlinearities and uncertainties, in the robot dynamics, by a static neural network.

To make the dynamic of the robot manipulator, defined in (1), follow a prescribed desired trajectory $\theta_d(t) \in \mathbb{R}^n$; the tracking error $e(t)$ and the filtered tracking error $r(t)$ are defined as follows:

$$e = \theta_d - \theta, \tag{2}$$

$$r = \dot{e} + \Lambda e. \tag{3}$$

$\Lambda > 0$ is a symmetric positive definite design parameter matrix. The dynamic of the robot (1), in terms of the filtered error (3), is as follows:

$$J(\theta)\dot{r}(t) = -h(\theta, \dot{\theta}) r(t) - \tau(t) + f(x) + \tau_d(t), \tag{4}$$

where the unknown nonlinear robot function is defined as

$$\begin{aligned} f(x) = {} & J(\theta)\left(\ddot{\theta}_d + \Lambda\dot{e}\right) + h(\theta, \dot{\theta})\left(\dot{\theta}_d + \Lambda e\right) \\ & + G(\theta) + F(\dot{\theta}), \end{aligned} \tag{5}$$

with

$$x = \left[e^T \ \ \dot{e}^T \ \ \theta_d{}^T \ \ \dot{\theta}_d{}^T \ \ \ddot{\theta}_d{}^T\right]^T. \tag{6}$$

3.1. Approximation of Nonlinearities and Uncertainties by a Static Neural Network. *The universal Function Approximation Property* [16]. Let $f(x)$ be a general smooth function from $\mathbb{R}^n$ to $\mathbb{R}^m$. Then, it can be shown that, as long as x is restricted to a compact set S of $\mathbb{R}^n$, there exist weights and thresholds such that one has

$$f(x) = W^T\sigma\left(M^T x\right) + \varepsilon. \tag{7}$$

It is difficult to determine the ideal neural network weights, in matrices W and M that are required to best approximate a given nonlinear function $f(x)$. However, all one needs to know for controls purposes that, for a specified value of Neural Network, some ideal approximating weights exist. Then, an estimate of $f(x)$ can be given by

$$\hat{f}(x) = \widehat{W}^T\sigma\left(\widehat{M}^T x\right). \tag{8}$$

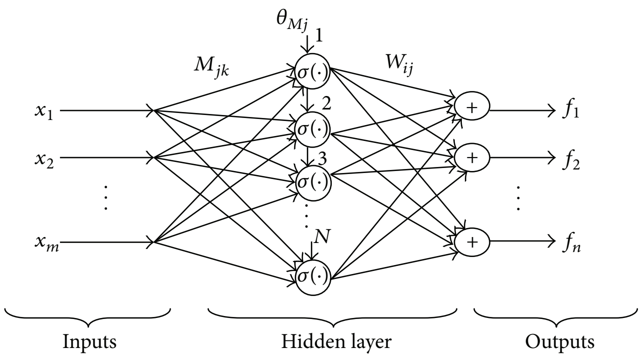


FIGURE 1: The static feed forward neural network architecture.

The neural network architecture, proposed for the approximation of nonlinearities and uncertainties in the robot dynamics, is shown in Figure 1, where $\sigma(\cdot) : \mathbb{R} \to \mathbb{R}$ is the activation functions and N is the number of hidden-layer neurons. The first-layer interconnection weights are denoted by M_{jk} and the second-layer interconnection weights by W_{ij}. The threshold offsets are denoted by θ_{Mj}.

3.2. Synthesis of the Control Law. A general sort of approximation-based controller is derived by setting:

$$\tau(t) = \hat{f}(x) + k_v r(t) - v(t) \tag{9}$$

with $\hat{f}(x)$ being the approximation of $f(x)$ by the neural network, k_v. $r(t)$ an outer PD tracking loop, and $v(t)$ an auxiliary signal to provide robustness. The proposed neural network control structure is shown in Figure 2.

Substituting the control law (9) in (4), the closed-loop error dynamics become as follows:

$$J(\theta)\dot{r}(t) = -\left(k_v + h(\theta, \dot{\theta})\right) r(t) + \tilde{f}(x) + \tau_d(t) + v(t). \tag{10}$$

Let us define the functional approximation error are

$$\tilde{f} = f - \hat{f}. \tag{11}$$

The weight approximation errors

$$\widetilde{W} = W - \widehat{W}, \qquad \widetilde{M} = M - \widehat{M}. \tag{12}$$

The Lyapunov function proposed for the stabilization of the error dynamic is

$$\begin{aligned} V\left(r, \widetilde{W}, \widetilde{M}\right) = {} & \frac{1}{2}r^T J(\theta) r + \frac{1}{2}\operatorname{tr}\left\{\widetilde{W}^T F^{-1}\widetilde{W}\right\} \\ & + \frac{1}{2}\operatorname{tr}\left\{\widetilde{M}^T G^{-1}\widetilde{M}\right\}, \end{aligned} \tag{13}$$

with any constant matrices being $F = F^T > 0$ and $G = G^T > 0$.

The robotic system is asymptotically stable if the following conditions, which guarantee $\dot{V}(r, \widetilde{W}, \widetilde{M}) < 0$, are satisfied.

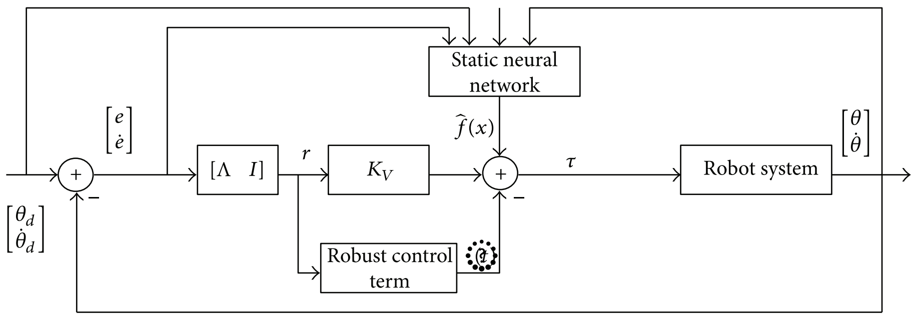

Static neural network
$\hat{f}(x)$
$\begin{bmatrix} e \\ \dot{e} \end{bmatrix}$
$\begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$
$[\Lambda \quad I]$
r
K_V
τ
Robot system
$\begin{bmatrix} \theta_d \\ \dot{\theta}_d \end{bmatrix}$
Robust control term
v
+
−

FIGURE 2: Direct neural controller via static neural network.

(i) The neural network weight tuning algorithms are

$$\dot{\widehat{W}} = F\widehat{\sigma}r^T - F\widehat{\sigma}'\widehat{M}^T xr^T - kF\|r\|\widehat{W},$$
$$\dot{\widehat{M}} = Gx\left(\widehat{\sigma}'^T\widehat{W}r\right)^T - kG\|r\|\widehat{M}. \quad (14)$$

With $k > 0$ being a small scalar design parameter and σ' the Jacobian of σ.

(ii) The robustifying term is

$$v(t) = -k_Z\left(Z_B + \left\|\widehat{Z}\right\|_F\right)r. \quad (15)$$

With $Z = \left[\begin{smallmatrix} W & 0 \\ 0 & M \end{smallmatrix}\right]$, $\|Z\|_F < Z_B$ such that Z_B is the bound of ideal weights, $k_Z > 0$ is positive scalar parameter.

(iii) PD controller gain is

$$k_{V\min} > \frac{C_0 + k\left(C_3^2/4\right)}{\|r\|}. \quad (16)$$

In practice, the tracking error can be kept as small as desired by increasing the gain k_v.

4. Indirect Neural Control via High-Order Dynamic Neural Network

In this section, the second approach, an indirect neural control via a high-order dynamic neural network proposed by Sanchez et al. [7], is briefly presented. This approach manages to use a dynamic neural network for the online identification of the robot state.

The proposed neural network control structure is shown in Figure 3.

The equation of the robot dynamics, defined in (1), under state representation is

$$\dot{X} = f(x) + g(X)\tau(t), \quad (17)$$

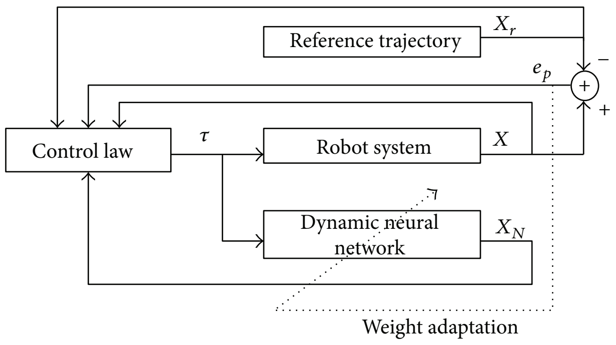


FIGURE 3: Indirect neural control via a high-order dynamic neural network.

with

$$f(X) = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix}$$
$$+ \begin{bmatrix} 0 \\ -J(\theta)^{-1}\left(h\left(\theta,\dot{\theta}\right)\dot{\theta} + G(\theta) + F\left(\dot{\theta}\right) + \tau_d(t)\right) \end{bmatrix},$$
$$g(X) = \begin{bmatrix} 0 \\ J(\theta)^{-1} \end{bmatrix},$$
$$X = \begin{bmatrix} \theta & \dot{\theta} \end{bmatrix}^T. \quad (18)$$

4.1. Identification of the Robot State by the High-Order Dynamic Neural Network. The proposed neural network structure, for the identification of the robot state, is shown in Figure 4.

$A_N = -\lambda_i I_{2n\times 2n}$ is the state matrix of the neural network with $\lambda_i > 0$ for $i = 1,\ldots,2n$, $X_N = [x_{N1} \ \cdots \ x_{N2n}] \in \mathbb{R}^{2n}$ is the state vector of the neural network, and $\tau(t) \in \mathbb{R}^n$ is the torque vector.

The dynamics of this neural network are resulted by the state feedback X_N around a neural structure formed by two static neural networks RN1 and RN2 shown in Figure 5 [7, 17]:

$$\text{RN1}\left(X_N\right) = W_f Z\left(X_N\right). \quad (19)$$

$W_f = [w_{f1} \ \cdots \ w_{f2n}]^T \in \mathbb{R}^{2n\times L}$ is weights matrix of RN1.

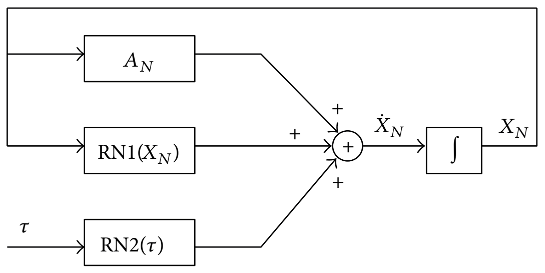


Figure 4: The high-order dynamic neural network architecture.

$Z(X_N) = [z_1 \ \cdots \ z_L]^T \in \mathbb{R}^L$ is nonlinear operator which defines the high-order connections, z_i are called high-order connections, and L is the number of high order connections.

$z_i = \prod_{j \in I_i} y_j^{d_j^{(i)}}$, $j = 1, \ldots, 2n$; $i = 1, \ldots, L$ and $d_j^{(i)}$ are positive integers, $y_j = \varphi(x_{Ni})$, $i = 1, \ldots, 2n$, $\varphi(\cdot) : \mathbb{R} \to \mathbb{R}$ is the activation function of RN1, here selected as the hyperbolic tangent.

The outputs of RN1 are denoted by $C = [c_1 \ \cdots \ c_{2n}]^T$

$$\text{RN2}(\tau) = W_g \phi(\tau(t)) = W_g \tau(t). \qquad (20)$$

$\phi(\cdot) : \mathbb{R} \to \mathbb{R}$ is the activation function of RN2, here selected as the linear one.

$W_g = [w_{g1} \ \cdots \ w_{g2n}]^T \in \mathbb{R}^{2n \times n}$ is weights matrix of RN2. The outputs of RN2 are denoted by $D = [d_1 \ \cdots \ d_{2n}]^T$. Let us denote $\widehat{W}_f$ and $\widehat{W}_g$ to be the estimated value, respectively, for the unknown weight matrices W_f and W_g.

The weight estimation errors are

$$\widetilde{W}_f = W_f - \widehat{W}_f, \qquad \widetilde{W}_g = W_g - \widehat{W}_g. \qquad (21)$$

The equation of this neural network dynamics is defined by

$$\dot{X}_N = A_N X_N + W_f Z(X_N) + W_g \tau(t). \qquad (22)$$

The identification of the robot dynamics by the neural network is ensured by the following pole placement:

$$\dot{X}_N - \dot{X} = -(X_N - X). \qquad (23)$$

Equation (23) is equivalent to the following equation:

$$\dot{X} = A_N X_N + W_f Z(X_N) + W_g \tau(t) + (X_N - X). \qquad (24)$$

4.2. Synthesis of the Control Law. It is desired to design a robust controller which enforces asymptotic stability of the tracking error between the system and the reference signal. The equation of the reference signal dynamic is

$$\dot{X}_r = f_r(X_r, \tau_r(t)). \qquad (25)$$

X_r is the reference signal state vector, $\tau_r(t)$ is the desired torque vector, and $f_r(\cdot)$ is the vector field for reference dynamics.

Let us denote the tracking error between the system and the reference signal to be

$$e_p = X - X_r. \qquad (26)$$

To ensure the desired dynamic, the asymptotic stability of the tracking error must be ensured.

The time derivative of (26) is

$$\dot{e}_p = A_N X_N + W_f Z(X_N) + W_g \tau(t) + (X_N - X) - \dot{X}_r. \qquad (27)$$

Now, it is proceeded to add and subtract in (27) the terms $\widehat{W}_f Z(X_r)$, $A_N e_p$, $A_N X_r$, and X_r so that

$$\begin{aligned} \dot{e}_p = {} & A_N e_p + W_f Z(X_N) + W_g \tau(t) \\ & + \left(-\dot{X}_r + A_N X_r + \widehat{W}_f Z(X_r) + X_r - X\right) \\ & - A_N e_p - \widehat{W}_f Z(X_r) - A_N X_r - X_r + X_N + A_N X_N. \end{aligned} \qquad (28)$$

It is assumed that there exists a function $\alpha_r(t, \widehat{W}_f, \widehat{W}_g)$ such that:

$$\begin{aligned} & \alpha_r\left(t, \widehat{W}_f, \widehat{W}_g\right) \\ & \quad = -\left(\widehat{W}_g\right)^+ \left(-\dot{X}_r + A_N X_r + \widehat{W}_f Z(X_r) + X_r - X\right). \end{aligned} \qquad (29)$$

$(\widehat{W}_g)^+$ is the pseudo inverse of $\widehat{W}_g$ calculated as follows: $(\widehat{W}_g)^+ = (\widehat{W}_g^T \widehat{W}_g)^{-1} \widehat{W}_g^T$.

Then, it is proceeded to add and subtract the term $\widehat{W}_g \alpha_r(t, \widehat{W}_f, \widehat{W}_g)$ in (28) so that we obtain:

$$\begin{aligned} \dot{e}_p = {} & A_N e_p + W_f Z(X_N) + W_g \tau(t) - \widehat{W}_g \alpha_r\left(t, \widehat{W}_f, \widehat{W}_g\right) \\ & - A_N (X - X_r) - \widehat{W}_f Z(X_r) + (A_N + I)(X_N - X_r). \end{aligned} \qquad (30)$$

Let us define

$$\widetilde{\tau} = \tau - \alpha_r\left(t, \widehat{W}_f, \widehat{W}_g\right) = \tau_1 + \tau_2, \qquad (31)$$

so that (30) is reduced to

$$\begin{aligned} \dot{e}_p = {} & A_N e_p + \widetilde{W}_f Z(X_N) + \widehat{W}_f (Z(X_N) - Z(X_r)) \\ & + \widetilde{W}_g \tau(t) + \widehat{W}_g \widetilde{\tau}(t) - A_N (X - X_r) \\ & + (A_N + I)(X_N - X_r). \end{aligned} \qquad (32)$$

Then, it is proceeded to add and subtract the term $Z(X)$ and X in (32) so that we obtain:

$$\begin{aligned} \dot{e}_p = {} & A_N e_p + \widetilde{W}_f Z(X_N) \\ & + \widehat{W}_f (Z(X_N) - Z(X) + Z(X) - Z(X_r)) \\ & + \widetilde{W}_g \tau(t) + \widehat{W}_g \widetilde{\tau}(t) - A_N (X - X_r) \\ & + (A_N + I)(X_N - X + X - X_r). \end{aligned} \qquad (33)$$

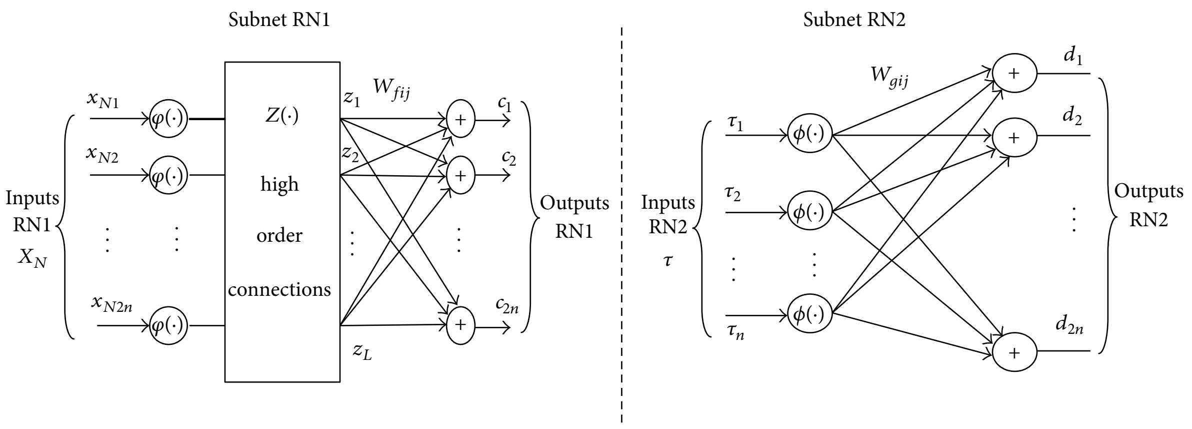


Figure 5: The two subnets RN1 and RN2 included in the dynamic neural network.

Then by defining

$$\tau_1 = \left(\widehat{W}_g\right)^+ \times \left(-\widehat{W}_f\left(Z\left(X_N\right) - Z(X)\right) - \left(A_N + I\right)\left(X_N - X\right)\right). \quad (34)$$

Equation (33) is reduced to

$$\dot{e}_p = \left(A_N + I\right) e_p + \widetilde{W}_f Z\left(X_N\right) + \widehat{W}_f\left(Z(X) - Z\left(X_r\right)\right) + \widetilde{W}_g \tau(t) + \widehat{W}_g \tau_2(t). \quad (35)$$

Then, the tracking problem is reduced to a stabilization problem of the error dynamics defined in (35).

The Control Lyapunov Function (CLF) proposed for the stabilization of the error dynamics is

$$V\left(e_p, \widetilde{W}_f, \widetilde{W}_g\right) = \frac{1}{2}\left\|e_p\right\|^2 + \frac{1}{2}\operatorname{tr}\left\{\widetilde{W}_f^{\;T}\Gamma^{-1}\widetilde{W}_f\right\} + \frac{1}{2}\operatorname{tr}\left\{\widetilde{W}_g^{\;T}\Gamma_g^{-1}\widetilde{W}_g\right\}. \quad (36)$$

$\Gamma = \operatorname{diag}\{\gamma_1, \ldots, \gamma_L\}$ and $\Gamma_g = \operatorname{diag}\{\gamma_{g1}, \ldots, \gamma_{gn}\}$ are symmetric positive definite diagonal matrices.

Let us define the function $\phi_Z = Z(X) - Z(X_r)$ and L_{ϕ_Z} to be its Lipschitz constant.

The robotic system is asymptotically stable if the following conditions, which guarantee $\dot{V}(e_p, \widetilde{W}_f, \widetilde{W}_g) < 0$, are satisfied.

(i) The control law τ_2 is

$$\tau_2 = -\mu\left(\widehat{W}_g\right)^+ \left(1 + L_{\phi_Z}^2 \left\|\widehat{W}_f\right\|^2\right) e_p. \quad (37)$$

Optimal with respect to the following cost [18]:

$$J(\widetilde{\tau}) = \lim_{t\to\infty}\left\{2\beta V + \int_0^t \left(l\left(e_p, \widetilde{W}_f, \widetilde{W}_g\right) + \tau_2^T R\left(e_p, \widetilde{W}_f, \widetilde{W}_g\right)\tau_2\right) dt\right\}. \quad (38)$$

(ii) The neural network weight tuning algorithms are

$$\dot{\widehat{w}}_{fij} = -e_{pi}\gamma_j Z\left(X_j\right), \text{ RN1 weights tuning algorithm,}$$
$$\dot{\widehat{w}}_{gij} = -e_{pi}\gamma_{gj}\,\tau_j(t), \text{ RN2 weights tuning algorithm.} \quad (39)$$

(iii) The parameter of the neural network state matrix is $\lambda_i > 1$.

(iv) The parameter which manages the control law τ_2 is $\mu > 1$.

5. Simulation Results and Comparative Study on a Two-Link Robot

In order to test the applicability and compare the performance of the two proposed neural control types, the trajectory tracking problem for a robot manipulators model is considered. The dynamics of a 2-link rigid robot arm on 2D environment, with friction and disturbance terms, can be written as

$$J(\theta)\ddot{\theta} + H\left(\theta, \dot{\theta}\right) + G(\theta) + F\left(\dot{\theta}\right) + \tau_d(t) = D\tau(t), \quad (40)$$

with

$$J(\theta) = \begin{bmatrix} I_1 + m_1 k_1^2 + m_2 l_1^2 & m_2 l_1 k_2 \cos\left(\theta_1 - \theta_2\right) \\ m_2 l_1 k_2 \cos\left(\theta_1 - \theta_2\right) & I_2 + m_2 k_2^2 \end{bmatrix},$$

$$H\left(\theta, \dot{\theta}\right) = h\left(\theta, \dot{\theta}\right)\dot{\theta} = \begin{bmatrix} 0 & m_2 l_1 k_2 \sin\left(\theta_1 - \theta_2\right) \\ -m_2 l_1 k_2 \sin\left(\theta_1 - \theta_2\right) & 0 \end{bmatrix}\begin{bmatrix} \dot{\theta}_1^2 \\ \dot{\theta}_2^2 \end{bmatrix},$$

$$G(\theta) = g\begin{bmatrix} \left(m_2 l_1 + m_1 k_1\right)\cos\theta_1 \\ m_2 k_2 \cos\theta_2 \end{bmatrix},$$

$$D = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix},$$

$$F\left(\dot{\theta}\right) = \operatorname{diag}(2,2)\dot{\theta} + 1.5\operatorname{sign}\left(\dot{\theta}\right). \quad (41)$$

The robot model parameters are shown in Table 1.

The simulations of the variation of positions and torques, exerted at each of the two joints, as well as the weights of the neural network, were carried out over a period of 10 seconds.

The initial conditions are selected as follows:

$$\theta(0) = [20 \ \ 20]^T, \qquad \dot{\theta}(0) = [0 \ \ 0]^T. \tag{42}$$

The reference signal is

$$\theta_d(t) = [45 \ \ 45]^T, \qquad \dot{\theta}_d(t) = [0 \ \ 0]^T. \tag{43}$$

The external disturbances are

$$\begin{aligned} &[0.1 \ \ -0.1] \quad \text{if } t \le 1\text{sec}, \\ &[-0.2 \ \ 0.2] \quad \text{if } t > 1\text{sec}. \end{aligned} \tag{44}$$

A variation in the coefficients of viscous friction by an error of 10% at time $t = 1$ sec.

A variation in the masses of the two bodies of the arm by an error of 10% at time $t = 2$ sec.

The neural network controller parameters are selected, for each of the two approaches, as follows.

(i) For the first approach: neural control for improvement of a classic controller proportional-derivative (PD), we have the following.

After several simulation tests, we have found suitable values for the initialization of the weights of the neural network and the various parameters as follows:

$$\begin{gathered} \mathbf{K_v} = 10 \times I_{2\times 2}, \qquad \mathbf{K_z} = 0.1, \\ \mathbf{Z_B} = 5, \qquad \mathbf{F} = 10 \times I_{6\times 6}, \\ \mathbf{G} = 10 \times I_{10\times 10}, \qquad \mathbf{K} = 0.1. \end{gathered} \tag{45}$$

The number of neurons in the hidden layer of neural network is $\mathbf{N} = 6$. The activation function sigmoid is $\sigma(x) = 1/(1+e^{-x})$ and its Jacobian is $\sigma'(x) = \operatorname{diag}\{\sigma(x)\} \times [I - \operatorname{diag}\{\sigma(x)\}]$. No initial neural network training phase was needed. The neural network weights were arbitrarily initialized at zero in this simulation.

(ii) For the second approach: neural control via a high-order dynamic neural network, we have the following.

Initial state vector of the neural network is $X_N = [0 \ \cdots \ 0]^T$, and the number of high-order connections is $\mathbf{L} = 13$.

The activation function of the subnet RN1 is the hyperbolic tangent $\varphi(X_N) = (e^{2X_N} - 1)/(e^{2X_N} + 1)$.

Let us have: $X_N = [x_{N1} \ \cdots \ x_{N4}]^T$

$$\begin{gathered} N1 = \operatorname{tangh}(x_{N1}), \qquad N2 = \operatorname{tangh}(x_{N2}), \\ N3 = \operatorname{tangh}(x_{N3}), \qquad N4 = \operatorname{tangh}(x_{N4}). \end{gathered} \tag{46}$$

So that we define the high-order connections of the neural networks:

$$\begin{aligned} Z(X_N) = [&N1, N2, N3, N4, N1 \times N2, N1 \times N3, N1 \times N4, \\ &N2 \times N3, N2 \times N4, N3 \times N4, N1 \times N2 \times N3, \\ &N1 \times N2 \times N4, N2 \times N3 \times N4]^T. \end{aligned} \tag{47}$$

Table 1: Parameters of 2-link rigid robot model.

Parameters	Designation	Value	Unit
Length of link 1	l_1	0.25	m
Length of link 2	l_2	0.16	m
Mass of link 1	m_1	9.5	Kg
Mass of link 2	m_2	5	Kg
Position of center of gravity of link 1	k_1	0.125	m
Position of center of gravity of link 2	k_2	0.08	m
Inertia of link 1	I_1	0.0043	Kg·m^2
Inertia of link 2	I_2	0.0061	Kg·m^2
Gravitational acceleration	g	9.8	N·Kg^{-1}

The activation function of the subnet RN2 is the linear function $\phi(\tau(t)) = \tau(t)$.

For this approach, the initialization of neural network weights is not arbitrary and a training phase is necessary.

After several simulation tests, we have found suitable values for the initialization of the weights of the neural network and the various parameters as follows:

$$\begin{gathered} \mathbf{\Gamma} = 0.01 \times I_{13\times 13}, \qquad \mathbf{\Gamma_g} = 0.0001 \times I_{2\times 2}, \\ \boldsymbol{\lambda}_\mathbf{i}=300, \qquad \mathbf{A_N} = -300 \ I_{4\times 4}, \\ \mathbf{L}_{\phi_\mathbf{z}} = 0.02, \qquad \boldsymbol{\mu} = 1000. \end{gathered} \tag{48}$$

The suitable values for the initialization of the weights are shown in Figure 11.

5.1. Simulation Results of the First Approach of Control: The Neural Control for Improvement of a Classic Controller Proportional Derivative.

Each line, that appears in both diagrams of Figure 8, represents one variation of weight value $W_{i,j}$ in the update of weight matrix W or $M_{j,k}$ in the update of weight matrix M.

Analysis Results. The analysis of the simulation results of the response system equipped with NN controller for improvement of a classic controller PD, seen in Figure 6, shows that this control law can satisfy the stability of the system despite the presence of disturbances and robotic uncertainties.

However, due to disturbances and robotic uncertainties, the peak in the torques response makes this control strategy not reliable Figure 7.

5.2. Simulation Results of the Second Approach of Control: The Neural Control via a High-Order Dynamic Neural Network.

For this approach, the initialization of neural network weights is not arbitrary and a training phase is necessary.

(i) *The Learning Step*. Each line, that appears in both diagrams of Figure 11, represents one variation of weight value $W_{fi,j}$

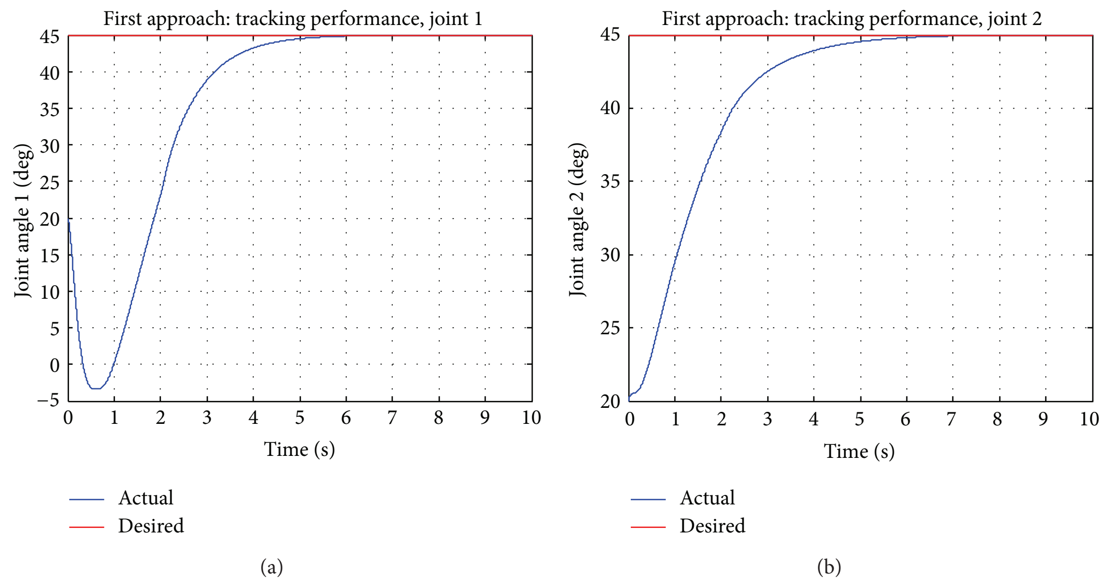


FIGURE 6: Response of joint angles: system equipped with NN controller for improvement of a classic controller PD.

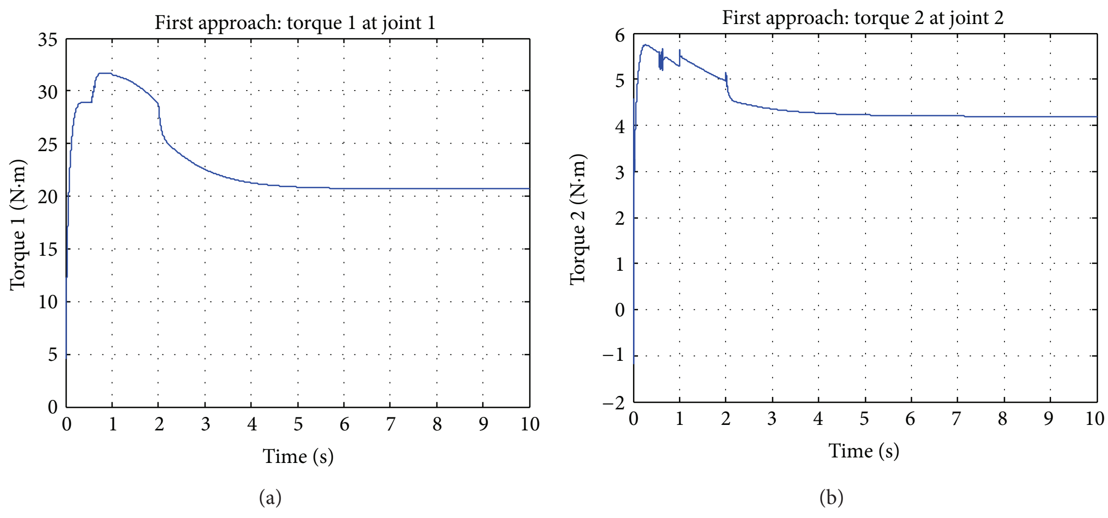


FIGURE 7: Response of torques in joint angles: system equipped with NN controller for improvement of a classic controller PD.

in the update of weight matrix W_f or $W_{gi,j}$ in the update of weight matrix W_g.

Analysis Results. The rigorous peak in the torques and joint angles response, due to disturbances and robotic uncertainties, seen in Figures 9 and 10, was able to be corrected thanks to the neural network adaptation.

At the end of this learning step, the best weight values, seen in Figure 11, are obtained.

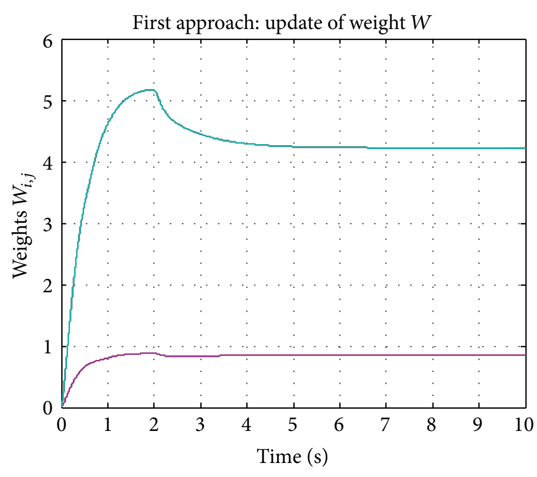
(ii) *Final Simulation Results*. The best weights values, obtained in the end of the learning step, are used as initial values for this step.

Each line, that appears in both diagrams of Figure 14, represents one variation of weight value $W_{fi,j}$ in the update of weight matrix W_f or $W_{gi,j}$ in the update of weight matrix W_g.

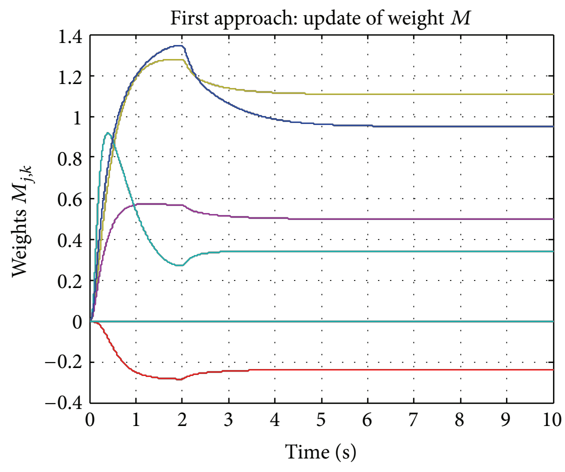
Analysis Results. The analysis of the simulation results of the response system equipped with NN controller via a high-order dynamic neural network, seen in Figure 12, shows that this control law can satisfy the stability of the system despite the presence of disturbances and robotic uncertainties.

The learning step is necessary to obtain the best weight values, seen in Figure 11, which represent the true initial weight values.

After the learning step and despite the presence of disturbances and robotic uncertainties, no malfunction was identified in the torques response, seen in Figure 13 (as the peak in the torques response, seen in Figure 7).

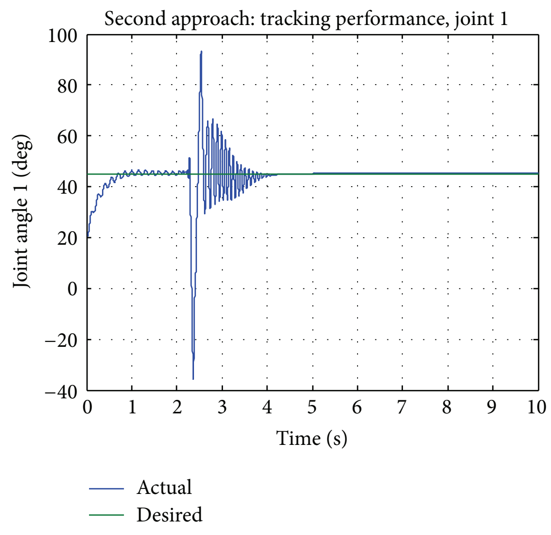


(a)

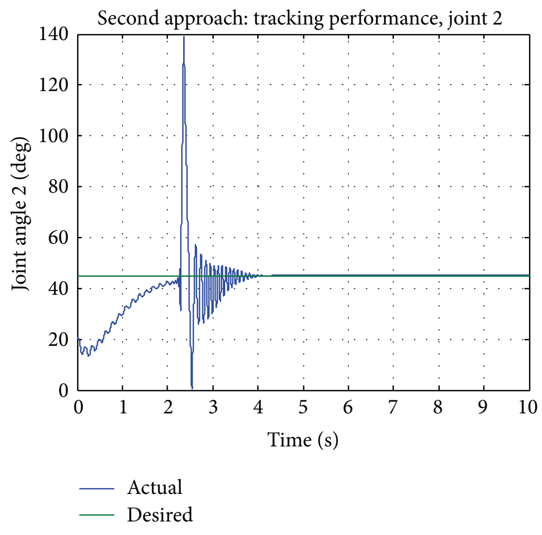


(b)

FIGURE 8: Weights update of the NN controller for improvement of a classic controller PD.

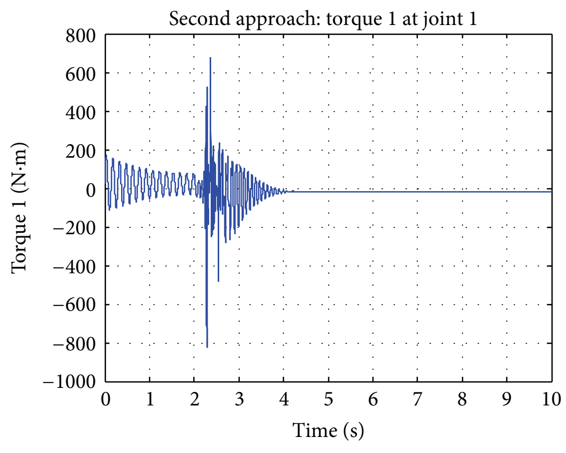


(a)

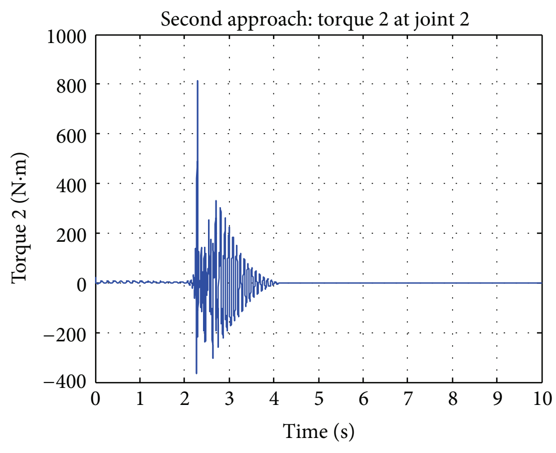


(b)

FIGURE 9: Response of joint angles: system equipped with NN controller via a high-order dynamic neural network (learning step).



(a)



(b)

FIGURE 10: Response of torques in joint angles: system equipped with NN controller via a high-order dynamic neural network (learning step).

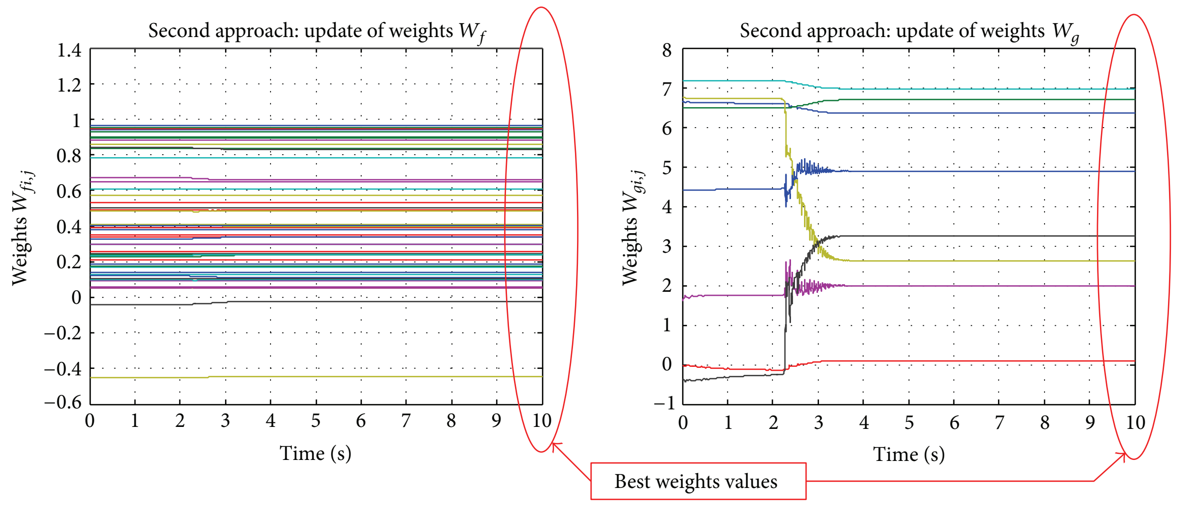


Figure 11: Weights update of the NN controller via a high-order dynamic neural network (learning step).

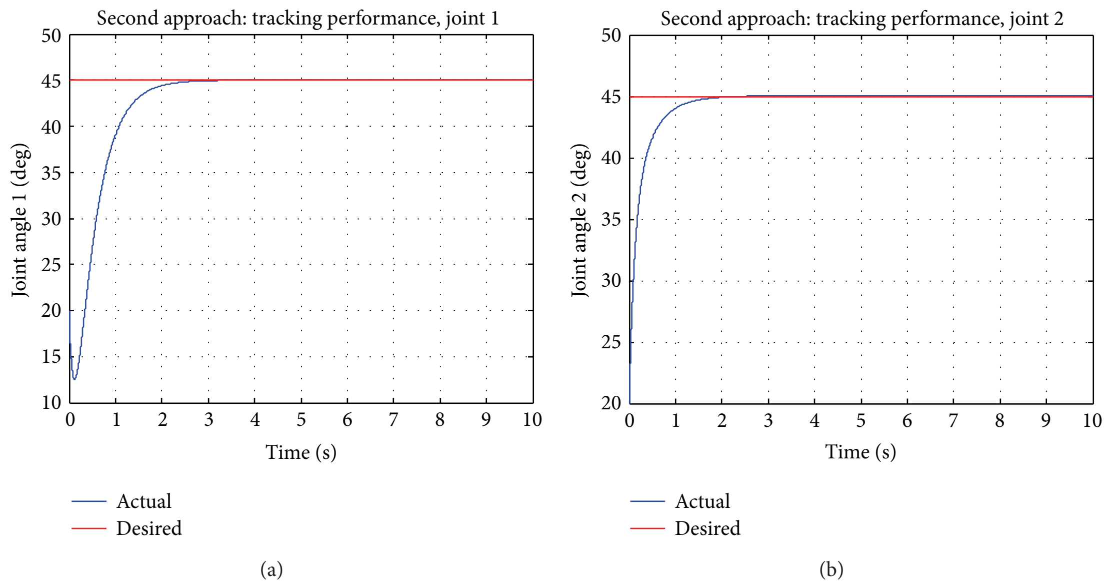


Figure 12: Response of joint angles: system equipped with NN controller via a high-order dynamic neural network.

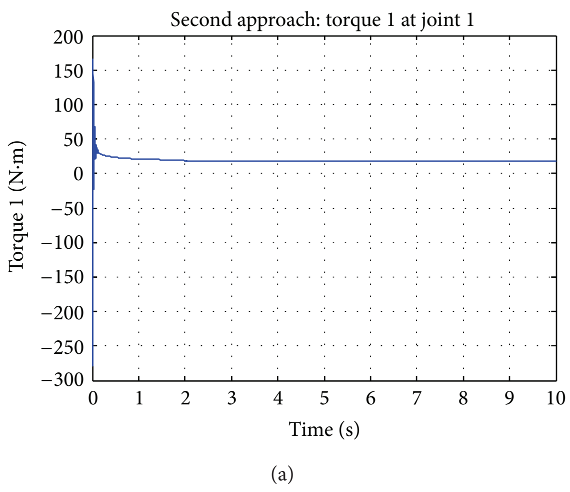
In Figure 14, it is easy to see that weights have really achieved their best values in the learning step and they remain nearly constant.

5.3. Comparative Study. The advantages and limitations of each approach, of neural network control, are presented in Table 2.

The run-time performance of each approach is presented in Table 3.

6. Discussion and Conclusion

In this paper, two approaches of neural network robot control were selected, exposed, and compared. The aim of this comparative study is to find the performance differences between static and dynamic neural networks in robotic systems control. So, one of these two approaches uses a static neural network; the other uses a dynamic neural network. The first approach is a direct neural control for improvement of a classic controller proportional derivative (PD), proposed by Lewis [1]; it employs a static neural network to approximate the nonlinearities and uncertainties in the robot dynamics, so that the static neural network is used to compensate the nonlinearities and uncertainties; therefore it overcomes some limitation of the conventional controller PD and improves its accuracy. The second approach is an indirect neural control via a high-order dynamic neural network, proposed by Sanchez et al. [7]; it employs the dynamic neural network for an exact online identification of the robot state, and then



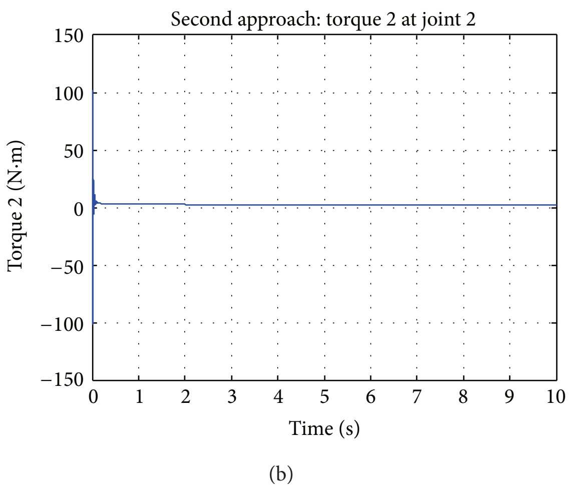
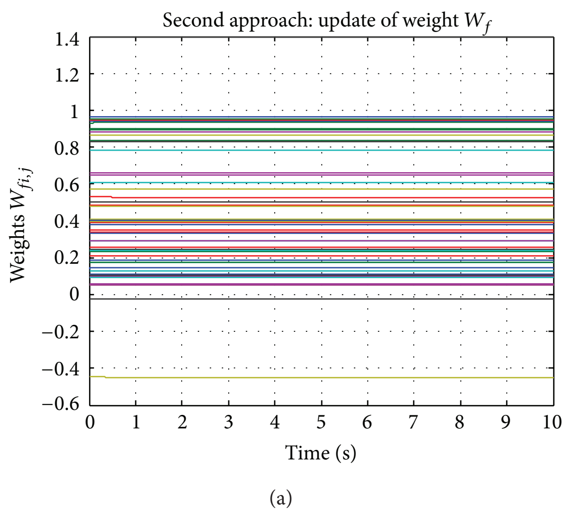


Figure 13: Response of torques in joint angles: system equipped with NN controller via a high-order dynamic neural network.



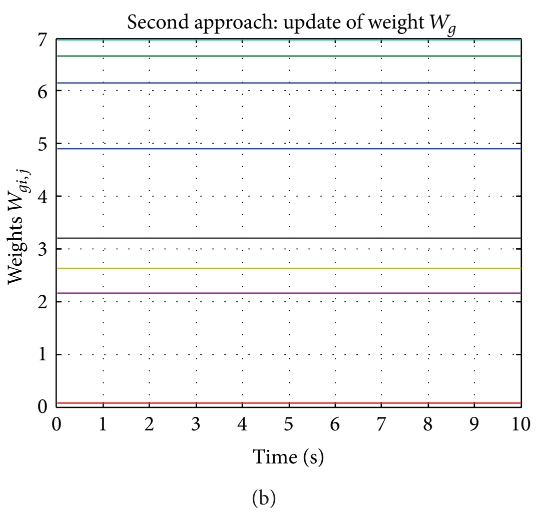


Figure 14: Weights update of the NN controller via a high-order dynamic neural network.

Table 2: Advantages and limitations of the neural network control approaches.

Control strategy	Advantages	Limitations
NN controller for improvement of a classic controller PD	(i) Initialization of the NN weights is arbitrary (ii) Online learning of the NN (iii) No offline training requirements (iv) Approximation, by the neural network, of the function which gathers the nonlinearities and uncertainties included in the robot dynamics (v) Overcoming some limitation of the conventional controller PD (vi) Guaranteed stability in presence of nonlinearities and uncertainties	No reliable response of the system, seen in Figure 7, due to the peak in the torques response, facesdisturbances and robotic uncertainties
NN controller via a high order dynamic neural network	(i) An exact online identification of the robot state thanks to the dynamic neural network (ii) Guaranteed stability in presence of nonlinearities and uncertainties (iii) Reliable response of the system faces disturbances and robotic uncertainties (Figures 12 and 13)	(i) Initialization of the NN weights is not arbitrary (ii) Offline training requirements to find the suitable initial NN weights values

Table 3: Run-time performance of the neural network control approaches (on a time-domain $t = [0–10]$ sec).

Control strategy	Elapsed time (in seconds)
NN controller for improvement of a classic controller PD	45.262542
NN controller via a high-order dynamic neural network	44.088512

it synthesizes the control law from this information recovered by this identification.

Simulation results under the framework MATLAB, of a two-link robot in 2D environment, showed the performance differences between the two neural network control approaches studied. Compared to the control with the static neural network, the neural control via dynamic neural network has significantly better tracking performance, has faster response time and is more reliable to face disturbances and robotic uncertainties. However, the indirect approach requires offline training to find the suitable initial neural network weights values, contrarily to the direct one in which the initialization of the neural network weights is arbitrary.

Although it is clear that further experimentation needs to take place, simulation results presented here indicate that dynamic neural networks have demonstrated a very good potential for applications in closed loop control of robot manipulators.

References

[1] F. L. Lewis, "Neural network control of robot manipulators," *IEEE Expert*, vol. 11, no. 3, pp. 64–75, 1996.

[2] W. Zhang, N. Qi, and H. Yin, "PD control of robot manipulators with uncertainties based on neural network," in *Proceedings of the International Conference on Intelligent Computation Technology and Automation (ICICTA '10)*, pp. 884–888, May 2010.

[3] Y. H. Kim, F. L. Lewis, and D. M. Dawson, "Intelligent optimal control of robotic manipulators using neural networks," *Automatica*, vol. 36, no. 9, pp. 1355–1364, 2000.

[4] Z. Tang, M. Yang, and Z. Pei, "Self-Adaptive PID control strategy based on RBF neural network for robot manipulator," in *Proceedings of the 1st International Conference on Pervasive Computing, Signal Processing and Applications (PCSPA '10)*, pp. 932–935, September 2010.

[5] D. Popescu, D. Selisteanu, and L. Popescu, "Neural and adaptive control of a rigid link manipulator," *WSEAS Transactions on Systems*, vol. 7, no. 6, pp. 632–641, 2008.

[6] M. A. Brdys and G. J. Kulawski, "Stable adaptive control with recurrent networks," *Automatica*, vol. 36, no. 1, pp. 5–22, 2000.

[7] E. N. Sanchez, L. J. Ricalde, R. Langari, and D. Shahmirzadi, "Rollover control in heavy vehicles via recurrent high order neural networks," in *Recurrent Neural Networks*, X. Hu and P. Balasubramaniam, Eds., Vienna, Austria, 2008.

[8] F. G. Rossomando, C. Soria, D. Patino, and R. Carelli, "Model reference adaptive control for mobile robots in trajectory tracking using radial basis function neural networks," *Latin American Applied Research*, vol. 41, no. 2, 2010.

[9] Z. Pei, Y. Zhang, and Z. Tang, "Model reference adaptive PID control of hydraulic parallel robot based on RBF neural network," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO '07)*, pp. 1383–1387, December 2007.

[10] M. G. Zhang and W. H. Li, "Single neuron PID model reference adaptive control based on RBF neural network," in *Proceedings of the International Conference on Machine Learning and Cybernetics*, pp. 3021–3025, August 2006.

[11] H. X. Li and H. Deng, "An approximate internal model-based neural control for unknown nonlinear discrete processes," *IEEE Transactions on Neural Networks*, vol. 17, no. 3, pp. 659–670, 2006.

[12] I. Rivals and L. Personnaz, "Nonlinear internal model control using neural networks: application to processes with delay and design issues," *IEEE Transactions on Neural Networks*, vol. 11, no. 1, pp. 80–90, 2000.

[13] C. Kambhampati, R. J. Craddock, M. Tham, and K. Warwick, "Inverse model control using recurrent networks," *Mathematics and Computers in Simulation*, vol. 51, no. 3-4, pp. 181–199, 2000.

[14] M. Wang, J. Yu, M. Tan, and Q. Yang, "Back-propagation neural network based predictive control for biomimetic robotic fish," in *Proceedings of the 27th Chinese Control Conference (CCC '08)*, pp. 430–434, July 2008.

[15] K. Kara, T. E. Missoum, K. E. Hemsas, and M. L. Hadjili, "Control of a robotic manipulator using neural network based predictive control," in *Proceedings of the 17th IEEE International Conference on Electronics, Circuits, and Systems (ICECS '10)*, pp. 1104–1107, December 2010.

[16] S. Ferrari and R. F. Stengel, "Smooth function approximation using neural networks," *IEEE Transactions on Neural Networks*, vol. 16, no. 1, pp. 24–38, 2005.

[17] E. B. Kosmatopoulos, M. A. Christodoulou, and P. A. Ioannou, "Dynamical neural networks that ensure exponential identification error convergence," *Neural Networks*, vol. 10, no. 2, pp. 299–314, 1997.

[18] E. N. Sanchez, J. P. Perez, and L. Ricalde, "Recurrent neural control for robot trajectory tracking," in *Proceedings of the 15th World Congress International Federation of Automatic Control*, Barcelona, Spain, July 2002.