Identification and Control of Three-Links Electrically Driven Robot Arm Using Fuzzy Neural Networks

Salam A. Abdulkereem salam_eng@yahoo.com
Department of Computers Engineer
University of Basra
Basra,Iraq

Prof. Dr. Abduladhem A. Ali abduladem1@yahoo.com
Department of Computers Engineer
University of Basra
Basra,Iraq

Abstract

This paper uses a fuzzy neural network (FNN) structure for identifying and controlling nonlinear dynamic systems such three links robot arm. The equation of motion for three links robot arm derived using Lagrange's equation. This equation then combined with the equations of motion for dc. servo motors which actuated the robot. For the control problem, we present the forward and inverse adaptive control approaches using the FNN. Computer simulation is performed to view the results for identification and control.

Keywords: Fuzzy Neural Control, Robot Control, Forward Adaptive Control, Inverse Control, Adaptive Systems

1. INTRODUCTION

In the past decade, the applications of intelligent control techniques (fuzzy control or neural-network control) to the motion control of robotic manipulators have received considerable attention [1], [5]. In general, robotic manipulators have to face various uncertainties in their dynamics, such as payload parameter, friction, and disturbance. It is difficult to establish an appropriate mathematical model for the design of a model based control system. Thus, the general claim of these intelligent control approaches is that they can attenuate the effects of structured parametric uncertainty and unstructured disturbance by using their powerful learning ability without a detailed knowledge of the controlled plant in the design processes. Feed forward neural networks have been shown to obtain successful results in system identification and control [6]. Such neural networks are static input/output mapping schemes that can approximate a continuous function to an arbitrary degree of accuracy. Results have also been extended to recurrent neural networks [7], [9]. For example, Jin et al. [8] studied the approximation of continuous-time dynamic systems using the dynamic recurrent (DRNN) and a Hopfield-type DRNN was presented by Funahashi and Nakamura [7]. As is widely known, both fuzzy logic systems and neural network systems are aimed at exploiting human-like knowledge processing capability. Moreover, combinations of the two have found extensive applications. This approach involves merging or fusing fuzzy systems and neural networks into an integrated system to reap the benefits of both [10]. For instance, Lin and Lee [11] proposed a general neural network model for a fuzzy logic control and decision system, which is trained to control an unmanned vehicle.
In this paper FNN is used to identify and control a three links robot arm. We present the forward and inverse identification as offline learning to use the parameters of this stage in control stage. For control problem, we present the indirect (forward) and direct (inverse) control. Computer simulation implements to view the results of robot arm application.

This paper is organized as follows. Section II presents the dynamic model of a three-link robot arm including actuator dynamics briefly [12], [14]. Section III shows the FNN structure. Section IV FNN identification .Section V presents the FNN control. Section VI presents the simulation results finally section IX shows the conclusion.

2. Dynamic model of three links Robot arm

Dynamic modeling of a robot manipulator consists of finding the mapping between the forces exerted on the structures and the joint positions, velocities and accelerations. Two formulations are mainly used to derive the dynamic model: namely the Lagrange formulation and the Newton-Euler formulation. A large number of authors and researchers [15], [17], used Lagrange's approach to drive the general form of robot equation of motion. The Lagrange equations thus taking on the alternative form [18]:

$$I(\theta)\ddot{\theta} + \dot{I}(\theta,\dot{\theta})\dot{\theta} - \frac{1}{2}\left[\frac{\partial(I\dot{\theta})}{\partial\theta}\right]^{T}\dot{\theta} + \frac{\partial F}{\partial\dot{\theta}} = \tau_n \qquad (1)$$

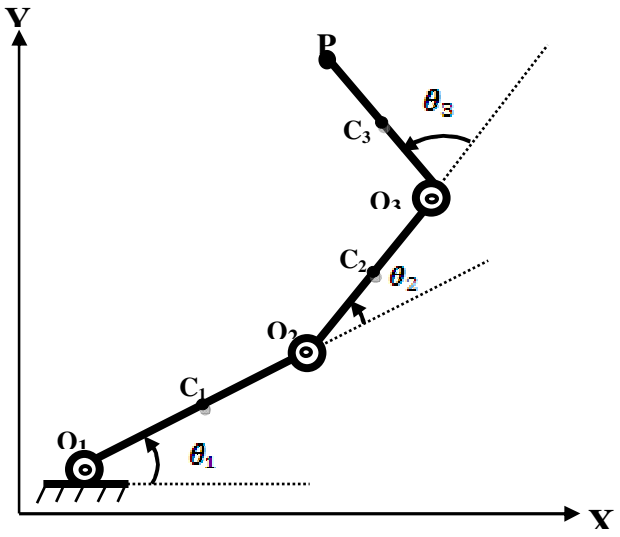

FIGURE 1: Three Links Robot Arm

Where $\theta, \dot{\theta} and\ \ddot{\theta} \in R^n$ denote the vectors of joint link positions, velocities and acceleration respectively, I (θ) $\in R^{n\times n}$ denotes the inertia matrix, n denote the number of link, P is the potential energy and τ_n denoted the torque of n link. Consider the manipulator of Fig. (1), with links designed so that their mass centers, C1, C2, and C3, are located at the midpoints of segments O1O2, O2O3, and O3P, respectively. Moreover, the ith link has a mass (m_n)and a centroidal moment of inertia in a direction normal to the plane of motion (I_n); while the joints are actuated by motors delivering torques τ_1, τ_2, and τ_3, the lubricant of the joints producing dissipative torques that we will neglect in this model. Under the assumption that gravity acts in the direction of Y axis. In general, the dynamic model of armature-controlled DC servo motors which shown below, on an n- link robot manipulator can be expressed in the following form [17]:

$$\tau_e = K_T i_a \qquad (2)$$

$$\tau_e = J_m \ddot{q}_m + B_m \dot{q}_m + \tau_m \qquad (3)$$

$$v_t = R_a i_a + L_a \frac{di_a}{dt} + K_E \dot{q}_m \qquad (4)$$

Where $\tau_e \in R^n$ is the vector of electromagnetic torque, $K_T \in R^{n\times n}$ is the diagonal matrix of

motor torque constants, $i_a \in R^n$ is the vector of armature currents, $J_m \in R^{n\times n}$ is the diagonal matrix of the moment inertia, $B_m \in R^{n\times n}$ is the diagonal matrix of torsional damping coefficients, $q_m, \dot{q}_m \; and \; \ddot{q}_m \in R^n$ denote the vectors of motor shaft positions, velocities, and accelerations, respectively, $\tau_m \in R^n$ is the vector of load torque, $v_t \in R^n$ is the vector of armature input voltages, $R_a \in R^{n\times n}$ is the diagonal matrix of armature resistance, $L_a \in R^{n\times n}$ is the diagonal matrix of armature inductance, and $K_E \in R^{n\times n}$ is the diagonal matrix of the back electromotive force (EMF) coefficients. In order to apply the dc servo motors for actuating an n-link robot manipulator, a relationship between the joint position θ and the motor-shaft position q_m can be represented as follows [17]:

$$G_r = \frac{q_m}{\theta} = \frac{\tau}{\tau_m} \qquad (5)$$

The governed equation of an n-link robot manipulator including actuator dynamics can be obtained as [1]:

$$I^*(\theta)\ddot{\theta} + D(\theta, \dot{\theta}, \ddot{\theta}) + d = U \qquad (6)$$

Where $U \in R^n$ represents the control effort vector, i.e. armature input voltages,

$$I(\theta) = I_n + I_q(\theta)$$

$$I^* = L_n[I_n + J_n] \qquad (7)$$

$$C(\theta, \dot{\theta}) = \dot{I}(\theta, \dot{\theta}) - \frac{1}{2}\left[\frac{\partial(I\dot{\theta})}{\partial\theta}\right]^T$$

$$D(\theta, \dot{\theta}, \ddot{\theta}) = \{L_n[\dot{I}(\theta, \dot{\theta}) + C(\theta, \dot{\theta}) + B_n] + [I(\theta) + J_n]\}\ddot{\theta} + [L_n\dot{C}(\theta, \dot{\theta}, \ddot{\theta})\dot{\theta} + R_nC(\theta, \dot{\theta})\dot{\theta} + R_nB_n\dot{\theta} + K_{En}\dot{\theta} + L_n\dot{G}(\theta, \dot{\theta}) + R_nG(\theta)] \qquad (8)$$

$$d = L_nI_q(\theta)\ddot{\theta} + L_n\dot{N} + R_nN \qquad (9)$$

Where $G(\theta)$ is gravity vector, N represents the vector of external disturbance t_l and friction term $f(\dot{\theta})$. Then we can re-write Eqn. (6) as:

$$\ddot{\theta} = I^*(\theta)^{-1}[U - (D(\theta, \dot{\theta}, \ddot{\theta}) + d)] \qquad (10)$$

By using method of numerical integration such Euler method for Eqn. (10) we can get position, velocity and acceleration for each link.

3. Fuzzy Neural Networks (FNN)

The Architecture of FNN shown in (fig, 2). FNN considered as a special type of neural network [19], this means special connection and node operation. Every layer and every node have its practical meaning because the FNN has the structure which is based on both the fuzzy rules and inference. In the following items each layer shown in (fig, 2) will be described:

1- Input layer
   Input layer transmits the input linguistic variables x_n to the output without changed**.**
2- Hidden layer I
   Membership layer represents the input values with the following Gaussian membership functions [20]:

$$\mu_j^i = \exp\left(-\frac{1}{2}\frac{(x_j - c_{ij})^2}{s^2}\right) \qquad (11)$$

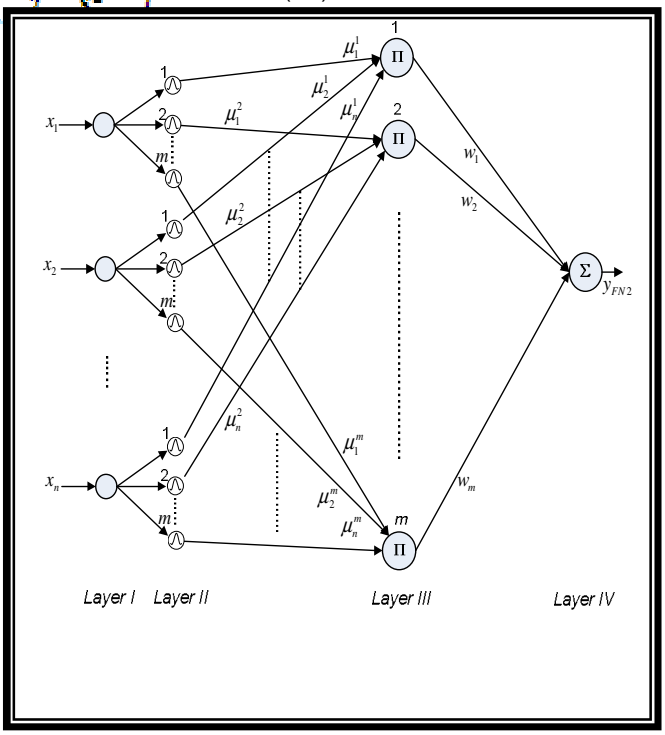

FIGURE 2: Architecture of FNN

Where C_{ij} and S_{ij} (i=1, 2, ... , n ; j=1, 2,.., m), respectively, are the mean and standard deviation of the Gaussian function in the j[th] term of the i[th] input linguistic variable x_n to the node of this layer.

3-Hidden layer II

Rule layer implements the fuzzy inference mechanism, and each node in this layer multiplies the input signals and outputs the result of the product. The output of this layer is given as [20]:

$$\emptyset_i = \prod_j^n \mu_j^i \qquad (12)$$

Where $\emptyset_i$ represent the i[th] output of rule layer.

4- Output layer

Layer four is the output layer, and nodes in this layer represent output linguistic variables. Each node $y_o (o = 1, \ldots, N_0)$, which computes the output as [20]:

$$y_o = \sum_i^m w_i^o \emptyset_i \qquad (13)$$

3.1 Learning Algorithm FNN Identifier

There are three types of parameters in the fuzzy-neural network can be adapted, in the primes part: the center values C_{ij} and width values S_{ij} of the Gaussian membership functions, whereas, in the consequence part: the consequence weights values w_i . Once the fuzzy-neural network has been initialized, a gradient decent based back-propagation algorithm is employed to adjust

the parameters of the fuzzy-neural network by using the training patterns. The main goal of supervised learning algorithm is to minimize the mean square error function [20]:

$$E = \frac{1}{2}\left(y_{Fnn} - y_p\right)^2 \qquad (14)$$

Where y_{Fnn} is the output of is fuzzy-neural network and y_p is the desired output. The gradient descent algorithm gives the following iterative equations for the parameter values [20]:

$$w_i(k+1) = w_i(k) - \eta_w \frac{\partial E}{\partial w_i} \qquad (15)$$

$$C_{ij}(k+1) = C_{ij}(k) - \eta_c \frac{\partial E}{\partial C_{ij}} \qquad (16)$$

$$S_{ij}(k+1) = S_{ij}(k) - \eta_s \frac{\partial E}{\partial S_{ij}} \qquad (17)$$

Where η is the learning rate for each parameter in the system, i=1,2…*n* and j=1,2…*m*. Taking the partial derivative of the error function given by Eqn. (14), we can get the following equations:

$$\frac{\partial E}{\partial w_i} = \left(y_{FNN} \quad y_p\right)\phi_i \qquad (18)$$

$$\frac{\partial E}{\partial C_{ij}} = \left(y_{FNN} - y_p\right)\phi_i w_i \frac{(x_j - C_{ij})}{S_{ij}^2} \qquad (19)$$

$$\frac{\partial E}{\partial S_{ij}} = \left(y_{FNN} - y_p\right)\phi_i w_i \frac{(x_j - C_{ij})^2}{S_{ij}^3} \qquad (20)$$

4. Identification

Two representations are available to identify a dynamical system depending on type of the output feedback these are parallel model and Series-parallel model [6]. In this paper the series-parallel identification model is desired. A series-parallel model that is obtained by feeding back the past values of the plant output (rather than the identifier output) as shown in (Fig,3). This implies that in this case the identification model has the form [6]:

$$\hat{y}(k+1) = \begin{bmatrix} y_p(k), y_p(k-1), \ldots, y_p(k-n+1), \\ u(k), u(k-1), \ldots u(k-m+1) \end{bmatrix} \qquad (21)$$

The identifier output is represented by $\hat{y}(k)$ and the plant output is denoted by $y_p(k)$.

5. FNN Control

For system control problems, we focus on the adaptive control of dynamic systems using FNN. These algorithms denoted as “Fuzzy Neural Model Reference Controller” (FNMRC) in this type of controllers, back propagation training algorithm is used [16]. There are two distinct approaches for the FNMRC, the first one result in a direct scheme (inverse control) for the controller and the second result in an indirect scheme (forward control), the difference between the two may be shown in figure (4.1) and (4.2).

5.1 Learning Algorithm for Indirect FNN Control (Forward)

Indirect control architecture usually requires an identified system model and the controller design is based on the learning algorithm. Our goal is to minimize the following cost function:

$$E_c = \frac{1}{2}\left(e_{c1}^2 + e_{c2}^2 + e_{c3}^2\right) \qquad (22)$$

Where e_{c1}, e_{c2} and e_{c3} are errors between reference outputs and robot's link1, link2 and link3

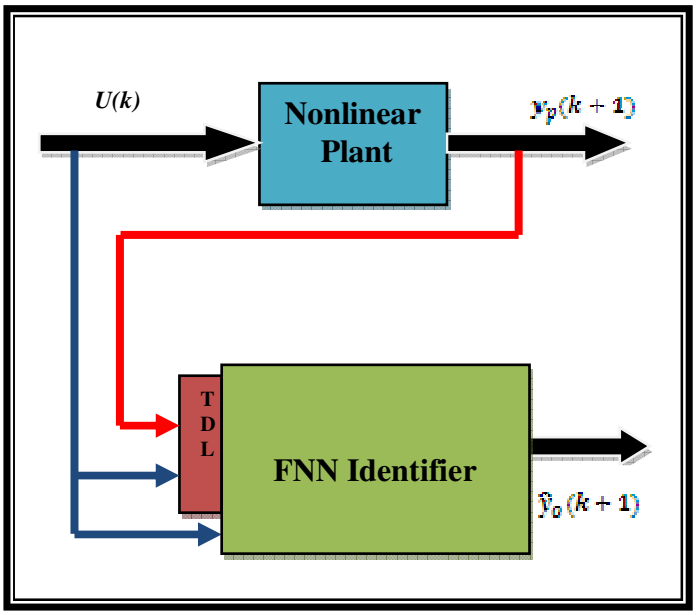

FIGURE 3: Series-Parallel identification model

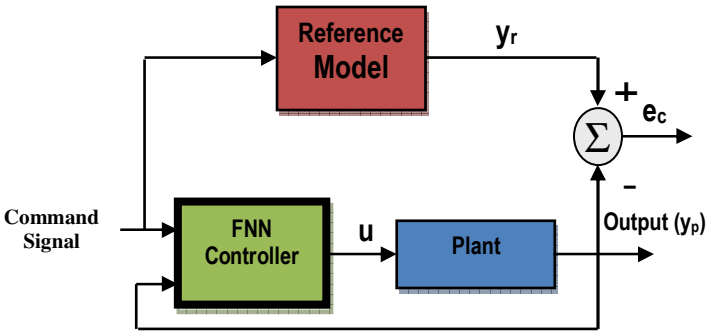

FIGURE 4.1: Direct FNN model reference learning controller

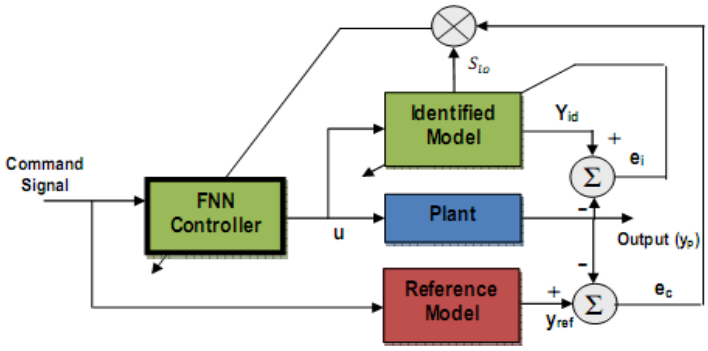

FIGURE 4.2: Indirect FNN model reference learning controller

output respectively, then the gradient of error E_c with respect to weights , mean and standard deviation of the Gaussian function are given:

$$\frac{\partial E_c}{\partial w_{cli}} = \emptyset_{ci} \sum_o^3 e_{co} S_{lc} \qquad (23)$$

$$\frac{\partial E_c}{\partial m_{cij}} = A1(e_{c1}S_{11} + e_{c2}S_{12} + e_{c3}S_{13}) + B1(e_{c1}S_{21} + e_{c2}S_{22} + e_{c3}S_{23}) + C1(e_{c1}S_{31} + e_{c2}S_{32} + e_{c3}S_{33}) \qquad (24)$$

$$\frac{\partial E_c}{\partial s_{cij}} = A2(e_{c1}S_{11} + e_{c2}S_{12} + e_{c3}S_{13}) + B2(e_{c1}S_{21} + e_{c2}S_{22} + e_{c3}S_{23}) + C2(e_{c1}S_{31} + e_{c2}S_{32} + e_{c3}S_{33}) \qquad (25)$$

The identifier can provide the system sensitivity S_{lo} and it can be computed by the chain rule:

$$\frac{\partial \hat{y}_{po}}{\partial u_l} = S_{lo} = \frac{\partial \hat{y}_{po}}{\partial \emptyset_i} \frac{\partial \emptyset_i}{\partial mf_{ij}} \frac{\partial mf_{ij}}{\partial u_l}$$

$$S_{lo} = -\sum_{i=1}^{n_i} w_{oi} \emptyset_i \frac{u_l - m_{il}}{s_{il}^2} \qquad (26)$$

Where u_l, y_{po} are l^{th} output of FNN controller, O^{th} output of robot plant respectively and where:

$$A1 = w_{c1i} \emptyset_{ci} \frac{x_{cj} - m_{cij}}{s_{cij}^2}$$

$$B1 = w_{c2i} \emptyset_{ci} \frac{x_{cj} - m_{cij}}{s_{cij}^2}$$

$$C1 = w_{c3i} \emptyset_{ci} \frac{x_{cj} - m_{cij}}{s_{cij}^2}$$

$$A2 = w_{c1i} \emptyset_{ci} \frac{(x_{cj} - m_{cij})^2}{s_{cij}^3}$$

$$B2 = w_{c2i} \emptyset_{ci} \frac{(x_{cj} - m_{cij})^2}{s_{cij}^3}$$

$$C2 = w_{c3i} \emptyset_{ci} \frac{(x_{cj} - m_{cij})^2}{s_{cij}^3}$$

5.2 Learning Algorithm for Direct FNN Control (Inverse)

The FNN inverse control is shown in the (fig, 5), in which two FNN are present, one for the inverse identification and the other for controller. The basic structure of the inverse controller consists of the controller network only, which is the same to the identifier network in the offline learning. The simple concept of the inverse controller is the controller block that represents the inverse transfer function of the robot plant, so the product result of the two blocks (robot plant and

controller) must equal unity. Hence the output of the robot plant will be equal to desired input of the controller.

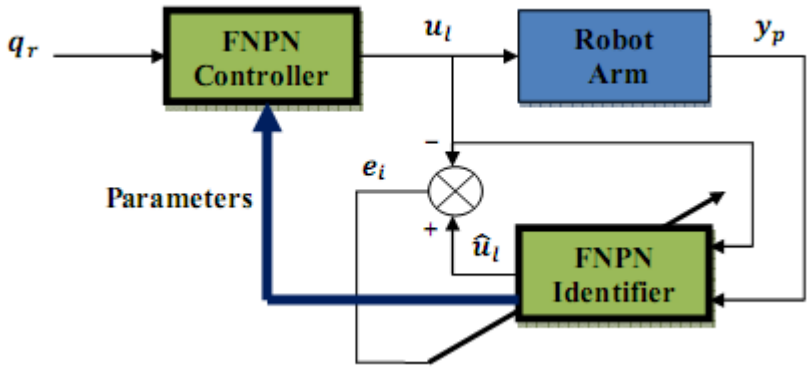

FIGURE 5: FNN inverse control

6. SIMULATION RESULTS

In the following examples two methods of control presented in above sections are implemented by FNN for three links robot arm. The simulation carried by MATLABT software. The no. of rules and outputs are 50 and 3 respectively, in each method of control. The initial mean and standard of membership function were computed as Eqns. (27) and (28) [21], beside the 0.001values for weights. Following two examples are viewed. The most important parameters that affect the control performance of the robotic system are the external disturbance $t_l(t)$, the friction term $f(\dot{\theta})$, and the parameter variation of 3rd link's mass m_3 . In all two example simulation, three circumstances including the:

1- Nominal situation ($m_3 = 1$kg and N=0) at beginning.
2- Parameter variation situation occurring at t=15 sec ($m_3 = 2$ kg).
3- Disturbance in addition, friction forces are also considered in this simulation.

Hence,

$$t_l(t) = [5sin(5t) \quad 3sin(5t) \quad sin(5t)]^T$$

$$f(\dot{\theta}) = [20\dot{\theta}_1 + 8sign(\dot{\theta}_1) \quad 10\dot{\theta}_2 + 4sign(\dot{\theta}_2) \quad 5\dot{\theta}_3 + 2sign(\dot{\theta}_3)]^T$$

$$N = t_l(t) + f(\dot{\theta})$$

$$m_{ij} = X_{n\,max} - (i-1)\frac{X_{n\,max} - X_{n\,min}}{N_i - 1} \qquad (27)$$

$$s_{ij} = 2\frac{X_{n\,max} - X_{n\,min}}{N_i - 1} \qquad (28)$$

Where $X_{n\,max}$, $X_{n\,min}$ are the predetermined maximal and minimal bounds of n^{th} input to FNN.

6.1 Example 1

In this example the forward control is implemented by FNN in each one the forward identifier is used to calculate the plant sensitivity, the initial parameters of identifier will take from final values proceed in the offline learning. The eighteen inputs are fed to FNN controller, the learning rate of weights, mean and standard are $\eta_{cw} = .01$, $\eta_{cm} = .0012$ and $\eta_{cs} = .005$ respectively . figures (6.a) to (6.f) are shown the FNN forward control position response and mean square error for link1, link2 and link3 respectively for 100 epochs.

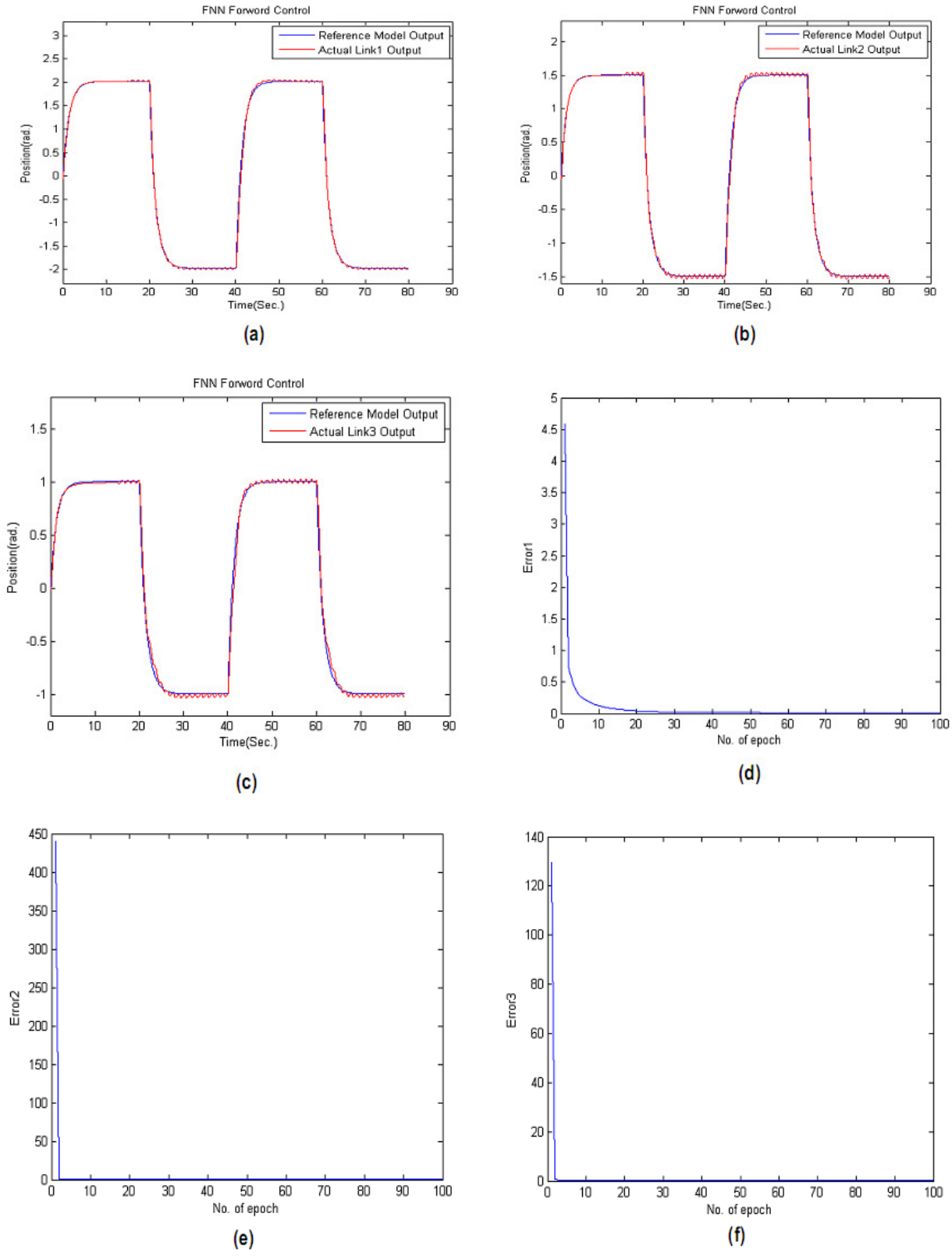

FIGURE 6: FNN forward control simulation results of position response and mean square error for Link1, Link2 and Link3 (a)-(f)

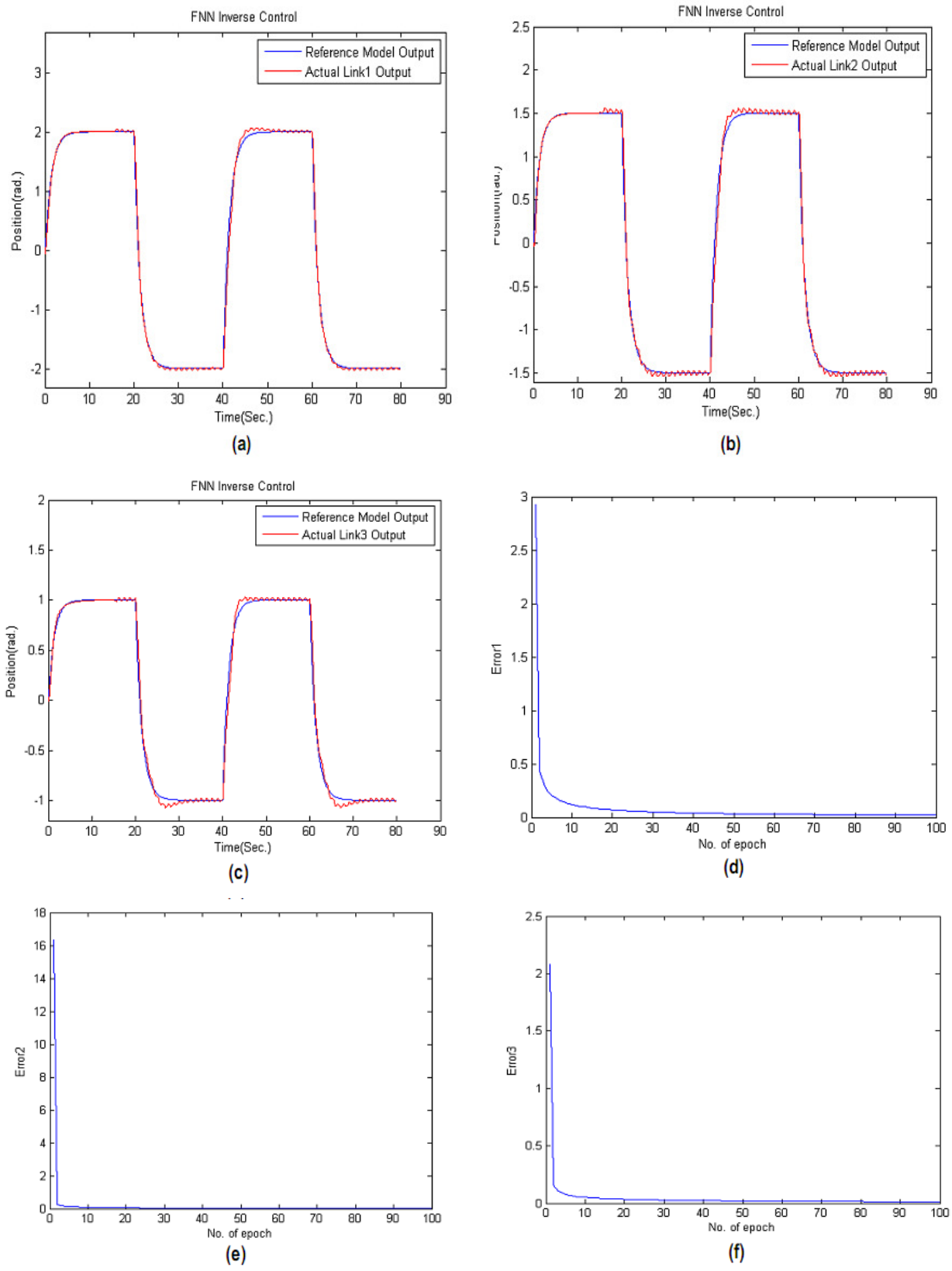

FIGURE 7: FNN inverse control simulation results of position response and mean square error for Link1, Link2 and Link3 (a)-(f)

6.2 Example 2

The FNN inverse control presented in this example, the structure of controller same the structure of the inverse identifier which only changes the $y_p(k+1)$ input to inverse identifier by reference input $q_r(k+1)$ to inverse controller. Inverse identifier used here so that the parameters generated in offline learning considered initial parameters to online inverse identifier and inverse controller. The eighteen inputs are fed to FNN controller, the learning rate of weights, mean and standard are $\eta_{cw} = 0.02$, $\eta_{cm} = 0.0012$ and $\eta_{cs} = 0.005$ respectively. Figures (7.a) to (7.f) are shown the FNN inverse control position response and mean square error for link1, link2 and link3 respectively for 100 epochs.

7. CONCLUSION

In this paper use FNN for identification and control for dynamic nonlinear systems such three links robot arm. From the previous examples we conclude that the FNN is powerful for identify and control nonlinear system, in example1 use indirect control with online forward identification and the gradient in mean square error is done and in example2 use the direct control technique with online inverse identification after use the parameters are get from offline inverse identification to use in online work. Table (1) shows the gradient mean square error for both examples for each link and the mean square error for each link when we applied the traditional PD control (Proportion and Derivative control) on them in order to compare the values of MSE among example1, example2 (they applied by FNN control) and PD control, the main difference between FNN control and traditional PD control is a PD control can't adapt its gains (k_p, k_d) when some disturbance insert to plant in otherwise the FNN control can adapt its parameters (w_c, m_{cij} and s_{cij}) by online learning algorithm.
For future work the control technique by FNN without identification will study to reduce load of computation.

Example	Link1	Link2	Link3
1	0.0093	0.0093	0.0085
2	0.018	0.0181	0.0125
PD control	0.0095	0.009	0.0098

TABLE 1: Mean square error

8. REFERENCES

[1] Rong-jong Wa and Po-Chen Chen,"Robust Neural-Fuzzy-Network Control for Robot Manipulator Including Actuator Dynamics", IEEE Trans. Indst. Elect. vol. 53, no. 4, Aug. 2006.

[2] S. J. Huang and J. S. Lee, "A stable self-organizing fuzzy controller for robotic motion control," IEEE Trans. Ind. Electron., vol. 47, no. 2, pp. 421–428, Apr. 2000.

[3] B. K. Yoo and W. C. Ham, "Adaptive control of robot manipulator using fuzzy compensator," IEEE Trans. Fuzzy Syst., vol. 8, no. 2, pp. 186–199, Apr. 2000.

[4] Y. C. Chang, “Neural network-based H-infinite tracking control for robotic systems,” Proc. Inst. Electr . Eng.—Control Theory Appl., vol. 147, no. 3, pp. 303–311, May 2000.

[5] Y. H. Kim and F. L. Lewis, “Optimal design of CMAC neural-network controller for robot manipulators,” IEEE Trans. Syst., Man, Cybern. C, Appl. Rev., vol. 30, no. 1, pp. 22–31, Feb. 2000.

[6] K. S. Narendra and K. Parthasarathy, “Identification and control of dynamical system using neural networks,” IEEE Trans. Neural Networks, vol. 1, pp. 4–27, Jan. 1990.

[7] K. Funahashi and Y. Nakamura, Approximation of dynamical systems by continuous-time recurrent neural network,” Neural Networks, vol.6, pp. 801–806, 1993.

[8] L. Jin, P. N. Nikiforuk, and M. Gupta, “Approximation of discrete-time state-space trajectories using dynamic recurrent neural networks,” IEEE Trans. Automat. Contr., vol. 40, pp. 1266–1270, July 1995.

[9] C. C. Ku and K. Y. Lee, “Diagonal recurrent neural networks for dynamic systems control,” IEEE Trans. Neural Networks, vol. 6, pp.144–156, Jan. 1995.

[10] Ching-Hung Lee and Ching-Cheng Teng, “Identification and Control of Dynamic Systems Using Recurrent Fuzzy Neural Networks”, IEEE Trans. Fuzzy system, vol. 8, no. 4, Aug. 2000

[11] C. T. Lin and C. S. G. Lee, “Neural-network-based fuzzy logic control and decision system,” IEEE Trans. Computer. , vol. 40, pp. 1320–1336, Dec. 1991.

[12] B. S. Chen, H. J. Uang, and C. S. Tseng, “Robust tracking enhancement of robot systems including motor dynamics: A fuzzy-based dynamic game approach,” IEEE Trans. Fuzzy Syst., vol. 6, no. 4, pp. 538–552,Nov. 1998.

[13] C. Ishii, T. Shen, and K. Tamura, “Robust model following control for a robot manipulator,” Proc. Inst. Electr. Eng.—Control Theory Appl., vol. 144, no. 1, pp. 53–60, Jan. 1997.

[14] R. J. Schilling, Fundamentals of Robotics: Analysis and Control. Hoboken, NJ: Prentice-Hall, 1998.

[15] Mark W.Spong, Seth H., M. Vidyasagar, “Robot Modeling and Control”, John Wiley and Sons, INC., 2001

[16] Abdul Baqi, J.N., "Neuro-Fuzzy Control of robot Arm" MSC. Thesis, University of Basrah, College of Engineering, Feb. 2004.

[17] Rong-Jong Wai, P. C. Chen, Chun-Yen Tu, “Robust Neural-fuzzy-network Control for Rigid-link Electrically Driven Robot Manipulator”, IEEE Trans. Ind. Electron., 30th annual conference, pp. 1328–1349, Nov. 2004.

[18] Jorge Angels, ”Fundamentals of robotic mechanical systems: theory, methods and Algorithms”, Springer, 2003.

[19] C. T. Leondes, "Fuzzy logic and expert systems applications", Academic Press, 1998.

[20] Rong-Jong Wai, Chia-Chin Chu," Robust Petri Fuzzy-Neural-Network Control for Linear Induction Motor Drive", IEEE trans. on Ind. Elect. , Vol. 54, No. 1, pp. 177-189, Feb. 2007.

[21] Rong-Jong Wai, Chia-Ming Liu," Design of Dynamic Petri Recurrent Fuzzy Neural Network and Its Application to Path-Tracking Control of Nonholonomic Mobile Robot", IEEE transactions on Ind. Elec., Vol. 56, NO. 7, pp.2667-2683, July 2009.