THE ANNALS OF "DUNAREA DE JOS" UNIVERSITY OF GALATI
FASCICLE III, 2000 ISSN 1221-454X
ELECTROTECHNICS, ELECTRONICS, AUTOMATIC CONTROL, INFORMATICS

NEURAL NETWORK SCHEMES IN CARTESIAN SPACE CONTROL OF ROBOT MANIPULATORS

Yiannis S. BOUTALIS, Adrian MOISE* and B. G. MERTZIOS****

*** Democritus University of Thrace, Department of Electrical and Computer Engineering*
67 100 Xanthi, Greece, e-mail: {ybout, mertzios}@ee.duth.gr
**"Petroleum-Gas" University of Ploiesti, Deptartment of Automatic Control & Computers*
Bd. Bucuresti, Nr. 39, 2000 Ploiesti, Prahova, Romania, e-mail : amoise_98@yahoo.com

Abstract: In this paper we are studying the Cartesian space robot manipulator control problem by using Neural Networks (NN). Although NN compensation for model uncertainties has been traditionally carried out by modifying the joint torque/force of the robot, it is also possible to achieve the same objective by using the NN to modify other quantities of the controller. We present and evaluate four different NN controller designs to achieve disturbance rejection for an uncertain system. The design perspectives are dependent on the compensated position by NN. There are four quantities that can be compensated: torque t, force F, control input U and the input trajectory X_d. By defining a unified training signal all NN control schemes have the same goal of minimizing the same objective functions. We compare the four schemes in respect to their control performance and the efficiency of the NN designs, which is demonstrated via simulations.

Keywords: Manipulation Robot, Neural Network, Cartesian Space Control, Jacobian.

1. INTRODUCTION

It is very common that manipulators are subject to structured and/or unstructured uncertainties. Structural uncertainty is characterised by having a correct dynamical model but with parameter uncertainty due to imprecision of the manipulator link properties, unknown loads, inaccuracies in the torque constants of the actuators, and so on. Unstructured uncertainty is characterised by unmodelled dynamics. Neural Network (NN) controllers are usually introduced to generate additional inputs to compensate for disturbances due to model uncertainties.

It is clear that the higher the degree of nonlinearly exists in the uncertainties, the greater benefits neural networks (NNs) can contribute. Cartesian space control (Fu et al, 2000) is such a case due to the following facts: - the inverse dynamic in Cartesian space is more complicated than that in joint space; - the singularity of Jacobian (J) becomes problem at hand when positions in Cartesian space are calculated from joint measurements; - more uncertainties are present. Although NN compensation for model uncertainties has been traditionally carried out by modifying the joint torque/force of the robot, it is also possible to achieve the same objective by using the NN to modify other quantities of the controller, like the reference Cartesian trajectory.

In this paper, we present and evaluate four different NN controller designs to achieve disturbance rejection for an uncertain robotic system. There are four locations that can be compensated: first, at torque level t, second, at force level F, third at control input level U and finally, compensation at input trajectory level X_d. By defining a unified training signal all NN control schemes have the same goal of minimizing the same objective functions. It has been shown (Moise et al, 2001) that, for the same neural network, the required internal signal level for the trajectory modification approach (fourth scheme) is much smaller. We compare the four schemes in respect to their control performance and the efficiency of the NN designs. We demonstrate via simulations that very significant performance improvement is obtainable by applying the NN compensation at the reference trajectory level instead of at the joint torque/force level. This approach is

This paper was recomended for publication by S. Caraman

very attractive in practice because it can be incorporated in the trajectory planner of any existing robot control system without having to alter the internal structure of the controller.

2. ROBOT DYNAMIC EQUATIONS IN CARTESIAN SPACE

The relationships between the joint space coordinates q and the end effector Cartesian space coordinates X are

$$\dot{X} = J(q)\dot{q}\ ,\ \ddot{X} = J(q)\ddot{q} + \dot{J}\dot{q} \tag{1}$$

where $J(q)$ is the $n \times n$ nonsingular Jacobian matrix. Thus, the robot dynamic equation in joint space can be expressed as

$$D(q)J^{-1}(q)(\ddot{X} - \dot{J}\dot{q}) + h(q,\dot{q}) + \tau_f(\dot{q}) = \tau \tag{2}$$

Since the end points forces F are related to the joint torques τ by (Fu et al (1987), Megahed 1993, Vukobratovic and Stokic 1989)

$$\tau = J^T F \tag{3}$$

we obtain the Cartesian space robot dynamic equation

$$D^*(X)\ddot{X} + h^*(X,\dot{X}) + F^*_f(\dot{X}) = F \tag{4}$$

where

$D^* = (J^T)^{-1}(q)D(q)J^{-1}(q)$,

$h^* = (J^T)^{-1}h(q,\dot{q}) - D^*(X)\dot{J}J^{-1}\dot{X}$ and

$F^*_f = (J^T)^{-1}\tau_f(\dot{q})$.

The robot dynamic equation (4) represents a highly nonlinear, coupled, multi-input multi-output system. In most practical cases, the functions $D(q)$, $h(q,\dot{q})$ and $\tau_f(q)$ are not exactly known, only the nominal estimates of $\hat{D}(q)$ and $\hat{h}(q,\dot{q})$ are available for controller design. For simplicity, $D^* = D^*(X)$, $h^* = h^*(X,\dot{X})$, $F^*_f = F^*_f(\dot{X})$ are used later on.

3. COMPUTED TORQUE CONTROL IN CARTESIAN SPACE

When using computed-torque control in Cartesian space the control law F is

$$F = \hat{D}^* U + \hat{h}^* \tag{5}$$

and the control input U is given by

$$U = \ddot{X}_d + K_D(\dot{X}_d - \dot{X}) + K_P(X_d - X) \tag{6}$$

where K_P and K_D are $n \times n$ symmetric positive definite desired dumping, and stiffness gain matrices, respectively and X_d is the desired trajectory.

Combining (4), (5) and (6) yields the closed loop tracking error dynamic equation

$$\ddot{E} + K_D\dot{E} + K_P E = (\hat{D}^*)^{-1}[\Delta D^*\ddot{X} + \Delta h^* + F^*_f] \tag{7}$$

where $\Delta D^* = D^* - \hat{D}^*$, $\Delta h^* = h^* - \hat{h}^*$ and $E = X_d - X$. In the ideal case where $\Delta D^* = \Delta h^* = 0$ and $F^*_f = 0$, the closed loop behavior satisfies the second order differential equation

$$\ddot{E} + K_D\dot{E} + K_P E = 0 \tag{8}$$

Since there are always uncertainties in the robot dynamics, the ideal equation (8) can not be achieved in general. Eq. (7), which is degraded and unpredictable govern the actual system performance. Thus, the computed torque based position control in Cartesian space is not robust in practice. To improve performance, NN controller can be introduced to generate additional input to compensate for disturbances due to model uncertainties. These additional compensating inputs can be placed in four different locations (as presented in the next paragraph) and the proposed architecture is shown in Fig. 2.

4. CARTESIAN SPACE NN CONTROLLER SCHEMES

Four different NN controller designs can be proposed, so as to achieve disturbance rejection for an uncertain system (Moise et al 2001). The design perspectives are dependent on the compensated position by NN as shown in Fig. 1 and Fig. 2. There are four locations that can be compensated: first, at torque level τ, second, at force level F, third at control input level U. Finally, compensation at input trajectory level X_d is depicted separately in Fig. 2. By defining a unified training signal all NN control schemes have the same goal of minimizing the same objective functions. The NN outputs Φ of all the schemes try to cancel out the uncertainties caused by inaccurate robot model in the computed-torque controller. It is noted that the NN inputs X can be either $X_d(t)$, $\dot{X}_d(t)$, $\ddot{X}_d(t)$ or the time-delayed values $X_d(t)$, $X_d(t-1)$, $X_d(t-2)$. The latter case is called time-delayed (Jung and Hsia 1994, Jung and Bonitz 1995, Naredra and Parthasarathy 1990). Delay time is chosen as the sampling period of the controller. Here we lay out each scheme analytically.

Scheme 1. Compensating at τ.

The control law of compensating at torque level is

$$\tau(t) = J^T[\hat{D}*U + \hat{h}*] + \Phi_\tau \quad (9)$$

where Φ_τ is output of NN at torque level. Combining with the robot dynamic equation yields the corresponding closed loop error system as

$$v = \ddot{E} + K_D\dot{E} + K_P E = \quad (10)$$
$$(\hat{D}*)^{-1}[\Delta D*\ddot{X} + \Delta h* + F*_f - (J^T)^{-1}\Phi_t]$$

Since the control objective is to generate Φ to reduce v to zero in (10), error signal *v*

$$v = \ddot{E} + K_D\dot{E} + K_P E \quad (11)$$

Is proposed to be used as the new error signal for training the NN in all schemes. The ideal value of Φ_τ at $v = 0$ then is

$$\Phi_\tau = J^T(\Delta D*\ddot{X} + \Delta h* + F*_f) \quad (12)$$

Thus NN is required to learn robot Jacobian and this degrades the performance comparing to other schemes. Clearly, minimizing the error signal *v* allows us to achieve ideal position control directly. Following a similar procedure one can derive the relevant equations for the other three control schemes (Moise et al 2001).

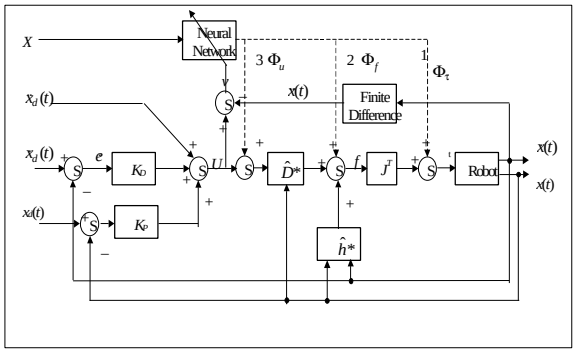

Fig. 1. NN control structure in cartesian space with different locations of compensation

Scheme 2. Compensating at F.

In order to avoid learning kinematic Jacobian, compensating location can be moved from the first level (torque) to the second level (force) in Fig. 1. Similarly, the control law from Fig. 1 and the error signal are derived as

$$\tau(t) = J^T(\hat{D}*U + h* + \Phi_f) \quad (13)$$

$$v = (\hat{D}*)^{-1}(\Delta D*\ddot{X} + \Delta h* + F*_f - \Phi_f) \quad (14)$$

while at convergence, the ideal value for Φ_f is

$$\Phi_f = \Delta D*\ddot{X} + \Delta h* + F*_f \quad (15)$$

which is a simplification of (12). The NN does not need to learn robot Jacobian.

Scheme 3. Compensating at U.

This scheme is to compensate at control input U. Accordingly, the control law and signal v are given by

$$\tau(t) = J^T[\hat{D}*(U + \Phi_u) + \hat{h}*] \quad (16)$$

$$v = (\hat{D}*)^{-1}(\Delta D*\ddot{X} + \Delta h* + F*_f) - \Phi_u \quad (17)$$

while the ideal NN output is

$$\Phi_u = (\hat{D}*)^{-1}(\Delta D*\ddot{X} + \Delta h* + F*_f) \quad (18)$$

Scheme 4. Compensating at X_d.

Finally, for compensating input trajectory, the control law and the error signal v become

$$t(t) = J^T[D*(\ddot{X} + K_D(\dot{X}_d - \dot{X}) \quad (19)$$
$$+ K_P(X_d - X + \Phi_x)) + \hat{h}*]$$

$$v = (\hat{D}*)^{-1}(\Delta D*\ddot{X} + \Delta h* + F*_f) - K_P\Phi_x \quad (20)$$

At $v = 0$, the ideal output of NN is

$$\Phi_x = (K_P)^{-1}(\hat{D}*)^{-1}(\Delta D*\ddot{X} + \Delta h* + F*_f) \quad (21)$$

It can be seen (Moise et al 2001) that Φ_x has the smallest magnitude among all schemes presented above. The performance is also dependent on the feedback gain K_P. Ideally, it is true that $\Phi_u = K_P \cdot \Phi_x$. Thus, the compensating magnitudes of NN output are different from Scheme 1 to Scheme 4 under the same NN structure with Scheme 4 having the smallest. This simply means that Scheme 4 is the best solution in terms of performances since the NN has to emulate a less complex function with smaller range of values. Experimental evaluation of the above schemes, given bellow in section 6, verifies the theoretical justification.

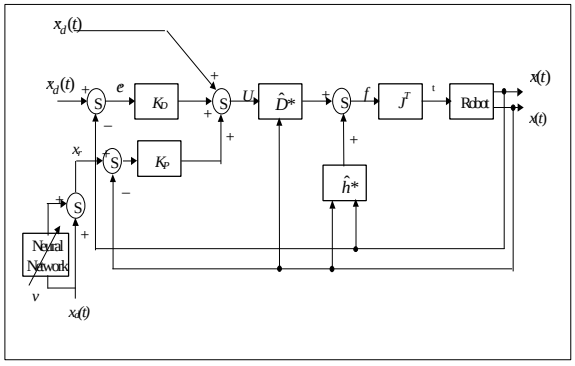

Fig. 2. NN control structure that compensates at input trajectory level..

5. TRAINING OF THE NEURAL NETWORK COMPENSATORS

The weight updating law minimizes the objective function I, which is a quadratic function of the training signal v:

$$I = \frac{1}{2}v^T v \quad (22)$$

Differentiating (22) and using (12) yields the gradient of I for Scheme 1 as follows (Moise et al 2001):

$$\frac{\partial I}{\partial w} = \frac{\partial v^T}{\partial w} v = -\frac{\partial \Phi_\tau^T}{\partial w} (J^T)^{-T} (\hat{D}^*)^{-T} v \quad (23)$$

For other schemes:

Scheme 2	Scheme3
$\frac{\partial v^T}{\partial w} = -\frac{\partial \Phi_f^T}{\partial w} (\hat{D}^*)^{-T}$	$\frac{\partial v^T}{\partial w} = -\frac{\partial \Phi_u^T}{\partial w}$

and $\frac{\partial v^T}{\partial w} = -\frac{\partial \Phi_w^T}{\partial w} K_P^T$ for Scheme 4. Thus, the learning algorithm based on gradient shows a decreasing order of complexity (Moise 2000) from Scheme 1 to Scheme 3 and 4. The back-propagation-updating rule with a momentum term is

$$\Delta w(t) = \eta \frac{\partial v^T}{\partial w} v + \alpha \Delta w(t-1) \quad (24)$$

where η is the update rate and α is the momentum coefficient.

6. EXPERIMENTAL EVALUATION

In this section we present some simulation results to evaluate the performance of the above-presented NN compensation schemes. For simplicity, a two-link revolute planar manipulator was selected having link lengths $\ell_1 = \ell_2 = 1m$ and link masses $m_1 = 0.8kg$, $m_2 = 2.4kg$. Figure 4 shows the desired manipulator's path in 2-D Cartesian space. This path was artificially computed by specifying the desired joint variables and then using the forward kinematic equations of the 2-link planar manipulator to obtain the desired Cartesian path. In Figure 3 (+) denotes the beginning and (*) the end of the path. The joint variables are computed by using the following formula

$$q_1 = 0.5 + 0.2\sin(0.25pt) \qquad t = 0, \cdots, 800$$
$$q_2 = 0.5 + 0.2\sin(0.4pt)$$

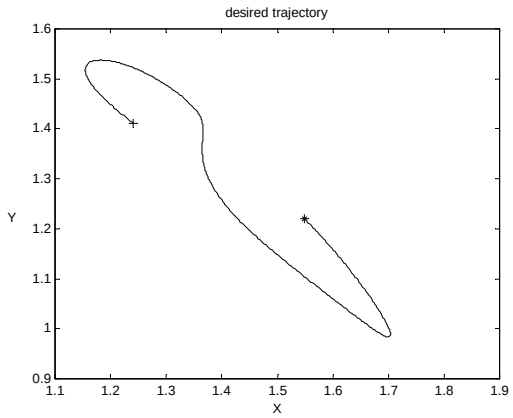

Fig. 3. The desired trajectory in Cartesian space

The PD controller gains are selected as $K_D = diag[20,20]$ and $K_P = diag[100,100]$ which give identical critically damped motions at the two axes. We assume, for simplicity, that our uncertainty about the model is reduced to incomplete knowledge of the exact values of the masses. More specifically we assume that the controller operates based on false values of the masses, which are 10% different from the real values. Figure 4 shows the performance of the conventional PD controller. The initial position of the end point of the manipulator is assumed to be far from the starting point of the desired path. The end point approaches the desired path, but, due to the uncertainties, it fails to follow the desired path with a satisfactory precision. For clearness, only the first parts of the trajectories are shown.

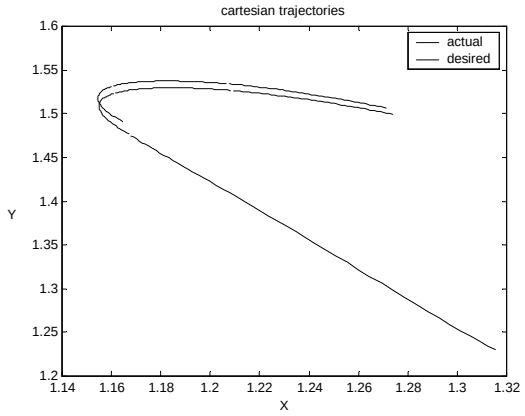

Fig. 4. Tracking of the desired trajectory by the conventional PD controller.

The performances of the various compensation schemes are then examined. For each one of the NN compensators, we have chosen the same thee layer (6-8-2) neural network (the number of hidden neurons, $n_H = 8$). The back-propagation algorithm parameters are: $w_{ij}^k(0) = 0$ (k refers to the output layer) and $\alpha = 0.9$. Fig. 5(a-d) shows the tracking of the desired trajectory by using the four NN control schemes. First, we observe that all of the schemes succeed in compensating for the existing uncertainties, therefore they perform much better than the conventional PD controller. It is also evident that the speed of getting near to the desired trajectory is faster when going from scheme 1 to scheme 4, with schemes 3 and 4 having almost identical performance. This implies that the NN presents faster its desired performance when going from scheme 1 to scheme 4. This is a direct consequence of the fact that the same NN has to learn less complex functions when going from scheme 1 to scheme 4. The same fact is also evident in Fig. 6, where the error signal v is displayed for the four different schemes. The experimental results fully comply with the theoretical expectations. The very significant performance improvement, which is obtainable by applying the NN compensation at the reference trajectory level instead of at the joint torque/force level is very useful for practical applications. Using this scheme, the NN

compensator can be incorporated in the trajectory planner of any existing robot control system without having to alter the internal structure of the controller.
It has to be noticed here that in scheme 4, in order to avoid instabilities during the training of the NN, instead of using K_p in (24) we are using the modified $K_p' = 0.01K_p$

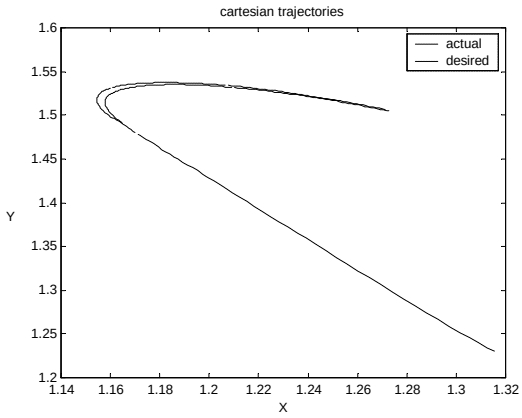

Fig. 5a. Tracking of the desired trajectory using the NN controller of scheme 1 (torque compensation).

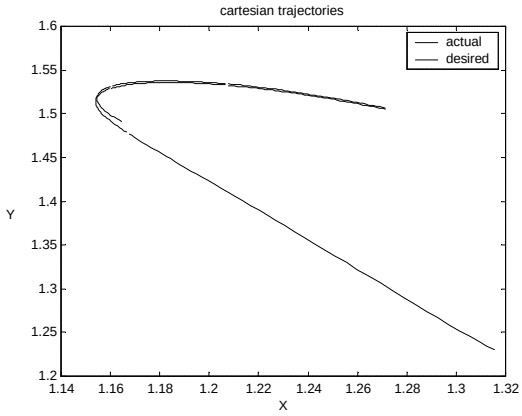

Fig. 5b. Tracking of the desired trajectory using the NN controller of scheme 2 (compensation at force).

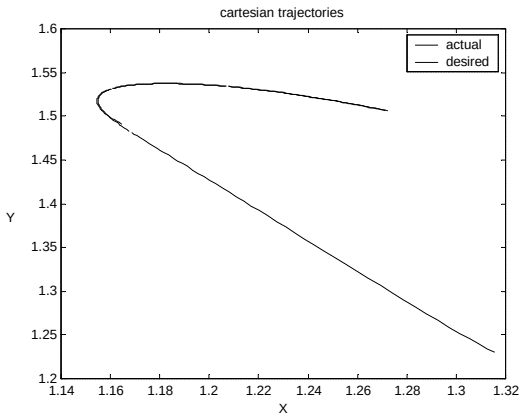

Fig. 5c. Tracking of the desired trajectory using the NN controller of scheme 3 (compensation at U).

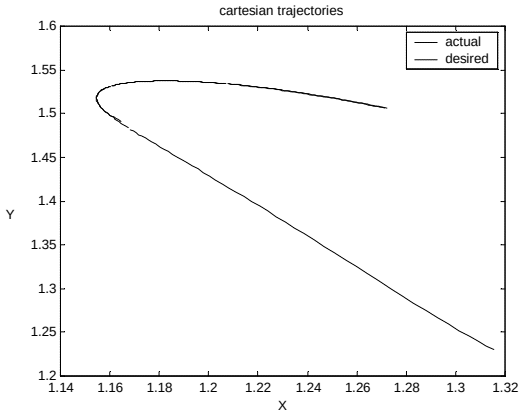

Fig. 5d. Tracking of the desired trajectory using the NN controller of scheme 4 (compensation at X_d).

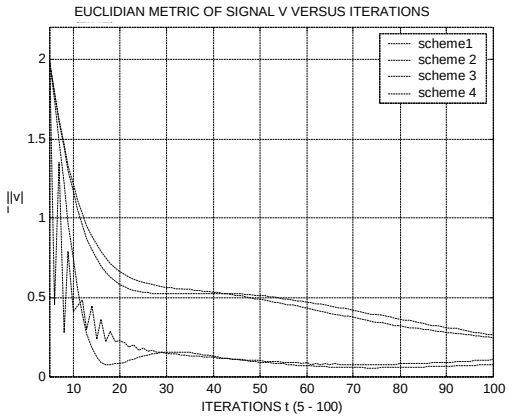

Fig. 6. Training signal v versus iterations (t) for the four NN compensated controllers.

7. CONCLUSION

In this paper the problem of Cartesian control of uncertain manipulators using neural networks is examine. We present and compare four different NN compensation schemes to achieve disturbance rejection for an uncertain system. It is shown that, although NN compensation for model uncertainties has been traditionally carried out by modifying the joint torque/force of the robot, it is also possible to achieve the same objective by using the NN to modify the reference Cartesian trajectory. We show that, for the same neural network, the required internal signal level for the trajectory modification approach is much smaller. Thus we are able to theoretically justify that very significant performance improvement is obtainable by applying the NN compensation at the reference trajectory level instead of at the joint torque/force level. Simulation results fully support this conclusion. This approach is very attractive in practice because it can be incorporated

in the trajectory planner of any existing robot control system without having to alter the internal structure of the controller.

REFERENCES

Boutalis, Y. S., Karras, D. A., Mertzios, B. G. (2000) Regularized supervised and unsupervised neural network models of function approximation in intelligent robot dynamic control, *Bulletin of Oil & Gas University of Ploiesti*, **vol. LII**, no. 1/2000, .99-104.

Fu, K. S., Gonzales, R. C., Lee, C. S. G. (1987) *Robotics. Control, Sensing, Vision and Intelligence*, McGraw-Hill.

Isiguro, A., Furuhashii, T., Okuma, S., Uchikawa, Y., (1992). A neural network compensator for uncertainties of robot manipulator, *IEEE Trans. on Industrial Electronics*, **39**,.61-66.

Jung, S. Hsia, T. C.,(1994), On-line neural network control of manipulators, *Proc. of International Conference on Neural Information Processing*, Seoul, 1663-1668.

Jung, S., Hsia, T. C., Bonitz, R. G., (1995), On force tracking impedance control with unknown environment stiffness, *Proc. of IASTED International Conference on Robotics and Manufacturing, Cancun*, 181-184.

Megahed, S. M., (1993), *Principles of robot modelling and simulation*, John Wiley & Sons, Chichester.

Moise, A. G. V., (2000) Mobile robot navigation using the mathematical model of the sensing system, *Proc. of 9th International Workshop on Robotics in Alpe-Adria-Danube Region, RAAD 2000, Maribor, Slovenia*, 53-58.

Moise, A., Boutalis Y. S., and Mertzios B. G., (2001) Cartesian Space Neural Network Control of Robot Manipulators, *European Workshop on Intelligent Forecasting, Diagnosis & Control (IFDICON-2001), Island of Santorini – Greece, June 25-27*.

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

Tap, J. M., Luh, J. Y. S., (1993) Application of neural Network with real-time training to robust position/force control of multiple robots, *Proc. of the IEEE International Conference on Robotics and Automation,* 142-148.

Vukobratovic, M., Stokic, (1989), D. *Applied Control of Manipulation Robots*, Springer Verlag, Berlin.