2002 FIRST INTERNATIONAL IEEE SYMPOSIUM "INTELLIGENT SYSTEMS", SEPTEMBER 2002

Neural Network Identification, Predictive Modeling and Control with a Sliding Mode Learning Mechanism: an Application to the Robotic Manipulators

Andon V. Topalov, Okyay Kaynak and Nikola G. Shakev

***Abstract*—The features of a novel adaptive PID-like neurocontrol scheme for nonlinear plants are presented. The controller tuning is based on an estimate of the command-error determined via one-step-ahead neural predictive model of the plant. An on-line learning sliding mode algorithm is applied to the model and to the controller as well. The control architecture developed has been simulated and its effect on the trajectory tracking performance of a simple two-degree-of-freedom robot manipulator has been evaluated. The results show that both learning structures, the neural predictive model and the controller, inherit some of the advantages of SMC: high speed of learning and robustness.**

***Index Terms*—Neural Networks, Modeling, Identification, Sliding Mode Control, Adaptive Control**

I. Introduction

In contemporary robotic systems, there is a need for more flexible and robust robot controllers in order to take full advantage of the inherent flexibility and versatility of manipulation robots. Due to the highly nonlinear and coupled dynamics of manipulators and the often unknown inertial properties of the objects being manipulated, accurate trajectory tracking is difficult to obtain. Conventional non-adaptive control algorithms are not robust enough, because they compensate only a small number of the uncertainties noted and the tracking performance obtained is generally poor. Hence, a more suitable approach would be the one using adaptive control techniques. Conventional adaptive control techniques in robotics are efficient in the case of compensation of structured uncertainties only. A solution to the robot control problem requires combining conventional approaches with new learning approaches in order to achieve good performance. Thus the need to meet demanding control requirement leads toward design of intelligent manipulation robots [1]. Connectionist methods with distributed processing provide the implementation tools for modeling the complex input/output relations of robot's dynamics and kinematics. The connectionist approach may, in principle solve the problem of variable - coupling complexity and state - dependency of robot dynamic model, because neural networks through the process of training can approximate input-output mappings. In this way connectionist structure as part of decentralized feedback control law can compensate wide range of robot uncertainties. Several neural network models and learning schemes were applied to learning of robot dynamics, recently and various neuro-controllers could be trained in a way that the desired plant output is attained as much as possible [2] – [3].

Another approach that has received much attention is the intelligent tuning of conventional controllers using neural networks [4], [5] – [7]. Efficient implementation of a neural network-based strategy for the on-line adaptive control of such systems centers on the rapidity of the convergence of the training scheme used for learning the system dynamics. In this paper a novel neuro-adaptive PID control scheme based on a neural predictive model with a fast on-line sliding mode learning mechanism is presented and a trajectory tracking control of a robot manipulator is simulated.

The main body of the paper contains six sections. Section 2 presents the analytical model of the robot manipulator dynamics. Section 3 describes the proposed neuro-adaptive control scheme. Section 4 presents the applied sliding mode learning algorithm. Simulation results are shown in Section 5. Finally, section 6 summarizes the findings of this investigation.

II. The Analytical Model of the Robot Manipulator Dynamics

The dynamic model for a general n-degree-of-freedom rigid manipulator in the absence of friction and other disturbances (deterministic model) can be written in the form

$$T = f(q,\dot{q},\ddot{q},\varphi) = M(q,\varphi)\ddot{q} + N(q,\dot{q},\varphi) \qquad (1)$$

The authors would like to acknowledge the fellowship provided to the first author by The Scientific and Technological Research Council of Turkey within the framework of the NATO-PC Advanced Fellowship Program..

A. V. Topalov is with the Control Systems Department, Technical University of Sofia, Plovdiv branch, 61, Sankt Petersburg Blvd., 4000 Plovdiv, Bulgaria (telephone: +359-32-659-528, e-mail: topalov@mbox.digsys.bg).

O. Kaynak is with the Department of Electrical and Electronic Engineering, Bogazici University, 80815 Bebek, Istanbul, Turkey (telephone: +90-212-287-2475, fax: +90-212-287-2465, e-mail: kaynak@boun.edu.tr).

N. G. Shakev is with the Control Systems Department, Technical University of Sofia, Plovdiv branch, 61, Sankt Petersburg Blvd., 4000 Plovdiv, Bulgaria (telephone: +359-32-659-528, e-mail: nikolashakev@yahoo.com).

where $T \in R^n$ - is the vector of driving torques or forces, $M(q,\varphi): R^n \times \varphi \Rightarrow R^{n \times n}$ is the inertial matrix of the system; $N(q,\dot{q},\varphi): R^n \times R^n \times \varphi \Rightarrow R^n$ is vector which includes Centrifugal, Coriolis and gravitational effects; $\varphi \in R^{nt}$ is the system parameter vector; n - is the number of degree of freedom; nt - is the number of system parameters.

Discretization of the non-linear dynamics for purposes of controller design has been well studied and some models that ensure the conversion of energy and momentum are developed [8]. Typically used discretized models for the i-th joint dynamics are given by the following equation, [9]:

$$q_i(k+1) = \Phi_i[q_a(k), \dot{q}_b(k), u_a(k)] + \Psi_i[q_c(k), \dot{q}_e(k), u_c(k)] \quad (2)$$

where $q_a(k) = [q(k), q(k-1) \ldots q(k-m_1)]^T$,

$$\dot{q}_b(k) = [\dot{q}(k), \dot{q}(k-1) \ldots \dot{q}(k-m_2)]^T,$$
$$u_a(k) = [u(k), u(k-1), u(k-2) \ldots u(k-m_3)]^T,$$
$$q_c(k) = [q(k), q(k-1) \ldots q(k-m_4)]^T,$$
$$\dot{q}_e(k) = [\dot{q}(k), \dot{q}(k-1) \ldots \dot{q}(k-m_5)]^T,$$
$$u_c(k) = [u(k), u(k-1) \ldots u(k-m_6)]^T,$$

and $\Phi_i(\cdot)$ and $\Psi_i(\cdot)$ are continuous non-linear functions of the arguments u, q and $\dot{q}$. m_1, m_2, m_3, m_4, m_5 and m_6 are appropriate positive integers. The subscripts a, b, c and e for $q, \dot{q}$ and u in (2) are used to define at instant k vectors of joint positions, joint velocities, and the control functions (forces and/or torques) constructed from past data of different lengths.

III. The Neuro-Adaptive Control Scheme

The control strategy designed is a decentralized one that permits implementation of independent joint controls. The controller tuning is based on an estimate of the command error on its output. The latter is determined by calculating the Jacobians and the one-step-ahead predicted trajectory error for each joint using a neural predictive model. A robust on-line learning algorithm, based on the direct use of sliding mode control (SMC) theory is applied to both: to the controller and to the model as well.

Let us consider that the required control $u_i(k)$ for i-th robot joint at time instant k will be computed using PID control law as follows:

$$u_i(k) = K_{1i}(k)\varepsilon_i(k) + K_{2i}(k)\sum_{l=1}^{k}\varepsilon_i(l) + K_{3i}(k)(\varepsilon_i(k) - \varepsilon_i(k-1)) \quad (3)$$

where $K_{si}(k)$, $s = 1,2,3$ are the controller parameters for i-th robot joint at time instant k, $\varepsilon_i(k) = q_i(k) - q_{di}(k)$ is the feedback error, and $q_i(k)$, and $q_{di}(k)$ are the real and desired coordinates for i-th robot joint at time instant k, respectively.

The proposed neuro-adaptive control architecture is presented on Fig. 1. The major concern is the application of neural networks in robot control at executive hierarchical level (motion control problem) as one-step-ahead predictors in the case where exact robot dynamics is unknown. The problem of interest is to evaluate the controls $u_i(k)$, $i = 1,2,\ldots,n$ to be applied to the individual joints such that a desired tracking motion of each joint specified by the trajectories $q_{di}(k)$, $i = 1,2,\ldots,n$ ensues.

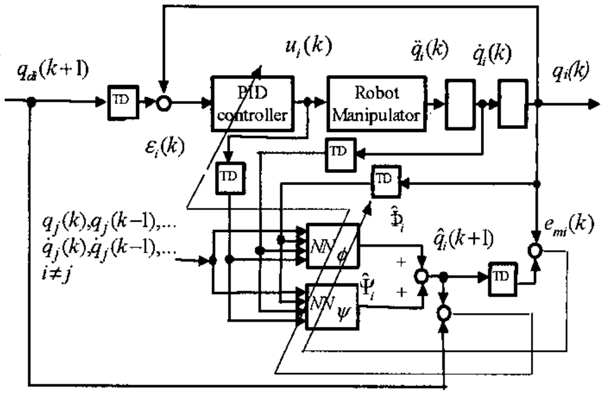

Fig. 1. The neuro-adaptive control architecture

For implementing an adaptive control that can deliver satisfactory performance, parameter estimation followed by updating of control is needed at each adaptation step. Since the functions $\Phi_i(.)$ and $\Psi_i(.)$ in (2) are not known a priori, a neural network-based approach for their identification at each time step k is implemented in order to facilitate adjustment of the controller gains and the computation of the required controls. Two two-layered feedforward neural networks with identical structure are used for on-line identification of functions $\Phi_i(\cdot)$ and $\Psi_i(\cdot)$ in the manipulator dynamics and for one-step-ahead prediction of the coordinates of the i-th robot joint by using the following model (see Fig. 1):

$$\hat{q}_i(k+1) = \hat{\Phi}_i[q_a(k), \dot{q}_b(k), u_a(k), W_\Phi] + \hat{\Psi}_i[q_c(k), \dot{q}_e(k), u_c(k), W_\Psi] \quad (4)$$

where $\hat{q}_i(k+1)$ is the one-step-ahead predicted coordinate for i-th robot joint, obtained from the connectionist joint model.

A weight-updating rule based on the direct use of SMC strategy is implemented and training is continued on-line during the execution of robot motion. The model error that is to be minimized is defined as follows:

$$e_{mi}(k) = q_i(k) - \hat{q}_i(k) \quad (5)$$

The input signals to the neural network model during the identification and control phases in the present scheme remain the same, which offers considerable implementation benefits.

It is possible to deduce the rule for parameters adjustment of the controller through the minimization of the following performance criterion:

$$J_i(k) = 1/2[q_{di}(k+1) - \hat{q}_i(k+1)]^2 \quad (6)$$

In the above $q_{di}(k+1)$ is the desired joint coordinate at time instant $k+1$.

In the approach adopted in this paper, an estimate of the command-error for i-th robot joint $e_{ui}(k+1)$, named "virtual" command-error, is obtained as in (7):

$$e_{ui}(k+1)=\frac{\partial J_i(k)}{\partial u_i(k)}=-[q_{di}(k+1)-\hat{q}_i(k+1)]\frac{\partial \hat{q}_i(k+1)}{\partial u_i(k)} \tag{7}$$

The connectionist model of the i-th robot joint in accordance with (4) can be expressed as follows:

$$\hat{q}_i(k+1)=\sum_{l=1}^{S1} W2_{li}^{\Phi}(k) f_l\left(\sum_{r=1}^{P} W1_{lri}^{\Phi}(k) A_{ri}^{\Phi}(k)\right)+ + \sum_{l=1}^{S1} W2_{li}^{\Psi}(k) f_l\left(\sum_{r=1}^{P} W1_{lri}^{\Psi}(k) A_{ri}^{\Psi}(k)\right) \tag{8}$$

where $S1$ is the number of neurons in the hidden layer, P is the number of the inputs of the network, $W1_{lri}^{\Phi}$ and $W1_{lri}^{\Psi}$ are the weights of the l-th neuron in the hidden layer from its r-th input for the first and second neural network of the i-th robot joint respectively, $W2_{li}^{\Phi}$ and $W2_{li}^{\Psi}$ are the weights of the neurons in the output layers, and $A_i^{\Phi}(k)=[q_a(k),\dot{q}_b(k),u_a(k)]$ and $A_i^{\Psi}(k)=[q_c(k),\dot{q}_e(k),u_c(k)]$ are data arrays with length P, presented to the inputs of the two networks respectively at time instant k, and composed in accordance with (2), $f(\cdot)$ is a log-sigmoid function.

The $\dfrac{\partial \hat{q}_i(k+1)}{\partial u_i}$ can be determined as in (9):

$$\frac{\partial \hat{q}_i(k+1)}{\partial u_i(k)}=\left(\frac{\partial \hat{\Phi}_i(\cdot)}{\partial u_i(k)}+\frac{\partial \hat{\Psi}_i(\cdot)}{\partial u_i(k)}\right) \tag{9}$$

Let $X_{li}^{\Phi}(k)=\sum_{r=1}^{P} W1_{lri}^{\Phi}(k) A_{ri}^{\Phi}(k)$ then

$$\frac{\partial \hat{\Phi}_i(\cdot)}{u_i(k)}=\sum_{l=1}^{S1} W2_{li}^{\Phi}(k) f_l(1-f_l)\frac{\partial X_{li}^{\Phi}(k)}{\partial A_{ri}^{\Phi}(k)}\cdot\frac{\partial A_{ri}^{\Phi}(k)}{\partial u_i(k)} \tag{10}$$

where $\dfrac{\partial X_{li}^{\Phi}(k)}{\partial A_{ri}^{\Phi}(k)}\cdot\dfrac{\partial A_{ri}^{\Phi}(k)}{\partial u_i(k)}=\sum_{r=1}^{P} W1_{lri}^{\Phi}(k)\dfrac{\partial A_{ri}^{\Phi}(k)}{\partial u_i(k)}$ and $f_{li}=\dfrac{1}{1+e^{-X_{li}^{\Phi}(k)}}$.

In the same way it can be proven that

$$\frac{\partial \hat{\Psi}_i(\cdot)}{\partial u_i(k)}=\sum_{l=1}^{S1} W2_{li}^{\Psi}(k) f_{li}(1-f_{li})\sum_{r=1}^{P} W1_{lri}^{\Psi}(k)\frac{\partial A_{ri}^{\Psi}(k)}{\partial u_i(k)} \tag{11}$$

where $f_{li}=\dfrac{1}{1+e^{-X_{li}^{\Psi}(k)}}$.

In that particular case when $A_{ri}^{\Phi}(k)=u_i(k)$ and $A_{ri}^{\Psi}(k)=u_i(k)$, $\dfrac{\partial A_{ri}^{\Phi}(k)}{\partial u_i(k)}=1$ and $\dfrac{\partial A_{ri}^{\Psi}(k)}{\partial u_i(k)}=1$. In all other cases an approximated value $\dfrac{A_{ri}^{\Phi}(k)-A_{ri}^{\Phi}(k-1)}{u_i(k)-u_i(k-1)}$, or $\dfrac{A_{ri}^{\Psi}(k)-A_{ri}^{\Psi}(k-1)}{u_i(k)-u_i(k-1)}$ respectively can be used.

IV. The Sliding Mode Learning Algorithm

In the theory of control engineering, one way of designing a robust and stable control system is to use the Variable Structure Systems (VSS) approach, which enables the designer to come up with a rigorous stability analysis. It is a well-known fact that a variable structure controller with a switching output will (under certain circumstances) result in a sliding mode on a predefined subspace of the state space. This mode has useful invariance properties in the face of uncertainties in the plant model and therefore is a good candidate for tracking control of uncertain nonlinear systems.

The studies demonstrating the high performance of the variable structure control in handling the uncertainties and imprecision have motivated the use of sliding mode control scheme in training of ANN. The results presented in [10] have shown that the convergence properties of the gradient-based training strategies can be improved by utilizing the SMC scheme. The method presented can be considered as an indirect use of VSS theory. Some studies on the direct use of SMC strategy are also reported in the literature [11] – [12]. In another paper [13], the existence of a relation between sliding surface for the plant to be controlled and the zero learning error level of the parameters of a flexible controller is discussed and the control applications of the method considered in [11] – [12] are studied with constant uncertainty bounds.

Recently, a new learning algorithm for training multilayer artificial neural networks, based on direct use of SMC strategy, was proposed by G. G. Parma et al. [14] – [15]. Its on-line version presented in [14] has been initially tested for the identification of a periodic time signal. In addition to the applicability of the newly proposed algorithm for updating the weights in the hidden layers of multilayer network structures it also differs from the algorithms in [11] – [12] due to the definition of the sliding surface which is now determined by taking not only the learning error variable, but also its time derivative. The latter contributes very much to the fast convergence capability of the algorithm. This property is crucial for on-line learning in applications demanding adaptation to constantly changing environmental parameters, such as adaptive real-time control.

In this work the algorithm proposed in [14] is applied to the both learning structures: the neural predictive model and the controller.

The design is divided into two phases. In the first, sliding surfaces to produce the input/output behavior are defined. In the second, the weights are updated in order to satisfy the conditions for tracking and sliding on the surfaces.

Consider the virtual errors e_{ui}, by adding to these terms the terms ρu_i, $(\rho \geq 0)$, respectively, for penalizing the controls u_i a simple type of optimal control criterion is achieved.

$$e_{ui}^* = e_{ui} + \rho u_i \tag{12}$$

As ρ is increased the control signal will become smoother and will attain smaller values.

For the output layer of the i-th joint neural predictive model and for the controller output respectively, the following sliding surfaces are then defined:

$$S_{mi} = \dot{e}_{mi} + \lambda_{mi} e_{mi}\ ;\ S_{ci} = \dot{e}_{ui}^* + \lambda_{ui} e_{ui}^* \tag{13}$$

where λ_{mi}, $\lambda_{ui} > 0$ are constants.

For the hidden layer of the i-th joint model, the sliding surface is defined as in (14):

$$S_{Hi} = \dot{e}_{Hi} + \lambda_{Hi} e_{Hi} \tag{14}$$

where $e_{Hi} = \frac{1}{2} e_{mi}^2$ and $\lambda_{Hi} > 0$, [14].

Therefore, the weight update rules for the output layer of the i-th joint model and for the controller can be defined as in (15) and (16) respectively (for $\alpha_{mi} > 0$ and $\alpha_{ui} > 0$), [14]:

$$\dot{W}2_{li} = \alpha_{mi} sign(S_{mi}) |e_{mi}| f_{li} \tag{15}$$

$$\dot{K}_{si} = \alpha_{ui} sign(S_{ci}) |e_{ui}^*| h_{si} \tag{16}$$

The weight update rule for the hidden layer of the neural model is defined as in (17) (for $\beta_{mi} > 0$):

$$\dot{W}1_{lri} = \frac{\beta_{mi} sign(S_{Hi}) |e_{Hi}|}{(e_{mi} W2_{li} + \eta)(1 - f_{li}) f_{li}} A_{ri} \tag{17}$$

where η is a small constant introduced to avoid $\dot{W}1_{lri} \to \infty$ when $e_{mi} \to 0$.

A natural solution to avoid the chattering behavior (which is a well known problem associated with SMC) is to smooth the discontinuity in the signum function in (15) – (17). One possible solution is to use a sigmoid-like function.

$$v_\delta(S) = \frac{S}{|S| + \delta} \tag{18}$$

where δ is a small positive scalar.

The conditions for the analysis of the algorithm from the sliding mode point of view are:

(i) r and y are limited with limited derivatives;

(ii) $f_H(\cdot)$ is differentiable and limited.

For the learning structures described, these conditions are obviously fulfilled.

Using the well-known conditions for sliding mode regime [16], the bounds for the constants λ_{mi}, λ_{ui} and λ_{Hi} that appear in (13) and (14) are obtained as:

$$\lambda_{mi} \geq \max\left\{ -\frac{2}{f_{li}} \cdot \frac{\partial f_{li}}{\partial t} - \frac{|\dot{e}_{mi}|}{|e_{mi}|} \right\} \tag{19}$$

$$\lambda_{ui} \geq \max\left\{ -\frac{|\dot{e}_{ui}^*|}{|e_{ui}^*|} \right\} \tag{20}$$

$$\lambda_{Hi} \geq \max\left\{ -\frac{2}{A_{ri}} \cdot \frac{\partial A_{ri}}{\partial t} - \frac{|\dot{e}_{Hi}|}{|e_{Hi}|} \right\} \tag{21}$$

The ideal situation would be to have always high values of λ_{mi}, λ_{ui} and λ_{Hi} to guarantee fast convergence, but there is in fact a trade-off between the values of λ_{mi}, λ_{ui} and λ_{Hi}, and the relevant values of α_{mi}, α_{ui} and β_{mi}, which by their turn, depend also on the amplitude variations of the system parameters. Therefore, for every system to be controlled, the bounds for α_{mi}, α_{ui} and β_{mi} (and λ_{mi}, λ_{ui} and λ_{Hi}) should be derived in advance in order to predict convergence and stability properties.

The upper limits of α_{mi}, α_{ui} and β_{mi} can be obtained from the sliding surface expression. For a given S, delimited by the training data and network topology, the upper limits for the gains, or learning rates, α_{mi}, α_{ui} and β_{mi} can be easily obtained. According to Utkin [16], the condition for existence of sliding mode and system stability is defined by (22). Since the adaptation of the weights of the learning network structures, which could be interpreted as control signals with regard to this structures, is a function of α_{mi}, α_{ui} and β_{mi}, the analysis of (22) results in their limit values.

$$S\frac{dS}{dt} \leq 0 \tag{22}$$

Convergence is guaranteed for any values of α_{mi}, α_{ui} and β_{mi}, within the limits established by S, since it implies on the existence of sliding mode. For discrete time, where Sarpturk *et al.* [17] defined the condition $|S(k+1)| < |S(k)|$, instead of (22), as a condition to guarantee the sliding manifold, a way to define the limits for gains α_{mi}, α_{ui} and β_{mi} is presented in [15].

With the surfaces defined as in (13) – (14), the weight updating rules defined as in (15) – (17), λ_{mi}, λ_{ui} and λ_{Hi} satisfying the boundary conditions ((19) – (21)), it can be affirmed that the model error e_{mi} and the "virtual" command-error e_{ui} tend to zero [14].

V. Simulation Experiments

The tracking performance and the adaptation capabilities of the presently developed neuro-adaptive PID control scheme have been evaluated by conducting several simulation experiments using a simple two-link manipulator shown in Fig. 2. The manipulator was modeled as two rigid links of lengths $l_1 = 0.5$ m and $l_2 = 0.5$m with point masses $m_1 = 10$ kg and $m_2 = 8$ kg at the distal ends of the links. The dynamic equations of the manipulator can be found in [18]. The simulations were carried out using a fourth order Runge-Kutta algorithm with a step size 0.002 sec.

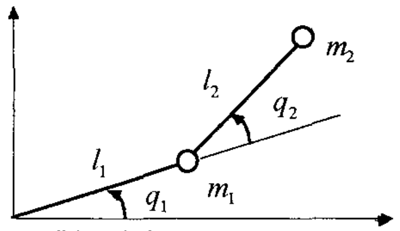

Fig. 2. A two-link manipulator

A discrete-time representation of the joint dynamics in the form of (2) results in the values for delay parameters m_1, m_2, m_3, m_4, m_5 and m_6. The results obtained in [9] were accepted here and the following values for delay parameters were applied for both robot joints: $m_1 = 2$; $m_2 = 2$; $m_3 = 0$; $m_4 = 2$; $m_5 = 2$ and $m_6 = 1$. They were used to form the data arrays $A_i(k)$ and $B_i(k)$, presented to the inputs of the networks. The number of nodes in the hidden layer of the neural networks used for identification of the unknown functions $\Phi_i(\cdot)$ and $\Psi_i(\cdot)$, $i = 1,2$, was set to 10. The accepted values of the different coefficients and constants during the simulations were: λ_{mi}, λ_{ui} and $\lambda_{Hi} = 1$; α_{mi}, $\beta_{mi} = 5.10^{-4}$; $\alpha_{ui} = 8.10^{-3}$; $\delta = 1.10^{-2}$ and $\eta = 1.10^{-8}$.

A component was included in the control law that takes over from the controller adaptation as its approximating ability begins to degrade (early stages of learning when the initial network approximations may be quite poor). So adaptation mechanism was stopped when $|q_{di}(k+1) - \hat{q}_i(k+1)| > 0.01$ or $|\varepsilon_i(k)| > 0.001$.

The overall algorithm was simulated and tested for trajectory tracking control tasks with respect to three criteria: convergence properties of the proposed controller, adaptation capability of the algorithm for sudden changes in manipulator dynamics and generalization properties of the proposed scheme.

For evaluating the performance of the control scheme in executing a typical trajectory task, a 4.8 s motion was considered during which the manipulator arm was required to follow the reference trajectories q_{d1} and q_{d2}. The desired joint position trajectories were chosen as:

$$q_{d1} = -0.77 + 0.8\sin((2\pi t/4.8) - \pi/2);$$
$$q_{d2} = -0.8 - 0.8\sin((2\pi t/4.8) - \pi/2) \qquad (23)$$

The results are presented on Fig. 3. The following denotations are used for figures 3, 4 and 5: Solid lines trajectories are actual trajectories. Desired trajectories are plotted with dashed lines and neural model outputs - with dash-doted lines. Position errors for joint 1 are plotted with solid line while for joint 2 - with a dashed line

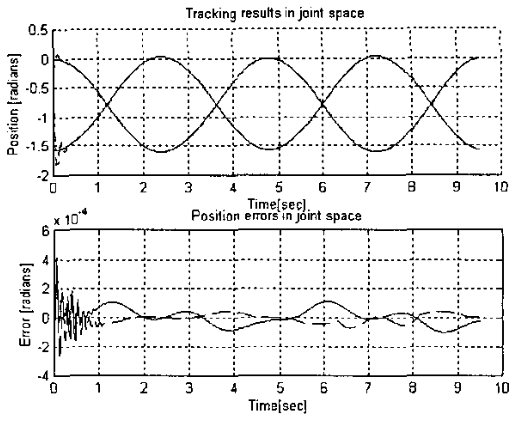

Fig. 3. Tracking performance of proposed neuro-adaptive PID control.

For testing the rate of convergence, an initial mismatch in the positions of both joints was included (0.3 rad for the first joint and -0.3 rad for the second joint). The results are presented on Fig. 4. It can be seen that after the first 0.5 s interval, the joint outputs closely follow (indicated in solid lines on Fig. 4) the required trajectories demonstrating a good tracking performance of the control scheme.

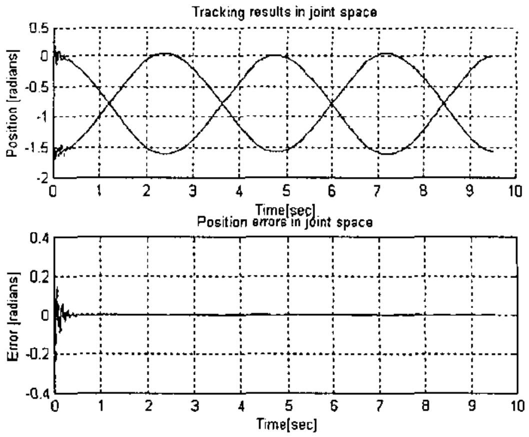

Fig. 4. Tracking performance under initial mismatch 0.3 rad in the positions of joint 1 and joint 2.

When the manipulator grasps an unknown object to move to a new position, there is a discontinuity in inertia and gravity forces at the moment the object is grasped. For ensuring a robust performance the control scheme should adapt very quickly to the new dynamical parameters.

In order to demonstrate the adaptation capability of the control scheme to execute desired motions in the face of unforeseen parameter variations, an example of pick and place task, commonly performed by industrial manipulators was carried out. In this example the manipulator was required during the following specified 4.8 s trajectory at time 2 s to pick up an unknown load. The pay-load is considered as an increasing of the point mass of the second link from 8 kg to 16 kg. In order to present greater challenges to the control scheme additional dynamical phenomenon was introduced. During the motion, the load carried out was "accidentally" dropped at 4.2 s. The performance delivered by the present controller under these conditions is depicted in Fig. 5, where the joint output profiles are indicated in solid lines while tracking the reference trajectories indicated in dashed lines. A rapid damping of the oscillations at the points of discontinuity and a quick return to following the reference motions can be noticed.

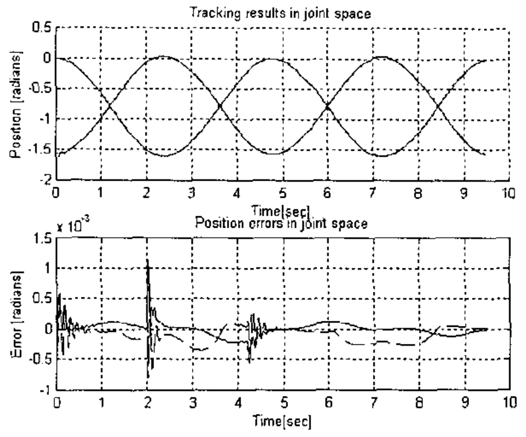

Fig. 6. Tracking performance under parameter variations (during the forward motion at time 2 s the mass of the joint 2 was doubled to simulate object grasping while during the return motion at time 4.2 s the load carried was "accidentally" dropped).

The good performance resulting even under the changed conditions underscores the fast adaptation and the robustness of the proposed neuro-adaptive controller.

VI. Conclusion

The proposed method enables to understand a physical meaning of the control parameters, and also to adjust controller's gains quickly. The use of neural networks for identifying the functions and from the input-output data not only facilitates the adaptation of controller parameters to the changes in manipulator dynamics and/or external conditions, but also eliminates problems associated with unmodeled dynamics. In contrast, it may be noted that the popular model-based approaches for robot control (such as the computed torque technique [18]) can result in poor control performance if the specific model structure selected (on which the controller design is based) does not completely reflect all of the existing dynamics. Training by proposed connectionist structure is accomplished exclusively in on-line regime with a variable structure systems theory based learning mechanism. The updating of the neural network weights during the control phase is useful in improving the tracking performance in the face of changes in system dynamics and noisy measurements. In the proposed control scheme the neural networks adapt to these changed conditions in a relatively fast manner. This control structure provides an internal teacher so that the control scheme works in an unsupervised manner, because we have no external teacher in this case.

The control architecture developed has been simulated and its effect on the trajectory tracking performance of a simple two-degree-of-freedom robot manipulator has been evaluated (convergence speed in identification and tracking accuracy). The adaptation capabilities of the controller are analyzed using different trajectory tracking scenarios. These experiments confirm that the neural network based adaptive PID control strategy proposed is effective in the trajectory tracking tasks.

References

[1] D. M. Katic and M. K. Vukobratovic, "Highly efficient robot dynamics learning by decomposed connectionist feedforward control structure," *IEEE Trans. on Syst., Man, and Cybern.*, 25, No. 1, pp. 145-158, 1995.

[2] W. T. Miller III, R. S. Sutton, and P. J. Werbos, *Neural Networks for Control.* MIT Press, 1992.

[3] D. A. White and D. A. Sofge, Edited, *Handbook of Intelligent Control - Neural, Fuzzy, and Adaptive Approaches.* Van Nostrand Reinhold, 1992.

[4] S. Omatu, "Dynamical systems regulation by neuro-controllers," *Proc. of IEEE Int. Conf. on Neural Networks*, pp.2912-2916, Perth, Nov.-Dec., 1995.

[5] T. Yamamoto, M. Kaneda, T. Oki, E. Watanabe, and K. Tanaka, "Intelligent tuning PID controllers," *Proc. of IEEE Int. Conf. on Syst., Man, and Cybern.*, pp. 2610-2615, 1995.

[6] S. Deris, S. Omatu, and K. Kitagawa, "Stabilization of inverted pendulum by genetic algorithm," *Proc. of IEEE Int. Conf. on Syst., Man, and Cybern.*, pp. 4372-4377, 1995.

[7] T.-Y. Kuc and W.-G. Han, "An adaptive PID learning control of robot manipulators," *Automatica*, 26, pp.717-725, 2000.

[8] C. P. Neuman and V. D. Tourassis, "Discrete dynamic robot models," *IEEE Trans. Syst., Man, and Cybern.*, 15, pp.193-204, 1985.

[9] A. Karakasoglu, S. I. Sudharsanan, and M. K. Sundareshan, "Identification and decentralized adaptive control using dynamical neural networks with application to robotic manipulators," *IEEE Trans. on Neural Networks*, 4, No. 6, pp.919-930, 1993.

[10] M. O. Efe and O. Kaynak, "Stabilizing and robustifying the learning mechanisms of artificial neural networks in control engineering applications," *Int. J. Intelligent Systems*, 15, No. 5, pp. 365-388, 2000.

[11] H. Sira-Ramirez and E. Colina-Morles, "A sliding mode strategy for adaptive learning in Adalines," *IEEE Trans. on Circuits and Systems – I: Fundamental Theory and Applications*, 42, No. 12, pp. 1001-1012, 1995.

[12] X. Yu, M. Zhihong, and S. M. M. Rahman, "Adaptive sliding mode approach for learning in a feedforward neural network," *Neural Computing & Applications*, 7, pp. 289-294, 1998.

[13] M. O. Efe, O. Kaynak, and X. Yu, "Sliding mode control of a three degrees of freedom anthropoid robot by driving the controller parameters to an equivalent regime," *ASME J. of Dynamic Systems, Measurement and Control*, 122, pp.632-640, 2000.

[14] G. G. Parma, B. R. Menezes, and A. P. Braga, "Sliding mode algorithm for training multilayer artificial neural networks," *Electronics Letters*, 34, No. 1, pp.97-98, 1998.

[15] G. G. Parma, B. R. Menezes, and A. P. Braga, "Neural networks learning with sliding mode control: the sliding mode backpropagation algorithm," *Int. Journal of Neural Systems*, 9, No. 3, pp. 187-194, 1999.

[16] V. I. Utkin, *Sliding Modes and Their Application in Variable Structure Systems.* MIR, Moscow, 1978.

[17] S. Z. Sarpturk, Y. Istefanopulos, and O. Kaynak, "On the stability of discrete time sliding mode control systems," *IEEE Trans. Automatic Control*, 39, No. 10, pp. 930-932, 1987.

[18] F. L. Lewis, C. T. Abdallah, and D. M. Dawson, *Control of Robot Manipulators.* Macmillan Publishing Company, 1993.