БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ . BULGARIAN ACADEMY OF SCIENCES

КИБЕРНЕТИКА И ИНФОРМАЦИОННИ ТЕХНОЛОГИИ · Том **4**, № 1
CYBERNETICS AND INFORMATION TECHNOLOGIES · Volume **4**, No 1

София . 2004 . Sofia

Adaptive Control Schemes Based on Recurrent Trainable Neural Networks

Ieroham Baruch[1], *Jose Martin Flores*[1], *Boyka Nenkova*[2]

[1]*CINVESTAV-IPN, av. IPN No 2508, Mexico D.F. 07560, Mexico*
E-Mail: baruch@ctrl.cinvestav.mx
[2]*Institute of Information Technologies, 1113 Sofia*

Abstract: *The aim of the present paper is to integrate a recurrent neural network in two schemes of real-time soft computing neural control. There are applied the following control schemes: an indirect and a direct trajectory tracking control, using the state and parameter information, given by an identification recurrent neural network. The applicability of the proposed control schemes is confirmed by simulation and experimental results, which exhibits a good convergence.*

Keywords: neural networks, soft computing control, adaptive schemes.

1. Introduction

The Neural Network (**NN**) modeling and application to system identification, prediction and control was discussed for many authors [1-10]. Mainly, two types of **NN** models are used: Feedforward (**FFNN**) and Recurrent (**RNN**). The main problem here is the use of different **NN** mathematical descriptions and control schemes, according to the structure of the plant model. For example, N a r e n d r a and P a r t h a s a r a t h y, [1, 2], applied **FFNN** for system identification and direct model reference adaptive control of various non-linear plants. They considered four plant models with a given structure and supposed that the order of the plant dynamics is known. P a o et al. [3, 4] solved control and prediction problems by means of a flat-type functional **FFNN**, used for direct inverse model learning control. P h a m, Y i l d i r i m [5] applied Jordan **RNN** for robot control. S a s t r y et al. [6] introduced two types of neurons – Network Neurons and Memory Neurons to solve identification and adaptive control problems, considering that the plant model is also autoregressive one. In [7], some schemes of **NN** and **RNN** applications to control, especially of direct model reference adaptive control, are surveyed. In [8, 9, 10], Recurrent Neural Networks are applied for adaptive control. All drawbacks of the described in the literature **NN** models could be summarized as follows:

1. There exists a great variety of **NN** models and universality is missing, [1-10];
2. All **NN** models are sequential in nature as implemented for systems identification. (The **FFNN** model uses one or two tap-delays in the input, [1, 2], and **RNN** models usually are based on the autoregressive model [6] which is one-layer sequential one;
3. Some of the applied **RNN** models are not trainable, others are not trainable in the feedback part, [5]. Most of them are dedicated to a **SISO** and not to a **MIMO** applications [3, 4];
4. In most of the cases, the stability of the **RNN** is not considered, [5], especially during the learning;
5. In the case of **FFNN** application for systems identification, the plant is given in one of the four described in [1] plant models. The linear part of the plant model, especially the system order, has to be known and the **FFNN** approximates only the non-linear part of the plant model [1];
6. All these **NN** models are non-parametric ones [1, 7] and so, not applicable for an indirect adaptive control systems design;
7. All this **NN** models does not perform state and parameter estimation in the same time [7-10].

The major disadvantage of all this approaches is that the identification **NN** model applied is a nonparametric one that does not permit them to use the obtained information directly for control systems design objectives. B a r u c h et al. [11] in their previous paper, applied the state-space approach to describe **RNN** in an universal way, defining a Jordan canonical two-layer **RNN** model, named Recurrent Trainable Neural Network (**RTNN**). This **NN** model is a parametric one, permitting the use of the obtained during the learning parameters for control systems design. Furthermore, the **RTNN** model is a system state predictor/estimator, which permits to use the obtained system states directly for state-space control. The aim of this paper is to use the **RTNN** as an identification and state estimation tool in direct and indirect adaptive control systems of nonlinear plants. The application of **RTNN** in two different indirect and direct control schemes shall be confirmed by simulation and experimental results.

2. Topology and learning of the RTNN model

The *Recurrent Trainable Neural Network* model (RTNN), and its learning algorithm of dynamic *Backpropagation-type*, (BP), is given in [11]. The RTNN model is described by the following equations:

(1) $$x_{k+1} = J_k x_k + B_k u_k \ ,$$

(2) $$z_k = \Gamma[x_k],$$

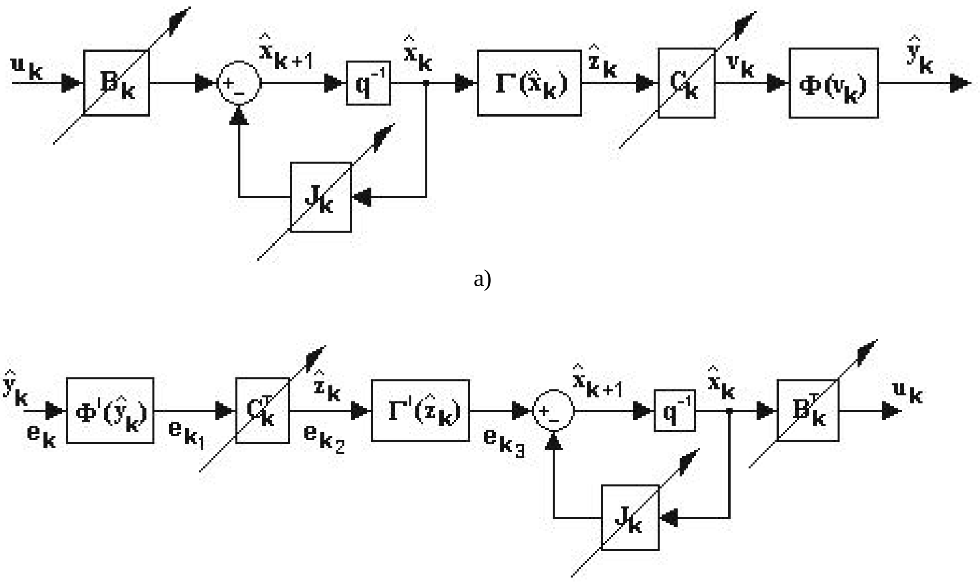
(3) $$y_k = \Phi[C_k z_k],$$

(4) $$J = \text{block} - \text{diag}(J_i); |J_i| < 1,$$

where y, x, and u are, respectively, output, state and input vectors with dimensions l, n, m; J = block–diag(J_i) is a (n?n)-state block-diagonal weight matrix; J_i is an i-th diagonal block of J with 1?1 or 2?2 dimension. Equation (4) represents the local stability conditions, imposed on all blocks of J; B and C are (n?m)- and (l?n)-input and output weight matrices; Γ, Φ are vector-valued sigmoid or hyperbolic tangent-activation functions [11] the sub-index k is a discrete-time variable. The stability of the RTNN model is assured by the activation functions and by the local stability condition (4). Block-diagram of the RTNN topology and its adjoint is given on Fig. 1 a, b. The adjoint RTNN is derived using the diagrammatic method, given in [12]. The controllability and observability of RNN are studied in [13, 14]. The most commonly used BP updating rule, given in matricial form [15], is the following:

$$W_{k+1} = W_k + \eta_k \Delta W_k + \alpha_k \Delta W_{k-1},$$

where W_k is a general weight, denoting each weight matrix (C_k, J_k, B_k) in the RTNN model, to be updated; ΔW_k, (ΔC_k, ΔJ_k, ΔB_k), is the weight correction of W_k; while η and α are learning rate parameters. Following the block-diagram of the adjoint RTNN, the next vector-matricial algorithm of learning [15] could be derived:



a)

b)

Fig. 1. Topology and learning of the RTNN: block-diagram of the RTNN (a); block-diagram of the adjoint RTNN (b)

(5) $$\Delta C_k = e_{k_1} z_k^{\mathrm{T}}; \quad e_{k_1} = \Phi'(y_k) e_k,$$

(6) $$\Delta J_k = e_{k_3} x_k^{\mathrm{T}}; \; e_{k_3} = \Gamma_k'(z_k) e_{k_2}; \; e_{k_2} = C_k^{\mathrm{T}} e_{k_1},$$

(7) $$\Delta vJ_k = e_{k_3} \otimes x_k,$$

(8) $$\Delta B_k = e_{k_3} u_k^{\mathrm{T}},$$

where ΔJ_k , ΔB_k , ΔC_k are weight corrections of the of the learned matrices J_k, B_k, C_k, respectively; $e_k = d_k - y_k$ is an error vector of the output RTNN layer, where d_k is a desired target vector and y_k is a RTNN output vector, both with dimensions l; x_k is an i-th element of the state vector, and e_{jk} are j-th error vectors, illustrated in Fig. 1 b; Γ_k', Φ_k' are Jacobean diagonal matrices with appropriate dimensions, which elements are derivatives of the activation functions. The equation (6) represents the learning of the feedback weight matrix of the hidden layer, where it is supposed to be full (nxn) matrix. The equation (7) gives the learning solution in the case when this matrix is diagonal, which is our case. Stability proof of the learning algorithm is given in [15]. In the next parts an indirect and direct adaptive control schemes, are described.

3. Indirect adaptive trajectory tracking control

The block-diagram of the indirect trajectory tracking control scheme is given in Fig. 2. It contains one identification and state estimation RTNN which issued parameters and states to the controller block. Let us linearize the activation functions of the neural identifier, given by equations (1) to (4) so to obtain:

(9) $$X(k+1)=JX(k)+BU(k),$$

(10) $$Y(k)= CX(k).$$

Following the design procedure, developed in [16], we could obtain the following linear control law, which contain a built in first order reference model:

(11) $$U(k) =(CB)^{-1}\{-CJX(k)+R(k+1)+\gamma[R(k)-Y(k)]\}.$$

The substitution of the control (11) in the identified linear plant model (9), (10), yields:

(12) $$E(k) = R(k) - Y(k);\ E(k+1) + \gamma E(k) = 0,$$

where $R(k)$ is the reference signal, $Y(k)$ is the system output, γ is a constant control parameter with values between –0.999 and 0.999, and $(C_i B_i) \neq 0$ is supposed.

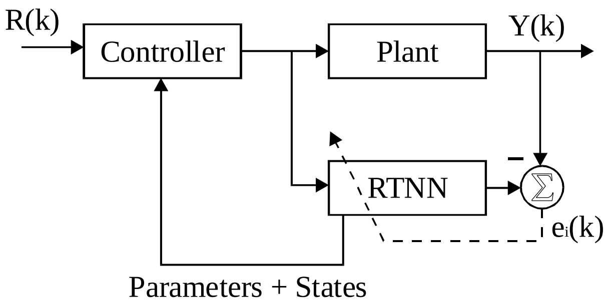


Fig. 2. Block-diagram of the indirect adaptive NN control system

4. Direct adaptive trajectory tracking control

The block-diagram of the direct adaptive trajectory tracking control system is given in Fig. 3. The control system have tree RTNNs: the firs one RTNN-1 is a neural identifier, which issues plant states to the feedback neural controller RTNN-2; the third one RTNN-3 is a feedforward neural controller which has as input the reference signal. The plant control have the total input as a sum of the feedback and feedforward control signals:

$$U(k) = -N_1[X(k)] + N_2[R(k)], \tag{13}$$

where N_1 (*) is a feedback control, generated by RTNN-2; N_2 is a feedforward control, generated by RTNN-3.

The identification RTNN-1 is trained by the identification error $e_i(k)$, and the feedback and feedforward controllers RTNN-2, RTNN-3, respectively – by the control error $e_c(k)$. In the following part, graphical simulation results, obtained using a nonlinear plant model, and both indirect and direct adaptive control schemes, are given.

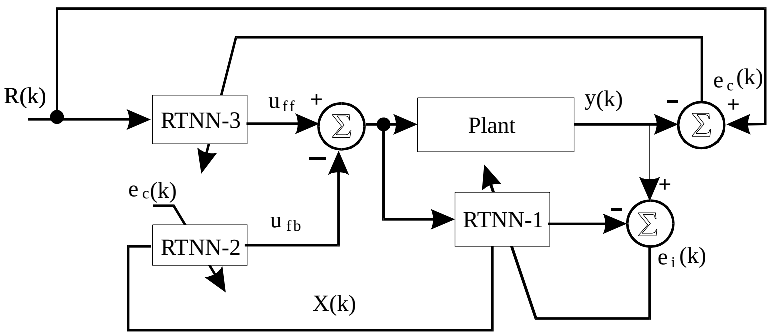


Fig. 3. Block-diagram of the direct adaptive NN control system

5. Simulation results

The nonlinear plant is given by the following discrete-time nonlinear model:

$$y(k+1) = \frac{y(k)y(k-1)y(k-2)u(k-1)[y(k-2)-1]+u(k)}{1+y(k-1)^2+y^2(k-2)}.$$

The obtained simulation results using the on-line indirect adaptive control scheme, are given on Fig. 4 a, b, and Fig. 5 a, b. The RTNN identifier has topology (1, 8, 1), which means one input, one output and eight neurons in the hidden layer. The BP learning parameters are: α=0.01, η=0.03, and the control parameter is γ=0.001; the period of discretization is Ts=0.001 sec. The reference signal is:

$$u(kT_s)=0.5\sin(2\pi kT_s)+0.5\sin(4\pi kT_s)$$

The results of control shows a good convergence which is seen from the comparison of the plants output and the reference signals (Fig. 4a) and the Means

Squared Error of control (MSE%) which is below 1.5% (see Fig. 5b). The identification results also exhibits a good convergence, which is seen on Fig. 4b. and it is a good base for a good control (see Fig. 5a).

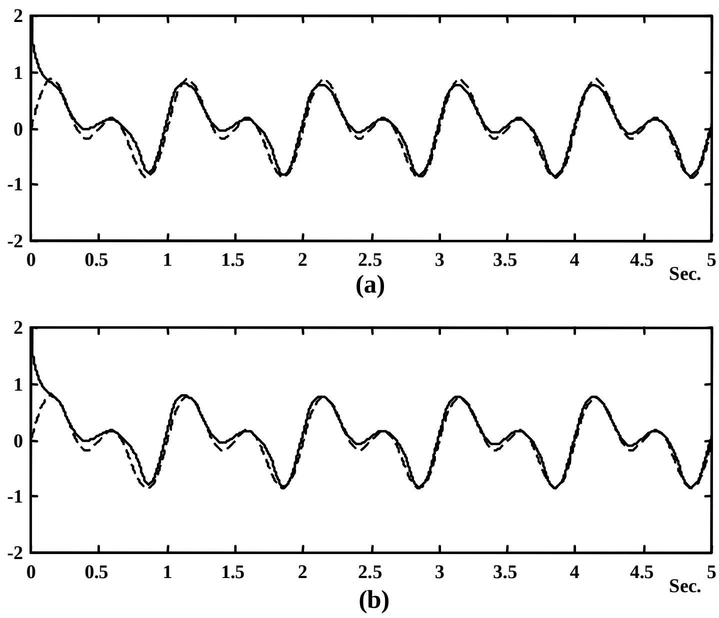


Fig. 4. Graphical results of the Indirect Adaptive Neural Control: comparison between the plant output (dashed line) and the reference signal (continuous line) (a); comparison between the plants output (continuous line) and the RTNN output (dashed line) (b).

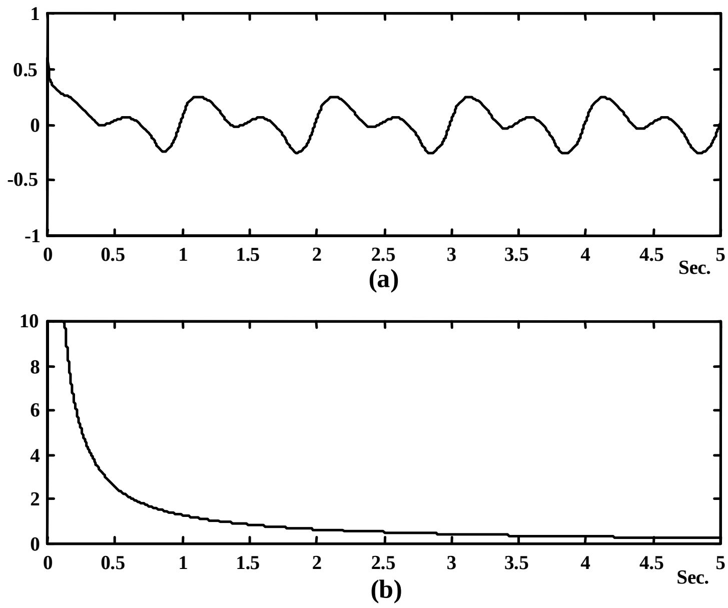


Fig. 5. Graphical results of the Indirect Adaptive Control: control signal (a); MSE% of control (b)

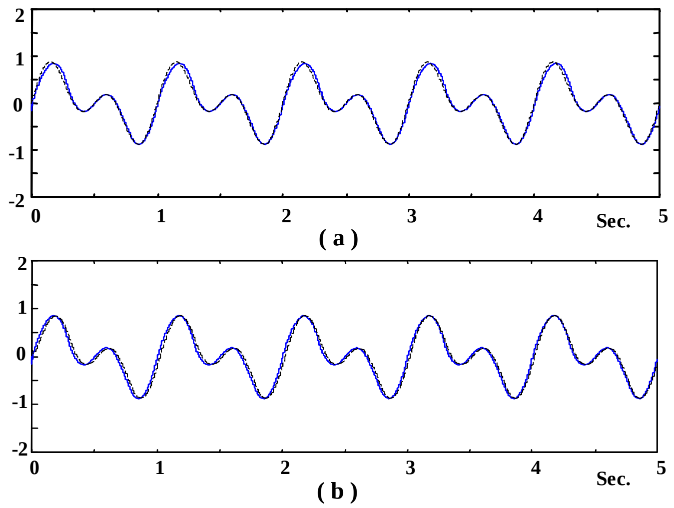


Fig.6. Graphical results of the Direct Adaptive Neural Control. a) comparison between the plant output (dashed line) and the reference signal (continuous line); b) comparison between the plant output (continuous line) and the RTNN output (dashed line).

The obtained simulation results using the on-line direct adaptive control scheme, are given on Fig.6 a, b, and Fig. 7 a, b. The RTNN -1 identifier has topology (1, 2, 1). The RTNN-2 feedback controller has topology (2, 2, 1). The RTNN-3 feedforward controller has topology (1, 2, 1). The BP learning parameters for all three RTNNs are: α=0.01, η=0.03, and the period of discretization is T_s=0.001 s. The reference signal is the same as given above. The results of control shows a good convergence which is seen from the comparison of the plants output and the reference signal (Fig. 6a) and the Means Squared Error of control (MSE%) which is below 0.5% (see Fig. 7b), in spite of the low dimension of the hidden layers of the RTNNs. The identification results also exhibits a good convergence, which is seen on Fig. 6 b. and it is a good base for a good control (see Fig. 7a).

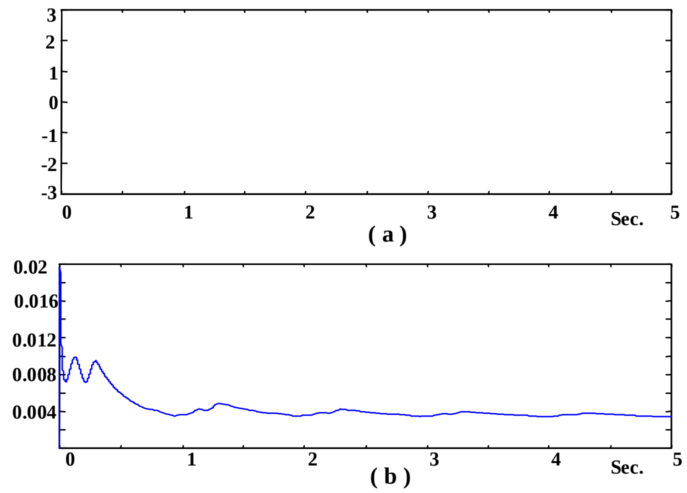


Fig. 7. Graphical results of the Direct Adaptive Control. a) control signal; b) MSE% of control

6. Experimental results

The objective of this section is to describe the experimental results of a RTNN application for real time identification and direct adaptive neural control of a DC motor driven mechanical system. The system uses one RTNN for system identification and two RTNNs for system control. The configuration of the experimental DC-motor mechanical system platform, together with its control and measurement components, are shown on Fig.8. We use a 25 **V**, 8 **A** DC-motor driven by a *power amplifier* and connected by a *data acquisition and control board (DACB), Multi-Q^{TM}* , with the PC. The RTNN was programmed in *MatLab^{TM}-Simulink^{TM}* and *WinCon^{TM}* that is a real-time *Windows 95* application that runs *Simulink generated code* using *Real-time Workshop* to achieve digital real-time control on a *PC* equipped with a DACB. The block diagram of the *soft-computing control scheme* is given on the Fig. 3. The feedback part of the control uses the state vector given from the state estimation/ identification RTNN. The identification RTNN is learned by the error between the DC-motor output position and the RTNN output. The two control RTNNs are learned by the error between the DC-motor output and the reference signal. To preserve the stability of the adaptive control system at the beginning of the working – first the identification RTNN is put on work and learned until converges; then the feedback control RTNN is put on work and learned until converges and finally the same is done with the feedforward control RTNN. The first experiment (Figs.9, 10, 11) corresponds to *a closed-loop input/position real-time identification* of the DC-Motor.

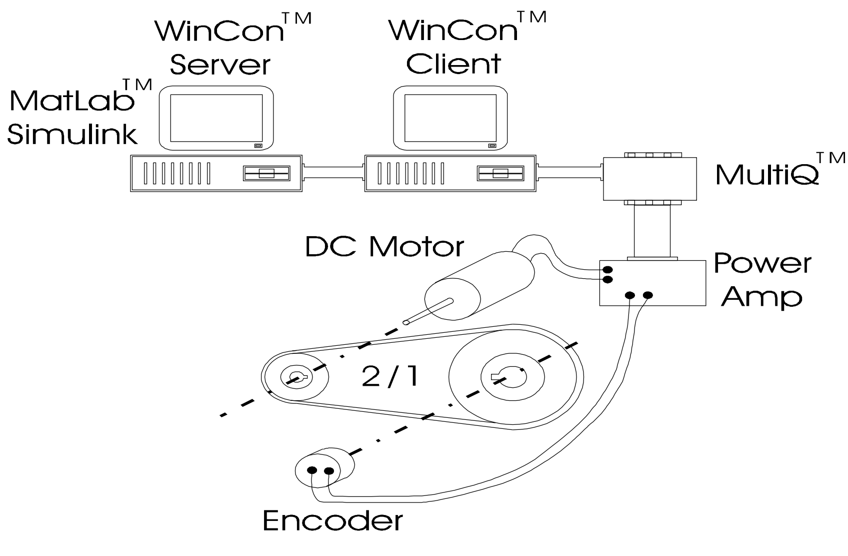


Fig.8. Configuration of the experimental DC-motor mechanical system platform for real-time systems identification and control.

The RTNN model has the following topology: one input neurone, four hidden neurone nodes and one output neurone (1, 4, 1). The learning rate parameters are: $\eta=0.005$, and α = 0.005. The sampling period is T_0= 0.001. The DC- Motor input is chosen as a saturated sinusoid given by:

$$u(kT_s) = 0.277\ \mathrm{sat}[\pi \sin(0.8\pi k]$$

The Figs. 9, 10, 11 shows the graphical results of the firs experiment, where the output position of the DC-motor, the mean squared error and the systems state signals, generated by the RTNN, are given.

The second experiment is the *real-time closed loop identification and state feedback/ feedforward control o*f the DC-motor. The control scheme is the same as in the first experiment, but here the reference signal changes its frequency. The Fig.12 shows the graphical results of the DC-motor control (DC-motor position output, control signal and instantaneous control error). In this experiment the reference signal *changes* its frequency *from 0.8 to 0.5* and a good control systems reaction, is observed.

The third experiment is also a *real-time closed loop* DC-motor *identification and state feedback/ feedforward control*, performed with another shaft load. The same identification and control scheme is used. The Fig.13 gives the load configuration and the Fig. 14, 15, 16, 17 show the graphical results of the DC-motor control (DC-motor position, reference signal, control signal and control error).

The obtained results show a good convergence of al RTNNs and a good reaction and adaptation to frequency and load changes.

6.1. First experiment

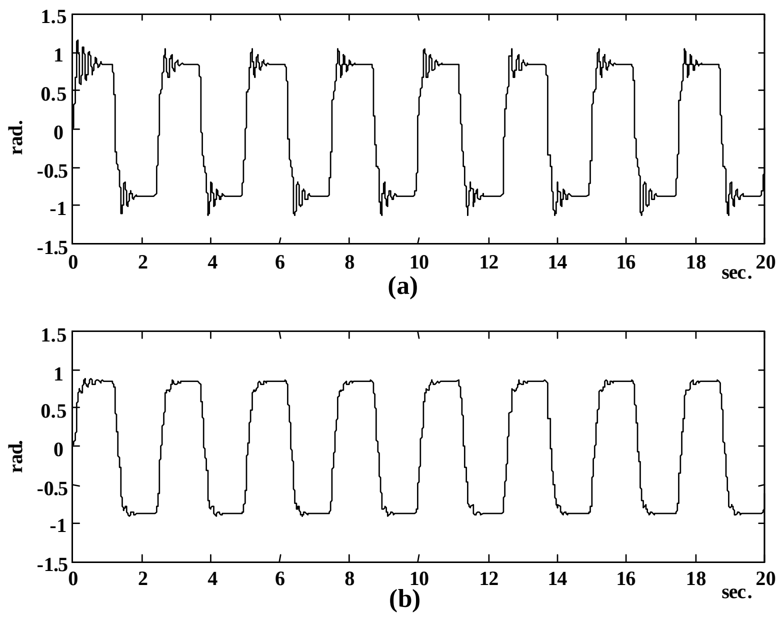


Fig. 9. Graphical results of the Real-Time DC-motor drive identification by a RTNN model, η=0.005, α=0.005, RTNN topology (1, 4, 1); input signal: $u(k)$=0.277sat[πsin(0.8πk)]; output position of the DC-motor (a); output of the identification RTNN (b)

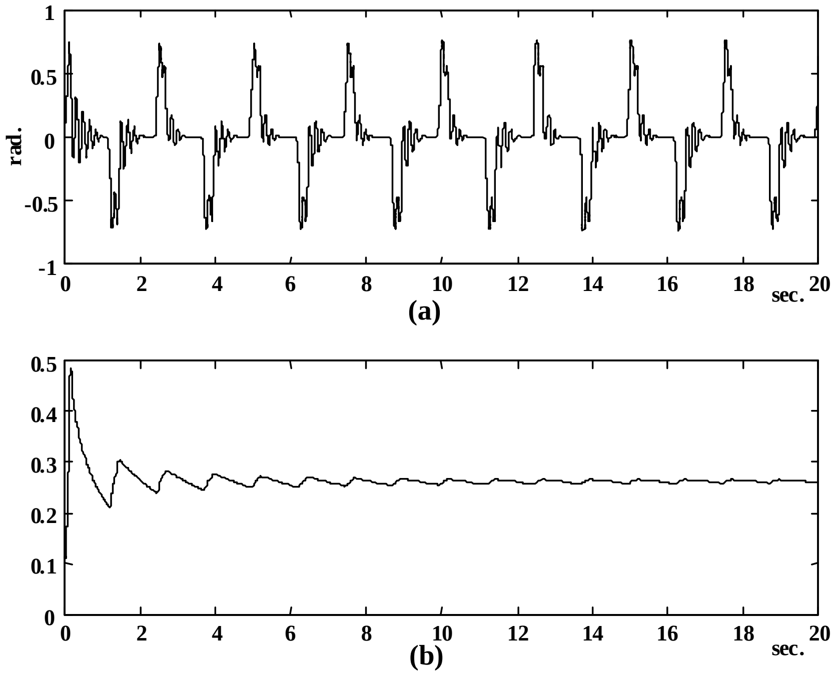


Fig. 10. Errors in the DC-motor drive identification, η=0.005, α=0.005, RTNN topology (1, 4, 1); input signal: $u(k) = 0.277\text{sat}[\pi \sin(0.8\pi k)]$; the sampling rate is T_s=0.001 s.; instantaneous error of identification (a); Mean Square Error (MSE) of the identification (b)

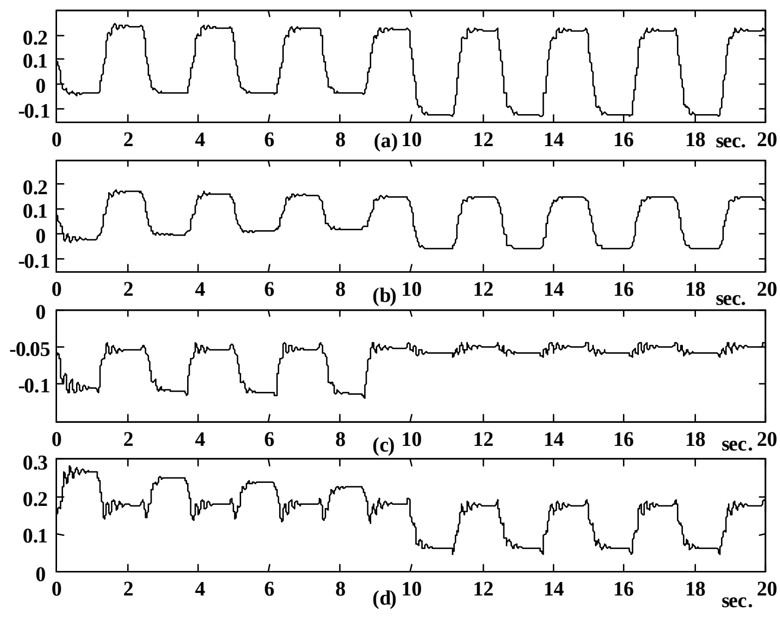


Fig. 11. Systems states estimated by a RTNN model, η = 0.005, α = 0.005, RTNN topology (1, 4, 1); input signal: $u(k)$=0.277 sat [$\pi\sin(0.8\pi k)$]; sampling rate T_s=0.001 s.; $x_1(k+1)$ (a); $x_2(k+1)$ (b); $x_3(k+1)$ (c); $x_4(k+1)$ (d)

6.2. Second experiment

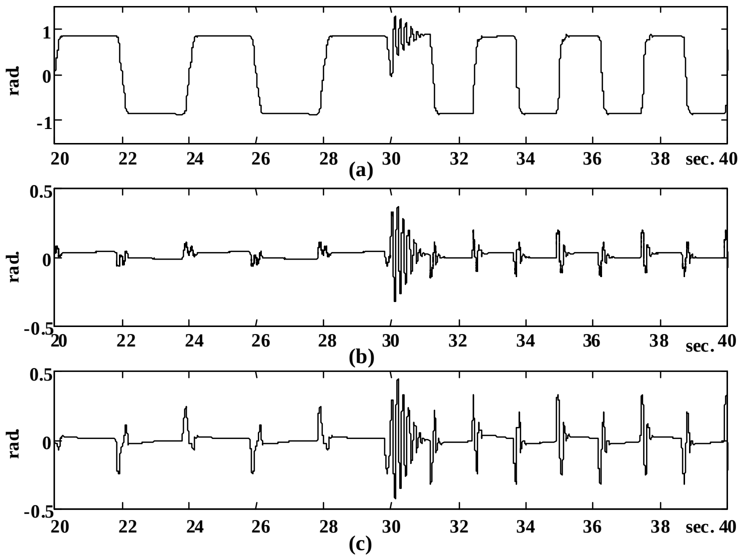


Fig. 12. Real-time state-feedback/feedforward control by two RTNN models and states, estimated by an identification RTNN, η=0.001, α=0.0005, RTNN topologies (1, 4, 1) and (4, 4, 1); the reference signal changes its frequency from *u(k)* = 0.277 sat [πsin (0.8π*k*)] to *u*(*k*)= 0.277 sat [πsin(0.5π*k*)]; the sampling rate is *Ts*=0.001 s.: output position of the DC-motor (a); control signal of the DC-motor (b); instantaneous control error (c)

6.3. Third experiment

The load configuration is given on the Fig.13. where an additional inercia and a gravity force is applied on the shaft of the *DC*-motor.

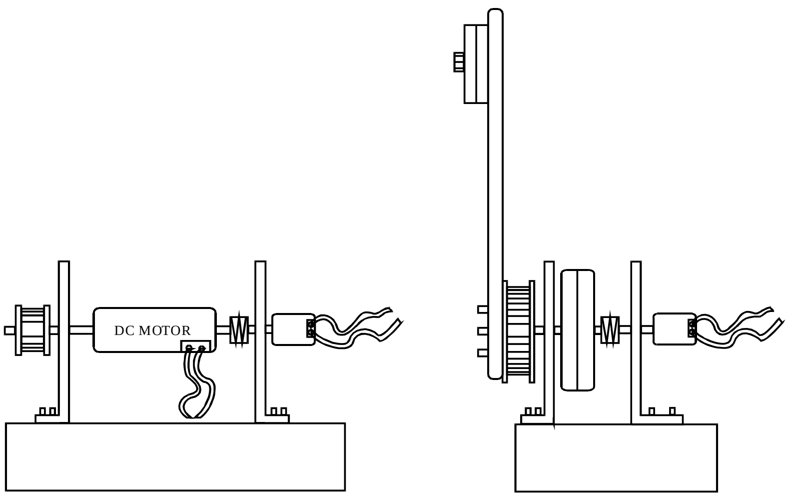


Fig. 13. DC Motor load configuration

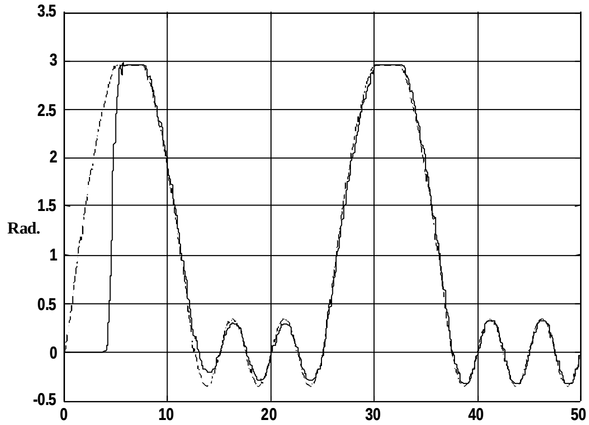


Fig. 14. Comparison of the reference signal and the shaft position. Shaft position (solid line) and reference signal (dashed line); the RTNN topology is (1, 4, 1);. the learning parameters are: for the RTNN-1,2 η, α=0.0005; for the RTNN-3, η=α=0.01. The reference signal is a sequence of two sinusoid functions, which are: $u(kT_s)$= $0.8\pi\sin(T_s k\ \pi/15)$ and $u(kT_s)$= $(1/9)\pi\sin(T_s k\ 2\pi/5)$

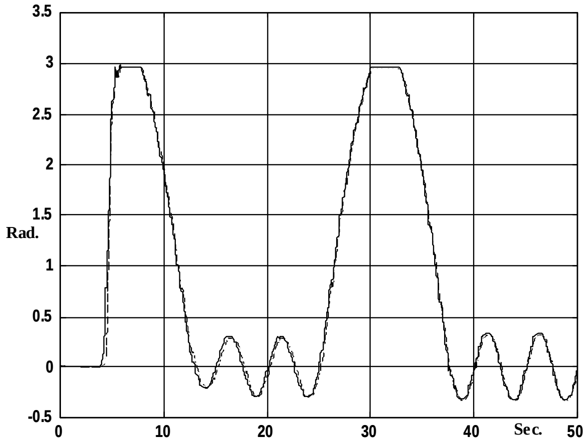


Fig. 15. The response of the RTNN-3 for identification. Output of the RTNN-3 (solid line) and shaft position (dashed line)

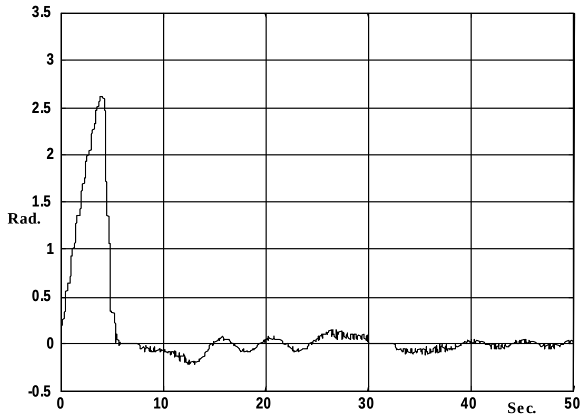


Fig.16. Shaft position instantaneous error

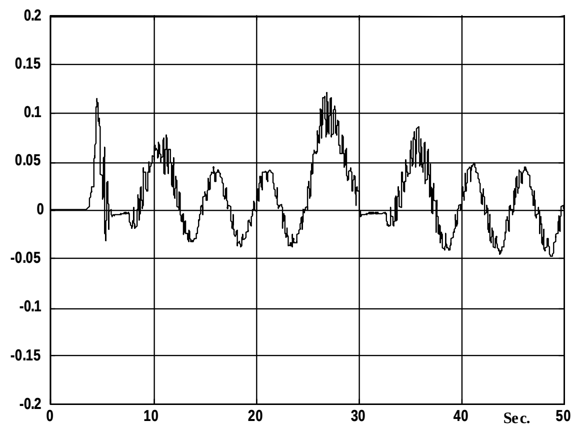


Fig. 17. Control signal, composed of feedback and feedforward components

7. Conclusions

The paper proposed to use two control schemes for an adaptive neural control of nonlinear plants. The first one is an indirect adaptive neural control scheme where the state and parameters, identified by a neural identifier are used to design a control law. The second one is a direct adaptive neural control which uses three RTNN models – one neural identifier which issued a state vector as input to the feedback controller. The feedforward part of the control is generated by a feedforward controller which input is the reference signal. The work of both adaptive neural control schemes is confirmed by simulation results. The direct adaptive neural control scheme is experimented also for real-time DC-motor driven mechanical system identification and state feedback/feedforward control. The applied RTNN model is a Jordan canonical model**,** permitting to use the generated vector of states directly for DC-motor feedback control. The dynamic Backpropagation-type learning algorithm for RTNN model training is also described. The three groups of experimental results, obtained in different operational conditions, confirms the applicability of the described identification and control methodology in practice and also show a good convergence of the applied RTNN's as elements of the adaptive control scheme.

Acknowledgements. The second author of the paper J. Martin Flores has been supported by CONACYT-MEXICO during his studies at CINVESTAV-IPN as a Ph.D. student.

R e f e r e n c e s

1. N a r e n d r a, K. S., K. P a r t h a s a r a t h y. Identification and control of dynamic systems using neural networks. – IEEE Transactions on NNs, **1(1)**, 1990, 4-27.
2. N a r e n d r a, K. S., K. P a r t h a s a r a t h y. Gradient methods for the optimisation of dynamical systems containing neural networks. – IEEE Transactions on NNs, **2(2)**, 1991, 252-262.
3. Y i p, P. P. C., Y. H. P a o. A recurrent neural net approach to one-step ahead control problems. – IEEE Transactions on SMC, **24(4)**, 1994, 678-683.

4. P a o, S. A., S. M. P h i l l i p s, D. J. S o b a j i c. Neural net computing and intelligent control systems. – International Journal of Control, **56**, 1992, 263-289.
5. P h a m, D. T., S. Y i l d i r i m. Robot control using Jordan neural networks. – In: Proc. of the Internat. Conf. on Recent Advances in Mechatronics, Istanbul, Turkey, Aug. 14-16, 1995 (O. Kaynak, M. Ozkan, N. Bekiroglu, I. Tunay, Eds. Bogazicci University Printhouse, Istanbul, Turkey, 1995), Vol. II, 888-893.
6. S a s t r y, P. S., G. S a n t h a r a m, K. P. U n n i k r i s h n a n. Memory networks for identification and control of dynamical systems. – IEEE Transactions on NNs, **5(2)**, 1994, 306-320.
7. H u n t, K. J., D. S b a r b a r o, R. Z b i k o w s k i, P. J. G a w t h r o p. Neural network for control systems – a survey. – Automatica, Vol. 28, 1992, No 6, 1083-1112.
8. J i n, L., P. N. N i k i f o r u n k, M. M. G u p t a. Adaptive model reference control of discrete-time nonlinear systems using neural networks. – Control Theory and Advanced Technology, Vol. 10, September, 1995, No 4. Part 3, 1379-1399.
9. J i n, L., M. G u p t a. Stable dynamic backpropagation learning in recurrent neural networks. – IEEE Transactions on Neural Networks, **10**, 1999, 1321-1334.
10. C h e n, S., S. A. B i l l i n g s. Neural networks for nonlinear dynamics system modeling and identification. – International Journal of Control, **56**, 1992, 263-289.
11. B a r u c h, I., I. S t o y a n o v, E. G o r t c h e v a. Topology and learning of a class RNN. – ELEKTRIK, Vol. 4, Supplement, 1996, 35-42.
12. W a n, E., F. B e a u f a y s. Diagrammatic method for deriving and relating temporal neural networks algorithms. – Neural Computations, **8**, 1996, 182-201.
13. S o n t a g, E., H. S u s s m a n n. Complete controlability of continuos time recurrent neural network. – System and Control Letters, **30**, 1997, 177-183.
14. S o n t a g, E., F. A l b e r t i n i. State observability in recurrent neural networks. – System and Control Letters, **22**, 1994, 235-244.
15. N a v a, F. R., I. B a r u c h, A. P o z n y a k, B. N e n k o v a. Stability proofs of an advanced recurrent neural network topology and learning. Comp. Rend., Bulg. Acad. Sci., Sofia, Bulgaria, 2003 (to appear).
16. B a r u c h, I., J. M. F l o r e s, B. N e n k o v a. Design of indirect adaptive neural control systems. – Cybernetics and Information Technologies, Vvol. 1, Sofia, Bulgaria, 2001, No 1, 81- 94.

Схеми за адаптивно управление, основани на рекурентни обучаеми невронни мрежи

Йерохам Барух[1], Хосе Мартин Флорес[1], Бойка Ненкова[2]

[1]CINVESTAV-IPN, av. IPN #2508, Mexico D.F. 07560, Mexico
E-Mail: baruch@ctrl.cinvestav.mx
[2]Институт по информационни технологии, 1113 София

(Р е з ю м е)

Целта на настоящата статия е да интегрира една рекурентна невронна мрежа в две схеми за невронно soft computing управление в реално време. Приложени са следните схеми на управление: на индиректно и директно управление по траектория, използвайки информация за състоянието и параметрите, получавано от идентификационна рекурентна невронна мрежа. Приложимостта на предложените схеми за управление се потвърждава от симулационните и експерименталните резултати, които показват добра сходимост.