UDC 531.78:681.3.06
Original scientific paper
Received: 02.02.2001.

Emulation of nonlinear mechanical loads using multi-layer neural networks

Muammer Gökbulut, Z. Hakan Akpolat and **Hanifi Güldemir**

University of Firat, Technical Education Faculty, Department of Electronics & Computer Science,
23119 Elazig, TURKEY
e-mail: mgokbulut@firat.edu.tr, hakpolat@firat.edu.tr, hguldemir@firat.edu.tr

SUMMARY

This study describes the torque control of a vector controlled load machine (dynamometer) mechanically coupled to a drive machine for the emulation of nonlinear loads. Proposed dynamometer control strategy is based on model reference control using an on-line trained Multi-layer Neural Networks (MNN). The emulation is involved in the closed loop speed control of the drive machine. After the training of the neuro-controller, the drive machine will see the desired nonlinear mechanical load. An integral compensator supporting the trained MNN is used for eliminating or reducing the model tracking steady state errors. Training problems of the MNN in drive systems are also discussed. Variety of load models which are the nonlinear function of the speed, friction and inertia are successfully emulated and the generalization capability of the trained MNN is tested for various reference inputs. Simulation results showing the excellent dynamometer control performance are presented.

Key words*: load emulation, multi layer neural network, nonlinear load, dynamometer, torque control.*

1. INTRODUCTION

The use of torque-controlled load machines (dynamometers) is common in the testing of electrical machines [1, 2]. In these applications, electrical machine is normally tested under steady state or slowly changing conditions. Recent research, aimed at emulating loads having faster dynamics [3-5], has resulted in simulated load emulation under open-loop conditions i.e. the emulated load is not a part of closed loop speed or position control system. Dynamic load emulation under closed-loop conditions is desirable for evaluating motor drive controllers. However, adaptive and robust control schemes are attracting considerable attention. To verify the effectiveness of these, it is desirable to provide a load machine in which mechanical parameters (such as inertia and friction) can either be pre-programmed or else vary with speed or position. In such cases, it is very desirable that the emulation preserves the model mechanical dynamics.

In addition to machine testing, another application of mechanical load emulation (either in open or closed loop) is to provide off-site testing of converter drives driving real industrial applications. Examples include high-stiction loads (e.g. reciprocating pumps, escalators), period impact loads (large washing machines, compressors), the catching of spinning loads (after power interrupt) and many underhauling/ overhauling applications. If the parameters of such loads are even only approximately known, the ability to evaluate and test such applications off-site would be advantageous.

Previous dynamic emulation research [3-5] is based on the principle of inverse mechanical dynamics in which the shaft speed is measured and used to derive the desired torque for the load machine. In Ref. [3, 4] a model-reference approach is presented in which it is implied that the shaft speed or position could be used as a tracking variable and so avoid the inverse dynamics. In Ref. [5], an integrator back-stepping design technique is presented which claims to emulate a dynamic load under closed loop conditions. However, the desired torque trajectory is still derived from an inverse mechanical model. In Refs. [6] and [7], a new load emulation strategy, based on model reference torque feed-forward control which preserves the dynamics of a desired load model when the emulation is placed in a closed loop speed control system, is proposed for both linear and nonlinear load models accurately.

In the last decade, neural network control has emerged as an attractive research area. The most useful property of the neural networks in control is their ability to approximate any nonlinear mapping in addition to learning, generalization and parallel processing [8-10]. These properties make the neural networks attractive in control applications since they can be applied even if no exact mathematical model of the system exists. Models can be derived from the input-output patterns of the process [11-13]. Thus, neuro-controllers do not require more knowledge of the process and the complex design procedure. Multi-layer neural networks are widely used in control applications and, by selecting a suitable neural network structure and learning algorithm for the training of the network, it is reported that good control performance can be achieved [14-18].

Some researches are reported about the application of neural network control to DC and AC drives [19-23]. They show the effectiveness of the neural network controllers in the nonlinear motor control. However, the implementation problems of the neural network control are not explicitly emphasized (e.g., control performance of the neuro controllers under realistic operating conditions of the electrical drives, problems related with the on-line training of the MNN under the parameter variations, and steady state errors). In addition, to our best knowledge, the neural network control strategy is not applied to the emulation of nonlinear mechanical loads.

This paper presents the neural network torque control of the load machine in order to emulate nonlinear loads by considering practical problems such as, on-line training of the load machine under the realistic operating conditions, limited drive and load machine torque input, and steady state errors. In addition, electrical circuit dynamics of the load machine is not included in the neural network structure thus, this reduces the complexity of the neural network. This paper is organized as follows: In Section 2, we summarize the conventional approaches to the load emulation. In Section 3, the proposed neuro-load emulation scheme is presented. The training of the neural network and its implementation are discussed in Section 4. The emulation is placed in the closed loop speed control system of the drive machine and the simulation results showing the effectiveness of the proposed dynamometer control strategy for various nonlinear loads are presented in Section 5.

2. CONVENTIONAL LOAD EMULATION STRATEGIES

Main idea in the load emulation is to control the torque of the dynamometer mechanically coupled to the drive machine such that, the drive machine will see a load equal to a desired load model. The simplest method for the load emulation is to use the inverse model approach. It is noted that simulations of inverse model approach are often successful [3-5]. However, in practice, noise considerations prohibit the use of small time steps for the computation of the inverse dynamics and discretization effects lead to stability problems. Further, it may not always be possible to derive the inverse dynamics of some nonlinear loads. Some new load emulation strategies are developed to overcome the problems mentioned above [6, 7]. In these methods, basically, the real shaft speed is forced to follow a model reference speed (the desired shaft speed) which is obtained by applying the drive machine torque to the desired emulated load dynamics. It is shown that load emulation strategies mentioned above give excellent results for wide variety of linear and nonlinear loads.

The control strategies of the load machine may be different for the purpose of the emulation. However, the reasonable approach is to find the right load machine torque by minimization of the error between the desired load model speed (ω_m) and the actual speed (ω) as shown in Figure 1 (Note that ω^* is the reference input to the control system). Although the model reference approach is used in Refs. [6] and [7], the control strategy is not adaptive and thus the learning process does not exist. This paper uses the model reference emulation strategy based on neural network control and aims to overcome the design complexity utilizing the approximation, learning and generalization properties of neural networks. Thus, finding an inverse dynamic model of the load model both the noise problems of inverse dynamics approach [3-5] and the control design procedure of the model reference approach [6, 7] are eliminated.

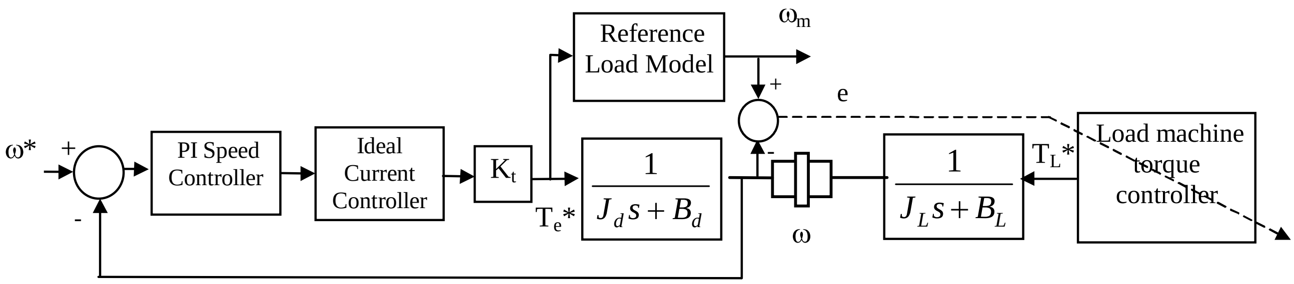

Fig. 1 Model reference load emulation approach

3. LOAD EMULATION SCHEME USING THE MNN

In Figure 1, total inertia and friction seen from the drive machine and the load machine for direct coupled system is $J=J_d+J_L$, and $B=B_d+B_L$ respectively. Thus, linear dynamic torque relation of drive machine can be written as:

$$T_e - T_L = J\frac{d}{dt}\omega + B\omega \tag{1}$$

where T_e is the drive machine torque *(Nm)*, T_L is the load machine torque *(Nm)*, and ω is the shaft speed (rad/s). Let the inertia $J_m(\cdot)$ and friction $B_m(\cdot)$ of the reference load model be a nonlinear function of the total inertia, friction and the speed respectively. We can write the reference load model as:

$$T_e = J_m(\cdot)\frac{d}{dt}\omega_m + B_m(\cdot)\omega_m \tag{2}$$

where ω_m is the reference load model speed. Note that, the type of the nonlinearity in Eq. (2) does not matter when the neural network implementation is considered.

Although some internally dynamics neural network structures which do not need the dynamic information of the system are investigated for the modeling and control purpose [24, 25], the common approach in the neural network control of a system is to find the correct neural network model of the system [8-15]. Therefore, the signals which represent the dynamics of the system should be included in the input vector of the neural networks. For this purpose, backward discrete time equivalence of Eq. (1) can be written as:

$$\omega(k) = \frac{J}{J+BT_s}\omega(k-1) + \frac{T_s}{J+BT_s}\left(T_e(k) - T_L(k)\right) \tag{3}$$

where $T_e(k)$-$T_L(k)$ is the net torque input for the drive machine and T_s is the sampling time. Similarly, the reference load model can be represented:

$$\omega_m(k) = \frac{J_m(.)}{J_m(.)+B_m(.)T_s}\omega_m(k-1) + \frac{T_s}{J_m(.)+B_m(.)T_s}T_e(k) \tag{4}$$

Assume that the error $e(k)$ between reference load model and actual speed is zero. Obviously, this means that $\omega_m(k)$ equals to $\omega(k)$. Thus, the net torque should be:

$$T_e(k) - T_L(k) = \frac{J_m(\cdot)}{T_s}\frac{J+BT_s}{J_m(\cdot)+B_m(\cdot)T_s}\omega_m(k-1) + \\ + \frac{J+BT_s}{J_m(\cdot)+B_m(\cdot)T_s}T_e(k) - \frac{J}{T_s}\omega(k-1) \tag{5}$$

Equation (5) gives us the desired load machine torque to be provided by the neural network as:

$$T_L(k) = f\{\,\omega_m(k-1), \omega(k-1), T_e(k)\} \tag{6}$$

where $f(\cdot)$ is the unknown nonlinear function. If the $J_m(\cdot)$ and $B_m(\cdot)$ consist of a static nonlinear function of the speed, only the previous values of the actual and model speed will be used in Eq. (6). This study assumes that the J and B are unknown and Eq. (6) is to be learned by the neural network. The load emulation control scheme of the dynamometer using the MNN which includes an auxiliary integral compensator is shown in Figure 2.

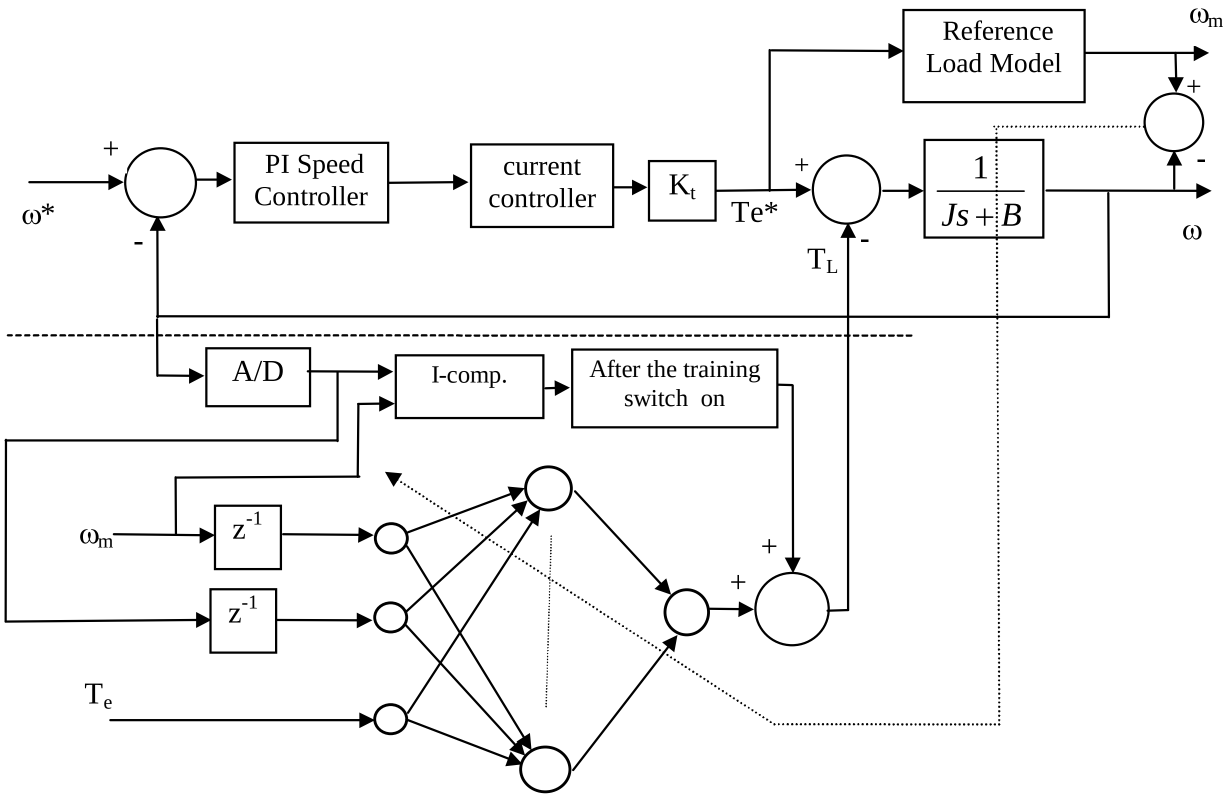

Fig. 2 Model reference load emulation control structure using MNN

3.1 An auxiliary integral controller design

In the previous studies of neural network control [18-25], there is no mention about steady state error which should be especially taken into account in the presence of external disturbances. A desirable solution is to use an integral compensator supporting the neuro-controller when the full trained neuro-controller is not able to decrease the steady state error in an acceptable level. Figure 3 shows the model tracking and steady state errors due to the lack of the integral compensator. Actually, the integral compensator has not considerable effect on the control system when the error (e) is small and no effect when the MNN controller provides the desired performance.

Algorithm for integral compensator can be written as follows:

if $T_{L(MNN)} = T_{L(max)}$ **then** $T_{(i)} = 0$

else $T_{(i)} = K_i \int e\, dt$

if $T_{(i)} > | T_{L(max)} - T_{L(MNN)} |$ **then** $T_{(i)} = | T_{L(max)} - T_{L(MNN)} |$

else $T_{(i)} = T_{(i)}$

$T_L = T_{L(MNN)} + T_{(i)}$

where $T_{L(MNN)}$ is the output of the MNN and $T_{(i)}$ is the output of the integrator. Note that an anti-windup integrator is employed in order to stop the integration during the saturation [7]. Integral compensator gain K_i may be set manually however, in order to increase the performance of the compensator, it can be determined adaptively by using the gradient descent method after the training of the MNN. In this case, the value of the learning rate used to train K_i becomes important since the error will be zero in a short time during the training and, the best value of K_i should be calculated in this period.

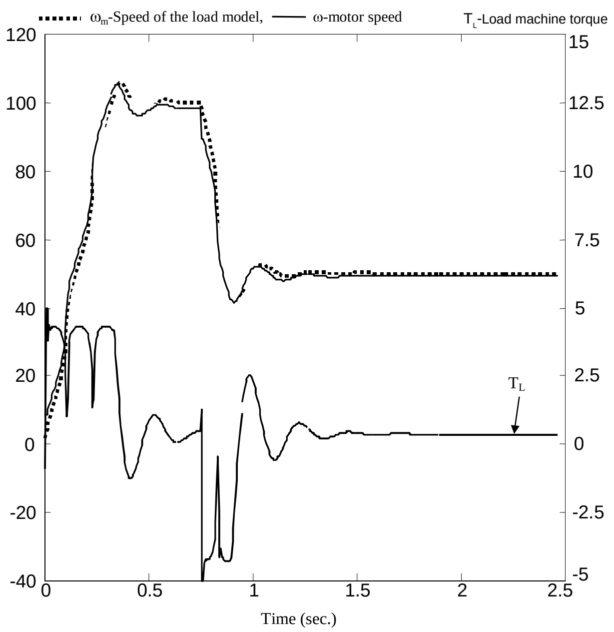

Fig. 3 The steady state errors in the neuro load emulation

4. TRAINING OF THE MULTILAYER NEURAL NEWORKS

Training of the MNN means finding a procedure for the adaptation of the MNN weights which minimize the selected performance criteria. Pattern learning algorithm for the training of the neuro-controller is inevitable for on-line adaptation. For pattern learning, the error (e) between the model and actual speed (model tracking error), and the square of the error as a performance criteria (E) are defined in discrete time as:

$$\begin{aligned} e(k) &= \omega_m(k) - \omega(k) \\ E(k) &= \frac{1}{2} e^2(k) \end{aligned} \tag{7}$$

where $\omega_m(k)$ is the speed of the load model. Note that the MNN output error (e_t) between the ideal and actual load machine torque is not known explicitly. Therefore, this error should be estimated using the model tracking error as follows:

$$e_t(k) = \frac{\partial E(k)}{\partial T_L(k)} = \frac{\partial E(k)}{\partial e(k)} \frac{\partial e(k)}{\partial \omega(k)} \frac{\partial \omega(k)}{\partial T_L(k)} \tag{8}$$

If the derivative of $\dfrac{\partial \omega}{\partial T_L}$ is calculated in discrete time, the MNN output error can be found as:

$$e_t(k) = \frac{\partial E(k)}{\partial T_L(k)} = -\frac{\omega(k) - \omega(k-1)}{T_L(k) - T_L(k-1)} e(k) \tag{9}$$

Actually, instead of accurate value of $\dfrac{\partial \omega}{\partial T_L}$, the sign of this derivative is sufficient for the training of the MNN and, this procedure is known as direct adaptive control, since the forward MNN model of the motor is not used for the training of the MNN controller.

In this study, 3-6-1 (the number of inputs, hidden layer neurons and output respectively) MNN which has a hidden layer with sigmoid and an output neuron with linear activation function is used. Feedforward mathematical relation of the MNN can be written as:

$$v_j(k) = \varphi\left\{ \sum_{i=0}^{n} W_{ji}(k) x_i(k) \right\}, \quad T_L(k) = \sum_{j=0}^{m} \theta_j(k) v_j(k) \tag{10}$$

where, $x_i(k) = [\omega_m(k-1), \omega(k-1), T_e(k)]$ is the i^{th} input vector of the MNN, $v_j(k)$ is the j^{th} output of the hidden layer, W_{ji} is the weight between the i^{th} input and the j^{th} hidden layer neuron and θ_j is the j^{th} weight of the output layer. As an example of training, correction to be applied to any weight in the hidden layer can be calculated as follows:

$$\Delta W_{ji}(k) = \mu \delta_j(k) x_i(k) - \eta \Delta W_{ji}(k-1) \tag{11}$$

where, δ_j is known as local error for any neuron in the hidden layer and can be calculated as:

$$\delta_j(k) = e_t(k)\theta_j(k)\varphi'_j(\cdot) \qquad (12)$$

where $\varphi'_j(\cdot)$ is the derivative of the hidden layer activation function.

If the neural network control strategy given above is implemented in practice, due to the difficulty of the off-line adaptation, the on-line adaptation, which is usually known as pattern learning in the neural network literature, should be used. However, this may result in undesired responses in the initial training stage since the MNN weights are assigned randomly. In addition, external torque or speed dependent inertia and friction may cause the wrong gradient which is calculated according to the actual input (T_L) in Eq. (8). It is clear that a suitable solution for this type of problems is to choose appropriate initial values of the weights of the MNN. Thus, this requires the simulation of the system and then the weights obtained from the simulated system can be used for practical applications. Since the MNN learns the nonlinear motor dynamics instead of memorizing the motor parameters, it is expected that the pre-training provides a reasonable start for the training of the MNN in practice.

In this study, the MNN is trained for the constant values of $J_m=2J$ and $B_m=5B$ and then the weights obtained are used as initial weights of the MNN controller which is used in the nonlinear parameter variation cases. Actually, it has been seen that the weights obtained from the simulation for constant J_m and B_m values provide a reasonable control performance for the nonlinear load models however, a better performance is obtained by continuing the training.

5. SIMULATION RESULTS

Performance of the neural network control structure given in Figure 2 is tested for the drive system which has the nominal parameters: J=3.5×10^{-3} kgm^2, B=7×10^{-4} Nms, T_{emax}=5 Nm, T_{Lmax}=5 Nm. A suitable PI controller is designed for the drive machine since the aim of this paper is to control the load machine using the MNN to provide a desired load model. The initial training of the MNN is implemented for a linear load model which has the parameters of $J_m=2J$ and $B_m=5B$. For the training, 300.000 (T_s is 5 ms) patterns are used with the learning rate of 2×10^{-3}. Good emulation performance is obtained and neural networks weights are saved.

When the control performance of the trained MNN (without the integral compensator) is tested for nonlinear load models of Eqs. (13) and (14), reasonable model tracking performances are observed. However, in order to provide the accurate dynamics of the load model for the emulation, more training is required. Note that the integral compensator is used to eliminate only small errors and thus, more training is necessary to reduce the model tracking error. In Eqs. (13) and (14), J and B are the total nominal inertia and friction of the drive and load machine, J_m and B_m are the nonlinear inertia and friction to be emulated:

$$\begin{aligned} T_{ext} &= 2Nm,\;\; 60 < \omega_m < 80 \\ J_m &= 4J + K \cdot \omega^2 \\ B_m &= 10B + B_a \cdot \omega \end{aligned} \qquad (13)$$

where T_{ext} is the external disturbance torque and the constants are K=2×10^{-6} and B_a=1×10^{-4}. Equation (13) implies that the drive machine is faced on the speed dependent inertia and friction in addition to the external pulse disturbance torque. Training of the MNN controller is carried on for 150.000 patterns for sinusoidal speed reference and then the weights are saved. The performance of the trained MNN and the integral compensator is tested for the sinusoidal and step input references as shown in Figures 4 and 5.

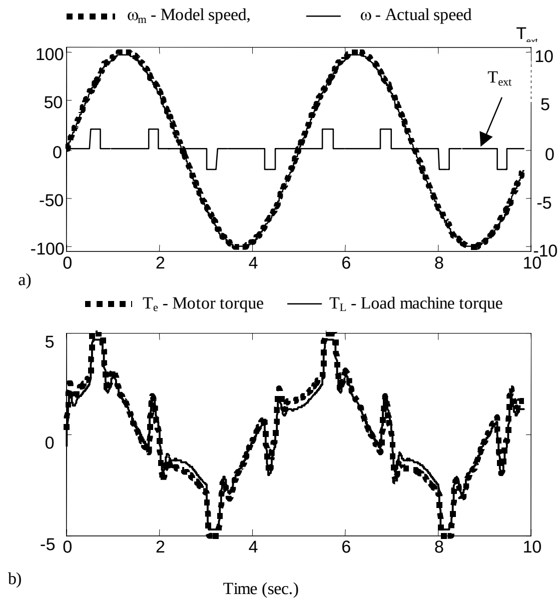

Fig. 4 (a) The model speed, actual speed and the external disturbance torque; (b) Drive machine (motor) and load machine torques

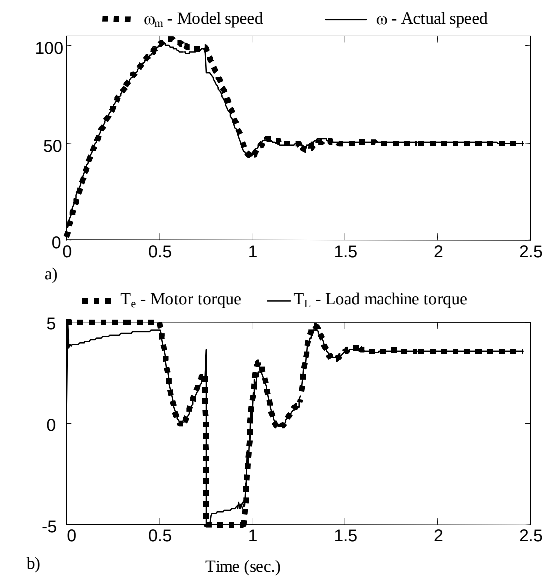

Fig. 5 (a) The model and actual speeds; (b) Drive machine and load machine torques

Figure 4(a) shows the model and actual speeds for the sinusoidal speed reference. An external pulse torque of 2 Nm is also applied as seen in Figure 4(a). Figure 4(b) shows the torques of the drive and load machines (T_e and T_L). In addition, Figure 5 shows the performance of the controllers for a step speed reference with an external torque of 4 Nm applied at t=1.25 s. Note that the step input reference has a value of 100 rad/s between t=0 and 0.75 seconds and then the step is changed to 50 rad/s at t=0.75s. As seen in these figures, very good emulation performances are obtained.

Another generalization performance of the trained MNN controller is given in Figure 6 with the load model of Eq. (14) in which the emulated inertia and friction change as a function of the speed. Note that the drive machine torque is not shown in Figure 6(b) for clarity. The step reference input (ω*) used in Figure 6(c) is the same as in Figure 5(a):

$$\begin{aligned} J_m &= 4J + 3J\sin(0.15\omega) \\ B_m &= 10B + 5B\cos(0.15\omega) \end{aligned} \tag{14}$$

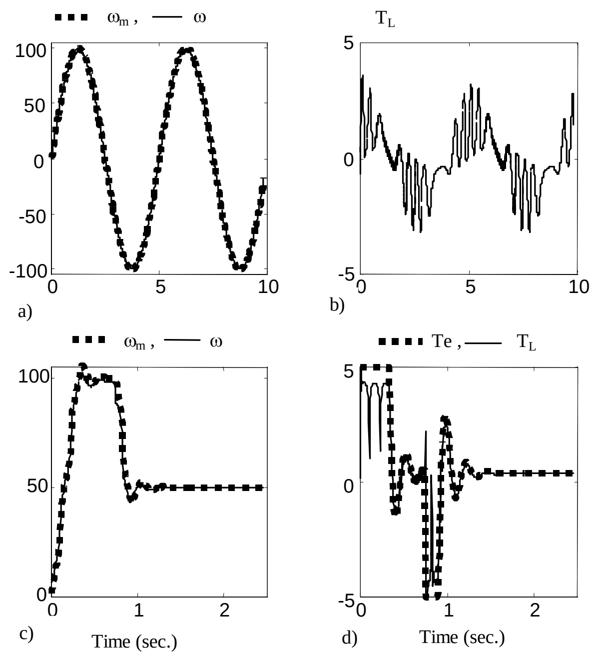

Fig. 6 (a) Model and actual speeds for a sinusoidal reference input; (b) Load machine torque for the sinusoidal reference input; (c) Model and actual speeds for a step reference; (d) Drive machine and load machine torques for the step reference

6. CONCLUSION

In this paper, a new load emulation technique, based on the neural network control strategy and model reference approach, is proposed. In the proposed emulation method, an integral compensator supporting the neural network controller is used for the elimination of the small steady state model tracking errors. Comparing it to the previous emulation techniques, this emulation strategy has the advantage of design simplicity and it has not got the discretisation or inverse dynamics problems. In this study, simulation results showing the performance of the emulation strategy are presented and some practical constraints (e.g., the torque demand limitation, the training problems due to the on-line implementation) are taken into consideration in the simulations. The experimental implementation of the proposed emulation strategy will be the subject of further work.

7. REFERENCES

[1] C.R. Wasko, A universal AC dynamometer for testing motor drive systems, Proc. IEEE-IAS Conf., Vol. 1, pp. 409-412, 1987.

[2] A.C. Williamson and K.M.S. Al-Khalidi, An improved engine testing dynamometer, Proc. 4th Int. Conf. Elec. Mach. And Drives, Vol. 1, pp. 374-378, 1989.

[3] R.W. Newton, R.E. Betz and H.B. Penfold, Emulating dynamic load characteristics using a dynamic dynamometer, Proc. Int. Conf. Power Electron and Drive Systems, Vol. 1, pp. 465-470, 1995.

[4] R.E. Betz, H.B. Penfold and R.W. Newton, Local vector control of an AC drive system load simulator, Proc. IEEE Conf. Contr. Appl., Vol. 1, pp. 721-726, 1994.

[5] J.J. Carrol, D.M. Dawson and E.R. Collins, A non-linear control technique for the development of a computer controlled dynamometer, *Proc. Dynamic Sys. and Contr. Division, American Soc. of Mech. Eng.*, Vol. 53, pp. 31-36, 1993.

[6] Z.H. Akpolat, G.M. Asher and J.C. Clare, Dynamic emulation of mechanical loads using a vector controlled induction motor-generator set, *IEEE Trans. on Industrial Electronics*, Vol. 46, No. 2, pp. 370-379, April 1999.

[7] Z.H. Akpolat, G.M. Asher and J.C. Clare, Experimental dynamometer emulation of non-linear mechanical loads, *IEEE Trans. on Industry Applications*, Vol. 35, No. 6, pp. 1367-1373, 1999.

[8] S. Tamura and M. Tateishi, Capabilities of a four-layered feedforward neural network: Four layers versus three, *IEEE Transactions on Neural Networks*, Vol. 8, No. 2, pp. 251-255, 1997.

[9] K.S. Narendra and K. Parthasarthy, Identification and control of dynamical system using neural networks, *IEEE Transactions on Neural Networks*, Vol. 1, No. 1, pp. 4-27, 1990.

[10] P.J. Werbos, Backpropagation through time: What it does and how to do it, *Proc. of the IEE*, Vol. 78, No. 10, pp. 1550-1560, 1990.

[11] J. Sjöberg, Q. Zhang, L. Ljung et al, Nonlinear black-box modeling in system identification, *Automatica*, Vol. 31, No. 12, pp. 1691-1724, 1995.

[12] K.S. Narendra and K. Parthasarthy, Gradient methods for the optimization of dynamical systems containing neural networks, *IEEE Transactions on Neural Networks*, Vol. 2, No. 2, pp. 252-262, 1991.

[13] K.J. Hunt, D. Sbarbaro and R. Zbikowski, Neural networks for control systems - A Survey, *Automatica*, Vol. 28, No. 6, pp. 1083-1112, 1992.
[14] K.S. Narendra and S. Mukhopadhyay, Adaptive control using neural networks and approximate models, *IEEE Transactions on Neural Networks*, Vol. 8, No. 3, pp. 475-485, 1997.
[15] J. Tanomaru and S. Omatu, Process control by on-line trained neural controllers, *IEEE Transactions on Industrial Electronics*, Vol. 39, No. 6, pp.511-521, 1992.
[16] S.W. Piche, Steepest descent algorithms for neural network controllers and filters, *IEEE Transactions on Neural Networks*, Vol. 5, No. 2, pp. 198-212, 1994.
[17] S.Z. Qin, H.T. Su and T. J. McAvoy, Comparison of four neural net learning methods for dynamic system identification, *IEEE Transactions on Neural Networks*, Vol. 3, No. 1, pp. 122-130, 1992.
[18] L. Jin, P.N. Nikiforuk and M.M. Gupta, Adaptive control of discrete-time nonlinear systems using recurrent neural networks, *IEE Proceedings Control Theory Applications*, Vol. 141, No. 3, pp. 169-176, 1994.
[19] S. Weerasooriya and M.A. El-Sharkawi, Idenfification and control of a DC motor using back-propagation neural networks, *IEEE Transactions on Energy Conversion*, Vol. 6, No. 4, pp. 663-669, 1991.
[20] M.A. El-Sharkawi, A.A. El-Samahy and M.L. El-Sayed, High performance drive of DC brushless motors using neural networks, *IEEE Transactions on Energy Conversion*, Vol. 9, No. 2, pp. 317-322, 1994.
[21] A. Albostan and M. Gokbulut, Modeling and adaptive control of permanent magnet synchronous motors using multilayer neural networks, Mechatronics 98, Elsevier Science Ltd., pp. 109-116, 1998.
[22] G. Fathala and M. Farsi, Modelling and control of a brushless DC motor using RBF neural network, Proc. 12th Int. Conf. on Systems Eng., ICSE'97, pp. 248-251, 1997.
[23] M.A. Abido and Y.L. Abdel-Magid, On-line identification of synchronous machines using radial basis function neural networks, *IEEE Transactions on Power Systems*, Vol. 12, No. 4, pp. 1500-1506, 1997.
[24] P.S. Sastry, G. Santharam and K.P. Unnikrishnan, Memory neuron networks for identification and control of dynamical systems, *IEEE Transactions on Neural Networks*, Vol. 5, No. 2, pp. 306-319, 1994.
[25] T. Fukuda, T. Shibata and M. Tokita, Neuromorphic control: Adaptation and learning, *IEEE Transactions on Industrial Electronics*, Vol. 39, No. 6, pp. 497-503, 1992.

OPONA[ANJE NELINEARNIH MEHANI^KIH OPTERE]ENJA POMO]U VIŠESLOJNIH NEURALNIH MRE@A

SA@ETAK

Ovaj rad opisuje kontrolu obrtanja vektorom kontroliranog stroja (dinamometra) koji je mehani~ki povezan za pogonski stroj zbog opona{anja nelinearnih optere}enja. Predlo`ena strategija kontrole dinamometra zasniva se na modelu odgovaraju}e kontrole koja koristi on-line trenirane višeslojne neuralne mre`e (MNN). Opona{anje se uklju~uje u kontrolu brzine zatvorene petlje pogonskog stroja. Nakon treniranja neurokontrolora, pogonski stroj }e polu~iti `eljeno nelinearno mehani~ko optere}enje. Integralni kompenzator koji podupire trenirani MNN koristi se za eliminaciju ili smanjenje grešaka modela traganja za stabilnim stanjem. Ovaj rad govori i o problemima treniranja MNN kod pogonskih sustava. Brojni modeli optere}enja koji su nelinearna funkcija brzine, trenja i tromosti uspješno se opona{aju, a sposobnost generalizacije tretiranog MNN testira se za razli~ite odgovaraju}e ulazne podatke. Predstavljeni su rezultati simulacije koji pokazuju izvrsnu kontrolu rada dinamometara.

Klju~ne rije~i: *opona{anje optere}enja, vi{eslojne neuralne mre`e, nelinearno optere}enje, dinamometar, kontrola obrtanja.*