A HYBRID TRAINING PROCEDURE FOR ARTIFICIAL NEURAL NETWORKS LEADING TO PARAMETRIC STABILITY AND COST MINIMIZATION

Efe, M. O.*, and Kaynak, O.**

*Bogazici University, Electrical and Electronic Engineering Dept., Mechatronics Research and Application Center, Bebek, 80815, Istanbul, (TURKEY) T. +90-212-263 15 40 x 1855 F. +90-212-287 24 65
E-mail: efemond@boun.edu.tr

**Bogazici University, Electrical and Electronic Engineering Dept., UNESCO Chair on Mechatronics, Bebek, 80815, Istanbul, (TURKEY) T. +90-212-287 24 75 F. +90-212-287 24 65
E-mail: kaynak@boun.edu.tr

Abstract - *This paper presents a novel training algorithm for artificial neural networks. The algorithm combines the gradient descent technique with variable structure systems approach. The combination is performed by expressing the conventional weight update rule in continuous time and application of sliding mode control method to the gradient based training procedure. The proposed combination therefore exhibits a degree of robustness with respect to the unmodeled multivariable internal dynamics of gradient descent. With conventional training procedures, the excitation of this dynamics during a training cycle can lead to instability, which may be difficult to alleviate due to the multidimensionality of the solution space and the ambiguities on the free design parameters, such as learning rate or momentum coefficient. This paper demonstrates that a computationally intelligent system can be trained such that the adjustable parameter values are forced to settle down (parameter stabilization) while minimizing an appropriate cost function (cost optimization). The proposed approach is applied to the control of a robotic arm using feedforward neural networks.*

1. INTRODUCTION

Artificial Neural Networks (ANN) are well known with their property of representing complex nonlinear mappings. Earlier works on the mapping properties of these architectures have shown that neural networks are universal approximators [1-2]. The mathematical power of intelligence is commonly attributed to the neural systems because of their massively interconnected, fault tolerant architecture. Various architectures of neural systems are studied in the literature. Feedforward and recurrent neural networks, Gaussian radial basis function neural networks, dynamical neural networks [3] constitute typical structurally different models. The models mentioned have successfully applied to problems extending from control applications to image/pattern recognition problems.

In control engineering practice stability and robustness are of crucial importance. Because of this, the implementation-oriented control engineering expert is always in pursuit of a design, which provide accurate tracking as well as insensitivity to environmental disturbances and structural uncertainties. At this point, it must be emphasized that these ambiguities can never be modeled accurately. When the designer tries to minimize the ambiguities by the use of a detailed model, then the design becomes so tedious that its cost increases dramatically. A suitable way of tackling with uncertainties without the use of complicated models is to introduce Variable Structure Systems (VSS) theory based components into the system structure.

Variable Structure Control (VSC) has successfully been applied to a wide variety of systems having uncertainties in the representative system models. The philosophy of the control strategy is simple, being based on two goals. First, the system is forced towards a desired dynamics, second, the system is maintained on that differential geometry. In the literature, the former dynamics is named the reaching mode, while the latter is called the sliding mode. The control strategy borrows its name from the latter dynamic behavior, and is called *Sliding Mode Control (SMC)*.

Earliest notion of SMC strategy was constructed on a second order system in the late 1960s by Emelyanov [4]. The work stipulated that a special line could be defined on the phase plane, such that any initial state vector can be driven towards the plane and then be maintained on it, while forcing the error dynamics towards the origin. Since then, the theory has greatly

0-7803-5670-5/99/$10.00 ©1999 IEEE

been improved and the sliding line has taken the form of a multidimensional surface, called the *sliding surface* and the function defining it is called the *switching function.*

Numerous contributions to VSS theory have been made during the last decade, some of them are as follows. Hung *et al* [5] has reviewed the control strategy for linear and nonlinear systems. In [5], the switching schemes, putting the differential equations into canonical forms and generating simple SMC based controls are considered in detail. Gao *et al* [6], apply the SMC scheme to robotic manipulators and discuss the quality of the scheme. One of the crucial points in SMC is the selection of the parameters of the sliding surface. Some studies devoted to the adaptive design of sliding surfaces have shown that the performance of control system can be refined by interfacing it with an adaptation mechanism, which regularly redesigns the sliding surface [7-8]. This eventually results in a robust control system. The performance of SMC scheme is proven to be satisfactory in the face of external disturbances and uncertainties in the system model representation. The latest studies consider this robustness property by equipping the system with computationally intelligent methods. In [9] and [10], fuzzy inference systems are proposed for SMC scheme. A standard fuzzy system is studied and the relevant robustness analyses are carried out. Particularly, the work presented in [9] emphasizes that the robustness and stability properties of soft computing based control strategies can be analyzed through the use of SMC theory. It is shown in this paper that the approach is robust i. e. it can compensate the deficiencies caused by poor modeling of plant dynamics and external disturbances.

The objective of this paper is to develop a training procedure for artificial neural networks, which will enforce the adjustable weights and biases to settle down to a steady state solution while minimizing an appropriate cost function. This is achieved through a suitable integration of error backpropagation algorithm [11] with VSS philosophy.

This paper is organized as follows: The second section briefly reviews the conventional error backpropagation. The third section is devoted to the derivation of the proposed method starting with the continuous time representation of gradient descent approach. The parameter stabilizing law is derived and its applicability is discussed. The section continues with an explanation of how the proposed law and error backpropagation is combined. The fourth section presents the application of the proposed scheme to artificial neural networks. In the fifth section, the plant, which is a two degrees of freedom SCARA robotic manipulator, is presented. Next the simulation results are discussed. Conclusions constitute the last part of this study.

2. GRADIENT BASED CONVENTIONAL TRAINING PROCEDURE

In this section, a widely used technique of parameter adjustment is briefly reviewed. The method has first been formulated by Rumelhart *et al* [11] in 1980s. The approach has successfully been applied to a wide variety of optimization problems.

Let d and F denote the target signal and the neural network output in response to the input u respectively. The discrepancy between these two quantities is given in (1). The realization cost can be formulated with this error measure as given in (2).

$$e = d - F(\phi, u) \tag{1}$$

$$J = \frac{1}{2} e^2 \tag{2}$$

In order to minimize the realization cost in (2), gradient descent method proposes the following update strategy for a generic parameter denoted by ϕ.

$$\Delta\phi = -\eta_\phi \frac{\partial J}{\partial \phi} \tag{3}$$

Since the target signal is not dependent to the adjustable parameters of the neural network, the rule described by (3) can more clearly be stated as in (4).

$$\Delta\phi = \eta_\phi e \frac{\partial F(\phi, u)}{\partial \phi} \tag{4}$$

The minimization proceeds recursively as given in (4), for which the sensitivity derivative with respect to the generic parameter ϕ is needed. It is apparent that the method is applicable to the architectures in which the outputs are differentiable with respect to the subject of optimization.

3. DERIVATION OF THE PARAMETER STABILIZING LAW BY USING VARIABLE STRUCTURE SYSTEMS APPROACH

If the formula given in (4) is assumed to be activated at integer multiples of the sampling period T_s, the dynamic behavior of the parameter change can be formulated as given in (5) by utilizing Euler's first order approximation.

$$\dot{\Delta\phi} = -\frac{1}{T_s}\Delta\phi + \frac{\eta e}{T_s}\frac{\partial F(\phi, u)}{\partial \phi} \tag{5}$$

Since (5) has been derived from the update formula, sampling period will simply drop out from the equations. In (6), the equivalency between the

continuous and discrete forms of update dynamics is clarified.

$$\frac{\Delta\phi(k+1)-\Delta\phi(k)}{T_s} = -\frac{\Delta\phi(k)}{T_s} + \frac{1}{T_s}\eta_\phi N_\phi(k) \qquad (6)$$

Where $N_\phi = e\partial F(\phi, u)/\partial\phi$. The synthesis of the parameter stabilizing criterion is based on the stabilization of the system in (5) by VSS methodology, which assumes the learning rate as the input of the system in (5). In the design of variable structure controllers, one method that can be followed is the reaching law approach [5]. For the use of this theory, each adjustable parameter is assigned a switching function described by (7), and the switching scheme, which is adopted for all design parameters in the architecture, is described by (8).

$$s_\phi = \Delta\phi \qquad (7)$$

$$\dot{s}_\phi = -\frac{Q_\phi}{T_s}\tanh\left(\frac{s_\phi}{\varepsilon}\right) - \frac{K_\phi}{T_s}s_\phi = \dot{\Delta\phi} \qquad (8)$$

In (8), Q_ϕ and K_ϕ are the gains, ε is the width of the boundary layer. Equating (8) and (5) and solving for $\Delta\phi$ yields the following;

$$\Delta\phi = \eta_\phi N_\phi + Q_\phi \tanh\left(\frac{s_\phi}{\varepsilon}\right) + K_\phi\Delta\phi \qquad (9)$$

With the solution given in (9), the update dynamics is forced to behave as that defined by (8), which is actually a stable dynamics defined by the adopted switching function. In the derivations presented below, a key point is the fact that the system described by (5) is driven by η, which is known as learning rate in the related literature. Now we demonstrate that some special selection of this quantity simplifies the parameter stabilizing rule. At this point, one should notice that the equality in (9) is required for the convergence in the adjustable parameters. However, the objective is also to minimize the realization cost. For this purpose, learning rate selection will be made first and the parameter stabilizing part of the training information will be derived. Define the following quantity;

$$A_\phi = Q_\phi \tanh\left(\frac{\Delta\phi}{\varepsilon}\right) + K_\phi\Delta\phi \qquad (10)$$

Now we have a model described by (5), and an equality formulated by (9). If one chooses a positive definite Lyapunov function as given by (11), the time derivative of this function must be negative definite for stability of parameter change ($\Delta\phi$) dynamics. Clearly the stability in parameter change space implies the convergence in system parameters.

$$V = \frac{1}{2}s_\phi^2 = \frac{1}{2}(\Delta\phi)^2 \qquad (11)$$

$$\dot{V} = (\Delta\phi)(\dot{\Delta\phi}) \qquad (12)$$

If (5) and (9) are substituted into (12), the constraint stated in (13) is obtained for stability in the Lyapunov sense.

$$\eta_\phi^2 + \frac{1}{N_\phi}(A_\phi - \Delta\phi)\eta_\phi - \frac{1}{N_\phi^2}A_\phi\Delta\phi < 0 \qquad (13)$$

Equation (13) can be rewritten in a more tractable form as follows.

$$\left(\eta_\phi + \frac{1}{N_\phi}A_\phi\right)\left(\eta_\phi - \frac{1}{N_\phi}\Delta\phi\right) < 0 \qquad (14)$$

Since A_ϕ and $\Delta\phi$ have the same signs, the roots of the expression (14) clearly have opposite signs. The expression on the left-hand side assumes negative values between the roots. Therefore, in order to satisfy the inequality (14), the learning rate must satisfy the constraint given by (15).

$$0 < \eta_\phi < \min\left\{\left|\frac{1}{N_\phi}\Delta\phi\right|, \left|-\frac{1}{N_\phi}A_\phi\right|\right\} \qquad (15)$$

In (15), the interval of learning rate is restricted to positive values. This is due to preserve the compatibility between the gradient technique and the proposed approach. An appropriate selection of η_ϕ could be as follows:

$$\eta_\phi = \beta \min\left\{\left|\frac{1}{N_\phi}\Delta\phi\right|, \left|-\frac{1}{N_\phi}A_\phi\right|\right\}, \quad 0 < \beta < 1 \qquad (16)$$

By substituting the learning rate formulated in (16) into the stabilizing solution given in (9), the stabilizing component of the parameter change formula reduces to (17).

$$\Delta\phi_{VSS} = \beta \min\left(|\Delta\phi|, |A_\phi|\right)\mathrm{sgn}\left(N_\phi\right) + A_\phi \qquad (17)$$

The law introduced in (17) minimizes the cost of stability, which is the Lyapunov function defined by (11). In the right hand side of (17), the term $\Delta\phi$ is the eventual value of the displacement yet to be determined. The question now reduces to the following; can the cost defined by (2) be minimized by this rule? The answer is obviously not, because the stabilizing information is derived from the displacement of parameter vector denoted by $\Delta\phi$, whereas the minimization of (2) is achieved when ϕ tends to ϕ^* (the optimal value of the parameter) regardless of what the displacement is. Therefore the rule formulated in (17) needs a final

modification. In order to minimize (2), the parameter change anticipated by ordinary gradient rule, which has been reviewed in the second section, should somehow be integrated into the final form of parameter update mechanism. As introduced in the second section, error backpropagation (EBP) evaluates a parameter change as given by (18).

$$\Delta\phi_{EBP} = \zeta \, N_{\phi} \tag{18}$$

where, ζ is the learning rate in the conventional sense, i. e. it is chosen from the interval (0,1). Combining the laws formulated in (17) and (18) in a weighted average, the final form of the parameter update law is obtained.

$$\Delta\phi = \frac{\alpha_1 \Delta\phi_{VSS} + \alpha_2 \Delta\phi_{EBP}}{\alpha_1 + \alpha_2} \tag{19}$$

The parameter update rule as given by (19) meets the objectives of both the parametric stabilization and the cost minimization. The parameter update formula given by (19) carries mixed information containing both the parametric convergence, which is introduced by VSS part, and the cost minimization, which is due to the gradient rule. The balancing in this mixture is left to the designer by an appropriate selection of the weights α_1 and α_2.

The approach devised has first been applied to the estimation of signals by cosine neural networks [12]. The results have clearly stipulated that the cost described by the squared estimation error is minimized in time but the when the parameter vector falls into a region where the sensitivity derivatives assume large values, a locally divergent behavior is observed.

In the next section, the neural network application of the algorithm is discussed.

4. APPLICATION TO ARTIFICIAL NEURAL NETWORKS

In this section, a multilayer perceptron is introduced as the intelligent controller, the parameters of which are to be updated by using the technique presented. In [13], Narendra and Parthasarathy demonstrate that this structure can effectively be used for identification and control purposes. In the conventional error backpropagation technique, propagating the output error back through the neural network, as illustrated in Fig. 1, minimizes the cost function given in (2). Based on the derivation presented in detail in [11], the delta values for each of the neurons belonging to output layer and hidden layers are evaluated as given by (20) and (21) respectively.

$$\delta_j^{k+1,p} = (d_j^p - f_j^{k+1,p}) \, \Psi'(S_j^{k+1,p}) \tag{20}$$

$$\delta_j^{k+1,p} = \left(\sum_{h=1}^{\#neurons_{k+2}} \delta_h^{k+2,p} \, w_{jh}^{k+1} \right) \Psi'(S_j^{k+1,p}) \tag{21}$$

In (20) and (21), superscripts denote the layer and pattern order whereas subscripts denote the position of the neuron in the layer it belongs. Besides, Ψ, S_j and $f_j^{k+1,p}$ are the neuronal nonlinearity, the net summation, which is to be passed through the nonlinearity and the output of the j[th] neuron in the output layer respectively. Having evaluated the delta values during the backward pass, the weight update rule described by (22) is applied for each training pair. The rule below is known as the error backpropagation method in the literature and is abbreviated as *EBP*.

$$\Delta w_{ij\,EBP}^{k} = \zeta_{ij}^{k} \, \delta_j^{k+1,p} \, o_i^{k,p} \tag{22}$$

The proposed approach estimates the following update value for parametric stability.

$$\Delta w_{ij\,VSS}^{k} = \beta \min\left(\left| \Delta w_{ij}^{k} \right|, \left| A_{w_{ij}^{k}} \right| \right) * \operatorname{sgn}\left(N_{w_{ij}^{k}} \right) + A_{w_{ij}^{k}} \tag{23}$$

The two update laws are then combined as a weighted average.

$$\Delta w_{ij}^{k} = \frac{\alpha_{1ij}^{k} \Delta w_{ij\,VSS}^{k} + \alpha_{2ij}^{k} \Delta w_{ij\,EBP}^{k}}{\alpha_{1ij}^{k} + \alpha_{2ij}^{k}} \tag{24}$$

The rule described by (24) is the final form of the update formula and is applied to all of the neurons in the network.

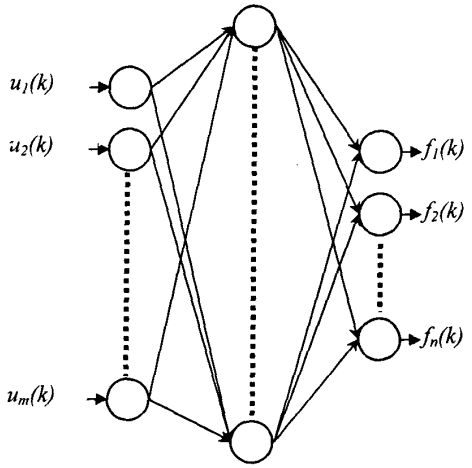

Figure 1. Architecture of a Feedforward Neural Network

5. PLANT MODEL

In this study, a two degrees of freedom direct drive SCARA robotic manipulator is used as the test bed. The physical view of the manipulator is illustrated in Fig. 2. Since the dynamics of such a mechatronic system is modeled by nonlinear and coupled differential equations, precise output tracking becomes a difficult objective due to the strong interdependency between the variables involved. Besides, the ambiguities on the friction related dynamics in the plant model make the design much more complicated. Therefore the methodology adopted must be intelligent in some sense. The general form of robot dynamics is described by (25) where $M(\theta)$, $V(\theta,\dot{\theta})$, $\tau(t)$ and f stand for the state varying inertia matrix, vector of coriolis terms, applied torque inputs and friction terms respectively. The plant parameters are given in Table 1 in standard units.

$$M(\theta)\ddot{\theta} + V(\theta,\dot{\theta}) = \tau - f \tag{25}$$

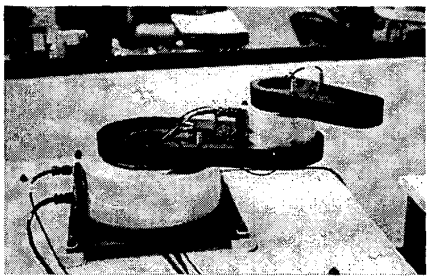

Figure 2. Physical View of the Manipulator

Table 1. Manipulator Parameters

Arm 1 Center of Gravity	0.1360	L_3
Arm 1 Inertia	0.3340	I_2
Arm 1 Length	0.3590	L_1
Arm 1 Mass	9.7800	M_2
Arm 2 Center of Gravity	0.1020	L_4
Arm 2 Inertia	0.0630	I_4
Arm 2 Length	0.2400	L_2
Arm 2 Mass	4.4500	M_4
Axis 1 Friction	5.3000	F_1
Axis 2 Friction	1.1000	F_2
Motor 1 Mass	73.000	M_1
Motor 1 Rotor Inertia	0.2670	I_1
Motor 2 Mass	14.000	M_3
Motor 2 Rotor Inertia	0.0075	I_3
Motor2 Stator Inertia	0.0400	I_{3C}
Payload Inertia	0.0000	I_P
Payload Mass	0.0000	M_p
Torque Limit 1	245.00	U_{max1}
Torque Limit 2	39.200	U_{max2}

If the angular positions and angular velocities are described as the state variables of the system, four coupled and first order differential equations can define the model. In (26) and (27), the terms seen in (25) are given explicitly.

$$M(\theta) = \begin{bmatrix} p_1 + 2p_3\cos(\theta_2) & p_2 + p_3\cos(\theta_2) \\ p_2 + p_3\cos(\theta_2) & p_2 \end{bmatrix} \tag{26}$$

$$V(\theta,\dot{\theta}) = \begin{bmatrix} -\dot{\theta}_2(2\dot{\theta}_1 + \dot{\theta}_2)p_3\sin(\theta_2) \\ \dot{\theta}_1^2 p_3\sin(\theta_2) \end{bmatrix} \tag{27}$$

In above, $p_1 = 2.0857$, $p_2 = 0.1168$ and $p_3 = 0.1630$. The details of the plant model are presented in [14].

6. SIMULATION STUDIES

In the simulations presented, the plant introduced in the Sec. 5 is controlled by the use of feedforward neural networks considered in Sec. 4. During the simulations, the main objective is to keep the update dynamics in a stable region. This is achieved through a suitable combination of conventional gradient based learning strategy with that based on the variable structure systems methodology.

The vector of error and rate of error for eack link are applied to the neurocontroller. In response to this input, the necessary torques are produced. Initially, the neural network is trained to realize a constant coefficient PD control surface. This phase is continued until the sum squared error decreases to 5e-6. Next, the on-line training phase takes place. During this phase, the weights and biases of the network are adjusted such that the controller reaches to the parametric stability while the tracking error is minimized for each link. In the simulations, a neural network, which has 4 inputs, 4 hidden neurons with hyperbolic tangent neuronal nonlinearities and 2 linear output neurons are employed. The control system architecture is illustrated in Fig. 3.

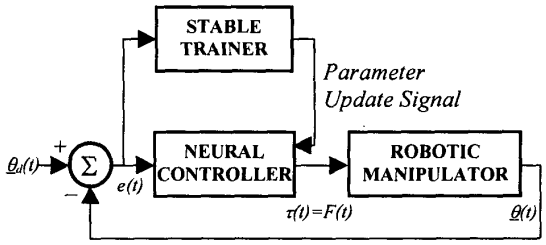

Figure 3. Control of a Robot Using the Proposed Training Method

The reference velocity trajectory described by (28) and depicted in Fig. 4 is used in simulations with zero initial errors.

$$\dot{\theta}_{d\,1,2} = \sin\frac{2\pi t}{5} \tag{28}$$

The behavior of the state tracking errors and applied torque inputs is illustrated in Figs. 5 and 6 respectively.

One should notice that the fluctuations in parameter space are damped out in an acceptable time. This fact is due to the use of VSS philosophy. The settings of the simulations are given in Table 2.

During the on-line training of the controller, the squared sum of parametric changes is defined to be the cost of stability. The cost function is described by (29), in which the summation is over all adjustable parameters of the neural network. The time behavior of the cost of stability is illustrated in Fig. 7.

$$J(t) = \sum_{\phi} (\Delta\phi(t))^2 \qquad (29)$$

As can be inferred from Fig. 7, the parametric stabilization performance of the proposed methodology is highly promising.

7. CONCLUSIONS

In this paper, a novel technique for improving the learning performance of artificial neural networks is presented. An approximate model of ordinary gradient based training procedure (error backpropagation) is constructed and variable structure systems approach is incorporated into the proposed form of the parameter update law. In this procedure, error backpropagation rule is responsible for the minimization of squared error while the variable structure systems based law is responsible for the stability in the parameter space.

The conventional approaches suffer from some handicaps, such as imperfect modeling, noisy observations or time varying parameters. If the effects of these factors are transformed to the cost hypersurface, whose dimensionality is determined by the adjustable design parameters, it is evident that the surface may have directions along which the sensitivity derivatives assume large values. In these cases, error backpropagation technique evaluates large parametric displacements, which can eventually lead to a locally divergent behavior. In control engineering practice, such a behavior constitutes a potential danger from a safety point of view. The approach presented in this paper takes care of the instantaneous fluctuations in the parameter space. Since the variable structure systems approach is well known with its robustness property, an appropriate combination of gradient rule and variable structure systems can eliminate the handicaps stated above. The fluctuations that are most likely to occur in the parameter space during training are dampened out. The combination is therefore a good candidate for efficient parameter tuning.

In the application example presented a feedforward neural network structure is utilized as the computationally intelligent architecture. It must be emphasized that the task is accomplished with only four hidden neurons in the ANN structure and the tuning is performed on-line. The results presented confirm the prominent features of the proposed approach. The algorithm is applicable to any neuro-fuzzy system model provided that the model output is differentiable with respect to the parameter of interest. However, the problem of getting stuck to a local minimum is still there because the VSS part of the approach enforces the parameters to settle down. Whenever the parameter vector gets trapped to a local minimum, the solution should be sought in the EBP part of the training algorithm.

Table 2. Simulation Parameters

T_s	2.5msec.
β	0.1
ζ^k_{ij}	0.01 for all i,j,k
$\alpha^k_{1ij,2ij}$	5.0
α^k_{2ij}	1.0
Q	0.1
K	0.1
ε	1.0
#Input Neurons	4
#Hidden Neurons	4
#Output Neurons	2

8. REFERENCES

[1] Hornik, K., "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, v.2, pp 359-366, 1989.

[2] Funahashi, K., "On the Approximate Realization of Continuous Mappings by Neural Networks," *Neural Networks*, v.2, pp 183-192, 1989.

[3] Gupta, M. M., D. H. Rao, "Dynamical Neural Units with Applications to the Control of Unknown Nonlinear Systems," *Journal of Intelligent and Fuzzy Systems*, v.1, no.1, pp. 73-92, 1993.

[4] Emelyanov, S. V., *Variable Structure Control Systems*, Moscow, Nauka, 1967.

[5] Hung, J. Y., W. Gao, J. C. Hung, "Variable Structure Control: A Survey," *IEEE Trans. on Industrial Electronics*, v.40, no. 1, pp.2-22, Feb. 1993.

[6] Gao, W., J. C. Hung, "Variable Structure Control of Nonlinear Systems: A New Approach," *IEEE Trans. on Industrial Electronics*, v.40, no. 1, pp.45-55, Feb. 1993.

[7] Kaynak, O., F. Harashima and H. Hashimoto, "Variable Structure Systems Theory, as Applied to Sub-time Optimal Position Control with an Invariant Trajectory," *Trans. IEE of Japan*, Sec. E, v.104, no.3/4, pp.47-52, 1984.

[8] Bekiroglu, N., "Adaptive Sliding Surface Design for Sliding Mode Control Systems," Ph.D. Thesis, Bogazici University, 1996.

[9] Byungkook, Y., W. Ham, "Adaptive Fuzzy Sliding Mode Control of Nonlinear Systems," *IEEE Trans. on Fuzzy Systems*, v.6, no.2, pp.315-321, May 1998.

[10] Erbatur, K., O. Kaynak, A. Sabanovic and I. Rudas, "Fuzzy Adaptive Sliding Mode Control of a Direct Drive Robot," *Robotics and Autonomous Systems*, v.19, no:2, pp.215-227, Dec. 1996.

[11] Rumelhart, D. E., G. E. Hinton and R. J. Williams, "Learning Internal Representations by Error Propagation," in D. E. Rumelhart and J. L. McClelland, eds., *Parallel Distributed Processing*, v.1, pp.318-362, MIT Press, Cambridge, M.A., 1986.

[12] Efe, M.O., Kaynak, O., and Keresteciogiu, F., "Intelligent Signal Estimation Using Cosine Neural Networks with Variable Structure Systems Based Training Procedure." The 2nd Int. Conf on Recent Advances in Mechatronics (ICRAM'99), May 24 - 26, Istanbul, Turkey, pp.57-61, 1999.

[13] Narendra, K. S., K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Trans. on Neural Networks*, v.1, no. 1, pp.4-27, Mar. 1990.

[14] Direct Drive Manipulator R&D Package User Guide, Integrated Motions Incorporated, 704 Gillman Street, Berkeley, California 94710, U.S.A.

9. ACKNOWLEDGMENTS

This work is supported by the Bogazici University Research Fund (Grant No: 99A202) and the grant of TUBITAK, Turkey and OFMB, Hungary.

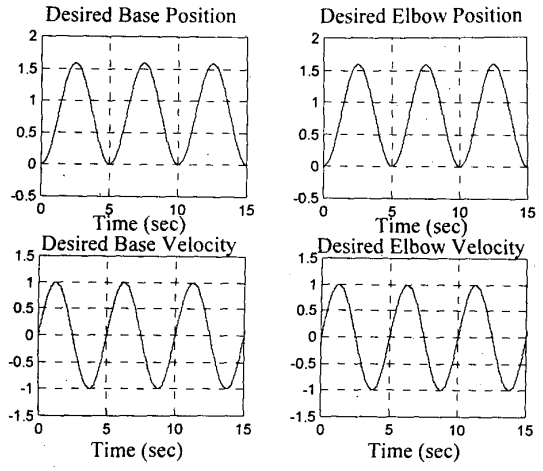

Figure 4. Reference Position and Velocity Trajectories

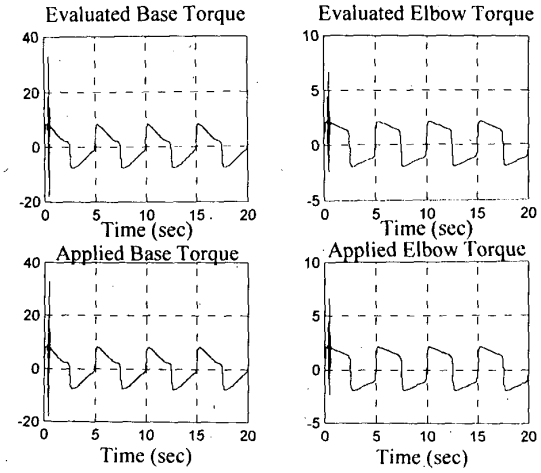

Figure 6. Evaluated and Applied Torque Inputs

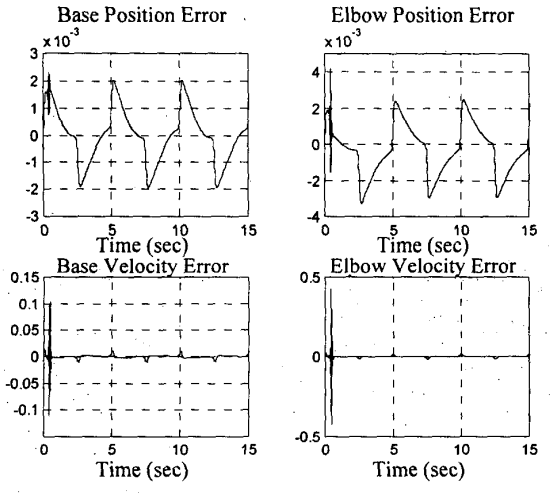

Figure 5. State Tracking Errors

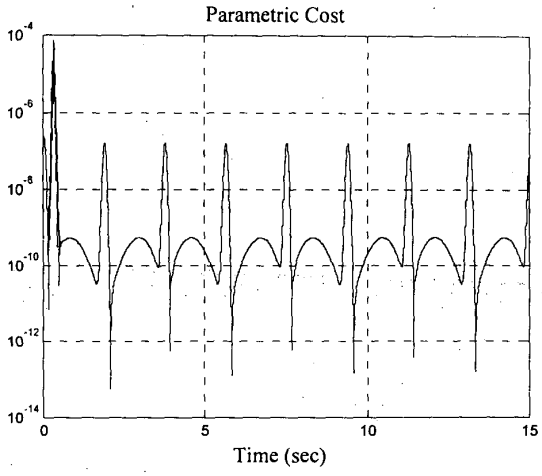

Figure 7. Time Behavior of the Stability Cost of Operation