# Stable Training of Computationally Intelligent Systems by Using Variable Structure Systems Technique

M. Onder Efe, *Student Member, IEEE*, Okyay Kaynak, *Senior Member, IEEE*, and Bogdan M. Wilamowski, *Senior, Member IEEE*

***Abstract*—This paper presents a novel training algorithm for computationally intelligent architectures, whose outputs are differentiable with respect to the adjustable design parameters. The algorithm combines the gradient descent technique with the 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 and to the environmental disturbances. 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 develops a heuristic that a computationally intelligent system, which may be a neural network architecture or a fuzzy inference mechanism, 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 in two different ways. In one, a standard fuzzy system architecture is used, whereas in the second, the arm is controlled by the use of a multilayer perceptron. In order to demonstrate the robustness of the approach presented, a considerable amount of observation noise and varying payload conditions are also studied.**

***Index Terms*—Computational intelligence, stabilization, variable-structure systems.**

## Nomenclature

| | |
|---|---|
| NFS | Neuro fuzzy system response. |
| $\phi$ | Generic parameter of neuro fuzzy system. |
| $\phi^*$ | Optimal value of the generic parameter. |
| $\Delta\phi$ | Change in parameter $\phi$. |
| $e$ | Observed output error. |
| $d$ | Desired output. |
| $J$ | Realization cost. |
| $J_s$ | Parametric cost. |
| $\eta_\phi$ | Learning rate for parameter $\phi$. |
| $T_s$ | Sampling interval of update dynamics. |
| $s_\phi$ | Switching function for parameter $\phi$. |
| $Q_\phi$ | Gain of the switching scheme. |
| $K_\phi$ | Gain of the switching scheme. |
| $\varepsilon$ | Boundary layer parameter |
| $N_\phi$. | Backpropagated error value for parameter $\phi$. |
| $\beta$ | Scaling factor for parameter-stabilizing law. |
| $\zeta$ | Learning rate for cost-minimizing law. |
| $V_\phi$ | Lyapunov function for parameter $\phi$. |
| $\alpha_i$ | Weighting factor. |
| $\mu_{ij}$ | Membership function of $i$th rule's $j$th input. |
| $c_{ij}$ | Center of membership function $\mu_{ij}$. |
| $u_j$ | $j$th input of computationally intelligent architecture. |
| $a_{ij}$, $b_{ij}$ | Shape parameters of membership function $\mu_{ij}$. |
| $w$ | Vector of firing strengths. |
| $w_n$ | Vector of normalized firing strengths. |
| $S_j^{k+1}$ | Net summation of the $j$th neuron in the $(k+1)$th layer. |
| $\psi$ | Neuronal nonlinear activation function. |
| $\delta_j^{k+1,p}$ | Delta value of the neuron in the $(k+1)$th layer for $p$th pattern. |
| $d_j^p$ | $j$th entry of target output for $p$th pattern. |
| $o_i^{k,p}$ | Output of the neuron in the $k$th layer in response to $p$th pattern. |
| $\theta_d(t)$ | Desired state trajectory. |
| $\theta(t)$ | Actual state trajectory. |

Manuscript received March 1, 1999; revised November 26, 1999. Abstract published on the Internet December 23, 1999. This work was supported by the Bogazici University Research Fund under Project 99A202.

M. O. Efe and O. Kaynak are with the Electrical and Electronic Engineering Department, Bogazici University, Bebek, 80815 Istanbul, Turkey (e-mail: efemond@boun.edu.tr; kaynak@boun.edu.tr).

B. M. Wilamowski is with the Department of Electrical Engineering, University of Wyoming, Laramie, WY 82701 USA (e-mail: wilam@ieee.org).

Publisher Item Identifier S 0278-0046(00)02493-X.

## I. Introduction

The 20th century has witnessed widespread innovations in both hardware and software design. In the first half of the century, the emphasis was mainly on the development of accurate mechanical component design, whereas in the second half, new technologies emerged together with new needs and new directions in industry. The development of fast microprocessors enabled the design and implementation of *expert–machine-interaction*-based computation environments. Ever-increasing needs brought about by the multidimensionality of the problem space and time-varying behavior of real-life physical systems further required to reduce the role of the expert and to increase the role of the machine. A natural consequence of this rapid growth is the emergence of the field of intelligent systems.

The word *intelligence* in this context should be understood in the sense of a machine's capability of self-adaptation (parametric), self-organization, and self-diagnostics (architectural) in the face of varying environmental conditions without external intervention. This clearly implies a large spectrum in the domain of intelligence. In this respect, the degree of autonomy gains a crucial importance. This quantity is referred to as *machine intelligence quotient* (MIQ) in the related literature. Conceptually, the degree of intelligence is closely related to the design methodology followed. The limits of the intelligent behavior are determined by the flexibility of the architecture, the ability to realize the human expertise, laws of inference procedure, and the speed of learning. All of these titles are the main constituents of the research area called *soft computing.*

Soft computing is a practical framework for solving complex problems through the use of human expertise and *a priori* knowledge about the problem at hand. Two of the main subtitles in this field are the following:

- artificial neural networks;
- fuzzy inference systems.

*Artificial neural networks* 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], and Runge–Kutta neural networks [4], [5] constitute typical structurally different models. The models mentioned have successfully been applied to problems extending from control applications to image/pattern recognition problems.

*Fuzzy inference systems* are the most popular constituent of the soft computing area since they are able to represent human expertise in the form of *IF antecedent THEN consequent* statements. In this domain, the system behavior is modeled through the use of linguistic descriptions. Although the earliest work by Zadeh on fuzzy systems has not been paid as much attention as it deserved in the early 1960's, since then, the methodology has become a well-developed framework. The typical architectures of fuzzy inference systems are those introduced by Wang [6], [7], Takagi, and Sugeno [8], and Jang [9]. In [6], a fuzzy system having Gaussian membership functions, product inference rule, and weighted average defuzzifier is constructed and has become the standard method in most applications. Takagi and Sugeno change the defuzzification procedure where dynamic systems are used in the defuzzification stage. The potential advantage of the method is that, under certain constraints, the stability of the system can be studied. Jang *et al.* [9] propose an adaptive neuro-fuzzy inference system, in which a polynomial is used as the defuzzifier. This structure is commonly referred to as ANFIS in the related literature. The choice concerning the order of the polynomial and the variables to be used in the defuzzifier are left to the designer.

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 provides 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 toward a desired dynamics, second, the system is maintained on that differential geometry. In the literature, the former dynamics are 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).

The earliest notion of SMC strategy was constructed on a second-order system in the late 1960's by Emelyanov [10]. The work stipulated that a special line could be defined on the phase plane, such that any initial state vector can be driven toward the plane and then be maintained on it, while forcing the error dynamics toward the origin. Since then, the theory has greatly been improved and the sliding line has taken the form of a multidimensional surface, called the *sliding surface*, around which a switching control action takes place.

Numerous contributions to VSS theory have been made during the last decade, and some of them are as follows. Hung *et al.* [11] have reviewed the control strategy for linear and nonlinear systems. In [11], the switching schemes, putting the differential equations into canonical forms, and generating simple SMC-based controls are considered in detail. Gao *et al.* [12], 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 the control system can be refined by interfacing it with an adaptation mechanism, which regularly redesigns the sliding surface [13], [14]. This eventually results in a robust control system. The performance of the 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 [15] and [16], fuzzy inference systems are proposed for the SMC scheme. A standard fuzzy system is studied and the relevant robustness analyses are carried out. Particularly, the work presented in [15] 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 [15] 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 computationally intelligent architectures, which enforces the adjustable parameters to settle down to a steady-state solution while minimizing an appropriate cost function. This is achieved through an appropriate combination of error backpropagation (EBP) algorithm [17] with VSS philosophy.

This paper is organized as follows. Section II briefly reviews the conventional error backpropagation. Section III 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. Section IV presents the application of the proposed scheme to a fuzzy inference system, while Section V deals with the application to a multilayer perceptron. In Section VI, the plant, which is a two-degrees-of-freedom direct-drive SCARA robotic manipulator, is presented. Next, the simulation results are discussed in Section VII. Conclusions constitute Section VIII.

## II. Gradient-Based Conventional Training Procedure

In this section, a widely used technique of parameter adjustment is briefly reviewed. The method was first formulated by Rumelhart *et al.* [17] in the 1980's. The approach has successfully been applied to a wide variety of optimization problems. The algorithm can be stated as follows:

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

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

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

The observation error in (1) is used to minimize the cost function in (2) by utilizing the rule described by (3)

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

The minimization proceeds recursively as given in (4), for which the sensitivity derivative with respect to the generic parameter $\phi$ 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.

## III. Derivation of the Parameter-Stabilizing Law by Using VSS Approach

A continuous-time dynamic model of the parameter update rule prescribed by the gradient descent technique can be written as in

$$\dot{\Delta\phi} = -\frac{1}{T_s}\Delta\phi + \frac{\eta_\phi}{T_s} N_\phi. \tag{5}$$

The above model is composed of the sampling time denoted by $T_s$, the gradient-based nonscaled parameter change denoted by $N_\phi = e\,[\partial \mathrm{NFS}(\phi, u)/\partial\phi]$, and a scaling factor denoted by $\eta_\phi$, for the selection of which a detailed analysis is presented in the subsequent discussion. Using Euler's first-order approximation for the derivative term, one obtains the following relation, which obviously validates the constructed model in (5) and which leads to the representation in (7):

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

$$\Delta\phi(k+1) = \eta_\phi N_\phi(k). \tag{7}$$

By comparing (4) and (7), the equivalency between the continuous and discrete forms of the update dynamics is, thus, clarified. The synthesis of the parameter-stabilizing component is based on the integration of the system in (5) with variable structure systems methodology. In the design of variable structure controllers, one method that can be followed is the reaching law approach [11]. For the use of this theory in the stabilization of the training dynamics, let us define the switching function as in (8) and its dynamics as in (9)

$$s_\phi = \Delta\phi \tag{8}$$

$$\dot{s}_\phi = -\frac{Q_\phi}{T_s}\tanh\left(\frac{s_\phi}{\varepsilon}\right) - \frac{K_\phi}{T_s} s_\phi \tag{9}$$

where $Q_\phi$ and $K_\phi$ are the gains, and $\varepsilon$ is the width of the boundary layer. In the derivations presented below, a key point is the fact that the system described by (5) is also driven by $\eta_\phi$, which is known as learning rate in the related literature. Now, we demonstrate that some special selection of this quantity leads to a rule that minimizes the magnitude of parametric displacement. Equating (9) and (5) and solving for $\Delta\phi$ yields the following relation:

$$\Delta\phi = \eta_\phi N_\phi + A_\phi \tag{10}$$

where

$$A_\phi = Q_\phi \tanh\left(\frac{\Delta\phi}{\varepsilon}\right) + K_\phi \Delta\phi. \tag{11}$$

The values of the learning rate imposed by (10) might be seen as the desired values at the first glance. However, this selection cancels out the backpropagated error value $N_\phi$ from (5), consequently, the update dynamics exactly behaves as that defined by the adopted switching function (9), which does not necessarily minimize the cost in (2). Therefore, the further analysis explores the restrictions on $\eta_\phi$, as well as the construction of the mixed training criterion.

Now, we have a model described by (5), and an equality to be enforced and formulated by (10). If one chooses a positive definite Lyapunov function as given by (12), 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

$$V_\phi = \frac{1}{2} s_\phi^2 = \frac{1}{2}(\Delta\phi)^2 \tag{12}$$

$$\dot{V}_\phi = (\Delta\phi)(\dot{\Delta\phi}). \tag{13}$$

If (5) and (10) are substituted into (13), the constraint stated in (14) is obtained for stability in the Lyapunov sense

$$\eta_\phi^2 + \frac{1}{N_\phi}(A_\phi - \Delta\phi)\eta_\phi - \frac{1}{N_\phi^2} A_\phi \Delta\phi < 0. \tag{14}$$

Equation (14) can be rewritten in a more tractable form as follows:

$$\left(\eta_\phi + \frac{1}{N_\phi} A_\phi\right)\left(\eta_\phi - \frac{1}{N_\phi} \Delta\phi\right) < 0. \tag{15}$$

Since $A_\phi$ and $\Delta\phi$ have the same signs, the roots of the expression (15) clearly have opposite signs. The expression on the left-hand side assumes negative values between the roots. Therefore, in order to satisfy the inequality in (15), the learning rate must satisfy the constraint given in (16)

$$0 < \eta_\phi < \min\left\{\left|\frac{1}{N_\phi}\Delta\phi\right|, \left|-\frac{1}{N_\phi}A_\phi\right|\right\}. \tag{16}$$

In (16), the interval of learning rate is restricted to positive values. This is due to preserve the compatibility between the traditional gradient based approaches and the proposed approach. An appropriate selection of $\eta_\phi$ could be as follows:

$$\eta_\phi = \beta \min\left\{\left|\frac{1}{N_\phi}\Delta\phi\right|, \left|-\frac{1}{N_\phi}A_\phi\right|\right\}, \qquad 0 < \beta < 1. \tag{17}$$

By substituting the learning rate formulated in (17) into the equality given in (10), the stabilizing component $\Delta\phi_{\mathrm{VSS}}$ of the parameter change formula is obtained as

$$\Delta\phi_{\mathrm{VSS}} = \beta \min(|\Delta\phi|, |A_\phi|)\,\mathrm{sgn}(N_\phi) + A_\phi \tag{18}$$

where $\Delta\phi$ on the right-hand side is the final update value yet to be obtained. The law introduced in (18) minimizes the cost of stability, which is the Lyapunov function defined by (12). The question now reduces to the following: can this law minimize the cost defined by (2)? The answer is obviously not, because the stabilizing criteria in (18) is derived from the displacement of the parameter vector denoted by $\Delta\phi$, whereas the minimization of (2) is achieved when $\phi$ tends to $\phi^*$, regardless of what the displacement is. In order to minimize (2), the parameter change anticipated by gradient-based optimization technique, which is reviewed in the second section, should somehow be integrated into the final form of parameter update mechanism. As introduced in the second section, the error backpropagation algorithm (EBP) evaluates a parameter change as given in (19)

$$\Delta\phi_{\mathrm{EBP}} = \zeta N_\phi \tag{19}$$

where $\zeta$ is the constant learning rate in the conventional sense. It is reasonable to expect that a combination of the laws formulated in (18) and (19) in a weighted average will meet the objectives of both the parametric stabilization and the cost minimization. (The stability analysis of the mixed update rule has been completed. Due to its length, it will appear on another paper.) The parameter update rule will then be as in (20)

$$\Delta\phi = \frac{\alpha_1 \Delta\phi_{\mathrm{VSS}} + \alpha_2 \Delta\phi_{\mathrm{EBP}}}{\alpha_1 + \alpha_2}. \tag{20}$$

The parameter update formula given by (20) carries a mixed displacement value containing both the parametric convergence, which is introduced by the VSS part, and the cost minimization, which is due to the error backpropagation technique. The balancing in this mixture is left to the designer by an appropriate selection of $\alpha_1$ and $\alpha_2$.

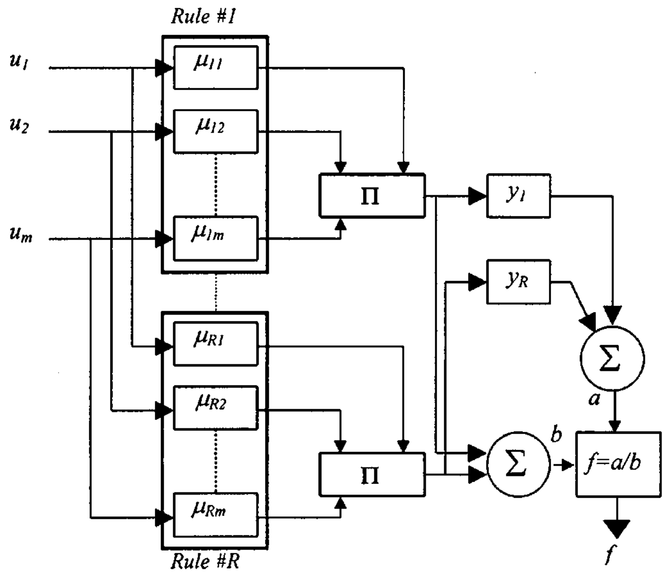


Fig. 1. Architecture of the standard fuzzy system.

## IV. Application to Standard Fuzzy Systems

This section considers the standard fuzzy system approach introduced in [6] as the computationally intelligent architecture. The system that is considered in this paper uses bell-shaped membership functions as described by (21)

$$\mu_{ij}(u_j) = \frac{1}{1 + \left|\dfrac{u_j - c_{i,j}}{a_{ij}}\right|^{2b_{ij}}} \tag{21}$$

where

| | |
|---|---|
| $c_{ij}$ | defines the center of membership function; |
| $a_{ij}$ | characterizes the slope of the function; |
| $b_{ij}$ | characterizes the flatness of the function. |

The structure of the fuzzy system is illustrated in Fig. 1, for which the following type of a rule base structure is adopted:

IF $u_1$ is $U_1$ AND $u_2$ is $U_2$ AND $\cdots$ AND $u_m$ is $U_m$
THEN $f = y_i$.

In the IF part of this representation, the lowercase variables denote the inputs and the uppercase variables stand for the fuzzy sets corresponding to the domain of each linguistic label.

During the simulations, $c_{ij}$, $a_{ij}$, and $b_{ij}$ parameters are kept constant and the adaptation is carried out on the $y$ parameters of the defuzzifier. The initial values of the membership functions are selected such that the region of interest is covered appropriately.

The overall realization performed by the system considered is given in (22), where the weighted average defuzzifier is used with algebraic product aggregation method

$$f = \frac{\sum_{i=1}^{\#\,\mathrm{Rules}} y_i \prod_{j=1}^{\#\,\mathrm{Inputs}} \mu_{ij}(u_j)}{\sum_{i=1}^{\#\,\mathrm{Rules}} \prod_{j=1}^{\#\,\mathrm{Inputs}} \mu ij(u_j)} = \sum_{i=1}^{\#\,\mathrm{Rules}} y_i w_{ni}. \tag{22}$$

In (22), the vector of firing strengths denoted by $w$ is normalized and the resulting vector is represented by $w_n$

$$w_{ni} = \frac{\prod_{j=1}^{\#\,\mathrm{Inputs}} \mu_{ij}(u_j)}{\sum_{k=1}^{\#\,\mathrm{Rules}} \prod_{j=1}^{\#\,\mathrm{Inputs}} \mu_{ij}(u_j)}. \tag{23}$$

With the definition given by (23), and the realization described by (22), the adjustable parameter set is selected as the $y$ parameters of the defuzzifier. The backpropagated error measure can now be formulated as given by

$$N_{yi} = e\frac{\partial \mathrm{NFS}(y_i, u)}{\partial y_i} = e\frac{\partial f}{\partial y_i} = e w_{ni}. \tag{24}$$

By construction of the algorithm presented, the internal parameter $A_{yi}$ is defined as follows:

$$A_{yi} = Q_{yi} \tanh\left(\frac{\Delta y_i}{\varepsilon}\right) + K_{yi}\Delta y_i. \tag{25}$$

The parameter $\varepsilon$ that defines the boundary layer is selected as unity for all adjustable parameters and for all simulations presented in this paper. The parameter-stabilizing law defined in (18) imposes the update rule formulated in (26), whereas the cost-minimizing update rule, which is the ordinary error backpropagation method, predicts the necessary parameter change value as described by (27). The final form of the update rule proposed can now be formulated as a weighted average of these two values. This is described by (28)

$$\Delta y_{i\,\mathrm{VSS}} = \beta \min(|\Delta y_i|, |A_{yi}|)\,\mathrm{sgn}(N_{yi}) + A_{yi} \tag{26}$$

$$\Delta y_{i\,\mathrm{EBP}} = \zeta N_{yi} \tag{27}$$

$$\Delta y_i = \frac{\alpha_1 \Delta y_{i\,\mathrm{VSS}} + \alpha_2 \Delta y_{i\,\mathrm{EBP}}}{\alpha_1 + \alpha_2}. \tag{28}$$

## V. Application to Multilayer Perceptron

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 [18], 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, whose structure is illustrated in Fig. 2, minimizes the cost function given in (2). Based on the derivation presented in detail in [9] and [17], the delta values for the neurons belonging to the output layer and the hidden layers are evaluated as given by (29) and (30), respectively

$$\delta_j^{k+1,p} = (d_j^p - f_j^p)\Psi'(S_j^{k+1,p}) \tag{29}$$

$$\delta_j^{k+1,p} = \left(\sum_{h=1}^{\#\,\mathrm{neurons}_{k+2}} \delta_h^{k+2,p} w_{jh}^{k+1}\right)\Psi'(S_j^{k+1,p}). \tag{30}$$

Having evaluated the delta values during the backward pass, the gradient-based weight update rule described by (31) is applied for each training pair

$$\Delta w_{ij\,\mathrm{EBP}}^k = \zeta \delta_j^{k+1,p} o_i^{k,p}. \tag{31}$$

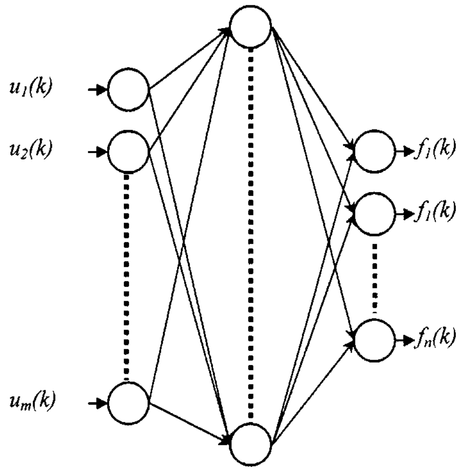


Fig. 2. Architecture of multilayer perceptron.

The VSS part of the proposed approach estimates the following update value for parametric stability:

$$\Delta w_{ij\,\mathrm{VSS}}^k = \beta \min(|\Delta w_{ij}^k|, |A_{w_{ij}}^k|)\,\mathrm{sgn}(N_{w_{ij}}^k) + A_{w_{ij}}^k. \tag{32}$$

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

$$\Delta w_{ij}^k = \frac{\alpha_1 \Delta w_{ij\,\mathrm{VSS}}^k + \alpha_2 \Delta w_{ij\,\mathrm{EBP}}^k}{\alpha_1 + \alpha_2}. \tag{33}$$

## VI. Plant Model

In this paper, a two-degrees-of-freedom direct-drive robotic manipulator, which is illustrated in Fig. 3, is used as the test bed. 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 (34) where $M(\theta)$, $V(\theta, \dot\theta)$, $\tau$, and $f_c$ 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 I in standard units

$$M(\theta)\ddot\theta + V(\theta, \dot\theta) = \tau - f_c. \tag{34}$$

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 (35) and (36), the terms seen in (34) are given explicitly

$$M(\theta) = \begin{bmatrix} p_1 + 2p_3\cos(\theta_2) & p_2 + p_3\cos(\theta_2) \\ p_2 + p_3\cos(\theta_2) & p_2 \end{bmatrix} \tag{35}$$

$$V(\theta, \dot\theta) = \begin{bmatrix} -\dot\theta_2(2\dot\theta_1 + \dot\theta_2)p_3\sin(\theta_2) \\ \dot\theta_1^2 p_3 \sin(\theta_2) \end{bmatrix}. \tag{36}$$

In the above, $p_1 = 2.0857 + 0.0576M_p$, $p_2 = 0.1168 + 0.0576M_p$, and $p_3 = 0.1630 + 0.0862M_p$. Here, $M_p$ denotes the payload mass. The details of the plant model are presented in [19].

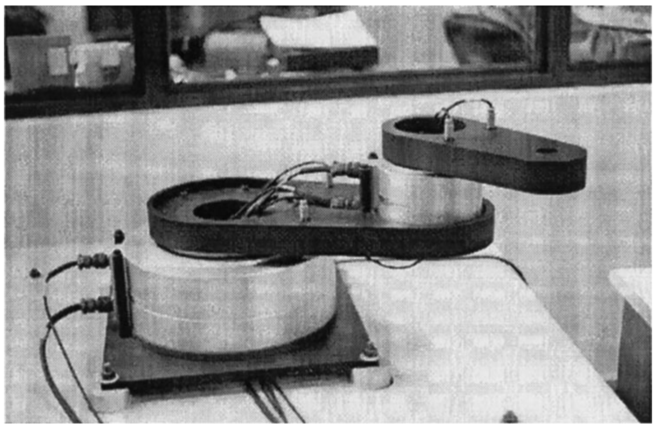


Fig. 3. Physical view of the direct-drive robotic manipulator.

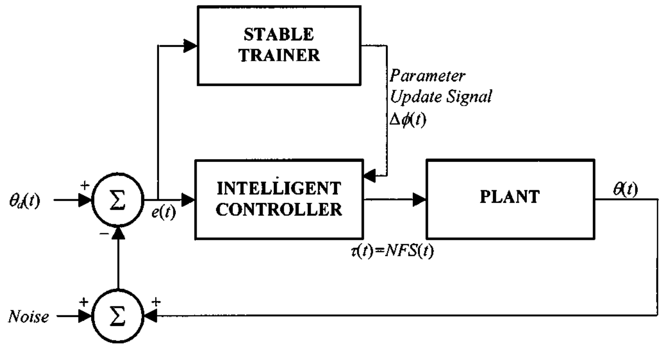


Fig. 4. Control of a plant using the proposed training method.

TABLE I
MANIPULATOR PARAMETERS

| | | | | | |
|---|---|---|---|---|---|
| Motor 1 Rotor Inertia | 0.2670 | $I_1$ | Payload Mass | See text | $M_p$ |
| Arm 1 Inertia | 0.3340 | $I_2$ | Arm 1 Length | 0.3590 | $L_1$ |
| Motor 2 Rotor Inertia | 0.0075 | $I_3$ | Arm 2 Length | 0.2400 | $L_2$ |
| Motor 2 Stator Inertia | 0.0400 | $I_{3C}$ | Arm 1 Center of Gravity | 0.1360 | $L_3$ |
| Arm 2 Inertia | 0.0630 | $I_4$ | Arm 2 Center of Gravity | 0.1020 | $L_4$ |
| Motor 1 Mass | 73.000 | $M_1$ | Axis 1 Friction Bound | 5.3000 | $f_{c1}$ |
| Arm 1 Mass | 9.7800 | $M_2$ | Axis 2 Friction Bound | 1.1000 | $f_{c2}$ |
| Motor 2 Mass | 14.000 | $M_3$ | Torque Limit 1 | 245.00 | |
| Arm 2 Mass | 4.4500 | $M_4$ | Torque Limit 2 | 39.200 | |

## VII. Simulation Studies

Two sets of simulation studies are presented. In the first set, the plant introduced in the Section VI is controlled by the standard fuzzy system considered in Section IV. The second set demonstrates how the same task can be accomplished by utilizing artificial neural networks discussed in Section V.

During the simulations, the main objective is to achieve precise state tracking together with small parameter update effort. This is achieved through a suitable combination of conventional gradient-based learning strategy with that based on the VSS methodology.

The reference angular position and velocity profiles used in all simulations are depicted in Fig. 5. The simulations are started with initial rest conditions.

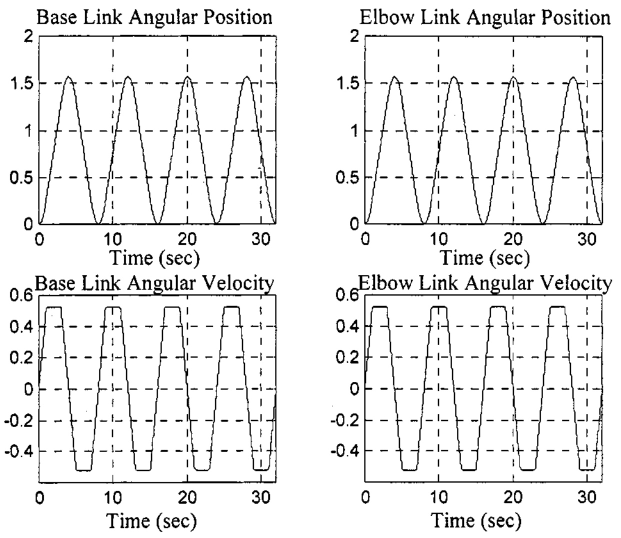


Fig. 5. Reference position and velocity trajectories.

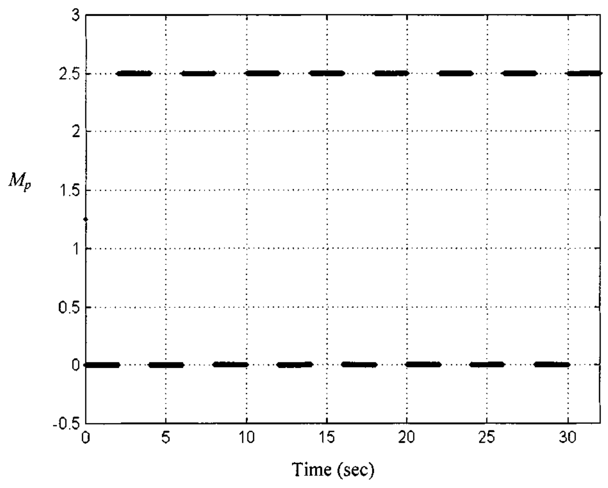


Fig. 6. Time behavior of the load mass.

In order to demonstrate the robustness property of the approach discussed, a payload of 2.5 kg is regularly grasped and released by the robot. The time behavior of the payload conditions is demonstrated in Fig. 6. Another difficulty to be alleviated by the algorithm discussed is the observation noise. It is assumed that the encoders provide noisy measurements to the controller. The noise sequence is Gaussian distributed and has the same statistical properties for all four state variables, namely, each sequence has zero mean and variance equal to 0.33$e$-6. The perturbing signal is illustrated in Fig. 7. It is expected that the stabilizing forces created on the adjustable design parameters will lead to the elimination of the adverse effects of the noisy observations, which excites the high-frequency dynamics of the learning algorithm. Therefore, the results obtained will enable the designer to make a fair comparison between the pure gradient descent and the proposed combination, especially in the sense of rejecting the high-frequency components entering into the training dynamics.

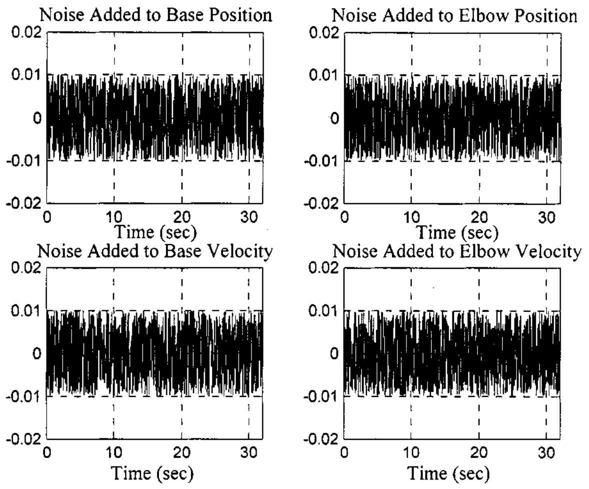


Fig. 7. The noise sequence added to the state variables.

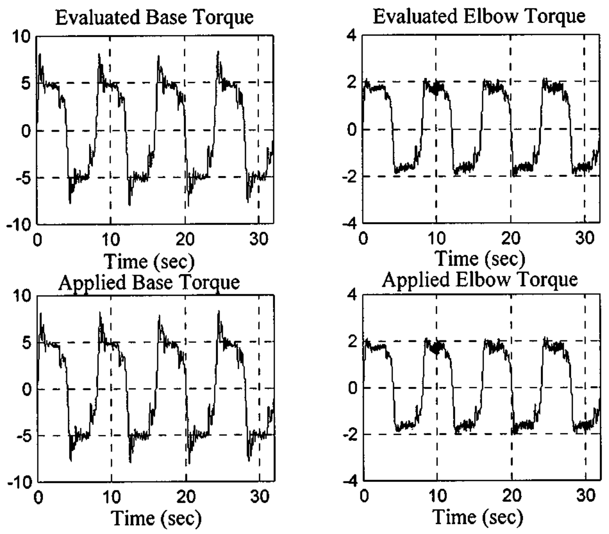


Fig. 10. Applied torque inputs in fuzzy control with VSS-based approach.

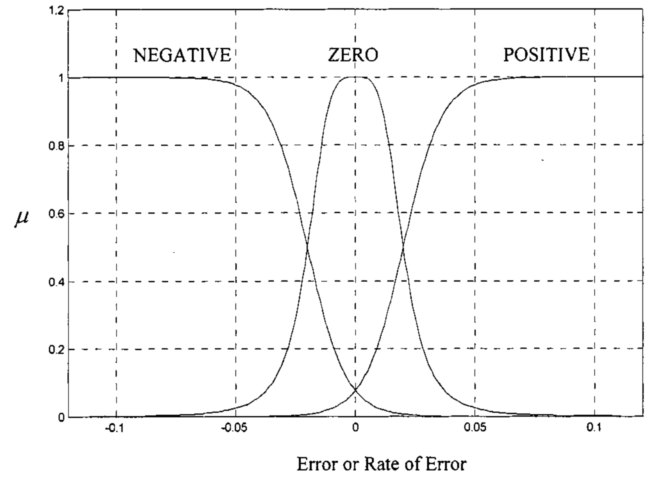


Fig. 8. Definitions of membership functions.

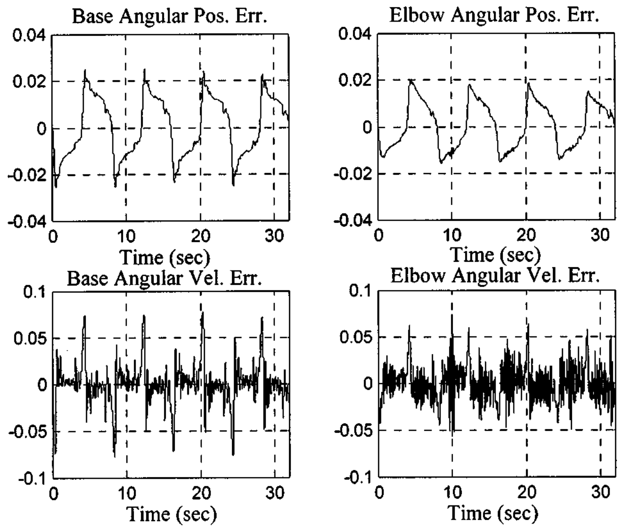


Fig. 9. State tracking errors in fuzzy control with VSS-based approach.

In the training of the controller structures discussed in the paper, the squared sum of parametric changes is defined to be the cost of stability

$$J_s(t) = \sum_{\phi} (\Delta\phi(t))^2. \tag{37}$$

For the case of the fuzzy controller, since there are two controllers, the summation in (37) is over the adjustable parameter set of both of the controllers, whereas, in the neural control application, there is only one structure and the summation in (37) runs over all the adjustable weights and biases of the neural network.

The first set of simulations is on the control of the manipulator by using the standard fuzzy system architecture. In this part, only the defuzzifier parameters are adjusted during the learning process and the membership functions are kept constant. The choice on the initial values of the membership function parameters is made by trial and error. Fuzzy quantization of the input variables is illustrated in Fig. 8. The state tracking errors and applied torque inputs are depicted in Figs. 9 and 10, respectively. It is evident from Fig. 9 that the proposed combination results in precise state tracking under the existence of environmental disturbances stated above. Furthermore, Fig. 10 emphasizes the control signal evaluated by the controller lies within the limits of applicable control range. Therefore, the signal is directly applied to the manipulator without requiring saturation. The behavior of the total parametric cost is figured out in Fig. 11, which clearly indicates the parameter-stabilizing property of the approach presented. For the use of the proposed algorithm, $\alpha_1$ is set to 20 while $\alpha_2$ is equal to 1. The response of the manipulator under the same conditions but only with gradient descent technique ($\alpha_1 = 0$) is illustrated in Fig. 12. As is clearly seen, after 15 s, the method diverges and the controller produces nonapplicable controls. Due to the space limit, only the behavior of the system under control is depicted. The simulation settings of this application are tabulated in Table II.

In the second set of simulations, a feedforward neural network structure is used as the controller. As in the case of fuzzy control, state tracking errors are fed to the controller as inputs

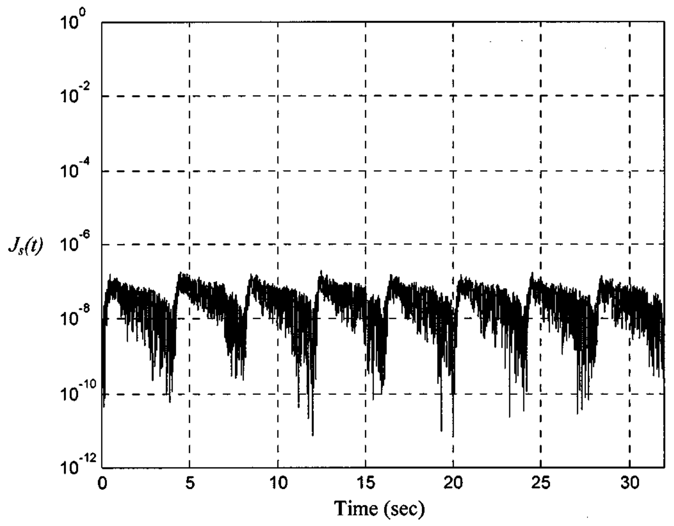


Fig. 11. Time behavior of the parametric cost in fuzzy control with VSS-based approach.

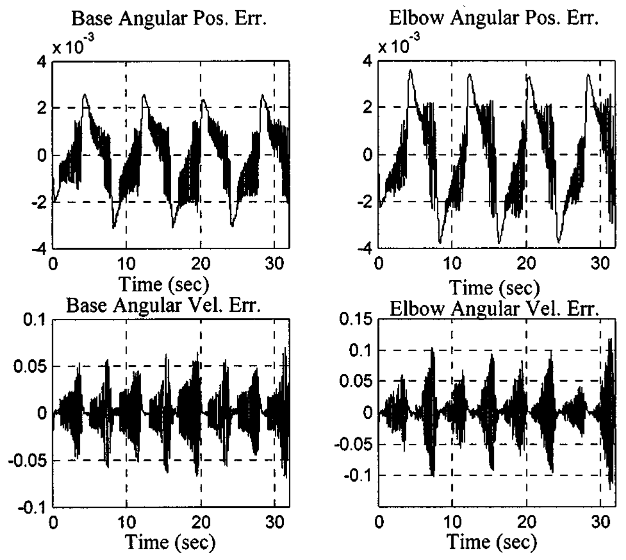


Fig. 13. State tracking errors in neural control with VSS-based approach.

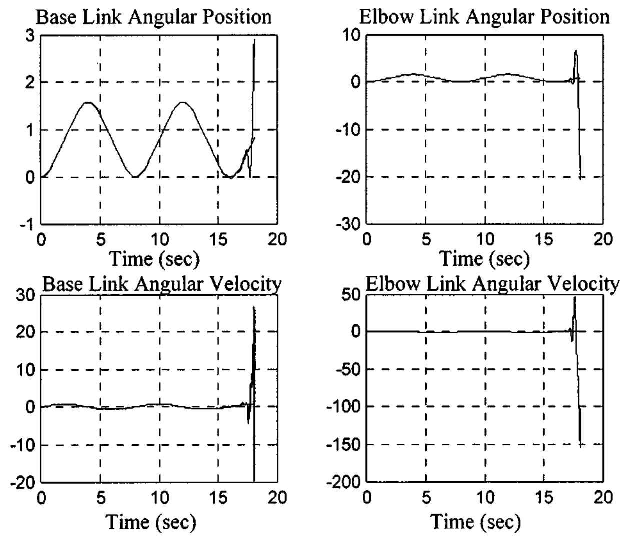


Fig. 12. Response of the manipulator with pure gradient descent.

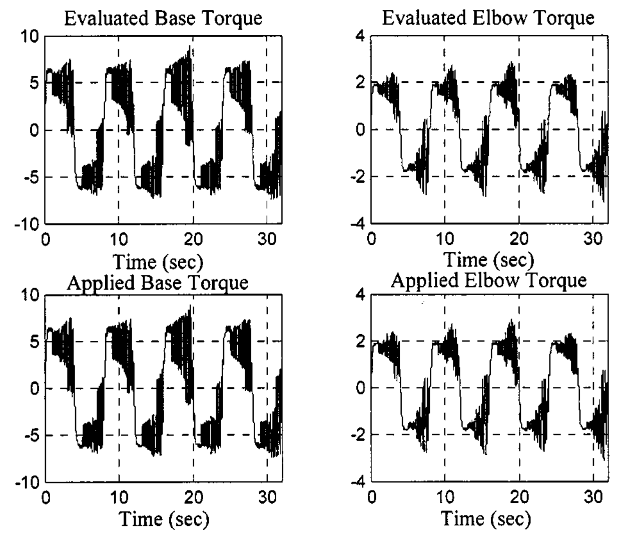


Fig. 14. Applied torque inputs in neural control with VSS-based approach.

TABLE II
SETTINGS USED IN FUZZY CONTROL

| | |
|---|---|
| $T_s$ | 1.0 msec. |
| $\beta$ | 0.1 |
| $\zeta$ | 0.3 |
| $\alpha_1$ | See Sec. 7 |
| $\alpha_2$ | 1.0 |
| $Q$ | 0.1 |
| $K$ | 0.1 |
| $\varepsilon$ | 1.0 |
| *#Rules* | 9 (for each link) |
| *#FIS Inputs* | 2 (for each link) |

and the desired trajectory is the one used in the fuzzy control example. The same environmental disturbances and payload conditions are used for the neural control application. For the simulations, a neural network, which has four inputs, eight hidden

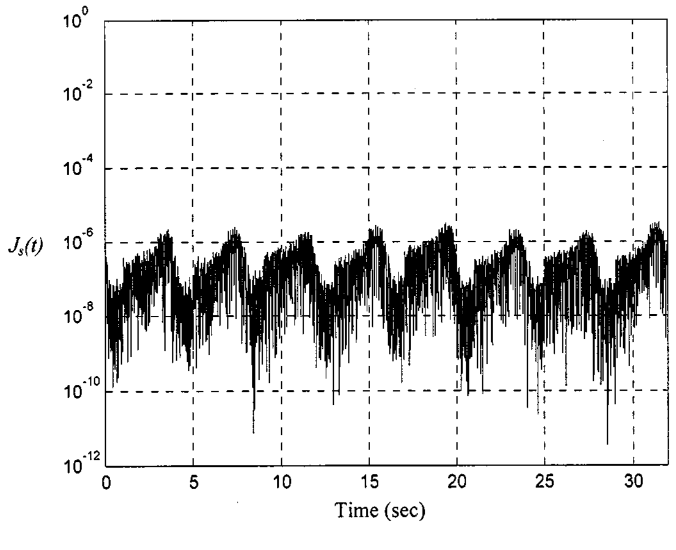


Fig. 15. Time behavior of the parametric cost in neural control with VSS-based approach.

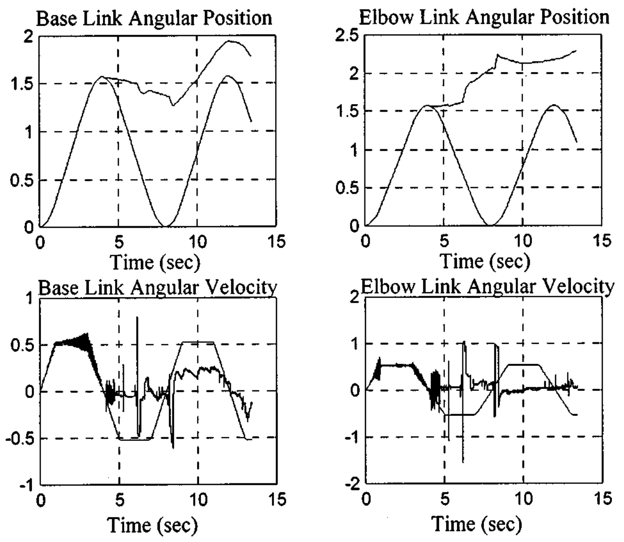


Fig. 16. Response of the manipulator with pure gradient descent.

TABLE III
SETTINGS USED IN NEURAL CONTROL

| | |
|---|---|
| $T_s$ | 1.0 msec. |
| $\beta$ | 0.01 |
| $\zeta$ | 0.01 |
| $\alpha_1$ | See. Sec. 7 |
| $\alpha_2$ | 1.0 |
| $Q$ | 0.1 |
| $K$ | 0.1 |
| $\varepsilon$ | 1.0 |
| *#Inputs* | 4 |
| *#Hidden Neurons* | 8 |
| *#Outputs* | 2 |

neurons with hyperbolic tangent neuronal nonlinearities, and two linear output neurons is used. The resulting behavior of the state tracking errors and applied torque inputs is depicted in Figs. 13 and 14, respectively. Again, a precise state tracking is observed under the adverse effects of the operating environment. The comments on the produced and applied control signals are the same as in the fuzzy control example. The time behavior of the total parametric cost is demonstrated in Fig. 15. For this case, the same weight values, namely, $\alpha_1 = 20$ and $\alpha_2 = 1$ are used in the proposed technique. The manipulator response with only using the gradient descent method $(\alpha_1 = 0)$ is depicted in Fig. 16. After approximately 4 s, the gradient descent diverges. The simulation settings of this application are tabulated in Table III.

## VIII. CONCLUSIONS

In this paper, a useful heuristic for improving the learning performance of computationally intelligent architectures has been presented. An approximate model of ordinary gradient-based training procedure is constructed and VSS approach is incorporated into the proposed form of the parameter update law. In this procedure, the error backpropagation rule is responsible for the minimization of squared error while the VSS-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 due to the difficulties mentioned. In these cases, error backpropagation technique evaluates large parametric displacements, which can eventually lead to a 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 parameter space. Since the VSS approach is well known with its robustness property, an appropriate combination of gradient rule and VSS can eliminate the handicaps stated. The undesired fluctuations that are most likely to occur in the parameter space during training are eliminated. The combination is, therefore, a good candidate for efficient parameter tuning.

Two application examples are elaborated, in which a standard fuzzy system and a feedforward neural network are utilized as the computationally intelligent architectures. The results presented confirm the prominent features of the proposed combination. The algorithm is applicable to any neuro-fuzzy system model, provided that the model output is differentiable with respect to the parameter of interest.

## REFERENCES

[1] K. Hornik, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989.
[2] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," *Neural Networks*, vol. 2, pp. 183–192, 1989.
[3] M. M. Gupta and D. H. Rao, "Dynamical neural units with applications to the control of unknown nonlinear systems," *J. Intell. Fuzzy Syst.*, vol. 1, no. 1, pp. 73–92, 1993.
[4] M. O. Efe and O. Kaynak, "A comparative study of neural network structures in identification of nonlinear systems," *Mechatron.*, vol. 9, no. 3, pp. 287–300, 1999.
[5] ——, "A comparative study of soft computing methodologies in identification of robotic manipulators," *Robot. Auton. Syst.*, vol. 23, no. 30, pp. 221–230, 2000.
[6] L. Wang, *Adaptive Fuzzy Systems and Control, Design and Stability Analysis*. Englewood Cliffs, NJ: PTR Prentice-Hall, 1994.
[7] L. Wang, *A Course in Fuzzy Systems and Control*. Englewood Cliffs, NJ: PTR Prentice-Hall, 1997.
[8] T. Takagi and M. Sugeno, "Fuzzy Identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. 15, pp. 116–132, Jan. 1985.
[9] J.-S. R. Jang, C.-T. Sun, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing*. Englewood Cliffs, NJ: PTR Prentice-Hall, 1997.
[10] S. V. Emelyanov, *Variable Structure Control Systems*. Moscow, U.S.S.R.: Nauka, 1967.
[11] J. Y. Hung, W. Gao, and J. C. Hung, "Variable structure control: A survey," *IEEE Trans. Ind. Electron.*, vol. 40, pp. 2–22, Feb. 1993.
[12] W. Gao and J. C. Hung, "Variable structure control of nonlinear systems: A new approach," *IEEE Trans. Ind. Electron.*, vol. 40, pp. 45–55, Feb. 1993.
[13] O. Kaynak, F. Harashima, and H. Hashimoto, "Variable structure systems theory, as applied to sub-time optimal position control with an invariant trajectory," *Trans. Inst. Elect. Eng. Jpn. E*, vol. 104, no. 3/4, pp. 47–52, 1984.

[14] N. Bekiroglu, "Adaptive sliding surface design for sliding mode control systems," Ph.D. dissertation, Dep. Elect. Electron. Eng., Bogazici University, 1996.

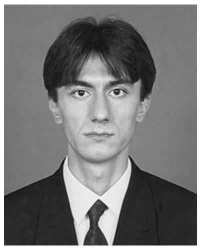
[15] Y. Byungkook and W. Ham, "Adaptive fuzzy sliding mode control of nonlinear systems," *IEEE Trans. Fuzzy Syst.*, vol. 6, pp. 315–321, May 1998.

[16] K. Erbatur, O. Kaynak, A. Sabanovic, and I. Rudas, "Fuzzy adaptive sliding mode control of a direct drive robot," *Robot. Auton. Syst.*, vol. 19, no. 2, pp. 215–227, Dec. 1996.

[17] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," *Parallel Distributed Processing*, vol. 1, pp. 318–362, 1986.

[18] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4–27, Mar. 1990.

[19] *Direct Drive Manipulator R&D Package User Guide*, Integrated Motions Inc., Berkeley, CA, 1992.



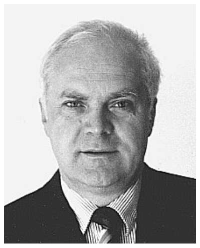
**M. Onder Efe** (S'95) received the B.Sc. degree from the Electronics and Communications Engineering Department, Istanbul Technical University, Istanbul, Turkey, and the M.S. degree from the Systems and Control Engineering Department, Bogazici University, Istanbul, Turkey, in 1993 and 1996, respectively. He is currently working toward the Ph.D. degree in the Electrical and Electronic Engineering Department, Bogazici University.

Since 1996, he has also been working at the Mechatronics Research and Application Center, Bogazici University, as a Research Assistant. His research interests include artificial neural networks, fuzzy inference systems, variable-structure control, and stabilizing training strategies for computationally intelligent systems and robotics.

**Okyay Kaynak** (M'80–SM'90) received the B.Sc. (first class honors) and Ph.D. degrees in electronic and electrical engineering from the University of Birmingham, Birmingham, U.K., in 1969 and 1972, respectively.

After spending seven years in industry, in January 1979, he joined the Department of Electrical and Electronics Engineering, Bogazici University, Istanbul, Turkey, where he is presently a Full Professor. He served as the Chairman of the Computer Engineering Department for three years and as the Director of the Biomedical Engineering Institute for one year. Currently, he is the Chairman of the Electrical and Electronic Engineering Department and the holder of the UNESCO Chair on Mechatronics and the Director of Mechatronics Research and Application Center. He has held long-term Visiting Professor/Scholar positions at various institutions in Japan, Germany, the U.S., and Singapore. His current research interests are in the field of intelligent control and mechatronics. He has published two books and edited two. He has also authored or coauthored more than 100 papers which have appeared in various journals and conference proceedings.

Dr. Kaynak is presently the Vice President of the IEEE Industrial Electronics Society and an Associate Editor of the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.



**Bogdan M. Wilamowski** (SM'83) received the M.S. degree in computer engineering, the Ph.D. in neural computing, and the D.Sc. degree in integrated circuit design from the Technical University of Gdansk, Gdansk, Poland, in 1966, 1970, and 1977, respectively.

He joined the Technical University of Gdansk in 1966 and became an Associate Professor in 1978 and a Professor in 1987. He was the Director of the Institute of Electronics from 1979 to 1981 and the Head of Solid State Electronics. He was with the Nishizawa Laboratory, Tohoku University, Japan, from 1968 to 1970, and he spent one year at the Semiconductor Research Institute, Sendai, Japan, as a JSPS Fellow during 1975–1976. He was a Visiting Scholar at Auburn University, Auburn, AL, during 1981–1982, a Visiting Professor at the University of Arizona, Tucson, during 1982–1984, and a Visiting Scholar at the Alabama Microelectronics, Science, and Technology Center, Auburn University, during 1995–1996. Since 1989, he has been with the Electrical Engineering Department, University of Wyoming, Laramie.

Dr. Wilamowski has served on the Organizing Committees for several IEEE conferences. Currently, he is an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS, IEEE TRANSACTIONS ON EDUCATION, and IEEE Industrial Electronics Society Newsletter. He is also Treasurer of the IEEE Industrial Electronics Society and a member of the IEEE Neural Networks Council.