ELSEVIER

Fuzzy Sets and Systems 122 (2001) 153–165

FUZZY sets and systems

www.elsevier.com/locate/fss

A novel optimization procedure for training of fuzzy inference systems by combining variable structure systems technique and Levenberg–Marquardt algorithm

M. Onder Efe, Okyay Kaynak *

Bogazici University, Electrical and Electronic Engineering Department, Bebek, 80815, Istanbul, Turkey

Received 12 March 1999; received in revised form 7 January 2000; accepted 14 February 2000

Abstract

This paper presents a novel training algorithm for fuzzy inference systems. The algorithm combines the Levenberg–Marquardt algorithm with variable structure systems approach. The combination is performed by expressing the parameter 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 to the unmodeled multivariable internal dynamics of Levenberg–Marquardt technique. 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 concerning the environmental conditions. This paper proves that 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). In the application example, control of a two degrees of freedom direct drive SCARA robotic manipulator is considered. As the controller, a standard fuzzy system architecture is used and the parameter tuning is performed by the proposed algorithm. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Fuzzy inference systems; Robotics

1. Introduction

Twentieth 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 expert and to increase the role of 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

* Corresponding author. Tel.: +90 212 287 2475; fax: +90 212 287 2465.

E-mail addresses: efemond@boun.edu.tr (M.O. Efe), Kaynak@boun.edu.tr (O. Kaynak).

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 alternative for solving complex problems through the use of human expertise and a priori knowledge about the problem in hand. Fuzzy Inference Systems are the most popular constituent of the soft computing area because of their ability 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 Prof. Zadeh on fuzzy systems has not been paid as much attention as it deserved in early 1960s, since then the methodology has become a well-developed framework. The typical architectures of fuzzy inference systems are those introduced by Wang [13,14], Takagi and Sugeno [12], and Jang [10]. In [14], 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. [10] 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 has always been in pursuit of a design, which provide accuracy 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 [5]. 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 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. [9] has reviewed the control strategy for linear and nonlinear systems. In [9], the switching schemes putting the differential equations into canonical forms and generating simple SMC-based controls are considered in detail. Gao et al. [7], 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 [1,11]. 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 [2,6], 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 [2] emphasizes that the robustness and stability properties of soft computing based control strategies can be studied through the use SMC theory. It is shown in the paper in this way 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 stable training procedure for fuzzy inference systems, which will force 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 Levenberg–Marquardt algorithm [8] with a parameter stabilizing law.

This paper is organized as follows: The second section summarizes the conventional method followed in Levenberg–Marquardt optimization technique. The third section presents the derivation of parameter stabilizing law. In the fourth section, a standard fuzzy system model is considered and the relevant formulation for the architecture is given. Next section is devoted to the plant to be controlled in this study. This is followed by the simulation studies. Conclusions constitute the last part of the paper.

2. Levenberg–Marquardt training method

Levenberg–Marquardt method is an approximation to Newton's method [8]. The algorithm uses the second-order derivatives of the cost function so that a better convergence behavior is observed. In the ordinary gradient descent search, only the first-order derivatives are evaluated and the parameter change information contains solely the direction along which the cost is minimized, whereas the Levenberg–Marquardt technique extracts a better parameter change vector. It is motivated by this problem that, on the cost surface, there may be many solutions leading to the convergence, raising the possibility of an excessively long time to reach to the solution. Using the notations given in Appendix, the algorithm can be stated as follows:

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

$$J = \tfrac{1}{2}e^2, \tag{2}$$

$$\Delta\phi = -(\nabla^2 J(\phi))^{-1}\nabla J(\phi), \tag{3}$$

where $\nabla^2 J(\phi)$ is the Hessian matrix and $\nabla J(\phi)$ is the gradient relevant to the cost of (2). The observation error in (1) is used to minimize the realization cost in (2) by utilizing the rule described by (3). The objective is to minimize instantaneous cost defined by (2). If the Taylor series expansion is applied to $e(\phi)$ around the operating point, the first derivatives result in the Jacobian given by (4):

$$J_s = \begin{bmatrix} \dfrac{\partial e_1}{\partial \phi_1} & \cdots & \dfrac{\partial e_1}{\partial \phi_B} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial e_L}{\partial \phi_1} & \cdots & \dfrac{\partial e_L}{\partial \phi_B} \end{bmatrix}. \tag{4}$$

In (4), B is the number of adjustable parameters and L is the number of outputs. The final form of the parameter update algorithm is described by (5) and the details are presented in [8].

$$\Delta\phi = N_\phi = -(J_s^{\mathrm{T}} J_s + qI)^{-1} J_s^{\mathrm{T}} e \tag{5}$$

For large q, the update formula given by (5) becomes the standard gradient descent with stepsize $1/q$; conversely for small q, the behavior is as that of Newton's method. Therefore, by the introduction of such a term, a smooth transition between Newton's method and steepest descent is achieved. Furthermore, this term introduces the elimination of invertibility problem in (5).

3. Derivation of the parameter stabilizing law by using variable structure systems approach

If the formula given in (5) 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 (6) by utilizing Euler's first-order approximation:

$$\dot{\Delta\phi} = -\frac{1}{T_s}\Delta\phi + \frac{\eta_\phi}{T_s}N_\phi. \tag{6}$$

In above, the evaluated parameter change is multiplied by a scaling factor denoted by η_ϕ, for the selection of which a detailed analysis is presented in the subsequent sections. It should be noted that since (6) is based on the update formula of (3), the term T_s drops out from the equations after the discretization and one ends up with (7):

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

In (7), the equivalency between the continuous and discrete forms of update dynamics is thus clarified. The synthesis of the parameter stabilizing component is based on the integration of the system in (6) with VSS methodology. In the design of variable structure controllers, one method that can be followed is the reaching law approach [9]. 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). Since the order of the system in (6) is one, the switching function in (8) is selected as a zero order one [9]; and it does not use any differentiated quantity. The design strategy in VSS technique necessitates the desired values of the system state, which can be denoted by $\Delta\phi_{\rm d}$. However, since the aim of the design is based on the minimization of parametric displacements in time, the desired value of the $\Delta\phi$ quantity is zero. Therefore the switching function in (8) suitably fulfills the design requirements of VSS strategy. In (9), the adopted reaching law is described. This selection corresponds to the constant plus proportional rate reaching mode dynamics. The details on the selection of reaching laws can be found in [9].

$$s_\phi = \Delta\phi - \Delta\phi_{\rm d} = \Delta\phi \tag{8}$$

$$\dot{s}_\phi = -\tilde{Q}_\phi \tanh(s_\phi/\varepsilon) - \tilde{K}_\phi s_\phi = \dot{\Delta\phi}. \tag{9}$$

Equating (9) and (6) and solving for $\Delta\phi$ yields the following:

$$\Delta\phi = \eta_\phi N_\phi + T_s\tilde{Q}_\phi \tanh(s_\phi/\varepsilon) + T_s\tilde{K}_\phi s_\phi. \tag{10}$$

Define the following quantities:

$$Q_\phi = T_s\tilde{Q}_\phi \quad \text{and} \quad K_\phi = T_s\tilde{K}_\phi, \tag{11}$$

$$\Delta\phi = \eta_\phi N_\phi + Q_\phi \tanh(\Delta\phi/\varepsilon) + K_\phi\,\Delta\phi. \tag{12}$$

The values of the η_ϕ imposed by (12) might be seen as the desired values at the first glance. However, this selection cancels out the cost minimizing quantity N_ϕ from (6), 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 η_ϕ as well as the construction of the mixed training criterion.

In the derivations presented below, a key point is the fact that the system described by (6) is driven by η_ϕ, which is known as learning rate in the related literature. Now we demonstrate that some special selection of this quantity leads to the parameter stabilizing rule. Let us define the following quantity for keeping analytic comprehensibility:

$$A_\phi = Q_\phi \tanh(\Delta\phi/\varepsilon) + K_\phi\,\Delta\phi. \tag{13}$$

Now we have a model described by (6), and an equality to be imposed and given by (12). If one chooses a positive definite Lyapunov function as in (14), the time derivative of this function must be negative definite for stability in the parameter change ($\Delta\phi$) space. Clearly the stability in parameter change space implies the convergence in system parameters.

$$V = \tfrac{1}{2}s_\phi^2 = \tfrac{1}{2}(\Delta\phi)^2, \tag{14}$$

$$\dot{V} = (\Delta\phi)(\dot{\Delta\phi}). \tag{15}$$

If (6) and (12) are substituted into (15), the constraint stated in (16) is obtained for stability in the Lyapunov sense.

$$\eta_\phi^2 + \frac{1}{N_\phi}(A_\phi - \Delta\phi)\eta_\phi - \frac{1}{N_\phi^2}A_\phi\,\Delta\phi < 0. \tag{16}$$

The inequality in (16) can be rewritten in a more tractable form as follows:

$$\left(\eta_\phi + \frac{1}{N_\phi}A_\phi\right)\left(\eta_\phi - \frac{1}{N_\phi}\Delta\phi\right) < 0. \tag{17}$$

Since A_ϕ and $\Delta\phi$ have the same signs, the roots of expression (17) clearly have opposite signs. The expression on the left-hand side assumes negative values between the roots. Therefore, in order to satisfy inequality (17), the learning rate must satisfy the constraint given by (18):

$$0 < \eta_\phi < \min\left\{\left|\frac{1}{N_\phi}\Delta\phi\right|, \left|-\frac{1}{N_\phi}A_\phi\right|\right\}. \tag{18}$$

In (18), the interval of learning rate is restricted to positive values. This is due to preserve the compatibility between the gradient-based approaches and the proposed approach. An appropriate selection of η_ϕ could be as follows:

$$\eta_\phi = \beta \min\left\{\left|\frac{1}{N_\phi}\Delta\phi\right|, \left|-\frac{1}{N_\phi}A_\phi\right|\right\}, \quad 0 < \beta < 1. \tag{19}$$

By substituting the learning rate formulated in (19) into the stabilizing solution given in (10), the stabilizing component of the parameter change formula can be obtained as in (20).

$$\Delta\phi_{\mathrm{VSS}} = \beta \min(|\Delta\phi|, |A_\phi|)\,\mathrm{sgn}(N_\phi) + A_\phi \tag{20}$$

where $\Delta\phi$ on the right-hand side is the final update value yet to be obtained. The law introduced in (20) minimizes the cost of stability, which is the Lyapunov function defined by (14). 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^* regardless of what the displacement is. Therefore, the rule formulated in (20) needs a final modification. In order to minimize (2), the parameter change anticipated by Levenberg–Marquardt 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, Levenberg–Marquardt algorithm (LM) evaluates a parameter change as given by (21):

$$\Delta\phi_{\mathrm{LM}} = N_\phi. \tag{21}$$

Combining the laws formulated in (20) and (21) in a weighted average, the parameter update law of (22) is obtained:

$$\Delta\phi = \frac{\alpha_1\,\Delta\phi_{\mathrm{VSS}} + \alpha_2\,\Delta\phi_{\mathrm{LM}}}{\alpha_1 + \alpha_2}. \tag{22}$$

The parameter update formula given by (22) carries mixed information containing both the parametric convergence, which is introduced by VSS part, and the cost minimization, which is due to the Levenberg–Marquardt technique. The balancing in this mixture is left to the designer by an appropriate selection of α_1 and α_2.

The global stability analysis of the approach presented can be found in [4]. In this reference, the control of a 3-DOF anthropoid robotic manipulator is achieved with artificial neural networks trained by the proposed technique.

4. Application to standard fuzzy systems

This section considers the standard fuzzy system approach introduced in [14] as the computationally intelligent architecture. The system that is considered in this study uses bell shaped membership functions as described by (23).

$$\mu_{ij}(u_j) = \frac{1}{1 + \left|\dfrac{u_j - c_{ij}}{a_{ij}}\right|^{2b_{ij}}} \tag{23}$$

In above, c_{ij} defines the center of ith rule's jth membership function, a_{ij} and b_{ij} characterize the slope and flatness of that function, respectively. The structure of 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* … *AND* u_m *is* U_m
THEN $F = y_i$

In this representation, lowercase variables denote the inputs; 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 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 (24), where weighted average defuzzifier is used with algebraic product aggregation

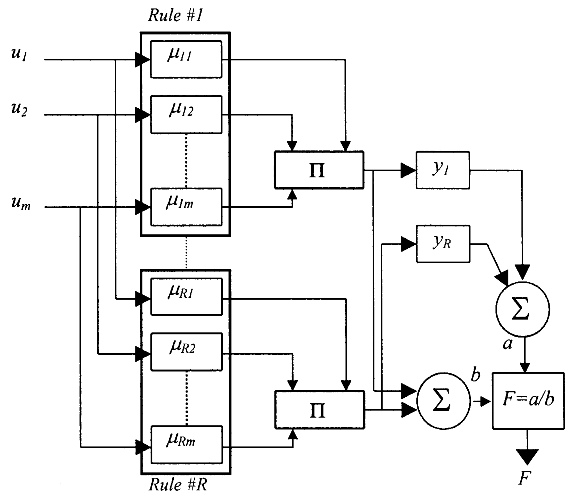

Fig. 1. Architecture of the standard fuzzy system.

method.

$$F = \frac{\sum_{i=1}^{\#Rules} y_i \prod_{j=1}^{\#Inputs} \mu_{ij}(u_j)}{\sum_{i=1}^{\#Rules} \prod_{j=1}^{\#Inputs} \mu_{ij}(u_j)} = \sum_{i=1}^{\#Rules} y_i w_{ni}. \tag{24}$$

In (24), the vector of firing strengths denoted by w is normalized and the resulting vector is represented by w_n.

$$w_{ni} = \frac{\prod_{j=1}^{\#Inputs} \mu_{ij}(u_j)}{\sum_{i=1}^{\#Rules} \prod_{j=1}^{\#Inputs} \mu_{ij}(u_j)}. \tag{25}$$

With the definition given in (25), and the realization described by (24), the adjustable parameter set is selected as the y parameters of the defuzzifier. The Jacobian can now be formulated as given by (26).

$$J_s = -[w_{n1}, w_{n2}, \dots, w_{nR}]. \tag{26}$$

The method presented in this paper uses solely the instantaneous observations contrary to what commonly adopted in the literature. Therefore, the Jacobian has only one row. By construction of the algorithm presented, the internal parameter A_y is defined as follows:

$$A_{yi} = Q \tanh\left(\frac{\Delta y_i}{\varepsilon_i}\right) + K\,\Delta y_i. \tag{27}$$

The parameter ε that defines the boundary layer is selected as unity for all adjustable parameters and for all simulations presented in this study. The parameter stabilizing law defined in (20) imposes the update rule formulated in (28), whereas the cost minimizing update rule, which is based on the Levenberg–Marquardt method, predicts the necessary parameter change value as described by (29). The final form of the update rule proposed can then be formulated as a weighted average of these two values. This is described by (30).

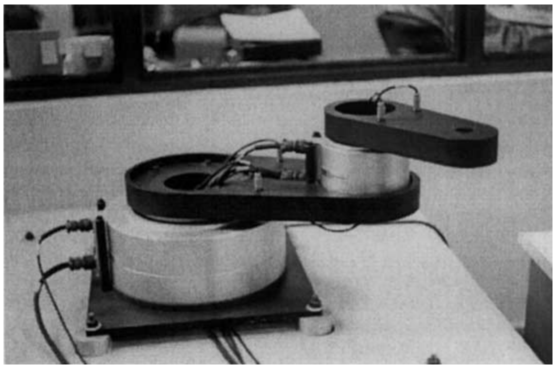

Fig. 2. Physical view of the direct drive robotic manipulator.

$$\Delta y_{i\mathrm{VSS}} = \beta \min(|\Delta y_i|, |A_{yi}|)\,\mathrm{sgn}(N_{yi}) + A_{yi}, \tag{28}$$

$$\Delta y_{i\mathrm{LM}} = N_{yi}, \tag{29}$$

$$\Delta y_i = \frac{\alpha_{1i}\,\Delta y_{i\mathrm{VSS}} + \alpha_{2i}\,\Delta y_{i\mathrm{LM}}}{\alpha_{1i} + \alpha_{2i}}. \tag{30}$$

5. Plant model

In this study, a two degrees of freedom direct drive robotic manipulator, which is illustrated in Fig. 2, is used as the test bed for the proposed training method, depicted in Fig. 3. 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. Furthermore, the ambiguities concerning 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 (31) where $M(\theta)$, $V(\theta,\dot{\theta})$, τ and f stand for the state varying inertia matrix, vector of Coriolis terms, applied torque inputs and friction terms, respectively.

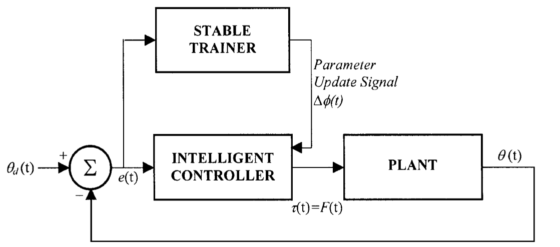

Fig. 3. Control of a plant using the proposed training method.

Table 1
Manipulator parameters

Motor 1 rotor inertia	0.2670	I_1
Arm 1 inertia	0.3340	I_2
Motor 2 rotor inertia	0.0075	I_3
Motor 2 stator inertia	0.0400	I_{3C}
Arm 2 inertia	0.0630	I_4
Motor 1 mass	73.000	M_1
Arm 1 mass	9.7800	M_2
Motor 2 mass	14.000	M_3
Arm 2 mass	4.4500	M_4
Arm 1 length	0.3590	L_1
Arm 2 length	0.2400	L_2
Arm 1 center of gravity	0.1360	L_3
Arm 2 center of gravity	0.1020	L_4
Axis 1 friction	5.3000	f_1
Axis 2 friction	1.1000	f_2
Torque limit 1	245.00	$\tau_{1\,\max}$
Torque limit 2	39.200	$\tau_{2\,\max}$

The plant parameters are given in Table 1 in standard units.

$$M(\theta)\ddot{\theta} + V(\theta,\dot{\theta}) = \tau - f. \tag{31}$$

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 (32) and (33), terms seen in (31) are given explicitly:

$$M(\theta) = \begin{bmatrix} p_1 + 2p_3\cos(\theta_2) & p_2 + p_3\cos(\theta_2) \\ p_2 + p_3\cos(\theta_2) & p_2 \end{bmatrix}, \tag{32}$$

$$V(\theta,\dot{\theta}) = \begin{bmatrix} -\dot{\theta}_2(2\dot{\theta}_1 + \dot{\theta}_2)p_3\sin(\theta_2) \\ \dot{\theta}_1^2 p_3\sin(\theta_2) \end{bmatrix}. \tag{33}$$

In above, $p_1 = 2.0857$, $p_2 = 0.1168$ and $p_3 = 0.1630$. The details of the plant model are presented in [3].

6. Simulation studies

In the simulation studies presented, the plant introduced in Section 5 is controlled by the fuzzy system considered in Section 4. The main objective is to keep the update dynamics in a stable region. This is achieved through a suitable combination of Levenberg–Marquardt optimization technique and the strategy based on the VSS approach.

The reference velocity trajectory, described by (34) and depicted in Fig. 4, is used in all simulations with zero initial errors. With this selection, the each link of the manipulator is enforced to position at 2 rad in the angular space. The maximum value of the reference velocity trajectory has been set such that the Coriolis terms become highly active, which makes the task more difficult:

$$\dot{\theta}_{d1,2} = \pi(1 - \tanh^2(\pi[t-2])) \\ -\pi(1 - \tanh^2(\pi[t-12])). \tag{34}$$

The results presented concern the adjustment of only the defuzzifier parameters 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. 5. The state tracking errors and applied torque inputs are depicted in Figs. 6 and 7, respectively. It is evident from Fig. 6 that once a fluctuation occurs on the error or rate of error, it is dampened out by the use of VSC philosophy in the learning strategy. The torque signals illustrated in Fig. 7 indicate that the algorithm produces physically meaningful values.

The time behavior of the defuzzifier parameters is illustrated in Figs. 8 and 9 for the base and elbow links, respectively. The behavior in the adjustable parameter space clearly indicates the stabilizing activity introduced by the approach presented. If an instant fluctuation is detected on the tracking error, the cost of operation increases and the algorithm gives appropriate changes to the adjustable parameters and tries to maintain the desired performance specifications with less parametric change effort. The settings used to achieve this behavior are tabulated in Table 2.

During the simulations, the squared sum of parametric changes is defined to be the total cost of parametric stability. The cost function is described by

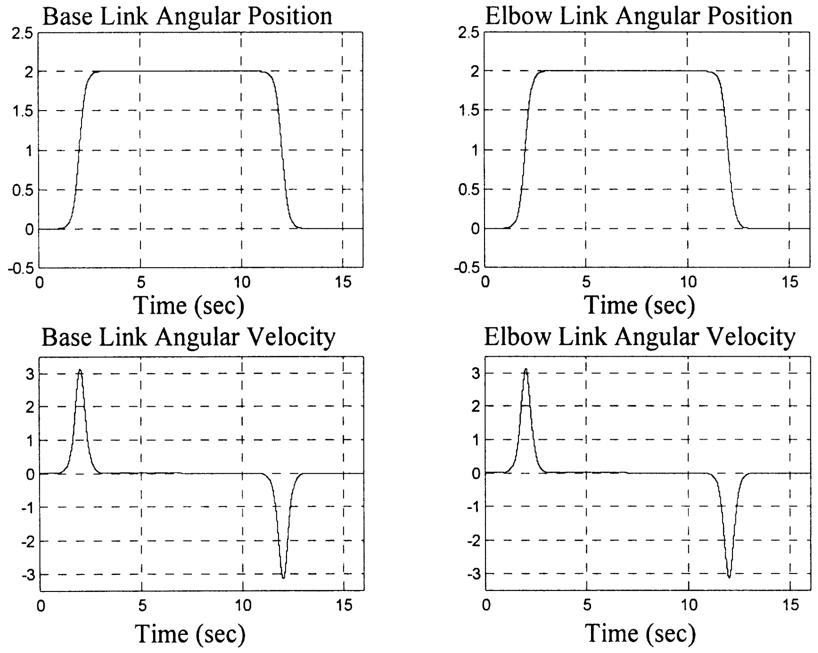

Fig. 4. Reference position and velocity trajectories.

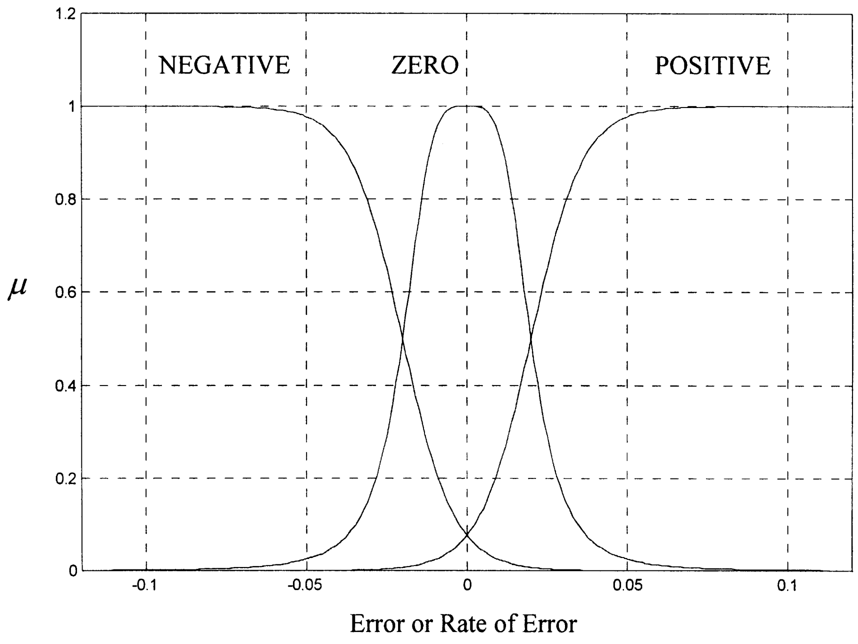

Fig. 5. Definitions of membership functions.

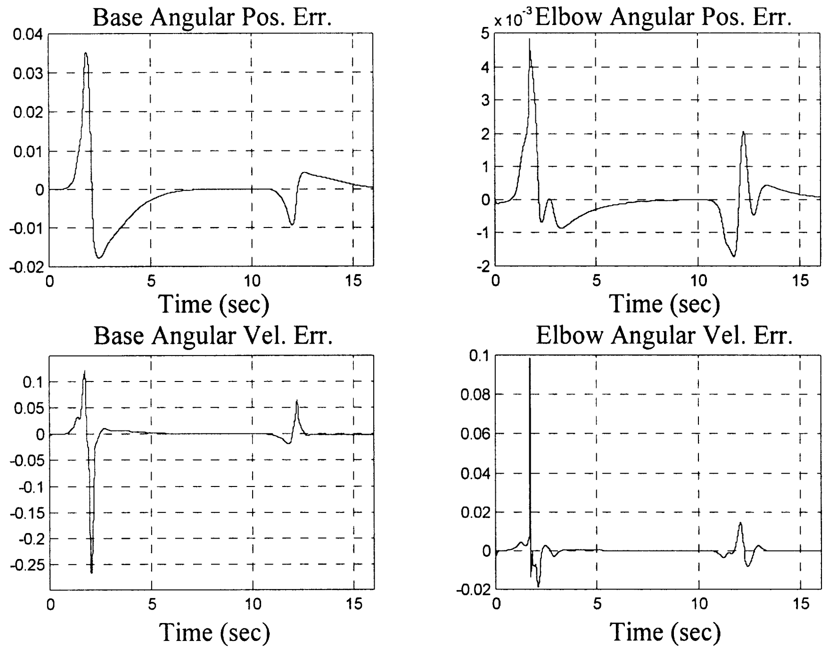

Fig. 6. State tracking errors in fuzzy control.

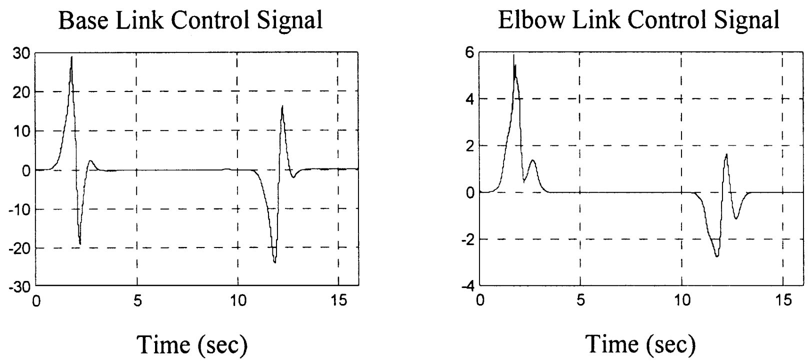

Fig. 7. Applied torque inputs in fuzzy control.

(35) and its time behavior is illustrated in Fig. 10.

$$J(t) = \left[\sum_{j=1}^{\text{Rules}} (\Delta y_j(t))^2\right]_{\substack{\text{ELBOW}\\\text{AXIS}}} + \left[\sum_{j=1}^{\text{Rules}} (\Delta y_j(t))^2\right]_{\substack{\text{BASE}\\\text{AXIS}}}. \tag{35}$$

As can be inferred from Fig. 10, the parametric stabilization performance of the proposed methodology is highly promising especially in the regions where the positional reference values are almost constant. As the reference velocity values change, the fluctuations appear also in the cost graph but they are dampened out in a reasonable time.

A remarkable property of the algorithm presented is the fact that it operates on-line. The potential dis-

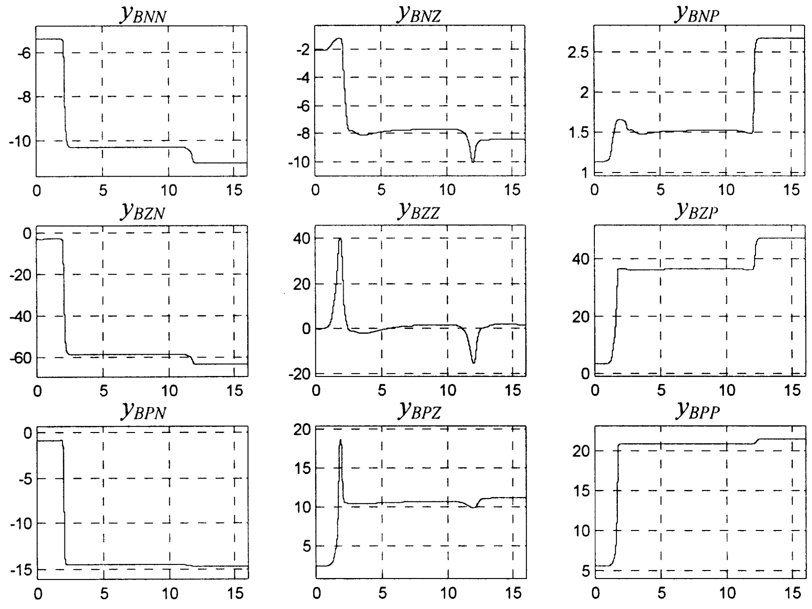

Fig. 8. Defuzzifier parameters (y_B) for base link controller (time versus magnitude).

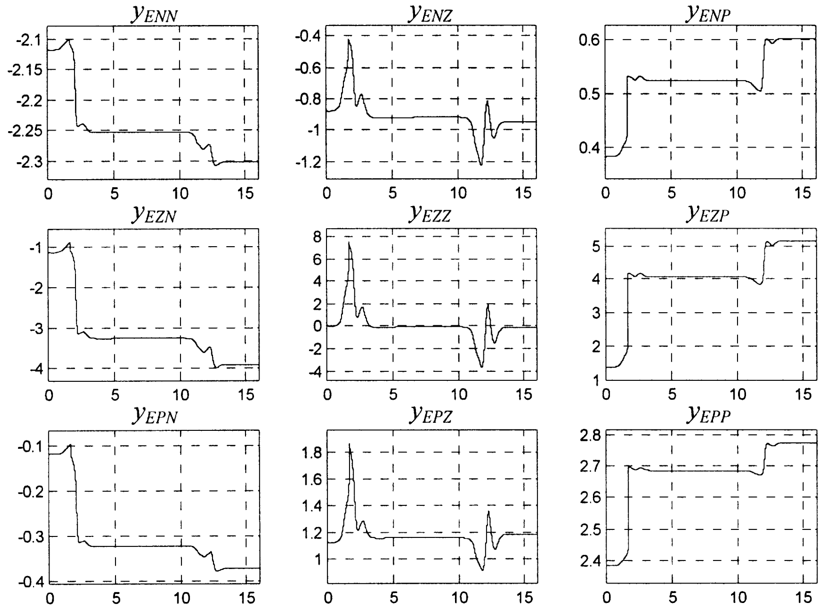

Fig. 9. Defuzzifier parameters (y_E) for elbow link controller (time versus magnitude).

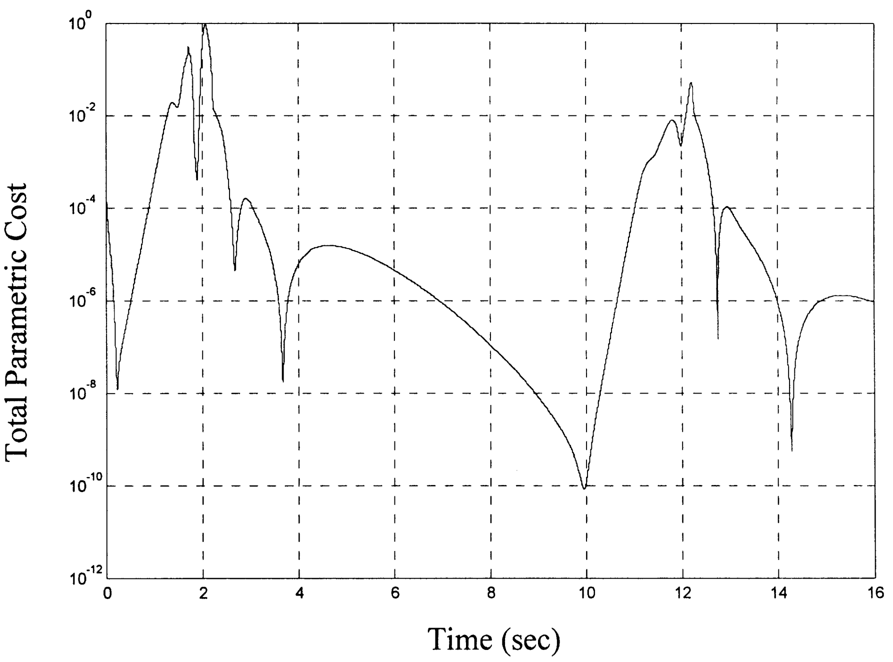

Fig. 10. Time behavior of the stability cost of operation.

Table 2
Settings used in the simulations

T_s	2.5 ms
β	0.1
μ	1.0
α_{1i}	3.0 for all i
α_{2i}	2.0 for all i
Q	0.1
K	0.1
ε	1.0
#Rules	9 (for each link)
#FIS Inputs	2 (for each link)

advantage of on-line training strategies is the lack of rigorous stability considerations. Due to this reason, learning algorithm must be robustified against disturbing effects. In this study, the difficulties that are likely to occur in on-line learning and control are alleviated by the robustness provided by VSS technique.

Finally, for the evaluation of Jacobian matrix, the conventional Levenberg–Marquardt optimization technique requires the outputs of a computationally intelligent system for a set of input patterns. Therefore, the dimensions of the Jacobian increase and the computational cost increases. Since the methodology adopted in this paper utilizes an on-line learning strategy, the response to a single input pattern is sufficient to construct an instant value of Jacobian. This stipulates that the computational burden introduced by the VSS part is compensated by the reduction in Jacobian dimensions. This dimensionality is the fundamental problem in Levenberg–Marquardt optimization procedure due to the necessity of matrix inversion at each step.

7. Conclusions

In this paper, a novel technique for improving learning performance of computationally intelligent architectures is presented. An approximate model of the Levenberg–Marquardt optimization procedure is constructed and VSS approach is incorporated into the proposed form of the parameter update law. In this procedure, Levenberg–Marquardt optimization method is responsible for the minimization of squared error while the VSS based law is responsible for the stability in the parameter change 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, Levenberg–Marquardt optimization procedure 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 parameter space. Since the VSS approach is well known with its robustness property, an appropriate combination of Levenberg–Marquardt optimization technique and VSS methodology can eliminate the handicaps stated in the preceding paragraph. 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, the results confirm the prominent features of the approach, which are discussed in the previous section. The algorithm is applicable to any neuro-fuzzy system model provided that the model output is differentiable with respect to the parameter of interest.

Acknowledgements

This work is supported by Bogazici University Research Fund (Project no: 99A202).

Appendix Notation

F	fuzzy system response
ϕ	a generic parameter of fuzzy system
ϕ^*	cost minimizing value of a generic parameter
$\Delta\phi$	change in parameter ϕ
e	observed output error
d	desired output
J	cost function
J_s	Jacobian
η_ϕ	variable learning rate
T_s	sampling interval of update dynamics
s_ϕ	switching function for parameter ϕ
$\tilde{Q}_\phi$	constant rate component parameter of switching scheme
$\tilde{K}_\phi$	proportional rate component parameter of switching scheme
ε	boundary layer parameter
N_ϕ	change prescribed by LM algorithm
β	scaling factor for parameter stabilizing law
V	Lyapunov function
α_i	weighting factor
μ_{ij}	membership function of ith rule's jth input
c_{ij}	center of membership function μ_{ij}
u_j	jth input of fuzzy inference system
a_{ij}, b_{ij}	shape parameters of membership function μ_{ij}
w	vector of firing strengths
w_n	vector of normalized firing strengths

References

[1] N. Bekiroglu, Adaptive sliding surface design for sliding mode control systems, Ph.D. Thesis, Bogazici University, 1996.

[2] Y. Byungkook, W. Ham, Adaptive fuzzy sliding mode control of nonlinear systems, IEEE Trans. Fuzzy Systems 6 (2) (1998) 315–321.

[3] Direct Drive Manipulator R&D Package User Guide, Integrated Motions Incorporated, 704 Gillman Street, Berkeley, CA 94710, USA.

[4] M.O. Efe, O. Kaynak, Stabilizing and robustifying the learning mechanisms of artificial neural networks in control engineering applications, Int. J. Intell. Systems (2000), to appear.

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

[6] K. Erbatur, O. Kaynak, A. Sabanovic, I. Rudas, Fuzzy adaptive sliding mode control of a direct drive robot, Robotics Autonomous Systems 19 (2) (1996) 215–227.

[7] W. Gao, J.C. Hung, Variable structure control of nonlinear systems: a new approach, IEEE Trans. Ind. Electron. 40 (1) (1993) 45–55.

[8] M.T. Hagan, M.B. Menhaj, Training feedforward networks with the Marquardt algorithm, IEEE Trans. Neural Networks 5 (6) (1994) 989–993.

[9] J.Y. Hung, W. Gao, J.C. Hung, Variable structure control: a survey, IEEE Trans. Ind. Electron. 40 (1) (1993) 2–22.

[10] J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing, PTR Prentice-Hall, Englewood Cliffs, NJ, 1997.

[11] O. Kaynak, F. Harashima, H. Hashimoto, Variable structure systems theory, as applied to sub-time optimal position control with an invariant trajectory, Trans. IEE Japan, Sec. E 104 (3/4) (1984) 47–52.
[12] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Trans. Systems Man, Cybernet. 15 (1) (1985) 116–132.
[13] L.X. Wang, A Course in Fuzzy Systems and Control, PTR Prentice-Hall, Englewood Cliffs, NJ, 1997.
[14] L.X. Wang, Adaptive Fuzzy Systems and Control, Design and Stability Analysis, PTR Prentice-Hall, Englewood Cliffs, NJ, 1994.