Online Learning in Adaptive Neurocontrol Schemes with a Sliding Mode Algorithm

Andon Venelinov Topalov and Okyay Kaynak

***Abstract*—The novel features of an adaptive PID-like neurocontrol scheme for nonlinear plants are presented. The controller tuning is based on an estimate of the command-error on its output by using a neural predictive model. A robust online learning algorithm, based on the direct use of sliding mode control (SMC) theory is applied. The proposed approach allows handling of the plant–model mismatches, uncertainties and parameters changes. The results show that both the plant model and the controller inherit some of the advantages of SMC, such as high speed of learning and robustness.**

***Index Terms*—Adaptive control, intelligent control, learning control systems, neurocontrollers, variable structure systems.**

I. Introduction

In the practice of control engineering, stability and robustness are of critical importance. The applications of artificial neural networks (ANN) in closed-loop feedback control systems have only recently been rigorously studied [1]–[7]. When placed in a feedback system, even a static ANN becomes a dynamical system and takes on new and unexpected behaviors. As a result, properties such as internal stability and robustness of the ANN must be studied before conclusions about the closed-loop performance can be made.

In the theory of control engineering, one way of designing a robust and stable control system is to use the variable structure systems (VSS) approach, which enables the designer to come up with a rigorous stability analysis. It is a well-known fact that a variable structure controller with a switching output will (under certain circumstances) result in a sliding mode on a predefined subspace of the state space. This mode has useful invariance properties in the face of uncertainties in the plant model and therefore is a good candidate for tracking control of uncertain nonlinear systems.

The studies demonstrating the high performance of the variable structure control in handling the uncertainties and imprecision have motivated the use of sliding mode control scheme in training of ANN. The results presented in [8] have shown that the convergence properties of the gradient-based training strategies can be improved by utilizing the SMC scheme. The method presented can be considered as an indirect use of VSS theory. Some studies on the direct use of SMC strategy are also reported in the literature [9], [10]. In [9] the zero level set of the learning error variable in Adaline neural networks is regarded as a sliding surface in the space of learning parameters. A sliding mode trajectory can then be induced, in finite time, on such a desired sliding manifold. Sliding mode invariance conditions determine a least squares characterization of the adaptive weights average dynamics, the stability features of which can be studied using standard time-varying linear systems results. Robustness of the learning algorithm, with respect to bounded external perturbation signals, and measurement noises, is also demonstrated. Yu *et al.* [10] extend further the results of [9] by introducing adaptive uncertainty bound dynamics of the signals. The drawback of the dynamic uncertainty bound adaptation strategy in [10] is the existence of noise on the measured variables. The approach requires the integration of the absolute value of the error signal observed on the outputs. When the error signal is close to zero, it clearly leads to the integration of the absolute value of the noise signal, which gradually increases the bound value and leads to instability in the long run. In another paper [11], the existence of a relation between sliding surface for the plant to be controlled and the zero learning error level of the parameters of a flexible controller is discussed and the control applications of the method considered in [9], [10] are studied with constant uncertainty bounds.

Recently, a new learning algorithm for training multilayer artificial neural networks, based on direct use of SMC strategy, was proposed by G. G. Parma *et al.* [12], [13]. Its online version presented in [12] has been initially tested for the identification of a periodic time signal. In addition to the applicability of the newly proposed algorithm for updating the weights in the hidden layers of multilayer network structures, it also differs from the algorithms in [9] and [10] due to the definition of the sliding surface which is now determined by taking not only the learning error variable, but also its time derivative. The latter contributes very much to the fast convergence capability of the algorithm. This property is crucial for online learning in applications demanding adaptation to constantly changing environmental parameters, such as adaptive real-time control. This is the motivation behind this work. It presents a new architecture of adaptive controller, which combines techniques of PID controllers, artificial neural network controllers, and sliding mode controllers. This combination has a potential to exploit the advantages of these techniques and may be used as a general-purpose robust adaptive controller for nonlinear systems. Some improvements of the algorithm are suggested in order to avoid the possibility of updated weight values tending to infinity when the learning error variable tends to zero and to avoid the well-known chattering problem associated with SMC.

The main body of the paper contains six sections. Section II describes the proposed neuro-adaptive control scheme. Section III presents the plant predictive model. The PID-like controller is presented in Section IV. Section V presents the applied sliding mode learning algorithm. Simulation results are shown in Section VI. Finally, Section VII summarizes the findings of this investigation.

II. The Neuro-Adaptive Control Scheme

One way to implement neural network-based control is by using the well-known principles of one-step-ahead control. In such an approach, a closed-loop predictor monitors the plant inputs and outputs and predicts the output one-sample interval in the future. A one-step-ahead controller monitors the past plant inputs and current and past outputs and manipulates the current control input of the plant so that the predicted output will approach the desired output. One-step-ahead linear control scheme and its adaptive counterpart are well described by Goodwin and Sin [14]. The neural network-based control scheme for nonlinear plants is similar in spirit. A neural-net model of the plant is used for prediction. The controller (usually also a neural network structure) monitors inputs and outputs of the plant and manipulates the control input to make the predicted output, based on the internal model of the system discovered and stored in the neural-net model (predictive model) close to the desired output. Based on these principles a neurocontrol scheme that can be used to control nonlinear plants with model uncertainties and parameter changes can be constructed as depicted on Fig. 1. This structure is general in the sense that it can be easily extended toward other control design approaches. Should the closed-loop system have to follow a reference model, the control design will be in

Manuscript received August 29, 2000; revised January 21, 2001. This paper was recommended by Associate Editor L. O. Hall.

A. V. Topalov is with the Department of Control Systems, Technical University of Sofia, 4000 Plovdiv, Bulgaria (e-mail: topalov@mbox.digsys.bg).

O. Kaynak is with the Electrical and Electronic Engineering Department, Bogazici University, 80815 Istanbul, Turkey (e-mail: kaynak@boun.edu.tr).

Publisher Item Identifier S 1083-4419(01)05660-6.

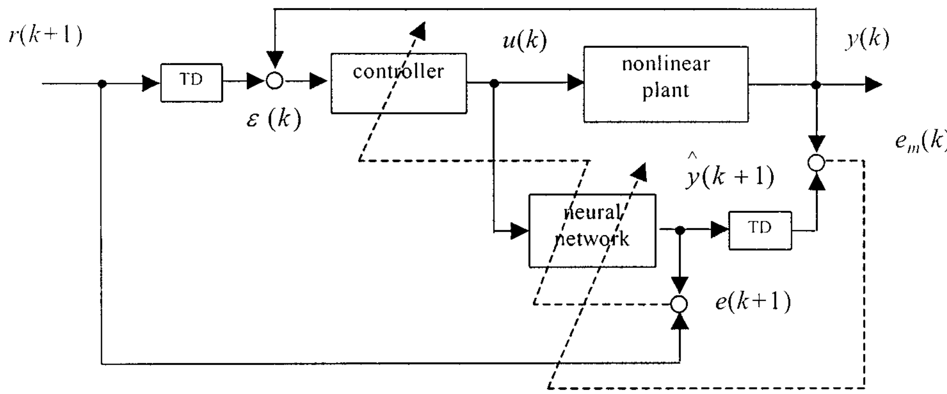


Fig. 1. Neuro-adaptive control scheme.

this case related to a *Model-Reference Adaptive System (MRAS)* approach [15], or should we add to the criterion to be minimized a term penalizing the squared controls then a simple type of optimal control criterion could be achieved and an optimal controller could be trained [16].

It is to be noted that the error used for the adaptation of the controller should be the command-error (i.e., the plant input error) [17], [18]. However, this is normally not available and what is generally done is the use of the difference between the desired response and the actual response (i.e., the plant performance error, or the tracking error). In the approach adopted in this paper, an estimate of the command-error is obtained, named "virtual" command-error, by using a forward dynamics plant predictive model:

$$e_u(k+1) = \frac{\partial \hat{y}(k+1)}{\partial u(k)} e(k+1) \tag{1}$$

where $e(k+1) = r(k+1) - \hat{y}(k+1)$ and $\partial \hat{y}(k+1)/\partial u(k) \cong \partial y(k+1)/\partial u(k)$ is an estimate of the Jacobians of the system, obtained by using a suitably trained forward model of the system, and $e_u(k+1)$ is the "virtual" command-error. This approach is similar in spirit to the one adopted in [17] under the name "distal supervised learning" and in [16]. The novel aspect of the approach presented in this paper is that the one-step-ahead predicted virtual error $e_u(k+1)$ is used instead of the present virtual error that may be noted as $e_u(k)$.

III. Plant Predictive Model

Neural networks can be classified as feedforward networks and recurrent networks. In feedforward networks, the processing elements are connected in such a way that all signals flow in one direction from input units to output units. In recurrent networks there are both feedforward and feedback connections along which signals can propagate in opposite directions.

Feedforward networks have been applied to system identification with success [19], [20]. However, there are also drawbacks, which include a large number of units in the input layer (thus, high susceptibility to external noise and slow computation) and stringent requirements on input signals (which must arrive at exactly the correct rate) [21], [22]. Due to their structure, recurrent networks do not suffer from the above drawbacks. The Elman net [23] is one of the simplest types of recurrent network that can be trained using the standard backpropagation learning algorithm.

Although feedforward networks could be also used in the proposed control architecture, taking into account that modeling with recurrent networks does not require inclusion of a large number of external loops with time delays, a modified Elman recurrent network [24] shown in Fig. 2 is applied to obtain the plant model and to predict its output.

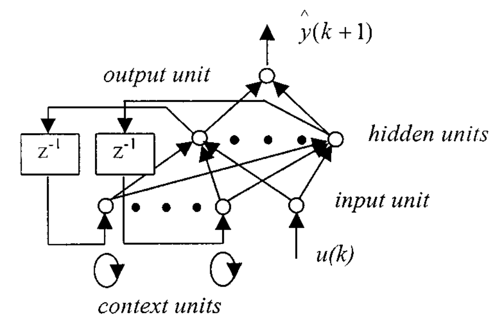


Fig. 2. Structure of the modified Elman network used as a predictor.

The network has a layered structure that consists of one input unit, one output unit, a number of hidden units and the same number of context units. The outputs of the hidden units are sent to the context units, which can be considered as memory units with one-step time delay. The outputs of the context units are also part of the inputs to the hidden units. Only the feedforward connections are modifiable. The strengths of the recurrent connections from hidden units to context units are fixed as 1. Since a dynamic system represents a relation between its present output and its past inputs, the introduction of self-feedback in the context units increases the possibility of the Elman net to model higher-order systems.

At the beginning of training, the outputs of the context units are set to zero (for hyperbolic tangent activation functions) or 0.5 (for sigmoidal activation functions). The output of the jth context unit is given by the following equation:

$$x_{cj}(k+1) = \alpha x_{cj}(k) + x_j(k) \tag{2}$$

where $x_{cj}(k)$ and $x_j(k)$ are the outputs of j-th context unit and j-th hidden unit, respectively and α is the feedback gain of the self-connections. The value of α is between 0 and 1. In this work, the strength α of the added self-connections was adopted to be fixed and to be the same for all self-connections. The results, obtained by Pham *et al.* [24] from the application of modified Elman nets to the identification of dynamic systems are used here to set appropriately the network structure. It is to be noted that the performance of the modified Elman nets for a specific dynamic system depends highly on α. A future extension of the work presented in this paper will be the development of an appropriate training algorithm so that α is automatically selected to suit a given identification problem.

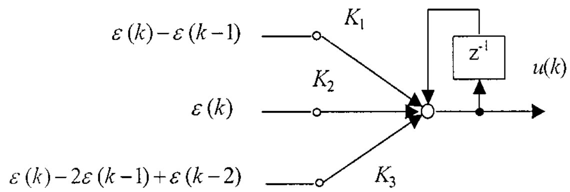


Fig. 3. The PID-like controller.

The output of the neural network can be determined as follows:

$$\hat{y}(k+1) = f\left\{\sum_{i=1}^{q} W2_i f_H\left[\sum_{j=1}^{n} W1_{ij} I_j(k)\right]\right\}. \quad (3)$$

where q is the number of neurons in the hidden layer (four in this work), n is the sum of the number of network inputs (one in this work), and the number of the context units (four in this work), $W1_{ij}$ is the weight of the ith neuron in the first (hidden) layer from its jth input, $W2_i$ is the weight of the output neuron from its ith input (from ith neuron in the hidden layer), $I(k)$ is a set of data consisting of control input signal $u(k)$ fed to the neural network input at time instant k and output signals from the context units at the same time step, $f_{Hi}(\cdot)$ is the activation function for the hidden node i, and $f(\cdot)$ is the linear activation function of the output node.

The estimate of the system's Jacobians that is used to calculate the "virtual" command-error in (1) can be determined by application of the chain rule as in (4)

$$\begin{aligned}\frac{\partial \hat{y}(k+1)}{\partial u(k)} &= \sum_{i=1}^{q} \frac{\partial \hat{y}(k+1)}{\partial f_H(x_i)} \cdot \frac{\partial f_H(x_i)}{\partial x_i} \cdot \frac{\partial x_i}{\partial u(k)} \\ &= \sum_{i=1}^{q} W2_i \left[1 - f_H(x_i)\right] f_H(x_i) W1_{i1} \end{aligned} \quad (4)$$

where $f_H(x_i) = -1/(1+e^{-x_i})$ is a log-sigmoid function and $x_i = \sum_{i=1}^{n} W1_{ij} I_j(k)$.

If a tan-sigmoid function $f_H(x_i) = (1-e^{-x_i})/(1+e^{-x_i})$ is used, the last formula needs to be modified as follows:

$$\frac{\partial \hat{y}(k+1)}{\partial u(k)} = \sum_{i=1}^{q} W2_i \left[1 - f_H^2(x_i)\right]\left(\frac{1}{2}\right) W1_{i1} \quad (5)$$

IV. Controller

An adaptive PID-like controller structure as depicted on Fig. 3 is considered, although it should be noticed that the range of possible controller implementations can be much wider and structures having one or more hidden layers can be used too. The required control $u(k)$ at time instant k is computed by using PID control law as follows:

$$\begin{aligned} u(k) =& u(k-1) + K_1(k)h_1(k) + K_2(k)h_2(k) \\ &+ K_3(k)h_3(k) \end{aligned} \quad (6)$$

where $K_s(k)$ and $s = 1, 2, 3$ are the controller parameters/weights at the time instant k, $\varepsilon(k) = r(k) - y(k)$ is the feedback error, $h_1(k) = \varepsilon(k) - \varepsilon(k-1)$, $h_2(k) = \varepsilon(k)$, $h_3 = \varepsilon(k) - 2\varepsilon(k-1) + \varepsilon(k-2)$ and $y(k)$, and $r(k)$ are the real and the desired system output at time instant k, respectively.

V. Sliding Mode Learning Algorithm

If the training is never turned off, the scheme depicted on Fig. 1 should in principle work as an adaptive controller for time-varying systems as well. Naturally the "forward" neural model providing the Jacobians must be updated online, simultaneously with the update of the controller parameters. When the above-specialized training principle was originally introduced, a recursive gradient method was considered. However, it is well known that the convergence speed of gradient based learning (e.g., dynamic back propagation method) is very slow, especially when the search space is complex. Additionally, the tuning process can easily be trapped into a local minimum. Due to the number of weights typically present in the both networks, it can only be expected to work for systems with very slow variations in the dynamics. Furthermore, a number of numerical robustness issues need to be taken also into account when recursive estimation algorithms are applied over long periods of time [25]. To overcome some of these drawbacks a genetic based method for fast online learning of neural networks in neuro-adaptive systems is proposed in [26]. This approach results in a reduction of the mapping error and an improvement in the tracking performance. However, its major disadvantages are the higher computational complexity and lack of study on stability and robustness issues. The online learning algorithm based on SMC has the advantage that the robustness and stability properties of soft-computing-based control strategies can be analyzed through the use of VSS theory.

The SMC design is divided into two phases. In the first, a sliding surface to produce the input/output behavior is defined. In the second, the weights are updated in order to satisfy the conditions for tracking and sliding on the surface. For the output layer of the model and for the controller output, the sliding surfaces are defined as in (7) and (8) respectively [27]:

$$S_m = \dot{e}_m + \lambda_1 e_m \quad (7)$$

$$S_c = \dot{e}_u + \lambda_2 e_u \quad (8)$$

where $e_m(k) = y(k) - \hat{y}(k)$ is the model error and $\lambda_1, \lambda_2 > 0$ are constants.

For the hidden layer of the neural model of the plant, the following sliding surface is defined as

$$S_H = \dot{e}_H + \lambda_H e_H \quad (9)$$

where $e_H = (1/2)e_m^2$ and $\lambda_H > 0$ [12].

Therefore, considering the fact that the output units of the both learning structures have linear activation functions, the weight update rules for the output layer of the plant model and for the controller can be obtained as in (10) and (11) respectively (for $\alpha_1 > 0$ and $\alpha_2 > 0$):

$$\dot{W}2_i = \alpha_1 \operatorname{sign}(S_m)|e_m| f_H(x_i) \quad (10)$$

$$\dot{K}_s = \alpha_2 \operatorname{sign}(S_c)|e_u| h_s \quad (11)$$

The weight update rule for the hidden layer of the plant model is obtained as in (12) (for $\beta > 0$) [12]:

$$\begin{aligned}\dot{W}1_{ij} &= \frac{\beta \operatorname{sign}(S_H)|e_H|}{(e_m W2_i + \eta)\frac{\partial f_H(x_i)}{\partial x_i}} I_j \\ &= \frac{\beta \operatorname{sign}(S_H)|e_H|}{(e_m W2_i + \eta)\left[1 - f_H(x_i)\right] f_H(x_i)} I_j \end{aligned} \quad (12)$$

where $f_H(x_i)$ is a log-sigmoid activation function of the hidden nodes and η is a small constant introduced to avoid $\dot{W}1_{ij} \to \infty$ when $e_m \to 0$.

If a tan-sigmoid activation function is applied instead, (12) should be transformed as follows:

$$\dot{W}1_{ij} = \frac{\beta \operatorname{sign}(S_H)|e_H|}{(e_m W2_i + \eta)\left[1 - f_H^2(x_i)\right]\left(\frac{1}{2}\right)} I_j \tag{13}$$

A natural solution to avoid the chattering behavior (which is a well known problem associated with SMC) is to smooth the discontinuity in the signum function in (10)–(13) to obtain an arbitrarily close but continuous approximation. One possible approximation is the sigmoid-like function

$$\nu_\delta(S) = \frac{S}{(|S| + \delta)} \tag{14}$$

where δ is a small positive scalar.

The conditions for the analysis of the algorithm from the sliding mode point of view are

i) r and y are limited with limited derivatives;

ii) $f_H(\cdot)$ is differentiable and limited.

For the learning structures described, these conditions are obviously fulfilled.

Using the well-known conditions for sliding mode regime [27], the bounds for the constants λ_1, λ_2 and λ_H that appear in (7), (8) and (9) are obtained as

$$\lambda_1 \geq \max\left\{-\frac{1}{f_H(x_i)} \cdot \frac{\partial f_H(x_i)}{\partial t} - \frac{|\dot{e}_m|}{|e_m|}\right\} \tag{15}$$

$$\lambda_2 \geq \max\left\{-\frac{|\dot{e}_u|}{|e_u|}\right\} \tag{16}$$

$$\lambda_H \geq \max\left\{-\frac{2}{I_j} \cdot \frac{\partial I_j}{\partial t} - \frac{|\dot{e}_H|}{|e_H|}\right\}. \tag{17}$$

The ideal situation would be to have always high values of λ_1, λ_2 and λ_H to guarantee fast convergence, but there is in fact a trade-off between the values of λ_1, λ_2, and λ_H, and the relevant values of α_1, α_2, and β, which by their turn, depend also on the amplitude variations of the system parameters. Therefore, for every system to be controlled, the bounds for α_1, α_2, and β (and λ_1, λ_2, and λ_H) should be derived in advance in order to predict convergence and stability properties.

The upper limits of α_1, α_2, and β can be obtained from the sliding surface expression. For a given S, delimited by the training data and network topology, the upper limits for the gains, or learning rates, α_1, α_2, and β can be easily obtained. According to Utkin [27], the condition for existence of sliding mode and system stability is defined by (18). Since the adaptation of the weights of the learning network structures, which could be interpreted as control signals with regard to these structures, is a function of α_1, α_2, and β, the analysis of (18) results in their limit values:

$$S\frac{dS}{dt} \leq 0. \tag{18}$$

Convergence is guaranteed for any values of α_1, α_2 and β, within the limits established by S, since it implies on the existence of sliding mode. For discrete time, where Sarpturk *et al.* [28] defined the equation $|S(k+1)| < |S(k)|$, instead of (18), as a condition to guarantee the sliding manifold, a way to define the limits for gains α_1, α_2 and β is presented in [13].

With the surfaces defined as in (7)–(9), the weight updating rules defined as in (10)–(13), λ_1, λ_2, and λ_H satisfying the boundary conditions [(15)–(17)], it can be affirmed that the model error e_m and the "virtual" command-error e_u tend to zero [12].

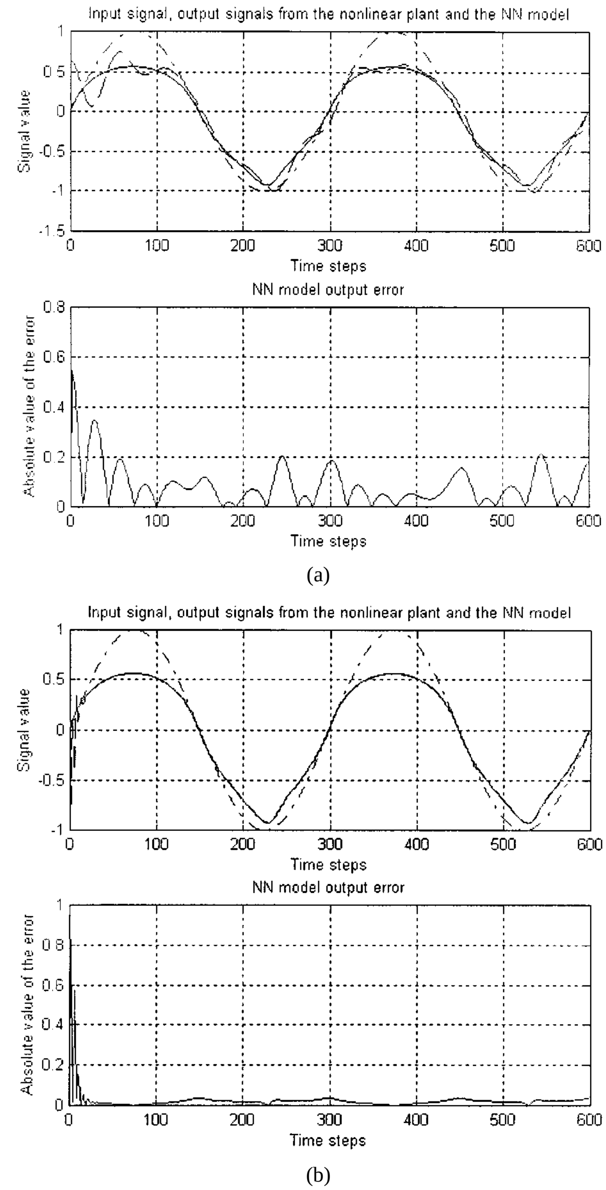


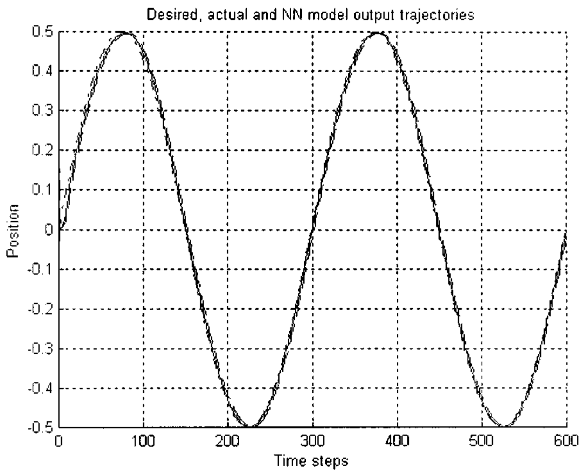
Fig. 4. Results from the performance testing of the recurrent Elman net as a nonlinear plant identifier with (a) dynamic BP learning algorithm and with (b) the proposed sliding mode algorithm. ____ output of the plant, _ _ _ output of the neural model, _ . _ . _ input signal.

VI. Simulation Results

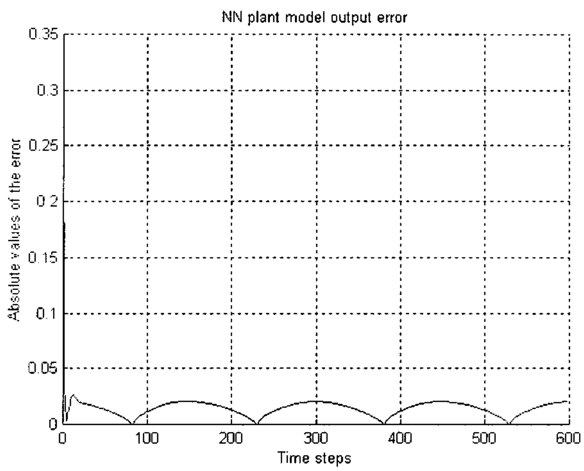
Extensive simulations are conducted to investigate the ability of the scheme depicted on Fig. 1 to work as an adaptive controller when the sliding mode learning algorithm is applied. A nonlinear plant described by the following difference equation:

$$y(k) = \frac{y(k-1)y(k-2)y(k-3)u(k-2)[y(k-3)-1] + u(k-1)}{1 + y(k-2)^2 + y(k-3)^2} \tag{19}$$

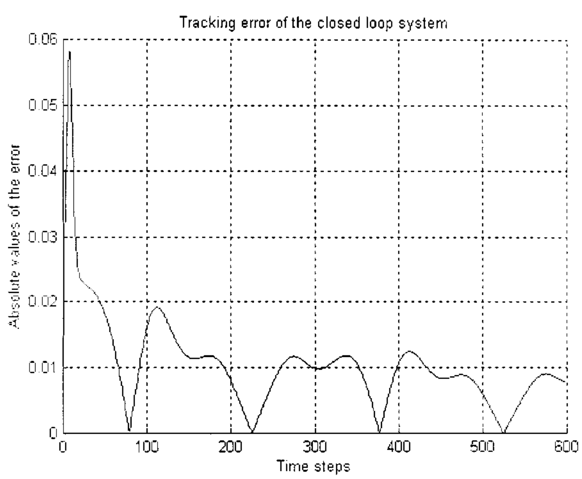
is adopted from [19]. Log-sigmoidal activation functions are adopted for all neurons in the plant predictive model with the exception of the output node that has a linear activation function. Two experiments are carried out. Firstly, the ability of the modified Elman network to model the plant dynamics in an open loop system is investigated. The input to the plant and to the model is taken as a sinusoid $r(k) = \sin(2\pi k/300)$. The initial values of the network weights and biases are chosen randomly. As seen from Fig. 4(a) and (b) when the sliding mode learning



(a)

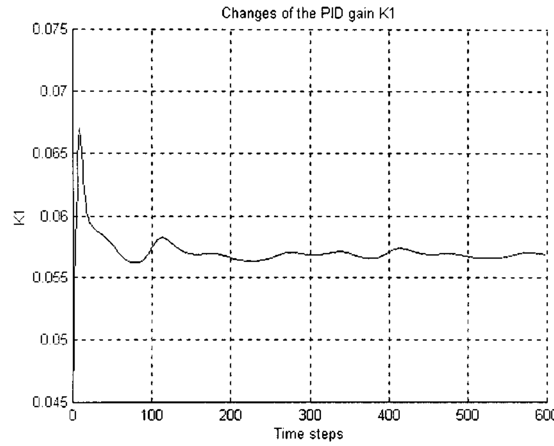


(b)

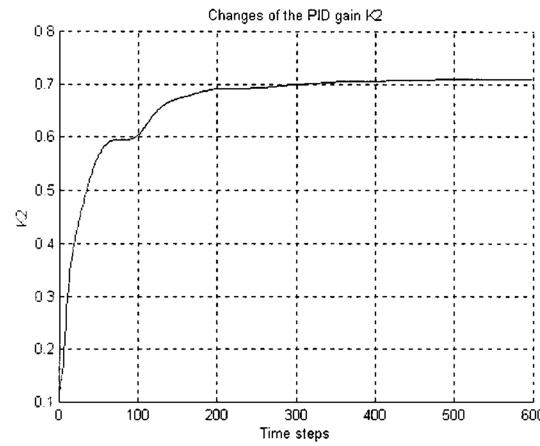


(c)

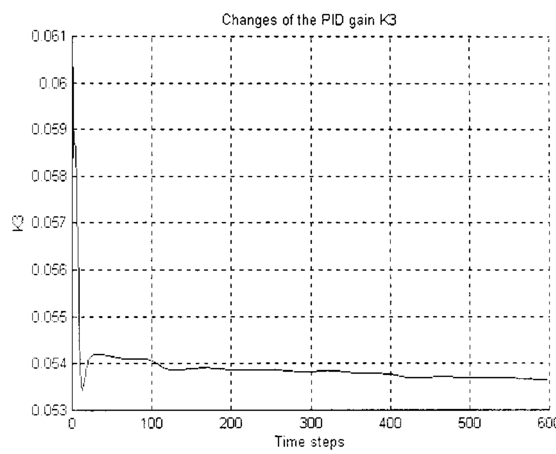
Fig. 5. Results from the performance testing of the neuro-adaptive control scheme with online learning algorithm, based on SMC. ____ output of the system, _ _ _ output of the neural net model, _ . _ . _ reference signal.



(a)



(b)



(c)

Fig. 6. Changes of the parameters of the adaptive controller during the time of simulation.

algorithm is applied, the average error of the model is more than three times smaller in comparison with the dynamic BP learning algorithm. The learning rate of the BP algorithm was set to 0.15. In this experiment, acceptable results for the dynamic BP were obtained only when the input signal was set to $r(k) = \sin(2\pi k/500)$, i.e., a lower frequency.

In the second experiment, the recurrent network has to learn online the forward dynamics' model in the closed loop system. The controller structure is also tuned online. Both, the neural model of the plant and the controller are tuned using the sliding mode learning algorithm. The reference signal is chosen to be $r(k) = 0.5\sin(2\pi k/300)$. Both, the initial gains of the controller and the initial weights and bias terms of the neural model are randomly chosen. As seen from Fig. 5(a)–(c), the system is able to track appropriately the reference signal and appropriate tuning of the controller takes place. Fig. 6(a)–(c) shows the changes of the controller parameters during the time of simulation.

VII. Conclusions

The possibilities for implementation of a sliding mode algorithm for training multilayer NN proposed in [12] as an online mechanism for adaptation in closed-loop feedback neurocontrol systems have been investigated. It is confirmed that the algorithm can be used to train the network structures as they interact with the external environment. The learning rule can be generalized for any number of layers and for recurrent networks also. The network structures trained with the sliding mode learning algorithm are robust and learn fast, both of which are features inherited from SMC. By using the proposed approach we have obtained results that show faster convergence ability and better performance on reducing mapping error in case of online learning neural network structures which can lead to an improvement of the tracking performance of neuro-adaptive systems.

Acknowledgment

The authors would like to acknowledge the fellowship provided to the first author by The Scientific and Technological Research Council of Turkey (TUBITAK) within the framework of the NATO-PC Advanced Fellowship Program.

References

[1] R. M. Sanner and J.-J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Trans. Neural Networks*, vol. 3, pp. 837–864, Nov. 1992.

[2] ——, "Structurally dynamic wavelet networks for adaptive control of robotic systems," *Int. J. Contr.*, vol. 70, no. 3, pp. 405–421, 1998.

[3] T. Parisini, M. Sanguineti, and R. Zoppoli, "Nonlinear stabilization by receding-horizon neural regulators," *Int. J. Contr.*, vol. 70, no. 3, pp. 341–362, 1998.

[4] J. A. K. Suykens, B. L. R. De Moor, and J. Vandewalle, "$\mathrm{NL_q}$ Theory: A neural control framework with global asymptotic stability criteria," *Neural Networks*, vol. 10, no. 4, pp. 615–637, June 1997.

[5] M. M. Polycarpou, "Stable adaptive neural control scheme for nonlinear systems," *IEEE Trans. Automat. Contr.*, vol. 41, pp. 447–451, 1996.

[6] F. C. Chen and H. K. Khalil, "Adaptive control of a class of nonlinear discrete-time systems using neural networks," *IEEE Trans. Automat. Contr.*, vol. 40, pp. 791–801, 1995.

[7] S. Jagannatan and F. L. Lewis, "Multilayer discrete-time neural net controller with guaranteed performance," *IEEE Trans. Neural Networks*, vol. 7, pp. 107–130, 1996.

[8] M. O. Efe and O. Kaynak, "Stabilizing and robustifying the learning mechanisms of artificial neural networks in control engineering applications," *Int. J. Intell. Syst.*, vol. 15, no. 5, pp. 365–388, 2000.

[9] H. Sira-Ramirez and E. Colina-Morles, "A sliding mode strategy for adaptive learning in Adalines," *IEEE Trans. Circuits Syst. I*, vol. 42, pp. 1001–1012, Dec. 1995.

[10] X. Yu, M. Zhihong, and S. M. M. Rahman, "Adaptive sliding mode approach for learning in a feedforward neural network," *Neural Comput. Applicat.*, vol. 7, pp. 289–294, 1998.

[11] M. O. Efe, O. Kaynak, and X. Yu, "Sliding mode control of a three degrees of freedom anthropoid robot by driving the controller parameters to an equivalent regime," *ASME J. Dyn. Syst. Meas. Contr.*, vol. v.122, pp. 783–792, 2000.

[12] G. G. Parma, B. R. Menezes, and A. P. Braga, "Sliding mode algorithm for training multilayer artificial neural networks," *Electron. Lett.*, vol. 34, no. 1, pp. 97–98, Jan. 1998.

[13] ——, "Neural networks learning with sliding mode control: The sliding mode backpropagation algorithm," *Int. J. Neural Syst.*, vol. 9, no. 3, pp. 187–194, 1999.

[14] G. C. Goodwin and K. S. Sin, *Adaptive Filtering Prediction and Control*. New York: Prentice-Hall, 1984.

[15] R. Isermann, K. Lachmann, and D. Matko, "Adaptive Control Systems," in *Systems and Control Engineering*. New York: Prentice-Hall, 1992.

[16] M. Norgaard, O. Ravn, N. K. Poulsen, and L. K. Hansen, *Neural Networks for Modeling and Control of Dynamic Systems*. Berlin, Germany: Springer, 2000.

[17] M. I. Jordan, "Forward models: Supervised learning with a distal teacher," *Cogn. Sci.*, vol. 16, pp. 307–354, 1992.

[18] H. Gomi and M. Kawato, "Neural Network Control for a Closed-Loop System Using Feedback-Error-Learning," *Neural Networks*, vol. 6, pp. 933–946, 1993.

[19] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 4–27, 1992.

[20] S. Z. Qin, H. T. Su, and J. McAvoy, "Comparison of four neural net learning methods for dynamic system identification," *IEEE Trans. Neural Networks*, vol. 3, no. 1, pp. 122–130, 1992.

[21] J. Hertz, A. Krogh, and R. G. Palmer, "Recurrent networks," in *Introduction to the Theory of Neural Computation*. Reading, MA: Addison-Wesley, 1991, ch. 7.

[22] R. J. Williams, "Adaptive state representation and estimation using recurrent connectionist networks," in *Neural Networks for Control*, W. T. Miller, R. S. Sutton, and P. J. Werbos, Eds. Cambridge, MA: MIT Press, 1990, pp. 97–114.

[23] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, pp. 179–211, 1990.

[24] D. T. Pham and X. Liu, "Identification of linear and non-linear dynamic systems using reccurent neural networks," *Artif. Intell. Eng.*, vol. 8, pp. 67–75, 1993.

[25] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed. Reading, MA: Addison-Wesley, 1995.

[26] A. V. Topalov, K.-Ch. Kim, J.-H. Kim, and B.-K. Lee, "Fast genetic on-line learning algorithm for neural network and its application to temperature control," in *Proc. of IEEE Int. Conf. Evolutionary Computation*, Nagoya, Japan, May 1996, pp. 649–654.

[27] V. I. Utkin, *Sliding Modes and Their Application in Variable Structure Systems*. Moscow, Russia: MIR, 1978.

[28] S. Z. Sarpturk, Y. Istefanopulos, and O. Kaynak, "On the stability of discrete time sliding mode control systems," *IEEE Trans. Automat. Contr.*, vol. AC-39, pp. 930–932, Oct. 1987.

Evolutionary Programming-Based Univector Field Navigation Method for Fast Mobile Robots

Yong-Jae Kim, Jong-Hwan Kim, and Dong-Soo Kwon

***Abstract*—Most of navigation techniques with obstacle avoidance do not consider the robot orientation at the target position. These techniques deal with the robot position only and are independent of its orientation and velocity. To solve these problems this paper proposes a novel univector field method for fast mobile robot navigation which introduces a normalized two-dimensional vector field. The method provides fast moving robots with the desired posture at the target position and obstacle avoidance. To obtain the sub-optimal vector field, a function approximator is used and trained by evolutionary programming. Two kinds of vector fields are trained, one for the final posture acquisition and the other for obstacle avoidance. Computer simulations and real experiments are carried out for a fast moving mobile robot to demonstrate the effectiveness of the proposed scheme.**

***Index Terms*—Evolutionary programming, navigation, soccer robots, univector field navigation method, wheeled mobile robots.**

I. Introduction

Navigation with obstacle avoidance is one of the key issues to be looked into for successful applications of autonomous mobile robots. Navigation involves three tasks: mapping and modeling the environ-

Manuscript received April 17, 1999; revised February 3, 2001. This paper was recommended by Associate Editor R. A. Hess.

The authors are with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), Taejon, Korea (e-mail: yjkim@vivaldi.kaist.ac.kr; johkim@vivaldi.kaist.ac.kr).

Publisher Item Identifier S 1083-4419(01)05219-0.