Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Neural Sliding-Mode Control of Engine Torque ⋆

Ting Huang * **Derong Liu** * **Hossein Javaherian** ** **Ning Jin** *

* *Department of Electrical and Computer Engineering, University of Illinois at Chicago, IL 60607, USA (e-mails: thuang20@uic.edu, dliu@ece.uic.edu, njin@uic.edu)*
** *Electrical & Controls Integration Lab, General Motors R&D and Planning, Warren, MI 48090, USA (e-mail: hossein.javaherian@gm.com)*

Abstract: In this paper, we investigate the applications of neural sliding-mode control method to automotive engine control. The scheme of neural sliding-mode control is realized by two parallel neural networks. The first neural network estimates the equivalent control term and the other one generates the corrective control term. The goal of the present learning control design of automotive engines is to track the commanded torque under various operating conditions. Using the data from a test vehicle with a V8 engine, we have developed a neural network engine model and neural network controllers based on the idea of sliding-mode control to achieve optimal torque control. In simulation studies of the neural sliding-mode design method, very good transient performance and fast speed of convergence have been observed. In this process, the tedious task of parameter tuning by trial-and-error has been eliminated. Distinct features of the present technique are the controller's real-time adaptation capability based on observed real vehicle data and its rapid convergence which allow the neural network controller to be further refined and improved in real-time vehicle operation through continuous learning and adaptation.

Keywords: Sliding-Mode Control; Neural Control; Torque Control; Engine Control; Neural Networks.

1. INTRODUCTION AND BACKGROUND

Sliding-mode control (SLMC) has been widely used due to its robustness to system parameter uncertainties and external disturbances. The theory has been developed mainly for continuous-time systems in which the sliding mode is generated by discontinuous controls on certain sliding surfaces; see Utkin (1992). Meanwhile, researchers have been developing discrete sliding mode control (DSMC) for more than two decades and there are many successful industrial applications including Koshkouei et al. (2000), Lee et al. (1999), Li et al. (2000), Matas et al. (2002). Essentially, SLMC utilizes a high-speed switching control law to drive state trajectory of nonlinear plant onto a specified, user-chosen surface in the state space (called the sliding or switching surface), and to maintain the plant state trajectory on this surface for all subsequent times. The plant dynamics restricted to this surface represent the controlled systems behavior. By proper design of the sliding surface, SLMC attains the conventional goals of control such as stabilization, tracking and regulation.

Automotive engines are known to be complex nonlinear dynamical systems. The control problems of automotive engines have been investigated by many researchers (Alippi et al. (2003), Kovalenko et al. (2004), Moskwa et al. (1990), Park et al. (2003), Won et al. (1998) and references cited therein). The present study considers the neural sliding-mode control (NSLMC) for automotive engine control. SLMC has been mainly applied to motion control and robotics. There are also some applications to engine control reported in the literature, such as Khan et al. (2003), Yang et al. (1997), Lu et al. (2000), Ouenou-Gamo et al. (1997).

Although SLMC has advantages over an adaptive approach regarding its good adaptation to unmodeled dynamics and disturbances, and guaranteed transient performance, two drawbacks typically limit the application of this technique. One is the discontinuity in the control law when the system crosses the sliding surface and the other is the lack of a learning capability. The chattering caused by high frequency switching control activity may excite unmodeled high-frequency dynamics leading to the degradation of system performance and potential instability. It is difficult to learn complex nonlinearities such as friction, using a conventional linearly parmetrized adaptive framework. Neural networks which represent a class of parametrizations with attractive properties including learning would solve these two problems. The present work will use two parallel neural networks to realize the equivalent control and the corrective control of the SLMC design. The calculation of equivalent control is realized by adaptively learning without knowing the plant dynamics. The proposed adaption scheme directly results in a chatter-free control action for the corrective control.

⋆ This work was supported in part by the General Motors Corporation and by the National Science Foundation under Grant EECS-0355364.

Incorporation of some degree of "computational intelligence" into SLMC can be made by the use of neural networks. The purpose of integration of the computational intelligence methodologies in SLMC is to deal with uncertainties of the controlled plant, the problem of chattering and the calculation of equivalent control which are quite difficult to solve by the conventional SLMC. The basic idea proposed in Du et al. (1997) and Won et al. (1998) improves the control performance by the use of neural network to approximate the plant nonlinearities or uncertainties. Large uncertainty results in large sliding control parameters which in turn results in inferior performance. The only way to reduce control parameter values is to decrease the system uncertainty. A Gaussian network parameterization is used to capture part of the uncertain system dynamics and thus to decrease the system uncertainty. In Karakasoglu et al. (1995), a neural network was used for the adaptation of the SLMC parameters where the SLMC parameters, such as the slope of the sliding surface and the controller gain, are progressively updated. Both in Ertugrul et al. (2000) and Tsai et al. (2004), two parallel neural networks are used to realize the equivalent control and corrective control of SLMC design. The difference between these two schemes is that, in Ertugrul et al. (2000), the error for updating the neural network for equivalent control is the output of the corrective control while in Tsai et al. (2004), it is the sliding function S. However, the speed of either of the algorithms is slower than the one proposed in this paper.

This paper is organized as follows. In Section 2, a brief introduction of SLMC will be discussed. In Section 3, neural sliding-mode controller will be developed. In Section 4, simulation studies of engine torque control using neural sliding-mode method will be presented. In the final section, Section 5, conclusions will be drawn.

2. SLIDING-MODE CONTROL

The most salient feature of an SLMC is that the feedback control is discontinuous on one or more manifolds in the state space. When the state crosses each discontinuity surface, the structure of the feedback system is altered. Under certain circumstances, all motions in the neighborhood of the manifold are directed toward the manifold and, thus, a sliding motion on a predefined subspace of the state space is established in which the system state repeatedly crosses the switching surface. 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 (Kaynak et al. (2001)). In general, the phase trajectory of the system with SLMC consists of two parts. The first part is the reaching mode in which the trajectory starting from anywhere on the phase plane under certain circumstances moves toward a switching surface and reaches the surface in finite time. The second part is the sliding mode in which the trajectory slides along the surface to the origin of the phase plane.

Consider the following nonlinear, multi-input multi-output system:

$$\dot{X}(t) = F(X, U) \tag{1}$$

where the state space has a dimension of $\mathrm{Dim}(X) = n$ and the control space has $\mathrm{Dim}(U) = m$. The error of the system is defined as

$$e = X_d(t) - X(t), \tag{2}$$

where $X_d(t)$ represent the desired targets and $X(t)$ represents the actual values. Both are column vector with dimension n.

The sliding mode control design approach consists of two steps. The first step is to select a sliding surface that models the desired closed-loop performance in state variable space according to design specifications. The second is concerned with the selection of a control law which will drive the system state trajectories toward the sliding surface and stay on it. Sliding surface is defined as:

$$S(e) = c^T e + d^T \dot{e} \tag{3}$$

where $c = [c_1, c_2, \cdots, c_n]^T$, $d = [d_1, d_2, \cdots, d_n]^T$, and $e = [e_1, e_2, \cdots, e_n]^T$ which is defined by (2).

The aim of SLMC is to drive the system states to the sliding surface and remain on it. From *Lyapunov theorem for global stability*, the control input for the system is:

$$U(t) = U_{eq}(t) + U_c(t) \tag{4}$$

where U_{eq} represents equivalent control which is the control action necessary to maintain an ideal sliding motion on sliding surface and U_c represents corrective control which drives the phase trajectory towards the sliding surface. It is given by $U_c = K\mathrm{sign}(S)$, where K is a matrix (Ertugrul et al. (2000)). The corrective control given by $K\mathrm{sign}(S)$ exhibits high frequency oscillations in its output which is known as chattering. Chattering is an undesirable phenomena since it excites unmodeled high-frequency plant dynamics and this can result in unforeseen instabilities. To eliminate it, in general, a saturation function or a sigmoid is used instead of the sign function. In this case, the corrective control is computed as:

$$U_c(t) = KG(S). \tag{5}$$

where $G(S)$ is a saturation or a shifted sigmoid function, which can be chosen as the following function:

$$G(S) = \frac{1 - e^{-S}}{1 + e^{-S}}. \tag{6}$$

3. NEURAL SLIDING-MODE CONTROL

3.1 Structure of NSLMC

Two neural networks in parallel are used to realize the equivalent control and corrective control of SLMC design as in Figure 1 which shows where neural network #1 is used to estimate the equivalent control, and neural network #2 is employed to generate the corrective control to estimate the chattering effect. The sum of U_{eq} and U_c form the control signal to be applied to the controlled plant.

When the state of system reaches the sliding mode, the equivalent control term takes control of the system and the corrective control term goes to zero. The difference between the equivalent control and the estimate of the

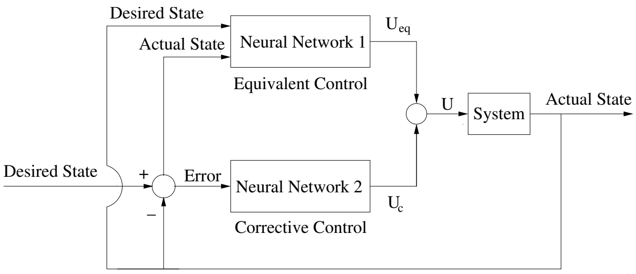

Fig. 1. The neural sliding mode control structure

equivalent control is reflected as corrective control which take effect only when the states of the system deviate from the sliding surface. When sliding takes place, the equivalent control has the same role as the inverse dynamics of the controlled system (Ertugrul et al. (2000)).

3.2 Neural Computation of the Equivalent Control

The structure will be chosen as a two-layer feedforward neural network with one hidden layer and one output layer. The inputs to the neural network are the desired target and actual value of the output. The output of the neural network is the equivalent control U_{eq}. The weight adaption of the neural network is based on a minimization of the cost function as follows:

$$E = \frac{1}{2}(U_{eq} - \hat{U}_{eq})^2 = \frac{1}{2}\zeta^2 \tag{7}$$

where $\zeta = U_{eq} - \hat{U}_{eq}$.

The Levenberg–Marquardt (L–M) algorithm is used to update the weights of neural network 1 instead of the backpropagation algorithm which is a steepest descent algorithm. The selection of L–M algorithm is based on the fact that the L–M algorithm is widely accepted as the most efficient one in the sense of realization accuracy for nonlinear least squares (Hagan et al. (1994)).

The formula for updating weights is given as follows:

$$\triangle W = [J^T(W)J(W) + \mu I]^{-1} J^T(W)\zeta \tag{8}$$

where the parameter μ is adjustable and $J(W)$ is the Jacobian matrix. μ is multiplied by some factor β whenever a step would result in an increased E. When a step reduces E, μ is divided by β. By adjusting μ in this way, the search direction interpolates between the gradient and the Gaussian-Newton direction. That is the reason why the rate of convergence is satisfactory. Jacobian matrix $J(W)$ can be expressed as follows:

$$J(W) = \left[\frac{\partial\zeta}{\partial W_1} \;\; \frac{\partial\zeta}{\partial W_2} \;\; \cdots \;\; \frac{\partial\zeta}{\partial W_n}\right]^T. \tag{9}$$

From equation (7),

$$\frac{\partial\zeta}{\partial W_i} = \frac{\partial(U_{eq} - \hat{U}_{eq})}{\partial W_i} = -\frac{\partial\hat{U}_{eq}}{\partial W_i}. \tag{10}$$

Equation (10) can be calculated using the standard backpropagation algorithm. Thus, the Jacobian matrix can be computed by (9).

From (7), we find that the desired equivalent control is unknown. To overcome this problem, $U_{eq} - \hat{U}_{eq}$ was replaced

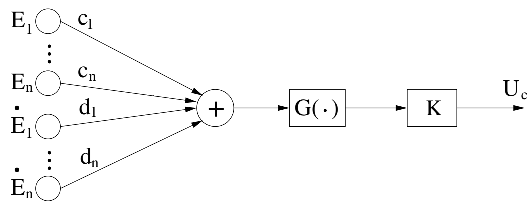

Fig. 2. The structure of NN2

by the value of sliding function S since the characteristics of $U_{eq} - \hat{U}_{eq}$ and S are similar (Tsai et al. (2004)), that is, when S is close to 0, $U_{eq} - \hat{U}_{eq} \to 0$.

3.3 Neural Computation of the Corrective Control

One of the problems in the application of neural networks is how to choose the number of layers, the number of neurons in each layer and the connections among neurons. This is not a problem here since the structure of the neural network #2, which is shown in Figure 2, is decided by the design of SLMC. From (3) and (5), the gains of SLMC are represented as the weights of neural network #2. In this way, the gains of SLMC are adapted gradually to the best values.

An adaption scheme to minimize the sliding function is proposed using gradient descent method. The cost function is defined as

$$J = \frac{1}{2}SS^T. \tag{11}$$

Since S is the error in (3), minimization of S results in minimization of the error. To minimize J, the weights are changed in the direction of the negative gradient,

$$\triangle K = -\mu\frac{\partial J}{\partial K}, \quad \triangle c_i = -\mu\frac{\partial J}{\partial c_i}, \quad \triangle d_i = -\mu\frac{\partial J}{\partial d_i},$$

where μ is the learning rate, K is defined in (5), and c_i and d_i are both defined in (3). A learning rate is selected by user in order to determine how much the link weights and node biases can be modified based on the change direction and change rate. An adaptive learning rate is a better choice which attempts to keep the learning step size as large as possible while keeping learning stable.

The gradient descent for c_i can be derived using (3) as:

$$\triangle c_i = -\mu\frac{\partial J}{\partial c_i} = -\mu\frac{\partial J}{\partial S}\frac{\partial S}{\partial c_i} = -\mu S\frac{\partial S}{\partial c_i} = -\mu \cdot S \cdot e_i. \tag{12}$$

The gradient descent for d_i can be derived using (3) as:

$$\triangle d_i = -\mu\frac{\partial J}{\partial d_i} = -\mu\frac{\partial J}{\partial S}\frac{\partial S}{\partial d_i} = -\mu S\frac{\partial S}{\partial d_i} = -\mu \cdot S \cdot \dot{e}_i. \tag{13}$$

The gradient descent for K can be derived as:

$$\triangle K = -\mu\frac{\partial J}{\partial K} = -\mu\frac{\partial J}{\partial S}\frac{\partial S}{\partial K} = -\mu S\frac{\partial S}{\partial K}. \tag{14}$$

From (3),

$$S(e) = c^T e + d^T\dot{e} = c^T(X_d - X) + d^T\dot{e}. \tag{15}$$

That is, from (4) and (5)

$$\frac{\partial S}{\partial K} = -\frac{\partial X}{\partial K} = -\frac{\partial X}{\partial U}\frac{\partial U}{\partial U_c}\frac{\partial U_c}{\partial K} = -\frac{\partial X}{\partial U}G(S), \tag{16}$$

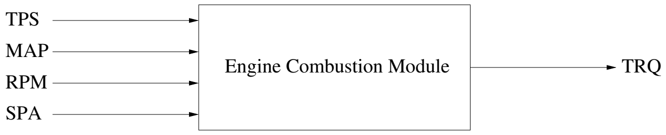

Fig. 3. The model structure of the test engine

where X is the output of the system and $G(S)$ is defined in (6). Since the engine system will be represented by a neural network, according to the backpropagation algorithm, $\partial X/\partial U$ can be derived easily. Finally,

$$\triangle K = -\mu S \frac{\partial X}{\partial U} G(S). \tag{17}$$

4. NSLMC SIMULATIONS OF ENGINE TORQUE CONTROL

4.1 Engine Description

A test vehicle with a V8 engine and 4-speed automatic transmission is instrumented with engine and transmission torque sensors, wide-range air-fuel ratio sensors in the exhaust pipe located before and after the catalyst on each bank, as well as exhaust gas pressure and temperature sensors. The vehicle is equipped with a dSpace rapid prototyping controller for data collection and controller implementation. Data is collected at each engine event under various driving conditions, such as Federal Test Procedure (FTP cycles), as well as more aggressive driving patterns, for a length of about 95,000 samples during each test. The engine is run under closed-loop fuel control using switching-type oxygen sensors. The dSpace is interfaced with the powertrain control module (PCM) in a by-pass mode.

4.2 Control Objectives

The objective of the present engine controller simulations is to provide control signals so that the torque generated by the engine will track the desired (or demanded) torque. The measured torque values are obtained using a commercial engine controller under warmup conditions. Based on the data collected we use the neural sliding-mode controller to generate control signal TPS (throttle position) with the goal of producing exactly the same torque response as in the data set. That is to say, for the simulation purposes, the demanded torque is given by the torque level in the vehicle data set and we build a controller that provides control signal which achieve the required torque performance. The performance is measured by a norm of deviations between the demanded and measured torque levels.

4.3 Engine Combustion Model

We consider a model of the test engine shown as in Figure 3 where TRQ (engine torque) is the output. The model structure chosen here is compatible with the mathematical engine model developed by Dobner (1980), Dobner (1983) and others.

The engine model is constructed by a two-layer feedforward neural network with four inputs: TPS (throttle position), MAP (manifold absolute pressure), RPM (engine speed) and SPA (spark advance), where TPS is the control signal and the other three inputs are reference signals compatible with the control signal at any operating conditions.

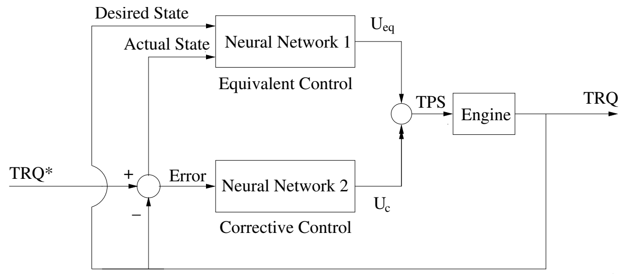

Fig. 4. The structure of the torque control system

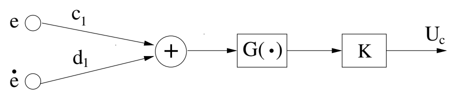

Fig. 5. The structure of neural network 2

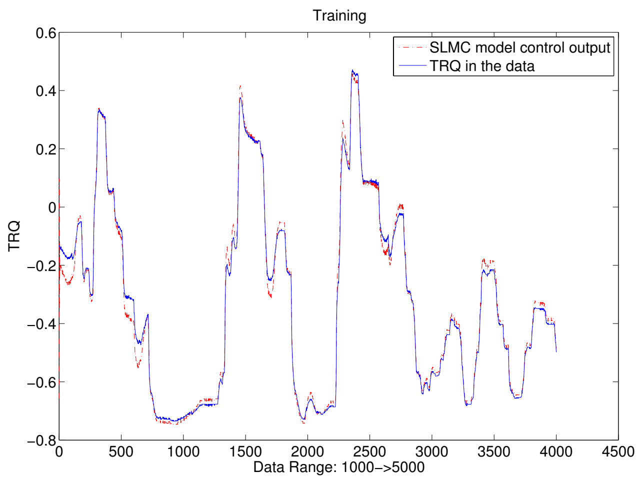

Fig. 6. NSLMC training control effect

4.4 Control Algorithm

The structure of the torque control system is shown in Figure 4. The inputs to the NN1 are desired TRQ* which is in the data set and actual TRQ which tries to track the desired TRQ*. The sum of two outputs of the NN1 and NN2 is TPS which is the control signal for the engine.

The sliding surface S is defined as:

$$S = c_1 e + d_1 \dot{e} \tag{18}$$

where e represents error between target TRQ* and actual TRQ which is TRQ* − TRQ. From the defined S, the structure of NN2 can be decided and it is shown in Figure 5. The inputs to the NN2 are the error between the target TRQ and the actual TRQ and the derivative of the error. The other part of control algorithm and the weight adaption rules were described in Section 3.

4.5 Simulation Results

The training data range for the evaluation of the control algorithm for TRQ control is arbitrarily selected from 1000

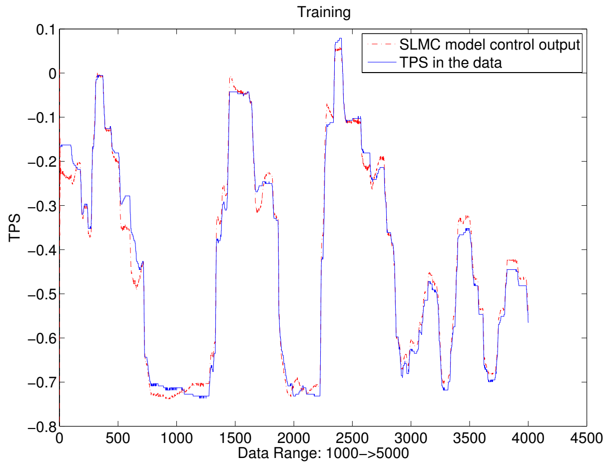

Fig. 7. The training output of equivalent control and TPS in the data

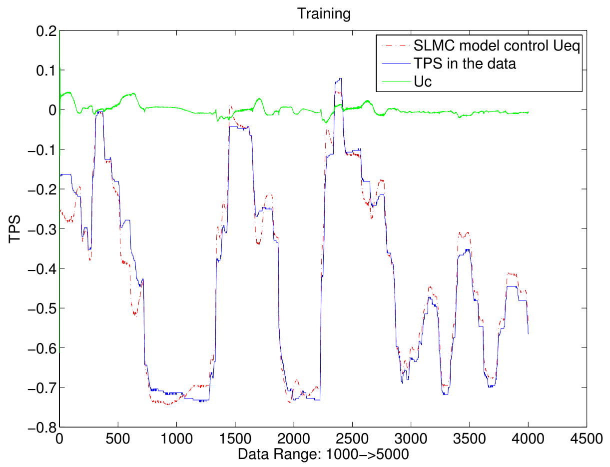

Fig. 8. The training output of equivalent control, corrective control and TPS in the data

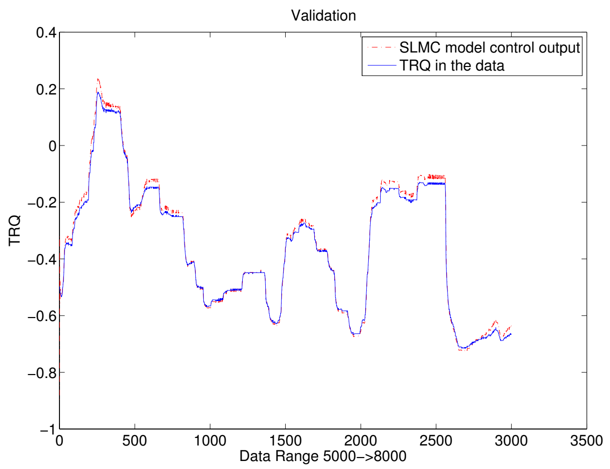

Fig. 9. NSLMC validation control effect

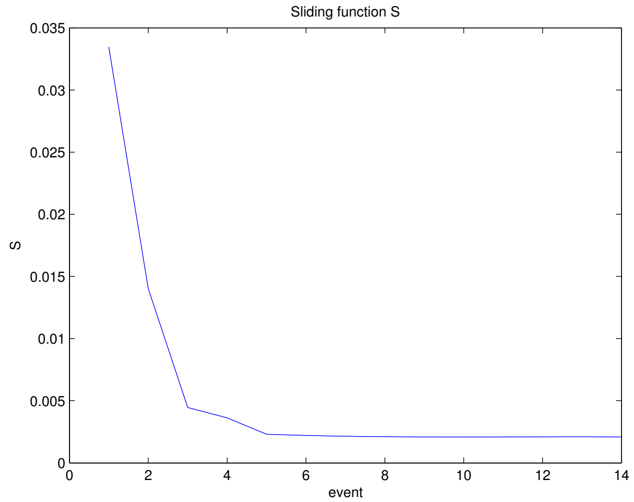

Fig. 10. Trajectory of sliding function S

to 5000 events in the data set for Federal Test Procedure (FTP) test, while the valudation data is from 5000 to 8000 events. All values of signals are normalized to a range between -1 and 1 for convenience in the neural network training. Figure 6 shows the output (TRQ) of the engine using neural sliding-mode control as compared to the demanded TRQ in the data set. From Figure 6, we can see that the TRQ controlled by the NSLMC controller tracks the TRQ demand very well. Figure 7 shows the output of the control input to the engine compared to the TPS in the data. From Figure 7, we can also see that the control output of the NSLMC controller is very close to the measured TPS in the data set. Figure 8 shows the output of the equivalent control and corrective control compared to the TPS in the data. From this figure, we can see that most of the time, the corrective control is around zero. Only when the system states deviate from the sliding mode, the controller takes action to pull the system states back to the sliding surface. Figure 9 shows the validation effect of the control method. From Figure 9, we can see that the control effect is very good even for the data that never trained before.

The initial values for c_1, d_1 and K are 1 in the experiments. In the training process, d_1 keeps unchanged. They can be selected randomly, that is, they do not have to be selected with big initial values. The trajectory of sliding function S during training is as in Figure 10. From the figure, we can see that only after 6 events, the sliding function S reaches the acceptable value (under 0.005 is acceptable).

4.6 Discussions of the Achieved Results

Compared to the method in Tsai et al. (2004), the training speed is very quick and there's no oscillation during the training. Actually, it just took only 15 steps (about 1 minute) to get very good results (The computer used is a 3.2GHz Pentium 4 with 1G RAM). While using backpropagation (delta training rule), it took around 20 steps to make MSE to be below 0.003. But at that time, there are some big oscillations in the output where the target data has the oscillation in a small range. It took 400 steps to smoothen out these big oscillations which requires one additional hour running on the computer. Because of the generalization of the neural networks, the neural sliding mode controller still works very well for the data in the range of 5000–9000 which is not trained during the simulation. Considering the speed and control and learning simultaneously, neural sliding-mode control method may be a good candidate for online learning control.

The parameters of neural network for the corrective control are set by trial and error in Ertugrul et al. (2000) and Tsai et al. (2004), which is a time-consuming task and are selected big to achieve fast convergence on purpose. In our scheme, the parameters are selected as 1 which is enough to guarantee performance. The way of computation of K here is different from the method in Ertugrul et al. (2000) and Tsai et al. (2004) where the computation of the gradient of K is acquired from the integral of $G(S)$ which would be unpredictably big in real experiments.

5. CONCLUSIONS

Our research results show that the method of neural sliding mode control is a good approach for engine torque control. The scheme of neural sliding mode control is realized by two parallel neural networks. The first neural network estimates the equivalent control term and the other one generates the corrective control term.

The following advantages are observed in the numerical experiments:

- The method does not require any prior knowledge about the engine under control and its characteristics.
- Chattering is eliminated without any performance degradation.
- The speed of convergence is surprisingly fast using the Levenberg–Marquardt algorithm.
- The structure of neural network for the corrective control is well defined by the SLMC design.

Research work on the application of this technique to engine air-fuel ratio control is underway and will be reported in a future paper.

REFERENCES

C. Alippi, C. de Russis, and V. Piuri, A neural-network based control solution to air-fuel ratio control for automotive fuel-injection systems. *IEEE Trans. Systems, Man and Cybernetics, Part C,* vol. 33, no. 2, pp. 259–268, May 2003.

D. J. Dobner, Dynamic engine models for control development Part I: Non-linear and linear model formation. *Int. J. of Vehicle Design*, Special Publication SP4, pp. 54–74, 1983.

D. J. Dobner, A mathematical engine model for development of dynamic engine control. *SAE Paper 800054*, 1980.

H. Du and S. S. Nair, A neuro-sliding control approach for a class of nonlinear systems. *Proc. First International Conference on Knowledge-Based Intelligent Electronic Systems*, Adelaide, Australia, May 1997, vol. 2, pp. 331–337.

M. Ertugrul and O. Kaynak, Neuro sliding mode control of robotic manipulators. *Mechatron*, vol. 10, no. 12, pp. 243–267, 2000.

M. T. Hagan and M. B. Menhaj, Training feedforward networks with the Marquardt algorithm. *IEEE Trans. Neural Networks*, vol. 5, no. 6, pp. 989–993, Nov. 1994.

A. Karakasoglu and M. K. Sundareshan, A recurrent neural networkbased adaptive variable structure model-following control of robotic manipulators. *Automatica*, vol. 31, no. 10, pp. 1495–1507, 1995.

O. Kaynak, K. Erbatur, and M. Ertugnrl, The fusion of computationally intelligent methodologies and sliding-mode control-a survey. *IEEE Trans. Industrial Electronics*, vol. 48, no. 1, pp. 4–17, Feb. 2001.

M. K. Khan and S. K. Spurgeon, MIMO control of an IC engine using dynamic sliding modes. *Proc. Intelligent Systems and Control*, Salzburg, Austria, June 2003.

A. J. Koshkouei and Zinober, Sliding mode control of discrete-time systems. *J. of Dynamic System, Measurement and Control*, vol. 122, pp. 793–802, Dec. 2000.

O. Kovalenko, D. Liu, and H. Javaherian, Neural network modeling and adaptive critic control of automotive fuel-injection systems. *Proc. the 2004 IEEE International Symposium on Intelligent Control,* Taipei, Taiwan, pp. 368–373, Sept. 2004.

P. Lee, S. Hong, Y. Lim, C. Lee, B. Jeon, and J. Park, Discrete-time quasi-sliding mode control of an autonomous underwater vehicle. *IEEE J. of Oceanic Engineering*, vol. 24, no. 3, pp. 388–395, July 1999.

X. Li and S. Yurkovich, Neural network based, discrete adaptive sliding mode control for idle speed regulation in IC engines. *J. of Dynamic System, Measurement and Control*, vol. 122, no. 2, pp. 269–275, 2000.

M. X. Lu and N. K. Robert, Modeling, design and implementation of discrete sliding mode control for an engine idle speed control system. *Proc. American Control Conference*, Anchorage, AK, May, 2000.

J. Matas, L. G. de Vicuna, J. M. Guerrero, J. Miret, and O. Lopez, A discrete sliding mode control of a buck-boost inverter. *Proc. IEEE 28th Annual Conference of the Industrial Electronics Society*, Sevilla, Spain, vol. 1, pp. 5–8, Nov. 2002.

J. J. Moskwa and J. K. Hedrick, Nonlinear algorithms for automotive engine control. *IEEE Control Systems Magazine*, vol. 10, pp. 88–93, Apr. 1990.

S. Ouenou-Gamo, A. Rachid and M. Ouladsine, A non-linear controller of a turbocharged diesel engine using sliding mode. *Proc. the 1997 IEEE International Conference on Control Applications*, pp. 803–805, Oct. 1997.

S. Park, M. Yoon, and M. Sunwoo, Feedback error learning neural networks for air-to-fuel ratio control in SI engines. *SAE Technical Paper 2003-01-0356*, 2003.

C. Tsai, H. Chung, and F. Yu, Neuro-sliding mode control with its applications to seesaw systems. *IEEE Trans. Neural Networks*, vol. 15, no. 1, pp. 124–134, Jan. 2004.

V. I. Utkin, *Sliding Modes in Control and Optimbation*, New York: Springer, 1992.

M. Won, S. B. Choi, and J. K. Hedrick, Air-to-fuel ratio control of spark ignition engines using Gaussian network sliding control. *IEEE Trans. Control Systems Technology*, vol. 6, no. 5, pp. 678–687, Sept. 1998.

Kyung-jinn Yang, Jae Weon Choi, Keum-Shik Hong, Advanced sliding mode idle speed control for a nonlinear engine model: coordinated throttle/spark advance control. *Proc. the 36th SICE Annual Conference*, pp. 1301–1304, Jul. 1997.