ICIC Express Letters
Part B: Applications
Volume **7**, Number **4**, April **2016**

ICIC International ©2016 ISSN 2185-2766
pp. 773–780

IMPLEMENTATION OF DC MOTOR POSITION CONTROL USING A CMAC-NEURAL NETWORK APPROACH

Chin-Pao Hung*, Wen-Cheng Pu and You-Cheng Lai

Department of Electrical Engineering
National Chin-Yi University of Technology
No. 57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 41170, Taiwan
*Corresponding author: cbhong@ncut.edu.tw

Received September 2015; accepted November 2015

Abstract. *This paper proposed a novel control architecture with the combination of a PD-type controller and a CMAC-neural network (CMAC-NN) controller to achieve the precision position loop control under system that has the rough model only. PD-type controller is designed as a stabilizer controller and a CMAC-NN is constructed as a learning controller to alleviate the dependency to system parameters. The CMAC-NN detects the output error and the output of PD-type controller to update the CMAC weights by a real-time learning algorithm. Then the summation of PD-type controller and CMAC-NN controller is sent to the DC motor system, and the experimental results demonstrate the effectiveness and robustness of the proposed scheme.*
Keywords: DC servo system, CMAC, PID, Learning controller, Hybrid controller

1. Introduction.

Most traditional controllers design usually requires the mathematical model of system. Unfortunately, an exact plant model is difficult to be obtained in most real systems. Engineers usually spend a lot of time to obtain so called nominal model with bounded uncertainty and accordingly to design a robust control law. That is assuming the system models are unknown or partially known, and traditional control laws are difficult to be implemented. In the past two decades, many researchers proposed intelligent control schemes, such as fuzzy based, neural network based, or hybrid fuzzy neural based schemes, and spent a lot of strength to overcome these problems.

Fuzzy based scheme uses IF-THEN linguistic rule to achieve a robust controller [1-3]. It usually requires expert experience to obtain better performance. Neural network based scheme emphasizes self-learning ability to reduce the dependence on experts [4]. Gorgeous mathematical analysis has proved its feasibility. However, complex calculations often limit the practical application. Learning process usually requires a long computation time and cannot be applied to real time control problem. Other investigators also presented learning control schemes for improving the control performance of traditional control methods or to solve the unknown model problems [5,6]. Based on the repetitive operations, the control objective can be achieved perfectly under system that has invariant system model. It is also not suitable for real-time control problem. For unknown or partially known system, how to use a simple and low-cost method to solve the real time control problem is the main research objective of this paper.

Miller et al. in [7] proposed a real-time learning scheme for industrial manipulator control using the CMAC-NN. In [7], the CMAC-NN is used to learn the inverse dynamic of industrial robot by adjusting the weights in the CMAC-NN on-line based on the observations of the robot input/output relationships. The combination of CMAC and a fixed gain controller makes the real-time control possible. However, Miller's results showed to obtain small enough average position errors require 8 trials at least, and how to improve this scheme is our concern.

Extending our previous research results [8], we construct a different combination of PD-type controller and CMAC-NN to achieve real-time learning control objective. Generally, learning an optimal control signal belongs to an unsupervised learning field. It is a difficult problem because of lacking a teacher as model. The designer must take many indirect learning schemes, such as chain rules, to adjust the memory weights and spent longer calculation time. To accelerate the learning speed, a teacher is required. In our observation, take the PD-type controller's output as the teacher of the CMAC-NN, a virtual teacher is beneficial to the learning operation. Though the PD-type controller's output is not the optimal control signal, it is better than nothing. The simulation and experimental results proved the feasibility and success of the proposed scheme, different to other researchers that only show the simulation results [9,10].

The remainder of this paper is organized as follows. Section 2 is a brief description about the principle of CMAC-NN. Section 3 shows the architecture of the proposed hybrid controller. How to update the memory weight also appears in this section. Experimental system setup, simulation and test results are shown in Section 4. Finally, the conclusions are stated in Section 5.

2. **Brief Description of CMAC-NN.** The CMAC-NN is proposed by Albus [11] in 1970s. It is similar to the neural structure of the human cerebellum and possesses the characteristics of rapid learning and quick responses. In a table look-up fashion, it produced a vector output in response to a state vector input. Just like the models of the human memory, using local data to perform reflexive processing and achieve rapid learning results. Figure 1 schematically depicts the CMAC-NN, and it uses a series of mappings to transform input state S into output value y [4]. The mapping processes that satisfy similar inputs activate similar memory addresses and produce similar outputs; restated, if the input states are close in input space, then their corresponding sets of association cells (fired memory) overlap. For example, if S_1 and S_2 are similar (close), then S_1 activates the memory addresses a_1, a_2, a_3, a_4, and S_2 should activate the memory addresses a_2, a_3, a_4, a_5 or a_3, a_4, a_5, a_6. The outputs are said to be highly similar if two inputs activate the same memory addresses. Lower similarity outputs would activate fewer same memory addresses. Such architecture is the so called supervised learning and CMAC-NN outperforms other neural network on learning speed, such as multi-layer neural network. Therefore, let the output of PD-type controller be y_d and the output of CMAC-NN be y, and CMAC-NN can learn the behavior of PD-type controller quickly and dominate the

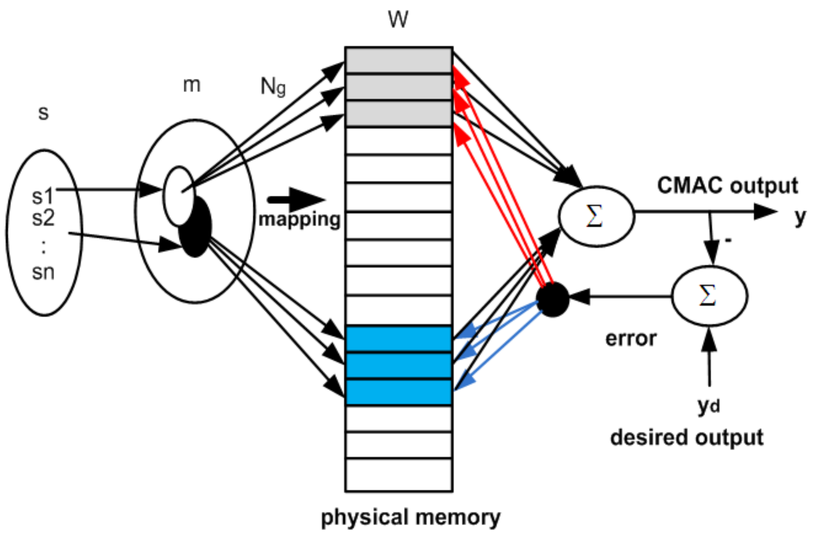

FIGURE 1. Schematic of CMAC neural network

control job gradually. The characteristic of local reflexive action makes CMAC attractive to on-line application in control system.

3. PD-tpye Controller with CMAC-NN Compensator. Figure 2 shows the proposed hybrid controller architecture with PD and CMAC-NN. Even though the system model is partially known only, a PD-type controller is easily tuned by using Ziegler-Nichols method. To improve the control performance, a CMAC-NN is added, as shown in shadow area of Figure 2. The feedback error signals, including position error and velocity error, are sent to the CMAC-NN as the input state space. However, to adjust the memory weight requires a teacher. Otherwise, CAMC don not know how to update the old memory weights. Because PD-type controller usually satisfied elementary requirement, maybe it is not a perfect teacher, but it can be the next best selection. Therefore, in the memory weight update operation, we select the output of PD-type controller as the teacher, and it indicates the CMAC-NN how to adjust the memory weights. Quantization, fired address mapping, CMAC output calculation, and the memory weights adjusted will be discussed as follows.

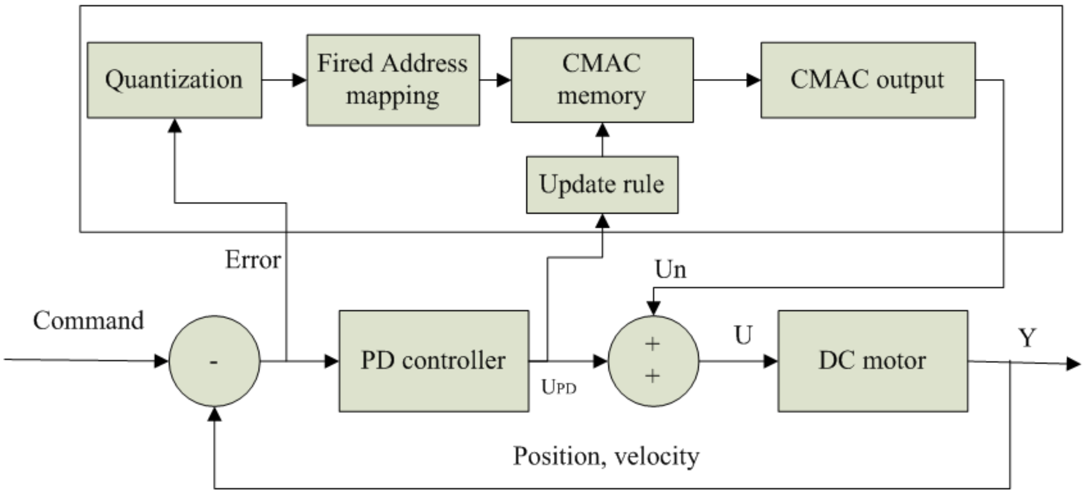

Figure 2. Hybrid controller architecture with PD-type controller and CMAC-NN

3.1. Quantization. The input values are sent to the CMAC-NN, and it first goes through the quantization mapping Q to produce a quantization level output for benefitting following fired address mapping. The quantization output can be described as

$$q_i = Q(x_i, x_{i\min}, x_{i\max}, q_{i\max}), \quad i = 1, \ldots, n \tag{1}$$

where n is the input numbers. The resolution of this quantization depends on the expected maximum and minimum inputs, $x_{i\max}$ and $x_{i\min}$, and on the number of quantization levels, $q_{i\max}$. High resolution will have good generalization ability but more memory is required. Referring to our previous research [12] and Figure 3, the quantization level of each input signal can be calculated as

$$q_{xi}(x_i) = ceil\left((x_i - x_{i\min})/\left[(x_{i\max} - x_{i\min})/(q_{i\max} - 1)\right]\right) \tag{2}$$

where $ceil(x)$, rounds the elements of x to the nearest integers towards infinity.

On dc servo control, the position command and motion speed are subject to the mechanism and the maximum and minimum value can be assigned first.

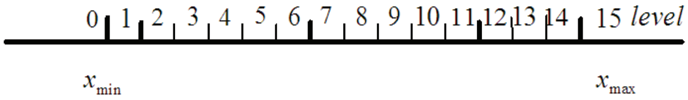

FIGURE 3. Quantization mapping diagram

3.2. **Fired address mapping.** As described above, fired address mapping that must satisfy similar inputs activates similar memory addresses and produces similar outputs. Referring to Table 1, each quantization level then through V mapping outputs A^* segment addresses, A^* the number of association (fired) memory cells. Table 1 lists the mapping of quantization level and the segment address, in which the quantization level $q_{\max}$ is 8 and A^* is 4. For example, the quantization level 3 will map a group segment address $[v_{11}, v_{12}, v_{13}, v_{14}] = [5, 6, 3, 4]$ and 6 will map to $[v_{21}, v_{22}, v_{23}, v_{24}] = [9, 6, 7, 8]$. After obtaining the segment address of each input, the concatenating operation of the following will generate the fired memory address used for recording the corresponding characteristics value,

$$v_j = concat(v_{j1}, v_{j2}, \dots, v_{jn}), \quad j = 1, \dots, A^* \tag{3}$$

For example, assuming the feedback signal error and change of error equal $(3, 6)$ then the segment addresses generated by error are $[v_{11}, v_{12}, v_{13}, v_{14}] = [5, 6, 3, 4]$ and by change of error are $[v_{21}, v_{22}, v_{23}, v_{24}] = [9, 6, 7, 8]$. Then the concatenation operation can be expressed as follows

$$\begin{aligned} V_1 &= concat[v_{11}, v_{21}] = concat[5, 9] = 01011001B \\ V_2 &= concat[v_{12}, v_{22}] = concat[6, 6] = 01100110B \\ V_3 &= concat[v_{13}, v_{23}] = concat[3, 7] = 00110111B \\ V_4 &= concat[v_{14}, v_{24}] = concat[4, 8] = 01001000B \end{aligned}$$

TABLE 1. Segment address mapping

Segment address								
:							:	:
9						v_1	v_1	v_1
8					v_4	v_4	v_4	v_4
7				v_3	v_3	v_3	v_3	
6			v_2	v_2	v_2	v_2		
5		v_1	v_1	v_1	v_1			
4	v_4	v_4	v_4	v_4				
3	v_3	v_3	v_3					
2	v_2	v_2						
1	v_1							
	1	2	3	4	5	6	7	8

Quantization level

3.3. **Output mapping.** The finial mapping computes the output y by summing the weights w_{Vj} located at the fired memory addresses. It can be described as

$$y = \sum_{j=1}^{A^*} w_{Vj} \tag{4}$$

3.4. **Updating the fired memory weights.** On each control cycle ending, the hybrid controller updates the fired memory addresses weights using the following steepest-descent update rule:

$$w_{v_i}^{new} \leftarrow w_{v_i}^{old} + \beta \frac{y_d - y}{A^*}, \quad i = 1, 2, \ldots, A^* \tag{5}$$

In this equation, y_d is the output of PD-type controller, y the CMAC-NN output, and $0 < \beta \leq 1$ the learning gain.

4. **Simulation and Experimental Results and Discussion.** In order to demonstrate the feasibility of the proposed scheme, an experimental dc servo system is built as shown in Figure 4. The control kernel is a dsPIC30F3010 MPU and the test data are recorded on MPLAB workspace and plotted using EXCEL software. The built system has a rough model. Figure 5 shows the simulation results using Matlab software and Figure 6 is the

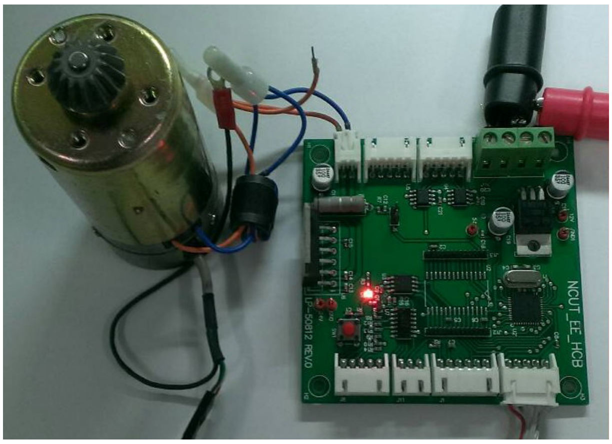

FIGURE 4. Experimental system of dc servo motor

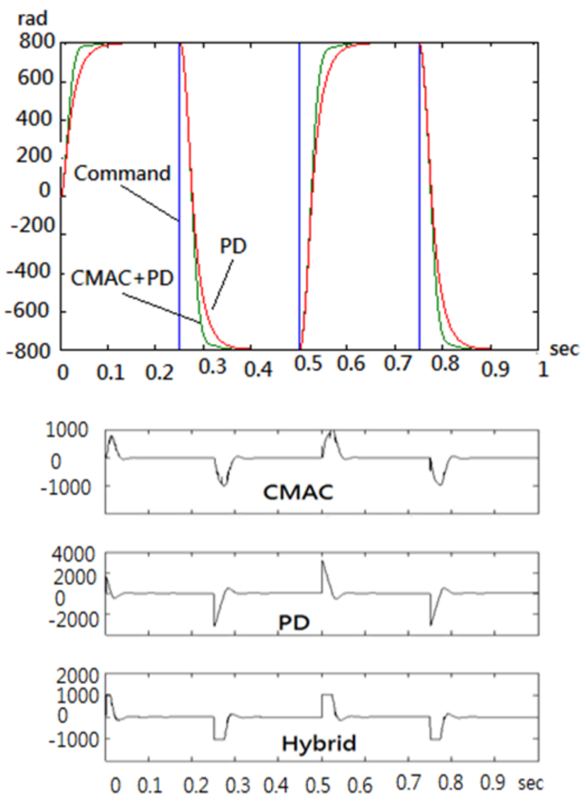

FIGURE 5. Time response and control commands for tracking 2Hz square wave

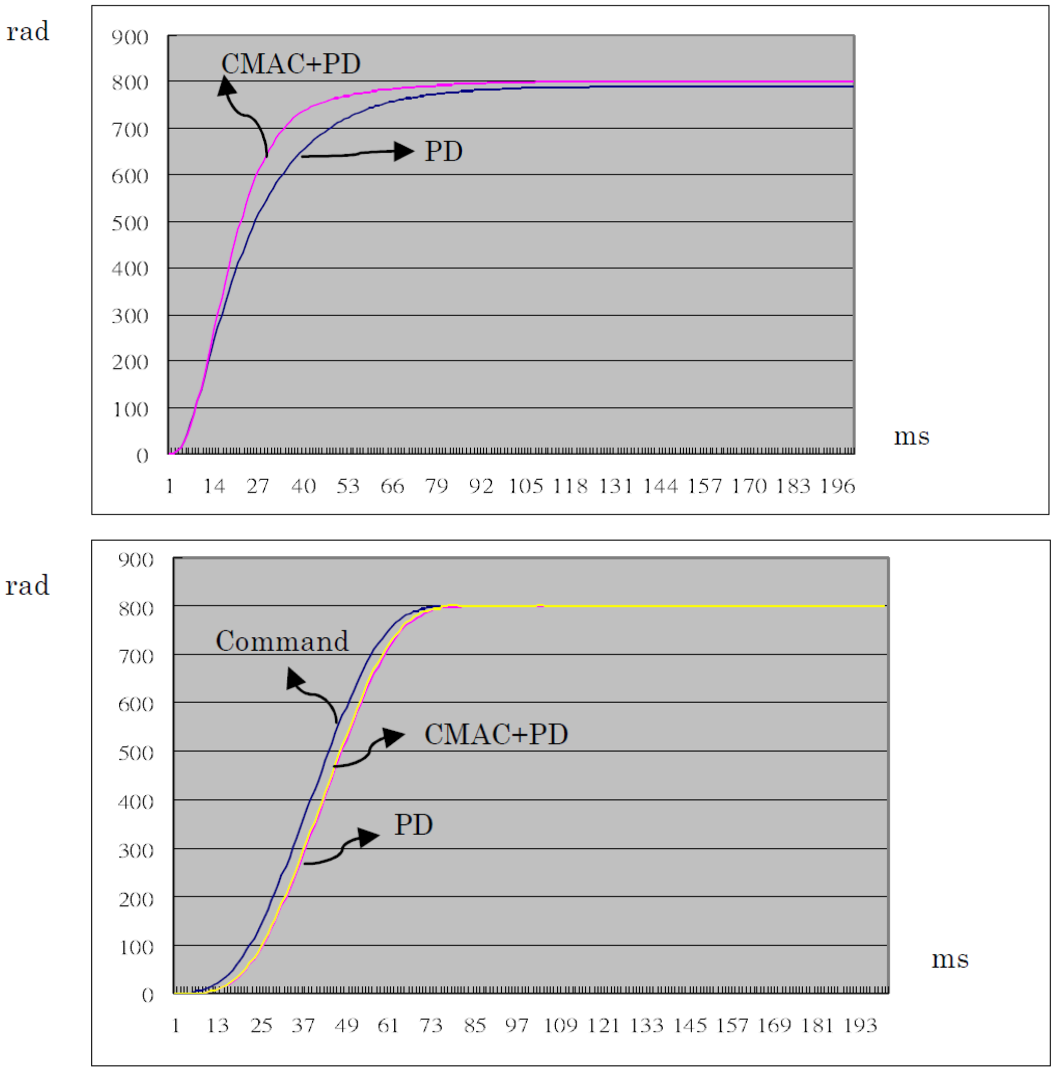

FIGURE 6. Time response for step input and S curve input command

TABLE 2. Design parameters of CMAC-NN

$q_{\max}$	A^*	$x_{1\max}$	$x_{1\min}$	$x_{2\max}$	$x_{2\min}$	β	k_p	k_d	Open loop model
7	4	800	-800	40	-40	0.9	37	0.119	$4512/(s(s+76.92))$

experimental results. The associated design parameters of CMAC-NN are list as Table 2. In Figure 5, input command is 2Hz blue square wave. Red line is the output response using only PD controller and green line is the output response using hybrid controller. It is clear, adding the CMAC-NN achieves better tracking performance. Right side of Figure 5 shows the PD part, CMAC part and the hybrid control signal. In Figures 6 and 7, the experimental results also show the better performance for step, S curve, and S square wave tracking ability.

5. **Conclusions.** This paper proposes a hybrid control architecture integrating PD-type and CMAC-NN to solve the rough system model problem. By adding CMAC-NN as a compensator and letting the output of PD control as a teacher, it makes the proposed scheme can be implemented easily by lower cost. The simulation and experimental results demonstrate the feasibility and success of the proposed scheme. Because the space is limited, more rigorous theoretical proof will appear soon in the near future.

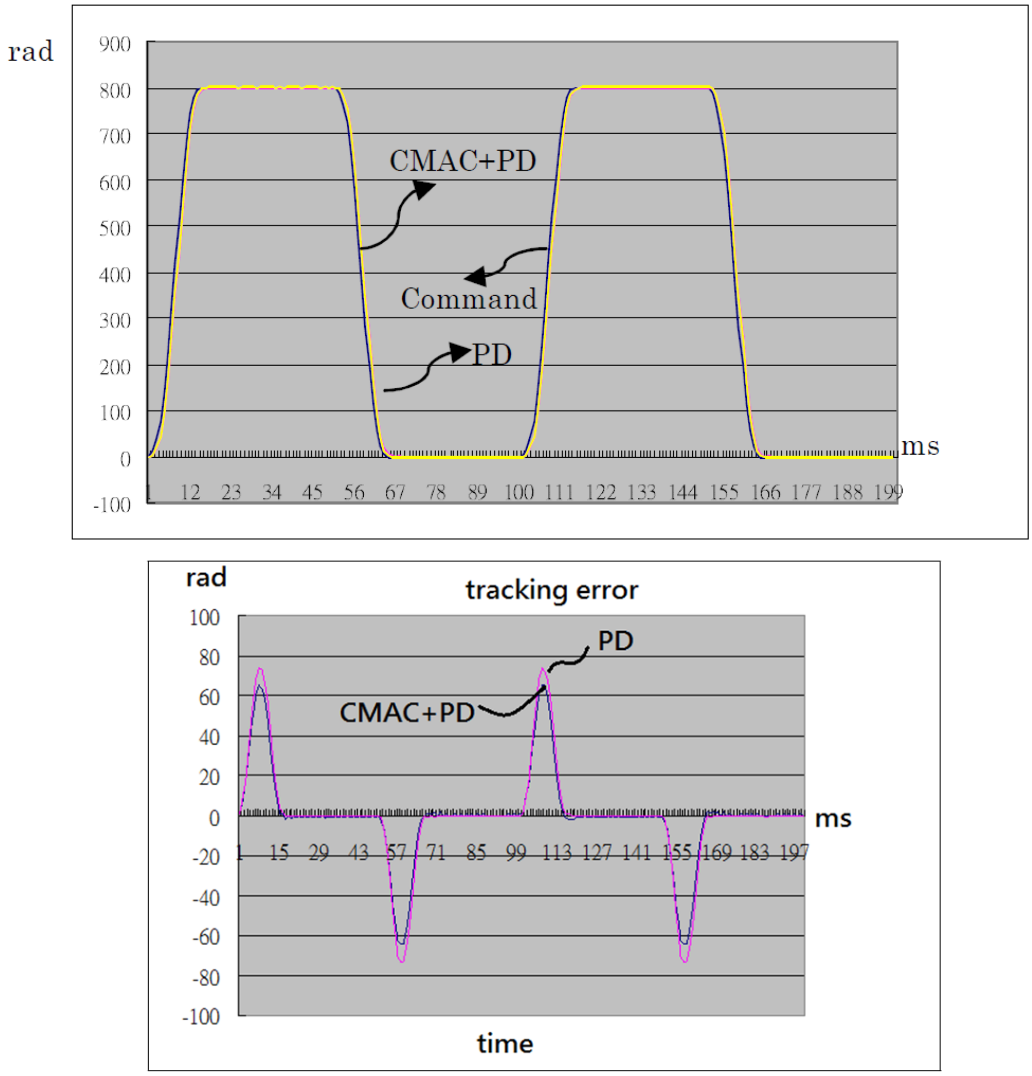

Figure 7. Time response and output error for tracking S square wave command

Acknowledgements. The authors gratefully acknowledge the support of Chin-Yi University of Technology under Research Group Plan and the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] C. C. Kung and C. C. Liao, Fuzzy-sliding mode controller design for tracking control of non-linear system, *Proc. of American Control Conf.*, pp.180-184, 1994.

[2] B. Yoo and W. Ham, Adaptive fuzzy sliding mode control of nonlinear system, *IEEE Trans. Fuzzy System*, vol.6, no.2, pp.315-321, 1998.

[3] L. X. Wang, Stable adaptive fuzzy control of nonlinear system, *IEEE Trans. Fuzzy System*, vol.1, no.2, pp.146-155, 1993.

[4] D. A. Handeiman, S. H. Lane and J. J. Gelfand, Integrating neural networks and knowledge-based systems for intelligent robotic control, *IEEE Control System Magazine*, pp.77-86, 1990.

[5] S. Kawamura, F. Miyazaki and S. Arimoto, Realization of robot motion based on a learning method, *IEEE Trans. Syst., Man, Cybern.*, vol.18, no.1, pp.126-134, 1988.

[6] S. R. Oh, Z. Bien and I. H. Suh, A model algorithmic learning method for continuous-path control of a robot manipulator, *Robotica*, vol.8, pp.31-36, 1990.

[7] W. T. Miller, R. P. Hewes, F. H. Glanz and L. G. Kraft, Real-time dynamical control of an industrial manipulator using a neural-network-based learning controller, *IEEE Trans. Robotics Automat.*, vol.6, no.1, pp.1-9, 1990.

[8] C. P. Hung, Integral variable structure control of nonlinear system using a CMAC neural network learning approach, *IEEE Trans. Syst., Man, and Cyber. Part B*, vol.34, no.1, pp.702-709, 2004.

[9] M. F. Yeh and S. P. Hsu, A single-input CMAC control system, *Proc. of CACS Automatic Control Conference*, 2005.

[10] M. F. Yeh and C. H. Tsai, Standalone CMAC control system with online learning ability, *IEEE Trans. Syst., Man, and Cyber. Part B*, vol.40, no.1, pp.43-53, 2010.

[11] J. S. Albus, A new approach to manipulator control: The cerebeller model articulation controller, *Trans. ASME J. Dynam., Syst., Meas., and Contr.*, vol.97, pp.220-227, 1975.

[12] C.-P. Hung, W.-G. Liu and H.-J. Su, Fault diagnosis of steam turbine-generator sets using CMAC neural network approach and portable diagnosis apparatus implementation, *Lecture Notes in Computer Science*, pp.724-734, 2009.