STRUCTURAL CONTROL USING CMAC NEURAL NETWORK

Dong-Hyawn Kim, Kyu-Hong Shim, In-Won Lee
Dept. of Civil Eng., Korea Advanced Institute of Science & Technology, Republic of Korea

Ju-Won Oh
Dept. of Civil and Env. Eng., Hannam University, Republic of Korea

Abstract

A new neuro-control method is proposed. Cerebeller Model Articulation Controller (CMAC) having quick learning convergence is used for vibration controller. Because the learning speed of multilayer-type neural network is very slow, it cannot be used as a real-time controller. CMAC, however, can serve as a real-time controller. CMAC is trained as a controller and the convergence is compared with that of multilayer neural network. CMAC shows great possibility as a controller that learns very quickly. In the simulation study, three-story shear building is controlled. The structure is assumed to be linear and nonlinear in each example. An active mass damper (AMD) is installed at the roof of the structure. To make it practical, the dynamics of hydraulic actuator and time delay of the controller are considered.

Keywords

Control, Vibration, Neural Netwrok, CMAC, Training

1. Introduction

Artificial neural networks (ANNs) have emerged as new promising tools of many engineering problems. ANNs do not use mathematical model to obtain solutions but use input-output relationships that can be measured from the concerning systems. Final solution is obtained through learning those relationships. Once ANN is trained, it can predict desired output, namely solution, to a given input based on trained intelligence. The reason why ANNs appeals engineers is that it can generalize the trained relationships to the neighborhood of them. Therefore, it can solve problems with limited training data.

Structural control has been one of the most frequent application areas of ANNs during last decade. ANNs have easily designed controller of structures with nonlinearity and uncertainty that had been challenging tasks of conventional control approaches. Learning property of ANNs make it possible to solve those problems. With high nonlinearity and modeling uncertainty, it is not easy or impossible to design a controller by conventional approaches that are based on mathematical model of structure. However, design methodology based on ANNs needs not the mathematical model. Instead of the model, only the structural responses are used for design of ANN controller, simply called the neuro-controller.

A number of structural control algorithm using ANNs have been proposed by many researchers such as J. Ghaboussi et al., H. M. Chen et al., K. Nikzad et al., K. Bani-Hani et al., J. T. Kim et al. Multilayer neural network (MLNN) is used in all their works. It is the most widely-used type of neural network due to its simplicity of structure and learning algorithm. However, it has not been pointed out that the training convergence of MLNN is very slow. Hence, all of the control algorithms now proposed are based on off-line training. In result, one has to wait long for the end of training process before vibration control is applied by trained neuro-controller. But there are some needs for on-line learning controller such as the control of damaged structures or time-varying structures. Therefore, new training algorithms or even new neural networks having fast learning convergence is required.

To speed up the training process, it is proposed that cerebellar model articulation controller

(CMAC) can be applied to structural control. CMAC is one of the ANNs which is known to be very fast in learning. It has been usually applied to the control of robot manipulators that needs fast learning convergence. For the first time, CMAC is applied to control of civil structures in this study. Training rule of CMAC is derived based on cost function proposed by D. H. Kim et al. In addition, sensitivity evaluation algorithm also proposed by D. H. Kim et al is used. Therefore, only one neural network is used in constructing control system.

In the simulation study, three-story shear building is controlled. The structure is assumed to be linear and nonlinear, respectively, in the numerical examples. An active mass damper (AMD) is installed at the roof of the structure for the generation of control force. The dynamics of hydraulic actuator and time delay of the controller are also considered.

2. Cerebellar model articulation controller

2.1. Overview

Cerebellar model articulation controller (CMAC) was first proposed by J. S. Albus in 1975. CMAC was invented by imitating the function of human cerebellum. The schematic diagram of CMAC is shown in Fig. 1. If a vector is inputted to CMAC, a set of addresses in the space of associate cell vector is activated. Then, the output is obtained by summing the weights stored in the activated addresses. If another vector that slightly differs from the first one is inputted, another set of addresses that slightly differs from the first set is activated. Hence, the output also slightly differs from the first one. This can be done by sharing memories. The memory sharing is the so-called hash mapping. It is indispensable for CMAC to share memory for the saving of physical memory. Without the hash mapping CMAC may not solve any engineering problems due to the limitation of memory.

CMAC uses only the local information in making output. And simultaneously, the learning of the weights is done very locally; i.e., only the weights at the activated memory are updated. Therefore, the speed of learning is very fast compared with MLNN whose learning is accomplished globally.

2.2. Elements of CMAC

Moving input space

Each input space is first shifted to the right-half plane in real axis for the simple formulation of the addressing the activated memory. If x_i is the i-th element of input vector to CMAC, it can be done through

$$\bar{x}_i = x_i - x_{i,\min} \; ; \quad i = 1, 2, \cdots, \Omega \tag{1}$$

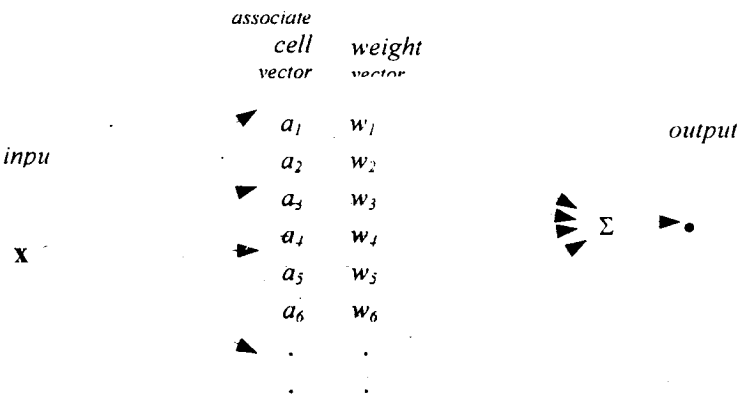


Fig. 1 Schematic diagram of CMAC

where $x_{i,\min}$ is the minimum of x_i, and Ω is the dimensionality of input space. Then, all the elements of input vectors have minimum values of zero and remain in positive domain.

Addressing associate memory (hash mapping)

Let's consider the case of CMAC having two inputs and single output. Each input variable, $\bar{x}_1$ and $\bar{x}_2$, is quantized to the size of *block* as shown in Fig. 2. Then, a mesh is constructed by the block quantization. A cell divided by the mesh is the so-called *hypercube*. A hypercube can be seen as a physical memory. Shifting the first mesh to $S_2 = [s_{21} \quad s_{22}]^T$ makes the second mesh as shown in Fig. 2 (b). s_{ij} denotes the shifting value of j-th direction on the i-th mesh. That is, s_{21} stands for the amount of the shift of the first variable, $\bar{x}_1$, on the second mesh. The shifting vector S_1 equals $[0 \quad 0]^T$, because there is no shifting in the first quantization. The number of meshes is called *generalization width*. In this example the generalization width is two. For an input $\bar{\mathbf{x}}^* = [\bar{x}_1^* \quad \bar{x}_2^*]^T$ that is dotted on Fig. 2, hypercubes of A_{11} and A_{34} are selected on each mesh. Then the weights corresponding to the hypercubes are activated. In general, when the generalization width is N_g and the dimensionality of input space is Ω, the indices of the activated weights in physical memory are calculated as

$$I_m = ceil\left(\frac{\bar{x}_1 - s_{m1}}{q_1}\right) + \sum_{n=2}^{\Omega}\left[ceil\left(\frac{\bar{x}_n - s_{mn}}{q_n}\right)\cdot\prod_{j=1}^{n-1}(b_j+1)\right] + I_m^*, \quad m = 1, 2, \cdots, N_g \tag{2}$$

where $ceil(y)$: the least integer greater than or equal to y.

s_{mn} : the amount of the shift of the n-th variable on the m-th mesh.

q_n : the quantization interval of the n-th variable.

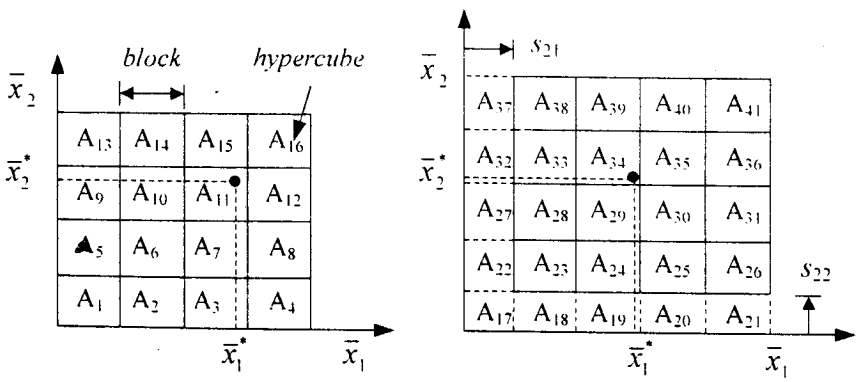
b_j : the number of blocks covering the domain of the j-th variable.

I_m^* : the starting position of the address for the m-th mesh and can be expressed as

$$\begin{cases} I_{m-1}^* + \prod_{n=1}^{\Omega}(b_n+1), & (m \geq 2) \\ 1 & (m = 1) \end{cases} \tag{3}$$

Calculating output and training

The calculation of the output of CMAC is simple. The weights stored in the activated addresses are summed as



(a) first quantization **(b) second quantization**

Fig. 2 Quantization scheme for hash mapping

$$u = \sum_{m=1}^{N_g} w(I_m) \tag{4}$$

where $w(I_m)$ is the weight stored at the I_m-th location. To make an adequate output to a given input, the CMAC should be trained, i.e., the weights are updated so that the output is close to the desired one within allowable error. The training rule of CMAC as a controller will be described in the next chapter.

Required size of memory

Since the basic idea of CMAC is to store information in table look-up fashion, the size of memory is extremely large compared with MLNN. The total number of weights used for CMAC is

$$N_w = N_g \prod_{n=1}^{\Omega} (b_n + 1) \cdot \tag{5}$$

To reduce the size of required memory, the hash-mapping algorithm described in the previous section is used. If there are Ω inputs, and each input can take r_c different values, then the required size of memory to store full information in table look-up mapping is r_c^{Ω}. However, the size is reduced to $N_g(r_c/N_g)^{\Omega}$ in hash-mapping algorithm. The ratio of the two is $N_g^{1-\Omega}$. For example, if n equals 3, and N_g equals 50, the ratio will be $50^{-2}=4\times10^{-4}$. Therefore, the hash mapping is a requisite to the implementation of CMAC into hardware. Without the mapping, CMAC may not be used in any engineering problems due to the limitation of hardware memory.

3. Vibration Control using CMAC

3.1. Control algorithm and training rule

Fig. 3 shows the block diagram for the proposed neuro-control. CMAC serves as a controller in the diagram. To train the CMAC neuro-controller, a cost function comprised of the structural response and the control signal is defined as

$$\begin{aligned} J &= \sum_{k=0}^{N_f-1} J_k \\ &= \sum_{k=0}^{N_f-1} \frac{1}{2}\left(\mathbf{z}_{k+1}^T \mathbf{Q} \mathbf{z}_{k+1} + \mathbf{u}_k^T \mathbf{R} \mathbf{u}_k\right) \end{aligned} \tag{6}$$

where $\mathbf{z}(n\times1)$ and $\mathbf{u}(m\times1)$ are the state and the control signal; $\mathbf{Q}(n\times n)$ and $\mathbf{R}(m\times m)$ the weighting matrices; k, N_f sampling number, total number of sampling time, respectively. By applying the gradient descent rule to the cost at the k-th step (J_k), the weight update at the k-th step

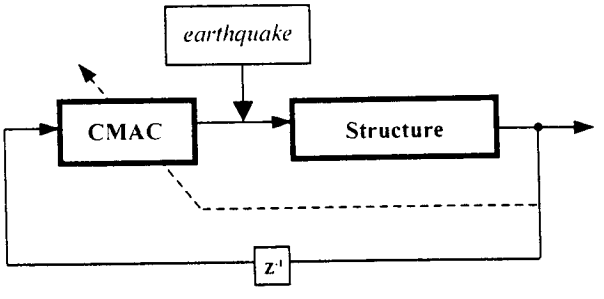


Fig. 3 Control diagram for the proposed neuro-control

can be expressed as

$$\Delta w = -\eta \frac{\partial J_k}{\partial w} \tag{7}$$

where η is the rate of training. By chain rule, the derivative of Eq. (7) can be rewritten as

$$\frac{\partial J_k}{\partial w} = \frac{\partial J_k}{\partial u} \frac{\partial u}{\partial w} \tag{8}$$

In the above equation, let's define the generalized error as

$$\delta = -\frac{\partial J_k}{\partial u} \tag{9}$$

Then, the amount of weight update can be derived as

$$\Delta w = \eta \delta \tag{10}$$

where

$$\delta = -\left(\mathbf{z}_{k+1}^T \mathbf{Q} \left[\frac{\partial \mathbf{z}_{k+1}}{\partial \mathbf{u}_k} \right] + \mathbf{u}_k^T \mathbf{R} \right) \tag{11}$$

The response sensitivity in Eq. (11) can be obtained through the sensitivity evaluation algorithm proposed by D. H. Kim et al.

3.2. Limitations of CMAC

Although CMAC quickly learns functional input-output relationships, it should be noticed that it has limitation in application. Since it stores information to a few localized memories, the size of memory is enormously increased in storing full information. In addition, since any one instance of training data is used to train only a few localized weights, the training data should be scattered as uniformly as possible for having generalization potential. This usually increases the size of training data. Therefore, the memory mapping and generalization algorithm should be developed to widen application areas of CMAC.

4. Numerical examples

4.1: Structure with Active Mass Damper (AMD) System

Fig. 4 shows three-story example structure. An AMD attached to the roof and driven by actuator applies reaction force to the structure. Since the dynamics of the actuator and the structure are coupled, actuator dynamics should be included in analysis.

Hydraulic actuator

The equation of motion of hydraulic actuator can be divided in two parts, the valve dynamics and the piston equation. First, valve equation is expressed as

$$\frac{\tau}{g_1 g_2} \dot{q} + \frac{1}{g_1 g_2} q = u \tag{20}$$

where g_1 and g_2 denote the valve gains; τ the time constant of the valve; q and u the flow rate of oil and the control signal, respectively. The change of the oil flowing through the valve induces the motion of the piston and the relationship can be modeled by

$$a_r \dot{x}_r + \frac{c_l}{a_r} f + \frac{V}{2\beta a_r} \dot{f} = q \tag{21}$$

where a_r, β, c_l and V denote the area of the piston, the compressibility coefficient, leakage coefficient, and the volume of the cylinder respectively; x_r the relative displacement between the roof and the piston; f the force exerted on the structure by the AMD.

Three-story shear building

The equation of motion of the 3-story building with an AMD can be expressed as

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}(\mathbf{x}, \dot{\mathbf{x}}) = \mathbf{L}f - \mathbf{M}[1]\ddot{x}_g \tag{22}$$

where $\mathbf{M}$ and $\mathbf{C}$ are (4×4) mass and damping matrices; $\mathbf{x}$ is the (4×1) relative displacement vector consisting of three stories and an AMD; $\mathbf{K}(\mathbf{x}, \dot{\mathbf{x}})$ is the (4×1) restoring-force vector; $\mathbf{L}$ is the (4×1) vector indicating the location of the actuator; $\ddot{x}_g$ is ground acceleration; $[1]$ is the direction vector of ground motion.

Nonlinear dynamic model

Nonlinear model proposed by Baber and Wen is used to simulate the motion of nonlinear structure. The model has been used for the control simulation in many works. The restoring force of the model is composed of the linear and the nonlinear terms as

$$k_s(x_s, \dot{x}_s) = \alpha k_0 x_s + (1-\alpha) k_0 dy \tag{23}$$

where x_s denotes the inter-story displacement; k_0 and α are the linear stiffness and its contribution to restoring force, respectively. And d and y are the constant and the variable, respectively, satisfying the following equation.

$$\dot{y} = \frac{1}{d}(\rho \dot{x}_s - \mu |\dot{x}_s| |y|^{p-1} y - \sigma \dot{x}_s |y|^p) \tag{24}$$

where ρ, μ and σ are the constants that affect the hysteretic behavior.

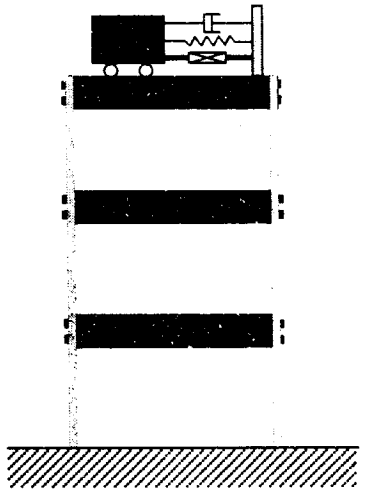


Fig. 4 Shear building with active mass damper (AMD)

4.2. Simulation

Parameters

The structural properties are as follows: story mass equals 200kg; inter-story stiffness 2.25×10^5N/m; damping ratios of three modes are 0.6, 0.7 and 0.3% respectively. The parameters of AMD are designed based on the suggestions for optimal tuned mass damper (TMD). Mass of AMD is 18kg that corresponds to the 3% of the total mass of the structure. The actuator parameters are adopted from the reference as follows: $g_1 = 32.14\times10^2$ and $g_2 = 2.8\text{m}^3/\text{sec}$; $\tau = 0.1$; $a_r = 1.52\times10^{-3}\text{m}^2$; $V = 4.56\times10^{-4}\text{m}^3$; $c_l = 1.0\times10^{-11}$; $\beta = 2.1\times10^9$. For the linear structure $\alpha = 1.0$ is used and the nonlinear structure $\alpha = 0.5$. For nonlinear model d equals 0.01; ρ, μ, σ and p are 1.0, 0.5, 0.5 and 5.0 respectively. The sampling time is 0.005sec, delay time is assumed to 0.0005sec. The equation of motion is integrated at every 0.00025sec on Matlab 5.1.

Sensitivity

When the states of structure are all zero, 1.0volt control signal is applied to the actuator. Then, the acceleration response of the third floor is measured during one sampling time. The measured acceleration is shown in Figure 6. Random white noise with amplitude 0.1g is added to the measured signal. Then, velocity and displacement responses are obtained by integrating the measured acceleration.

Training CMAC

To train a CMAC, cost function of Eq. 25 is defined. Only the state of the third floor normalized to the uncontrolled responses participates in the cost function. The cost at the k-th step used for training criterion is

$$J_k = \mathbf{z}_{3,k+1}^T \mathbf{Q} \mathbf{z}_{3,k-1} + r u_k^2 \tag{25}$$

where $\mathbf{z}_{3,k-1}$ and u_k denote the state of third floor and control signal, respectively. And weighting matrix $\mathbf{Q}$ and r are as follows.

$$\mathbf{Q} = \begin{bmatrix} \left|\frac{1}{\tilde{x}_3}\right|^2 & 0.0 \\ 0.0 & \left|\frac{1}{\dot{\tilde{x}}_3}\right|^2 \end{bmatrix}, \quad r = 0.1\left|\frac{1}{\tilde{u}}\right|^2 \tag{26), (27}$$

In these equations, $\tilde{x}_3$ and $\dot{\tilde{x}}_3$ are the maximum displacement and velocity of third floor under El Centro earthquake when control input is off. $\tilde{u}$ is the maximum control input voltage.

CMAC has two inputs, relative displacement and velocity of the third floor. Each input space has two quantization blocks. And the generalization width is 200. Therefore, the total number of weights, N_w, equals $200\prod_{i=1}^{2}(2+1) = 1800$.

CMAC was trained during 10sec ground motion of El Centro earthquake. The convergence curve of cost function is compared with the case of MLNN in Fig. 5. After 500 epochs the final costs for CMAC and MLNN are compared in Table 1. The cost for CMAC is slightly lager than that for MLNN. Epochs until the cost reach below the same values are compared in Table 2. It can be said that CMAC has great potential for the real-time controller.

The reason why the minimum cost function when using CMAC is slightly larger than the case of

MLNN may be said that the output resolution of CMAC may not be sufficient to describe full information for control space. Hence, by increasing output resolution, the convergent point of the cost function can be decreased.

Table 1 Final value of cost functions after 500 epochs

linear (α=1.0)		nonlinear (α=0.5)	
MLNN	CMAC	MLNN	CMAC
1.77×10^{-2}	1.94×10^{-2}	1.91×10^{-2}	2.02×10^{-2}
(1.00)	(1.09)	(1.00)	(1.06)

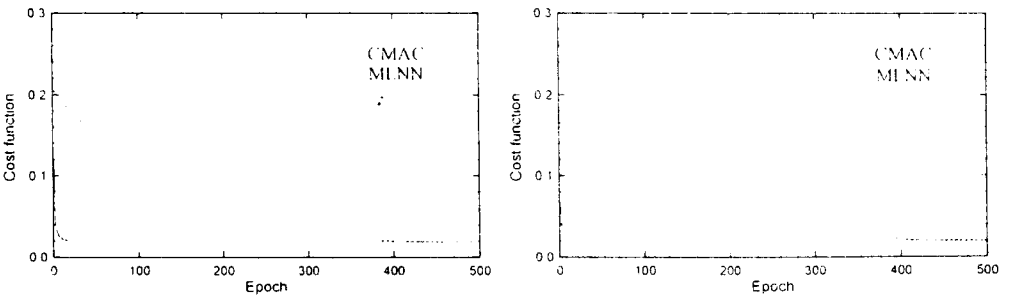
Table 2 Training epochs to reach same costs

linear (α=1.0) ($J=1.94\times10^{-2}$)		nonlinear (α=0.5) ($J=2.02\times10^{-2}$)	
MLNN	CMAC	MLNN	CMAC
412	65	427	34
(1.00)	(0.15)	(1.00)	(0.08)

Control results

Fig. 6 compares the controlled responses when MLNN and CMAC are used for neuro-controller, respectively. Since the cost function in the case of CMAC converges to slightly larger value than that of MLNN, the controlled response with CMAC is slightly larger than that of MLNN. However, the difference is not serious.

Control results for both linear and nonlinear structure under Northridge earthquake (1994) are shown from Figs. 7 to 8. They show that vibration is successfully controlled when using CMAC as a controller.



(a) linear case (α=1.0) (b) nonlinear case(α=0.5)

Fig. 5 Cost function vs. epoch

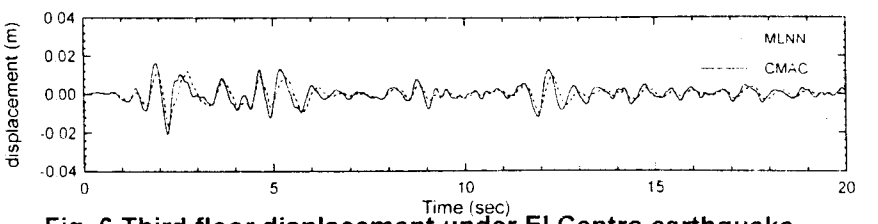


Fig. 6 Third floor displacement under El Centro earthquake

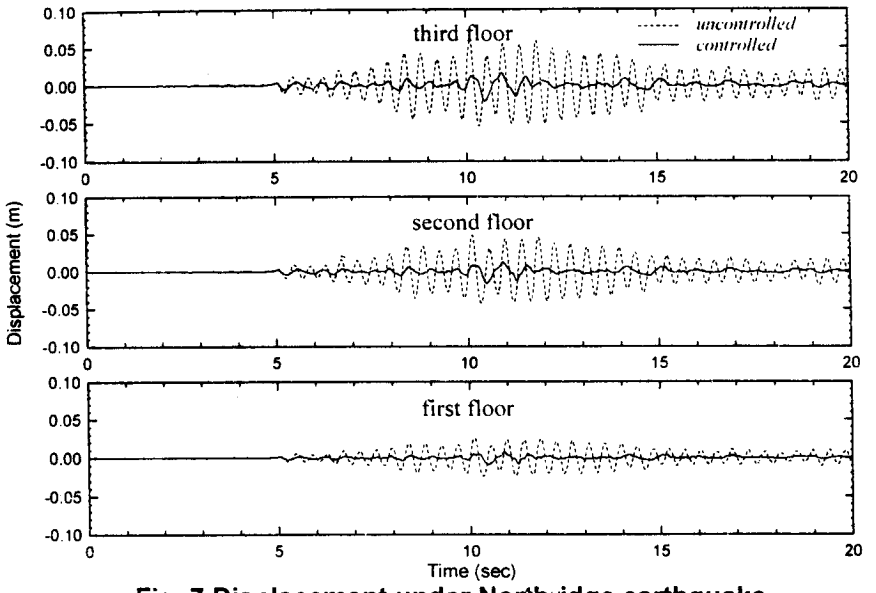
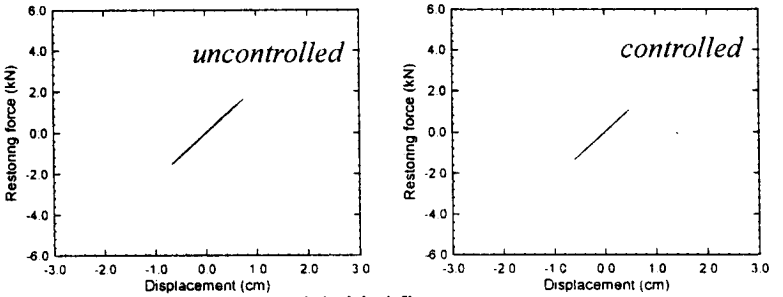
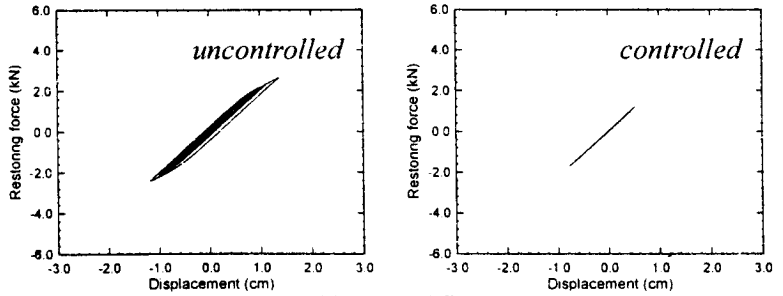


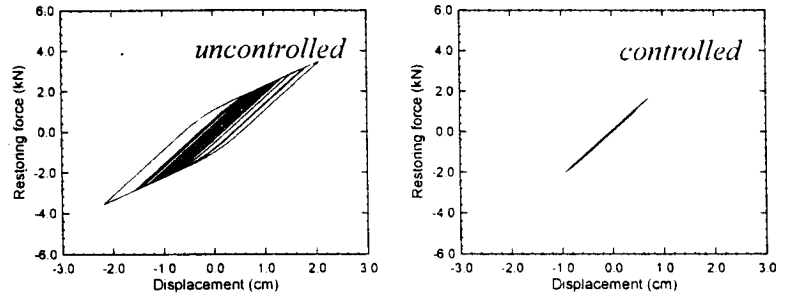
Fig. 7 Displacement under Northridge earthquake



(a) third floor



(b) second floor



(c) first floor

Fig. 8 Restoring force vs. displacement under Northridge earthquake

5. Conclusions

CMAC is exploited for the control of structural vibration. The structure and training algorithm of CMAC is presented. The training convergence of CMAC is quite fast compared with MLNN. In the numerical example, when CMAC is used for controller, it takes only 15% (linear case) and 8% (nonlinear case) of the epoch taken by MLNN for cost function to reach below the same values. It is expected, therefore, that CMAC can be used for real-time learning controller.

Acknowledgement

This research is supported by the National Research Laboratory (NRL) Program for Aseismic Control of Structures. The financial support is gratefully acknowledged.

References

[1] J. T. Kim, H. J. Jung and I. W. Lee, *Optimal structural control using neural networks*, Journal of Engineering Mechanics, ASCE, Vol. 126, No. 2, pp. 201-205, 2000.

[2] D. H. Kim, C. B. Yun and I. W. Lee, *Vibration control of steel structure using neuro-controller trained with sensitivity data*, Proceedings of Korena Society of Steel Construction, Seoul, Jun. 3, 2000

[3] H. M. Chen, K. H. Tsai, G. Z. Qi, J. C. S. Yang, and F. Amini, *Neural network for structural control*, Journal of Computing in Civil Engineering, ASCE, Vol. 9, No. 2, pp. 168-176, 1995.

[4] J. Ghaboussi, A. Joghataie, *Active control of structure using neural networks*, Journal of Engineering Mechanics, ASCE, Vol. 121, No. 4, pp. 555-567, 1995.

[5] J. N. Yang, Z. Li, S. Vongchavalitkul, *Generalization of optimal control theory : linear and nonlinear control*, Journal of Engineering Mechanics, ASCE, Vol. 120, No. 2, pp. 266-283, 1994.

[6] J. S. Albus, *A new approach to manipulator control : the cerebellar model articulation controller*, ASME Trans. on Dynamic Systems Measurements & Control, Vol. 97, pp. 220-227, 1975.

[7] K. Bani-Hani, and J. Ghaboussi, *Nonlinear structural control using neural networks*, Journal of Engineering Mechanics, ASCE, Vol. 124, No. 3, pp. 319-327, 1998.

[8] K. Nikzad, and J. Ghaboussi, *Actuator dynamics and delay compensation using neurocontrollers*, Journal of Engineering Mechanics, ASCE, Vol. 122, No. 10, pp. 966-975, 1996.

[9] T. T. Soong and G. F. Dargush, *Passive Energy Dissipation Systems in Structural Engineering*, John Wiley & Sons, West Sussex, England, pp. 227-235, 1997.

[10] T. T. Baber and Y. K. Wen, *Random vibration of hysteretic degrading systems*, ASCE Journal of Engineering Mechanics, Vol. 107, No. 6, pp. 1069-1087, 1981.

[11] The Math Works, Inc., MATLAB, Natick, MA, 1997.