Simulation study of CMAC control for the robot joint actuated by McKibben muscles

Huailin Zhao[*,**] and Masanori Sugisaka[*]

[*] *Dep. of Electrical & Electronic Eng., Oita University, Japan*
[**] *School of Physical eng. & Tec., Zhengzhou University, China*
(Tel : 81-97-554-7841; Fax : 81-97-554-7841)
(zhao_Huailin@yahoo.com, msugi@cc.oita-u.ac.jp)

Abstract: The paper simply introduces the CMAC model. It explains how to train the CMAC off-line by the data from fuzzy control theory. Then two control systems based on the CMAC for controlling a humanoid robot joint actuated by a pair of McKibben muscles are designed. One is based on pure CMAC and another one based on both the CMAC and PID. The simulations show the control systems are stable. As to the two controllers, the paper discusses their on-line learning algorithms and compares them with each other.

Keywords: CMAC, McKibben muscle, robot joint, simulation

1. Introduction

In our lab a humanoid bipedal robot actuated by McKibben muscles is constructed. Each joint is actuated by at least two McKibben muscles. We try to control the whole robot and start from a single joint. A one-dimension robot joint actuated by a pair of McKibben muscles is established, shown in Fig.1. Fig.2 is its input and output model. It makes the system become a single-input and single-output system. This will make the control system simpler. The joint system is a serious nonlinear system and there are uncertain factors such as load varying, friction and outer disturbing. Its exact modeling is difficult. The CMAC as an intelligent control algorithm is selected. It doesn't need precise model of the object. And its learning is based on local approximation instead of the global approximation. So it is fit to on-line learning. The robot joint can be controlled in time.

2. Control algorithm based on CMAC

2.1 CMAC neural network

The CMAC is a simple and fast associative memory type neural network based on the local approximation [1-2]. A CMAC model example is shown in Fig.3. The input and output are related by a receptive area in which each block corresponds to a neuron. Each output only relates with a small part of the neurons. So the weights to be corrected are not many, the learning is very fast. Therefore it is fit to on-line learning. This is very useful to control robots, because the robot needs the real time control. It has local generalization ability – similar inputs will get the similar outputs and long-range inputs will get almost independent outputs. There are some important parameters - the generalization parameter n and the quantization step q. One point in the input space will activate the related n neurons in the receptive area at the same time. Quantization step q will decide how the input space is divided. Both n and q will decide the final structure of the CMAC network. The product of n and q will decide the generalization area. There are two arrays related with the CMAC network – C and W. Array C is the associative vector. It indicates which and how many neurons are excited to a special input. Of course the number of the excited neurons is n. Array W is the weight vector. The relationship between input and output is related with C and W. Fig.3 expresses the CMAC model very clearly. In Fig.3, n equals to 5.

The weight vector depends on the training result. The training algorithm is as the following:

$$W(k) = W(k-1) + \eta \times e / n \tag{1}$$

Here, e is the error between the output and the expected one and n is the generalization parameter. η is the learning rate, which should be in [0,1].

2.2 The CMAC controller and its training

To our problem, the control object is the robot joint angle. The CMAC controller tries to make the joint angle varying along with the expected one. In another word, the controller wants to make the error between the real output and the expected one be zero. For simplifying, we decide a single-input and single-output controller. The input is the angle error and the output is used to regulate the gas pressure. So the whole control system should be the one shown in Fig.4.

First, let's design the structure of the CMAC controller. In another word, the generalization parameter n and the quantization parameter q should be determined. We decide that n=5. The learning will be slow if n is large. The largest error is about 100 degree, because the maximal robot joint angle is about 100 degree. We decide that q equals to 5 degree. So the generalization area is nq (5x5=25 degree). It means that their outputs will be independent of each other if the difference of two angle errors is larger than 25 degree. Based on the two parameters, the CMAC controller can be decided.

Second, let's train the CMAC controller. The training should include off-line learning and on-line learning. Off-line learning should be finished before the CMAC controller works. On-line learning of course should be completed when it is working. Usually a neural network needs some experiment data to be trained. But here we use another method. We don't get any data from experiments. We lend the data directly from the expert experience data. Based on the fuzzy control theory, a general control rule expressed by data can be gotten from the expert experience, shown in Table 1 [3-4]. The controller inputs are error e and its differentiation de/dt and the controller output is u. All the data are expressed in a standardized field [-6,6]. To our problem, we have only one input – the angle error. So we only use the group of the data corresponding to the row that de/dt equals to 0. Based on the training algorithm (formula (1)) and the training data from Table 1, we do the off-line training.

3. Simulation of the control system based on pure CMAC

The control system is shown in Fig.4. When the system is working, probably the load varies, the outer disturbance may get in, and there is friction. The weight of CMAC needs to be regulated. So on-line learning is necessary. On-line learning is different from off-line learning, because the data for learning can't be gotten directly. So we must define an object function. Here we define it as

$$J_{on} = \frac{1}{2}(\theta_d - \theta)^2 = \frac{1}{2}e_\theta{}^2 \tag{2}$$

The expected output of the CMAC controller can be calculated by the gradient descent method. We calculate it as following:

$$\frac{\partial J_{on}}{\partial u} = \frac{\partial J_{on}}{\partial \theta} \times \frac{\partial \theta}{\partial u} = -e_\theta \times \frac{\partial \theta}{\partial u} \tag{3}$$

$$\frac{\partial \theta}{\partial u} \approx \frac{\theta(t) - \theta(t-1)}{u(t) - u(t-1) + \varepsilon} \tag{4}$$

$$u_d(t+1) \approx u(t) - \eta \times \frac{\partial J_{on}}{\partial u} \tag{5}$$

Here, u_d is just the expected output of the CMAC controller and ε is a very small positive number. Then the on-line learning can be done based on the formula (1).

Based on the control system shown in Fig.4 and the joint model (omitted), we suppose the sine input as

$$\theta_d = \sin(\pi/2t) \tag{6}$$

To simulate the disturbance, we add a special item u_i which's frequency is 5 times of θ_d. It is input into the system before the joint. The tracking time is 2 seconds. The result is shown in Fig.5. From the result, we can conclude that the system is stable.

4. Simulation based on CMAC and PID

After all, CMAC is a kind of neural networks. Sometime a neural network controller will fail, because that its training data only include a small part of the whole data. There have been many successful cases of PID control. So we try to combine CMAC and PID algorithms. We call the controller as the hybrid controller. The control system is shown in Fig.6. The CMAC controller and the PID controller are parallel. They work together to control the robot joint. The output of the PID controller is used to train the CMAC network. At last the CMAC controller will play the important role. But at the beginning the important role should be the PID controller. So we define the following object function for on-line learning:

$$J_{on} = \tfrac{1}{2}(u - u_c)^2 / n \tag{7}$$

$$\Delta W = -\eta \times \frac{\partial J_{on}}{\partial W} = \mu \times C \times (u - u_c)/n \tag{8}$$

$$W(k) = W(k-1) + \Delta W(k) \tag{9}$$

Based on the control system and the on-line algorithm, we do the simulation. We suppose the sine input as

$$\theta_d = \sin(\pi/2t) \tag{10}$$

To simulate the disturbance, we add a special item u_i which's frequency is 5 times of θ_d. It is input into the system before the joint, shown in Fig.6. The tracking time is 2 seconds. The three PID parameters are given. Here the PID parameters especially the parameter P needn't be precise. As long as the control system works stable, that is OK. The result is shown in Fig.7. From the result, we can conclude that the system is stable.

Comparing the two control systems, the later seems better, because there is less vibration when the disturbance getting in.

Conclusion

The control systems for the robot joint actuated by a pair of McKibben muscles based on CMAC are designed. The simulations show that the systems are stable. The hybrid controller based on both CMAC and PID is better than the one based on pure CMAC.

Acknowledgement

This study is supported by the project from Dep. of Science & Technology, Henan Province, China (the project No. 072102240022)

Reference

[1] J.S.Albus, A new approach to manipulator control: the cerebellar model articulation controller (CMAC), Journal of Dynamic System, Measurement, and Control, Transactions of ASME, Vol.7, pp.220~227, 1975.

[2] C.Sabourin and O.Bruneau, Robustness of the dynamic walk of a biped robot subjected to disturbing external forces by using CMAC neural networks, Robotics and Autonomous System, Vol.38, pp.81-99, 2005.

[3] N.Zhang and P.Yan, Neural network and fuzzy control, Tsinghua University publisher, Beijing, 1998.

[4] Z.Dou, The fuzzy logic control technology and its application. Aviation and aerospace publisher, Beijing, 1995.

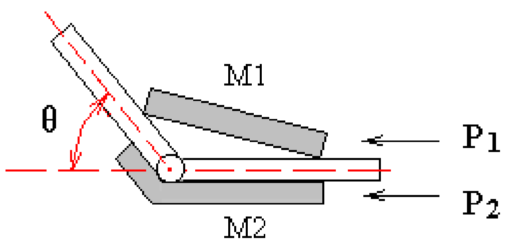

Fig.1 The robot joint actuated by two McKibben muscles

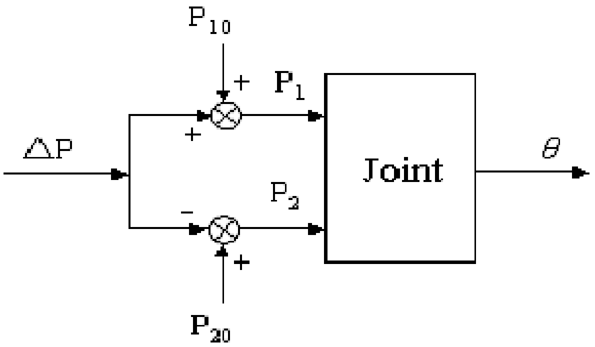

Fig.2 Single-input and single-output system

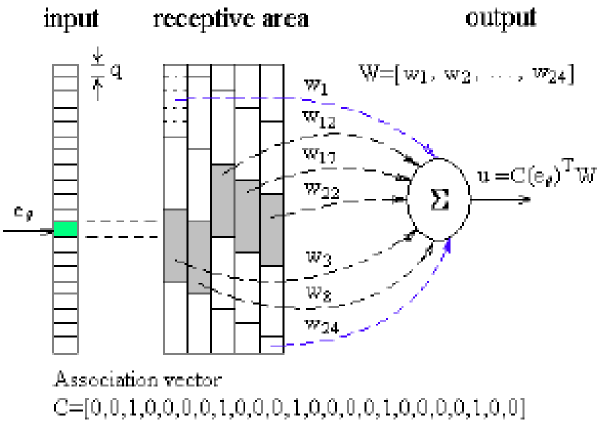

Fig.3 CMAC model

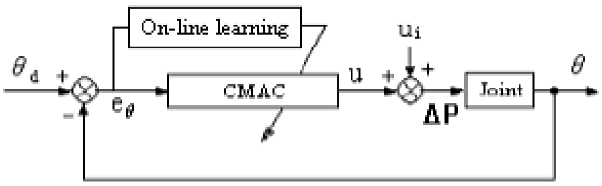

Fig.4 Control system based on pure CMAC

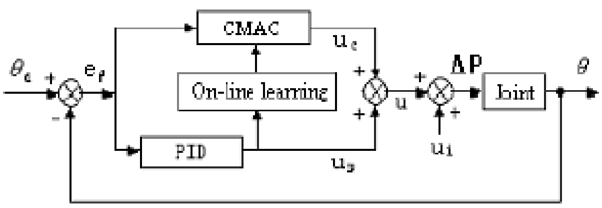

Fig.6 Control system based on CMAC and PID

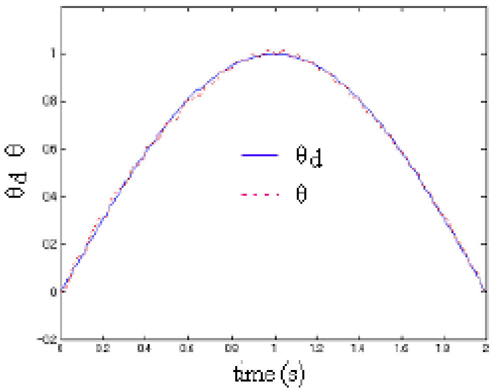

Fig.5 (a) Tracking 1

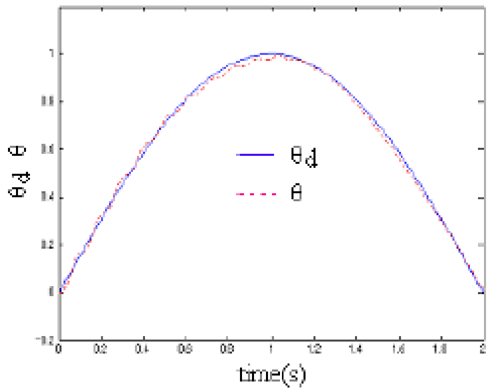

Fig.7 (a) Tracking 2

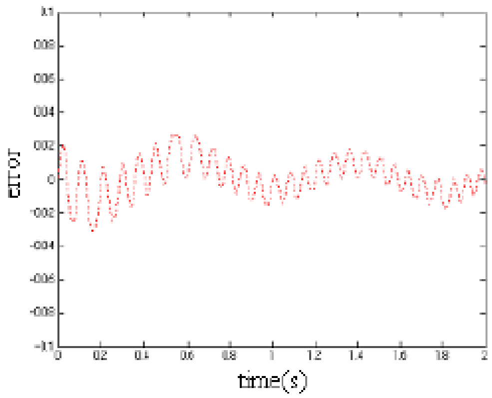

Fig.5 (b) Tracking 1 error

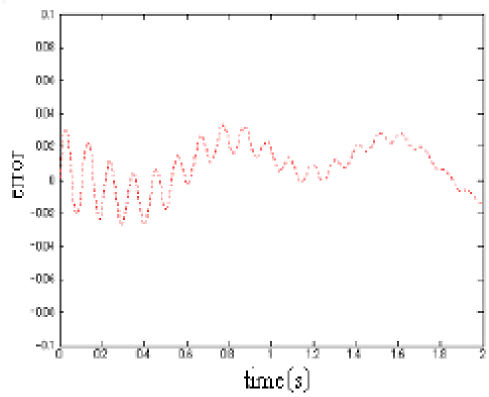

Fig.7 (b) Tracking 2 error

Table 1. Fuzzy control regulation table

e	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
u	5	5	5	4	4	3	0	-3	-4	-4	-5	-5	-5