LETTER

Trajectory-Tracking Control for Mobile Robot Using Recurrent Fuzzy Cerebellar Model Articulation Controller

Jinzhu Peng, Yaonan Wang and Wei Sun

College of Electrical and Information Engineering,
Hunan University, Changsha, 410082, China
E-mail: pengjinzhu@126.com, yaonan@mail.hunu.edu.cn, wei_sun@hnu.cn

(Submitted on October 15, 2005)

Abstract — A kind of recurrent fuzzy cerebellar model articulation controller (RFCMAC) model is presented. The recurrent network is embedded in the RFCMAC by adding feedback connections on the first layer to embed temporal relations in the network. A nonconstant differentiable Gaussian basis function is used to model the hypercube structure and the fuzzy weight. A gradient descent learning algorithm is used to adjust the parameters. Simulation experiments are made by applying proposed RFCMAC on mobile robots tracking control problem to confirm its effectiveness, and has better dynamic performance than FCMAC.

Keywords — Fuzzy Neural Network, CMAC, Recurrent Network, Mobile Robot, Trajectory-Tracking Control

1. Introduction

The trajectory-tracking control of mobile robot has been a topic of research during recent years. And lots of control methods have been applied, such as sliding mode control [1], backstepping technique [2], adaptive control [3], fuzzy control [4], and neural network control [5-8], etc. Among available options, the neural networks have great potential for its self-learning ability. The cerebellar model articulation controller (CMAC) as a kind of neural network is developed by Albus in 1975 [9], represents one kind of associative memory technique, which can be used in control learning, implementing a mapping or function approximation. Using this technique, at each sampling time, the inputs are quantized into discrete states. One advantage of CMAC technique, compared with multilayer neural networks is its excellent learning characteristics and fast computation speed. It is very useful to the self-adaptive control of complicated plant such as robot. The conventional CMAC can be viewed as a basis function network with supervised learning, and performs well in terms of its fast learning speed and local generalization capability for approximating nonlinear functions. Convergence of CMAC has been proved in different ways by several researchers [10-12]. However, there exists some problems when a conventional CMAC model is applied [13]. The conventional CMAC has an enormous memory requirement for resolving high-dimensional classification problems, and its performance heavily depends on the approach of input space quantization. For acquiring the derivative information of input and output variables, Chiang and Lin proposed a CMAC with a nonconstant differentiable Gaussian basis function, indicating that this CMAC can converge to the least square error [12]. Geng and Mcculloung [14] combined the preferred features of the fuzzy logic controller and the CMAC, and the structure is called fuzzy CMAC (FCMAC). The FCMAC is applied to an integrated guidance and control system design for a bank-to-turn missile. This makes CMAC a suitable candidate for online, real-time implementation of a missile control system that typically exhibits nonlinear behavior.

Neural network can be classified as feed forward neural network and recurrent neural network. Feed forward neural network can approximate a continuous function to an arbitrary degree of accuracy. However, feed forward neural network is a static mapping, which can not represent a dynamic mapping well. Although this problem can be solved by using tapped delays, but it requires a large number of neurons to represent dynamical responses in the time domain. On the other hand, recurrent neural networks [15-17] are able to represent dynamic mapping very well and store the internal information for updating weights later. Recurrent neural

network has an internal feedback loop. It captures the dynamical response of a system without external feedback through delays. Recurrent neural network is a dynamic mapping and demonstrates good performance in the presence of uncertainties, such as parameter variations, external disturbance, unmodeled and nonlinear dynamics. Recurrent fuzzy neural network (RFNN) [18, 19] is a modified version of recurrent neural network, which use recurrent network for realizing fuzzy inference. It is possible to train RFNN using the experience of human operators expressed in term of linguistic rules, and interpret the knowledge acquired from training data in linguistic form. And it is very easy to choose the structure of RFNN and determine the parameters of neurons from linguistic rules. Moreover, with its own internal feedback connections, RFNN can temporarily store dynamic information and cope with temporal problems efficiently.

In this paper, a recurrent fuzzy CMAC (RFCMAC) network structure is proposed, in which, the temporal relations are embedded by adding feedback connections on the first layer of FCMAC network. Back propagation algorithm is used to train the proposed RFCMAC. Finally, simulation experiments are made on mobile robot trajectory-tracking control problem to confirm its effectiveness.

The rest of this paper is organized as follows. In Section 2, the RFCMAC model is structured and the learning algorithms are derived. The dynamic model of mobile robot is given in Section 3. Section 4 presents some simulation results. Finally, conclusions are given in section 5.

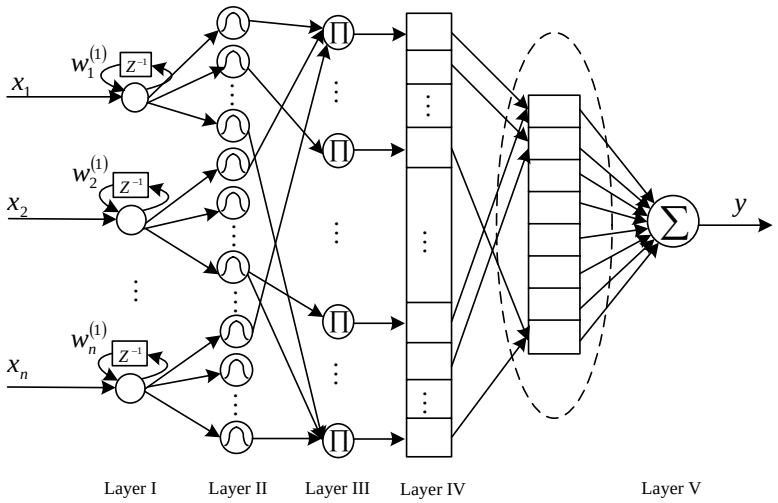


Figure 1. Structure of RFCMAC network

2. Structure of the RFCMAC Model

2.1 Recurrent Fuzzy CMAC Network

The RFCMAC structure is shown in Figure 1, which comprises the feedback layer, associate memory, hypercubes, weight memory and output layer, is adopted to implement the RFCMAC control in this study. The signal propagation and the basic function in each layer of the RFCMAC are introduced as follows.

(1) Layer I: *Feedback Layer*: This layer accepts input variables. Its nodes transmit input values to the next layer. Feedback connections are added in this layer to embed temporal relations in the network. For *i*th node in this layer, the input and output are represented as

$$Out_i^{(1)}(k) = In_i^{(1)}(k) = x_i(k) + w_i^{(1)} \cdot Out_i^{(1)}(k-1) \qquad i = 1,2,\ldots,n \tag{1}$$

where *k* denotes the number of iterations, $w_i^{(1)}$ are the recurrent weights for the units in the space.

(2) Layer II: *Membership Layer*: Nodes in this layer represent the terms of respective linguistic variables. Each node performs a Gaussian membership function.

$$\mu_{A_{ij}}\left(Out_i^{(1)}\right) = \exp\left\{-\frac{\left(Out_i^{(1)} - a_{ij}\right)^2}{\left(b_{ij}\right)^2}\right\}, \quad i = 1,2,\ldots,n;\ j = 1,2,\ldots,m \tag{2}$$

where a_{ij} and b_{ij} are the mean and the standard deviation of the Gaussian membership function, and represent the hypercube center and the hypercube variance, respectively. The subscript *ij* indicates the *j*th term of the *i*th input variable.

Suppose the *p*th neuron unit corresponds with the *j*th term of the *i*th input variable in this layer, then the input and output are represented as

$$Out_p^{(2)} = In_p^{(2)} = \mu_{A_{ij}}\left(Out_i^{(1)}\right),\ p = 1,2,\ldots,n \times m \tag{3}$$

(3) Layer III: *Rule Layer*: This layer forms the fuzzy rule base and realizes the fuzzy inference. Each node is corresponding to a fuzzy rule. Links before each node represent the preconditions of the corresponding rule, and the node output represents the "firing strength" of corresponding rule.

Then, the *qth* node of this layer performs the AND operation in the *q*th rule. It multiplies the input signals and output the product. Using

$$Out_q^{(3)} = In_q^{(3)} = \prod_i \mu_{A_{ij}}\left(Out_i^{(1)}\right), \quad i = 1,2,\ldots,n;\ \ q = 1,2,...,m^n \tag{4}$$

(4) Layer IV: *Associate Memory Layer*: Nodes in this layer performs the defuzzification operation. The input and output of this layer are represented as

$$Out_q^{(4)} = In_q^{(4)} = w_q^{(4)} \cdot Out_q^{(3)}, \quad q = 1,2,...,m^n \tag{5}$$

where $w_q^{(4)}$ is the weight, which represents the output action strength of the *s*th output associated with the *q*th rule.

(5) Layer V: *Output Layer*: Nodes in this layer performs the defuzzification operation. The input and output are represented as

$$y = Out^{(5)} = In^{(5)} = \sum_{q=1}^{m^n} Out_q^{(4)} \tag{6}$$

In addition, if the input space is very large, causes storage space which the association unit needs oversized, may join a hash mapping between Layer IV and Layer V. In this paper, only the RFCMAC without hash mapping is considered.

2.2 Learning Algorithm of RFCMAC

In the learning algorithm, by applying the back propagation algorithm, we can train the connection weights of the RFCMAC network. There are four adjustable parameters ($w_{sq}^{(4)}$, a_{ij}, b_{ij}, $w_i^{(1)}$) that need to be tuned. Our goal is to minimize the cost function *E*, defined as follows:

$$E = \frac{1}{2}\left(\hat{y}(t) - y(t)\right)^2 \tag{7}$$

where $\hat{y}(t)$ is the desired output at time *t* and $y(t)$ is the actual output at time *t*. Based on back propagation, the learning algorithm is described as follows.

The parameters of RFCMAC network are updated by the amount:

$$\frac{\partial E(k)}{\partial w_q^{(4)}(k)} = \frac{\partial E(k)}{\partial y(k)} \cdot \frac{\partial y(k)}{\partial w_q^{(4)}(k)} = -\left(\hat{y}(k) - y(k)\right) \cdot Out_q^{(3)}, \quad q = 1,2,...,m^n \tag{8}$$

$$\begin{aligned}\frac{\partial E(k)}{\partial a_{ij}(k)} &= \frac{\partial E(k)}{\partial y(k)} \cdot \frac{\partial y(k)}{\partial Out_q^{(3)}(k)} \cdot \frac{\partial Out_q^{(3)}(k)}{\partial Out_p^{(2)}(k)} \cdot \frac{\partial Out_p^{(2)}(k)}{\partial a_{ij}(k)} \\ &= -2\left(\hat{y}(k) - y(k)\right) \cdot \sum_{q=1}^{m^n} \left(Out_q^{(4)} \cdot \frac{Out_i^{(1)} - a_{ij}}{\left(b_{ij}\right)^2} \right), \quad i = 1,2,\ldots,n;\ j = 1,2,\ldots,m\end{aligned} \tag{9}$$

$$\frac{\partial E(k)}{\partial b_{ij}(k)} = \frac{\partial E(k)}{\partial y(k)} \cdot \frac{\partial y(k)}{\partial Out_q^{(3)}(k)} \cdot \frac{\partial Out_q^{(3)}(k)}{\partial Out_p^{(2)}(k)} \cdot \frac{\partial Out_p^{(2)}(k)}{\partial b_{ij}(k)}$$

$$= -2\left(\hat{y}(k)-y(k)\right)\cdot\sum_{q=1}^{m^n}\left(Out_q^{(4)}\cdot\frac{\left(Out_i^{(1)}-a_{ij}\right)^2}{\left(b_{ij}\right)^3}\right),\quad i=1,2,\ldots,n;\ j=1,2,\ldots,m \tag{10}$$

$$\frac{\partial E(k)}{\partial w_i^{(1)}(k)}=\frac{\partial E(k)}{\partial y(k)}\cdot\frac{\partial y(k)}{\partial Out_q^{(3)}(k)}\cdot\frac{\partial Out_q^{(3)}(k)}{\partial Out_p^{(2)}(k)}\cdot\frac{\partial Out_p^{(2)}(k)}{\partial Out_i^{(1)}(k)}\cdot\frac{\partial Out_i^{(1)}(k)}{\partial w_i^{(1)}(k)}$$

$$= 2\left(\hat{y}(k)-y(k)\right)\cdot\sum_{q=1}^{m^n}\sum_{j=1}^{m}\left(Out_q^{(4)}\cdot\frac{Out_i^{(1)}-a_{ij}}{\left(b_{ij}\right)^2}\cdot Out_i^{(1)}(k-1)\right),\quad i=1,2,\ldots,n \tag{11}$$

The weights and the receptive field functions are updated according to the following equations:

$$w_q^{(4)}(k+1)=w_q^{(4)}(k)-\eta_1\frac{\partial E(k)}{\partial w_q^{(4)}(k)} \tag{12}$$

$$a_{ij}(k+1)=a_{ij}(k)-\eta_2\frac{\partial E(k)}{\partial a_{ij}(k)} \tag{13}$$

$$b_{ij}(k+1)=b_{ij}(k)-\eta_3\frac{\partial E(k)}{\partial b_{ij}(k)} \tag{14}$$

$$w_i^{(1)}(k+1)=w_i^{(1)}(k)-\eta_4\frac{\partial E(k)}{\partial w_i^{(1)}(k)} \tag{15}$$

where $\eta_1,\eta_2,\eta_3,\eta_4$ are the learning rates of the weight of the output action strength, the mean ,the variance for the receptive field functions, and the weight of feedback unit, respectively.

3. Problem Formulation

3.1 Model of a Mobile Robot

The mobile robot considered here is shown in Figure 2. It consists of a vehicle with two driving wheels mounted on the same axis, and a front passive wheel for balance. The two driving wheels are controlled independently by motors.

Let the mobile robot be rigid moving on the plane. We assume that the absolute coordinate system *OXY* is fixed on the plane as shown in Figure 2. Then, the dynamic property of the robot is given by the following equation of motion [5]:

$$I_v\ddot{\phi}=D_r l-D_l l \tag{16}$$

$$M\dot{v}=D_r+D_l \tag{17}$$

For the right- and left-wheels, the dynamic property of the driving system becomes

$$I_\omega\ddot{\theta}_i+c\dot{\theta}_i=ku_i-rD_i\qquad(i=r,l) \tag{18}$$

where each parameter and variable are defined as follows:

I_v is the moment of inertia around the c.g. of robot; M is the mass of the robot; D_l and D_r are the left and right driving forces, respectively; l is the distance between left and right wheel; ϕ is the azimuth of the robot; v is the velocity of the robot; I_ω is the moment of inertia of wheel; c is the viscous friction factor; k is the driving gain factor; r is the radius of wheel; θ_i is the rotational angle of wheel; u_i is the driving input.

On the other hand, the geometrical relationships among variables ϕ, v, θ_i are given by

$$r\dot{\theta}_r=v+l\dot{\phi} \tag{19}$$

$$r\dot{\theta}_l=v-l\dot{\phi} \tag{20}$$

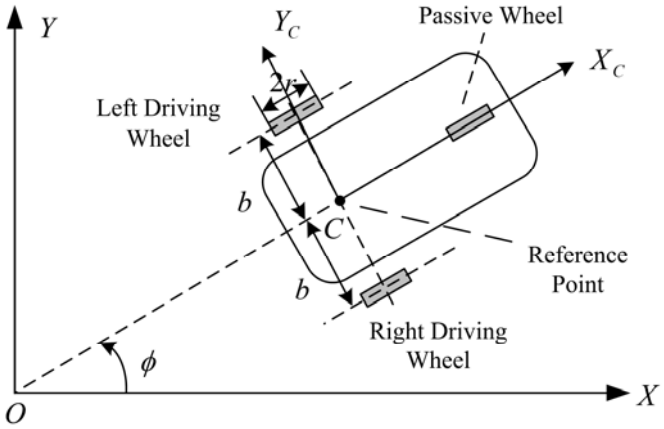


Figure 2. Model of a mobile robot

From these equations, defining the state variable for the robot as $x = [v, \phi, \dot{\phi}]^T$, the manipulated variable as $u = [u_r, u_l]^T$, and the output variable as $y = [v, \phi]^T$ yields the following state equation:

$$\dot{x} = Ax + Bu \tag{21}$$

$$y = Cx \tag{22}$$

where

$$A = \begin{bmatrix} a_1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & a_2 \end{bmatrix}, \quad B = \begin{bmatrix} b_1 & b_1 \\ 0 & 0 \\ b_2 & -b_2 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$a_1 = -\frac{2c}{Mr^2 + 2I_\omega}, \quad a_2 = -\frac{2cl^2}{I_v r^2 + 2I_\omega l^2}$$

$$b_1 = \frac{kr}{Mr^2 + 2I_\omega}, \quad b_2 = -\frac{krl}{I_v r^2 + 2I_\omega l^2}$$

3.2 Trajectory Tracking Problem

The velocity and azimuth of the robot are controlled by manipulating the torques for the left- and the right-wheels. That is, the control system considered here is of four-input and two-output. In the RFCMAC, The velocity error e_v and its rate, and the azimuth error e_ϕ and its rate are considered as the inputs, and the driving torque required for controlling the two wheels u_r and u_l is considered as the output. Here, the input deviations e_v and e_ϕ are defined by

$$e_v = v_d - v \tag{23}$$

$$e_\phi = \phi_d - \phi \tag{24}$$

where v_d, ϕ_d are the desire velocity and the desire azimuth, respectively. v, ϕ are the actual velocity and the actual azimuth of the robot, respectively.

According to above analysis, the control system of RFCMAC network for trajectory-tracking control of mobile robot is shown in Figure 3.

4. Simulation Experiments

Dynamics of mobile robotic are highly nonlinear and may contain uncertain elements. Many efforts have been made in developing control schemes to achieve the precise tracking control of mobile robot. In the simulation experiments of this paper, the proposed RFCMAC network is applied on mobile robot trajectory-tracking control to prove its effectiveness.

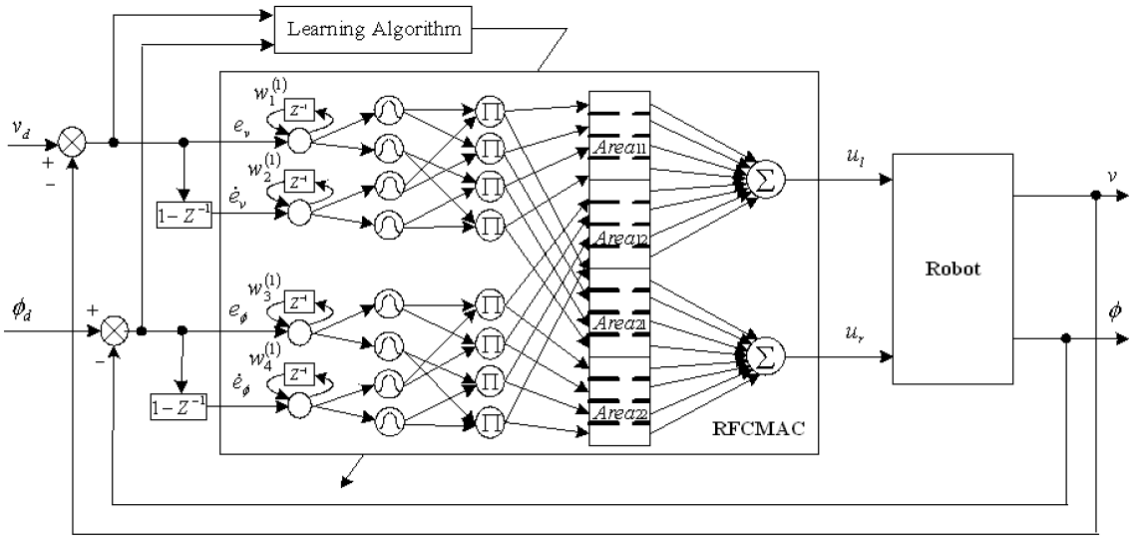


Figure 3. A RFCMAC network for trajectory-tracking control of mobile robot

The physical parameters of the mobile robot are as follows:

$$I_v = 10kg \cdot m^2, \quad M = 200kg, \quad l = 0.3m,$$

$$I_\omega = 0.005kg \cdot m^2, \quad c = 0.05kg/s, \quad r = 0.1m, \quad k = 5$$

In addition, the reference velocity v_d is given as $1m/s$ and the initial value of state variable is given as $x = [0.7322, 0.7493, 0]^T$. The initial value of velocity of the mobile robot is given as 0, and the initial value of the mobile robot is given as $x = [0,0,0]^T$.

A comparison analysis of the RFCAMC and the FCMAC are presented in this section.

4.1 Simulation Results using FCMAC

In this simulation, we apply the FCMAC to control the trajectory tracking of the mobile robot. That is, the initial values of the connection weights of feedback unit were all set to 0, and the learning rate $\eta_4 = 0$. and $\eta_1 = 0.01$, $\eta_2 = \eta_3 = 0.002$. Figure 4 shows the control results for the circular velocity and azimuth, Figure 5 shows the control torques of the two wheels, Figure 6 shows the *X*-*Y* circular trajectory-tracking curves, and the error of *X* and *Y* circular trajectory tracking are shown on Figure 7.

4.2 Simulation Results using RFCMAC

In this simulation, we apply the RFCMAC to control the trajectory tracking of the mobile robot. The learning rate $\eta_1 = 0.01$, $\eta_2 = \eta_3 = 0.002$, $\eta_4 = 0.001$. Figure 8 shows the control results for the circular velocity and azimuth, Figure 9 shows the control torques of the two wheels, Figure 10 shows the *X*-*Y* circular trajectory-tracking curves, and the error of *X* and *Y* circular trajectory tracking are shown on Figure 11.

5. Conclusions

In this paper, a RFCMAC network was proposed. The proposed model consists of four layers and the feedback connections are added in first layer, and uses a nonconstant differentiable Gaussian basis function to model the hypercube structure and the fuzzy weight. The proposed RFCMAC can be used for controlling the mobile robot. Finally, the simulation experiments were made on mobile robot trajectory-tracking control problem to confirm its effectiveness, and the results showed that the RFCMAC network has better control performance than FCMAC.

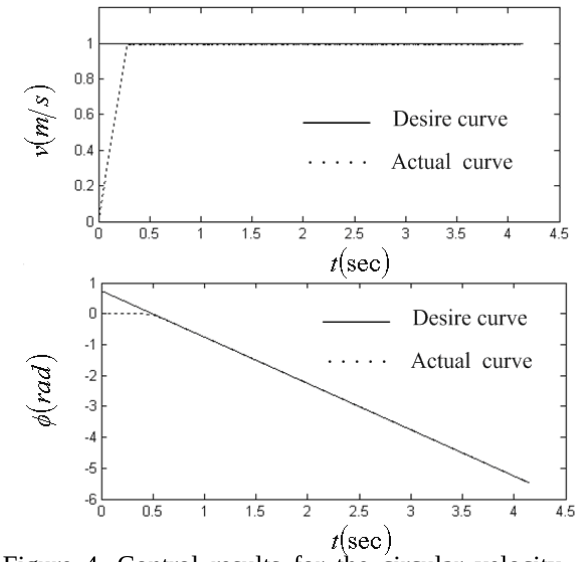


Figure 4. Control results for the circular velocity and azimuth using FCMAC

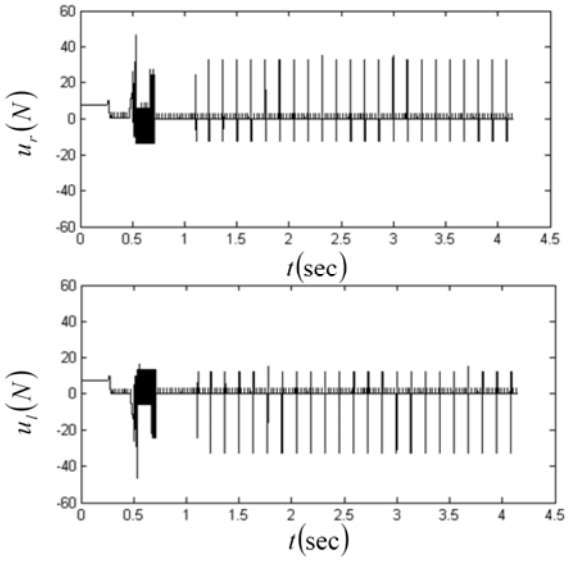


Figure 5. Control torques of the two wheels using FCMAC

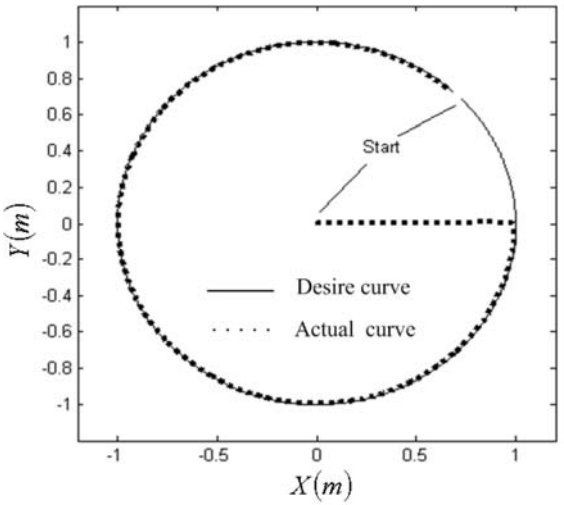


Figure 6. Control results for *X*-*Y* circular trajectory tracking using FCMAC

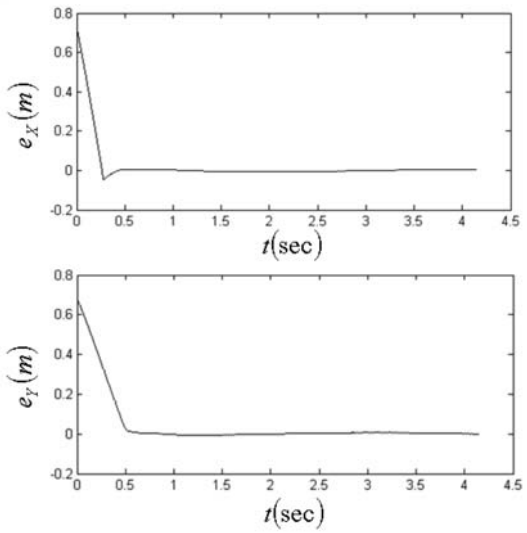


Figure 7. The error of *X* and *Y* circular trajectory tracking using FCMAC

References

[1] J. M. Yang and J. H. Kim. “Sliding Mode Control for Trajectory of Nonholonomic Wheeled Mobile Robots”. IEEE Trans. on Robotics and Automation, Vol. 15, No. 3, pp. 578-587, 1999.

[2] T. C. Lee, K. T. Song C. H. Lee and C. C. Teng. “Tracking Control of Mobile Robots Using Saturation Feedback Controller”. Proc. of IEEE International Conf. on Robotics & Automation. pp: 2639-2644,1999.

[3] F. Pourboghrat, M. P. Karlsson. “Adaptive control of dynamic mobile robots with nonholonomic constraints”. Computers and Electrical Engineering. Vol. 28, pp. 241-253, 2002.

[4] S. D. Wang and C. K. Lin. “Adaptive tuning of the fuzzy controller for robots”. Fuzzy Sets Systems, Vol. 110, No. 2, pp. 351-363, 2000.

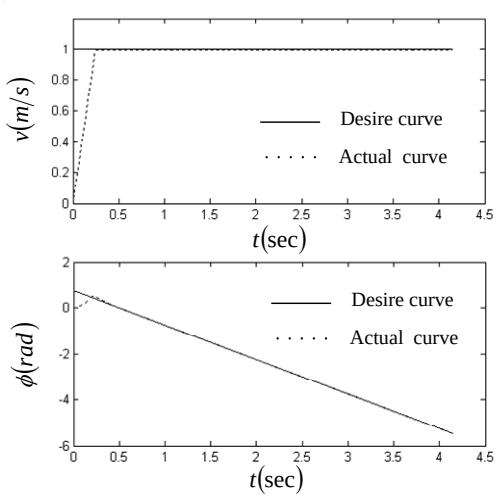


Figure 8. Control results for the circular velocity and azimuth using RFCMAC

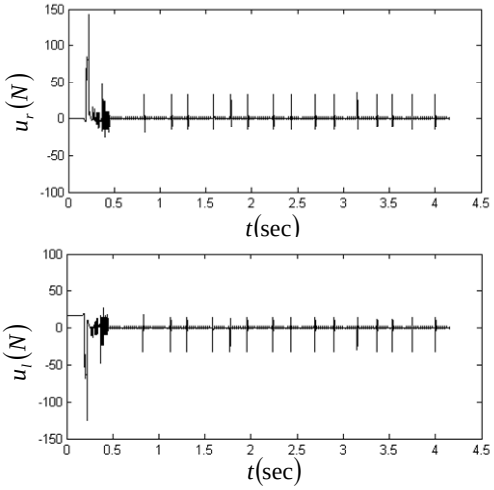


Figure 9. Control torques of the two wheels using RFCMAC

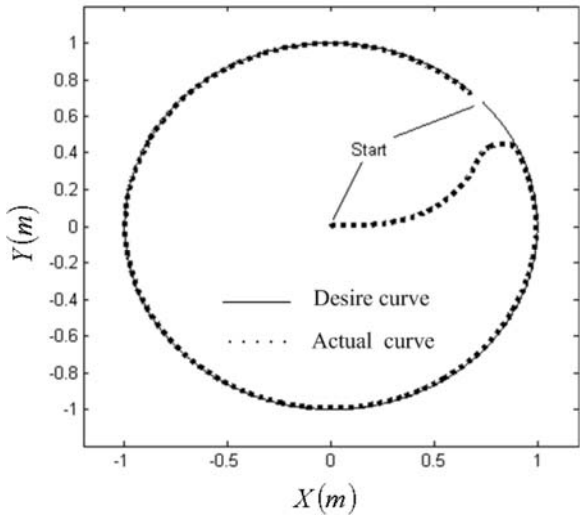


Figure 10. Control results for *X*-*Y* circular trajectory tracking using RFCMAC

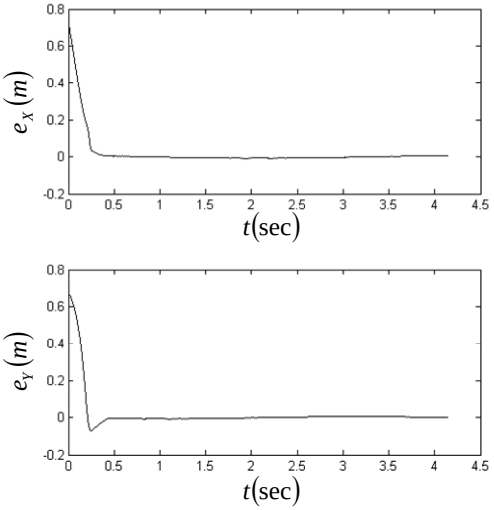


Figure 11. The error of *X* and *Y* circular trajectory tracking using RFCMAC

[5] K. Watanabe, J. Tang, M. Nakamura, S. Koga and T. Fukuda. “MOBILE ROBOT CONTROL USING FUZZY-GAUSSIAN NEURAL NETWORKS”. Proc. of IEEE/RSJ International Conf. on Robots and system, pp. 919-925, 1993.

[6] S. Lin and A. A. Goldenberg. “Neural-Network Control of Mobile Manipulators”. IEEE Trans. on Neural Network, Vol. 12, No. 5, pp. 1121-1133, 2001.

[7] R. Fierro and F. L. Lewis. “Control of a Nonholonomic Mobile Robot Using Neural Networks”. IEEE Trans. on Neural Network, Vol. 9, No. 4, pp. 589-600, 1998.

[8] S. Yildirim. “Adaptive robust neural controller for robots”. Robotics and Autonomous Systems, Vol. 46, No.1, pp. 175-184, 2004.

[9] J. S. Albus. “A new approach to manipulator control: The cerebellar model articulation controller (CMAC)”. Transactions of ASME: Journal of Dynamic Systems, Measurement, and Control, Vol. 97, No. 3, pp. 220-227, 1975.

[10] Y. Wong and A. Sideris. “Learning convergence in the cerebellar model articulation controller”. IEEE Trans

on Neural Networks, Vol. 3, No. 1, pp. 115-121, 1992.

[11] P. C. Parks amd J. Miller. “Convergence properties of associative memory storage for learning control system”. Automation and Remote Control, Vol. 50, No. 2, pp. 254-286, 1989.

[12] C. S. Lin and C. T. Chiang. “Learning convergence of CMAC technique”. IEEE Trans on Neural Network, Vol. 8, No. 6, pp. 1281-1292, 1997.

[13] T. W. Millerm F.H. Glanz, and L. G. Kraft, “An associative neural network alternative to backpropagation”. Proceedings of IEEE, Vol. 78, No. 10, pp. 1561-1567, 1990.

[14] Geng Z J, Mccullough C L. “Missile control using fuzzy cerebellar model arithmetic computer neural networks”. J. Guid, Control, Dynamic, Vol. 20, No. 3, pp. 557-565, 1997.

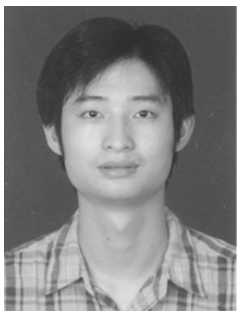
[15] M. A. Brdys and G. J. Kulawski. “Dynamic neural controllers for induction motor”. IEEE Trans. on Neural Networks, Vol. 10, No. 2, pp. 340-355, 1999.

[16] S. Ma and C. Ji. “Fast training of recurrent neural networks based on the EM algorithm”. IEEE Trans. on Neural networks, Vol. 9, No. 1, pp. 11-26, 1998.

[17] X. B. Liang and J. Wang. “A recurrent neural network for nonlinear optimization with a continuously differentiable objective function and bound constraints”. IEEE Trans. on Neural networks, Vol. 11, No. 5, pp. 1251-1262, 2000.

[18] S. M. Zhou and L. D. Xu. A new type of recurrent fuzzy neural network for modeling dynamic systems. Knowledge-Based Systems. Vol. 14, pp. 243-251, 2001.

[19] F. J. Lin, R. J. Wai, and C. M. Hong. “Hybrid Supervisory Control Using Recurrent Fuzzy Neural Network for Tracking Periodic Inputs”. IEEE Trans. on Neural networks, Vol. 12, No. 1, pp. 68-90, 2001.



Jinzhu Peng received his B.S and M.S degrees from the Department of Automation Engineering, Hunan University, P. R. China in 2002 and 2004, respectively. He now is a Ph. D. at the College of Electrical and Information Engineering, Hunan University. His research interests are neural networks and robotics.

Yaonan Wang received his M.S and Ph. D degrees from the Department of Automation Engineering, Hunan University, P.R.China in 1991 and 1994, respectively. From 1994 to 1995 he was a Postdoctoral Research Fellow in the National University of Defense Technology. Since 1995, he has been a professor in the Department of Electrical and Information Engineering at the University of Hunan, Changsha, P.R.China. He is also Director of National Engineering Research Center for computerized Numerical Control. His research interests are intelligent robot control, pattern recognition, computer integrated manufacturing (CIMS), intelligent information processing, industrial process control, computer networks system, Integrated circuit card and image recognition.

Wei Sun received his B.S, M.S, and Ph. D. degrees from the Department of Automation Engineering, Hunan University, P. R. China in 1997, 1999 and 2002, respectively. He now is working as an Associate Professor at the College of Electrical and Information Engineering, Hunan University. His areas of interests are neural networks, intelligent control and robotics.