Sensors & Transducers, Vol. 157, Issue 10, October 2013, pp. 42-50

Sensors & Transducers

http://www.sensorsportal.com

Design of Visual Feedback Tracking Algorithm for Nonholonomic Mobile Robots Based on Neural Network

[1] Huang Xuan, [2] Wenhua Zeng

[1] Cognitive Science Department, Xiamen University, Xiamen, China, 361005

[1] Fujian Key Laboratory of the Brain-like Intelligent Systems, Xiamen University, China, 361005

[1] Zhang Zhou institute of technology, Zhang Zhou, China, 363000

[2] School of Software, Xiamen University, Xiamen, China, 361005

Received: 16 August 2013 /Accepted: 25 September 2013 /Published: 31 October 2013

Abstract: With the rapid development of the computer and the electronic technique, the application of robots has been widen. The robot visual serving control system may mimic the human eyes. Then the vision information is used as a feedback to improve the ability of the robot adaption to the environment. However, traditional algorithms which need the calibration of visual parameters spend much time and become technical bottlenecks. This paper presents the development background of the robot and the concept of nonholonomic mobile robots with visual servoing feedback. Second, the deficiency exists in traditional algorithms and fuzzy controller. Third, BP neural network PID is proposed to design controller. Combining BP neutral network with PID controller is used to manipulate mobile robots firstly. The complex deduces of common tracking controllers is simplified and tracking control problem with non calibrated virtual parameters is solved. Finally, we program the simulation code. The simulation results show that the method is effective. *Copyright © 2013 IFSA.*

Keywords: Mobile Robot, Uncalibrated Visual Servoing, Stabilization, BP neural network PID.

1. Introduction

With the advancement of computer and camera equipment, nonholonomic mobile robot visual servo tracking has become very practical. Cao Yang and other people designed a robust speed tracking controller [1] according to the concept of servo in the study of trajectory tracking control of the global vision of the mobile robot. But it is in the case of known visual parameters, for the case of unknown parameters, it generally needs to be calibrated visual parameters. The traditional methods are DLT method [2], Tsai's method [3], Weng's iterative method [4]. However, the operation of these methods is not easy, feasibility of these methods is not high, so the research for the visual parameters calibration became the hot spot of this field. Zhangxue and other people Combined vision algorithms and Lyapunov and constructed an adaptive visual controllers, although the controller is the visual parameters calibration, the mathematical derivation of the design process is time-consuming, the structure of the controller is complex [5].

Nonholonomic mobile robot itself is non-linear, and it contains unknown parameters, it is difficult to use the traditional control methods to control it [6]. The fast parallel computing, strong adaptive and non-linear mapping of the neural network is applicable to

nonlinear systems tracking control of the nonholonomic mobile that is parameter uncertainty. Therefore, the choice of the neural network PID control can achieve a good tracking in the case of visual calibration parameters and can avoid the tedious formula derivation [7].

For such nonholonomic mobile robot, it's center of mass coincides with the geometric center. It's Nonholonomic constraints:

$$\dot{x}\sin\theta_i - \dot{y}\cos\theta_i = 0 \tag{1}$$

Based on (1), we can introduce the kinematic model of nonholonomic mobile robot in the image coordinate system:

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} v_i\alpha_1\cos(\theta_i - \theta_0) \\ v_i\alpha_2\sin(\theta_i - \theta_0) \\ w_i \end{bmatrix} \tag{2}$$

This model can be discretized into:

$$\begin{cases} x_i(k+1) = x_i(k) + v_i(k)\alpha_1 T\cos(\theta_i(k) - \theta_0) \\ y_i(k+1) = y_i(k) + v_i(k)\alpha_2 T\sin(\theta_i(k) - \theta_0) \\ \theta_i(k+1) = \theta_i(k) + w_i(k)T \end{cases} \tag{3}$$

The reference trajectory of the controlled system (3) is:

$$\begin{cases} x_r(k+1) = x_r(k) + v_r(k)\alpha_1 T\cos(\theta_r(k) - \theta_0) \\ y_r(k+1) = y_r(k) + v_r(k)\alpha_2 T\sin(\theta_r(k) - \theta_0) \\ \theta_r(k+1) = \theta_r(k) + w_r(k)T \end{cases} \tag{4}$$

In this formula, y_r, x_r, θ_r, respectively are the given ideal position and orientation of the nonholonomic mobile robot. v_r, w_r respectively are the given ideal linear and angular velocities of the mobile robot.

So the tracking error nonholonomic mobile robot at time k is:

$$\begin{cases} e_x(k) = x_r(k) - x_i(k) \\ e_y(k) = y_r(k) - y_i(k) \\ e_\theta(k) = \theta_r(k) - \theta_i(k) \end{cases} \tag{5}$$

2. The BP Neural Network

In 1986, Rumellhart put forward a error back-propagation of multilayer feedforward neural network, that is BP (Back Propagation) neural network, also known as multi-layer perceptron (multilayer Perceptrons) [8]. BP neural network uses gradient search, in order to minimize the error of artificial neural network actual output value and the desired output value, it is a learning process which spreads back while corrects the weights. BP algorithm as the typical network is the most mature in the artificial neural network [9].

2.1. Principle of BP Algorithm

The structure of BP network is shown in Fig. 1, it comprises input layers, intermediate layers and output layers. The intermediate layer is also known as a hidden layer, it may be a single layer or a multilayer.

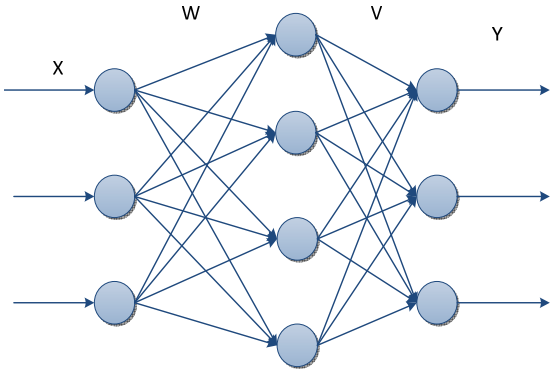

Fig. 1. Forward neural network.

Assumed that the number of layer of an artificial neural network is m, the sample of the input layer is X [10]. U_i^k is the sum of the inputs of the neuron i in the k layer, X_i^k is the output of the neuron i in the k layer.

$$X_i^k = f(U_i^k) \tag{6}$$

$$U_i^k = \sum_j W_{ij} X_j^{k-1}, \tag{7}$$

where W_{ij} is the weight coefficient of neurons i in the k–1 layer, $f(\cdot)$ is the activation function of the neurons.

The learning process of BP neural network is divided into forward propagation and back propagation.

Forward propagation

When it is forward propagation, the samples of network input pass through the layers of hidden layer from the input layer and then flow to the output layer. In this process, the state of each layer of neurons will only affect the state of the next layer neuron. Then, compare the output and the desired output in the output layer, if they are not identical, proceed to do the reverse propagation of next step [11].

Back-propagation

Back-propagation is a way to reverse transfer the error signal from the output layer to the hidden layer and then to the input layer, in the process of passing the weights of each neuron in the hidden layer will be fixed, and ultimately minimize the error [12].

2.2. Description of BP Algorithm Mathematical

The nature of BP algorithm is to use the negative gradient descent method to modify the weight value, so that the error function eventually reaches a minimum [13].

First define the error function *e*, to represent said the sum of squared differences between the actual output and the desired output.

$$e = \frac{1}{2}\sum_i (X_i^m - Y_i)^2 \tag{8}$$

Assume that the *m* layer in the BP network is the output layer, in the formula X_i^m represents the actual output value of the network, Y_i means the desired output value of the network.

$$\triangle W_{ij} = -\eta \frac{\partial e}{\partial W_{ij}} \tag{9}$$

where W_{ij} is the modification amount of the weight value coefficient W_{ij}, η is the step length, generally takes a value between 0-1.

$$\frac{\partial e}{\partial W_{ij}} = \frac{\partial e}{\partial U_i^k} \cdot \frac{\partial U_i^k}{\partial W_{ij}} \tag{10}$$

Due to the following equation:

$$\frac{\partial U_i^k}{\partial W_{ij}} = \frac{\partial (\sum_j W_{ij} X_j^{k-1})}{\partial W_{ij}} = X_j^{k-1} \tag{11}$$

We can get another formula as follow:

$$\frac{\partial e}{\partial W_{ij}} = \frac{\partial e}{\partial U_i^k} \cdot X_j^{k-1} \tag{12}$$

So we can get that:

$$\triangle W_{ij} = -\eta \frac{\partial e}{\partial W_{ij}} = -\eta \frac{\partial e}{\partial U_i^k} \cdot X_j^{k-1} \tag{13}$$

And we make:

$$d_i^k = \frac{\partial e}{\partial U_i^k} \tag{14}$$

So the modification amount is:

$$\triangle W_{ij} = -\eta d_i^k X_j^{k-1} \tag{15}$$

And the d_i^k in the above formula can be expressed as follow:

$$d_i^k = X_i^k (1 - X_i^k) \cdot \sum_j W_{ij} \cdot d_j^{k+1} \tag{16}$$

Summarized from the above mathematical relationship we can get that neural network BP algorithm first does the forward propagation, adds the input sample into the network, forward transfers through layers of network to get equation as follow:

$$X_i^k = f(U_i^k)$$

And then compare the actual output X_i^k and the desired output U_i^k, the error generated e modifies the value of weights through the back-propagation:

$$\triangle W_{ij} = -\eta d_i^k X_j^{k-1} \tag{17}$$

In order to speed up the convergence rate, we can add a weight coefficient to the modified formula.

$$\triangle W_{ij}(t+1) = -\eta d_i^k X_j^{k-1} + \partial \triangle W_{ij}(t) \tag{18}$$

In the formula, α is the weight coefficient correction constant.

Weights have been modified, and finally makes the error tends to be minimum. The more layers of the neural network, the more number of neurons, the greater the amount of computation is, the speed of convergence is slower.

2.3. The Implementation Steps of BP Neural Network

The BP algorithm flow chart is shown as Fig. 2:

3. Mobile Robot Tracking Controller Based on Neural Network

3.1. The Selection of the BP Neural Network PID in This Paper

PID incremental control algorithm:

$$\begin{aligned} u(k) &= u(k-1) + k_p[e(k) - e(k-1)] + k_i e(k) + k_d[e(k) - 2e(k-1) + e(k-2)] \\ &\triangleq F[u(k-1), k_p, k_i, k_d, e(k), e(k-1), e(k-2)] \end{aligned} \tag{19}$$

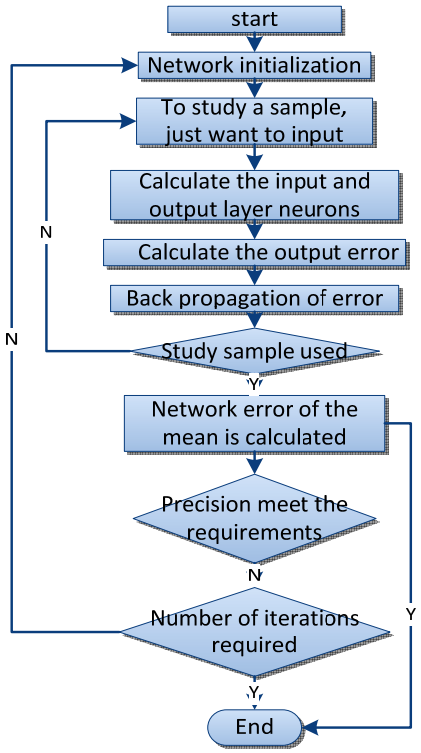

Fig. 2. Flowchart of BP algorithm.

$e(k)$ is the error between system's desired output and the actual output at time k, that is:

$$e(k)=y_r(k)-y(k) \tag{20}$$

The BP network selected in this paper is a three-layer structure for $M\times Q\times 3$, there are M nodes in the input layer of the network, Q nodes in the hidden layer, three nodes in the output layer (Fig. 3).

The input sample is:

$$o_j^{(1)}(k) = x_j, j = 1, 2, \cdots M \tag{21}$$

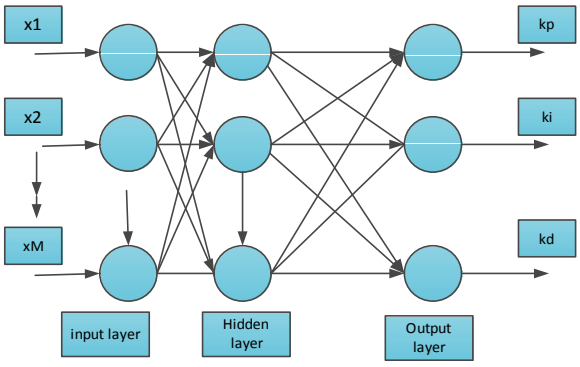

Fig. 3. Model of BP neural network.

The input-output relationship of hidden layer is:

$$\begin{cases} net_i^{(2)}(k)=\sum_{j=1}^{M} w_{ij}^{(2)} o_j^{(1)}(k)-\theta_i^{(2)}, i=1,2,\ldots Q \\ o_i^{(2)}(k)=f[net_i^{(2)}(k)] \end{cases} \tag{22}$$

$f(\bullet)$ in this formula takes Sigmoid function that is positive and negative symmetrical.

$$f(x)=\tanh(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}} \tag{23}$$

The input-output relationship of output layer is:

$$\begin{cases} net_l^{(3)}(k)=\sum_{i=1}^{Q} w_{li}^{(3)} o_i^{(2)}(k)-\theta_l^{(3)}, l=1,2,3 \\ o_l^{(3)}(k)=g[net_l^{(3)}(k)] \end{cases} \tag{24}$$

$g[\bullet]$ can be written as follow:

$$g[x]=\frac{1}{2}[1+\tanh(x)] \tag{25}$$

The output of the neural network is:

$$\begin{cases} o_1^{(3)}(k)=k_p \\ o_2^{(3)}(k)=k_i \\ o_3^{(3)}(k)=k_d \end{cases} \tag{26}$$

The function of the performance index is:

$$E(k)=\frac{1}{2}(y_r(k)-y(k))^2=\frac{1}{2}e^2(k) \tag{27}$$

The modified weight coefficient is:

$$\Delta w_{li}^{(3)}(k)=-\eta\frac{\partial E}{\partial w_{li}^{(3)}}+\alpha\Delta w_{li}^{(3)}(k-1) \tag{28}$$

And in the formula $\frac{\partial E}{\partial w_{li}^{(3)}}$ is:

$$\frac{\partial E}{\partial w_{li}^{(3)}}=\frac{\partial E(k)}{\partial y(k)}\frac{\partial y(k)}{\partial u(k)}\frac{\partial u(k)}{\partial o_l^{(3)}(k)}\frac{\partial o_l^{(3)}(k)}{\partial net_l^{(3)}(k)}\frac{\partial net_l^{(3)}(k)}{\partial w_{li}^{(3)}(k)} \tag{29}$$

But

$$\frac{\partial net_l^{(3)}(k)}{\partial w_{li}^{(3)}(k)}=o_i^{(2)}(k) \tag{30}$$

The conversion of $\frac{\partial y(k)}{\partial u(k)}$ quotes partial derivatives of formula put forward by Psaltis and some other people.

$$\frac{\partial y(k)}{\partial u(k)} \approx \frac{y(k)-y(k-1)}{u(k)-u(k-1)} \tag{31}$$

$\frac{\partial y(k)}{\partial u(k)}$ can approximately be replaced by a sign function $\mathrm{sgn}[\frac{\partial y(k)}{\partial u(k)}]$.

Learning algorithm of the coefficient of output layer in the neural network can be derived from the above derivation:

$$\Delta w_{li}^{(3)} = \alpha \Delta w_{li}^{(3)}(k-1) + \eta \delta_l^{(3)} o_i^{(2)}(k) \tag{32}$$

$$\delta_l^{(3)} = e(k)\mathrm{sgn}(\frac{\partial y(k)}{\partial u(k)})\frac{\partial u(k)}{\partial o_l^{(3)}(k)} g'(net_l^{(3)}(k)), (l=1,2,3) \tag{33}$$

Use the same token, learning algorithm of the coefficient of hidden layer in the neural network can be derived from the above derivation:

$$\Delta w_{ij}^{(2)} = \alpha \Delta w_{ij}^{(2)}(k-1) + \eta \delta_i^{(2)} o_j^{(1)}(k) \tag{34}$$

$$\delta_i^{(2)} = f'(net_i^{(2)}(k))\sum_{l=1}^{3} \delta_l^{(3)} w_{li}^{(3)}, (i=1,2,\cdots,Q) \tag{35}$$

In the formula,

$$g'(\bullet) = g(x)[1-g(x)] \tag{36}$$

$$f'(\bullet) = \left[1-f^2(x)\right] \tag{37}$$

We can obtain from equations (21)~(37) that:

$$\begin{aligned} \frac{\partial u(k)}{\partial o_1^{(3)}(k)} &= e(k)-e(k-1) \\ \frac{\partial u(k)}{\partial o_2^{(3)}(k)} &= e(k) \\ \frac{\partial u(k)}{\partial o_3^{(3)}(k)} &= e(k)-2e(k-1)+e(k-2) \end{aligned} \tag{38}$$

3.2. Mobile Robot Tracking Controller Based on Neural Network PID

Traditional controller of the mobile robot usually designs directly the linear and angular velocities of the nonholonomic mobile robot [14]. The nonlinear structure of the nonholonomic mobile robot causes the design process of the nonholonomic mobile robot cumbersome [15]. The method in this paper is deferent from the conventional method, it is based on the error between actual coordinates ($x_i(k)$, $y_i(k)$, $\theta_i(k)$) and the coordinates of the tracking targets ($x_r(k)$, $y_r(k)$, $\theta_r(k)$), obtains the amount of control ($u_x(k)$, $u_y(k)$, $u_\theta(k)$) of each step, through the PID controller of BP neural network. Model for position and orientation of the controlled objects is:

$$\begin{cases} x_i(k+1) = u_x(k) + x_i(k) \\ y_i(k+1) = u_y(k) + y_i(k) \\ \theta_i(k+1) = u_\theta(k) + \theta_i(k) \end{cases} \tag{39}$$

And then substitute ($u_x(k)$, $u_y(k)$, $u_\theta(k)$) into the kinematic model with unknown parameters, through recurrence, get the corresponding amount of control ($v_i(k)$, $w_i(k)$).

Choose three BP neural networks PID control system to respectively control the x, y, θ in order to prevent mutual interference.

The structure of PID tracking controller of BP neural network of the mobile robot is shown in Fig. 4:

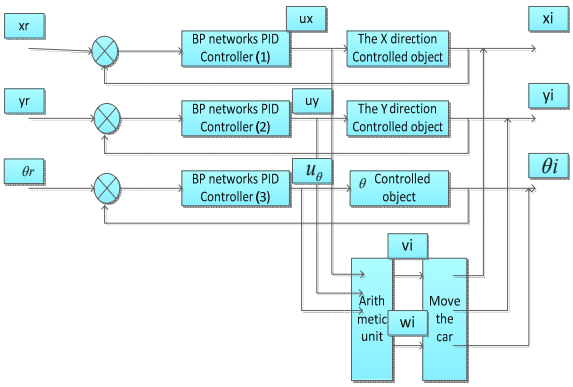

Fig. 4. PID controller of neural network.

The specific algorithm is as follows. As the formula (3) (39) show that:

$$\begin{cases} u_x(k) = v_i(k)\alpha_1 T\cos(\theta_i(k)-\theta_0) \\ u_y(k) = v_i(k)\alpha_2 T\sin(\theta_i(k)-\theta_0) \\ u_\theta(k) = w_i(k)T \end{cases} \tag{40}$$

Assumes that θ_0 is unknown, $\alpha_1 = \alpha_2 = \alpha$ and are unknown. It's physical meaning is that the magnification of the longitudinal direction and a transverse pixel of a CCD camera is the same.

What can be calculated by the formula (40) is:

$$\begin{cases} v_i(k)T\alpha = u_x(k)\cos(\arctan\frac{u_y(k)}{u_x(k)}) + u_y(k)\sin(\arctan\frac{u_y(k)}{u_x(k)}) \\ w_i(k) = \frac{u_\theta(k)}{T} \end{cases} \tag{41}$$

In the formula $\arctan\frac{u_y(k)}{u_x(k)}$ is denoted as $\varphi(k)$, and then:

$$v_i(k)T\alpha = u_x(k)\cos(\varphi(k)) + u_y(k)\sin(\varphi(k)) \quad (42)$$

$$v_i(k+1)T\alpha = u_x(k+1)\cos(\varphi(k+1)) + u_y(k+1)\sin(\varphi(k+1)) \quad (43)$$

The left and right ends of formula (42) and (43) are respectively divided, and then we can get that:

$$\frac{v_i(k+1)}{v_i(k)} = \frac{u_x(k+1)\cos(\varphi(k+1)) + u_y(k+1)\sin(\varphi(k+1))}{u_x(k)\cos(\varphi(k)) + u_y(k)\sin(\varphi(k))} \quad (44)$$

The right side of the equation of the formula (44) is known, and then the following equation can be recursive:

$$\frac{v_i(k+1)}{v_i(1)} = \frac{u_x(k+1)\cos(\varphi(k+1)) + u_y(k+1)\sin(\varphi(k+1))}{u_x(1)\cos(\varphi(1)) + u_y(1)\sin(\varphi(1))} \quad (45)$$

The given line speed of nonholonomic mobile robot given at the first time is $v_i(1)$, then according to equation (45) we can get that:

$$v_i(k+1) = \frac{u_x(k+1)\cos(\varphi(k+1)) + u_y(k+1)\sin(\varphi(k+1))}{u_x(1)\cos(\varphi(1)) + u_y(1)\sin(\varphi(1))} v_i(1) \quad (46)$$

The actual needs of linear velocities v_i and angular velocities w_i of every step of the nonholonomic mobile robot can be obtained as follow:

$$\begin{cases} v_i(k) = \dfrac{u_x(k)\cos(\varphi(k)) + u_y(k)\sin(\varphi(k))}{u_x(1)\cos(\varphi(1)) + u_y(1)\sin(\varphi(1))} v_i(1) \\ w_i(k) = \dfrac{u_\theta(k)}{T} \end{cases} \quad (47)$$

Thus in the case that the visual parameters of nonholonomic mobile robot are unknown, and at last the tracking control of the mobile car is achieved.

4. Results and Analysis

In the structure diagram, the PID controllers of the three BP neural network use the 4×5×3 BP network structure, and then according to the above algorithm to do the simulation. In the formula, v_r=1.5, w_r=2.3, $v_i(1)$=0.5.

§6.5.1 the parameter selection of BP neural network:

1. Assume that learning rate is unchanged and η=0.25, and then change the inertia coefficient α.

When η = 0.25, α = 0.1, the tracking error curve is shown as Fig. 5.

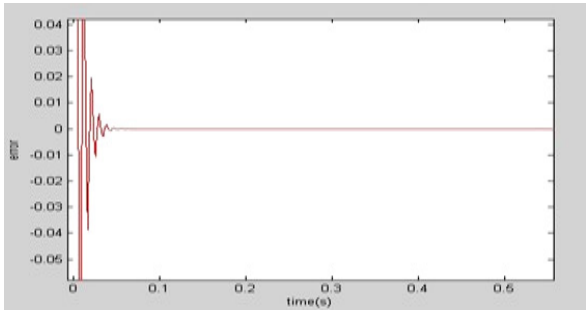

Fig. 5. Error in the condition that η = 0.25, α = 0.1.

When η = 0.25, α = 0.2, the tracking error curve is shown as Fig. 6.

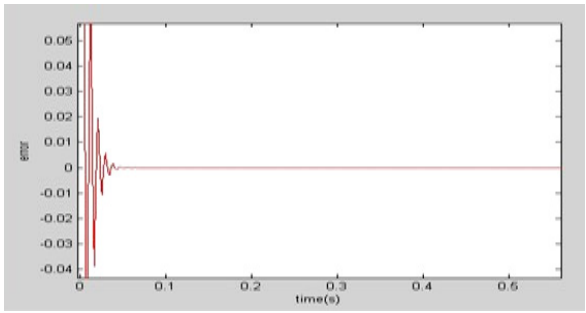

Fig. 6. Error in the condition that η = 0.25, α = 0.2.

When η = 0.25, α = 0.5, the tracking error curve is shown as Fig. 7.

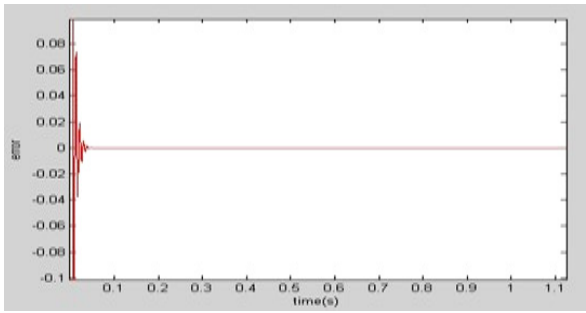

Fig. 7. Error in the condition that η = 0.25, α = 0.5.

When η = 0.25, α = 0.52, the tracking error curve is shown as Fig. 8.

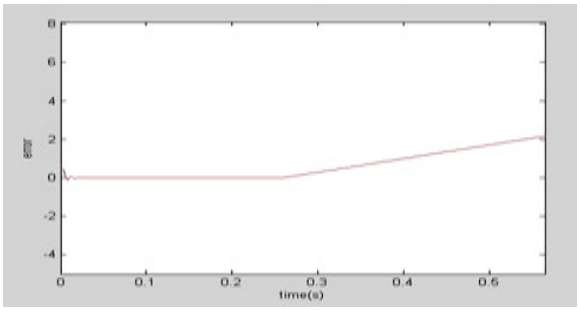

Fig. 8. Error in the condition that η = 0.25, α = 0.52.

When η = 0.25, α = 0.6, the tracking error curve is shown as Fig. 9.

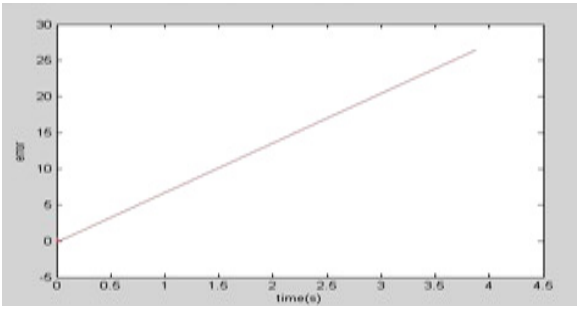

Fig. 9. Error in the condition that η = 0.25, α = 0.6.

According to the above simulation results, when the neural network is in the case that η is not changed, when make the value of α less than 0.5, the control effect is good, when the value of α exceeds 0.5 and the error increases as the value increases, and even there will be phenomenon out of control as shown in Fig. 8 and Fig. 9.

2. assume that the inertia coefficient is constant and α = 0.05, change the learning rate η.

When α=0.05, η=0.3, the tracking error curve is shown as Fig. 10.

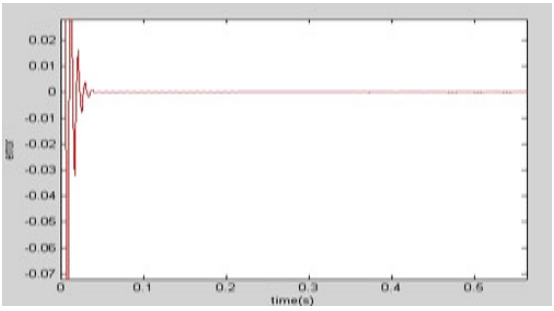

Fig. 10. Error in the condition that α=0.05, η=0.3.

When α=0.05, η=0.4, the tracking error curve is shown as Fig. 11.

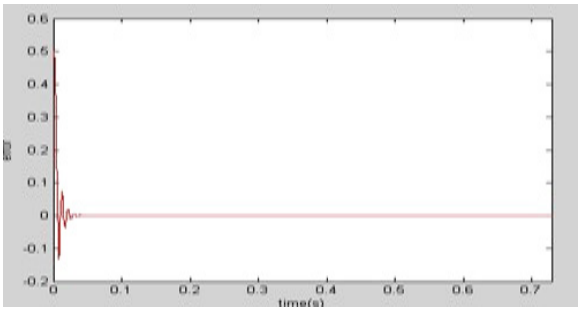

Fig. 11. Error in the condition that α=0.05, η=0.4.

When α=0.05, η=0.5, the tracking error curve is shown as Fig. 12.

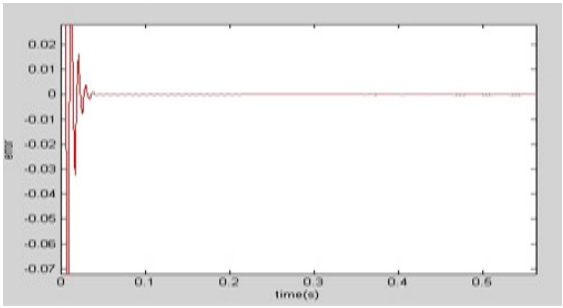

Fig. 12. Error in the condition that α=0.05, η=0.5.

When α=0.05, η=0.6, the tracking error curve is shown as Fig. 13.

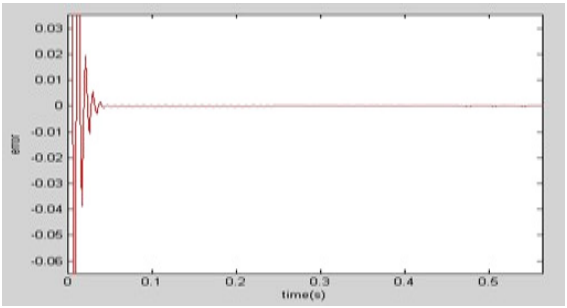

Fig. 13. Error in the condition that α=0.05, η=0.6.

According to the above simulation results, when the neural network is in the case that α is not changed, the value of η has little effect on the tracking effect.

Thus make the inertia coefficient A = 0.05, learning rate B = 0.25, and then this will allow the controller to meet the tracking requirements.

Assume that the start position and orientation of the target that the system want to track is (2,2,0), the actual start position and orientation of the moving car is (0,0,0).

The tracking results as shown as Fig. 14. Because the tracking target is moving randomly, the PID parameters are changing for adaptive adjustment, the

PID parameters are shown as Fig. 15, the tracking error is shown as Fig. 16.

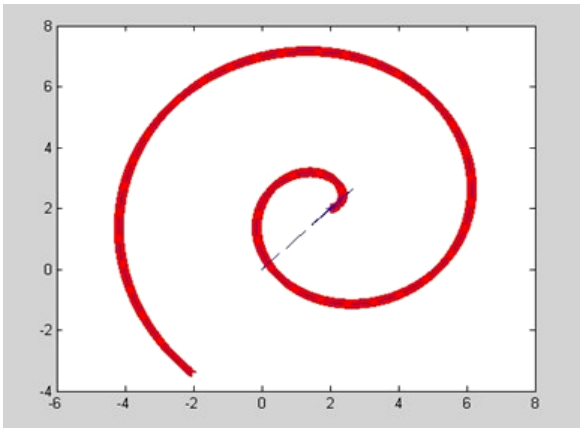

Fig. 14. Spiral curve tracking.

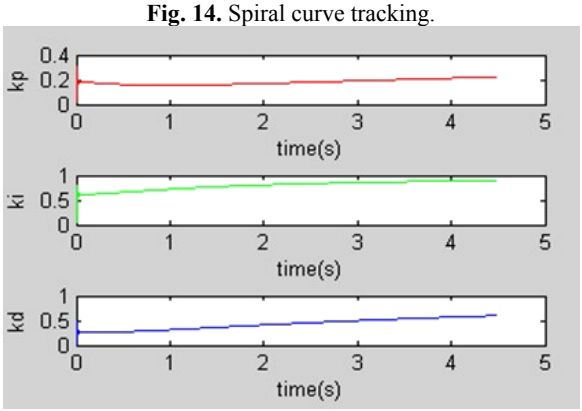

Fig. 15. The PID parameters in the Y direction.

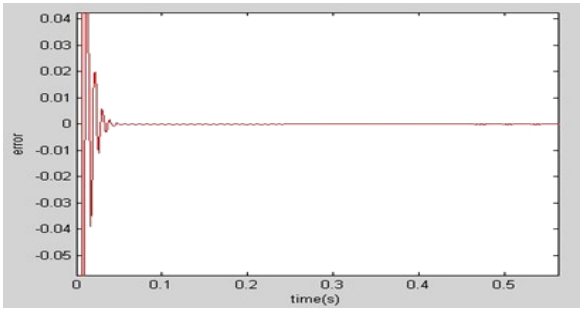

Fig. 16. Tracking error.

Assume that the starting position and orientation of the target of the nonholonomic mobile robot is (2,1,2.3), the actual starting position and orientation of the mobile car is (0,0,0), and the tracking results are shown as Fig. 17.

5. Conclusions

In this paper in connection with nonholonomic mobile robot model that the visual parameters are not calibrated, we put forward a method to design the tracking controller that uses the BP neural network PID method. Mobile robot that based on visual parameters are not calibrated, has been a research focus because of usefulness in the field of robotics. However, as far as is concerned trajectory tracking problem of the robot that visual parameters are not calibrated, there is a different shortcomings of existing methods.

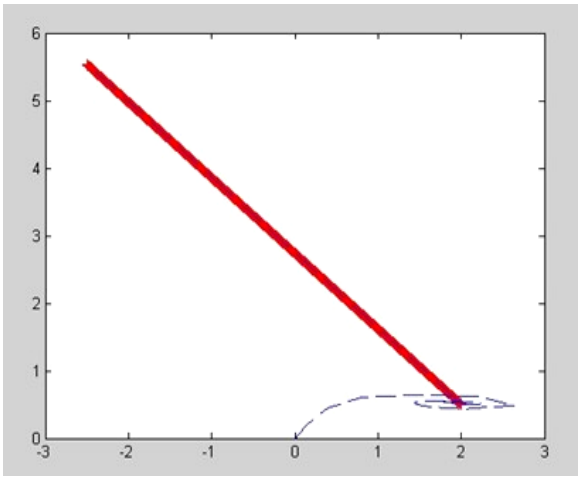

Fig. 17. Straight-line tracking.

This paper analyzes and compared the inadequacies of the existing methods, and put forward a way that is to apply BP neural network PID to the robot, and design a new tracking control algorithm, in order to achieve the tracking of nonholonomic mobile robot. In this paper we prove the validity of the BP neural network PID controller by using a simulation method. We put forward a dynamic tracking control of nonholonomic mobile robot that visual parameters are not calibrated, and results of the simulation show the effectiveness of this method. In connection with nonholonomic mobile robot model that the visual parameters are not calibrated, we put forward BP neural network PID control theory, design the tracking controller, and the results of simulation verify the tracking results.

References

[1]. P. Montesinos, L. Ceze, J. Torrellas, DeLorean: recording and deterministically replaying shared-memory multiprocessor execution efficiently, in *Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA'08)*, 2008, pp. 289-300.

[2]. J. Devietti, B. Lucia, L. Ceze, Luis Ceze, Mark Oskin, DMP: deterministic shared memory multiprocessing, in *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2009, pp. 85-96.

[3]. J. Yu, S. Narayanasamy, A case for an interleaving constrained shared-memory multi-processor, in *Proceedings of the 36th Annual International*

Symposium on Computer Architecture (ISCA'09), 2009, pp. 325-336.
[4]. L. Ceze, J. Tuck, J. Torrellas, Calin Cascaval, Bulk disambiguation of speculative threads in multiprocessors, *ACM SIGARCH Computer Architecture*, Vol. 34, No. 2, 2006, pp. 227-238.
[5]. Leslie Lamppost, Time, clocks and the ordering of events in a distributed system, *Communications of the ACM*, Vol. 21, No. 7, 1978, pp. 558-565.
[6]. Zhenying Liang, Chaoli Wang, Robust exponential stabilization of nonholonomic wheeled mobile robots with unknown visual parameters, *Journal of Control Theory and Applications*, Vol. 11, No. 8, 2011, pp. 58-65.
[7]. M. A. Yongmei, Guanghong Yang, Stability analysis for linear discrete-time systems subject to actuator saturation, *Journal of Control Theory and Applications*, Vol. 02, No. 4, 2010, pp. 113-116.
[8]. Jinkun Liu, Yu Lu, Adaptive RBF neural network control of robot with actuator nonlinearities, *Journal of Control Theory and Applications*, Vol. 1, No. 9, 2011, pp. 66-70.
[9]. Ruiquan Lin, Silian Chen, Xuwei Ding, Nonfragile guaranteed cost control for Delta operator-formulated uncertain time-delay systems, *Journal of Control Theory and Applications*, Vol. 3, No. 4, 2011, pp. 48-55.
[10]. Qing Shen, Weihua Gui, Ying Xiong, Chunhua Yang, Predictive control and scheduling codesign in network control systems, *Journal of Control Theory and Applications*, Vol. 5, No. 7, 2011, pp. 158-165.
[11]. Xiaoyu Zhang, Xiangna Li, Robust integral sliding mode control for uncertain systems with multiple time delays, *Journal of Control Theory and Applications*, Vol. 10, No. 5, 2011, pp. 1455-1459.
[12]. Yi Liu, Jun Zhao, Nonfragile control for a class of uncertain switching fuzzy time-delay systems, *Journal of Control Theory and Applications*, Vol. 21, No. 18, 2011, pp. 222-225.
[13]. Kok Kiong Tan, Andi Sudjana Putra, Tong Heng Lee, Compensation of hysteresis in piezoelectric actuator with iterative learning control, *Journal of Control Theory and Applications*, Vol. 12, No. 11, 2011, pp. 143-145.
[14]. Jinbo Wu, Guohua Xu, Zhouping Yin, Robust adaptive control for a nonholonomic mobile robot with unknown parameters, *Journal of Control Theory and Applications*, Vol. 12, No. 13, 2011, pp. 358-365.
[15]. Yaonan Wang, Jinzhu Peng, Wei Sun, Hongshan Yu, Hui Zhang, Robust adaptive tracking control of robotic systems with uncertainties, *Journal of Control Theory and Applications*, Vol. 1, No. 8, 2011, pp. 218-225.