Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

DECOUPLING NEURAL SLIDING MODE CONTROL FOR MULTI-LINK ROBOTS

Mu Xiaojiang, Chen Yangzhou

Beijing University of Technology, Chaoyang district, Beijing, China, 100022
China (Tel: 86-10-67396189; emails: muxiaojiang@emails.bjut.edu.cn).

Abstract: A decoupling neural sliding mode controller is given for trajectory tracking control of multi-link robots with external disturbances and uncertain system parameter errors. Different from general combining the sliding mode control (SMC) and neural network control (NNC), **this approach decouples a robot with n links into n subsystems and the state response of each subsystem can be governed by a corresponding sliding mode manifold. The whole system is controlled by a whole sliding mode manifold selected to be a global fast terminal sliding mode manifold, which guarantees that the controlled system can reach the sliding mode manifold and equilibrium point in finite time. A radius basis function (RBF) neural network is applied to learn the limit of system parameter errors and external disturbances for every subsystem, and enforce sliding mode motion by minimizing the cost function that is with respect to the distance from the sliding mode manifold. The switching gain of sliding mode control can be automatically adjusted according to system parameter errors and external disturbances by RBF neural network's learning. The controller's chattering is reduced without sacrificing robustness, even eliminated after RBF neural network's learning for a short time. Moreover, the stability of the controlled system is proven and the convergence of tracking error is also proven by Lyapnov function.** Simulation results verify the performance of the control scheme.

1. INTRODUCTION

Sliding mode control (SMC) is an effective approach to trajectory tracking control for multi-link robots with external disturbances and uncertain system parameter errors, since it has strong robustness in Lin C. K., [2006], Feng Y. [2002], and Yu S. H. [2000]. A global fast terminal sliding mode controller (GFTSMC) based on general mode control is proposed by Yu S. H [2000], which presented a new terminal sliding mode manifold that could guaranteed the controlled system reach the sliding mode manifold and equilibrium point from any initial state in finite time, but it still had followed disadvantages because of using the same principle as general sliding mode control: (1) The controller still has chattering. Though quasi sliding mode control can reduce chattering, it also sacrifices the robustness. (2) The boundary of external disturbances and system parameter errors must be evaluated in advance, while it is usually very difficult to be implemented in actuated control system.

Merging sliding mode control with neural networks (NN) appeared to be a good idea and many researchers have published various control scheme base on this idea. A few main ideas seem to be prevailing. The first one attempts to apply NN as an observer in the estimation of equivalent control by H. Morioka [1995] and in some cases disturbances by K. Jezernik [1997]. Also, a sliding mode controller with a modified switching function that produces a low-chattering control is used in parallel with an artificial NN for online identification of the modeling error by D. Munoz [2000], which imposes the controller performance. A novel approach that combines SMC and NN is presented by H. Hu [2006], the weights of which are determined by a fuzzy supervisory controller.

In this paper, combining NN control and GFTSMC, a decoupling neural sliding mode controller (DNSMC) by global fast terminal sliding mode manifold for multi-link robots is proposed, which decouples a robot with n links into n subsystems and the state response of each subsystem can be governed by a corresponding sliding mode manifold. The whole system is controlled by a whole sliding mode manifold. A radius basis function (RBF) neural network is applied to learn the limit of system parameter errors and external disturbances for every subsystem, and enforce sliding mode motion by minimizing the cost function that is with respect to the distance from the sliding mode manifold. The switching gain of sliding mode control can be automatically adjusted according to system parameter errors and external disturbances by RBF neural network's learning. The controller's chattering is reduced without sacrificing robustness, even eliminate after RBF neural network's learning for short time.

2. PROBLEM DESCRIPTION

2.1 Dynamics of Multi-link Robot

The dynamics of a rigid robot with n rotating links can be described by a second order nonlinear differential equation in Feng Y [2002]:

$$M_0(q)\ddot{q} + C_0(q,\dot{q})\dot{q} + G_0(q) = \tau + \rho(t) \tag{2.1}$$

where

$$\rho(t) = d(t) - \Delta M(q) - \Delta C(q,\dot{q})\dot{q} - \Delta G(q) \tag{2.2}$$

where $q,\ \dot{q},\ \ddot{q} \in R^n$ are angular position vectors, angular speed vectors and angular acceleration vectors of rotating joints respectively, but only angular position vectors q are measurable. $M_0(q)$, $C_0(q,\dot{q})$, $G_0(q)$ is nominal parameters of the robot. $M(q) \in R^{n\times n}$ is symmetric positive definite inertia matrix of the robot. $C(q,\dot{q}) \in R^n$ is a vector containing Coriolis and centrifugal forces. $G(q) \in R^n$ is gravitational torque. $\Delta M(q)$, $\Delta C(q,\dot{q})$, $\Delta G(q)$ are system parameter errors. $d(t)$ is external disturbance torque. $\rho(t) \in R^n$ is sum of system parameter error torque and disturbance torque. $\tau \in R^n$ is a vector of applied joint torques that are actually the control inputs.

We make the following assumption about $\rho(t)$:

$$\|\rho(t)\| < b_0 + b_1\|q\| + b_2\|\dot{q}\| \tag{2.3}$$

where $b_0,\ b_1,\ b_2$ are positive constants respectively.

There are following property for a multi-link robot described by (2.1) in general[2]:

Property 1: $M(q)$ is a positive symmetric definite matrix, and its inverse matrix $M^{-1}(q)$ is valid.

Property 2: $\dot{M}(q) - 2C(q,\dot{q})$ is a skew-symmetric matrix, i.e. it meets following equation for any vectors $x\in R^n$:

$$x^T[\dot{M}(q) - 2C(q,\dot{q})]x = 0 \tag{2.4}$$

2.2 GFTSMC for Multi-link Robot

The sliding mode manifold of global fast terminal sliding mode control can be written by following equation for a robot with n links[3] :

$$s = \dot{e} + \alpha e + \beta e^{q/p} \tag{2.5}$$

where

$$e^{q/p} = [e_1^{q/p}, e_2^{q/p}, \cdots, e_n^{q/p}]^T \tag{2.6}$$

$\alpha = \mathrm{diag}[\alpha_1, \alpha_2, \cdots \alpha_n]$, $\beta = \mathrm{diag}[\beta_1, \beta_2, \cdots \beta_n]$ are constants of the sliding mode manifold. $q < p < 2q$ are positive odd numbers. $e = q - q_r$ is tracking error of the controlled system, $\dot{e} = \dot{q} - \dot{q}_r$ is differential of tracking error.

Lemma in Yu S. H [2000]: For the rigid n links robot which can be described by (2.1), if the GFTSMC manifold is chosen as (2.5), and the GFTSMC is designed as follows, then the controlled system tracking error will converge to zero in finite time.

$$\tau = u_0 + u_1 + u_2 \tag{2.7}$$

where

$$u_0 = M_0(q)\ddot{q}_r + C_0(q,\dot{q})\dot{q} + G_0(q) \tag{2.8}$$

$$u_1 = -M_0(q)\alpha\dot{e} - C_0(q,\dot{q})s - \frac{q}{p}M_0(q)\beta \cdot \mathrm{diag}(e^{q/p-1})\dot{e} \tag{2.9}$$

$$u_2 = -M_0(q)K\left\|M_0^{-1}(q)\right\|\frac{s}{\|s\|} \tag{2.10}$$

$$K = \eta + b_0 + b_1\|q\| + b_2\|\dot{q}\| \tag{2.11}$$

$\eta > 0$ is a positive definite constant.

This control approach requires that the boundary of the external disturbances and system parameter errors must be estimated in advance. In order to guarantee the stability with an unknown uncertainty boundary and meet the sliding mode control condition:

$$s^T\dot{s} < 0 \tag{2.12}$$

K must adequately compensate the external disturbances and system parameter errors. So K is usually selected to be a conservative large constant, while this enhances the chattering and degrades the performances of SMC.

To alleviate or eliminate chattering, merging the good features of GFTSMC and neural networks, a decoupling neural sliding mode controller by global fast terminal sliding mode manifold is designed in this paper.

3. DECOUPING NEURAL SLIDING MODE CONROL

A Lyapunov function candidate can be selected as

$$V = \frac{1}{2}s^T s + \frac{1}{2}\dot{s}^T\dot{s} \tag{3.1}$$

In order to control the multi-link robot to be stable, the controller output must satisfy:

$$\dot{V} = s^T\dot{s} + \dot{s}^T\ddot{s} < 0 \tag{3.2}$$

For a robot with n links, (3.2) can be written as

$$\dot{V} = \sum_{i=1}^{n}(s_i\dot{s}_i + \dot{s}_i\ddot{s}_i) = \sum_{i=1}^{n}\dot{V}_i < 0 \tag{3.3}$$

If $\dot{V}_i<0,\ \ i=1, 2, \cdots, n$, then the controlled system is stable. Considering the interaction among links as external disturbances, the coupling robots with n links can be decoupled into n subsystem. The state response of each subsystem can be governed by a corresponding SMC condition: $\dot{V}_i<0$. And the whole system is controlled by a whole SMC condition.

The structure of a decoupling neural sliding mode controller is shown Fig. 1. There is a neural network for each link of the robot. The neural network can enforce sliding mode motion by minimizing its cost function that is with respect to the distance from the sliding mode manifold, and can adaptively adjust its output to compensate external disturbances and system parameter errors by its learning. The structure of a RBF neural network includes only two inputs and one output. So the neural network has simple structure and fast computation speed. The controller does not need offline training, and has strong robustness to external disturbances and system parameter errors. All neural networks in Fig.1 are radius basis function (RBF) neural networks, and have the same structure.

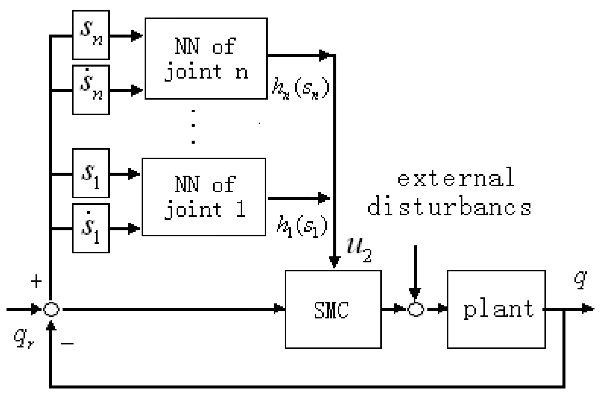

Fig. 1 The DNSMC structure

3.1 RBF Neural Networks

Suppose that RBF neural network input of joint i is $x_i = s_i$; output is $h_i(x)$. Then

$$h_i(x) = h_i(x_i \mid \omega_i) = \omega_i^T \phi_i(x) \tag{3.4}$$

where $\omega_i = [\omega_{i1}, \omega_{i2}, \cdots \omega_{im}]^T$ are weights of RBF neural network. $\phi_i(x) = [\phi_{i1}(x), \phi_{i2}(x), \cdots \phi_{im}(x)]^T$, $\phi_{ij}(x)$ is Gausses function, i.e.:

$$\phi_{ij}(x) = \exp(-\frac{\|x - c_{ij}\|^2}{\sigma_{ij}^2}) \tag{3.5}$$

where m is number of hide layer neural cell. c_{ij} is center of RBF. σ_{ij} is width of RBF.

RBF neural networks can approximates any real continuous function on a compact set to arbitrary accuracy. So the controller can reach the control cost function to arbitrary accuracy by neural network's learning the external disturbances and system parameter errors.

3.2 DNSMC

Theorem 1: For the rigid n links robot which can be described by (2.1), if the DNSMC manifold is selected as (2.5), and the DNSMC control is designed as follows, then the controlled system tracking error will converge to zero in finite time.

$$\tau = u_0 + u_1 + u_2 \tag{3.6}$$

where

$$u_0 = M_0(q)\ddot{q}_r + C_0(q, \dot{q})\dot{q} + G_0(q) \tag{3.7}$$

$$u_1 = -M_0(q)\alpha\dot{e} - \frac{q}{p} M_0(q)\beta e^{q/p-1}\dot{e} - C_0(q, \dot{q})s \tag{3.8}$$

$$\begin{aligned} u_2 &= [u_{21}(s_1), u_{22}(s_2), \cdots, u_{2n}(s_n)]^T \\ &= -[h_1(s_1), h_2(s_2), \cdots, h_n(s_n)]^T \end{aligned} \tag{3.9}$$

$h_i(s_i)$ is RBF neural network output of the i th joint.

The cost function of the i th joint neural network is as follow:

$$E_i = \frac{1}{2} s_i^2 + \frac{1}{2} \dot{s}_i^2 \to 0 \tag{3.10}$$

According to gradient descent algorithm, the weights differential of RBF neural network is computed as follows:

$$\dot{\omega}_{ij} = -\lambda \frac{\partial E_i}{\partial \omega_{ij}} = -\lambda \frac{\partial E_i}{\partial u_{2i}} \frac{\partial u_{2i}}{\partial \omega_{ij}} = -\lambda \dot{s}_i \frac{\partial \dot{s}_i}{\partial u_{2i}} \frac{\partial u_{2i}}{\partial \omega_{ij}} \tag{3.11}$$

$$\dot{c}_{ij} = -\lambda \frac{\partial E_i}{\partial c_{ij}} = -\lambda \frac{\partial E_i}{\partial u_{2i}} \frac{\partial u_{2i}}{\partial c_{ij}} = -\lambda \dot{s}_i \frac{\partial \dot{s}_i}{\partial u_{2i}} \frac{\partial u_{2i}}{\partial c_{ij}} \tag{3.12}$$

$$\dot{\sigma}_{ij} = -\lambda \frac{\partial E_i}{\partial \sigma_{ij}} = -\lambda \frac{\partial E_i}{\partial u_{2i}} \frac{\partial u_{2i}}{\partial \sigma_{ij}} = -\lambda \dot{s}_i \frac{\partial \dot{s}_i}{\partial u_{2i}} \frac{\partial u_{2i}}{\partial \sigma_{ij}} \tag{3.13}$$

where, λ is learning rate. Due to

$$\frac{\partial(\dot{s}_i)}{\partial u_{2i}} = b_{ii} \tag{3.14}$$

$$\frac{\partial u_{2i}}{\partial \omega_{ij}} = \exp(-\|s_i - c_{ij}\|^2 / \sigma^2{}_{ij}) = \phi_{ij}(s_i) \tag{3.15}$$

$$\frac{\partial u_{2i}}{\partial c_{ij}} = 2\omega_{ij} \exp(-\|s_i - c_{ij}\|^2 / \sigma^2{}_{ij})(s_i - c_{ij}) / \sigma^2{}_{ij}$$

$$= 2\omega_{ij}\phi_{ij}(s_i)(s_i - c_{ij}) / \sigma^2{}_{ij} \tag{3.16}$$

$$\frac{\partial u_{2i}}{\partial \sigma_{ij}} = 2\omega_{ij} \exp(-\|s_i - c_{ij}\|^2 / \sigma^2{}_{ij}) \|s_i - c_{ij}\|^2 / \sigma^3{}_{ij}$$

$$= 2\omega_{ij}\phi_{ij}(s_i) \|s_i - c_{ij}\|^2 / \sigma^3{}_{ij} \tag{3.17}$$

The learning algorithm of neural network weights is as follows:

$$\omega_{ij}(k) = \omega_{ij}(k-1) + \dot{\omega}_{ij} + \gamma[\omega_{ij}(k-1) - \omega_{ij}(k-2)] \tag{3.18}$$

$$c_{ij}(k) = \omega_{ij}(k-1) + \dot{c}_{ij} + \gamma[c_{ij}(k-1) - c_{ij}(k-2)] \tag{3.19}$$

$$\sigma_{ij}(k) = \sigma_{ij}(k-1) + \dot{\sigma}_{ij} + \gamma[\sigma_{ij}(k-1) - \sigma_{ij}(k-2)] \tag{3.20}$$

where, γ is inertia coefficient.

Remark: in (3.14), b_{ii} is the element of $M^{-1}{}_0(q)$. Since

$$\begin{aligned} \dot{s} &= \ddot{e} + \alpha\dot{e} + \frac{q}{p}\beta \cdot \text{diag}(e^{q/p-1})\dot{e} \\ &= (\ddot{q} - \ddot{q}_r) + \alpha\dot{e} + \frac{q}{p}\beta \cdot \text{diag}(e^{q/p-1})\dot{e} \\ &= M_0{}^{-1}(q)(\tau - C_0(q, \dot{q}) - G_0(q) + \rho(t)) - \ddot{q}_r \\ &+ \alpha\dot{e} + \frac{q}{p}\beta \cdot \text{diag}(e^{q/p-1})\dot{e} \end{aligned}$$

Substituting (3.6) into above, then

$$\dot{s} = M_0{}^{-1}(q)(u_2 + \rho(t))$$

Suppose that b_{ij} is the element of matrix $M^{-1}{}_0(q)$, then

$$\dot{s}_i = b_{ii}u_{2i} + \sum_{j=1, j\neq i}^{n} b_{ij}u_{2j} + \rho_i(t)$$

$$\frac{\partial(\dot{s}_i)}{\partial u_{2i}} = \frac{\partial(\dot{s}_i)}{\partial u_{2i}} = b_{ii}$$

3.3 Stability and Convergence Analysis

Define a Lyapunov function candidate

$$V=\sum_{i=1}^{n}V_i=\sum_{i=1}^{n}E_i=\sum_{i=1}^{n}(\frac{1}{2}s_i^2+\frac{1}{2}\dot{s}_i^2)\geq 0 \tag{3.21}$$

The derivative of the Lyapunov function can be written as

$$\dot{V}=\sum_{i=1}^{n}\dot{V}_i=\sum_{i=1}^{n}\sum_{j=1}^{m}(\frac{\partial V_i}{\partial \omega_{ij}}\frac{\partial \omega_{ij}}{\partial t}+\frac{\partial V_i}{\partial \sigma_{ij}}\frac{\partial \sigma_{ij}}{\partial t}+\frac{\partial V_i}{\partial c_{ij}}\frac{\partial c_{ij}}{\partial t})+g(z)\dot{z} \tag{3.22}$$

where, $g(z)\dot{z}$ represents the derivative of V with respect to the variables other than the controller parameters, and it can be easily proven that $g(z)\dot{z}\to 0$ if $s\to 0$ and $\dot{s}\to 0$.

Putting (3.11), (3.12), (3.13) into (3.22), then

$$\dot{V}=-\sum_{i=1}^{n}\sum_{j=1}^{m}\lambda[(\frac{\partial V_i}{\partial \omega_i})^2+(\frac{\partial V_i}{\partial \sigma_i})^2+(\frac{\partial V_i}{\partial c_i})^2]+g(z)\dot{z} \tag{3.23}$$

Therefore, the controlled system is stable if the learning gain is large enough to make the derivative of the Lyapnov function negative definite. Fortunately, the stability of the system can be easily achieved by increasing the amplifier gain of e. However, increasing the amplifier gain of e too much is harmful to controlled system accuracy.

In order to analyze the convergence of tracking error, define a Lyapnov function

$$V_e=\frac{1}{2}e^Te=\frac{1}{2}\sum_{i=1}^{n}e_i^2>0 \tag{3.24}$$

then

$$\dot{V}_e=e^T\dot{e}=\sum_{i=1}^{n}e_i\dot{e}_i \tag{3.25}$$

Putting (2.5) into (3.25), then

$$\dot{V}_e=e^T\dot{e}=e^T(s-\alpha e-\beta e^{q/p})=e^Ts-\alpha e^Te-\beta e^Te^{q/p} \tag{3.26}$$

Considering $s\to 0$, then

$$\dot{V}_e\approx-\alpha e^Te-\beta e^Te^{q/p}=-\sum_{i=1}^{n}(e_i^2+e_i^{(p+q)/p})<0 \tag{3.27}$$

Therefore, $e\to 0$. The tracking error is convergent, and the controlled system has not steady error.

The time of tracking error being zero can get by solving following equation:

$$\dot{e}=-\alpha e-\beta e^{q/p} \tag{3.28}$$

Ii is obvious that the solution of (3.28) is definite. So the controlled system can converge in finite time.

4. SIMULATIONS AND DISCUSSIONS

In order to verify above algorithm, a two-joint robot simulated in reference [2] is selected to be applied, whose dynamics equation is as follow:

$$\begin{bmatrix} a_{11}(q_2) & a_{12}(q_2) \\ a_{21}(q_2) & a_{22} \end{bmatrix}\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}+\begin{bmatrix} \lambda_1(q_1,q_2)g \\ \lambda_2(q_1,q_2)g \end{bmatrix}$$

$$+\begin{bmatrix} -\beta_{12}(q_2)\dot{q}_1 & -2\beta_{12}(q_2)\dot{q}_1 \\ 0 & \beta_{12}(q_2)\dot{q}_2 \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}=\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

where

$a_{11}(q_2)=(m_1+m_2)r_1^2+m_2r_2^2+2m_2r_1r_2\cos(q_2)+J_1$

$a_{12}(q_2)=m_2r_2^2+m_2r_1r_2\cos(q_2)$

$a_{22}=m_2r_2^2+J_2$

$\beta_{12}(q_2)=m_2r_1r_2\sin(q_2)$

$\lambda_1(q_1,q_2)=((m_1+m_2)r_1\cos(q_2)+m_2r_2\cos(q_1+q_2))$

$\lambda_2(q_1,q_2)=m_2r_2\cos(q_1+q_2)$

The nominal parameters of the controlled system are selected as follow:

$r_1=1m$, $r_2=0.8m$, $J_1=J_2=5kg\cdot m$, $m_1=0.5kg$, $m_2=1.5kg$.

The parameters of sliding mode manifold are $p=3$, $q=5$, $\alpha_1=\alpha_2=1$, $\beta_1=\beta_2=1$.

Suppose that the initial state of the controlled system is $q_1=1$ rad, $\dot{q}_1=0$ rad/s, $q_2=1.5$ rad, $\dot{q}_2=0$ rad/s. A RBF neural network includes 1 input cell, 3 hide cell, 1 output cell. The expected trajectories are as follows:

$q_{r1}=1.25-(7/5)e^{-t}+(7/20)e^{-4t}$

$q_{r2}=1.25+e^{-t}-(1/4)e^{-4t}$

When system parameter errors and external disturbances are selected as $\rho(t)=0.1+0.2q+0.3\dot{q}$, the simulation are shown in Fig. 2 by GFTSM. The simulations are shown in Fig.3 by DNSMC, whose learning rate $\lambda=0.6$, inertial coefficient $\gamma=0.1$, initial value of RBF network weights are $\omega=[0.5,0.5,0.5]$, the initial center value of RBF are $C=[-0.5,0,0.5]$, the initial width value of RBF are $\sigma=[1,1,1]$. qr1(t), q1(t), u1(t) in Fig. 2 and Fig. 3 represent the expected trajectory, actual running trajectory and controller output for the joint 1 respectively, and qr2(t), q2(t), u2(t) represent the expected trajectory, actual running trajectory and controller output for the joint 2 respectively.

When the DNSMC controller is fixed, but the system parameter errors and external disturbances are increased as $\rho(t)=1+2q+3\dot{q}$, the simulation are shown in Fig. 4.

When the system parameter errors and external disturbances are unknown, Fig. 3 shows that simulation results by DNSMC are better than one by GFTSMC shown in Fig. 2, and chattering is eliminated. When system parameter errors and external disturbances are increased, Fig. 4 shows that the trajectory tracking errors have the same result as Fig. 3, only the controller output are stronger than that in Fig.3. So the DNSMC controller has strong robustness and is suitable to trajectory tracking control of multi-link robot with uncertain system parameter errors and external disturbances.

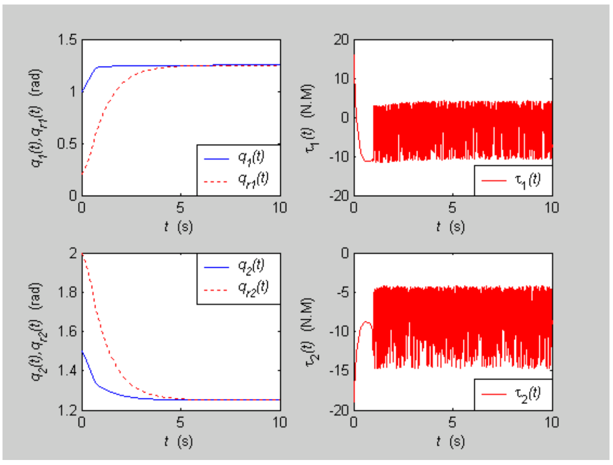

Fig. 2 Simulation by GFTSMC

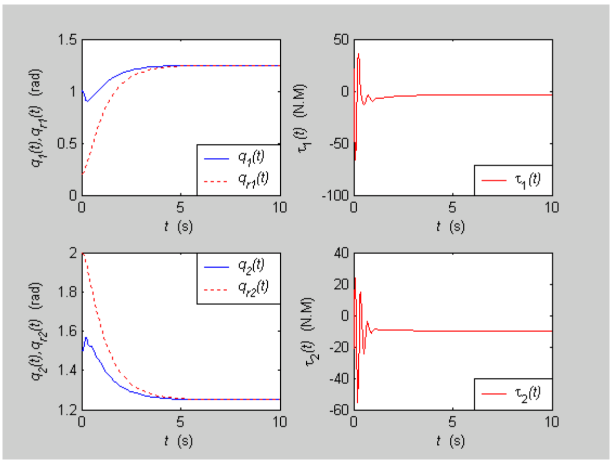

Fig. 3 Simulation by DNSMC

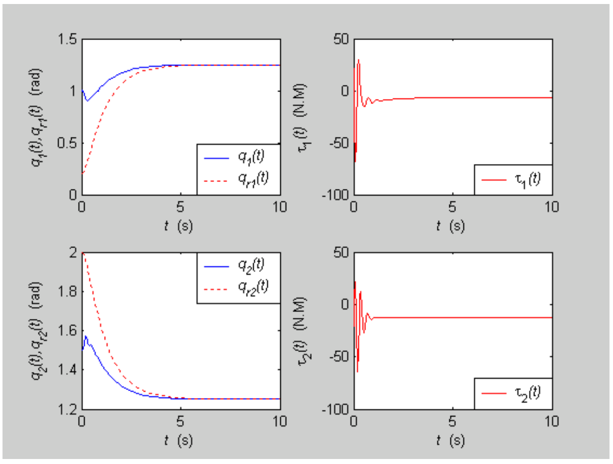

Fig. 4 Simulation by DNSMC in stronger disturbances

5. CONCLUSION

This paper presents a new method for trajectory tracking control of multi-link robots with uncertain system parameter errors and external disturbances. This method uses a global fast terminal sliding mode control manifold, which guarantees that the controlled system can reach the manifold and equivalent point in finite time. A RBF neural network is utilized to learn the boundary of system parameter errors and external disturbances for every joint. The switching gain of sliding mode control can be automatically adjusted by RBF neural networks learning according to model error and external disturbance. Chattering is also eliminated. Simulation verifies the validity of the algorithm, but this method need a short time to learn system parameter errors and external disturbances before it runs well. How to shorten this learning time needs to study in the future.

ACKNOWLEDGE

This work was supported by Doctoral Fund of Ministry of Education of China, 20060005014 and by the National Natural Sciences Foundation of China 60774037

REFERENCES

D. Munoz, D. Sbarbaro, 2000. An adaptive sliding mode controller for discrete nonlinear systems. IEEE Tans on industry and electronics. 47(3): 574-581.

Feng Y, Yu X. H, Man Z. H, 2002. Non-singular terminal sliding mode control of rigid manipulators. Automatica. 38(12): 2159-2167

H. Hu, P. Y. Woo, 2006. Fuzzy supervisory sliding mode and neural network control for manipulators. IEEE Tans on industry and electronics. 53(3): 929-940.

H. Morioka, K. Wada, A. Sabanovic, et al, 1995. Neural network based chattering free sliding mode control. Proceeding of the 34th SICE anniversary conference, 1303-1308.

Huang S. J., Huang K. S., Chiou K. C, 2003. Development and application of a novel radial basis function sliding mode controller. Mechatronics. 13(4): 313-329.

K. Jezernik, M. Rodic, R. Safaric, et al, 1997. Neural network sliding mode robot control. Robtica, 15(1): 23-30.

Li C. T., Tan Y. H, 2005. Neural sliding mode control for systems with hysteresis. Proceedings of the 2005 IEEE International Symposium on Intelligent Control. 467-472.

Lin C. K., 2006. Nonsingular Terminal Sliding Mode Control of Robot Manipulators Using Fuzzy Wavelet Networks. IEEE Transactions on Fuzzy Systems, 14(6):849-859.

Song Z. S., Yi J. Q., Zhao D. B., et al, 2005. A computed torque controller for uncertain robotic manipulator systems: Fuzzy approach. Fuzzy Sets and Systems. 154 (2): 208–226

Sun F C, Sun Z Q, Feng G, 1999. An adaptive fuzzy controller based on sliding mode for robot manipulators. IEEE Trans on Systems, Man and Cybernetics, Part B: Cybernetic, 29(4): 661-667.

Y. Yildiz, A. Sabanovic, K.Abidi, 2007. Sliding mode neuro controller for uncertain systems. IEEE Tans on industry and electronics. 54(3): 1676-1684.

Yu S. H, Yu X. H, 2000. Robust Global Terminal Sliding Mode Control of SISO Nonlinear Uncertain Systems. Proceedings of the 39th IEEE Conference on Decision and Control, 2198-2203