Adaptive Neural Network Multiple Models Sliding Mode Control of Robotic Manipulators Using Soft Switching

Nasser Sadati, Rasoul Ghadami, Mahdi Bagherpour
Department of Electrical Engineering, Shrif University of Thechnology, Tehran, Iran.
E-mail: sadati@sina.sharif.edu

Abstract

In this paper, an adaptive neural network multiple models sliding mode controller for robotic manipulators is presented. The proposed approach remedies the previous problems met in practical implementation of classical sliding mode controllers. Adaptive single-input single-output (SISO) RBF neural networks are used to calculate each element of the control gain vector; discontinuous part of control signal, in a classical sliding mode controller. By using the multiple models technique the nominal part of the control signal is constructed according to the most appropriate model at different environments. The key feature of this scheme is that prior knowledge of the system uncertainties is not required to guarantee the stability. Also the chattering phenomenon is completely eliminated. Moreover, a theoretical proof of the stability and convergence of the proposed scheme using Lyapunov method is presented. To demonstrate the effectiveness of the proposed approach, a practical situation in robot control is simulated.

1. Introduction

Robotic manipulators are highly nonlinear, highly time-varying and highly coupled. Moreover, there always exists uncertainty in the system model which cause unstable performance in the robotic system. Therefore, they are complex systems and in general very difficult to control. Robust control is a powerful tool to control complex systems [1]. The typical structure of a robust controller is composed of a nominal part, similar to a feedback linearization or inverse control law, and additional terms aimed at dealing with uncertainties. Almost all kinds of robust control schemes, including the classical sliding mode control [2], have been proposed in the field of robotic control during the past decades. Classical sliding mode controller design provides a systematic approach to the problem of maintaining stability in the face of modeling imprecision and uncertainty.

Multiple models control strategy is another scheme to control the complex systems, such as robotic manipulators, [3], [4]. Because of uncertainties mentioned above, these systems operate in multiple environments which may change abruptly from one to another. Therefore, a good way to improve controller performance is to use multiple models control if models are approximately available for different environments [5]. The objective in multiple models controller is to determine the most appropriate model at any instant, using a suitable performance criterion based on the identification error, and to activate the corresponding controller.

Although classical sliding mode control is a powerful scheme for nonlinear systems with uncertainty, such as robotic manipulators [1], this control scheme has important drawbacks limiting its practical applicability, such as chattering and large control authority. Moreover, in order to guarantee the stability of the sliding mode control systems, the boundary of the uncertainty has to be estimated.

Recently, much research works have been done to use soft-computing methodologies such as artificial neural networks in order to improve the performance and remedy the problems met in practical implementation of sliding mode controllers [6]. The use of neural network (NN) for calculation of the equivalent term of a sliding mode controller (SMC) is proposed in [7]. In [8] two NNs in parallel are used to realize the equivalent control and corrective control terms of an SMC. This scheme is based on the fact that if the NN learns the equivalent control, the corrective term goes to zero and any difference between them is reflected as a nonzero corrective term. In [9], by adaptively estimating the bound of system uncertainty using a multi-input single-output RBF neural network, the requirement for having prior knowledge of uncertainty is eliminated. However, there is still chattering in the control input. In [10] the gains of an SMC are accepted as the weights of the NN and the weights are updated to minimize the defined cost function. The proposed adaptation scheme is MIT rule and there is no guarantee for convergence and stability.

In this paper, the combination of neural network, sliding mode control and multiple models control are used for controlling the robotic manipulator with robust characteristics. This is accomplished in such a way that the multiple models approach is used to construct the equivalent control of the sliding mode control signal. The use of multiple models approach offers more robustness and reduces both the control gain authority and tracking error of the transient state. In other side, the discontinuous part of the control signals in the classical sliding mode controllers are substituted by single-input single-output RBF neural network functions, which are nonlinear and continuous, to eliminate the chattering phenomenon. To relax the requirements for the knowledge of upper bound of the uncertainties, the weights of the hidden layer of RBF neural networks are updated in an on-line manner to compensate the system uncertainties and to guarantee the stability of the overall system without having any prior knowledge of the system uncertainties. The adaptive law is designed based on the Lyapunov method and mathematical proof for the stability and convergence of the overall system is provided.

The outline of this paper is as follows. Preliminaries about the model of the robotic manipulator, as partly known system, and the classical sliding mode controller for robotic manipulators are summarized in section II. The structure of classical multiple models approach is described in section III. The adaptive neural network multiple models sliding mode controller for robotic manipulators is presented in section IV. The simulation results are given in section V to demonstrate the effectiveness of the proposed control scheme. Finally, section VI presents some concluding remarks.

2. Preliminaries

2.1. Model or robotic manipulators

In the absence of friction or other disturbances, the dynamic equation of an n-link rigid robotic manipulator system is described by the following second order nonlinear vector differential equation

$$\boldsymbol{M(q)\ddot{q}} + \boldsymbol{B(q,\dot{q})\dot{q}} + \boldsymbol{G(q)} = \boldsymbol{u} \qquad (1)$$

where $\boldsymbol{q} = [q_1,...,q_n]^T$ is an $n\times 1$ vector of joint angular position, as shown is Fig. 1 for a two-link robot manipulator, $\dot{\boldsymbol{q}} = [\dot{q}_1,...,\dot{q}_n]^T$ and $\ddot{\boldsymbol{q}} = [\ddot{q}_1,...,\ddot{q}_n]^T$ are $n\times 1$ vectors of corresponding velocity and acceleration, $\boldsymbol{u}$ is an $n\times 1$ vector of applied joint torques (control inputs), $\boldsymbol{M(q)}$ is an $n\times n$ inertial matrix, $\boldsymbol{B(q,\dot{q})}$ is an $n\times n$ matrix of Coriolis and centrifugal forces and $\boldsymbol{G(q)}$ is an $n\times 1$ gravity vector. The inertial matrix $\boldsymbol{M(q)}$ is symmetric and positive definite. It is also bounded as a function of $\boldsymbol{q}$: $\mu_1 I \le \boldsymbol{M(q)} \le \mu_2 I$. $\dot{\boldsymbol{M}}\boldsymbol{(q)} - 2\boldsymbol{B(q,\dot{q})}$ is skew symmetric matrix, that is, $x^T[\dot{\boldsymbol{M}}\boldsymbol{(q)} - 2\boldsymbol{B(q,\dot{q})}]\,x = 0$, where x is an $n\times 1$ nonzero vector.

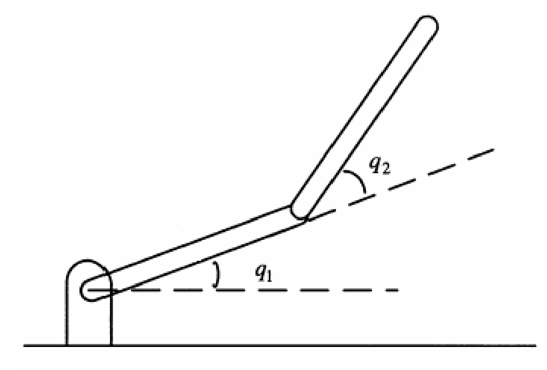

Fig.1 Two-link robot manipulator.

It is assumed that a robotic manipulator, as is described by (1), has some known parts and some unknowns and therefore, there exist uncertainty in the system model. Thus $\boldsymbol{M(q)}$, $\boldsymbol{B(q,\dot{q})}$ and $\boldsymbol{G(q)}$ can be described as

$$\begin{aligned}
\boldsymbol{M(q)} &= \hat{\boldsymbol{M}}\boldsymbol{(q)} + \Delta\boldsymbol{M(q)} \\
\boldsymbol{B(q,\dot{q})} &= \hat{\boldsymbol{B}}\boldsymbol{(q,\dot{q})} + \Delta\boldsymbol{B(q,\dot{q})} \\
\boldsymbol{G(q)} &= \hat{\boldsymbol{G}}\boldsymbol{(q)} + \Delta\boldsymbol{G(q)} \qquad (2)
\end{aligned}$$

where $\hat{\boldsymbol{M}}\boldsymbol{(q)}$, $\hat{\boldsymbol{B}}\boldsymbol{(q,\dot{q})}$ and $\hat{\boldsymbol{G}}\boldsymbol{(q)}$ are the known parts or estimated parameters, and $\Delta\boldsymbol{M(q)}$, $\Delta\boldsymbol{B(q,\dot{q})}$ and $\Delta\boldsymbol{G(q)}$ are the unknown parts. For simplification in notation, we avoid writing the variables in the parentheses of the above matrices and vectors from now on.

2.2. Classical sliding mode controller

In the design of sliding mode controller for robotic manipulators, the control objective is to drive the joint position $\boldsymbol{q}$ to the desired position $\boldsymbol{q_d}$. So by defining the tracking error to be in the following form

$$\boldsymbol{e} = \boldsymbol{q} - \boldsymbol{q_d} \tag{3}$$

the sliding surface can be written as

$$\boldsymbol{s} = \dot{\boldsymbol{e}} + \boldsymbol{\lambda e} \tag{4}$$

where $\boldsymbol{\lambda} = diag[\lambda_1,...,\lambda_i,...,\lambda_n]$, in which λ_i is a positive constant. The control objective can now be achieved by choosing the control input so that the sliding surface satisfies the following sufficient condition

$$\frac{1}{2}\frac{d}{dt}s_i^{\,2} \le -\eta_i |s_i| \tag{5}$$

where η_i is a positive constant. Equation (5) indicates that the energy of $\boldsymbol{s}$ should decay as long as $\boldsymbol{s}$ is not zero. To set up the control $\boldsymbol{u}$, define the reference states to be

$$\begin{aligned} \dot{\boldsymbol{q}}_{\boldsymbol{r}} &= \dot{\boldsymbol{q}} - \boldsymbol{s} = \dot{\boldsymbol{q}}_{\boldsymbol{d}} - \boldsymbol{\lambda e} \\ \ddot{\boldsymbol{q}}_{\boldsymbol{r}} &= \ddot{\boldsymbol{q}} - \dot{\boldsymbol{s}} = \ddot{\boldsymbol{q}}_{\boldsymbol{d}} - \boldsymbol{\lambda}\dot{\boldsymbol{e}} \end{aligned} \tag{6}$$

Now the control input $\boldsymbol{u}$ can be chosen to be in the following form

$$\begin{aligned} \boldsymbol{u} &= \hat{\boldsymbol{u}} - \boldsymbol{As} - \boldsymbol{K}\,\mathrm{sgn}(\boldsymbol{s}) \\ \hat{\boldsymbol{u}} &= \hat{\boldsymbol{M}}\ddot{\boldsymbol{q}}_{\boldsymbol{r}} + \hat{\boldsymbol{B}}\dot{\boldsymbol{q}}_{\boldsymbol{r}} + \hat{\boldsymbol{G}} \end{aligned} \tag{7}$$

where $\boldsymbol{K} = diag[k_{11},...,k_{ii},...,k_{nn}]$ is a diagonal positive definite matrix in which k_{ii}'s are positive constants and $\boldsymbol{A} = diag[a_1,...,a_i,...,a_n]$ is a diagonal positive definite matrix in which a_i's are also positive constants. Putting (7) into (1) leads to

$$\boldsymbol{M}\dot{\boldsymbol{s}} + \boldsymbol{(B + A)s} = \boldsymbol{\Delta f} - \boldsymbol{K}\,\mathrm{sgn}(\boldsymbol{s}) \tag{8}$$

where $\boldsymbol{\Delta f} = \boldsymbol{\Delta M}\ddot{\boldsymbol{q}}_{\boldsymbol{r}} + \boldsymbol{\Delta B}\dot{\boldsymbol{q}}_{\boldsymbol{r}} + \boldsymbol{\Delta G}$. It can be proved that by choosing $\boldsymbol{K}$ such that

$$k_{ii} \ge |\Delta f_i|_{bound} \tag{9}$$

where $|\Delta f_i|_{bound}$ is the boundary of $|\Delta f_i|$, the overall system is asymptotically stable. It has been proven ([2]) that by considering the Lyapunov function candidate as

$$V = \frac{1}{2}\boldsymbol{s}^T \boldsymbol{Ms} \tag{10}$$

where $|\Delta f_i|_{bound}$ is the boundary of $|\Delta f_i|$, the overall system is asymptotically stable. Therefore the decay of the energy of $\boldsymbol{s}$, as long as $\boldsymbol{s} \neq 0$, is guaranteed and the sufficient condition in (5) is satisfied.

3. Classical multiple models control

As stated before, the multiple models control strategy is to determine the best model for the current environment at every instant and to apply the appropriate control signal correspondingly. Therefore it can be divided into two separated phases; identification and control.

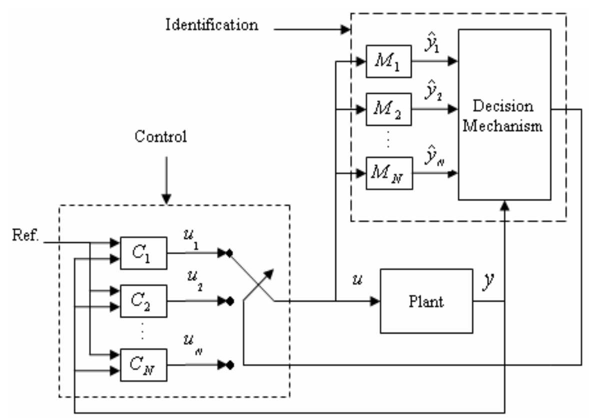

Fig.2 Block diagram of classical multiple models controller.

3.1. Identification

Identification part is composed of predefined models which cover different environments of the plant and decision mechanism in order to decide about which model is the closest one to the plant. Many techniques are used in identify the best model in multiple models controller [3]-[5]. Classical types of these controllers work based on the switching. Switching action is determined in decision mechanism part and in which the model with the smallest error, according to some criterion, is chosen rapidly. The structure of classical multiple models controller is presented in Fig. 1. $M_1, M_2, \ldots, M_N$ are N models which are used in parallel with the plant. y And $\hat{y}_n$ are the output of the plant and the nth model, respectively, and u is control signal or input of plant. There are many options to choose appropriate criterions or performance indices, and different types are introduced in literatures [3]-[5]. One specific performance index is given below

$$J_n(t) = a\, e_n{}^2(t) + b\int_0^t e^{-\kappa(t-\tau)} e_n{}^2(\tau)\, d\tau \qquad (11)$$

$$a \geq 0 \quad b, \kappa > 0$$

where

$$e_n(t) = y(t) = \hat{y}_n(t) \qquad (12)$$

a and b are the weighting factors on the instantaneous measure and the long term accuracy. κ is a forgetting factor which makes previous measures have different weightings and as time goes on, old values have less effect on the result. These parameters plays key role to have quick response or slow one. By choosing larger value for a/b and κ, more quick response can be achieved. But it leads to unwanted switching in the presence of disturbances and deteriorates the system performance. In other hand smaller value for them reduces unwanted switching but makes slower response for environment changing. Therefore a good option is a trade-off between two characteristics.

3.2. Control

The control problem in the multiple models technique is preformed in two steps:

- Generation of the basic control signal for each model separately
- Switching to the best controller according to the selected model

In classical multiple models approach, as seen in Fig. 2, N separated controllers, C_n for $n = 1, \ldots, N$, are designed for predefined models so that the control objective is satisfied for each of them. The output of nth controller is denoted by u_n and the final control, u, determined among N calculated control signals based on the best selected model in the control mechanism at each instant of time.

4. Adaptive Neural Network Multiple Models Sliding Model Control

There are major disadvantages in designing the classical sliding mode controllers. First, because of the control actions which are discontinuous across $\boldsymbol{s}$, there is chattering in a boundary of the surface $\boldsymbol{s}$. Such high frequency switching (chattering) might excite unmodeled dynamics and impose undue wear and tear on the actuators, so the control law would not be considered acceptable. Second, the prior knowledge of the boundary of uncertainty is required in compensators. If boundary is unknown, a large value has to be applied to the gain of discontinuous part of control signal and this large control gain may intensify the chattering on the sliding surface.

In this section, an adaptive neural network multiple models sliding mode controller, to avoid the aforementioned problems, has been proposed. For this purpose, first adaptive single-input single-output (SISO) RBF neural networks, as continuous function, is applied to calculate each elements of $\boldsymbol{K}\,\mathrm{sgn}(\boldsymbol{s})$ in the control law (7). The control input is written as

$$\boldsymbol{u} = \hat{\boldsymbol{u}} - \boldsymbol{A}\boldsymbol{s} - \boldsymbol{K} \qquad (13)$$

where $\boldsymbol{K} = [k_1, \ldots, k_i, \ldots, k_n]^T$ is an $n \times 1$ vector in which k_i is the output of the ith RBF neural network. The RBF neural networks have the following structure

$$k_i = \boldsymbol{W}_{k_i}{}^T \Phi_{k_i}(s_i) \qquad (14)$$

where $\boldsymbol{W}_{k_i}$ is the $m \times 1$ vector of the output layer weights and m is the number of nodes in the hidden layer and $\Phi_{k_i}(s_i) = [\varphi_{k_i}^1(s_i), \ldots, \varphi_{k_i}^j(s_i), \ldots, \varphi_{k_i}^m(s_i)]^T$ is the $m \times 1$ vector of outputs of the hidden layer nodes, whose elements (basis functions) are chosen as Gaussian-type function, expressed by

$$\varphi_{k_i}^j(s_i) = \beta_i \exp(-\left\| s_i - \mu_i^j \right\|^2 / 2\sigma_i^{j\,2}) \qquad (15)$$

where μ_i^j and σ_i^j are the center and variance of the jth basis function of the ith RBFN, respectively and β_i is a positive constant. $\boldsymbol{W}_{k_i}$ is chosen as the parameter to be updated and therefore is called the parameter vector, and $\Phi_{k_i}(s_i)$ is called the basis function vector which can be regarded as the weight of the parameter vector.

A well-known problem in adaptive control is the poor transient response which is observed when adaptation is initiated after an abrupt change. Such abrupt changes frequently occur in the practical situations in robotic manipulators. As mentioned, a good way to improve controller performance of such systems is to use multiple models control if models are approximately available for different environments. Therefore the multiple models technique is used to construct the equivalent term $\hat{\boldsymbol{u}}$ in the control law (13). The control block diagram of the proposed architecture is shown in Fig. 3, where the PD block indicates the computation of $d/dt + \lambda$ and the input of each RBF network is s_i and the corresponding output is k_i.

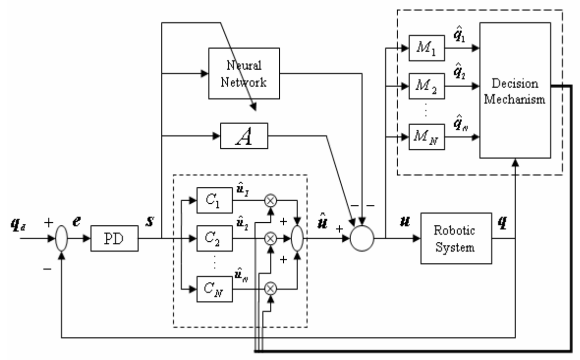

Fig.3 Adaptive neural network multiple models sliding mode control of robotic manipulator.

$M_1, M_2, \ldots, M_N$ are estimated models in different environment of robotic action which is used in parallel with the plant. In other words, $\hat{\boldsymbol{M}}_n$, $\hat{\boldsymbol{B}}_n$ and $\hat{\boldsymbol{G}}_n$, for $n=1,\ldots,N$, are the estimated parameters of the nth environment, and the unknown parts of the plant in each environment is defined as

$$\begin{aligned} \Delta\boldsymbol{M}_n &= \hat{\boldsymbol{M}}_n - \boldsymbol{M} \\ \Delta\boldsymbol{B}_n &= \hat{\boldsymbol{B}}_n - \boldsymbol{B} \\ \Delta\boldsymbol{G}_n &= \hat{\boldsymbol{G}}_n - \boldsymbol{G} \qquad n=1,\ldots,N \end{aligned} \tag{16}$$

Moreover $\boldsymbol{q}$ and $\hat{\boldsymbol{q}}_n$ are the output of the plant and nth model, respectively. $\hat{\boldsymbol{u}}_n$, for $n=1,\ldots,N$, is the output of nth local controller corresponding to nth model which is defined as

$$\hat{\boldsymbol{u}}_n = \hat{\boldsymbol{M}}_n \ddot{\boldsymbol{q}}_r + \hat{\boldsymbol{B}}_n \dot{\boldsymbol{q}}_r + \hat{\boldsymbol{G}}_n \tag{17}$$

In identification mechanism part, the performance index is selected as

$$J_n(t) = a\,\|\hat{\boldsymbol{e}}_n(t)\| + b\int_0^t e^{-\kappa(t-\tau)}\|\hat{\boldsymbol{e}}_n(\tau)\|\,d\tau$$

$$a \ge 0 \quad b,\kappa > 0 \tag{18}$$

where

$$\hat{\boldsymbol{e}}_n(t) = \boldsymbol{q}(t) - \hat{\boldsymbol{q}}_n(t) \tag{19}$$

Since we use blending instead of switching between each controller , as seen in Fig. 3, to achieve continuous control signal, the control signal is written by

$$\boldsymbol{u} = \sum_{n=1}^{N} \alpha_n \hat{\boldsymbol{u}}_n - \boldsymbol{A}\boldsymbol{s} - \boldsymbol{K} \tag{20}$$

where blending coefficient α_n is defined as

$$\alpha_n(t) = \frac{e^{-|J_n(t)|^2/\delta^2}}{\sum_{n=1}^{N} e^{-|J_n(t)|^2/\delta^2}} \tag{21}$$

The soft-max transforms the performance index using exponential function and then normalize these values so that α_n's have the property to lie between 0 and 1 , and $\sum_{n=1}^{N}\alpha_n = 1$.

In continue, an adaptive law is designed to guarantee that k_i's can compensate the system uncertainties. Now putting (20) into (1) leads to

$$\boldsymbol{M}\dot{\boldsymbol{s}} = -(\boldsymbol{B}+\boldsymbol{A})\,\boldsymbol{s} + \Delta\boldsymbol{f} - \boldsymbol{K} \tag{22}$$

where $\Delta\boldsymbol{f} = \sum_{n=1}^{N}\alpha_n \Delta\boldsymbol{f}_n$ in which $\Delta\boldsymbol{f}_n = \Delta\boldsymbol{M}_n\ddot{\boldsymbol{q}}_r + \Delta\boldsymbol{B}_n\dot{\boldsymbol{q}}_r + \Delta\boldsymbol{G}_n$. It is proved that the RBF neural networks are universal approximators if their basis functions are chosen as a scaled version of Gaussian functions, which means that these neural networks are capable of approximating any real continuous function on a compact set to arbitrary accuracy [11].

Now, defining $\boldsymbol{W}_{k_{id}}$ so that $k_i = \boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i)$ is the optimal compensation for Δf_i, according to the property of universal approximation of RBF neural networks, there exists $\delta_i > 0$ satisfying

$$\left|\Delta f_i - \boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i)\right| \le \delta_i \tag{23}$$

where δ_i is arbitrary and can be chosen as small as possible. Defining

$$\tilde{\boldsymbol{W}}_{k_i} = \boldsymbol{W}_{k_i} - \boldsymbol{W}_{k_{id}} \tag{24}$$

It can be shown that by choosing the adaptive law as

$$\dot{\tilde{\boldsymbol{W}}}_{k_i} = s_i\Phi_{k_i}(s_i) \tag{25}$$

the overall system is asymptotically stable with respect to $\boldsymbol{s}$ and the actual joint angular positions converge to the desired ones.

Proof: Choose the Lyapunov candidate as

$$V = \frac{1}{2}\boldsymbol{s}^T\boldsymbol{M}\boldsymbol{s} + \frac{1}{2}\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\tilde{\boldsymbol{W}}_{k_i} \tag{26}$$

where $\boldsymbol{M}$ is symmetric positive matrix and $\tilde{\boldsymbol{W}}_{k_i}{}^T\tilde{\boldsymbol{W}}_{k_i} > 0$, therefore V is positive definite. Now

Consider the derivative of V , given by

$$\dot{V}=\frac{1}{2}[\dot{\boldsymbol{s}}^T\boldsymbol{Ms}+\boldsymbol{s}^T\dot{\boldsymbol{M}}\boldsymbol{s}+\boldsymbol{s}^T\boldsymbol{M}\dot{\boldsymbol{s}}]+\frac{1}{2}\sum_{i=1}^{n}[\dot{\tilde{\boldsymbol{W}}}_{k_i}{}^T\tilde{\boldsymbol{W}}_{k_i}+\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}]$$

$$=\frac{1}{2}[\boldsymbol{s}^T\dot{\boldsymbol{M}}\boldsymbol{s}+2\boldsymbol{s}^T\boldsymbol{M}\dot{\boldsymbol{s}}]+\frac{1}{2}\sum_{i=1}^{n}2\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=\boldsymbol{s}^T[\boldsymbol{M}\dot{\boldsymbol{s}}+\boldsymbol{Bs}]+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=\boldsymbol{s}^T[-(\boldsymbol{B}+\boldsymbol{A})\boldsymbol{s}+\boldsymbol{\Delta f}-\boldsymbol{K}+\boldsymbol{Bs}]+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=\boldsymbol{s}^T[-\boldsymbol{As}+\boldsymbol{\Delta f}-\boldsymbol{K}]+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=-\boldsymbol{s}^T\boldsymbol{As}+\boldsymbol{s}^T[\boldsymbol{\Delta f}-\boldsymbol{K}]+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=-\boldsymbol{s}^T\boldsymbol{As}+\sum_{i=1}^{n}(s_i[\Delta f_i-k_i])+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

Since $k_i=\tilde{\boldsymbol{W}}_{k_i}{}^T\Phi_{k_i}(s_i)+\boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i)$, then

$$\dot{V}=-\boldsymbol{s}^T\boldsymbol{As}+\sum_{i=1}^{n}(s_i[\Delta f_i-(\tilde{\boldsymbol{W}}_{k_i}{}^T\Phi_{k_i}(s_i)+\boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i))])$$

$$+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=-\boldsymbol{s}^T\boldsymbol{As}+\sum_{i=1}^{n}(s_i[\Delta f_i-\boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i)])$$

$$+\sum_{i=1}^{n}(-s_i\tilde{\boldsymbol{W}}_{k_i}{}^T\Phi_{k_i}(s_i))+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T\dot{\tilde{\boldsymbol{W}}}_{k_i}$$

$$=-\boldsymbol{s}^T\boldsymbol{As}+\sum_{i=1}^{n}(s_i[\Delta f_i-\boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i)])$$

$$+\sum_{i=1}^{n}\tilde{\boldsymbol{W}}_{k_i}{}^T[-s_i\Phi_{k_i}(s_i))+\dot{\tilde{\boldsymbol{W}}}_{k_i}]$$

Moreover, since the adaptive law in (17) is chosen as $\dot{\tilde{\boldsymbol{W}}}_{k_i}=s_i\Phi_{k_i}(s_i)$

$$\dot{V}=-\boldsymbol{s}^T\boldsymbol{As}+\sum_{i=1}^{n}(s_i[\Delta f_i-\boldsymbol{W}_{k_{id}}{}^T\Phi_{k_i}(s_i)]) \qquad (27)$$

From (23), there exist

$$\left|\Delta f_i-\boldsymbol{W}_{k_i}{}^T\Phi_{k_i}(s_i)\right|\le\delta_i$$

where δ_i can be chosen as small as possible. Now by assuming

$$\left|\Delta f_i-\boldsymbol{W}_{k_i}{}^T\Phi_{k_i}(s_i)\right|\le\delta_i\le\rho_i|s_i| \qquad (28)$$

where $0<\rho_i<1$, the second term at the right side of (27) satisfies

$$s_i[\Delta f_i-\boldsymbol{W}_{k_i}{}^T\Phi_{k_i}(s_i)]\le\rho_i|s_i|^2=\rho_i s_i{}^2$$

Therefore

$$\dot{V}\le-\boldsymbol{s}^T\boldsymbol{As}+\sum_{i=1}^{n}\rho_i s_i{}^2$$

$$\le\sum_{i=1}^{n}(-a_i s_i{}^2+\rho_i s_i{}^2)$$

Now by simply choosing $a_i>\rho_i$

$$\dot{V}\le\sum_{i=1}^{n}(\rho_i-a_i)s_i{}^2\le 0 \qquad (29)$$

In (29) since $(\rho_i-a_i)<0$, $\dot{V}=0$ only when $s_i=0$. Thus, the overall system with the adaptive law in (25) is asymptotically stable. In other words

$$\lim_{t\to\infty}\boldsymbol{s}=\lim_{t\to\infty}(\dot{\boldsymbol{e}}+\boldsymbol{\lambda e})=0 \qquad (30)$$

or equivalently

$$\lim_{t\to\infty}\boldsymbol{e}=0\Rightarrow\lim_{t\to\infty}\boldsymbol{q}=\boldsymbol{q_d}$$
$$\lim_{t\to\infty}\dot{\boldsymbol{e}}=0\Rightarrow\lim_{t\to\infty}\dot{\boldsymbol{q}}=\dot{\boldsymbol{q}}_{\boldsymbol{d}} \qquad (31)$$

Therefore, it is proved that the adaptive multiple models sliding mode control input (20) drives the actual joint positions to their desired values. Q.E.D.

5. Simulation

In this section, the proposed adaptive multiple models sliding mode controller is used on a two-link robotic manipulator, whose parameter matrices are as follows [12]

$$\boldsymbol{M(q)}=\begin{bmatrix}M_{11} & M_{12}\\ M_{21} & M_{22}\end{bmatrix},\ \boldsymbol{B(q,\dot{q})}=\begin{bmatrix}B_{11} & B_{12}\\ B_{21} & 0\end{bmatrix},$$

$$\boldsymbol{G(q)}=\begin{bmatrix}G_1\\ G_2\end{bmatrix}$$

where

$$M_{11}=(m_2+m_3)l_2{}^2+(m_2+m_3)l_1l_2\cos q_2$$
$$M_{12}=(m_1+m_2+m_3)l_1^2+(m_2+m_3)l_2^2+2(m_2+m_3)l_1l_2\cos q_2$$
$$M_{12}=M_{21}$$
$$M_{22}=(m_2+m_3)l_2{}^2$$
$$B_{11}=-2(m_2+m_3)l_1l_2\dot{q}_2\sin q_2$$
$$B_{12}=-(m_2+m_3)l_1l_2\dot{q}_2\sin q_2$$
$$B_{21}=(m_2+m_3)l_1l_2\dot{q}_1\sin q_2$$

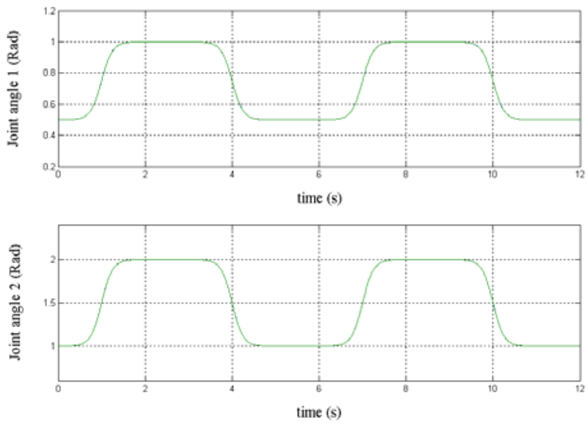

Fig.4 Tracking of joint angles 1 and 2 (In this figure, the obtained trajectory coincides with the desired one).

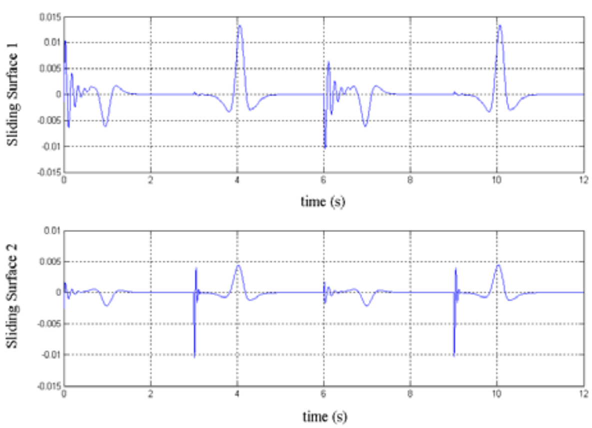

Fig.5 Sliding surface of joints 1 and 2.

$$G_1 = (m_1 + m_2 + m_3) g l_1 \cos q_1 + (m_2 + m_3) g l_2 \cos(q_1 + q_2)$$
$$G_2 = (m_2 + m_3) g l_2 \cos(q_1 + q_2)$$

where m_1 and m_2 are the masses, and l_1 and l_2 are the lengths of the links 1 and 2, respectively. The values of these parameter are chosen as $m_1 = 2\,kg$, $m_2 = 1\,kg$, $l_1 = 2\,m$ and $l_2 = 1\,m$. m_3 represents the mass of the load at the end of the link 2. The situation that is simulated is as following. The manipulator is expected to take a load from position two at $[1,2]^T\ Rad$ to position one at $[0.5,1]^T\ Rad$ repeatedly. In the first stage, the manipulator moves from position 1 to the position 2 along a predefined trajectory during 2 sec. It stays there for 1 sec. to take the load $m_3 = 2\,kg$ and start to move from position 2 to position 1 at $t = 3$ sec. It puts the load in position 1 at $t = 6$ sec. and repeats the above actions.

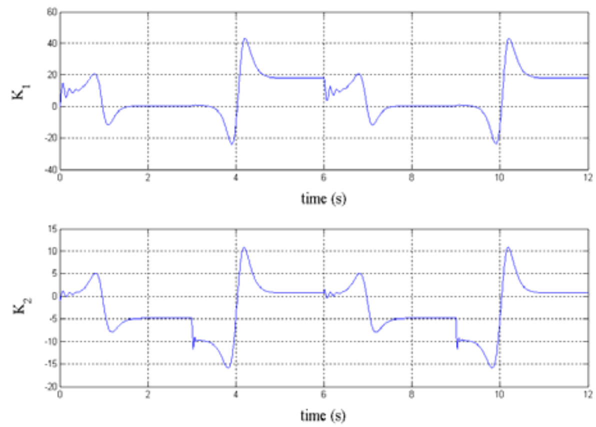

Fig.6 Control gains of joints 1 and 2

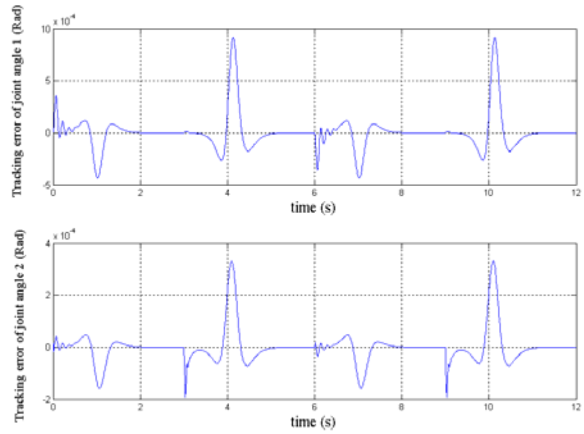

Fig.7 Tracking errors of joint angles 1 and 2.

In this simulation, two models are selected for the situation of $m_3 = 0$ and $m_3 = 2\,kg$, i.e. $N = 2$. Each model is estimated by applying a factor to the corresponding parameter matrices of the original system in each environment to count uncertainties, i.e., $\hat{\boldsymbol{G}}_n = 0.85\boldsymbol{G}$, $\hat{\boldsymbol{B}}_n = 0.8\boldsymbol{B}$, $\hat{M}_{n11} = 0.9M_{11}$, $\hat{M}_{n12} = 0.9M_{12}$, $\hat{M}_{n21} = 0.9M_{21}$ and $\hat{M}_{n22} = M_{22}$ for $n = 1,2$, using $m_3 = 0$ and $m_3 = 2\,kg$ respectively. Parameters of identification part is chosen as $a/b = 1$, $\kappa = 8$ and $\delta = 0.01$. The control input $\boldsymbol{u}$ is chosen as in (20), where $\boldsymbol{A} = diag[50,50]$, $\boldsymbol{\lambda} = diag[10,10]$ and each element of $\boldsymbol{K}$ is constructed by an RBF neural network with 10 nodes in its hidden layer. The initial parameters of each RBF neural network are evaluated by gradient descent algorithm to approximate a continuous quasi-signum function.

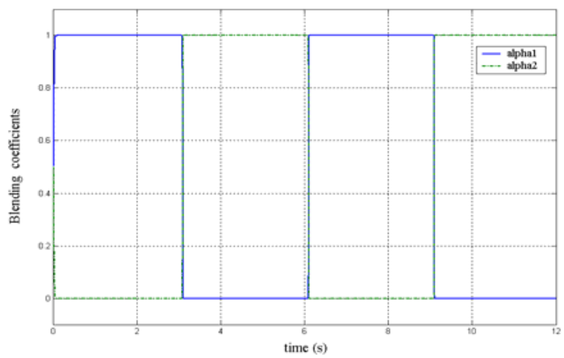

Fig.8 Blending coefficient of control action.

The simulation results are shown in Fig. 4 - Fig. 9. As seen in Fig. 4, the joint angles track the desired trajectories and the proposed control scheme drives the robotic manipulator to its desired positions. Moreover, there is no chattering in the sliding surface as shown in Fig. 5, and also the values of $\boldsymbol{K}$ converge to constant in the steady-state as is shown in Fig. 6. Also Fig. 7 shows that the tracking errors converge to zero. Finally, Fig. 8 shows the blending coefficients of control action.

6. Conclusion

In this paper an adaptive multiple models sliding mode controller using RBF neural network is proposed for robotic manipulators. The discontinuous parts of the classical sliding mode controller are replaced by SISO RBF neural networks, which are nonlinear and continuous, to avoid the chattering. The weights of the output layer of RBF neural networks are updated in an on-line manner to compensate the system uncertainties and the system stability without any prior knowledge of the system uncertainties is guaranteed. The equivalent control term of control signal is constructed by blending the local control action according to the most appropriate model at any environment. In this way the tracking errors of the transient state is reduced considerably. Since the RBF neural networks which are used in the controller are SISO systems, therefore the learning process is simple compared to that of multi-input systems developed previously. Also the design and implementation of the controller is simplified. Moreover, the stability and convergence of the overall system are proved by the Lyapunov method. The simulation results demonstrate that the proposed adaptive multiple model sliding mode control scheme, as proved theoretically, is a stable control scheme for robotic manipulators and works well in complicated situations

7. References

[1] J. E. Slotine, W. Li, *Applied Nonlinear Control*. Prentice-Hall Inc. New Jersy, 1991

[2] K. D. Yong, "Controller design for a manipulator using theory of variable structure system,".*IEEE Trans. Systs., Man, Cybern*., vol. SMC-8, no .2, pp. 210-218, Feb. 1978.

[3] K. S. Narendra, J. Balakrishnan, "Adaptive control using multiple models, " *IEEE Trans. Automatic Control*, vol. 42, no. 2, pp. 171-187, Feb. 1997.

[4] Kumpati. S. Narendra, Cheng xiang, "Adaptive control of discrete-time systems using multiple models," *IEEE Trans. Automatic control*, vol. 45, no. 9, pp. 1669-1686, sept. 2000.

[5] K S. Narendra, J. Balakrishnan, "Improving transient response of adaptive control systems using multiple models and switching," *IEEE Trans. Automatic Control*, vol. 39, pp. 1861-1866, Sept. 1994.

[6] O. Kaynak, K. Erbartur, M. Ertugrul. "The Fusion of computationally intelligent methodologies and sliding mode control-a survey," *IEEE Trans. on Industrial Electronics*, vol. 48, no. 1, pp. 4-17, Feb. 2001.

[7] K. Zerenik, M. Rodic, R. Safaric, B. Curk, "Neural network sliding mode robot control, " *Robotica*, vol. 15, pp. 23-30, 1997.

[8] M. Ertugrul, O. Kaynak, "Neuro sliding mode control of robotic manipulators," *Mechatron*, vol. 10, no 1-2, pp. 243-267, 2000.

[9] Man Zihang, X. H. Yo, K.Eshraghian, M. Panasiwami. "A robust adaptive sliding mode tracking control using an RBF neural network for robotic manipulator," in Proc. *IEEE International Conference on Neural Networks*, vol. 5,.1995, pp. 2403-2408.

[10] M. Ertugrul, O. Kaynak, "Neural network adaptive sliding mode control and its application to SCARA type robot manipulator," in Proc. *IEEE ICRA'97*, 1997, pp. 2932-2937.

[11] J. -S. Roger Jang, C. –T. Sun, "Functional equivalence between radial basis function networks and fuzzy systems," *IEEE Trans. on Neural Networks*. vol. 4, no. 1, pp. 156-159, Jan. 1993.

[12] J. J. Craig, *Introduction to Robotics*. Reading, MA: Addison-Wesley, 1989.