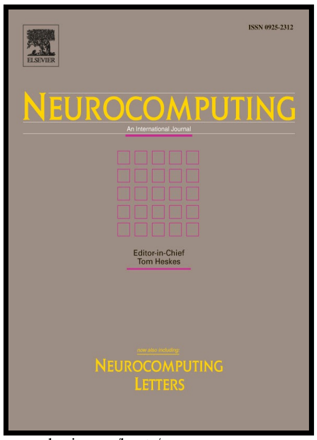
Author's Accepted Manuscript



www.elsevier.com/locate/neucom

Adaptive Terminal Sliding Mode Control of Uncertain Robotic Manipulators Based on Local Approximation of a Dynamic System

Minh-Duc Tran, Hee-Jun Kang

PII: S0925-2312(16)31275-9
DOI: http://dx.doi.org/10.1016/j.neucom.2016.09.089
Reference: NEUCOM17672

To appear in: *Neurocomputing*

Received date: 20 December 2015
Revised date: 6 September 2016
Accepted date: 18 September 2016

Cite this article as: Minh-Duc Tran and Hee-Jun Kang, Adaptive Termina Sliding Mode Control of Uncertain Robotic Manipulators Based on Loca Approximation of a Dynamic System, *Neurocomputing* http://dx.doi.org/10.1016/j.neucom.2016.09.089

Adaptive Terminal Sliding Mode Control of Uncertain Robotic Manipulators Based on Local Approximation of a Dynamic System

Minh-Duc Tran[a], Hee-Jun Kang[b2*]

[a]Graduate School of Electrical Engineering, University of Ulsan, 680-749, Ulsan, South Korea
[b]School of Electrical Engineering, University of Ulsan, 680-749, Ulsan, South Korea

ductm.ctme@gmail.com
hjkang@ulsan.ac.kr

[*]Corresponding author: Tel.: +82-10-3568-1289; fax: +82-52-259-1686

Abstract
This paper presents a novel adaptive finite-time control for robotic manipulators using terminal sliding mode control (TSMC) and radial basis function neural networks (RBFNNs). Firstly, the controller is developed based on terminal sliding mode which requires the prior knowledge of the robot dynamic model. Secondly, RBFNNs are adopted to directly approximate all parts of the system parameters through Ge-Lee (GL) matrix and its product operators. Moreover, an error estimator is added to suppress the approximation errors of neural networks (NNs) and external disturbances. And then, an adaptive finite-time control law with a proper update law is designed to guarantee the occurrence of the sliding motion in finite time without relying on a priori knowledge of uncertainties and external disturbances. The stability and finite-time convergence of the closed loop system are established by using the Lyapunov theory. Finally, the simulation results of a two-link robot manipulator are presented to illustrate the effectiveness of the proposed control method.

Keywords: Nonsingular terminal sliding mode control, radial basis function neural network, adaptive control, finite-time convergence, robot manipulator.

---

[2] Preliminary version of this manuscript has been selected in International Conference on Intelligent Computing (ICIC), 2015 (Paper ID: 470, Title: A Local Neural Networks Approximation Control of Uncertain Robot Manipulators) and its extended version is sub-selected for the Neurocomputing journal.

1 Introduction

In recent decades, robot manipulators play an important role in the field of flexible automation. They have been extensively used in various applications, in which many tasks require high-speed and high-precision trajectory tracking. However, robotic manipulators generally faced many uncertainties and external disturbances in their dynamics, such as payload variations, frictions, external disturbances, and sensor noises, etc. It is difficult to set up exact mathematical models of robotic manipulators for the design of model-based control systems. To deal with parameter uncertainties, many control approaches that attenuate the impact of robotic uncertainties have been proposed such as proportional-integral-derivative (PID) control [1], robust control [2-3], adaptive control [4-5], sliding mode control [6-17], and neural network control [18-26].

So far, sliding mode control (SMC) is one of the influential nonlinear control methods that have been widely applied to control for both certain and uncertain systems [6-8]. In order to design sliding mode control systems, the establishment of suitable sliding surfaces to ensure the desired dynamics is considered first, and then a sliding mode controller is designed to drive the states of the system on the sliding surfaces. Moreover, the main characteristic of SMC is to use discontinuous control effort to guarantee the occurrence of the sliding surface ($s=0$) which is not affected by any modeling uncertainties and external disturbances. However, the traditional SMC can only guarantee the asymptotic stability because a linear surface is used [9]. Although the parameters of the linear sliding mode can be adjusted to make faster the convergence time, the system states cannot converge to zero in finite time. Recently, terminal sliding mode control (TSMC) methods have been proposed and developed in [10-15] to achieve finite time convergence, high precision and strong robustness with respect to uncertainties. By introducing a nonlinear sliding surface instead of linear hyper-planes and suitably designing the controller based on that surface, the tracking errors and the discontinuous control gains of TSMC can be significantly reduced in comparison with those of the traditional SMC. However, for dealing large structured and unstructured uncertainties along with external disturbances, a large gain of the switching controls still must be used; this leads to low control accuracy, high wear of moving mechanical parts and much damages in the robot joints. To solve this problem, the most common approach is boundary layer method. This method defines a boundary layer around the sliding surface and then approximates the discontinuous control (sign function) to get states stayed inside this boundary layer [16]. However, there is a trade-off between chattering elimination and tracking performance, a thicker boundary layer can eliminate the chattering phenomenon but the tracking error will be increased. In addition, for choosing the value of switching control gain, most of the aforementioned works require the prior knowledge of the upper bounds of the uncertainty and external disturbance; it is, however, very hard requirement to achieve.

Artificial neural networks [20-23] (ANNs) are one of the most powerful tools for controlling many complex dynamic systems because of the ability to learn and approximate any arbitrary nonlinear function. Radial basis function neural network (RBFNN), known as a candidate of neural networks, has several important advantages such as simplicity of its structure, fast learning and better approximation capabilities. It has universal approximation properties and can avoid the local minima problem; it can not only reduce the number of the tuning parameters of neural network but also can make the initialization much easier. Due to these advantages of the RBFNNs some terminal sliding mode control combined with the RBFNNs for robot manipulators have been published recently [24-26]. In general, the neural network control techniques can be designed in two steps. First, neural networks are used to approximate the unknown dynamic model of a system or used to replace the discontinuous control in conventional TSMC. Then, when a sufficiently accurate approximation is achieved, a proper control strategy using this approximation can be established. Although, this method could work well for many systems, it does not have any built-in capability to handle changes in the system. In some recent works [27-30], neural networks have been successfully used to approximate individual element of $\hat{M}(q)$, $\hat{C}(q,\dot{q})$, and $\hat{G}(q)$ of the complex and unknown nonlinear dynamic functions of robot manipulators. As a result, the controllers can improve their robustness to parameter uncertainties and model changes.

In this paper, inspired by Ge's work [27], an adaptive terminal sliding mode control based on local approximation method is proposed for trajectory tracking of uncertain robotic manipulators. The main advantages of the control scheme are listed as follows:

(1) The proposed controller can effectively control the unknown nonlinear dynamic system with robustness to parameter uncertainties and model changes,

(2) RBFNNs with parameter adaptive laws are used to approximate individual elements of the inertia matrix, centripetal matrix and gravitation vector, respectively. Moreover, an error estimator is used to compensate the approximation error for improving the control performance and dealing with large structured and unstructured uncertainties and external disturbances,

(3) An adaptive finite-time control law with an appropriate update law is designed to drive the system states to reach the sliding surface and to converge to zero in finite time,

(4) The control scheme does not need the prior knowledge of the upper bound of system uncertainties,

(5) Finite time convergence and stability of the closed loop system can be guaranteed by Lyapunov theory,

(6) GL matrix and its product operators are used to make the stability analysis a lot easier,

The remainder of this paper is arranged as follows. The radial basis function neural network, the definitions for GL matrix and its product operators, and the dynamic model of the robotic manipulator are described in Section 2. In Section 3, the structure of terminal sliding mode neural networks controller is presented and a stability analysis is performed. In Section 4, simulation results for a two-link robot manipulator are provided to verify the effectiveness of the proposed controller. Finally, some concluding remarks are given in Section 5.

2 System Dynamics and Preliminaries

2.1 RBFNN model

In the field of control engineering, neural network is usually used to approximate a given nonlinear function $f(t)$ up to threshold error tolerance ε. Gaussian RBF neural network, known as a candidate of neural networks, have been quite successful in representing the complex nonlinear function. It has been proven that any continuous functions, not necessary infinitely smooth, can be uniformly approximated by a linear combination of Gaussians [35]. In addition, in comparison with multilayer perceptron neural network, the RBFNN has a simple structure and fast learning, good approximation capability for arbitrary smooth nonlinear functions. As shown in Fig. 1, RBFNN consists of three layers (input, output and hidden layer).

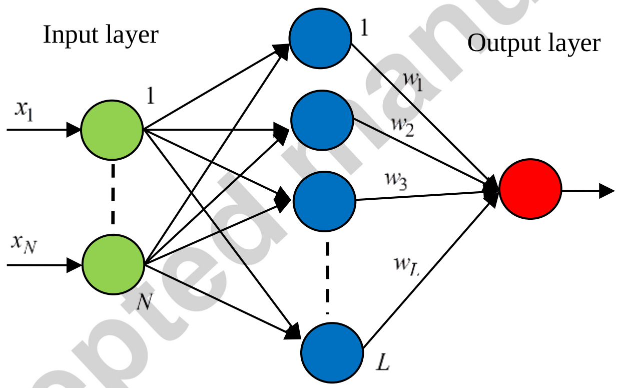


Fig. 1. Structure of the RBF neural network

The input layer: The input layer is simply a fan-out layer and the input vector of the neural network is denoted by:

$$x = [x_1, x_2, ..., x_N]^T \tag{1}$$

The hidden layer: The activation functions in the hidden layer are chosen as Gaussian function.

$$\sigma_i(x) = \exp(-\frac{(x-c_i)^2}{2b_i^2}), \quad i = 1, 2, ..., L \tag{2}$$

where x is an N-dimensional input vector, c_i is the center vector with the same dimension as x, b_i is the variance of the basic function, L is the number of hidden units in the hidden layer.

The output layer: The output of the RBFNN can be computed by the weighted sum method and given by:

$$y = \sum_{i=0}^{L} w_i \sigma_i(x) + \varepsilon \tag{3}$$

where w_i is the weight connecting hidden layer and output layer, ε is the threshold offset of output neuron.

2.2 GL matrix and operators

In this subsection, the GL matrix and its product operators [27, 28] are briefly presented here for the analysis of neural network. To avoid any possible confusion, $[.]$ is used to denote the ordinary vector or matrix, while $\{.\}$ denote the GL vectors and matrices, and denote the GL product operators by "$\bullet$".

Let I_0 is the set of integers, $\theta_{kj}, \xi_{kj} \in \mathbf{R}^{n_{kj}}$, where $n_{kj} \in I_0$ ($k = 1,2,...n;\ j = 1,2,...m$). The GL row vector $\{\theta_k\}$ and its transpose $\{\theta_k\}^T$ are defined in the following form:

$$\begin{aligned} \{\theta_k\} &= \{\theta_{k1} \quad \theta_{k2} \quad \cdots \quad \theta_{km}\} \\ \{\theta_k\}^T &= \{\theta_{k1}^T \quad \theta_{k2}^T \quad \cdots \quad \theta_{km}^T\} \end{aligned} \tag{4}$$

The GL matrix $\{\Theta\}$ and its transpose $\{\Theta\}^T$ are defined as

$$\{\Theta\} = \begin{Bmatrix} \theta_{11} & \theta_{12} & \cdots & \theta_{1m} \\ \theta_{21} & \theta_{22} & \cdots & \theta_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{n1} & \theta_{n2} & \cdots & \theta_{nm} \end{Bmatrix} = \begin{Bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{Bmatrix} \tag{5}$$

$$\{\Theta\}^T = \begin{Bmatrix} \theta_{11}^T & \theta_{12}^T & \cdots & \theta_{1m}^T \\ \theta_{21}^T & \theta_{22}^T & \cdots & \theta_{2m}^T \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{n1}^T & \theta_{n2}^T & \cdots & \theta_{nm}^T \end{Bmatrix} \tag{6}$$

It can be seen that the transpose of its elementary matrix is obtained to be the transpose of its elementary vector locally.

For a given GL matrix $\{\mathrm{E}\}$

$$\{\mathrm{E}\} = \begin{Bmatrix} \xi_{11} & \xi_{12} & \cdots & \xi_{1m} \\ \xi_{21} & \xi_{22} & \cdots & \xi_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \xi_{n1} & \xi_{n2} & \cdots & \xi_{nm} \end{Bmatrix} = \begin{Bmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_n \end{Bmatrix} \tag{7}$$

The GL product of $\{\Theta\}^T$ and $\{\mathrm{E}\}$ is a $n \times m$ matrix of element-wise product defined as

$$\left[\{\Theta\}^T \bullet \{\mathrm{E}\}\right] = \begin{bmatrix} \theta_{11}^T \xi_{11} & \theta_{12}^T \xi_{12} & \cdots & \theta_{1m}^T \xi_{1m} \\ \theta_{21}^T \xi_{21} & \theta_{22}^T \xi_{22} & \cdots & \theta_{2m}^T \xi_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{n1}^T \xi_{n1} & \theta_{n2}^T \xi_{n2} & \cdots & \theta_{nm}^T \xi_{nm} \end{bmatrix} \in \mathbf{R}^{n\times m} \tag{8}$$

The GL product of a square matrix and a GL row vector is defined as follows

$$\Gamma_k \bullet \{\xi_k\} = \{\Gamma_k\} \bullet \{\xi_k\} = \left[r_{k1}\xi_{k1} \quad r_{k2}\xi_{k2} \quad \cdots \quad r_{kn}\xi_{kn}\right] \tag{9}$$

where $\Gamma_k = \Gamma_k^T = \left[r_{k1} \quad r_{k2} \quad \cdots \quad r_{kn}\right]$, $r_{k1} \in \mathbf{R}^{m\times n_j}$, $m = \sum_{j=1}^{n} n_j$.

Note that the GL product should be computed first in a mixed matrix product. For instance, in $\{A\} \bullet \{B\} C$, the matrix $\left[\{A\} \bullet \{B\}\right]$ should be computed first, and then followed by the multiplication of $\left[\{A\} \bullet \{B\}\right]$ with matrix C.

2.3 RBF neural network modeling of robot manipulators

The dynamics of a serial n-links robot manipulator can be written as [31]

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + \tau_d = \tau \tag{10}$$

where $q(t), \dot{q}(t), \ddot{q}(t) \in \mathbf{R}^n$ are the vector of joint positions, velocities and accelerations, respectively. $M(q) \in \mathbf{R}^{n\times n}$ is the inertial matrix, $C(q,\dot{q}) \in \mathbf{R}^{n\times n}$ expresses the centripetal and Coriolis matrix, $G(q) \in \mathbf{R}^n$ represents the gravity torques vector, $\tau \in \mathbf{R}^n$ is the control torque, and $\tau_d \in \mathbf{R}^n$ is the bounded external disturbance vector.

For convenience, the above dynamic equation has the following useful structural properties;

Property 1: $M(q)$ is a symmetric positive definite matrix.

$$m_1\|x\|^2 \le x^T M(q) x \le m_2 \|x\|^2 ; \forall x \in \mathbf{R}^n \tag{11}$$

where m_1, m_2 are known positive scalar constants, $x \in \mathbf{R}^n$ is a vector, $\| \|$ denotes the Euclidean vector norm.

Property 2: $M(q) - 2C(q,\dot{q})$ is a skew symmetric matrix.

$$x^T[\dot{M}(q) - 2C(q,\dot{q})]x = 0 \qquad (12)$$

for any vector $x \in \mathbf{R}^n$.

It is observed that both $M(q)$ and $G(q)$ are functions of q only, and RBF neural network are sufficient to model them [27]. The neural network approximations for the elements $M_{ij}(q)$ of $M(q)$ and $G_i(q)$ of $G(q)$ can be written as follows

$$M_{ij}(q) = W_{Mij}^T \sigma_{Mij}(q) + \varepsilon_{Mij}(q) \qquad (13)$$

$$G_i(q) = W_{Gi}^T \sigma_{Gi}(q) + \varepsilon_{Gi}(q) \qquad (14)$$

where W_{Mij}, W_{Gi} are the vectors of neural network weights, $\sigma_{Mij}(q)$, $\sigma_{Gi}(q)$ are the vectors of activation functions with input vector q, $\varepsilon_{Mij}(q), \varepsilon_{Gi}(q) \in \mathbf{R}$ are the modeling errors of $M_{ij}(q)$ and $G_i(q)$, respectively, which are assumed to be bounded. On the other hand, $C(q,\dot{q})$ is a matrix of q and $\dot{q}$, RBF neural network of q and $\dot{q}$ is needed to model it. Assume that $C_{ij}(q,\dot{q})$ can be modeled as

$$C_{ij}(q,\dot{q}) = W_{Cij}^T \sigma_{Cij}(q,\dot{q}) + \varepsilon_{Cij}(q,\dot{q}) \qquad (15)$$

where W_{Cij} is the vector of neural network weights, $\sigma_{Cij}(q,\dot{q})$ is the vector of activation function with input vector q and $\dot{q}$, $\varepsilon_{Cij}(q,\dot{q}) \in \mathbf{R}$ is the modeling error of $C_{ij}(q,\dot{q})$, which are also assumed to be bounded.

Thus, $M(q)$, $C(q,\dot{q})$, and $G(q)$ can be expressed as follows

$$M(q) = \begin{bmatrix} W_{M11}^T \sigma_{M11}(q) & W_{M12}^T \sigma_{M12}(q) & \cdots & W_{M1n}^T \sigma_{M1n}(q) \\ W_{M21}^T \sigma_{M21}(q) & W_{M22}^T \sigma_{M22}(q) & \cdots & W_{M2n}^T \sigma_{M2n}(q) \\ \vdots & \vdots & \ddots & \vdots \\ W_{Mn1}^T \sigma_{Mn1}(q) & W_{Mn2}^T \sigma_{Mn2}(q) & \cdots & W_{Mnn}^T \sigma_{Mnn}(q) \end{bmatrix} + E_M(q) \qquad (16)$$

$$C(q,\dot{q}) = \begin{bmatrix} W_{C11}^T \sigma_{C11}(q,\dot{q}) & W_{C12}^T \sigma_{C12}(q,\dot{q}) & \cdots & W_{C1n}^T \sigma_{C1n}(q,\dot{q}) \\ W_{C21}^T \sigma_{C21}(q,\dot{q}) & W_{C22}^T \sigma_{C22}(q,\dot{q}) & \cdots & W_{C2n}^T \sigma_{C2n}(q,\dot{q}) \\ \vdots & \vdots & \ddots & \vdots \\ W_{Cn1}^T \sigma_{Cn1}(q,\dot{q}) & W_{Cn2}^T \sigma_{Cn2}(q,\dot{q}) & \cdots & W_{Cnn}^T \sigma_{Cnn}(q,\dot{q}) \end{bmatrix} + E_C(q,\dot{q}) \qquad (17)$$

$$G(q)=\begin{bmatrix} W_{G1}^T\sigma_{G1}(q) \\ W_{G2}^T\sigma_{G2}(q) \\ \vdots \\ W_{Gn}^T\sigma_{Gn}(q) \end{bmatrix}+E_G(q) \tag{18}$$

where $E_M(q), E_C(q,\dot{q})\in\mathbf{R}^{n\times n}$ are the neural network modeling errors matrices with their elements being the $\varepsilon_{Mij}(q)$ and $\varepsilon_{Cij}(q,\dot{q})$, respectively. $E_G(q)\in\mathbf{R}^n$ is the modeling error vector with their elements being $\varepsilon_{Gi}(q)$.

Using the GL matrix and its product operators introduced in Section 2.2, the RBF neural network approximations for $M(q)$, $C(q,\dot{q})$, and $G(q)$ can be further written as follows,

$$M(q)=\left[\{W_M\}^T\bullet\{\sigma_M(q)\}\right]+E_M(q) \tag{19}$$

$$C(q,\dot{q})=\left[\{W_C\}^T\bullet\{\sigma_C(q,\dot{q})\}\right]+E_C(q,\dot{q}) \tag{20}$$

$$G(q)=\left[\{W_G\}^T\bullet\{\sigma_G(q)\}\right]+E_G(q) \tag{21}$$

where $\{W_M\}\in\mathbf{R}^{n\times n}$, $\{W_C\}\in\mathbf{R}^{n\times n}$, $\{W_G\}\in\mathbf{R}^n$ are the GL matrices and vector with their elements being the neural network weights W_{Mij}, W_{Cij} and W_{Gi}, respectively. $\{\sigma_M\}\in\mathbf{R}^{n\times n}$, $\{\sigma_C\}\in\mathbf{R}^{n\times n}$, $\{\sigma_G\}\in\mathbf{R}^n$ are the GL matrices and vector with their elements being the neural network activation functions σ_{Mij}, σ_{Cij} and σ_{Gi}, respectively.

Using (19)-(21), the dynamic equation given in (10) can be written as

$$\left[\{W_M\}^T\bullet\{\sigma_M(q)\}\right]\ddot{q}+\left[\{W_C\}^T\bullet\{\sigma_C(q,\dot{q})\}\right]\dot{q}+\left[\{W_G\}^T\bullet\{\sigma_G(q)\}\right]+E+\tau_d=\tau \tag{22}$$

where $E=E_M(q)\ddot{q}_s+E_C(q,\dot{q})\dot{q}_s+E_G(q)$

Assumption 1: The term $E+\tau_d$ is bounded by

$$\left|E+\tau_d\right|\le d_0 \tag{23}$$

where d_0 are unknown positive constants.

Assumption 2: The desired trajectory $q_d(t)\in\mathbf{R}^n$ is a twice continuously differentiable function in terms of t.

Lemma 1 (see [33]). Assume that a continuous positive definite function $V(t)$ satisfies the differential inequality

$$\dot{V}(t)\le\lambda V^{\varphi},\ \forall t\ge t_0,\ V(t_0)\ge 0 \tag{24}$$

where $\lambda > 0,\ 0 < \varphi < 1$ are constants. Then for any given t_0, $V(t)$ satisfies the inequality

$$V^{1-\varphi}(t) \le V^{1-\varphi}(t_0) - \lambda(1-\varphi)(t-t_0),\ t_0 < t < t_1 \tag{25}$$

And $V(t) = 0,\ \forall t \ge t_1$, with t_1 given by

$$t_1 = t_0 + \frac{1}{\lambda(1-\varphi)} V^{1-\varphi}(t_0) \tag{26}$$

Lemma 2 (see [34]). Jensen's inequality

$$\left(\sum_{i=1}^{m} a_i^{b_2}\right)^{1/b_2} \le \left(\sum_{i=1}^{m} a_i^{b_1}\right)^{1/b_1},\ 0 < b_1 < b_2 \tag{27}$$

With $a_i \ge 0,\ 1 \le i \le m$.

The control objective of this paper is to design a stable control law to ensure that the tracking error between joint position vector q and desired joint position vector q_d converge to zero in finite time.

3 Controller Design

In this section, an adaptive finite-time controller is proposed for the nonlinear robot dynamic system (10). First, the controller structure is developed based on terminal sliding mode control. In the control scheme, RBFNNs are employed to approximate for the elements $M_{ij}(q)$, $C_{ij}(q,\dot{q})$, and $G_{ij}(q)$ of $M(q)$, $C(q,\dot{q})$, and $G(q)$, respectively. Meanwhile, an adaptive finite-time control law with appropriate update law is designed to drive the system states to reach the sliding surface and to converge to zero in finite time. Then, the stability and finite time convergence of the closed-loop system are strictly proven.

3.1 Controller structure

In order to apply the terminal sliding mode control, it is necessary to define the terminal sliding surface $s(t)$ for n-link robot manipulator as

$$s = \dot{e} + \beta sig(e)^{\varphi} \tag{28}$$

where $\beta = diag(\beta_1, \beta_2, ..., \beta_n)$, $\beta_1, \beta_2, ...\beta_n$ are positive constants, $0 < \varphi < 1$, $sig(e)^{\varphi} = \left(|e_1|^{\varphi} sign(e_1), |e_2|^{\varphi} sign(e_2), ..., |e_n|^{\varphi} sign(e_n)\right)$, $s = [s_1, s_2, ..., s_n]$, $e(t) = q_d(t) - q(t)$, $\dot{e}(t) = \dot{q}_d(t) - \dot{q}(t)$.

According to the sliding mode design procedure, the control input u consists of the components

$$\tau = u_{eq} - K_{SW} sign(s) \tag{29}$$

where $K_{SW} = diag(k_{SW1}, k_{SW2}, ..., k_{SWn})$, $k_{SW1}, k_{SW2}, ...k_{SWn}$ are positive constants. The main feature of the SMC is that it uses a high speed switching control term to drive the system states toward the sliding surface from any initial state condition. As soon as the system state hit the sliding surface, the equivalent control is applied to keeps the trajectory of the system state evolving on the sliding surface. The equivalent control can be interpreted as the continuous control law that is obtained by equation $\dot{s} = 0$ for nominal system in the absence of the uncertainties and external disturbances.

$$\dot{s} = \ddot{e} + \varphi\beta |e|^{\varphi-1} \dot{e} \tag{30}$$

From (10), the $\ddot{e}$ is given by

$$\ddot{e} = \ddot{q}_d - \ddot{q} = \ddot{q}_d - \frac{\tau - \tau_d - C(q,\dot{q})\dot{q} - G(q)}{M(q)} \tag{31}$$

Multiplying both sides of equation (30) by $M(q)$ and substituting (31) into it yields

$$M(q)\dot{s} = M(q)\ddot{q}_d + C(q,\dot{q})\dot{q} + G(q) + \varphi\beta M |e|^{\varphi-1} \dot{e} - \tau + \tau_d \tag{32}$$

Define that $\dot{q}_s = s + \dot{q}$, then $\ddot{q}_s = \dot{s} + \ddot{q}$, $\dot{q}_s = \dot{q}_d + \beta sig(e)^{\varphi}$, $\ddot{q}_s = \ddot{q}_d + \varphi\beta |e|^{\varphi-1} \dot{e}$. From (32), we have

$$M(q)\dot{s} = -C(q,\dot{q})s - \tau + \tau_d + M(q)\ddot{q}_s + C(q,\dot{q})\dot{q}_s + G(q) \tag{33}$$

If the nonlinear robot dynamic functions $M(q)$, $C(q,\dot{q})$, $G(q)$ are clearly known, then the equivalent control can be defined as

$$u_{eq} = M(q)\ddot{q}_s + C(q,\dot{q})\dot{q}_s + G(q) + Ks \tag{34}$$

where $K = diag(k_1, k_2, ..., k_n)$, $k_1, k_2, ...k_n$ are positive constants.

The stability of the close loop system (29) can be easily proved by Lyapunov theory if the gains of the switching controller are bigger than the upper bounds of uncertainties. Unfortunately, robot manipulators are complex nonlinear systems which involve many uncertainties such as friction, external disturbances, changing payload, sensor noise etc. These model uncertainties may decrease significantly the performance of this control method. Therefore, it is clear that we will adopt RBF neural network to approximate nonlinear dynamic model of the robot.

Substituting Eqs. (19), (20) and (21) into Eq. (33) yields

$$
\begin{aligned}
M(q)\dot{s} &= -C(q,\dot{q})s - \tau + \tau_d + \\
&+\left[\{W_M\}^T \bullet \{\sigma_M(q)\}\right]\ddot{q}_s + \left[\{W_C\}^T \bullet \{\sigma_C(q,\dot{q})\}\right]\dot{q}_s + \left[\{W_G\}^T \bullet \{\sigma_G(q)\}\right] + E
\end{aligned} \tag{35}
$$

Let $\hat{M}(q)$, $\hat{C}(q,\dot{q})$, and $\hat{G}(q)$ be the estimates of $M(q)$, $C(q,\dot{q})$, and $G(q)$, respectively, obtained by replacing the unknown constant neural network weights $\{W_M\}$, $\{W_C\}$, and $\{W_G\}$ in (19)-(21) by their estimates $\{\hat{W}_M\}$, $\{\hat{W}_C\}$, and $\{\hat{W}_G\}$.

For the system (35), the proposed controller is expressed by the following equation

$$
\begin{aligned}
\tau &= \tau_0 + K_I \int sdt + K_p s + \hat{d}_0 sign(s) \\
&= \hat{M}(q)\ddot{q}_s + \hat{C}(q,\dot{q})\dot{q}_s + \hat{G}(q) + K_I \int sdt + K_p s + \hat{d}_0 sign(s) \\
&= \left[\{\hat{W}_M\}^T \bullet \{\sigma_M(q)\}\right]\ddot{q}_s + \left[\{\hat{W}_C\}^T \bullet \{\sigma_C(q,\dot{q})\}\right]\dot{q}_s + \left[\{\hat{W}_G\}^T \bullet \{\sigma_G(q)\}\right] + \\
&\qquad + K_I \int sdt + K_p s + \hat{d}_0 sign(s)
\end{aligned} \tag{36}
$$

where $K_P = diag(k_{P1}, k_{P2}, \ldots, k_{Pn})$, $K_I = diag(k_{I1}, k_{I2}, \ldots, k_{In})$ with $k_{Pi}, k_{Ii} > 0$. $\tau_0 = \left[\{\hat{W}_M\}^T \bullet \{\sigma_M(q)\}\right]\ddot{q}_s + \left[\{\hat{W}_C\}^T \bullet \{\sigma_C(q,\dot{q})\}\right]\dot{q}_s + \left[\{\hat{W}_G\}^T \bullet \{\sigma_G(q)\}\right]$ is the model-estimated control law. Since the output of the neural network is not able to approximate τ_0 accurately, the error estimator $f_{est} = K_I \int sdt$ is used to attenuate the approximation errors. The auxiliary control $K_p s + \hat{d}_0 sign(s)$ is used to guarantee the occurrence of the sliding motion in finite time without the prior knowledge of the upper boundary of the uncertainties and extended disturbances ($\hat{d}_0$ is the estimation of the unknown upper bound d_0).

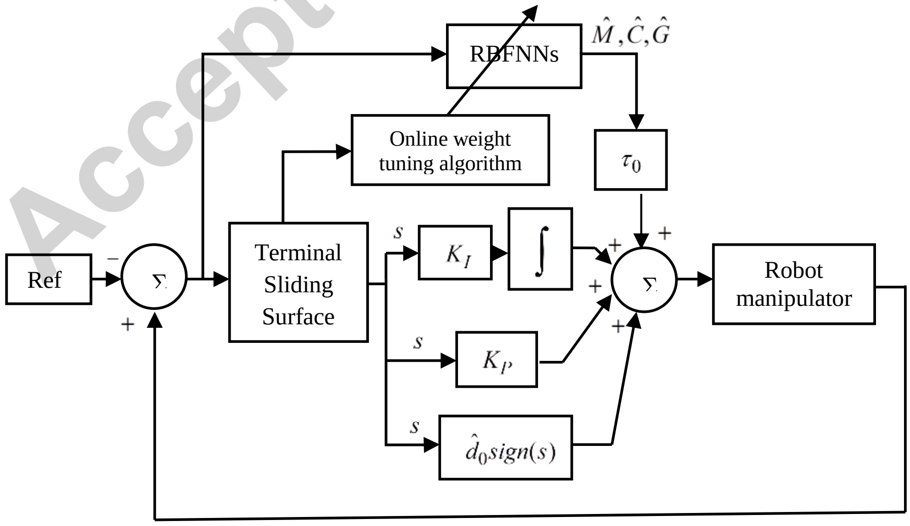


Fig. 2. Block diagram of the proposed controller

3.2 Stability analysis

Theorem 1: Consider the n–link robot manipulator (10), if the terminal sliding mode is chosen as (28), the controller is (36), and the neural network weight vectors are designed as in (37)-(39), the update law of robust term is designed as (40), all the signals of the closed loop system are bounded and the tracking errors converge to zero in finite time.

$$\dot{\hat{W}}_{Mk} = \Gamma_{Mk} \cdot \{\sigma_{Mk}(q)\} \ddot{q}_s s_k \tag{37}$$

$$\dot{\hat{W}}_{Ck} = \Gamma_{Ck} \cdot \{\sigma_{Ck}(q, \dot{q})\} \dot{q}_s s_k \tag{38}$$

$$\dot{\hat{W}}_{Gk} = \Gamma_{Gk} \cdot \{\sigma_{Gk}(q)\} s_k \tag{39}$$

$$\dot{\hat{d}}_0 = \kappa_0 |s| \tag{40}$$

where $k = 1, 2, ..., n$. Γ_{Mk}, Γ_{Ck}, and Γ_{Gk} are constant symmetric positive definite matrices. $\hat{W}_{Mk}$, $\hat{W}_{Ck}$, and $\hat{W}_{Gk}$ are column vectors with their elements being $\hat{W}_{Mkj}$, $\hat{W}_{Ckj}$, $\hat{W}_{Gkj}$, respectively.

Proof: Consider a Lyapunov function candidate as follows

$$\begin{aligned} V = {} & \frac{1}{2} s^T M s + \frac{1}{2} \left(\int_0^t s d\xi \right)^T K_I \left(\int_0^t s d\xi \right) + \frac{1}{2} \sum_{k=1}^{n} \tilde{W}_{Mk}^T \Gamma_{Mk}^{-1} \tilde{W}_{Mk} + \frac{1}{2} \sum_{k=1}^{n} \tilde{W}_{Ck}^T \Gamma_{Ck}^{-1} \tilde{W}_{Ck} \\ & + \frac{1}{2} \sum_{k=1}^{n} \tilde{W}_{Gk}^T \Gamma_{Gk}^{-1} \tilde{W}_{Gk} + \frac{1}{2\kappa_0} (\hat{d}_0 - d_0)^2 \end{aligned} \tag{41}$$

where $\tilde{W}_{Mk} = W_{Mk} - \hat{W}_{Mk}$, $\tilde{W}_{Ck} = W_{Ck} - \hat{W}_{Ck}$, $\tilde{W}_{Gk} = W_{Gk} - \hat{W}_{Gk}$.

Differentiating (41) with respect to time yields

$$\begin{aligned} \dot{V} = {} & s^T M \dot{s} + \frac{1}{2} s^T \dot{M} s + s^T K_I \left(\int_0^t s d\xi \right) - \sum_{k=1}^{n} \tilde{W}_{Mk}^T \Gamma_{Mk}^{-1} \dot{\hat{W}}_{Mk} - \sum_{k=1}^{n} \tilde{W}_{Ck}^T \Gamma_{Ck}^{-1} \dot{\hat{W}}_{Ck} \\ & - \sum_{k=1}^{n} \tilde{W}_{Gk}^T \Gamma_{Gk}^{-1} \dot{\hat{W}}_{Gk} + \frac{1}{\kappa_0} (\hat{d}_0 - d_0) \dot{\hat{d}}_0 \end{aligned} \tag{42}$$

Substituting Eq. (35) into Eq. (42), then applying the skew-symmetric property of $\dot{M}(q)-2C(q,\dot{q})$ yields

$$\dot{V}=s^{T}\left(\begin{array}{l}-\tau+\tau_{d}+\left[\{W_{M}\}^{T}\bullet\{\sigma_{M}(q)\}\right]\ddot{q}_{s}+\left[\{W_{C}\}^{T}\bullet\{\sigma_{C}(q,\dot{q})\}\right]\dot{q}_{s}+\\+\left[\{W_{G}\}^{T}\bullet\{\sigma_{G}(q)\}\right]+E+K_{I}\left(\int_{0}^{t}sd\xi\right)\end{array}\right)+ \\ -\sum_{k=1}^{n}\tilde{W}_{Mk}^{T}\Gamma_{Mk}^{-1}\dot{\hat{W}}_{Mk}-\sum_{k=1}^{n}\tilde{W}_{Ck}^{T}\Gamma_{Ck}^{-1}\dot{\hat{W}}_{Ck}-\sum_{k=1}^{n}\tilde{W}_{Gk}^{T}\Gamma_{Gk}^{-1}\dot{\hat{W}}_{Gk}+\frac{1}{\kappa_{0}}(\hat{d}_{0}-d_{0})\dot{\hat{d}}_{0} \tag{43}$$

Substituting the proposed controller (36) into (43) we have

$$\dot{V}=-s^{T}K_{p}s-s^{T}\hat{d}_{0}sign(s)+s^{T}\tau_{d}+\left[\{\tilde{W}_{M}\}^{T}\bullet\{\sigma_{M}(q)\}\right]\ddot{q}_{s}+\left[\{\tilde{W}_{C}\}^{T}\bullet\{\sigma_{C}(q,\dot{q})\}\right]\dot{q}_{s}+ \\ +\left[\{\tilde{W}_{G}\}^{T}\bullet\{\sigma_{G}(q)\}\right]+s^{T}E \\ -\sum_{k=1}^{n}\tilde{W}_{Mk}^{T}\Gamma_{Mk}^{-1}\dot{\hat{W}}_{Mk}-\sum_{k=1}^{n}\tilde{W}_{Ck}^{T}\Gamma_{Ck}^{-1}\dot{\hat{W}}_{Ck}-\sum_{k=1}^{n}\tilde{W}_{Gk}^{T}\Gamma_{Gk}^{-1}\dot{\hat{W}}_{Gk}+\frac{1}{\kappa_{0}}(\hat{d}_{0}-d_{0})\dot{\hat{d}}_{0} \tag{44}$$

Noting that

$$s^{T}\left[\{\tilde{W}_{M}\}^{T}\bullet\{\sigma_{M}(q)\}\right]\ddot{q}_{s}=\begin{bmatrix}s_{1} & s_{2} & \dots & s_{n}\end{bmatrix}\begin{bmatrix}\{\tilde{W}_{M1}\}^{T}\bullet\{\sigma_{M1}(q)\}\ddot{q}_{s}\\ \{\tilde{W}_{M2}\}^{T}\bullet\{\sigma_{M2}(q)\}\ddot{q}_{s}\\ \vdots\\ \{\tilde{W}_{Mn}\}^{T}\bullet\{\sigma_{Mn}(q)\}\ddot{q}_{s}\end{bmatrix} \\ =\sum_{k=1}^{n}\{\tilde{W}_{Mk}\}^{T}\bullet\{\sigma_{Mk}(q)\}\ddot{q}_{s}s_{k} \tag{45}$$

In a similar way, we have

$$s^{T}\left[\{\tilde{W}_{C}\}^{T}\bullet\{\sigma_{C}(q,\dot{q})\}\right]\dot{q}_{s}=\sum_{k=1}^{n}\{\tilde{W}_{Ck}\}^{T}\bullet\{\sigma_{Ck}(q,\dot{q})\}\dot{q}_{s}s_{k} \tag{46}$$

$$s^{T}\left[\{\tilde{W}_{G}\}^{T}\bullet\{\sigma_{C}(q)\}\right]=\sum_{k=1}^{n}\{\tilde{W}_{Gk}\}^{T}\bullet\{\sigma_{Gk}(q)\}s_{k} \tag{47}$$

Thus, Eq. (44) becomes

$$\dot{V} = -s^T K_p s - s^T \hat{d}_0 sign(s) + s^T \tau_d + s^T E + \sum_{k=1}^{n} \left\{\tilde{W}_{Mk}\right\}^T \bullet \{\sigma_{Mk}(q)\} \ddot{q}_s s_k + $$
$$+ \sum_{k=1}^{n} \left\{\tilde{W}_{Ck}\right\}^T \bullet \{\sigma_{Ck}(q,\dot{q})\} \dot{q}_s s_k + \sum_{k=1}^{n} \left\{\tilde{W}_{Gk}\right\}^T \bullet \{\sigma_{Gk}(q)\} s_k \tag{48}$$
$$- \sum_{k=1}^{n} \tilde{W}_{Mk}^T \Gamma_{Mk}^{-1} \dot{\hat{W}}_{Mk} - \sum_{k=1}^{n} \tilde{W}_{Ck}^T \Gamma_{Ck}^{-1} \dot{\hat{W}}_{Ck} - \sum_{k=1}^{n} \tilde{W}_{Gk}^T \Gamma_{Gk}^{-1} \dot{\hat{W}}_{Gk} + \frac{1}{\kappa_0}(\hat{d}_0 - d_0)\dot{\hat{d}}_0$$

Substituting adaptive law (37)-(40) into (48) we obtain

$$\begin{aligned} \dot{V} &= -s^T K_p s - s^T \hat{d}_0 sign(s) + s^T \tau_d + s^T E + (\hat{d}_0 - d_0)|s| \\ &= -s^T K_p s + s^T (E + \tau_d) - d_0 |s| \\ &\leq s^T K_p s \leq 0 \end{aligned} \tag{49}$$

Therefore, according to Lyapunov theorem, inequality (49) obtains that $s, \tilde{W}_{Mk}, \tilde{W}_{Ck}, \tilde{W}_{Gk}$, and $\hat{d}_0$ are bounded. Meanwhile, considering (28), it can be achieved that both e and $\dot{e}$ are bounded. In addition, since q_d is bounded as specified, q is bounded as well. As a result, all signals of the closed loop system (35) are bounded.

The estimation values $\hat{d}_0$ is bounded, that is, there exist positive constants d_0^* such that $\hat{d}_0 \leq d_0^*$ for $\forall t \geq 0$. Now it will be proven that the system states in (28) reach the nonsingular terminal sliding surface $s = 0$ within a finite time.

The following Lyapunov function candidate is considered

$$V_1 = \frac{1}{2} s^T M s + \frac{1}{2\gamma_0}(\hat{d}_0 - d_0^*)^2 \tag{50}$$

where γ_0 are positive constants. Differentiating (50) with respect to time and using the same procedure that was used to get (44), we can obtain

$$\dot{V}_1 = -s^T K_p s - s^T \hat{d}_0 sign(s) + s^T \tau_d + s^T E + \frac{\kappa_0}{\gamma_0}(\hat{d}_0 - d_0^*)|s| + $$
$$+ s^T \left(\left[\left\{\tilde{W}_M\right\}^T \bullet \{\sigma_M(q)\} \right] \ddot{q}_s + \left[\left\{\tilde{W}_C\right\}^T \bullet \{\sigma_C(q,\dot{q})\} \right] \dot{q}_s + \left[\left\{\tilde{W}_G\right\}^T \bullet \{\sigma_G(q)\} \right] \right) \tag{51}$$

The Gaussian function is bounded on the interval $[0,1]$; therefore $\{\sigma_M(q)\}$, $\{\sigma_C(q)\}$, $\{\sigma_G(q)\}$ are bounded.

According to the above proof, $\left\{\tilde{W}_M\right\}$, $\left\{\tilde{W}_C\right\}$, $\left\{\tilde{W}_G\right\}$ are bounded; therefore $\left[\left\{\tilde{W}_M\right\}^T \bullet \{\sigma_M(q)\}\right]$, $\left[\left\{\tilde{W}_C\right\}^T \bullet \{\sigma_C(q,\dot{q})\}\right]$, $\left[\left\{\tilde{W}_G\right\}^T \bullet \{\sigma_G(q)\}\right]$ are bounded. It can be obtained that $\left(\left[\{\tilde{W}_M\}^T \bullet \{\sigma_M(q)\}\right]\ddot{q}_s + \left[\{\tilde{W}_C\}^T \bullet \{\sigma_C(q,\dot{q})\}\right]\dot{q}_s + \left[\{\tilde{W}_G\}^T \bullet \{\sigma_G(q)\}\right]\right)$ is bounded. Hence, we can always find positive definite matrix $K_E = diag(k_{E1}, k_{E2}, ..., k_{En}),\ k_{Ei} \geq 0$ satisfying

$$\left(\left[\{\tilde{W}_M\}^T \bullet \{\sigma_M(q)\}\right]\ddot{q}_s + \left[\{\tilde{W}_C\}^T \bullet \{\sigma_C(q,\dot{q})\}\right]\dot{q}_s + \left[\{\tilde{W}_G\}^T \bullet \{\sigma_G(q)\}\right]\right) \leq K_E s \tag{52}$$

Consequently, we can obtain the following:

$$\dot{V}_1 \leq -s^T K_p s - \hat{d}_0|s| + \tau_d|s| + E|s| + \frac{\kappa_0}{\gamma_0}(\hat{d}_0 - d_0^*)|s| + s^T K_E s \tag{53}$$

The control gains K_p are selected to satisfy the following condition

$$K_p \geq K_E \tag{54}$$

Then, we have

$$\dot{V}_1 \leq -\hat{d}_0|s| + \tau_d|s| + E|s| + \frac{\kappa_0}{\gamma_0}(\hat{d}_0 - d_0^*)|s| \tag{55}$$

Substituting Eq. (23) into (55), one has

$$\begin{aligned}
\dot{V}_1 &\leq -\hat{d}_0|s| + d_0|s| + \frac{\kappa_0}{\gamma_0}(\hat{d}_0 - d_0^*)|s| \\
&\leq -\hat{d}_0|s| + d_0|s| - d_0^*|s| + d_0^*|s| + \frac{\kappa_0}{\gamma_0}(\hat{d}_0 - d_0^*)|s| \\
&\leq -(d_0^* - d_0)|s| + (\hat{d}_0 - d_0^*)\left[-|s| + \frac{\kappa_0}{\gamma_0}|s|\right]
\end{aligned} \tag{56}$$

Noting that $\hat{d}_0 \leq d_0^*$, inequality (56) can be rewritten as

$$\dot{V}_1 \leq -(d_0^* - d_0)|s| - \left|\hat{d}_0 - d_0^*\right|\left[-|s| + \frac{\kappa_0}{\gamma_0}|s|\right] \tag{57}$$

We define

$$\begin{aligned}
\lambda_s &= (d_0^* - d_0) \\
\lambda_d &= \left(-|s| + \frac{\kappa_0}{\gamma_0}|s|\right)
\end{aligned} \tag{58}$$

We always choose d_0^* and γ_0 such that $d_0^* \geq d_0$ and $\gamma_0 \leq \kappa_0$, which yields $\lambda_s \geq 0$ and $\lambda_d \geq 0$. Therefore, we obtain

$$\begin{aligned}\dot{V}_1 &\le -\lambda_s|s| - \lambda_d\left|\hat{d}_0 - d_0^*\right| \\ &\le -\frac{\sqrt{2}\lambda_s}{\sqrt{M}}\frac{\left|\sqrt{M}\,s\right|}{\sqrt{2}} - \sqrt{2}\lambda_d\frac{\left|\hat{d}_0 - d_0^*\right|}{\sqrt{2}} \\ &\le -\min\left(\frac{\sqrt{2}\lambda_s}{\sqrt{M}}, \sqrt{2}\lambda_d\right)\left(\frac{\left|\sqrt{M}\,s\right|}{\sqrt{2}} + \frac{\left|\hat{d}_0 - d_0^*\right|}{\sqrt{2}}\right)\end{aligned} \tag{59}$$

Applying Lemma 2 and denoting $\lambda = \min(\frac{\sqrt{2}\lambda_s}{\sqrt{M}}, \sqrt{2}\lambda_d)$ results in the following

$$\dot{V}_1 \le -\lambda\left(\left(\frac{\left|\sqrt{M}\,s\right|}{\sqrt{2}}\right)^2 + \left(\frac{\left|\hat{d}_0 - d_0^*\right|}{\sqrt{2}}\right)^2\right)^{1/2} = -\lambda V_1^{1/2} \tag{60}$$

By using Lemma 1, it is concluded that the system state in (10) will converge to the terminal sliding mode surface $s = 0$ in a finite time $T \le \frac{2V_1^{1/2}(0)}{\lambda}$. This completes the proof.

4 Simulation Results

In order to verify the validity and effectiveness of the proposed method, the performance of the proposed controller is tested via simulation for a two-link planar robotic manipulator as shown in Fig. 3. The simulations are performed in the MATLAB-Simulink environment using ODE 4 solver with a fixed-step size of $10^{-4}\ s$.

The dynamic equation of the two-link robot is described as follows

$$\begin{bmatrix} M_{11}(q) & M_{12}(q) \\ M_{12}(q) & M_{22}(q) \end{bmatrix}\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} C_{11}(q,\dot{q}) & C_{12}(q,\dot{q}) \\ C_{12}(q,\dot{q}) & C_{22}(q,\dot{q}) \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} G_1(q) \\ G_2(q) \end{bmatrix} + \begin{bmatrix} \tau_{d_1} \\ \tau_{d_2} \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} \tag{61}$$

where the inertia matrix $M_{ij}(q)$ is given by

$$\begin{aligned} M_{11}(q) &= (m_1 + m_2)l_1^2 + m_2 l_2^2 + 2m_2 l_1 l_2 \cos(q_2), \\ M_{12}(q) &= m_2 l_2^2 + m_2 l_1 l_2 \cos(q_2), \\ M_{21}(q) &= m_2 l_2^2 + m_2 l_1 l_2 \cos(q_2), \\ M_{22}(q) &= m_2 l_2^2, \end{aligned}$$

the Coriolis and centrifugal matrix $C_{ij}(q,\dot{q})$ is given by

$$C_{11}(q,\dot{q}) = -m_2 l_1 l_2 \sin(q_2)\dot{q}_2,$$
$$C_{12}(q,\dot{q}) = -m_2 l_1 l_2 \sin(q_2)(\dot{q}_1 + \dot{q}_2),$$
$$C_{21}(q,\dot{q}) = m_2 l_1 l_2 \sin(q_2)\dot{q}_1,$$
$$C_{22}(q,\dot{q}) = 0,$$

and the gravity torques vector $G_i(q)$ is given by

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

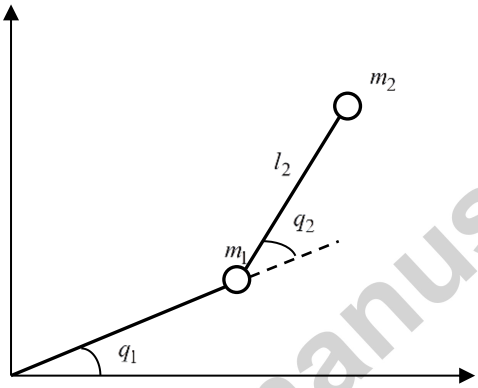


Fig. 3. Two-link robot manipulator model.

The parameters values employed for this simulation are given as of $l_1 = 1$ m, $l_2 = 0.8$ m, and we suppose that we have an uncertainty on masses of the order $\pm 10\%$ (see Fig. 4). The external disturbances are selected as

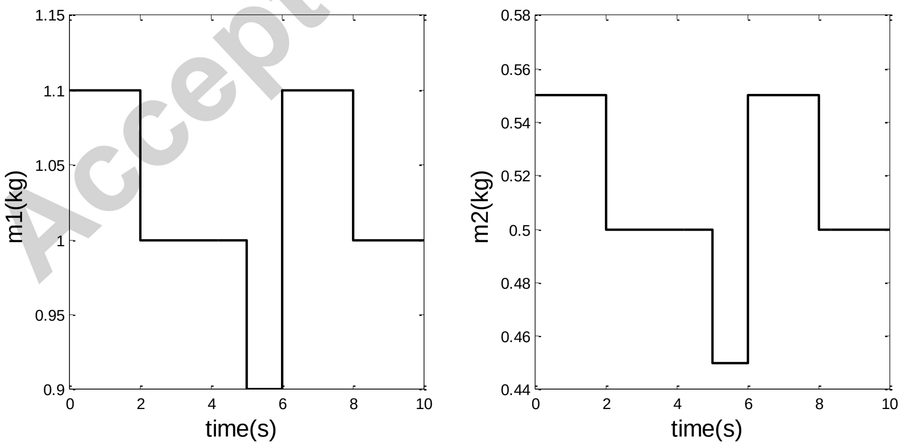


Fig. 4. Variation of mass; m1 and m2 respectively correspond to joints 1 and 2.

$$\tau_d = \begin{bmatrix} \tau_{d_1} \\ \tau_{d_2} \end{bmatrix} = \begin{bmatrix} 2\sin(2t) + \dot{q}_1 \\ 1.5\cos(2t) + 0.7\dot{q}_2 \end{bmatrix} \tag{62}$$

To this end, the Matlab/Simulink is used to perform all simulations, and the sampling time was set to $10^{-4}\ s$. For each element of $M(q)$ and $G(q)$, 21-node RBFNN is used, whereas, for each element of $C(q,\dot{q})$, a 41-node neural network is chosen. The value of c_i are chosen in the interval $-2 < c_i < 2$, and b_i^2 are fixed at 3.

The parameter of the sliding surface (24) are then selected as $\beta = diag(12,12)$, $\varphi = 0.8$, and $\hat{d}_0 = 0.01$.

The simulations are carried out with respect to 2 cases when the end-effector of the robot was driven to check a Hermite polynomial and periodic sinusoid. In order to verify that the proposed controller can effectively control the unknown nonlinear dynamic system with robustness to parameter uncertainties and model changes; two controllers (non-adaptive neural network control and adaptive neural network control) are performed and compared.

4.1 Trajectory Planning

Hermite polynomial trajectory: The design trajectory of the third degree Hermite polynomial [27] is given as

$$q_d(\mathrm{t},\mathrm{t}_d) = q_0 + \left(-2\frac{t^3}{t_d^3} + 3\frac{t^2}{t_d^2} \right)(q_f - q_0) \tag{63}$$

where t_d is the time at which the robot end-effector reaches the desired final position. q_0 and q_f represent the initial and final robot positions end-effector, respectively. The parameters in the simulation are chosen as $\mathrm{t}_d = 1$ s, $q_0 = [0,0]^T$ rad, $q_d(\mathrm{t}_d) = [1,2]^T$ rad.

The control parameters are selected as: $K_P = diag(4,1.5)$.

Periodic sinusoid trajectory: the second desired trajectory chosen as follows

$$\begin{aligned} q_{1d} &= 0.5\sin(\pi t) \\ q_{2d} &= \cos(\pi t) \end{aligned} \tag{64}$$

The initial states are chosen as

$$q_1(0) = 0.4, q_2(0) = -0.5, \dot{q}_1(0) = 0, \dot{q}_2(0) = 0 \tag{65}$$

The control parameters are selected as: $K_P = diag(13,3)$, $K_I = diag(5,2)$.

4.2 Non-Adaptive Neural Network Control

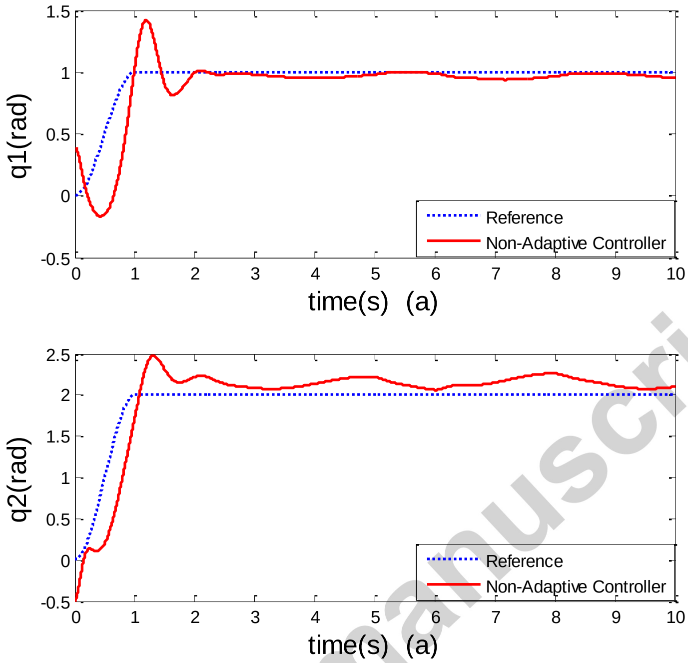


Fig. 5. Position tracking (Hermite polynomial): (a) at joint 1, (b) at joint 2.

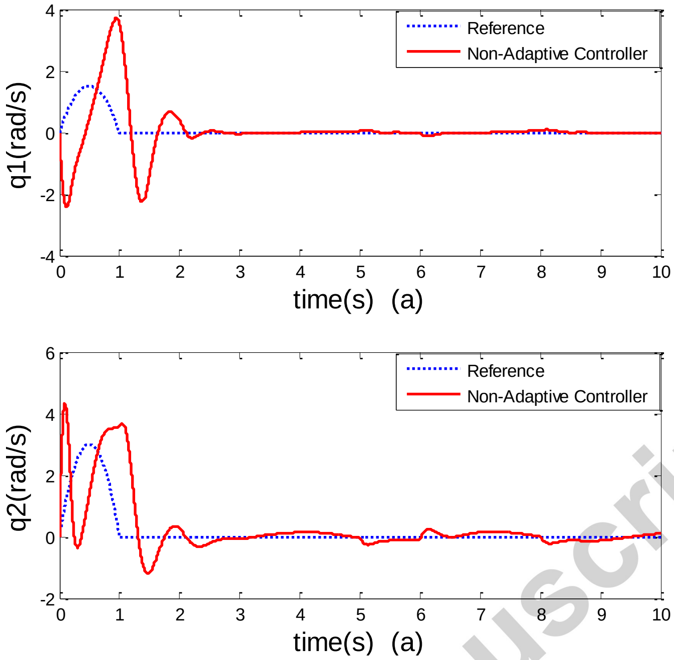


Fig. 6. Velocity tracking (Hermite polynomial): (a) at joint 1, (b) at joint 2.

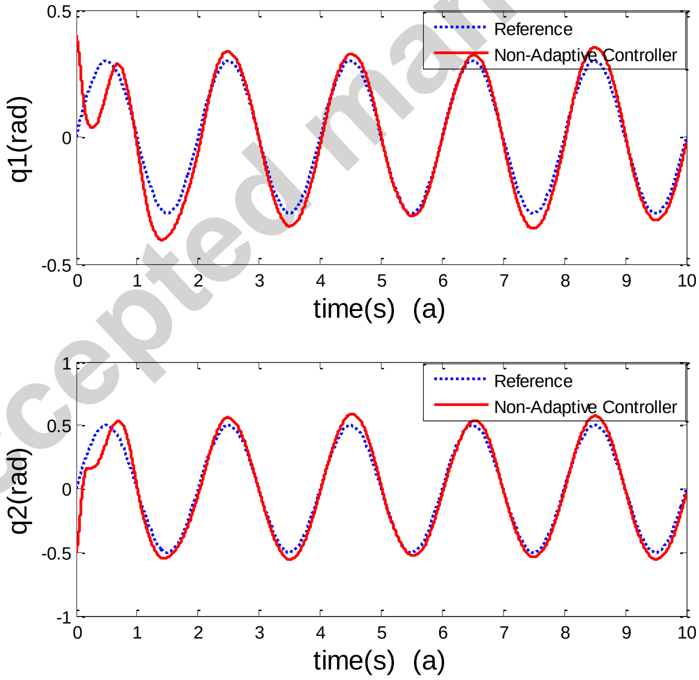


Fig. 7. Position tracking (periodic sinusoid): (a) at joint 1, (b) at joint 2.

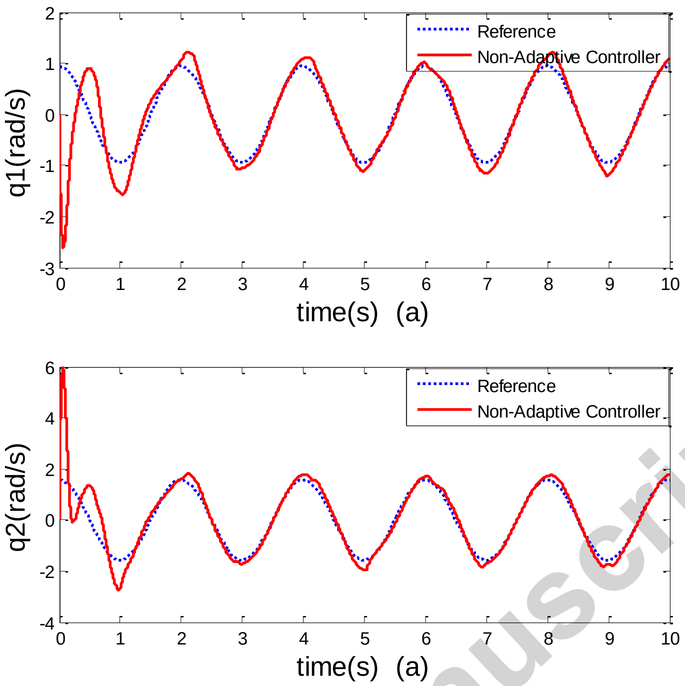


Fig. 8. Velocity tracking (periodic sinusoid): (a) at joint 1, (b) at joint 2.

First, in case 1, we show the control performance when the weight adaptation laws (37)-(39) of the RBF neural networks were not activated. The position and velocity tracking performances of the robot are shown in Figs. 5-6 (when the desired trajectory is a Hermite polynomial) and in Figs. 7-8 (when the desired trajectory is a periodic sinusoid). It can be observed from these figures that the non-adaptive neural network control can't track the desired trajectory well, and can't deal with changes in the system, such as model approximation errors, load disturbances and external disturbances.

4.3 Adaptive Neural Network Control

In the case of adaptive neural network control, the adaptive gains in (37)-(40) are activated as follow, $\Gamma_M = diag(0.05, 0.05)$, $\Gamma_C = diag(0.1, 0.1)$, $\Gamma_G = diag(10, 10)$, $\kappa_0 = 0.1$. In Fig. 9, the estimation of the parameter $\hat{d}_0$ is plotted. It can be seen that the convergence of the online tuning algorithms can be achieved fast enough. The position and velocity tracking performance of the 2 D.O.F robot manipulators are shown in Figs. 10 -13 in both cases of a Hermite polynomial trajectory and a periodic sinusoidal trajectory. It can clearly be observed that the proposed control method can guarantee to make the robot end-effector track the desired trajectory very well; the tracking errors are much smaller than those from the non-adaptive case. Therefore, the simulation results demonstrate that the proposed controller can effectively control the unknown nonlinear dynamic system with robustness to parameter uncertainties and model changes. The norms of $M(q)$ and $\hat{M}(q)$, $C(q,\dot{q})$ and $\hat{C}(q,\dot{q})$, $G(q)$ and $\hat{G}(q)$, are shown in Fig. 14. As can be seen, they are all bounded, which are attractive properties of the proposed controller. In summary, the simulation results demonstrate that the proposed controller can effectively control the unknown nonlinear dynamic system, achieve small tracking errors, and does not need to know the upper bound of any uncertainties and external disturbances.

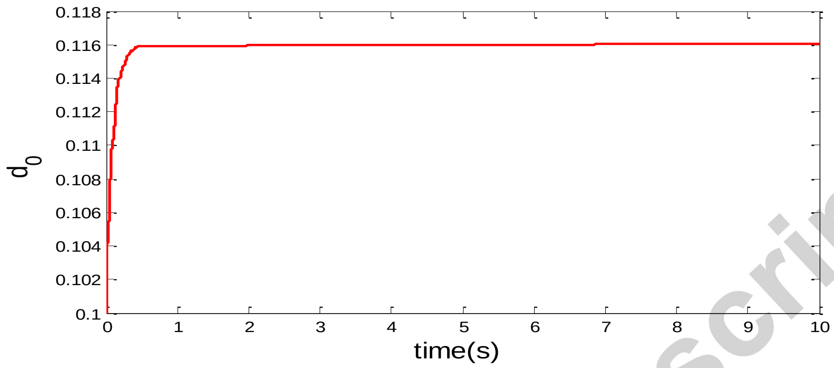


Fig. 9. Time responses of the estimated parameter.

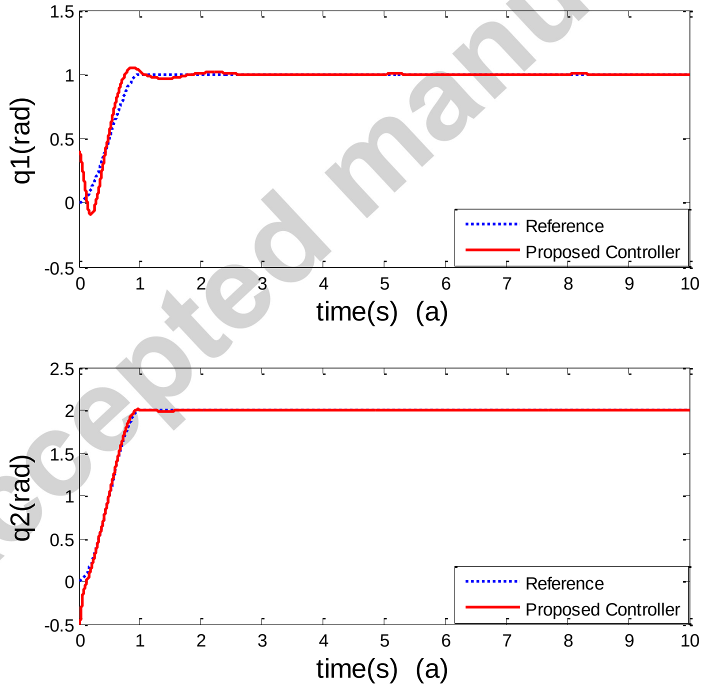


Fig. 10. Position tracking (Hermite polynomial): (a) at joint 1, (b) at joint 2.

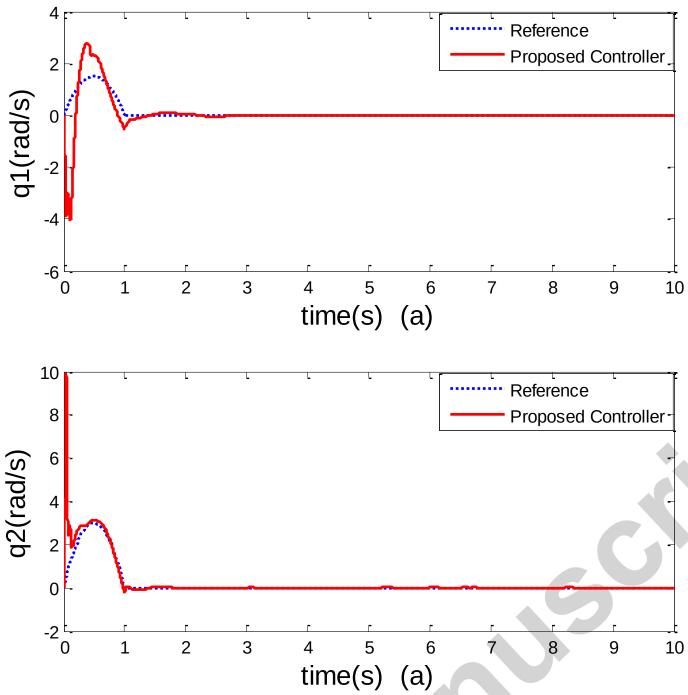


Fig. 11. Velocity tracking (Hermite polynomial): (a) at joint 1, (b) at joint 2.

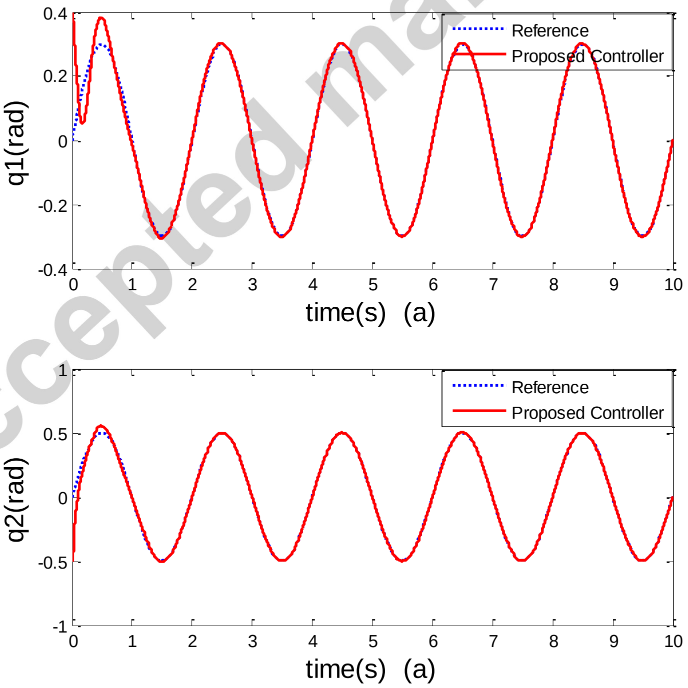


Fig. 12. Position tracking (periodic sinusoid): (a) at joint 1, (b) at joint 2.

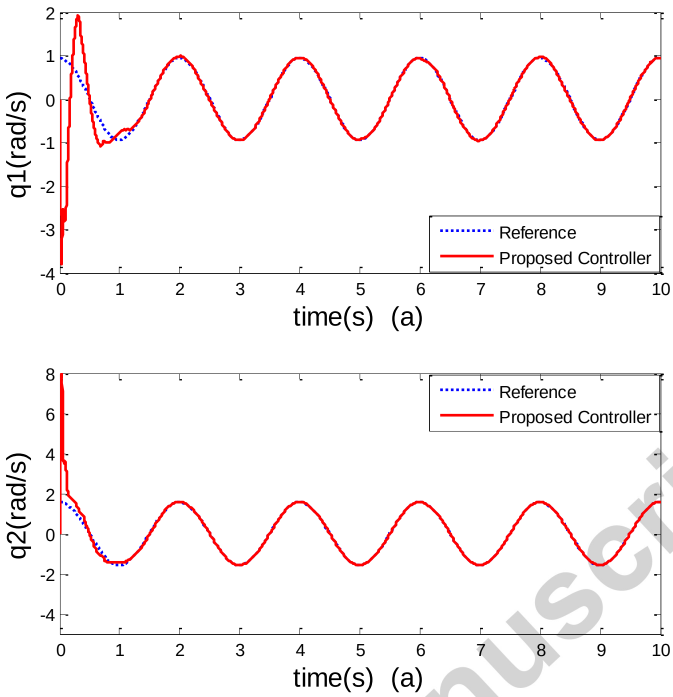


Fig. 13. Velocity tracking (periodic sinusoid): (a) at joint 1, (b) at joint 2.

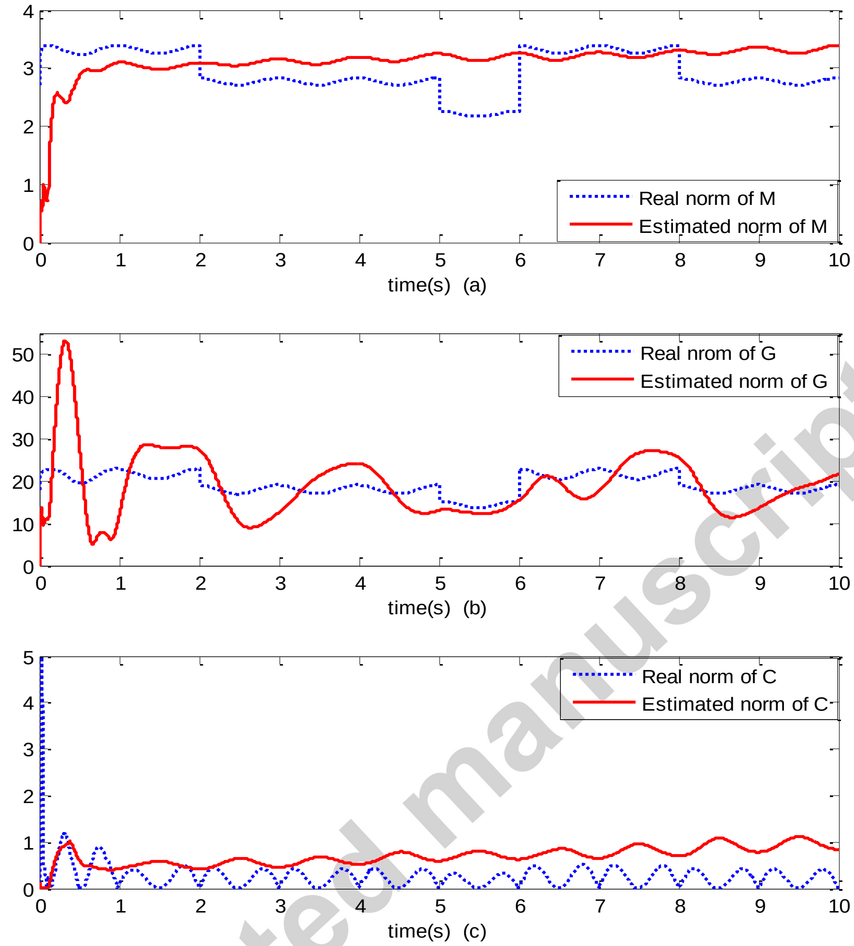


Fig. 14. Norm estimation of $M(q)$, $G(q)$, and $C(q,\dot{q})$: (a) comparison of $\|M(q)\|$ and $\|\hat{M}(q)\|$, (b) comparison of $\|G(q)\|$ and $\|\hat{G}(q)\|$, (c) comparison of $\|C(q,\dot{q})\|$ and $\|\hat{C}(q,\dot{q})\|$.

5 Conclusion

In this paper, a novel adaptive terminal sliding mode control based on local approximation method is proposed for trajectory tracking of uncertain robotic manipulators. By combining the techniques of neural network parameterization, adaptive control, and terminal sliding mode control, the results show the advantages of these methods, such as fast response time, finite time convergence and small tracking errors. The proposed controller can effectively control the unknown nonlinear dynamic system with robustness to parameter uncertainties and model changes. Moreover, it does not need to know the upper bound of any uncertainties and external disturbances. Adaptive learning algorithms have been derived to adjust on-line the output weights in the RBFNNs without any offline training phase. The strict proof of finite time convergence and stability of the closed-loop system have been accomplished. The simulation results of the two-link robotic manipulator system have demonstrated the effectiveness of the proposed method.

Acknowledgments
This work was supported by the Research Fund of University of Ulsan.

6 References

1. S. Arimoto and F.Miyazaki.: Stability and robustness of PID feedback control for robot manipulators of sensory capability. In Robotic Research, M.Brady and R.P. Paul, Eds. Cambridge, MA: MIT Press (1984).

2. M. W. Spong.: On the robust control of robot manipulators. IEEE Trans. Automat. Contr., Vol. 37, No. 11,pp. 1782-1786,1992

3. C. Abdallah, D. M. Dorato and M. Jamishidi.: Survey of The Robust Control Robots. Control System Magazine, Vol. 12, No. 2, 1991.

4. J. J. E. Slotine, and W. Li.: On the adaptive control of robot manipulators. Int. J. Robot. Res.,Vol. 6,No. 3,pp. 49-59,1987.

5. R. Ortega, and M. W. Spong.: Adaptive motion control of rigid robot: a tutorial. Automatica Vol. 25, No. 6, 1989.

6. A. Sabanovic (2011).: Variable structure systems with sliding modes in motion control- A Survey. IEEE Trans. Ind. Electron. 7(2): 212-223.

7. W. Ligang, Y. Rongni, S. Guanghui, Z. Xudong and S. Peng.: New Developments in Sliding Mode Control and Its Applications 2014. Mathematical Problems in Engineering, Vol. 2015, Article ID 783716, 2015.

8. Y. Shtessel, C. Edwards, L. Fridman, A. Levant. : Sliding mode control and observation. Birkhäuser, 2014.

9. V. I. Utkin.: Variable structure systems with sliding modes. IEEE Trans. Automat. Contr. Vol. 22, pp. 212-222, 1997.

10. Z. Man, A. P. Paplinski and H. R. Wu.: A robust MIMO terminal sliding mode control scheme for rigid robotic manipulators. IEEE Trans, Automat. Contr: 1994; 39(12): 2465-2469.

11. M. Chen, Q. X. Wu and R. X. Cui.: Terminal sliding mode control for a class of SISO uncertain nonlinear system. ISA Transactions 2013; 52(2):198-206.

12. Y. Wu, et al. (1998).: Terminal sliding mode control design for uncertain dynamic systems. Syst. Control. Lett. 34(5): p. 281-287.

13. K. B. Park, T. Tsuji (1999).: Terminal sliding mode control of second-order nonlinear uncertain systems. Int. J . Robust. Nonlin. 9(11): 769-780.

14. X. Yu and Z. Man.: Fast terminal sliding-mode control for nonlinear dynamic systems. IEEE Trans. Circuits –I. 2002; 49(2): 261-264.

15. S. Yu, et al.: Continuous finite-time control for robotic manipulators with terminal sliding mode. Automatica 2005 ; 41(11) : 1957-1964.

16. J. J. E. Slotine and W. Li.: Applied Nonlinear Control. EngleWood Cliffs, NJ: Prentice-Hall., pp. 41-190, 1991.

17. Z. Man, O. Mike and X. Yu.: A robust adaptive terminal sliding mode control for rigid robotic manipulators. J. Intell. Robot. Syst. Vol. 24, No. 1, pp. 23-41, 1991.

18. N. Sadati and R. Ghadami.: Adaptive multi-model sliding mode control of robotic manipulators using soft computing. Neurocomputing, Vol. 71, Issues 13–15, pp. 2702-2710, 2008.

19. D. Zhao, W. Ni and Q. Zhu.: A framework of neural networks based consensus control for multiple robotic manipulators. Neurocomputing, Vol. 140, pp. 8-18, 2014.

20. F. Girosi, and T. Poggio .: Networks and the best approximation property. Artificial Intelligence Lab., MIT, Cambridge, MA, Memo, 1164, Oct. 1989.

21. D. S. Huang.: A constructive approach for finding arbitrary roots of polynomials by neural networks. IEEE Transactions on Neural Networks, vol.15, no.2, pp.477-491, 2004.

22. W. Song, J. Xiao, G. Wang, et al.: Dynamic velocity feed-forward compensation control with RBF-NN system identification for industrial robots. Transactions of Tianjin University, 2013, 19: 118-126.

23. D. S. Huang.: Radial basis probabilistic neural networks: Model and application, International Journal of Pattern Recognition and Artificial Intelligence, 13(7), pp.1083-1101, 1999.

24. C. K. Lin.: Nonsingular terminal sliding mode control of robot manipulator using fuzzy wavelet networks. IEEE Trans. Fuzzy Syst. Vol. 14, No. 6, 2006.

25. L. Wang, T. Chai, and L. Zhai.: Neural network-based terminal sliding mode control of robotic manipulators including actuator dynamics. IEEE Trans. Ind. Electron. Vol. 56, No. 9, 2009.

26. Y. Jiang, Q. Wang, and C. Dong.: A reaching law neural network terminal sliding mode guidance law design. 2013 IEEE Region 10 Conference.

27. S. S. Ge and C. C. Hang.: Direct adaptive neural network control of robots. Int. J. Syst. Sci. Vol. 27, pp 533-542.

28. S. S. Ge, T. H. Lee and C. J. Harris.: Adaptive neural network control of robot manipulators. World Scuentific, London, 1998.

29. S. S. Ge, C. C. Hang, and L. C. Woon.: Adaptive neural network control of robot manipulators in task space. IEEE Trans. Ind. Electron. Vol 44, pp 746-752, 1997.

30. L. C. Woon, S. S. Ge, X. Q. Chen and C. Zhang.: Adaptive neural network control of coordinated manipulators. J. Robot. Syst. Vol 16. No. 4, pp 195-211, 1999.

31. Craig, J.J.: Introduction to robotics mechanics and control, 3rd edn.

32. H. K. Khalil.: “Nonlinear systems,” Prentice Hall, 3rd Ed, 2002.

33. Y. Tang.: Terminal sliding mode control for rigid robots. Automatica. Vol. 34. No. 1, pp 51-56, 1998.

34. E. Bechkenback and R. Bell Man.: Inequalities. Springer Verlag Berlin, Heidelberg 1992.

35. F. Girosi and T. Poggio.: Networks and the best approximation proporty. Artificial Intelligence Lab, MIT, Cambridge, MA, Memo, 1989.

Author Biography

Tran, Minh Duc received his B.S. degree in Faculty of Engineering Mechanics and Automation from University of Engineering and Technology, Vietnam National University, Ha Noi City, Vietnam, in 2009. He is currently working toward the M.S. /Ph.D degree in the School of Electrical Engineering, University of Ulsan, South Korea. His research interests include Intelligent control, Sliding mode control, Robot Calibration and Visual Servoing

Kang, Hee Jun received the B.S. degree in Mechanical Engineering from Seoul National University, Korea in 1985. He received the M.S. and Ph.D degrees in Mechanical Engineering from the University of Texas at Austin, USA in 1988 and 1991, respectively. Since March in 1992, he has been a Professor of Electrical Engineering in University of Ulsan. His current research interests are Sensor based robotic application, Robot Calibration, Haptics, Robot fault diagnosis and Mechanism analysis