Turkish Journal of Electrical Engineering and Computer Sciences

Volume 26 | Number 6 Article 35

1-1-2018

Adaptive antisingularity terminal sliding mode control for a robotic arm with model uncertainties and external disturbances

KIEM NGUYEN

TINH NGUYEN

QUYEN BUI

MINHTUAN PHAM

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 3224 – 3238

doi:10.3906/elk-1711-137

Adaptive antisingularity terminal sliding mode control for a robotic arm with model uncertainties and external disturbances

Kiem NGUYEN[1], **Tinh NGUYEN**[2,*], **Quyen BUI**[2], **Minhtuan PHAM**[3]

[1]Department of Electronics Power, Faculty of Electronics Engineering Technology, Hanoi University of Industry, Hanoi, Vietnam

[2]Institute of Information Technology, Vietnam Academy of Science and Technology, Hanoi, Vietnam

[3]Space Technology Institute, Vietnam Academy of Science and Technology, Hanoi, Vietnam

Received: 15.11.2017 • **Accepted/Published Online:** 07.09.2018 • **Final Version:** 29.11.2018

Abstract: In this paper, a radical adaptive terminal sliding mode control method for a robotic arm with model uncertainties and external disturbances is proposed in such a way that the singularity problem is completely dealt with. A radial basis function neural network (RBFNN) with an online weight tuning algorithm is employed to approximate unknown smooth nonlinear dynamic functions caused by the fact that there is no prior knowledge of the robotic dynamic model. Furthermore, a robust control law is utilized in order to eliminate total uncertainty composed of model uncertainties, external disturbances, and the inevitable approximation errors resulting from the finite number of the hidden-layer neurons of the RBFNN. Thanks to this proposed controller, a desired performance is achieved where tracking errors converge to zero within a finite time. In accordance with Lyapunov theory, the desired performance and the stability of the whole closed loop control system are ensured to be achieved. Finally, comparative computer simulation results are illustrated to confirm the validity and efficiency of the proposed control method.

Key words: Finite time convergence to zero, online weight tuning algorithm, online robust gain updating law, terminal sliding mode control, terminal sliding manifold

1. Introduction

It is evident that robotic arms have been widely applied thanks to their vital role in flexible automation processes with high speed and high accuracy. In contrast, they have often been subjected to external disturbances and a number of model uncertainties, such as payload variations and dynamic parameter variations, and therefore, in practice, it is impossible to get the precise expression of the dynamics of these arms. Consequently, various control approaches capable of dealing with such model uncertainties as well as making robustness against such external disturbances have been proposed and can be found in the literature.

The linear sliding mode control (SMC) method [1–4] is one of the most effective control methods to cope with the existence of the aforementioned model uncertainties as well as external disturbances and therefore has been applied widely. The primary principle of the linear SMC method is that proper sliding surfaces have been first established, followed by a procedure for designing robust control laws enabling the sliding variables to reach these sliding surfaces. In particular, discontinuous control efforts have been utilized in order to assure that the sliding variables have always been forced into and kept on the sliding surfaces. For this reason, the asymptotic convergence of the tracking errors to zero along the sliding surfaces has been ensured [3]. However, the asymptotic convergence to zero only implies that the tracking errors converge to equilibrium (zero) as time

*Correspondence: nvtinh@ioit.ac.vn

goes to infinity. So as to strengthen the convergence rate, the design gains of the linear SMC method must be augmented to be bigger. Thus, it may cause the harmful saturation of the control inputs. This is one of the drawbacks of the linear SMC method.

On the other hand, for robotic tasks requiring high precision, asymptotic convergence is insufficient. Consequently, in such a context, a finite-time convergence is required instead of the asymptotic convergence of the tracking errors. Accordingly, terminal sliding mode control (TSMC) techniques have been proposed to address this issue [5–16].

Thanks to establishing a nonlinear term in the terminal sliding manifold (terminal sliding surface) rather than the linear term as in the traditional linear SMC methods, the TSMC methods have made the tracking errors as well as the design gains reduce significantly as compared to traditional linear SMC methods, for instance the works in [5,6]. However, the disadvantages of the methods in [5,6] are that the singularity problem has not been fully considered, especially whenever a tracking error variable is equal to zero (i.e. $e_i = 0$ with $i = 1,...,n$), whereas the corresponding terminal sliding variable is not equal to zero (i.e. $s_i \neq 0$). Clearly, the singularity problem is a sensitive issue because it makes the control inputs unbounded. The works in [7,8] defined a nonsingular terminal sliding manifold in order to avoid the singularity problem, but the time needed to reach this nonsingular terminal sliding manifold heavily depended on the dynamics of both perturbations and external disturbances. In addition, the authors in [9] proposed an indirect approach with the aim of avoiding the singularity problem. To be specific, in this method, the singularity problem was prevented by switching between the terminal and linear sliding manifolds, but the results were that, still, the tracking performance could not be clearly improved owing to rough switches. The work in [10] addressed the singularity problem by means of a modified terminal sliding manifold in order to make switches smoother. The singularity problem was avoided by switching between the terminal and general sliding manifolds, which were constructed via quadratic functions. However, a very difficult problem in [10] was how to choose both K_{1i} and K_{2i} gains such that $\lambda_i(e_i)$ and its time derivative were continuous and bounded.

With regard to the radial basis function neural network (RBFNN), it is an undeniable fact that it is one of the strongest tools to approximate any unknown nonlinear smooth function with arbitrary precision by means of the capability of online learning of the weights. Furthermore, since this neural network (NN) has a simple NN weight updating law, a quickly convergent rate of the NN weights, and, above all, a simple structure, using it has facilitated procedures designing control laws and analyzing the stability for closed-loop control systems as opposed to other types of multilayer perceptron NNs. For instance, it was adopted in [6,10].

Our work in this paper has extended the works in [5,6,10] to overcome these aforementioned deficiencies. First, terminal sliding manifolds are proposed in the same way as in [6], but the singularity problem is dealt with fully. Second, the difficulty in [10], namely how to select both K_{1i} and K_{2i} gains, is also avoided. In addition, since it is very difficult to determine the upper bounds of the model uncertainties and external disturbances in practice, the robust gain of the robust control law is updated online via a robust gain updating law. Therefore, in this proposed controller, it is unnecessary to know the upper bounds in advance.

Finally, as a consequence of the discontinuous control efforts, chattering occurs, with the result that the tracking performance is reduced considerably. To eliminate the chattering, we replace sign functions by smooth functions.

This paper is organized as follows. Section 2 shows preliminaries. Section 3 represents procedures for designing the proposed control law. Section 4 illustrates the comparative computer simulation results for confirming the validity and advantage of this proposed control method. Section 5 describes our conclusion.

2. Preliminaries

2.1. The dynamics of an n-DOF rigid robotic arm

The dynamics of this arm is the following [6]:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{B}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{\tau_d} = \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}} \in \boldsymbol{R}^n$ are the position, velocity, and acceleration vectors of the joints, respectively. Next, $\boldsymbol{M}(\boldsymbol{q})$ expresses the inertial matrix. $\boldsymbol{B}(\boldsymbol{q},\dot{\boldsymbol{q}})$ shows the centripetal and Coriolis matrix. $\boldsymbol{G}(\boldsymbol{q})$ represents the vector of gravity components. $\boldsymbol{\tau_d}$ indicates the bounded vector of total uncertainty consisting of both model uncertainties and external disturbances. $\boldsymbol{\tau}$ reveals the vector of the torques considered to be the control inputs.

Property 1 *$\boldsymbol{M}(\boldsymbol{q})$ is positive definite, invertible, and bounded as follows:*

$$M_1 \|\boldsymbol{\theta}\|^2 \leq \boldsymbol{\theta}^T \boldsymbol{M}(\boldsymbol{q})\boldsymbol{\theta} \leq M_2 \|\boldsymbol{\theta}\|^2, \text{ with } \forall \boldsymbol{\theta} \in \boldsymbol{R}^n, \tag{2}$$

where M_1 and M_2 express known, positive, real constants.

Property 2 *: $\dot{\boldsymbol{M}}(\boldsymbol{q}) - 2\boldsymbol{B}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is a skew-symmetric matrix. In other words, we can write:*

$$\boldsymbol{\theta}^T \left[\dot{\boldsymbol{M}}(\boldsymbol{q}) - 2\boldsymbol{B}(\boldsymbol{q},\dot{\boldsymbol{q}})\right]\boldsymbol{\theta} = 0, \text{ for any } \boldsymbol{\theta} \in \boldsymbol{R}^n. \tag{3}$$

2.2. The structure of the RBFNN

In this subsection, we illustrate the structure of the RBFNN briefly. As can be seen in Figure 1, it constitutes three layers, namely the input, hidden, and output layers. In particular, the input vector is defined as $\boldsymbol{x} = [x_1, ..., x_{N_1}]^T$ with N_1 being the number of input nodes. The hidden layer includes N_2 activation functions. These activation functions, in this work, have been selected to be Gaussian type functions, as follows:

$$\sigma_i(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{\zeta}_i\|^2}{2\eta_i^2}\right) \text{ with } \quad i = 1, ..., N_2, \tag{4}$$

where $\boldsymbol{\zeta}_i$ and η_i are the center and width of the Gaussian function of the ith hidden-layer neuron, respectively.

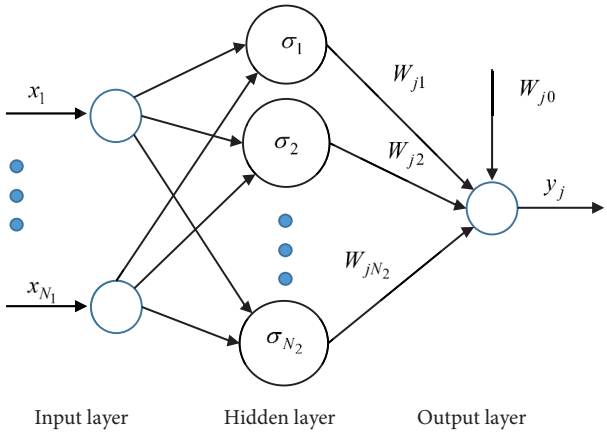

Figure 1. Structure of the RBFNN.

The output layer is a linear combination of the weights and the activation functions. To be specific, the expression of the jth output-layer node (neuron) is described as follows:

$$y_j = W_{j0} + \sum_{i=1}^{N_2} W_{ji}\sigma_i(\boldsymbol{x}) \text{ with } \quad j = 1, ..., N_3. \tag{5}$$

Here, W_{j0} is the threshold offset of the jth output-layer node, and W_{ji} is the weight connecting the ith hidden-layer neuron to the jth output-layer node. N_3 indicates the number of output nodes.

The output vector of the RBFNN can be written as follows:

$$\boldsymbol{y} = \boldsymbol{W}^T\boldsymbol{\sigma}(\boldsymbol{x}), \tag{6}$$

where $\boldsymbol{W}$ constitutes all the threshold offsets W_{j0} and the weights W_{ji}. $\boldsymbol{\sigma}(\boldsymbol{x}) = [1, \sigma_1(\boldsymbol{x}), \ldots, \sigma_{N_2}(\boldsymbol{x})]^T$ shows the vector of the hidden-layer activation functions. Here, it should be emphasized that 1 was inserted as the first element of $\boldsymbol{\sigma}(\boldsymbol{x})$. This can be explained by making $\boldsymbol{W}$ comprise both the threshold offsets and the weights.

For any bounded and continuous function vector $\boldsymbol{f}(\boldsymbol{x}) : \boldsymbol{R}^{N_1} \rightarrow \boldsymbol{R}^{N_3}$, there exists an ideal matrix $\boldsymbol{W}_*$ such that

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y} + \boldsymbol{\varepsilon} = \boldsymbol{W}_*^T\boldsymbol{\sigma} + \boldsymbol{\varepsilon}, \tag{7}$$

where $\boldsymbol{\varepsilon}$ is a reconstruction vector and can be made small arbitrarily.

3. Designing control law

3.1. The problem statement

The control goal here is to design a radical TSMC method such that the actual vector of the joint positions $\boldsymbol{q}$ tracks a predefined trajectory $\boldsymbol{q_d}$ with the tracking error vector $\boldsymbol{e} = \boldsymbol{q} - \boldsymbol{q_d}$ converging to zero within a finite time.

3.2. Describing terminal sliding manifolds

Similar to the work in [6], we have defined the ith terminal sliding manifold as follows:

$$s_i = \dot{e}_i + \beta_i sig(e_i)^{\varphi} = \dot{q}_i - \dot{q}_{ri}, \text{ with } \quad i = 1, ..., n, \tag{8}$$

where q_i is the ith position variable, $e_i = q_i - q_{di}$ is the ith tracking error variable, q_{di} shows the desired trajectory of q_i, β_i is a positive constant and can be chosen arbitrarily, $sig(e_i)^{\varphi} = |e_i|^{\varphi} sign(e_i)$, and $\varphi = \frac{\varphi_1}{\varphi_2}$ with φ_1 and φ_2 being positive odd integers satisfying the following inequality [10]:

$$\frac{1}{2} < \frac{\varphi_1}{\varphi_2} < 1. \tag{9}$$

Despite comprising the absolute value and signum operators, Eq. (8) is always smooth and differentiable [8].

It should be noted that $\dot{q}_{ri} = \dot{q}_{di} - \beta_i sig(e_i)^{\varphi}$ is an auxiliary variable utilized to compute the control inputs.

On the one hand, if $s_i = 0$, then

$$\dot{e}_i = -\beta_i sig\left(e_i\right)^{\varphi}, \tag{10}$$

and therefore $e_i = 0$ will clearly be the only attractor of Eq. (10). Moreover, $s_i = 0$ yields the following:

$$\ddot{q}_{ri} = \ddot{q}_{di} + \varphi\beta_i^2 \left|e_i\right|^{2\varphi-1}. \tag{11}$$

As a result, $\varphi > \frac{1}{2}$ stops the singularity problem occurring whenever $e_i \to 0$.

On the other hand, if $e_i \to 0$ while $s_i \neq 0$, then the singularity problem will happen due to the fact that there exists a negative fractional power in $\ddot{q}_{ri}$ as follows:

$$\ddot{q}_{ri} = \ddot{q}_{di} - \varphi\beta_i sig\left(e_i\right)^{\varphi-1}\dot{e}_i. \tag{12}$$

Accordingly, eliminating the singularity problem is a fundamental task in this situation.

3.3. Computing the control inputs

Unlike the work in [6], where the singularity problem was completely ignored, here, with the purpose of avoiding this problem, we have proposed more auxiliary variables as follows:

$$u_i = \begin{cases} \ddot{q}_{di} - \varphi\beta_i\gamma_i^{\varphi-1} sign\left(e_i\right)\dot{e}_i & if \quad s_i \neq 0 \quad and \quad \left|e_i\right| \leq \gamma_i \\ \ddot{q}_{ri} & otherwise \end{cases}, \tag{13}$$

with $i = 1, ..., n$, where γ_i illustrates a boundary layer around zero corresponding to e_i and can be chosen to be tiny arbitrarily.

Now we can rewrite Eqs. (8) and (13) in terms of vectors respectively, as follows:

$$\boldsymbol{s} = \dot{\boldsymbol{e}} + \boldsymbol{\beta sig}\left(\boldsymbol{e}\right)^{\varphi} = \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_{\boldsymbol{r}}, \tag{14}$$

$$\boldsymbol{u} = \left[u_1, ..., u_n\right]^T, \tag{15}$$

where $\boldsymbol{e} = \left[e_1, ..., e_n\right]^T$, $\boldsymbol{s} = \left[s_1, ..., s_n\right]^T$, $\dot{\boldsymbol{q}}_{\boldsymbol{r}} = \left[\dot{q}_{r1}, ..., \dot{q}_{rn}\right]$, $\boldsymbol{\beta} = \boldsymbol{diag}\left[\beta_1, ..., \beta_n\right]$ with **diag** showing a diagonal matrix, $\boldsymbol{sig}\left(\boldsymbol{e}\right)^{\varphi} = \left[sig\left(e_1\right)^{\varphi}, ..., sig\left(e_n\right)^{\varphi}\right]^T$. Next, we can define a nonlinear dynamic function vector as follows:

$$\boldsymbol{f}\left(\boldsymbol{x}\right) = \boldsymbol{M}\left(\boldsymbol{q}\right)\boldsymbol{u} + \boldsymbol{B}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right)\dot{\boldsymbol{q}}_{\boldsymbol{r}} + \boldsymbol{G}\left(\boldsymbol{q}\right), \text{ with } \boldsymbol{x} = \left[\boldsymbol{u}^T, \dot{\boldsymbol{q}}_{\boldsymbol{r}}^T, \dot{\boldsymbol{q}}^T, \boldsymbol{q}^T\right]^T. \tag{16}$$

Subtracting $\boldsymbol{f}\left(\boldsymbol{x}\right)$ from both sides of Eq. (1) yields

$$\boldsymbol{M}\left(\boldsymbol{q}\right)\dot{\boldsymbol{s}} = -\boldsymbol{B}\left(\boldsymbol{q}, \dot{\boldsymbol{q}}\right)\boldsymbol{s} - \boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\tau}_{\boldsymbol{d}} + \boldsymbol{\tau}. \tag{17}$$

Observing Eq. (17) reveals that if the dynamics of this arm, $\boldsymbol{f}\left(\boldsymbol{x}\right)$, is expressed accurately, and further $\boldsymbol{\tau}_{\boldsymbol{d}} = \boldsymbol{0}$, then the control inputs can be directly computed via a model-based control law as follows:

$$\boldsymbol{\tau} = \boldsymbol{f}\left(\boldsymbol{x}\right) - \boldsymbol{\Gamma sig}\left(\boldsymbol{s}\right)^{\rho}, \tag{18}$$

where $\boldsymbol{\Gamma}$ is a positive-definite design gain matrix and can be selected arbitrarily, and $\rho = \rho_1/\rho_2$ with ρ_1 and ρ_2 seeming to be positive odd integers satisfying $\rho_1 < \rho_2$ [10].

Substitution of Eq. (18) into Eq. (17) while noting that $\boldsymbol{\tau_d} = \mathbf{0}$ leads to

$$\boldsymbol{M}(\boldsymbol{q})\,\dot{\boldsymbol{s}} = -\boldsymbol{B}(\boldsymbol{q},\dot{\boldsymbol{q}})\,\boldsymbol{s} - \boldsymbol{\Gamma sig}(\boldsymbol{s})^{\rho}. \tag{19}$$

Observing Eq. (19) reveals that the stability of the whole closed-loop control system is easy to be proven by means of Lyapunov criteria.

Meanwhile, it is unlucky that there is, in practice, often no prior knowledge of the robotic dynamics perfectly, or, more precisely, $\boldsymbol{f}(\boldsymbol{x})$ is unknown, and further $\boldsymbol{\tau_d} \neq \mathbf{0}$. As a consequence, it is impossible to apply the model-based control law of Eq. (18). Thus, in this work, $\boldsymbol{f}(\boldsymbol{x})$ is approximated through the RBFNN shown by Eq. (6). In addition, a robust term is utilized to overcome the total uncertainty consisting of $\boldsymbol{\tau_d}$ and the inevitable approximation errors caused by the finite number of the hidden-layer neurons of the RBFNN. To specify, the proposed control law is defined as follows:

$$\boldsymbol{\tau} = \hat{\boldsymbol{f}}\left(\boldsymbol{x},\hat{\boldsymbol{W}}\right) - \boldsymbol{\Gamma sig}(\boldsymbol{s})^{\rho} - \hat{\boldsymbol{d}}, \tag{20}$$

where $\hat{\boldsymbol{d}}$ is the aforementioned robust term determined in a specific way subsequently. $\hat{\boldsymbol{f}}\left(\boldsymbol{x},\hat{\boldsymbol{W}}\right)$ is the output of the RBFNN and is adopted to estimate $\boldsymbol{f}(\boldsymbol{x})$, where $\hat{\boldsymbol{W}}$ is an estimation of $\boldsymbol{W}_*$ and can be updated online by means of the online weight tuning algorithm. Particularly, the expression of $\hat{\boldsymbol{f}}\left(\boldsymbol{x},\hat{\boldsymbol{W}}\right)$ is expressed as follows:

$$\hat{\boldsymbol{f}}\left(\boldsymbol{x},\hat{\boldsymbol{W}}\right) = \hat{\boldsymbol{W}}^{T}\boldsymbol{\sigma}(\boldsymbol{x}). \tag{21}$$

Substituting Eqs. (7), (20), and (21) into Eq. (17) results in

$$\boldsymbol{M}(\boldsymbol{q})\,\dot{\boldsymbol{s}} = -\boldsymbol{B}(\boldsymbol{q},\dot{\boldsymbol{q}})\,\boldsymbol{s} - \tilde{\boldsymbol{W}}^{T}\boldsymbol{\sigma} + \boldsymbol{d} - \boldsymbol{\Gamma sig}(\boldsymbol{s})^{\rho} - \hat{\boldsymbol{d}}, \tag{22}$$

where $\boldsymbol{d} = -\boldsymbol{\varepsilon} - \boldsymbol{\tau_d}$, $\tilde{\boldsymbol{W}} = \boldsymbol{W}_* - \hat{\boldsymbol{W}}$.

Next, the online weight tuning algorithm has been proposed in the following form:

$$\dot{\hat{\boldsymbol{W}}} = -\boldsymbol{H}\boldsymbol{\sigma}\boldsymbol{s}^{T}, \tag{23}$$

where $\boldsymbol{H}$ is a positive definite matrix and can be chosen arbitrarily.

Assumption 1 *Suppose that the uncertainty* $\boldsymbol{d}$ *is bounded as follows:*

$$\|\boldsymbol{d}\| \leq \Omega, \tag{24}$$

with Ω *being an unknown positive constant.*

The robust term $\hat{\boldsymbol{d}}$ *in Eq. (20) is proposed in the following:*

$$\hat{\boldsymbol{d}} = \hat{\Omega}\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|}, \tag{25}$$

with $\hat{\Omega}$ *being the robust gain.*

As opposed to the works in [5,10] where there was, in advance, a need to know the upper bound of the total uncertainty, and therefore those corresponding robust gains were chosen to be constants, in this work, due to the fact that it is, in practice, very hard to obtain the prior knowledge of this upper bound, the robust gain is updated online as follows:

$$\dot{\hat{\Omega}} = \Psi \left\| \boldsymbol{s} \right\|, \tag{26}$$

where Ψ is a positive constant and can be chosen arbitrarily.

For the sake of convenience in illustration, we have proposed the scheme of the whole closed loop control system as in Figure 2.

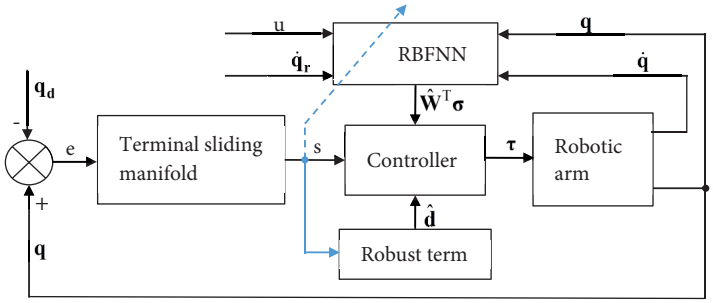

Figure 2. The scheme of the whole closed loop control system.

3.4. Stability analysis

Theorem 1 *Let us consider the n-DOF robotic arm described by Eq. (1). Let Assumption 1 hold. If the terminal sliding manifolds, the control law, the online weight tuning algorithm, the robust term, and the online robust gain updating law are proposed by Eqs. (8), (20), (23), (25), and (26), respectively, then all the signals in the whole closed-loop control system will be bounded and further the tracking errors will converge to zero within a finite time.*

Proof Let us consider a Lyapunov candidate function as follows:

$$V = \frac{1}{2}\boldsymbol{s}^T \boldsymbol{M}(\boldsymbol{q})\,\boldsymbol{s} + \frac{1}{2\Psi}\tilde{\Omega} + \frac{1}{2} tr\left(\tilde{\boldsymbol{W}}^T \boldsymbol{H}^{-1} \tilde{\boldsymbol{W}}\right), \tag{27}$$

where $\tilde{\Omega} = \Omega - \hat{\Omega}$, and tr(.) is the trace of the matrix.

Differentiating Eq. (27) and noting that $\dot{\tilde{\Omega}} = -\dot{\hat{\Omega}}$ and $\dot{\tilde{\boldsymbol{W}}} = -\dot{\hat{\boldsymbol{W}}}$ yields

$$\dot{V} = \boldsymbol{s}^T \left[\frac{1}{2}\dot{\boldsymbol{M}}(\boldsymbol{q})\,\boldsymbol{s} + \boldsymbol{M}(\boldsymbol{q})\,\dot{\boldsymbol{s}}\right] - \frac{1}{\Psi}\tilde{\Omega}\dot{\hat{\Omega}} - tr\left(\tilde{\boldsymbol{W}}^T \boldsymbol{H}^{-1}\dot{\hat{\boldsymbol{W}}}\right). \tag{28}$$

Noting Eq. (3) with the substitution of Eqs. (22), (23), (25), and (26) into Eq. (28) yields

$$\dot{V} = \boldsymbol{s}^T \left[-\boldsymbol{\Gamma sig}(\boldsymbol{s})^{\rho} - \tilde{\boldsymbol{W}}^T \boldsymbol{\sigma} + \boldsymbol{d}\right] - \hat{\Omega}\left\|\boldsymbol{s}\right\| - \tilde{\Omega}\left\|\boldsymbol{s}\right\| + tr\left(\tilde{\boldsymbol{W}}^T \boldsymbol{\sigma}\boldsymbol{s}^T\right). \tag{29}$$

It is clear that $tr\left(\tilde{\boldsymbol{W}}^T \boldsymbol{\sigma}\boldsymbol{s}^T\right) = \boldsymbol{s}^T \tilde{\boldsymbol{W}}^T \boldsymbol{\sigma}$ and $\hat{\Omega} + \tilde{\Omega} = \Omega$, which implies that

$$\dot{V} = \boldsymbol{s}^T \left[-\boldsymbol{\Gamma sig}(\boldsymbol{s})^{\rho} + \boldsymbol{d}\right] - \Omega\left\|\boldsymbol{s}\right\|. \tag{30}$$

Next, based on Assumption 1, we get

$$\dot{V} < -\boldsymbol{s}^T \boldsymbol{\Gamma} \boldsymbol{sig}\left(\boldsymbol{s}\right)^{\rho}. \tag{31}$$

Observing Eq. (31) implies that $\dot{V} \leq 0$ for any $\boldsymbol{s}$. For this reason, in accordance with the Lyapunov criteria and LaSalle extension, it follows that $V\left(t\right) \leq V\left(0\right)$, which in turn leads to that if the initial values $\boldsymbol{s}\left(0\right), \tilde{\boldsymbol{W}}\left(0\right)$, and $\tilde{\Omega}\left(0\right)$ are bounded, then $\boldsymbol{s}\left(t\right), \tilde{\boldsymbol{W}}\left(t\right)$, and $\tilde{\Omega}\left(t\right)$ will be bounded for all $t > 0$. As a result, all signals in the overall control system will be bounded.

Next, with the purpose of confirming the finite time convergence of $\boldsymbol{s}$ to zero, we choose another Lyapunov candidate as follows:

$$V_2 = \frac{1}{2} \boldsymbol{s}^T \boldsymbol{M}\left(\boldsymbol{q}\right) \boldsymbol{s}. \tag{32}$$

Taking the time derivative of Eq. (32) results in

$$\dot{V}_2 = -\boldsymbol{s}^T \boldsymbol{\Gamma} \boldsymbol{sig}\left(\boldsymbol{s}\right)^{\rho} + \boldsymbol{s}^T \left[-\tilde{\boldsymbol{W}}^T \boldsymbol{\sigma} + \boldsymbol{d} - \hat{\boldsymbol{d}}\right]. \tag{33}$$

It is obvious that

$$\dot{V}_2 \leq -\boldsymbol{s}^T \boldsymbol{\Gamma} \boldsymbol{sig}\left(\boldsymbol{s}\right)^{\rho} + \left\|\boldsymbol{s}\right\| \left(\left\|\tilde{\boldsymbol{W}}^T \boldsymbol{\sigma}\right\| + \Omega - \hat{\Omega}\right). \tag{34}$$

It should be emphasized that $\left\|\tilde{\boldsymbol{W}}^T \boldsymbol{\sigma}\right\|$ is bounded [10]. Meanwhile, $\hat{\Omega}$ is always updated online such that it increases as long as $\left\|\boldsymbol{s}\right\| \neq 0$, as can be seen from Eq. (26). Accordingly, if the initial value $\hat{\Omega}\left(0\right)$ and the gain Ψ are chosen to be big enough, then the following inequality will hold after a very short time interval:

$$\left\|\tilde{\boldsymbol{W}}^T \boldsymbol{\sigma}\right\| + \Omega - \hat{\Omega} \leq 0. \tag{35}$$

It follows that

$$\dot{V}_2 \leq -\boldsymbol{s}^T \boldsymbol{\Gamma} \boldsymbol{sig}\left(\boldsymbol{s}\right)^{\rho}. \tag{36}$$

Similar to [10], the following inequality is derived:

$$\dot{V}_2 \leq -\Gamma_{\min} \left(\frac{2}{M_2}\right)^{(1+\rho)/2} V_2^{(1+\rho)/2}, \tag{37}$$

where $\Gamma_{\min}$ is the minimum eigenvalue of $\boldsymbol{\Gamma}$.

Therefore, the finite-time interval taken for $\boldsymbol{s}$ reaching to zero is calculated as follows [10]:

$$t_s = \frac{V_2^{(1-\rho)/2}\left(0\right)}{\Gamma_{\min} \left(\frac{2}{M_2}\right)^{(1+\rho)/2} \frac{1-\rho}{2}}. \tag{38}$$

When the states of the system are on the terminal sliding manifolds, i.e. $\boldsymbol{s} = \boldsymbol{0}$, the finite-time interval taken for $e_i \to 0$ is computed as follows [10]:

$$t_{ei} = \frac{e_i\left(t_s\right)^{1-\varphi}}{\beta_i\left(1-\varphi\right)}. \tag{39}$$

Combining Eqs. (38) and (39), the total finite-time interval taken for each tracking error e_i coming to zero is expressed as follows:

$$t_{i-total} = t_s + t_{ei}. \tag{40}$$

This finishes the proof. □

Remark 1 *In order not to the chattering, we have replaced Eq. (25) by the following:*

$$\hat{\boldsymbol{d}} = \begin{cases} \hat{\Omega}\frac{\boldsymbol{s}}{\|\boldsymbol{s}\|} & if \quad \|\boldsymbol{s}\| \geq \kappa \\ \\ \hat{\Omega}\frac{\boldsymbol{s}}{\kappa} & if \quad \|\boldsymbol{s}\| < \kappa \end{cases}, \tag{41}$$

where κ shows a very small positive constant.

Thus, so as to ensure the boundedness of $\hat{\Omega}$, Eq. (26) must be replaced by the following:

$$\dot{\hat{\Omega}} = \begin{cases} \Psi \|\boldsymbol{s}\| & if \quad \|\boldsymbol{s}\| \geq \kappa \\ 0 & if \quad \|\boldsymbol{s}\| < \kappa \end{cases}. \tag{42}$$

4. Simulation results

To confirm the rightness and advantage of this proposed control method, we have implemented a computer simulation by means of MATLAB/Simulink software for the two-link planar robotic arm described in Figure 3.

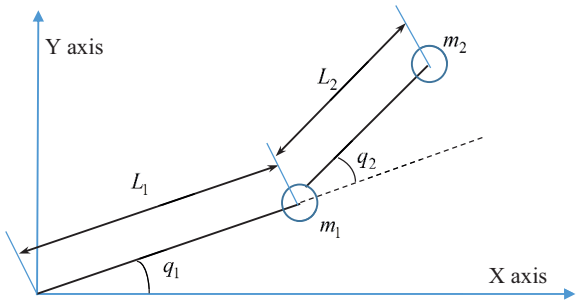

Figure 3. Two-link planar robotic arm.

The dynamics of this arm was derived from [6] with the dynamic parameters being given specifically as follows: $m_1 = 1$ (kg), $m_2 = 1$ (kg), $L_1 = 1$ (m), $L_2 = 0.8$ (m). Without loss of generality, it was given as follows:

$$\boldsymbol{\tau_d} = \begin{bmatrix} 2\sin(2t) + 0.5\dot{q}_1 \\ 1.5\cos(3t) - 0.3\dot{q}_2 \end{bmatrix}. \tag{43}$$

The structure of the RBFNN has been selected to have 8 input-layer nodes, 40 hidden-layer neurons, and 2 output-layer nodes. In addition, the center vector $\boldsymbol{\zeta}_i$ (with $i = 1, ..., 40$) of Gaussian function of the ith hidden-layer neuron was chosen to be an 8 ×1 vector of random numbers in the range (–1, 1). The width of each hidden-layer neuron was chosen to be $\eta_i = 2$ with $i = 1, ..., 40$. The initial values of the NN weights were selected to be a matrix of random values in the range (–1, 1) as well. The parameter matrix of the online weight tuning algorithm was selected as $\boldsymbol{H} = 10.\boldsymbol{I}_{41\times 41}$.

For the robust term, the initial value of the robust gain was chosen as $\hat{\Omega}(0) = 20$, and further the gain of the robust updating law was also chosen as $\Psi = 2$.

For the terminal sliding manifolds, we chose the gains as $\varphi = 7/11$ and $\beta_i = 10$ with $i = 1, 2$. For the control design parameters, we have selected $\rho = 1/3$, $\mathbf{\Gamma} = \boldsymbol{diag}(10, 10)$, $\gamma_i = 0.001$, $\kappa = 0.05$.

The desired vector $\boldsymbol{q_d}$ has been chosen to be the following:

$$\boldsymbol{q_d} = \begin{bmatrix} 1.6 + 0.4\sin(0.5\pi t) \\ 0.6 + 0.5\cos(0.6\pi t) \end{bmatrix} \text{ (rad)}. \tag{44}$$

Figure 4 depicts the tracking performance of this proposed TSMC method. Next, for comparative purposes, a linear SMC method with the same control gains other than $\varphi = \rho = 1$ was also stimulated, whose tracking performance is shown in Figure 5. The comparisons between the tracking errors of the proposed TSMC method and the linear SMC method are expressed in Figure 6 and Figure 7. Obviously, in the steady state, the tracking errors of the former are much smaller than those of the latter. To be specific, the tracking errors of the former are around 3×10^{-5} and 10^{-6} (rad) at joint 1 and joint 2, respectively, whereas those of the latter are about 10^{-2} and 10^{-3} (rad), respectively.

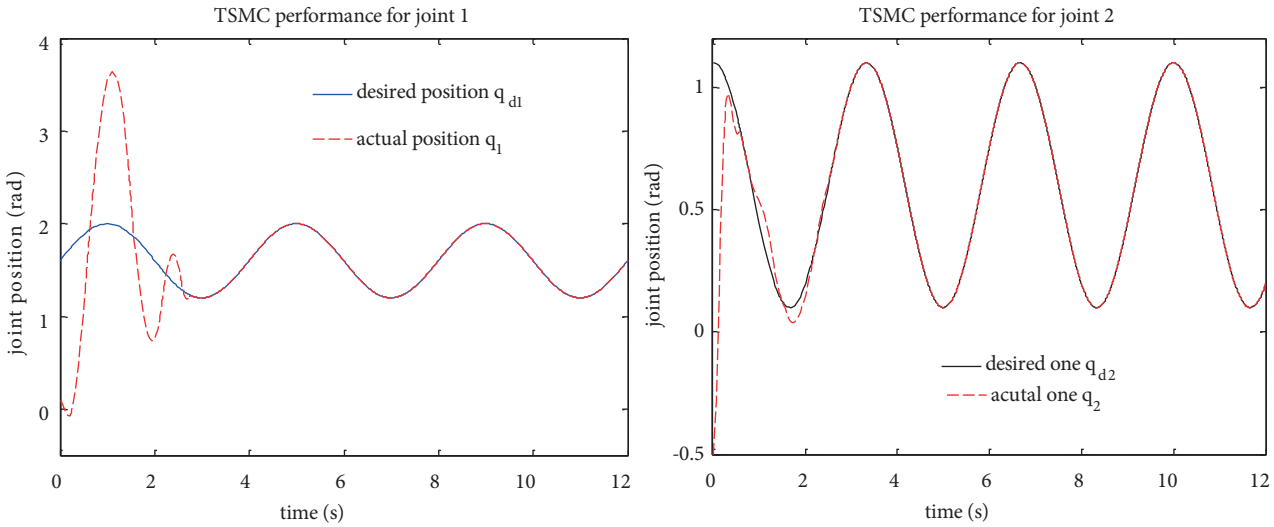

Figure 4. The tracking performance of the proposed TSMC method.

Furthermore, the torques (control inputs) of both the proposed TSMC method and the linear SMC method are described in Figure 8. Clearly, in the transient state, the torques of the former are significantly small as opposed to those of the latter. In particular, the biggest torques of the former are 110 and 50 (Nm) at joint 1 and joint 2, respectively, while those of the latter are 155 and 115 (Nm), respectively. It is vital to note that the big torques of the latter may also cause the harmful saturation of the actuators.

Combining Figures 6, 7, and 8, we are able to observe that in the transient state, the torques' initial values in our proposed TSMC method are smaller than those in the linear SMC approach. As a result, the periods for reaching zero of the tracking errors $e_{1,2}$ of the former are longer than those of the latter. This is the drawback of the former compared with the latter. However, this drawback is acceptable in robotic control engineering.

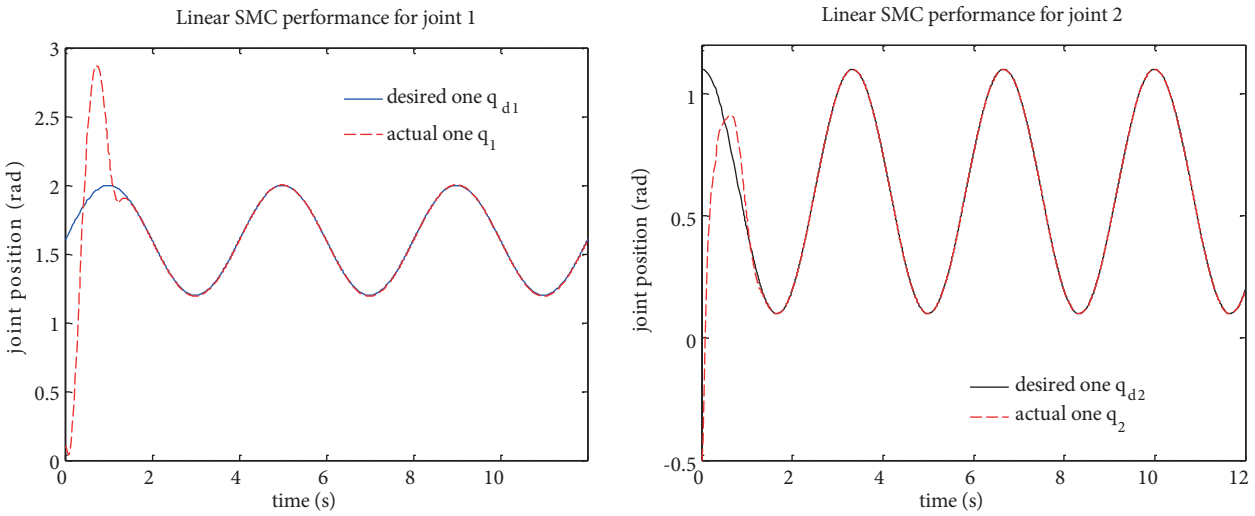

Figure 5. The tracking performance of the linear SMC method.

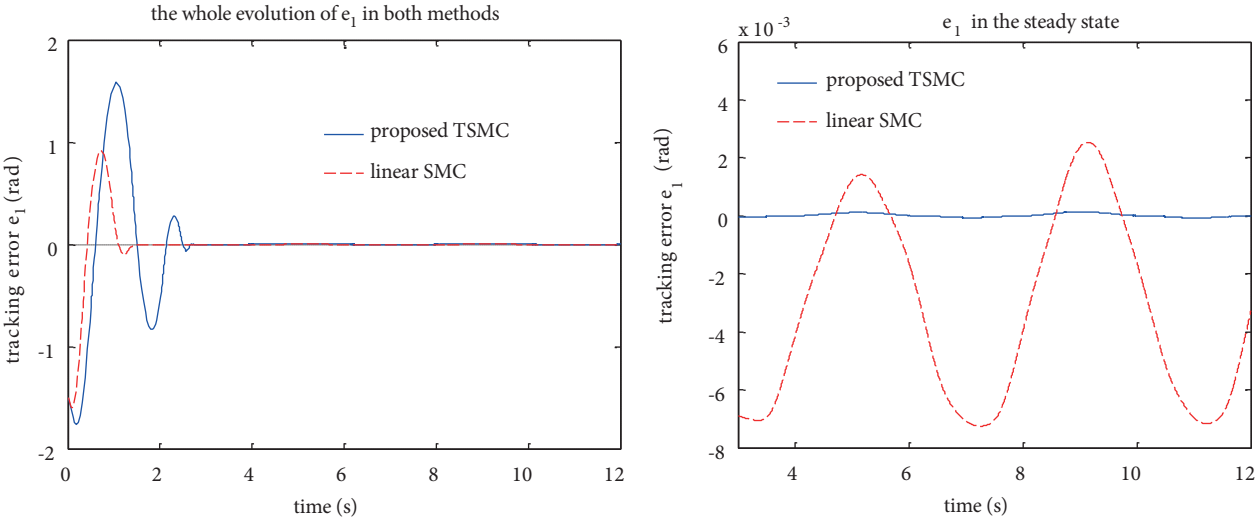

Figure 6. Comparison, in the steady state, between the tracking errors at joint 1 of the proposed TSMC and linear SMC method.

On the contrary, exerting small control efforts at the beginning to avoid the harmful saturation of the control inputs (torques) is one of the significant advantages of our proposed TSMC method compared with the linear SMC approach.

In contrast, to compare with the approach in [6], we have also implemented comparative simulations with $\boldsymbol{\tau_d}$ described in Eq. (43) and the desired vector $\boldsymbol{q_d}$ illustrated in Eq. (44). As can be seen from Figures 9, 10, and 11, it is obvious that in the steady state, not only the tracking errors $e_{1,2}$ but also the torques $\tau_{1,2}$ of our proposed method are much smaller and smoother than those of [6]. An explanation is that Eq. (13) in our control method enables both the tracking errors and the torques to be better.

Last but not least, to show the effects of replacing Eqs. (25, 26) with Eqs. (41, 42), respectively, we have carried out the simulations corresponding to the sign functions of Eqs. (25, 26) and smooth ones of Eqs. (41,

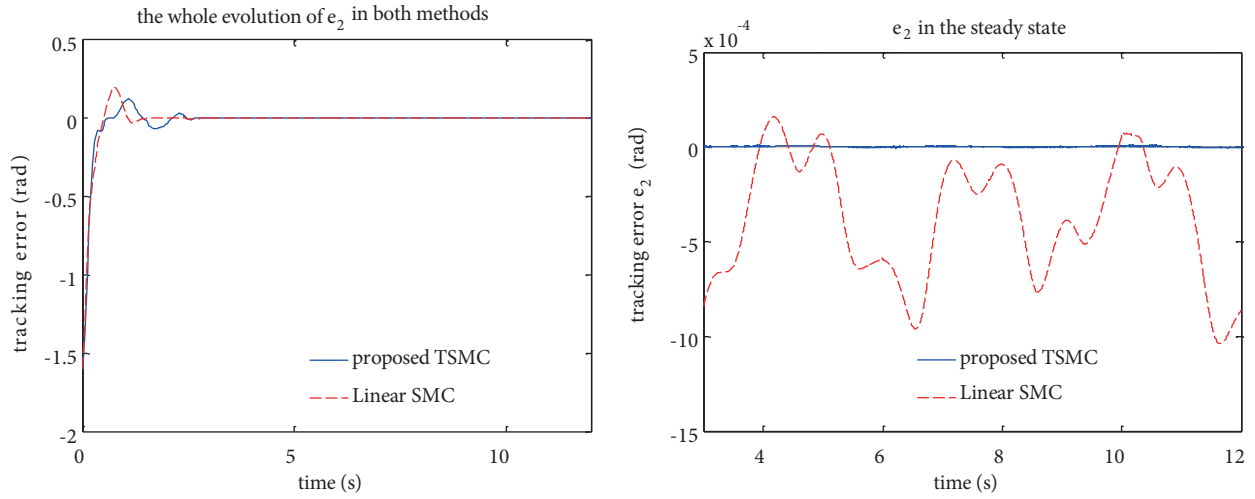

Figure 7. Comparison in the steady state between the tracking errors at joint 2 of the proposed TSMC and linear SMC method.

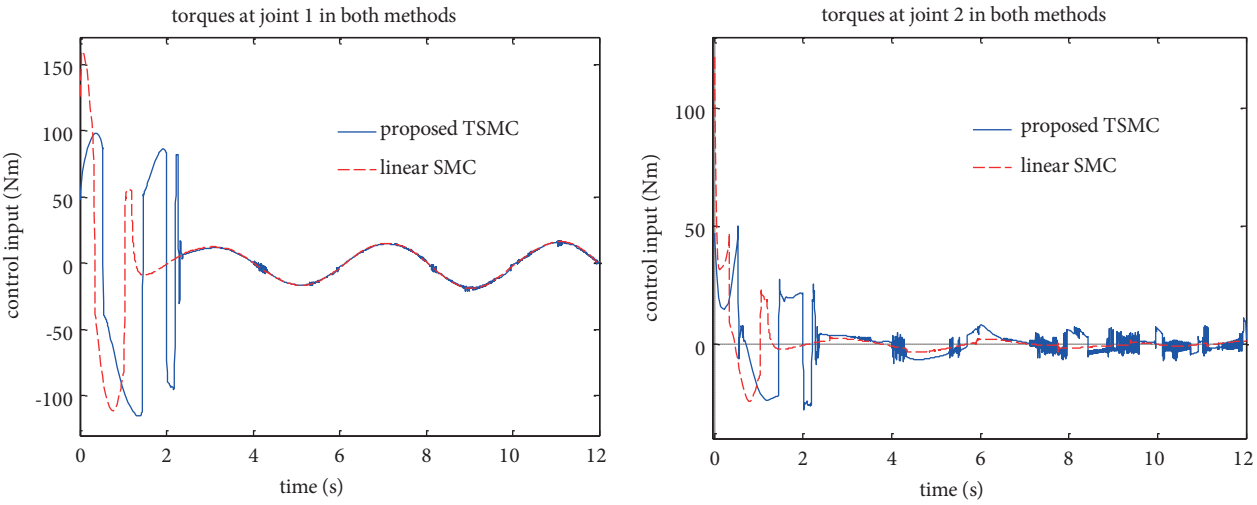

Figure 8. Comparison between the control inputs at both joints of the proposed TSMC and linear SMC method.

42). To be specific, as illustrated in Figure 12, the chattering phenomena have happened at both the joints in the case using sign function Eqs. (25) and (26), while there is no chattering phenomenon in the case utilizing smooth Eqs. (41) and (42). The cost of making these replacements is that in the steady state, the tracking errors of the latter are possibly bigger than those of the former. For example, Figure 13 shows the comparison between the former and the latter at joint 1. It is evident that the biggest of e_1 in the latter is bigger than that in the former, with the latter constituting 4 $\times 10^{-5}$ and the former 2 $\times 10^{-5}$ (rad). Nevertheless, the cost perhaps is allowed in robotic control engineering.

From the above simulation results, we can mention that our proposed TSMC method is more advantageous than both the linear SMC method and the approach in [6].

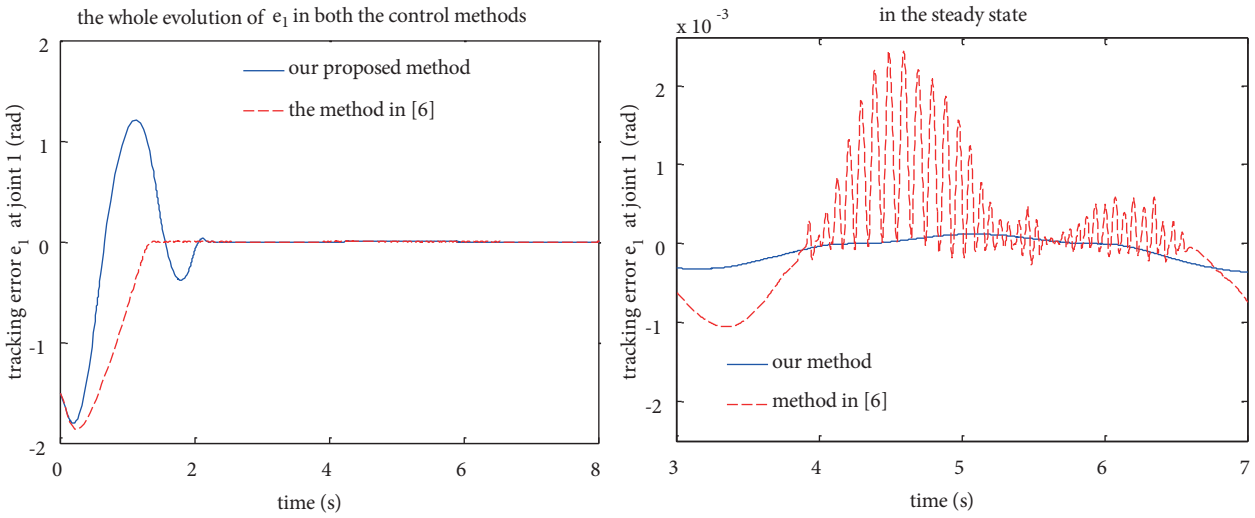

Figure 9. Comparison in the steady state between the tracking errors e_1, at joint 1, associated with our proposed method and the approach in [6].

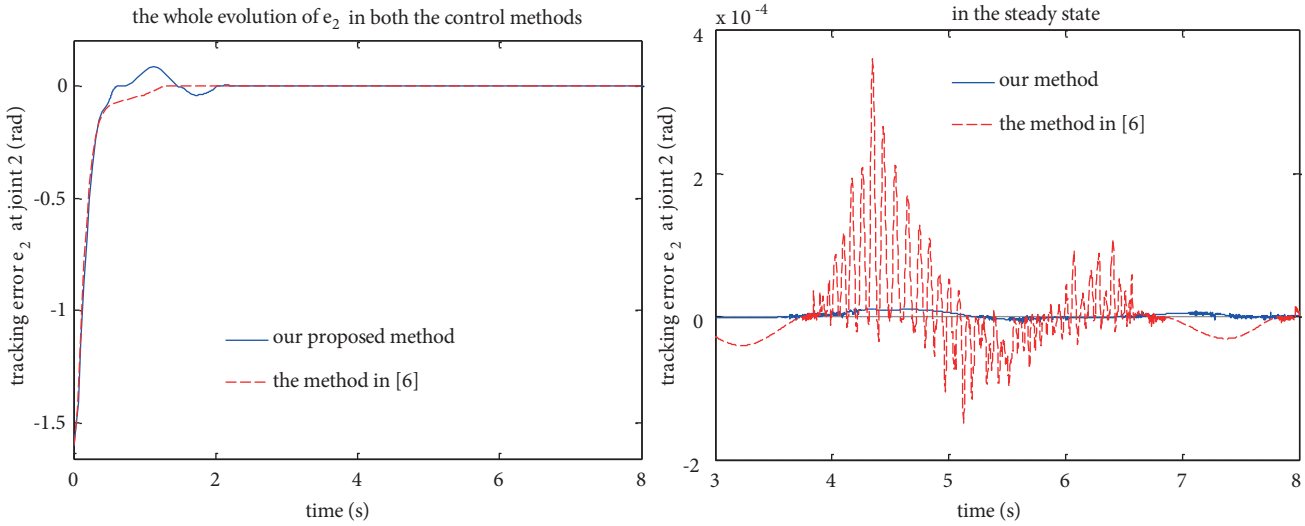

Figure 10. Comparison in the steady state between the tracking errors e_2, at joint 2, corresponding to our proposed method and the approach of [6].

5. Conclusion

This paper has represented our novel antisingularity TSMC method for a rigid robotic arm. Apart from easing some disadvantages of the linear SMC, such as making big control efforts in the transient state but getting rather big tracking errors in the steady state, this proposed TSMC method has still preserved the robustness against the model uncertainties as well as the external disturbances. Furthermore, the singularity problem has also been completely dealt with by means of Eq. (13). Thanks to the advantage and simplicity of this proposed TSMC method, it will be a great candidate for practical applications.

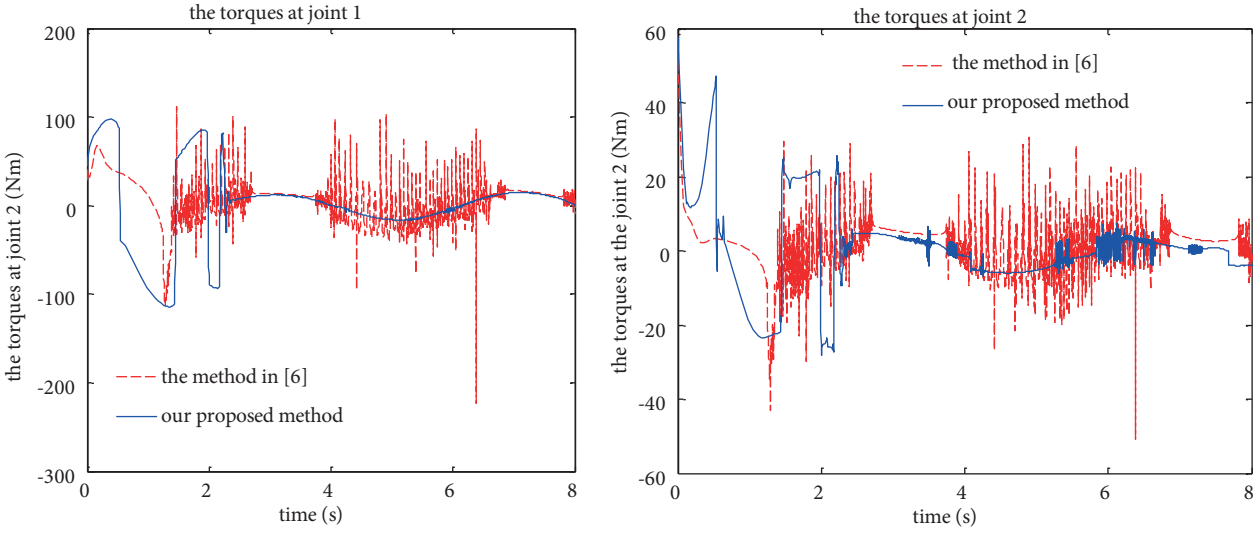

Figure 11. Comparison between the torques of our proposed method and the approach in [6].

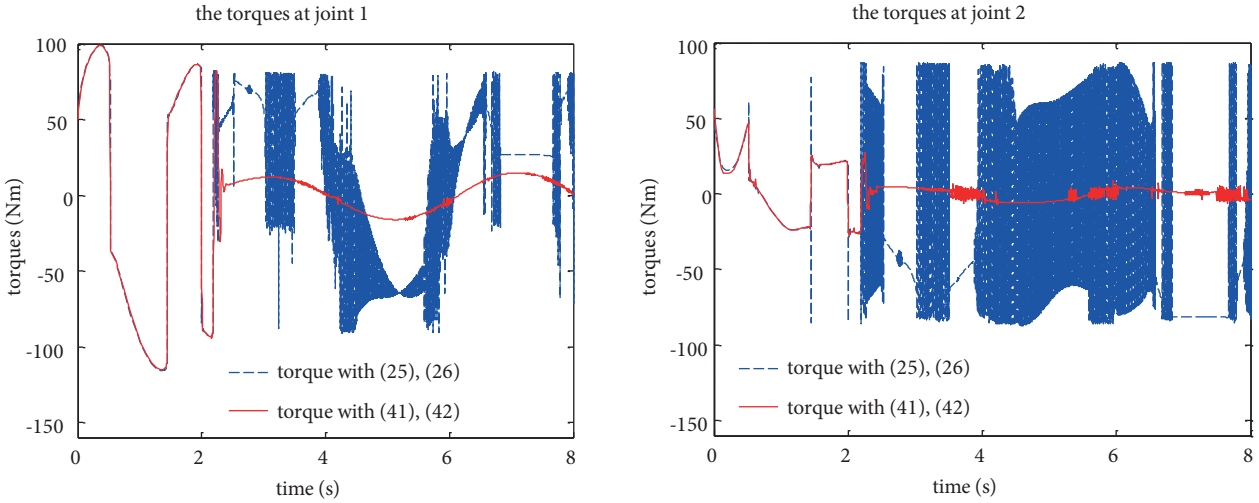

Figure 12. Comparison of the torques between the cases using sign (Eqs. (25) and (26)) and smooth (Eqs. (41) and (42)) functions.

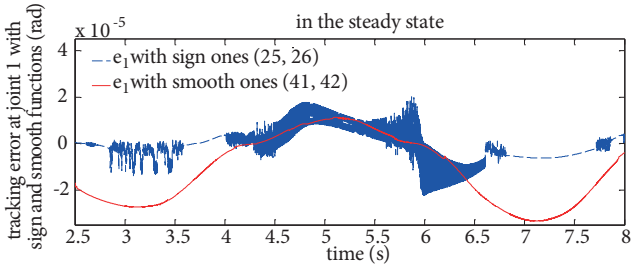

Figure 13. Comparison between the tracking error at joint 1, e_1, between the cases using sign (Eqs. (25) and (26)) and smooth (Eqs. (41) and (42)) functions.

Remark 2 *When it comes to the terminal sliding surface of Eq. (28) in [6] as follows:*

$$s = \dot{e} + \beta sig\,(e)^{\varphi} \;\; with\; 0 < \varphi < 1,$$

if $s = 0$*, then* $\dot{e} = -\beta sig\,(e)^{\varphi}$*. It is clear that if* φ *is chosen to be a certain value that satisfies the above condition and furthermore is smaller than 0.5, i.e.* $0 < \varphi < 0.5$*, then whenever both* $s = 0$ *and* $e = 0$*, the singularity will occur in Eq. (30) of [6] as follows:*

$$\dot{s} = \ddot{e} + \varphi\beta\,|e|^{\varphi-1}\,\dot{e} = \ddot{e} - \varphi\beta^2\,|e|^{2\varphi-1}\,sig\,(e)\,.$$

This is the reason why we mentioned that the singularity problem in [6] has been completely ignored.

Acknowledgment

This work was supported by the Institute of Information Technology, Vietnam Academy of Science and Technology, Hanoi, Vietnam.

References

[1] Sabanovic A. Variable structure systems with sliding modes in motion control - A Survey. IEEE T Indus Inform 2011; 7: 212-223.

[2] Husek P. Adaptive sliding mode control with moving sliding surface. Appl Soft Comp 2016; 42: 178-183.

[3] Slotine J, Li W. Applied Nonlinear Control. Englewood Cliffs, NJ, USA: Prentice Hall, 1991.

[4] Efe MÖ. Fractional fuzzy adaptive sliding-mode control of a 2-DOF direct-drive robot arm. IEEE T Syst Man Cyb 2008; 38: 1561-1570.

[5] Tang Y. Terminal sliding mode control for rigid robots. Automatica 1998; 34: 51-56.

[6] Tran M, Kang H. Adaptive terminal sliding mode control of uncertain robotic manipulators based on local approximation of a dynamic system. Neurocomputing 2017; 228: 231-240.

[7] Feng Y, Yu X, Man Z. Non-singular terminal sliding mode control of rigid manipulators. Automatica 2002; 38: 2159-2167.

[8] Yu S, Yu X, Shirinzadeh B, Man Z. Continuous finite-time control for robotic manipulators with terminal sliding mode. Automatica 2005; 41: 1957-1964.

[9] Man Z, Yu X. Terminal sliding mode control of MIMO linear systems. IEEE T Circ Syst-I 1997; 44: 1065-1070.

[10] Wang L, Chai T, Zhai L. Neural-network-based terminal sliding-mode control of robotic manipulators including actuator dynamics. IEEE T Indus Elect 2009; 56: 3296-3304.

[11] Jing X, Guo Z, Global fast dynamic terminal sliding mode control for a quadrotor UAV. ISA T 2017; 66: 233-240.

[12] Li S, Zhou M, Yu X. Design and implementation of terminal sliding mode control method for PMSM speed regulation system. IEEE T Ind Inform 2013; 9: 1879-1891.

[13] Li S, Du H, Yu X. Discrete-time terminal sliding mode control systems based on Euler's discretization. IEEE T Autom Cont 2014; 59: 546-552.

[14] Li S, Wu C, Sun Z. Design and implementation of clutch control for automotive transactions using terminal sliding mode control and uncertainty observer. IEEE T Veh Technol 2016; 65: 1890-1898.

[15] Chen M, Wu QX, Cui RX. Terminal sliding mode tracking control for a class of SISO uncertain nonlinear systems. ISA T 2013; 52: 198-206.

[16] Kamal S, Moreno JA, Chalanga A, Bandyopadhyay B, Fridman LM. Continuous terminal sliding-mode controller. Automatica 2016; 69: 308-314.