Control and Cybernetics
vol. **54** (2025) No. 4
pages: 363-388
DOI: 10.2478/candc-2025-0013

Radial basis function neural network based higher order sliding mode control*

by

Vishal Mehra[1], Dipesh Shah[2] and Axaykumar Mehta[3]

[1]Instrumentation and Control Department, Gujarat Technological University
and Department of Agricultural Science,
College of AIT Anand Agricultural University, Anand, India
Vishal.mehra@nsitonline.in
[2]Alstom India, C V Raman Nagar,
Bengaluru, India
dipeshshah.ic@gmail.com
[3]Electrical and Computer Science Engineering Department
IITRAM, Ahmedabad, Gujarat, India
draxaymehta@gmail.com

Abstract: This paper presents a Radial Basis Function neural network based higher-order Sliding Mode Control for robust control of a dynamical system. A conventional sliding mode controller is suffering from a chattering problem, and the Super Twisting Algorithm is a special kind of higher-order Sliding Mode Control that has the capability of minimizing the chattering problem. There are also unknown model parameters and external disturbances that exert a negative influence on the control performance. To address the issues of model uncertainties and chattering, a Radial Basis Function (RBF) neural network based Super Twisting Algorithm is designed. The RBF neural network evaluates the model parameters and uncertainties, while the Super Twisting approach mitigates chattering, hence improving the controller's overall performance. Lyapunov stability based adaptive laws are derived for online updating of the parameters of the neural network. The proposed control algorithm was tested on a 2-degree-of-freedom serial flexible joint robotic arm to investigate its efficacy. The controller has a lower control chattering amplitude, lower control energy consumption, and a good tracking response, when compared to the RBF based conventional Sliding Mode controller and simple STA controller, as shown by the results.

Keywords: higher-order sliding mode control, Lyapunov stability, radial basis function, super-twisting algorithm

*Submitted: April 2025; Accepted: December 2025

1. Introduction

Sliding Mode Control (SMC) is a well-known and widely used method for controlling dynamical systems that are subject to disturbances and parameter variations (Utkin, 1977; Gao, 1993; Young, 1999). Sliding Mode Control has many advantages over the other control algorithms, but it suffers from chattering—a high-frequency vibration that causes low control efficiency and damage to the actuator (Slotine, 1984; Bartolini, 1989; Edwards and Spurgeon, 1998; Fridman, 2003). Numerous studies have reported efforts to reduce the impact of chattering. A suboptimal control is proposed for chattering reduction for the system with an actuator with a relative degree of one (Bartolini, 1998). A second-order sliding mode control (SOSMC) generalized suboptimal algorithm is proposed to reduce chattering and analyze it in state-space and frequency domains for linear and nonlinear systems (see Lee and Utkin, 2007). A frequency-shaped Sliding Mode Controller is suggested for vibration mitigation of an intelligent structure (Mehta and Bandyopadhyay, 2008).

A. Levant established that higher-order sliding modes can potentially minimize the chattering issue (Levant, 2010). The Twisting, Super Twisting, Suboptimal, and Prescribed Convergence Law algorithms are examples of second-order Sliding Mode Control. The Twisting, Suboptimal, and Prescribed convergence law algorithms needed information about the rate of change of the sliding variable, $\dot{\sigma}$, or, at least, the sign($\dot{\sigma}$) to ensure finite time convergence, but STA (Super Twisting Algorithm) does not require this information. So, STA became a popular remedy for chattering problems (Derafa, Benaleague and Fridman, 2012; Chalanga et al., 2016; Shah et al., 2020; Biricik, Komurcugil and Babaei, 2020).

Implementation of conventional Sliding Mode Control or STA requires complete information on the plant parameters; however, in many situations, plant parameters are not available, and deriving an exact mathematical model is very difficult. On the other hand, neural network has proved its efficiency in the nonlinear function approximation (Kumpati et al., 1990; Hunt, Sbarbaro and Gawthrop, 1992; Hornik, 1989) and has become popular among the intelligent control community.

A multi-model neural network-based adaptive controller has been designed for the pH neutralization process. The control structure consists of multiple identifiers and controller models, and a hysteresis switching algorithm was applied to choose the best neuro-identifier and controller (Yu, 2006). A Radial Basis Function neural network-based Sliding Mode Controller was developed for position control of a levitated object in a magnetic levitation system. An RBF-based estimator was designed for the online prediction of system uncertainties. An experiment's results show more robustness to uncertainties and better transient control performance (Lin, 2007). A neural network based slid-

ing mode adaptive control has been proposed for robust trajectory tracking for a robot manipulator with and without full robot dynamics that are measurable in the presence of uncertainties and disturbances (Sun et al., 2011).

An adaptive dynamic sliding mode control with a Recurrent Radial Basis Function (RRBF) based observer for a field-oriented induction motor drive has been proposed by Sousy and Fayez (2013). An RRBF was used to design an observer to estimate the unknown time-varying uncertainties. An adaptive learning algorithm was derived from Lyapunov stability theory for online updating of the parameters of the RRBF. An RBF-based adaptive sliding mode controller with three control loops for a three-level neutral-point-clamped (NPC) power converter was designed to track the desired value of active and reactive power, where the RBF neural network is used to estimate parametric uncertainties and the rate of desired power change (Yin et al., 2018).

Combining Radial Basis Function (RBF) networks with Sliding Mode Control (SMC) enables the approximation of unknown nonlinear functions or system dynamics, enhancing the controller's capacity to manage uncertainties. However, the chattering issue in conventional sliding mode control remains, due to the discontinuous switching terms, used for robustness. Because of the discontinuous switching term, the RBF-based SMC does not minimize chattering, an inherent feature of the Sliding Mode Control. To address this, the Super-Twisting Algorithm (STA), a second-order sliding mode control technique, having robust control and chattering minimization capacity, is integrated with the RBF as an effective solution for complex, nonlinear, and uncertain systems. In this work, we propose an adaptive RBF neural network based Super-Twisting controller for flexible joint robotic systems with complex and uncertain dynamics. Our approach targets flexible joint robots' complex and uncertain dynamics by integrating an adaptive RBF nn with the STA, unlike the output feedback framework using Levant's differentiator for general nonlinear systems (Truong and Chung, 2024). A primary advancement over recent works (Kumar et al., 2024; Truong and Chung, 2024) is the online adaptive tuning of both centers and widths of the RBFs, whereas earlier studies typically employ fixed parameters. Online real-time updates of the centroid and widths of RBF enhance real-time approximation and robustness, which improves tracking accuracy and compensation for time-varying uncertainties in the systems.

The major contribution of the paper is summarized as below:

- A new control methodology design that integrates Radial Basis Function (RBF) neural networks with the super-twisting algorithm, with real-time updates for weights, centers, and widths of RBF neural network to enhance the robustness and adaptability of the dynamical system while decreasing chattering.
- Stability analysis using Lyapunov-based techniques to guarantee conver-

gence and robustness, and real-time implementation of the proposed controller on a 2-degree-of-freedom serial flexible joints robotic arm to demonstrate the efficiency of the proposed approach.

This paper is organized into several sections as follows: Section 2 provides an overview of the second-order sliding modes of the Super-Twisting Algorithm. Section 3 describes a Radial Basis Function nn architecture. Section 4 provides a method to design RBF nn (neural network) based SOSMC-STA, Lyapunov stability analysis, and adaptive laws for online weight updating of RBF nn. Section 5 presents an experimental setup of a two-link robotic arm with serial flexible joints, mechanical configuration, specifications, results, and discussion. Section 6 presents the conclusion.

2. Problem statement

A second-order nonlinear system is given by

$$\ddot{x} = f(x, \dot{x}) + g(x, \dot{x})u + L, \tag{2.1}$$

where $x \in \mathbb{R}^n$ is a system state, u is a control input, $f(\cdot)$ and $g(\cdot)$ are nonlinear functions, with $g_{min} \leq g \leq g_{max}$, $g(\cdot) \neq 0$, and L is a disturbance that is globally bounded by a positive constant δ such that $|\dot{L}| \leq \delta$.

The error between the desired trajectory x_d and the actual trajectory x is given as,

$$e = x_d - x. \tag{2.2}$$

For designing the Super Twisting algorithm, define a sliding surface ζ as:

$$\zeta = \dot{e} + ke, \tag{2.3}$$

where $k > 0$.

The derivative of the sliding surface is

$$\dot{\zeta} = \ddot{e} + k\dot{e}, \tag{2.4}$$
$$= \ddot{x}_d - \ddot{x} + k\dot{e}, \tag{2.5}$$
$$= \ddot{x}_d - f(\cdot) - g(\cdot)u - L + k\dot{e}. \tag{2.6}$$

If the functions $f(\cdot)$ and $g(\cdot)$ are known in (2.6), then a Super Twisting sliding mode controller can be formulated as

$$u = \frac{1}{g}\left(-f + \ddot{x}_d + k\dot{e} + u_{swi}\right), \tag{2.7}$$

where

$$u_{swi} = \lambda_1 \sqrt{|\zeta|} \cdot \text{sign}(\zeta) + \lambda_2 \int \text{sign}(\zeta).$$

Upon substituting (2.7) into (2.6), the closed-loop system is expressed as

$$\dot{\zeta} = -\lambda_1 \sqrt{|\zeta|} \cdot \text{sign}(\zeta) + v, \tag{2.8}$$
$$\dot{v} = -\lambda_2 \cdot \text{sign}(\zeta) - \dot{L}, \tag{2.9}$$

where $\lambda_1 > 0$ and $\lambda_2 > 0$ are controller gains.

The above Super Twisting algorithm (2.7) can be implemented successfully if there is complete knowledge of the functions $f(\cdot)$ and $g(\cdot)$ that is available. But in many instances, either $f(\cdot)$ is not available or it is difficult to measure, and in such a scenario, it is necessary to approximate $f(\cdot)$ for effective control of the dynamical system.

3. Radial Basis Function neural network

Figure 1 depicts the structure of the RBF and the data flow to each layer. The RBF structure has an input layer, a hidden layer, and an output layer. The flow of information through each layer is described as follows:

- Input Layer: The input layer consists of the input features that are transmitted into the network. The number of input neurons in the input layer corresponds to the number of input features.
- Hidden layer: The hidden layer consists of neurons that have radial basis activation functions. Each neuron at the hidden layer calculates the output using the Gaussian function

$$\phi_j = \exp\left(-\frac{\|x_i - c_{ij}\|^2}{2b_j^2}\right), \tag{3.1}$$

where ϕ_j is the output of the j-th radial basis function node given input x_i. x_i is the input vector. c_{ij} is the center vector of the j-th RBF neuron, associated with the i-th input, given as

$$c_{ij} = \begin{bmatrix} c_{11} & . & . & . & . & c_{1j} \\ c_{i1} & . & . & . & . & c_{ij} \end{bmatrix} \in \mathbb{R}^{i \cdot j}$$

and b_j is the width (or spread) parameter of the j-th RBF neuron, given by

$$b_j = [b_1, b_2 ..... b_j] \in \mathbb{R}^j.$$

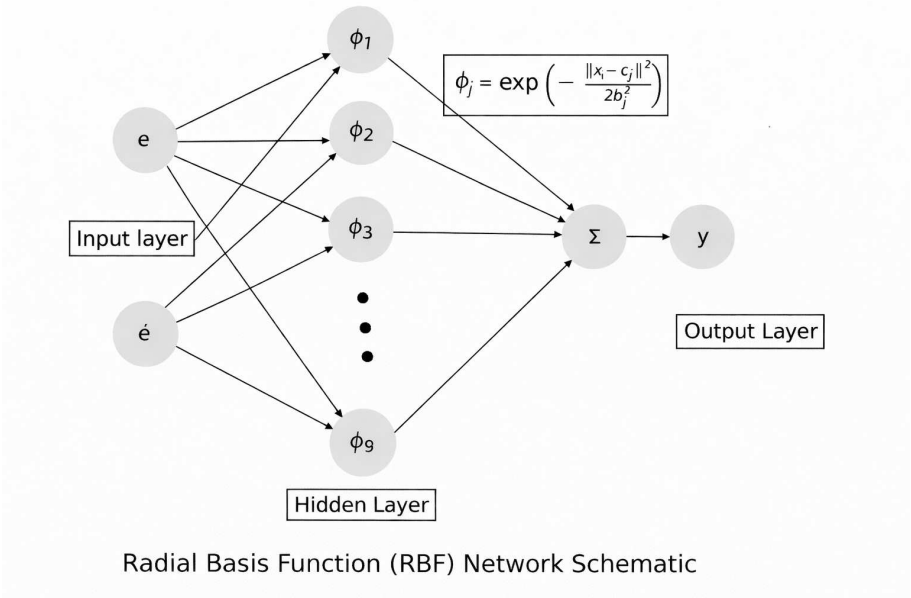

Figure 1: RBF neural network structure

- Output Layer: The output layer computes the final output by combining it with the hidden layer output and associated weights. Generally, the output layer has a linear activation function. It is computed as

$$Y = W_1\phi_1 + W_2\phi_2 + W_3\phi_3 + ... + W_j\phi_j \tag{3.2}$$

where

$$W = [W_1, W_2, ...W_j] \in R^j$$

is a weight of output layer.

Problem Statement: To design and implement the RBF nn based second order STA for the dynamical system by estimating modeling information for the plant with the unknown function $f(\cdot)$.

4. Design of STA using RBF neural network and stability analysis

This section presents the design of STA using the RBF nn structure, as per Theorem 1 given below.

THEOREM 1 *The Super Twisting controller for the system* (2.1) *with unknown system parameters is given as*

$$u = \frac{1}{g}\left(-\hat{f} + \ddot{\theta}_d + k\dot{e} + u_{swi}\right), \tag{4.1}$$

where $\hat{f}$ *is the estimation of the* f*, obtained by RBF neural network structure and* $g \neq 0$*. The unknown function* f*, computed by the RBF nn structure, can be written down as* $\hat{f} = \hat{W}^T\hat{\phi}(x, \hat{c}, \hat{b})$*.*

PROOF According to the universal approximation theorem (Hornik, 1989), any continuous function can be approximated by feed-forward neural networks.

ASSUMPTION 1 The exist optimal ideal weights W^*, centers c_{ij}^*, and widths b^* for approximating the unknown function f, such that we can write $f = W^{*T}\phi^* + \epsilon_n$, where ϵ_n is the approximation error of RBF nn, satisfying $|\epsilon_n| \leq \epsilon_b, \epsilon_b > 0$. So, the difference between the approximated value and the actual value of the unknown function f can be deduced to be

$$\begin{aligned}
\tilde{f} = f - \hat{f} &= W^{*T}\phi^* + \epsilon - \hat{W}^T\hat{\phi} \\
&= W^{*T}(\hat{\phi} + \tilde{\phi}) - \hat{W}^T\hat{\phi} + \epsilon \\
&= W^{*T}\hat{\phi} + W^{*T}\tilde{\phi} - \hat{W}^T\hat{\phi} + \epsilon \\
&= W^{*T}\hat{\phi} + (\tilde{W}^T + \hat{W}^T)\tilde{\phi} - \hat{W}^T\hat{\phi} + \epsilon \\
&= W^{*T}\hat{\phi} - \hat{W}^T\hat{\phi} + \hat{W}^T\tilde{\phi} + \tilde{W}^T\tilde{\phi} + \epsilon \\
&= \tilde{W}^T\hat{\phi} + \hat{W}^T\tilde{\phi} + \tilde{W}^T\tilde{\phi} + \epsilon \\
&= \tilde{W}^T\hat{\phi} + \hat{W}^T\tilde{\phi} + \epsilon_0
\end{aligned} \tag{4.2}$$

where $\tilde{W} = W^* - \hat{W}$, $\tilde{\phi} = \phi^* - \hat{\phi}$ and $\epsilon_0 = \tilde{W}^T\tilde{\phi} + \epsilon$ is the lumped approximation error.

The nonlinear activation function of the RBF network can be partially linearized using Taylor series expansion method. This enables the Lyapunov based adaptive stability analysis and the derivation of parameter update laws. The linearization of the activation function with respect to the centers and widths can be written down as

$$\begin{aligned}
\tilde{\phi} &= \frac{\partial\tilde{\phi}}{\partial c}|_{c=\hat{c}}(c^* - \hat{c}) + \frac{\partial\tilde{\phi}}{\partial b}|_{b=\hat{b}}(b^* - \hat{b}) + H.O.T. \\
\tilde{\phi} &= \Delta\phi_b\tilde{b} + \Delta\phi_c\tilde{c} + H.O.T.
\end{aligned} \tag{4.3}$$

where H.O.T. is a higher order term and $\tilde{c}, \tilde{b}$ are given as

$$\tilde{c} = \begin{bmatrix} \tilde{c}_{11} & \tilde{c}_{12} & \cdots & \tilde{c}_{1j} \\ \vdots & \vdots & \vdots & \\ \tilde{c}_{i1} & \tilde{c}_{22} & \cdots & \tilde{c}_{ij} \end{bmatrix} \in \mathbb{R}^{i \times j}$$

$$\tilde{b} = [\tilde{b}_1, \tilde{b}_2 ..... \tilde{b}_j] \in \mathbb{R}^j$$

where i and j are the indices of the input and RBF neurons in the neural network structure, respectively. The value of $\Delta\phi_c, \Delta\phi_b$ in (4.4) can be calculated as follows:

$$\Delta\phi_b = \begin{bmatrix} \frac{\partial\phi_1}{\partial b_1} & 0 & 0 & \cdots & 0_j \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \frac{\partial\phi_k}{\partial b_k} & \cdots & 0_j \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{\partial\phi_j}{\partial b_j} \end{bmatrix} \Delta\phi_c = \begin{bmatrix} \left(\frac{\partial\phi_1}{\partial c_1^T}\right) & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \left(\frac{\partial\phi_2}{\partial c_2^T}\right) & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \left(\frac{\partial\phi_j}{\partial c_j^T}\right) \end{bmatrix}$$

$$\in \mathbb{R}^{j \times (i \cdot j)}$$

The partial derivatives with respect to the RBF parameters are computed as:

$$\frac{\partial\phi_j}{\partial b_j} = \phi_j(\boldsymbol{x}) \cdot \frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|^2}{b_j^3}$$
$$\frac{\partial\phi_j}{\partial c_{ij}} = \phi_j(\boldsymbol{x}) \cdot \frac{(x_i - c_{ij})}{b_j^2}$$

The vector form of the center gradient is:

$$\frac{\partial\phi_j}{\partial \boldsymbol{c}_j^T} = \phi_j(\boldsymbol{x}) \cdot \frac{(\boldsymbol{x} - \boldsymbol{c}_j)^T}{b_j^2}$$

and so, by substituting (4.3) into (4.2), we get

$$\tilde{f} = f - \hat{f} = \tilde{W}^T\hat{\phi} + \hat{W}^T(\Delta\phi_b\tilde{b} + \Delta\phi_c\tilde{c} + HOT) + \epsilon_0. \tag{4.4}$$

When the neural network parameters are trained in such a way that $\tilde{f} = 0$, i.e., estimated function $\hat{f}$ and the real function f can be treated as identical,

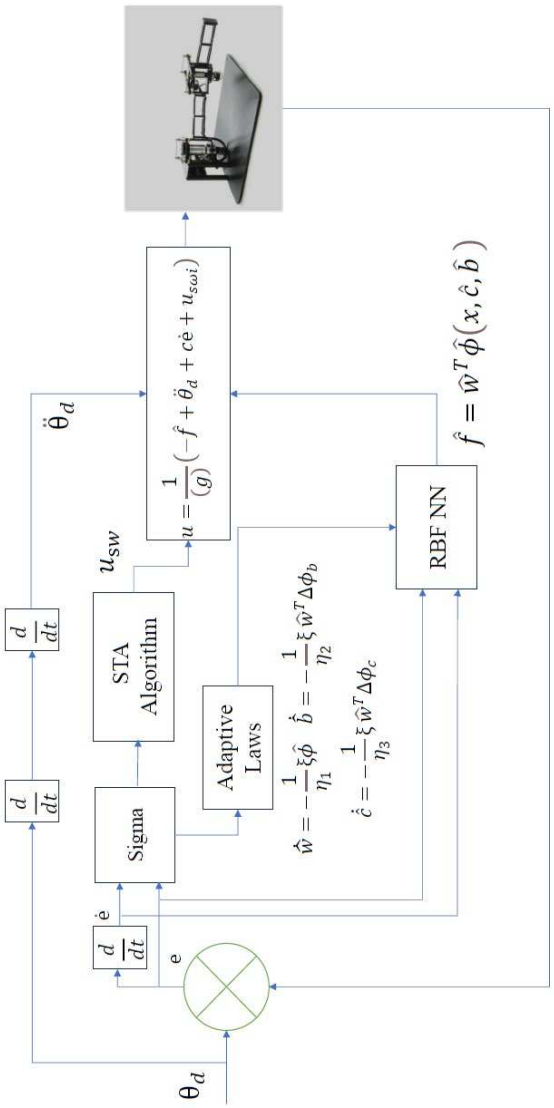

Figure 2: Block diagram for implementation of RBF based STA for position control of 2-DoF flexible joint robotic arm

the control equation (2.7) can be written down as

$$u = \frac{1}{g}\left(-\hat{f} + \ddot{x}_d + k\dot{e} + u_{swi}\right). \tag{4.5}$$

This completes the proof. ∎

The Lyapunov function is defined as follows to ascertain the stability and finite time convergence of the dynamical system:

$$V = \frac{1}{2}\zeta\zeta + \frac{1}{2}\eta_1(\tilde{W}^T\tilde{W}) + \frac{1}{2}\eta_2(\tilde{b}^T\tilde{b}) + \frac{1}{2}\eta_3 tr(\tilde{c}^T\tilde{c}). \tag{4.6}$$

Let us consider

$$\begin{aligned} N &= \frac{1}{2}\eta_1(\tilde{W}^T\tilde{W}) + \frac{1}{2}\eta_2(\tilde{b}^T\tilde{b}) + \frac{1}{2}\eta_3 tr(\tilde{c}^T\tilde{c}) \\ V &= \frac{1}{2}\zeta\zeta + N. \end{aligned} \tag{4.7}$$

we take the derivative of V,

$$\dot{V} = \zeta\dot{\zeta} + \dot{N}. \tag{4.8}$$

Placing the value of $\dot{\zeta}$ in the above equation leads to

$$\dot{V} = \zeta(\ddot{x}_d - f - g\cdot u - L + k\dot{e}) + \dot{N}. \tag{4.9}$$

Substituting for u from (4.5)

$$\begin{aligned} \dot{V} &= \zeta(\ddot{x}_d - f - g\cdot(\frac{1}{g}(-\hat{f} + \ddot{x}_d + k\dot{e} + u_{swi}) - L \\ &\qquad +k\dot{e}) + \dot{N} \\ &= \zeta(\ddot{x}_d - f + \hat{f} - \ddot{x}_d - k\dot{e} - u_{swi} - L + k\dot{e}) + \dot{N} \\ &= \zeta(-\tilde{f} - u_{swi} - L) + \dot{N} \\ &= \zeta(-\tilde{W}^T\hat{\phi} - \hat{W}^T\tilde{\phi} - \epsilon_0 - u_{swi} - L) + \dot{N}. \end{aligned} \tag{4.10}$$

Further, substituting the expression for $\tilde{\phi}$ from (4.3) into (4.10), gives

$$\begin{aligned} \dot{V} &= \zeta(-\tilde{W}^T\hat{\phi} - \hat{W}^T(\Delta\phi_b\tilde{b} + \Delta\phi_c\tilde{c} + H.O.T.) - \epsilon_0 \\ &\qquad -u_{sw} - L) + \eta_1(\tilde{W}^T\dot{\tilde{W}}) + \eta_2(\tilde{b}^T\dot{\tilde{b}}) + \eta_3 tr(\tilde{c}^T\dot{\tilde{c}}) \\ &= \zeta(-\tilde{W}^T\hat{\phi} - \hat{W}^T\Delta\phi_b\tilde{b} - \hat{W}^T\Delta\phi_c\tilde{c} - \hat{W}^T H.O.T. \\ &\qquad -\epsilon_0 - u_{sw} - L) + \eta_1(\tilde{W}^T\dot{\tilde{W}}) + \eta_2(\tilde{b}^T\dot{\tilde{b}}) + \eta_3 tr(\tilde{c}^T\dot{\tilde{c}}). \end{aligned} \tag{4.11}$$

Since $\dot{\tilde{W}} = -\dot{\hat{W}},\ \dot{\tilde{c}} = -\dot{\hat{c}},\ \dot{\tilde{b}} = -\dot{\hat{b}}$, the above equation becomes

$$\dot{V} = \zeta(-\tilde{W}^T\hat{\phi} - \hat{W}^T\Delta\phi_b\tilde{b} - \hat{W}^T\Delta\phi_c\tilde{c} - \hat{W}^T H.O.T. \\ -\epsilon_0 - u_{sw} - L) - \eta_1(\tilde{W}^T\dot{\hat{W}}) - \eta_2(\tilde{b}^T\dot{\hat{b}}) - \eta_3 tr(\tilde{c}^T\dot{\hat{c}}). \tag{4.12}$$

To minimize the tracking error and to ensure the stability of the closed-loop system, adaptive laws are designed in such a way that the parameter-dependent cross-terms are eliminated from the derivative of the Lyapunov function. This design approach ensures that the derivative of the Lyapunov function becomes negative semi-definite, guaranteeing system stability. The derived update rules are as follows:

$$\dot{\hat{W}} = -\frac{1}{\eta_1}\zeta\hat{\phi}, \tag{4.13}$$

$$\dot{\hat{b}} = -\frac{1}{\eta_2}\zeta\Delta\phi_b\hat{W}, \tag{4.14}$$

$$\dot{\hat{c}} = -\frac{1}{\eta_3}\zeta\Delta\phi_c^T\hat{W}, \tag{4.15}$$

Substitute equations (4.13) - (4.15) into (4.11), then $\dot{V}$ becomes

$$\dot{V} = \zeta(-\hat{W}^T \cdot H.O.T. - \epsilon_0 - u_{swi} - L). \tag{4.16}$$

The terms $\hat{W}^T \cdot H.O.T.$ and ϵ_0 are assumed to be uniformly bounded, so that

$$|\hat{W}^T \cdot \text{H.O.T.}| \leq \mathcal{H}, \quad |\epsilon_0| \leq \epsilon_u,$$

where $\mathcal{H}$ and ϵ_u are positive constants, representing the maximum bound on the higher-order terms and the approximation error, respectively. Now, consider, $(L + \mathcal{H} + \epsilon_u) = \gamma$, where $|(\dot{\gamma})| \leq \Gamma$, then

$$\begin{aligned}
\dot{V} &= -\zeta(u_{sw}+\gamma) \qquad (4.17)\\
&= -\zeta\left(\lambda_1\sqrt{|\zeta|}\mathrm{sgn}(\zeta)+\int\lambda_2\,\mathrm{sgn}(\zeta)\,dt+\gamma\right)\\
&= -\zeta\lambda_1\sqrt{|\zeta|}\mathrm{sgn}(\zeta)-\zeta\int\lambda_2\,\mathrm{sgn}(\zeta)\,dt-\zeta\gamma\\
&\leq -\lambda_1|\zeta|\sqrt{|\zeta|}-|\zeta|\int\lambda_2\,dt+|\zeta\gamma|\\
&= -\lambda_1|\zeta|\sqrt{|\zeta|}-|\zeta|\int\lambda_2\,dt+|\zeta||\gamma|\\
&= -\lambda_1|\zeta|\sqrt{|\zeta|}-|\zeta|\int\lambda_2\,dt+|\zeta|\left|\int\dot{\gamma}\,dt\right|\\
&= -\lambda_1|\zeta|\sqrt{|\zeta|}-|\zeta|\int\lambda_2\,dt+|\zeta|\int|\dot{\gamma}|\,dt. \qquad (4.18)
\end{aligned}$$

Consider now that $|\dot{\gamma}|\leq\Gamma<\lambda_2$, then the above equation can be rewritten as:

$$\begin{aligned}
\dot{V} &\leq -\lambda_1|\zeta|\sqrt{|\zeta|}-|\zeta|\int\lambda_2\,dt+|\zeta|\int\Gamma\,dt\\
\dot{V} &\leq -\lambda_1|\zeta|\sqrt{|\zeta|}-|\zeta|\int(\lambda_2-\Gamma)\,dt. \qquad (4.19)
\end{aligned}$$

If $\lambda_1>0$ and $\lambda_2>\Gamma$ then $-|\zeta|\int(\lambda_2-\Gamma)\,dt\leq 0$. So, for $\dot{V}$ we obtain

$$\dot{V} \leq -\lambda_1|\zeta|^{\frac{3}{2}}. \qquad (4.20)$$

Thus, the system trajectories reach the sliding surface in finite time, achieving robust finite-time convergence. Therefore, $V(t)\to 0$ in finite time, implying, $\zeta(t)\to 0$ in finite time. This ensures that the tracking error $e(t)$ also converges to zero in finite time.

From the Lyapunov function:

$$V=\frac{1}{2}\zeta^2+N \quad\Rightarrow\quad V\geq\frac{1}{2}\zeta^2 \quad\Rightarrow\quad |\zeta|\leq\sqrt{2V}. \qquad (4.21)$$

Now, substituting this into equation (4.20) yields:

$$\begin{aligned}
\dot{V} &\leq -\lambda_1(\sqrt{2V})^{3/2}\\
&\leq -\lambda_1(2V)^{3/4}\\
\dot{V} &\leq -\alpha V^{\beta}, \qquad (4.22)
\end{aligned}$$

where $\alpha = \lambda_1 \cdot 2^{3/4}, \qquad \beta = \frac{3}{4}$.

From (4.22), the convergence time is given by

$$T \leq \frac{V(0)^{1-\beta}}{\alpha(1-\beta)} = \frac{V(0)^{1/4}}{\lambda_1 \cdot 2^{3/4} \cdot \left(\frac{1}{4}\right)} = \frac{4V(0)^{1/4}}{\lambda_1 \cdot 2^{3/4}}. \tag{4.23}$$

Figure 3: 2-DOF serial flexible joint setup

5. Hardware experimentation and results

The effectiveness of the proposed RBF nn based STA controller was experimentally verified on the laboratory setup of a 2-DOF serial flexible joint (SFJ) robotic arm, developed by QUANSER (QUANSER, 2015). The system, as depicted in Fig. 3, consists of two rigid bars, serially connected by two flexible joints (joints 1 and 2) via a common link. These joints are driven by two separate actuators that rotate the bars. The position of the flexible joints is measured by quadrature optical encoders, and the velocity, error, and acceleration information are derived from the position data provided by the encoders, and it is controlled by two distinct DC motors. The system is interfaced with a PC via a data acquisition board and current amplifier using MATLAB/SIMULINK and the QUARC Real-Time interface software. The coupling effect was neglected

during the design of the controller, and two individual control loops were implemented to control the motors.

The mathematical model of the two joints is given in QUANSER (2015), while the information about the physical parameters of the plant is given in Table 1:

$$\ddot{x}_{11} = -\frac{K_{s1}}{J_{11}}x_{11} + \frac{K_{s1}}{J_{11}}x_{12} - \frac{B_{11}}{J_{11}}\dot{x}_{11} + \frac{u}{J_{11}}. \tag{5.1}$$

$$\ddot{x}_{12} = \frac{K_{s1}}{J_{12}}x_{11} - \frac{K_{s1}}{J_{12}}x_{12} - \frac{B_{12}}{J_{12}}\dot{x}_{11}. \tag{5.2}$$

$$\ddot{x}_{21} = -\frac{K_{s2}}{J_{21}}x_{21} + \frac{K_{s2}}{J_{22}}x_{22} - \frac{B_{21}}{J_{21}}\dot{x}_{21} + \frac{u}{J_{21}}. \tag{5.3}$$

$$\ddot{x}_{22} = \frac{K_{s2}}{J_{22}}x_{21} - \frac{K_{s2}}{J_{22}}x_{22} - \frac{B_{22}}{J_{22}}\dot{x}_{22}.$$

The above can be written as

$$\begin{aligned}
\ddot{x}_{11} &= f(x_{11}) + g(x_{11}) * u,\\
\ddot{x}_{12} &= f(x_{12}),\\
\ddot{x}_{21} &= f(x_{21}) + g(x_{21}) * u,\\
\ddot{x}_{22} &= f(x_{22}),
\end{aligned}$$

where

$$\begin{aligned}
f(x_{11}) &= -\frac{K_{s1}}{J_{11}}x_{11} + \frac{K_{s1}}{J_{11}}x_{12} - \frac{B_{11}}{J_{11}}\dot{x}_{11}\\
g(x_{11}) &= \frac{1}{J_{11}}.\\
f(x_{12}) &= \frac{K_{s1}}{J_{12}}x_{11} - \frac{K_{s1}}{J_{12}}x_{12} - \frac{B_{12}}{J_{12}}\dot{x}_{11}\\
f(x_{21}) &= -\frac{K_{s2}}{J_{21}}x_{21} + \frac{K_{s2}}{J_{22}}x_{22} - \frac{B_{21}}{J_{21}}\dot{x}_{21}\\
g(x_{21}) &= \frac{1}{J_{21}}\\
f(x_{22}) &= \frac{K_{s2}}{J_{22}}x_{21} - \frac{K_{s2}}{J_{22}}x_{22} - \frac{B_{22}}{J_{22}}\dot{x}_{22}.
\end{aligned}$$

A 2-9-1 RBF nn architecture is integrated with a Super Twisting Algorithm to employ an intelligent controller. The proposed nn architecture has the input layer with two inputs and the first hidden layer with 9 Radial Basis Neurons, and the output layer has one linear neuron.

Table 1: The 2-DOF SFJ robot model physical specifications (QUANSER, 2015)

Parameter	Value	Units
J_{11}	0.0637	kg·m^2
B_{11}	4.5	N·m·s/rad
J_{12}	0.2304	kg·m^2
B_{12}	0.0704	N·m·s/rad
K_{s1}	9	N·m/rad
J_{21}	0.0035	kg·m^2
B_{21}	0.5	N·m·s/rad
J_{22}	0.0107	kg·m^2
B_{22}	0.0282	N·m·s/rad
K_{s2}	4	N·m/rad

The initial weights of the RBF input and hidden layers are selected as small random values, i.e., $W = 0.1{*}\text{rand}(1, 9)$. The input to the RBF network consists of error and its derivative, and the robot operates within a range of approximately ± 0.4 radians, so that the centers of the basis functions are uniformly distributed across the normalized domain and defined as

$$c_{ij} = 0.4 * \begin{bmatrix} -1 & -0.75 & -0.5 & -0.25 & 0 & 0.25 & 0.5 & 0.75 & 1 \\ -1 & -0.75 & -0.5 & -0.25 & 0 & 0.25 & 0.5 & 0.75 & 1 \end{bmatrix},$$

and to ensure smooth and generalizable function approximation over the large operating region, the widths of the Gaussian basis functions are selected as:

$$b_j = \begin{bmatrix} 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \end{bmatrix}.$$

The adaptive parameters of update rules, η_1 and η_2 to η_3, are chosen as 0.01 and 0.001 for joint 1, and 0.001 and 0.01 for joint 2. The value of Super Twisting gain λ_1 and λ_2 is chosen as 1 and 110, respectively, for joint 1 and 10 and 110, respectively, for joint 2. The performance of the proposed controller is compared with RBF nn based SMC with the same size of the input, hidden, and output layers and the same initial weights used in the proposed structure and with the conventional Super Twisting Algorithm without a neural network. The gain for the SMC is selected as 15 and 20 for joints 1 and 2, and for conventional STA 50 and 300 for stage 1, and 100 and 350 for stage 2. The desired trajectory is set as a square wave of $\pm 20^\circ$ angular position and frequency of 0.1 Hz. The initial value of the state is chosen as [0 0 0 0]. The random signal is chosen as an external disturbance. There are $\pm 20\%$ parameter variations for the plant parameters with respect to their nominal values.

Figures 4 and 5 show the square wave tracking response of the conventional STA for joints 1 and 2, and Figs. 6 and 7 show the tracking response for the

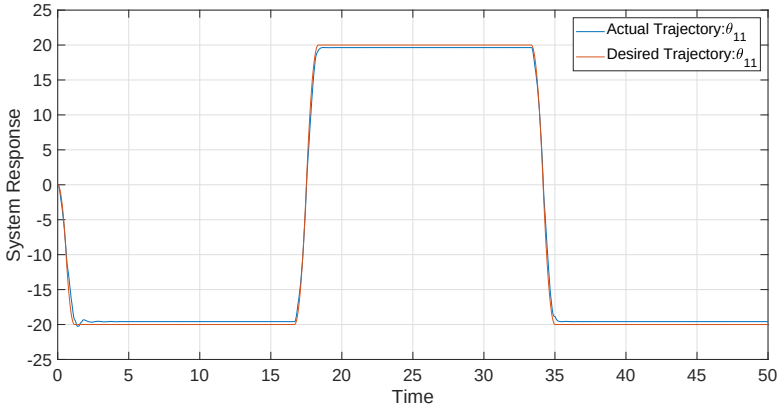

Figure 4: Position tracking response of conventional STA for joint 1

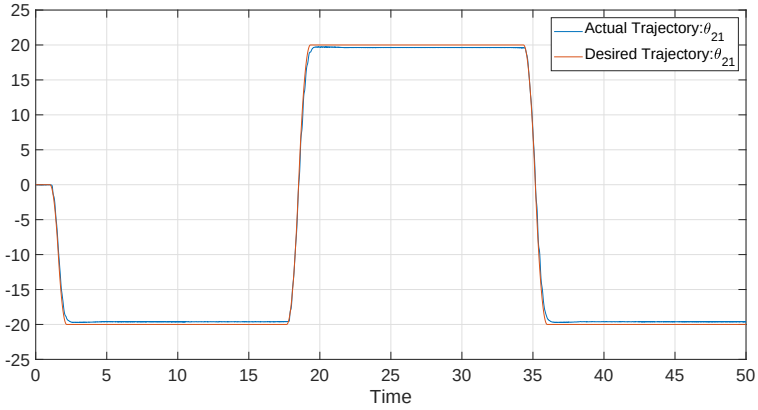

Figure 5: Position tracking response of conventional STA for joint 2

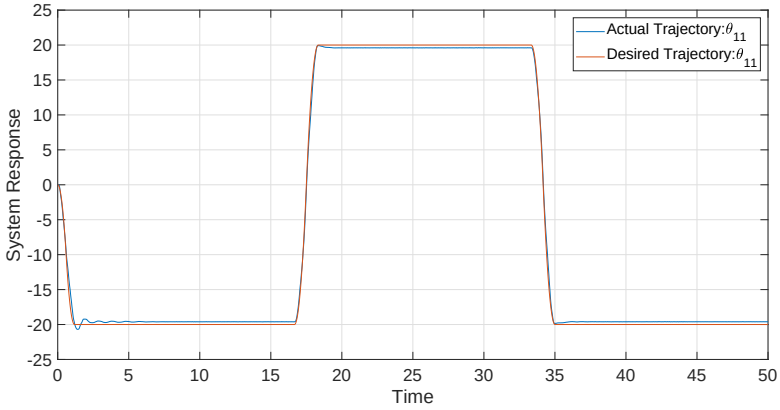

Figure 6: Position tracking response of RBF based SMC for joint 1

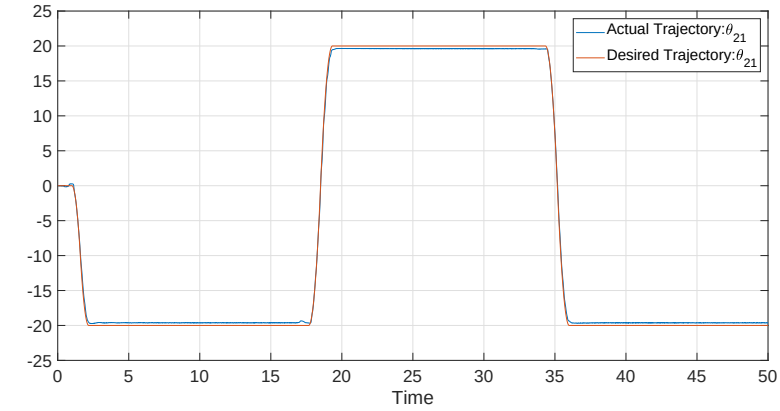

Figure 7: Position tracking response of RBF based SMC for joint 2

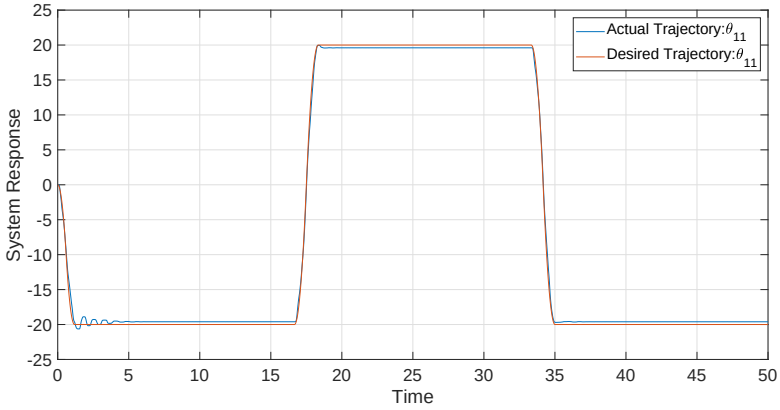

Figure 8: Position tracking response of RBF based STA for joint 1

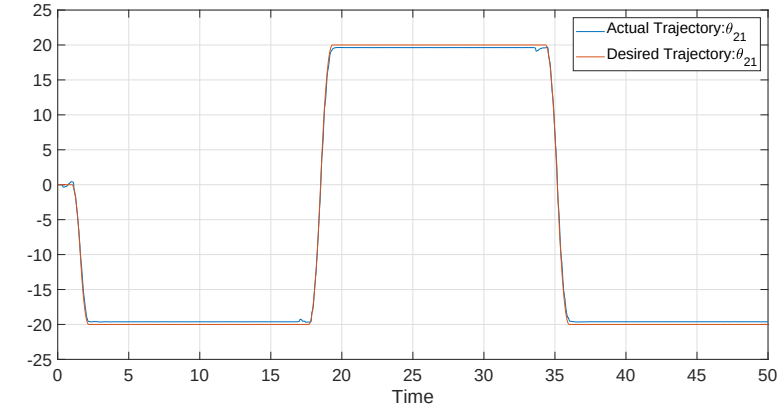

Figure 9: Position tracking response of RBF based STA for joint 2

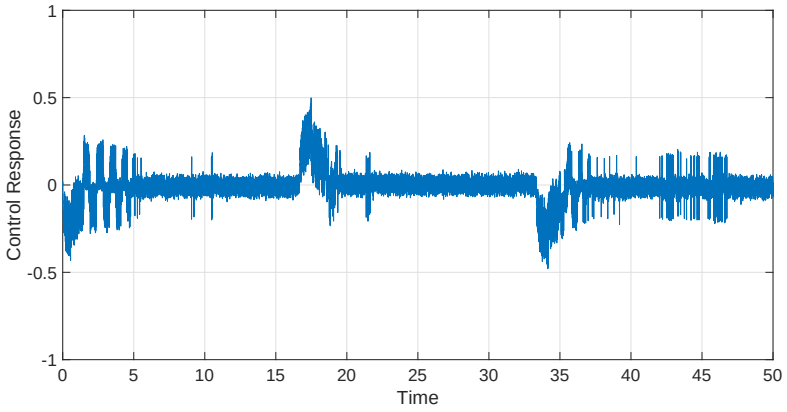

Figure 10: Control effort of conventional STA for joint 1

RBF nn based SMC, while Figs. 8 and 9 show the tracking response for the RBF nn based STA. As shown in the figures, it can be seen that the RBF nn based SMC has a steady-state error of 0.4 degrees and 0.39 degrees for joints 1 and 2, whereas the RBF nn based STA has a steady-state error of 0.4 degrees and 0.38 degrees for joints 1 and 2, respectively, while the conventional STA has a steady-state error of 0.41 degrees and of 0.38 degrees for joints 1 and 2.

Figures 10 and 11 present the control efforts from the conventional super twisting algorithm, and Figs. 12 and 13 show the control signal generated by RBF based Sliding Mode Control (SMC), whereas Figs. 14 and 15 illustrate the control efforts exerted on joints 1 and 2 by the proposed RBF based Super-Twisting Algorithm (STA).

Table 2 presents a comprehensive performance comparison of three distinct controllers—Super-Twisting Algorithm (STA), Radial Basis Function-Sliding Mode Control (RBF-SMC), and Radial Basis Function-Super-Twisting Algorithm (RBF-STA)—for joints 1 and 2. These controllers are evaluated on three key metrics: Control Effort (J), Integral of Squared Error (ISE), and Chattering Amplitude.

Metric 1. Control Effort (J) RBF-STA consistently demonstrates the lowest control effort for both joints, indicating higher energy efficiency compared to STA and RBF-SMC.

- For **joint-1**, RBF-STA requires 0.1588 J, which is approximately by 37.4% less than for STA (0.2536 J) and by 77.6% less than by RBF-SMC (0.7093 J).
- For **joint-2**, RBF-STA exhibits the control effort of 1.8414 J, representing

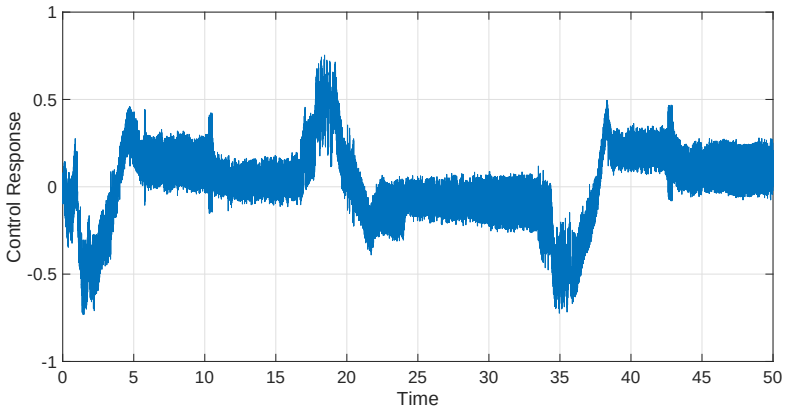

Figure 11: Control effort of conventional STA for joint 2

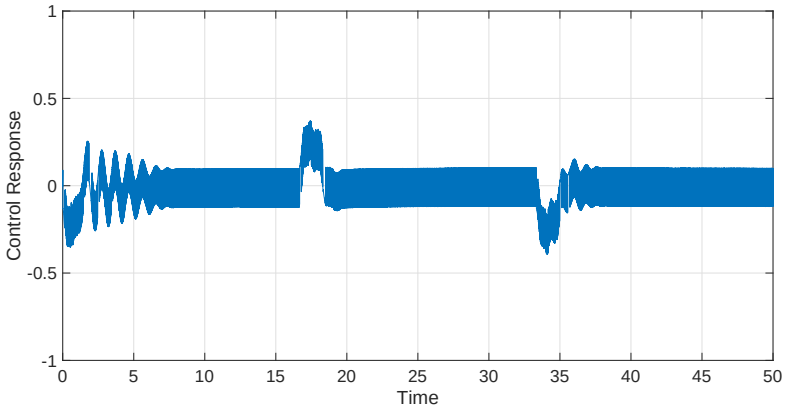

Figure 12: Control effort of RBF based SMC for joint 1

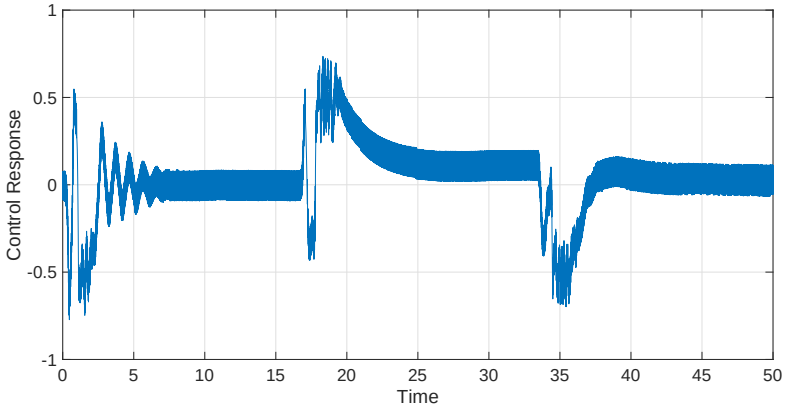

Figure 13: Control effort of RBF based SMC for joint 2

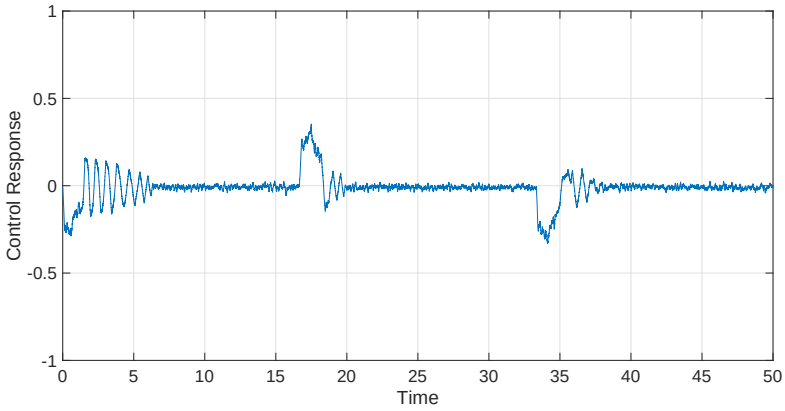

Figure 14: Control effort of RBF based STA for joint 1

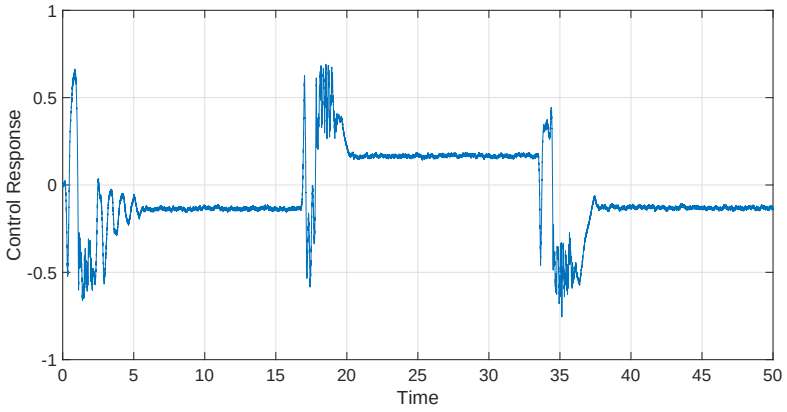

Figure 15: Control effort of RBF based STA for joint 2

14% of reduction when compared to STA (2.1413 J) and 27% of reduction when compared to RBF-SMC (2.5255 J).

RBF-SMC required the highest control effort for both joints, suggesting it is the least energy-efficient among the tested controllers. STA offers better efficiency than RBF-SMC, but worse than that of RBF-STA.

Metric 2. Integral of Squared Error (ISE) All three controllers achieve very low ISE values, indicating accurate tracking. RBF-STA displays the lowest ISE, suggesting superior tracking performance.

- For joint-1, RBF-STA and RBF-SMC achieve identical ISE values of 0.0036, which is 18.2% lower than for STA (0.0044). This suggests that the RBF network effectively enhances tracking accuracy for joint-1.
- For joint-2, RBF-STA demonstrates the lowest ISE at 0.0025, closely followed by RBF-SMC at 0.0026. STA lags slightly with the ISE value of 0.0033. Specifically, RBF-STA's ISE is by 24.2% lower than that of STA and by 3.8% lower than that of RBF-SMC for joint-2.

While RBF-STA generally shows the best tracking, the differences in ISE across controllers are relatively small, indicating that all three effectively minimize the tracking error.

Metric 3. Chattering Amplitude Chattering amplitude is a critical metric for practical implementation, representing unwanted high-frequency oscillations. RBF-STA outperforms the other controllers in chattering suppression.

- For joint-1, RBF-STA achieves an impressive chattering amplitude of 0.04, which represents a remarkable 60% reduction compared to STA (0.1) and an 80% reduction relative to RBF-SMC (0.2).

- For joint-2, the chattering reduction by RBF-STA is even more significant, with an amplitude of 0.015. This is a staggering 95% reduction compared to STA (0.3) and a 91.2% reduction compared to RBF-SMC (0.17).

STA and RBF-SMC exhibit noticeably higher chattering amplitudes, particularly for joint-2. The RBF-STA controller excels in minimizing chattering, a significant advantage for real-world applications, where smooth control is desired and actuator wear needs to be minimized.

Based on this quantitative analysis, RBF-STA emerges as the most effective controller across almost all metrics for both joints. It achieves the lowest control effort, leading to improved energy efficiency. It also shows superior tracking accuracy with the lowest ISE values. Most importantly, RBF-STA achieves significantly reduced chattering amplitude, making it a highly desirable choice for applications where smooth and precise control is required and harmful oscillations need to be mitigated. While STA and RBF-SMC offer reasonable tracking performance, their higher control efforts and chattering amplitudes make them less favorable than RBF-STA.

In summary, the proposed RBF neural network based STA combines the strengths of adaptive neural networks and higher-order sliding mode control to deliver superior performance in terms of robustness, energy efficiency, smoothness of control, and tracking accuracy, making it a more effective control strategy for nonlinear systems.

Table 2: Performance comparison of controllers for both joints

Controller	**joint**	**Control Effort (J)**	**ISE**	**Chattering Amplitude**
STA	1	0.2536	0.0044	0.1
	2	2.1413	0.0033	0.3
RBF-SMC	1	0.7093	0.0036	0.2
	2	2.5255	0.0026	0.17
RBF-STA	1	0.1588	0.0036	0.04
	2	1.8414	0.0025	0.015

6. Conclusion

This paper presents an intelligent second-order sliding mode controller designed for nonlinear dynamical systems. The proposed control scheme combines the Super-Twisting Algorithm with Radial Basis Function neural networks, resulting in a robust, smooth, and energy-efficient control strategy. While traditional

Sliding Mode Control suffers from chattering, due to its discontinuous control action, using STA mitigates this issue by providing continuous control signals. Furthermore, the integration of RBF nn enhances the controller's ability to approximate unknown system dynamics and handle nonlinearities, uncertainties, and external disturbances. The proposed RBF based STA employs Lyapunov-based adaptive laws for updating the weights, centers, and widths of the Gaussian basis functions. This enables the network to dynamically adjust in real time according to the system's behavior, leading to more accurate function approximation, smooth and energy efficient control, good tracking performance, and increased robustness.

References

BARTOLINI, G. (1989) Chattering phenomena in discontinuous control systems. *International Journal of Systems Science*, **20**(12), 2471–2481.

BARTOLINI, G., FERRARA, A. and USAI, E. (1998) Chattering avoidance by second-order sliding mode control. *IEEE Transactions on Automatic Control*, **43**(2), 241–246.

BIRICIK, S., KOMURCUGIL, H.A. and BABAEI, K. (2020) Super twisting sliding-mode control of DVR with frequency-adaptive Brockett oscillator. *IEEE Transactions on Industrial Electronics*, **68**(11), 10730–10739.

CHALANGA, A., KAMAL, S., MORENO, L.M.F.M.L.B.B. and MORENO, J.A. (2016) Implementation of super-twisting control: Super-twisting and higher order sliding-mode observer-based approaches. *IEEE Transactions on Industrial Electronics*, **63**(6), 3677–3685.

DERAFA, L., BENALLEAGUE, A. and FRIDMAN, L. (2012) Super twisting control algorithm for the attitude tracking of a four rotors UAV. *Journal of the Franklin Institute*, **349**(2), 685–699.

EDWARDS, C. and SPURGEON, S. (1998) *Sliding Mode Control: Theory and Applications*. CRC Press.

FRIDMAN, L.M. (2003) Chattering analysis in sliding mode systems with inertial sensors. *International Journal of Control*, **76**(9-10), 906–912.

GAO, W. and HUNG, J.C. (1993) Variable structure control of nonlinear systems: A new approach. *IEEE Transactions on Industrial Electronics*, **40**(1), 45–55.

HORNIK, K. (1989) Multilayer feed-forward networks are universal approximators. *Neural Networks*, **2**(5), 359–366. doi:10.1016/0893-6080(89)90020-8.

HUNT, K.J., SBARBARO, D. and GAWTHROP, P.J. (1992) Neural networks for control systems—A survey. *Automatica*, **28**(6), 1083–1112.

KUMAR, B., SWAIN, S. K., GHOSH, S., MISHRA, S. K. and SINGH, Y. K. (2024) Radial basis function-based adaptive gain super-twisting controller

for magnetic levitation system with time-varying external disturbance. *IEEE Transactions on Transportation Electrification*, **10**(4), 9121-9132.

KUMPATI, S.N., KANNAN, P. et al. (1990) Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, **1**(1), 4–27.

LEE, H. AND UTKIN, V.I. (2007) Chattering suppression methods in sliding mode control systems. *Annual Reviews in Control*, **31**(2), 179–188.

LEVANT, A. (2010) Chattering analysis. *IEEE Transactions on Automatic Control*, **55**(6), 1380–1389.

LIN, F.J., TENG, L.T. AND SHIEH, P.H. (2007) Intelligent sliding-mode control using RBFN for magnetic levitation system. *IEEE Transactions on Industrial Electronics*, **54**(3), 1752–1762.

MEHTA, A.J., AND BANDYOPADHYAY, B. (2008) Frequency-shaped sliding mode control using output sampled measurements. *IEEE Transactions on Industrial Electronics*, **56**(1), 28–35.

QUANSER INC. (2015) *2 DoF Serial Flexible Joint Robot - Reference and User Manual*.

SHAH, D., PATEL, K., MEHTA A. AND BARTOSZEWICZ, A. (2020) Event-triggered discrete higher-order SMC for networked control system having network irregularities. *IEEE Transactions on Industrial Informatics*, **16**(11), 6837–6847.

SLOTINE, J.J.E. (1984) Sliding controller design for nonlinear systems. *International Journal of Control*, **40**(2), 421–434.

SOUSY, E.L. AND FAYEZ .M.F. (2013) Adaptive dynamic sliding-mode control system using recurrent RBFN for high-performance induction motor servo drive. *IEEE Transactions on Industrial Informatics*, **9**(4), 1922–1936.

SUN, T., PEI, H., PAN, Y., ZHOU, H. AND ZHANG, C. (2011) Neural network-based sliding mode adaptive control for robot manipulators. *Neurocomputing*, **74**(14-15), 2377-2384.

TRUONG, H. V. A. AND CHUNG, W. K. (2024) Sliding-mode-based output feedback neural network control for electro-hydraulic actuators subject to unknown dynamics and uncertainties. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, **54**12, 7884-7896.

UTKIN, V.I. (1977) Variable structure systems with sliding modes. *IEEE Transactions on Automatic Control*, **22**(2), 212-222.

YIN, Y., LIU, J., VAZQUEZ, J.A.S.L.W., LEON, J.I. AND FRANQUELO, L.G. (2018) Observer-based adaptive sliding mode control of NPC converters: An RBF neural network approach. *IEEE Transactions on Power Electronics*, **34**(4), 3831-3841.

YOUNG, K.D., UTKIN, V.I. AND OZGUNER, U. (1999) A control engineer's guide to sliding mode control. *IEEE Transactions on Control Systems*

Technology, **7**(3), 328–342.
YU, W. (2006) Multiple recurrent neural networks for stable adaptive control. *Neurocomputing*, **70**(1-3), 430–444.