SLIDING MODE NEURO-ADAPTIVE CONTROLLER DESIGNED IN DISCRETE TIME FOR MOBILE ROBOTS

Francisco G. Rossomando,* Carlos Soria,* Eduardo O. Freire,** and Ricardo O. Carelli*

Abstract

It is well known that all robotic systems have additional dynamics and disturbances beyond those that are modelled. Hence, in this paper, the performance of a neuro-adaptive sliding mode control (NA-SMC) controller is analysed in the presence of disturbances and unmodelled dynamics. The proposed control strategy has two stages: first, a feedback linearization controller for a kinematic model; second, a neuro-adaptive SMC controller for a dynamic model. The entire control strategy is designed in discrete time using Lyapunov's criterion, and the stability problems caused by direct implementation in discrete time to a system designed in a continuous domain are thereby avoided. The unmodelled dynamics introduce tracking errors in the closed-loop system; however, experiments made using the proposed approach to control mobile robots show that output tracking error tends to zero.

Key Words

Adaptive neural control, nonlinear systems, mobile robot, sliding mode control.

1. Introduction

Due to the increase in mobile robot applications in both industry and services, there is plenty of research to refer regarding mobile robot control [1]–[4]. In robotic control, whenever a task must be performed that requires acceleration/deceleration, with/without friction and mass variations, it is necessary to consider the mobile robot dynamics as well as its kinematics. Recent works dealing with robot dynamics are listed in [1]–[4].

Literature involving the interaction between the kinematic controller and the dynamic controller for a mobile robot is already present [1], [5], [6]. From there, we can learn that dynamics compensation requires estimation of the robot's dynamics, and that neural nets, fuzzy logic and other AI techniques, with their excellent function mapping capability, have proven to be a suitable tool for obtaining inverse nonlinear dynamic systems [1], [5]. A general approach found in this literature is to use a neural network (NN) to identify the unknown nonlinear dynamics to obtain the dynamics compensation control law. In addition, to improve the overall performance, an adaptive capability can be introduced whenever necessary to model the system with the best possible accuracy. Regardless of the uncertainties or changes in dynamics parameters, the desired trajectory must be followed with acceptable precision.

1.1 Related Works

The work presented in [1] includes an adaptive capability, was designed in continuous time and employs a PI control term to reduce the external disturbances. The proposed controller is fully tuned and considers the neural weights, centres and spreads, as well as the PI parameters; but, it needs a linear approximation of the radial basis function (RBF) neurons to obtain the learning rules.

Reference [4] presents a good proposal that uses compound-cosine-function neural networks developed in continuous time and yields good simulation results. Another proposal of discrete-time inverse control that uses neural control is done by [6]. This work combines inverse optimal control with sliding mode for mobile robots and uses an EKF observer to obtain the state variables; the main objective being to obtain an equivalent Hamilton Jacobi Bellman (HJB) equation for optimal control.

An adaptive neural tracking control is designed for a class of multiple-input-multiple-output (MIMO) nonlinear systems [7]. This controller is implemented in discrete-time with static nonlinearities in its inputs (saturation and dead zones). The neural RBFs are used to identify the unknown dynamical structure of the systems to be controlled. Through a Lyapunov method, the control technique is proven to be uniformly semi-global and ultimately bounded. This work displays some simulation examples, though it does not include the SMC theory.

Based on discrete time analysis, an indirect data-driven method for the trajectory tracking control problem

* Instituto de Automática (INAUT-Conicet), National University of San Juan Capital, San Juan, Argentina; e-mail: {frosoma, csoria, rcarelli}@inaut.unsj.edu.ar

** Department of Electrical Engineering, Universidade Federal de Sergipe, São Cristóvão, Brazil; e-mail: efreire@ufs.edu

of a class of nonlinear discrete-time systems is developed in [8], which have unknown dynamics. Based on Lagrange's mean value theorem, this work gives an online linearization technique which is applicable to nonlinear discrete-time systems, whose dynamic models have continuous partial derivatives with respect to the input and the output. The basic principle of the proposed method is establishing an approximate model of the original system offline, using recorded input–output data and NN. The simulation results demonstrate that the output asymptotically converges to the reference trajectory.

1.2 Contribution

This paper proposes a neuro-adaptive sliding mode control (NA-SMC) controller to deal with the problem of disturbances and unmodelled dynamics. This work differs in its contribution from that of [9], [10] in the fact that in those works the analysis of the robot arm is not divided between kinematics and dynamics parts, but rather the control law of the dynamical structure is all that is considered. Here, the entire control technique is implemented in cascade mode for a mobile robot, with a kinematic controller actually developed from the robot's inverse kinematics. Specifically, the contribution proposed in this paper states that both controllers can operate simultaneously without any instability caused by external disturbances, by non-modelled dynamics and parametric variations, or by erroneous interactions between controllers. Furthermore, this work cannot be considered a simple discretization because the controller is designed while considering adaptive capacity by the use of a radial basis function neural network (NN-RBF) to attenuate all dynamic variations of WMR.

The proposed control technique has the following advantages:

- Introduces a stability analysis for an adaptive neural controller designed entirely in nonlinear discrete time using the Lyapunov theory without other condition or criterion.
- The proposed technique does not need any previous knowledge of system dynamics, being the main advantage of the proposed approach over others that are model based. Moreover, it can be tuned online by adjusting controller's parameters (neural weights).
- The chosen control technique uses a neuro-adaptive network, which is responsible to learn the inverse dynamics of the wheeled mobile robot (WMR). It adds a sliding surface compensation to remove the residual error introduced by the neuro-adaptive network.

This control scheme, compared to others in the literature, generates control actions in terms of velocities (angular and linear), but not in terms of torques, as that is the usual mode of commercial mobile robots.

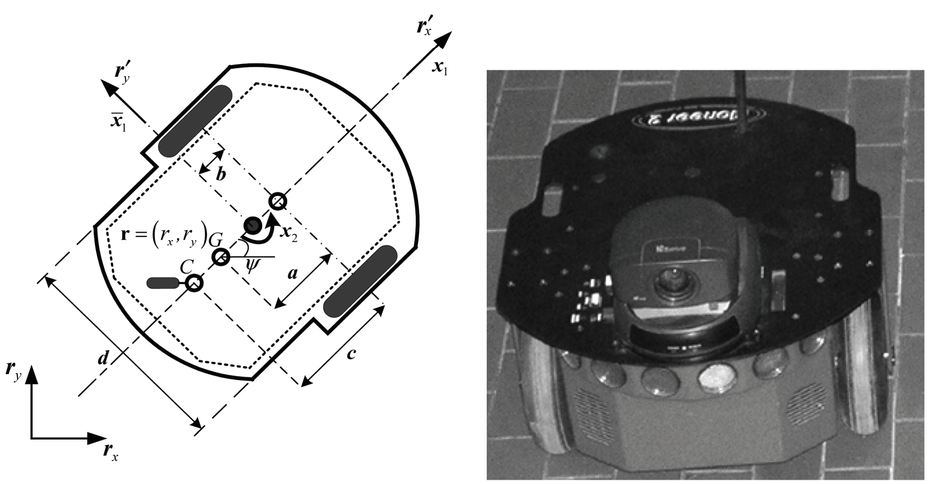
1.3 Organization

The paper is organized as follows. In Section 2, a description of the non-holonomic mobile robot system is given and the system model is proposed. Explicit expressions of the kinematics SMC and the discrete time sliding mode control are derived in Sections 3 and 4, respectively. Experiments based on mobile robots are presented in Section 5 for the evaluation of this control technique, demonstrating the performance of the proposed algorithm in the presence of dynamic variations and uncertainties. Finally, Section 6 presents the conclusions.

2. Mobile Robot Description

In this section, the discrete mathematical models (1) and (2) are described. They were proposed by [5] and this mathematical implementation allows the linear and angular reference velocities to be considered as the input signals. In these equations, T_0 is the sampled time and k is the discrete time. The identified parameters for the Pioneer 2DX mobile robot (κ_1 to κ_6) are described in [5].

Figure 1(a) shows the non-holonomic mobile robot with parameters and variables of interest, where u_1 and u_2 are controlled action inputs, the coordinates r_x, r_y define the position of WMR on the XY plane, expressed in



(a) (b)

Figure 1. (a) Mobile robot description and (b) Pioneer 2DX mobile robot.

vectorial form as $\mathbf{r}=(r_x, r_y)$, and the angle ψ represents the robot's orientation on the XY plane. The output speeds are denoted as x_1 and x_2, and are the linear and angular output speeds, respectively.

Other robot parameters are c is the location of the rear free wheel (castor wheel), G is the mass centre and a denotes the distance between point r and the virtual axis of the traction wheels.

The mathematical models are given as follows.

Discrete kinematics model

$$\begin{pmatrix} r_x(k+1) \\ r_y(k+1) \\ \psi(k+1) \end{pmatrix} = T_0 \begin{pmatrix} \cos\psi(k) & -a\sin\psi(k) \\ \sin\psi(k) & a\cos\psi(k) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix} + \begin{pmatrix} r_x(k) \\ r_y(k) \\ \psi(k) \end{pmatrix} + \begin{pmatrix} \delta_{r1}(k) \\ \delta_{r2}(k) \\ 0 \end{pmatrix} \tag{1}$$

Discrete dynamics model

$$\begin{pmatrix} x_1(k+1) \\ x_2(k+1) \end{pmatrix} = \begin{pmatrix} \kappa_3\kappa_1^{-1}x_2^2(k) + \kappa_4\kappa_1^{-1}x_1(k) \\ -\kappa_5\kappa_2^{-1}x_1(k)\,x_2(k) + \kappa_6\kappa_2^{-1}x_2(k) \end{pmatrix} + \begin{pmatrix} \kappa_1^{-1} & 0 \\ 0 & \kappa_2^{-1} \end{pmatrix} \begin{pmatrix} u_1(k) \\ u_2(k) \end{pmatrix} + \begin{pmatrix} \delta_{x1}(k) \\ \delta_{x2}(k) \end{pmatrix} \tag{2}$$

The uncertainties associated with the WMR are δ_{r1}, δ_{r2}, δ_{x1} and δ_{x2}, where δ_{r1} and δ_{r2} are functions of slip velocities and robot orientation, δ_{x1} and δ_{x2} depend on physical parameters, for instance: robot–load mass, inertia, wheel and tyres' dimensions, DC motor and its electrical actuators parameters, reaction forces on the wheels, *etc.* These parameters are considered as external disturbances.

The robot's model has been divided into a kinematics component and a dynamics component, as stated above in (1) and (2), respectively, and it is shown in Fig. 2. Therefore, two cascade controllers are used, the first one is proposed on inverse kinematics, and the second one is based on neuro-adaptive SMC, for the kinematics and dynamics part of the robot system.

3. Kinematic Controller Design

The aim of this section is to find the kinematic control law based on the robot's discrete time kinematics part. Considering (2) and based on the inverse kinematics, the control law is expressed as follows:

$$\begin{pmatrix} x^k_{1ref}(k) \\ x^k_{2ref}(k) \end{pmatrix} = \begin{pmatrix} T_0^{-1}\cos\psi(k) & T_0^{-1}\sin\psi(k) \\ -a^{-1}T_0^{-1}\sin\psi(k) & a^{-1}T_0^{-1}\cos\psi(k) \end{pmatrix} \times \begin{pmatrix} r_{xref}(k+1) + b_x \tanh\left(\dfrac{k_x}{b_x}\tilde{r}_x(k)\right) \\ r_{yref}(k+1) + b_y \tanh\left(\dfrac{k_y}{b_y}\tilde{r}_y(k)\right) \end{pmatrix} \tag{3}$$

Equation (3) corresponds to kinematics control and the stability analysis of this controller and its considerations are formulated in [11].

4. Neuro-Adaptive Control Design

4.1 Problem Formulation

Due to uncertainties and parameters variations in the robot's dynamics, it is necessary to design a controller with adaptive properties. This controller must be implemented using neural techniques in combination with SMC theory (NA-SMC controller). The dynamics controller is fed with the linear and angular speeds provided by the

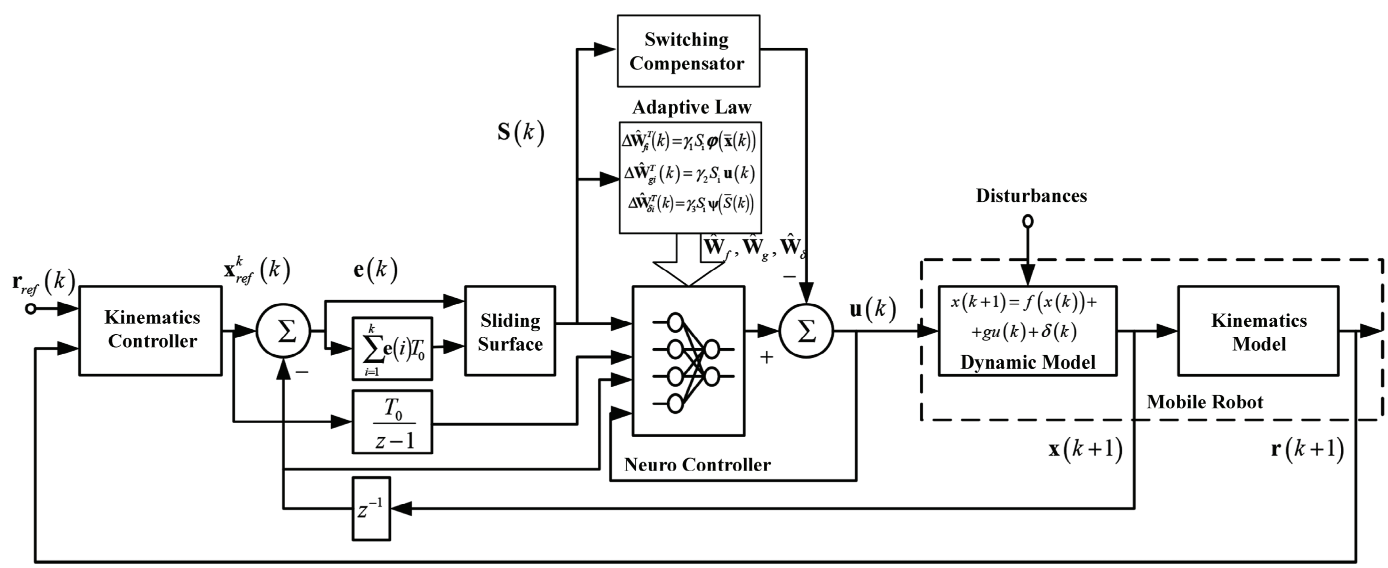


Figure 2. Control structure including the neuro-adaptive SMC and the kinematic controllers.

kinematics controller, noted by $\boldsymbol{x}_{ref} = (x^k_{1ref}, x^k_{2ref})^T$, where the superscript k indicates an output from the kinematics controller. The NA-SMC controller sends another set of linear and angular output speeds to be fed to the robot actuators $\boldsymbol{u} = (u_1, u_2)^T$, as shown in Fig. 2.

The WMR dynamics system (2) can be expressed in compact form as

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{f}(\mathbf{x}(k)) + \mathbf{g}\mathbf{u}(k) + \boldsymbol{\delta}(k) \\ &= \begin{pmatrix} f_1(\mathbf{x}(k)) \\ f_2(\mathbf{x}(k)) \end{pmatrix} + \begin{pmatrix} g_1 & 0 \\ 0 & g_2 \end{pmatrix} \mathbf{u}(k) + \begin{pmatrix} \delta_{x1}(k) \\ \delta_{x2}(k) \end{pmatrix}\end{aligned} \tag{4}$$

Noting that $\mathbf{g}$ is a diagonal matrix, in order for (4) to be controllable, diagonal components must be different to zero ($g_i \neq 0$). Vector $\boldsymbol{\delta}(k)$ represents the uncertainties (depends on pair $\delta_{x1}(k)$, $\delta_{x2}(k)$), with $\delta_{\max}$ being its upper bound as $\delta_{Max} = sup_{k\in\Re^+}\|\boldsymbol{\delta}(k)\|$,

The state variables, input vector and uncertainties parameters vector are defined as $\boldsymbol{x}(k) = (x_1(k), x_2(k))^T$, $\boldsymbol{u}(k) = (u_1(k), u_2(k))^T$ and $\boldsymbol{\delta}(k) = (\delta_{x1}(k), \delta_{x2}(k))^T$, respectively. The tracking control error is noted as follows:

$$\mathbf{e}(k) = \mathbf{x}(k) - \mathbf{x}_{ref}(k) = (x_1(k) - x^k_{1ref}(k), x_2(k) - x^k_{2ref}(k))^T \tag{5}$$

4.2 Sliding Surface Control

In this subsection, the aim is to design a neuro-adaptive SMC controller which guarantees the tracking of reference velocity $\boldsymbol{x}_{ref}(k)$ while keeping all variables bounded within the control loop. This controller must be implemented in a mobile robot using the neural feedback linearization technique, and it can minimize the velocity error vector [12], [13].

The desired velocity vector $\boldsymbol{x}_{ref}(k)$ provided by the kinematics control is introduced into the dynamic controller as a reference signal, and the designed control law should reduce the error between the velocity vector $\boldsymbol{x}$ and the desired velocity vector $\boldsymbol{x}_{ref}(k)$. Therefore, the tracking error should converge to zero.

To design a sliding surface $\mathbf{S}(k)$ for a mobile robotic system, the error state $\mathbf{e}(k)$ must be considered:

$$\mathbf{S}(k) = \begin{pmatrix} (\tau_d + \lambda_1 z^{-1}) & 0 \\ 0 & (\tau_d + \lambda_2 z^{-1}) \end{pmatrix}$$

$$\sum_{i=0}^{k} \mathbf{e}(i)T_0 = \begin{pmatrix} e_1(k) + \lambda_1 \sum_{i=0}^{k-1} e_1(i)T_0 \\ e_2(k) + \lambda_2 \sum_{i=0}^{k-1} e_2(i)T_0 \end{pmatrix} \tag{6}$$

where T_0 denotes the sampling time and τ_d represents $\tau_d = (1 - z^{-1})/T_0$. Now doing the discrete difference $\Delta\mathbf{S}(k)$:

$$\begin{aligned}\mathbf{S}(k+1) - \mathbf{S}(k) &= \begin{pmatrix} e_1(k+1) + \lambda_1 \sum_{i=0}^{k} e_1(i)T_0 \\ e_2(k+1) + \lambda_2 \sum_{i=0}^{k} e_2(i)T_0 \end{pmatrix} \\ &\quad - \begin{pmatrix} e_1(k) + \lambda_1 \sum_{i=0}^{k-1} e_1(i)T_0 \\ e_2(k) + \lambda_2 \sum_{i=0}^{k-1} e_2(i)T_0 \end{pmatrix} \\ &= \begin{pmatrix} e_1(k+1) + (\lambda_1 T_0 - 1)e_1(k) \\ e_2(k+1) + (\lambda_2 T_0 - 1)e_2(k) \end{pmatrix}\end{aligned} \tag{7}$$

where λ_i is a positive constant defined by the designer. Note that $\rho = \mathrm{diag}(\lambda_1 T_0 - 1, \lambda_2 T_0 - 1)$.

Now, it is necessary to define the equivalent control law $\mathbf{u^*}$, where $\Delta\mathbf{S}(k)$ is as defined previously. Next, set $\Delta\mathbf{S}(k)$ equal to zero, as follows:

$$\Delta\mathbf{S}(k) = \mathbf{S}(k+1) - \mathbf{S}(k)|_{u(k)=u^*(k)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{8}$$

Thus, replacing (7) into (8)

$$\begin{aligned}\Delta\mathbf{S}(k) &= \mathbf{e}(k+1) + \boldsymbol{\rho}\mathbf{e}(k) = (\mathbf{f}(\mathbf{x}(k)) + \mathbf{g}\mathbf{u}^*(k) \\ &\quad + \boldsymbol{\delta}(k) - \mathbf{x}_{ref}(k+1)) + \boldsymbol{\rho}\mathbf{e}(k) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}\end{aligned} \tag{9}$$

Now, taking into account the nonlinear system (4) as treated in [13], and applying a control law $\mathbf{u}^*$ that ensures the sliding condition of (9), which is formed by the proposed equivalent control

$$\mathbf{u}(k) = \mathbf{g}^{-1}[-\mathbf{f}(\mathbf{x}(k)) - \boldsymbol{\rho}\mathbf{e}(k) - \boldsymbol{\delta}(k) + \mathbf{x}_{ref}(k+1)] \tag{10}$$

And selecting a control strategy to ensure the control convergence to zero, as denoted below:

$$(\mathbf{S}(k+1) - \mathbf{S}(k)) = -\mathbf{A}\mathbf{S}(k)T_0 - \boldsymbol{\eta}T_0\mathrm{sign}(\mathbf{S}(k)) \tag{11}$$

where $\mathbf{A}$ is a diagonal matrix ($\mathbf{A}$=[α_1 0; 0 α_2]) and its elements (α_1 and α_2) are no bigger than some positive number ($|\alpha_i/ \leq 1$). From (11), $\boldsymbol{\eta}$ = [η_1 0; 0 η_2] matrix and the $sign(Si)$ function are denoted as $sign(S_i) = 1$ if $S_i > 0$, $sign(S_i) = 0$ if $S_i = 0$ and $sign(S_i) = -1$ if $S_i < 0$.

From (11) and taking into account (4) and (5)

$$\begin{aligned}\mathbf{S}(k+1) - \mathbf{S}(k) &= \mathbf{e}(k+1) + \boldsymbol{\rho}\mathbf{e}(k) = (\mathbf{f}(\mathbf{x}(k)) \\ &\quad + \mathbf{g}\mathbf{u}^*(k) + \boldsymbol{\delta}(k) - \mathbf{x}_{ref}(k+1)) + \boldsymbol{\rho}\mathbf{e}(k) \\ &= -\mathbf{A}\mathbf{S}(k)T_0 - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k))\end{aligned} \tag{12}$$

From (12), the equivalent control law $\mathbf{u}^*$ is obtained

$$\begin{aligned}\mathbf{u}^*(k) &= \mathbf{g}^{-1}[-\mathbf{f}(\mathbf{x}(k)) + \mathbf{x}_{ref}(k+1) - \boldsymbol{\rho}\mathbf{e}(k) - \mathbf{A}\mathbf{S}(k)T_0 \\ &\quad - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k)) - \boldsymbol{\delta}(k)]\end{aligned} \tag{13}$$

This equivalent control will determine the robotics system behaviour on the sliding surface.

4.3 Neural Adaptive System and Adjustment Laws

In real systems, $\boldsymbol{f}(\boldsymbol{x}(k))$, $\boldsymbol{g}$ and $\boldsymbol{\delta}(k)$ represent the dynamic part of the system and $\boldsymbol{\eta}$ is a control parameter defined by the designer. The dynamic part may be partially known or unknown and $sgn(\boldsymbol{S})$ is not a continuous function. Then, it is proposed to employ a neural system to approximate $\boldsymbol{f}(\boldsymbol{x}(k))$, $\boldsymbol{g}$ and $\boldsymbol{\delta}(k)$ by estimations $\hat{\boldsymbol{f}}(\boldsymbol{x}(k)|\mathbf{W}_f^*)$, $\hat{\boldsymbol{g}}(\mathbf{W}_g^*)$ and $\hat{\boldsymbol{\delta}}(k|\mathbf{W}_\delta^*)$, respectively, in (13).

The optimal weights vectors are defined as $\mathbf{W}_f^* = \text{argmin}_{\boldsymbol{W}_f \in \chi_W}\{sup_{x\in\chi W}|f(\boldsymbol{x}(k)) - \hat{f}(\boldsymbol{x}(k)|\mathbf{W}_f)\}$, $\mathbf{W}_g^* = \text{argmin}_{\boldsymbol{W}_g \in \chi_W}\{sup_{x\in\chi W}|g - \hat{g}(\mathbf{W}_g)\}$ and $\mathbf{W}_\delta^* = \text{argmin}_{\boldsymbol{W}_\delta \in \chi_W}\{sup_{x\in\chi W}|\delta(k) - \hat{\delta}(k|\mathbf{W}_\delta)\}$ where χ_W and χ_x are compact sets of suitable bounds on $\mathbf{W}_f$, $\mathbf{W}_g$, $\mathbf{W}_\delta$ and x, respectively, defined as $\chi_W = \{\mathbf{W}/|\mathbf{W}_f| \leq \varsigma_f \wedge |\mathbf{W}_g| \leq \varsigma_g \wedge |\mathbf{W}_\delta| \leq \varsigma_\delta\}$ and $\chi_x = \{x/|x| \leq \varsigma_x\}$ where ς_f, ς_g, ς_δ and ς_x are positive constants.

On the neuro-adaptive network, the hidden layer is implemented with exponential activation functions, and each function is called RBF (14), which is expressed as follows:

$$\begin{aligned}\varphi_{\mathrm{i}}(\bar{\mathbf{x}}(k)) &= \exp(-\|\bar{\mathbf{x}}(k) - \mathbf{c}_{\mathrm{i}}\|^2 \big/ \sigma_{\mathrm{i}}^2) \quad \text{and} \\ \psi_{\mathrm{i}}(\bar{\mathbf{S}}(k)) &= \exp(-\|\bar{\mathbf{S}}(k) - \mathbf{c}_{\mathrm{i}}\|^2 \big/ \sigma_{\mathrm{i}}^2) \end{aligned} \tag{14}$$

where i denotes the ith component of the hidden layer, $\boldsymbol{c}_i$ is the centroid position of the ith neural component and σ_i is its width factor. The neural inputs patterns are organized as $\bar{\mathbf{x}}(k) = [\mathbf{x}(k), \mathbf{x}(k-1), \mathbf{x}_{ref}(k+1), \mathbf{u}(k)]^T$ and $\bar{\mathbf{S}}(k) = [\mathbf{S}(k), \mathbf{S}(k-1), \mathbf{x}_{ref}(k+1), \mathbf{u}(k)]^T$

The control $\boldsymbol{u}$ can be composed by an NN-RBF through online learning

$$\begin{aligned}\mathbf{u}^* &= \hat{\mathbf{g}}^{-1}(\mathbf{W}_g^{T*})[-\hat{\mathbf{f}}(\mathbf{x}(k), \mathbf{W}_f^*) - \hat{\boldsymbol{\delta}}(\mathbf{S}(k), \mathbf{W}_\delta^*) \\ &\quad + \mathbf{x}_{ref}(k+1) - \boldsymbol{\rho}\mathbf{e}(k) - \mathbf{A}\mathbf{S}(k)T_0 - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k))]\end{aligned} \tag{15}$$

where $\mathbf{W}_f^* \in \Re^{m\times j}$ $\mathbf{W}_g^* \in \Re^{j\times j}$ and $\mathbf{W}_\delta^* \in \Re^{m\times j}$ are optimal weight vectors of $\mathbf{W}_f$, $\mathbf{W}_g$ and $\mathbf{W}_\delta$ respectively. The functions $\boldsymbol{\varphi} \in \Re^{m\times j}$ and $\boldsymbol{\psi} \in \Re^{m\times j}$ are the radial basis functions linked to the weights $\mathbf{W}_f$ and $\mathbf{W}_\delta$, respectively.

However, the optimal parameter vectors are unknown; therefore, it is necessary to estimate their values. Let us define an estimated control function:

$$\begin{aligned}\mathbf{u} &= \hat{\mathbf{g}}^{-1}(\hat{\mathbf{W}}_g^T)[-\hat{\mathbf{f}}(\mathbf{x}(k), \hat{\mathbf{W}}_f) - \hat{\boldsymbol{\delta}}(\mathbf{S}(k), \hat{\mathbf{W}}_\delta) + \mathbf{x}_{ref}(k+1) \\ &\quad -\boldsymbol{\rho}\mathbf{e}(k) - \mathbf{A}\mathbf{S}(k)T_0 - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k)) + \mathbf{u}_\Delta]\end{aligned} \tag{16}$$

A robust control action $\boldsymbol{u}_\Delta(k)$ was incorporated in order to reduce any possible non-considered dynamic variation. It is defined as $(\boldsymbol{\Delta} sign(\boldsymbol{S}))$. Variables $\mathbf{W}_f$, $\mathbf{W}_g$ and $\mathbf{W}_\delta$ are neural weights used by neuro-adaptive system $\hat{\boldsymbol{f}}(\boldsymbol{x}(k), \hat{\mathbf{W}}_f)$, $\hat{\boldsymbol{g}}(\hat{\mathbf{W}}_g)$ and $\hat{\boldsymbol{\delta}}(\boldsymbol{S}(k), \hat{\mathbf{W}}_\delta)$ respectively, and can be expressed as follows:

$$\hat{\mathbf{f}}(\mathbf{x}(k), \hat{\mathbf{W}}_f) = \hat{\mathbf{W}}_f^T \varphi(\bar{\mathbf{x}}(k)) = \begin{pmatrix} \hat{\mathbf{W}}_{f1}^T \varphi(\bar{\mathbf{x}}(k)) \\ \hat{\mathbf{W}}_{f2}^T \varphi(\bar{\mathbf{x}}(k)) \end{pmatrix} \tag{17}$$

And

$$\hat{\mathbf{g}}(\hat{\mathbf{W}}_g) = \hat{\mathbf{W}}_g^T \tag{18}$$

where $\hat{\mathbf{W}}_{f1,2}^T \in \Re^{1\times m}$ are row vectors with $m = 5$ and $\hat{\mathbf{W}}_g^T \in \Re^{j\times j}$ being $j = 2$.

Another neural network control term is used for attenuating external disturbances. The MIMO control term is shown as follows:

$$\hat{\boldsymbol{\delta}}(\mathbf{S}(k), \hat{\mathbf{W}}_\delta) = \hat{\mathbf{W}}_\delta^T \boldsymbol{\Psi}(\bar{\mathbf{S}}(k)) = \begin{pmatrix} \hat{\mathbf{W}}_{\delta 1}^T \boldsymbol{\Psi}(\bar{\mathbf{S}}(k)) \\ \hat{\mathbf{W}}_{\delta 2}^T \boldsymbol{\Psi}(\bar{\mathbf{S}}(k)) \end{pmatrix} \tag{19}$$

where $\hat{\mathbf{W}_T^{\delta 1}}$ and $\hat{\mathbf{W}_{\delta 2}^T}$ are adjustable weights, and $\psi(\bar{S}(k))$ is depending on the regressor vector.

This control technique is based on (16), and the minimum approximation error can be defined as follows:

$$\begin{aligned}\boldsymbol{\varepsilon}(k) &= \mathbf{f}(\mathbf{x}) - \hat{\mathbf{f}}(\mathbf{x}(k), \mathbf{W}_f^*) + [\mathbf{g} - \hat{\mathbf{g}}(\mathbf{W}_g^*)]\mathbf{u} \\ &\quad + \boldsymbol{\delta}(k) - \hat{\boldsymbol{\delta}}(\mathbf{S}(k), \mathbf{W}_\delta^*)\end{aligned} \tag{20}$$

Now, using (16) and considering the dynamic robot model (4), it can be rearranged as follows:

$$\begin{aligned}\Delta\mathbf{S}(k) &= \mathbf{S}(k+1) - \mathbf{S}(k) = [\mathbf{x}(k+1) - \mathbf{x}_{ref}(k+1)] \\ &\quad + \boldsymbol{\rho}\mathbf{e}(k) = \boldsymbol{\rho}\mathbf{e}(k) + [\mathbf{f}(\mathbf{x}(k)) + \mathbf{g}\mathbf{u}(k) + \boldsymbol{\delta}(k) \\ &\quad - \mathbf{x}_{ref}(k+1)]\end{aligned} \tag{21}$$

Substituting the proposed control action of (16) into (21):

$$\begin{aligned}\Delta\mathbf{S}(k) &= \boldsymbol{\rho}\mathbf{e}(k) + (\mathbf{f}(\mathbf{x}(k)) - \hat{\mathbf{f}}(\mathbf{x}(k), \hat{\mathbf{W}}_f)) + \mathbf{x}_{ref}(k+1) \\ &\quad + (\mathbf{g} - \hat{\mathbf{g}}(\hat{\mathbf{W}}_g))\mathbf{u}(k) - \boldsymbol{\rho}\mathbf{e}(k) + [\boldsymbol{\delta}(k) \\ &\quad - \hat{\boldsymbol{\delta}}(\mathbf{S}(k), \hat{\mathbf{W}}_\delta)] - \mathbf{x}_{ref}(k+1) - \mathbf{A}\mathbf{S}(k)T_0 \\ &\quad - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k)) + \mathbf{u}_\Delta \\ &= \Big[\hat{\mathbf{f}}(\mathbf{x}(k), \mathbf{W}_f^*) - \hat{\mathbf{f}}(\mathbf{x}(k), \hat{\mathbf{W}}_f)] + (\hat{\mathbf{g}}(\mathbf{W}_g^*) \\ &\quad - \hat{\mathbf{g}}(\hat{\mathbf{W}}_g))\mathbf{u}(k) + [\hat{\boldsymbol{\delta}}(\mathbf{S}(k), \mathbf{W}_\delta^*) - \hat{\boldsymbol{\delta}}(\mathbf{S}(k), \hat{\mathbf{W}}_\delta)\Big] \\ &\quad + \boldsymbol{\varepsilon} - \mathbf{A}\mathbf{S}(k)T_0 - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k)) + \mathbf{u}_\Delta\end{aligned} \tag{22}$$

Now, considering that

$$\begin{aligned}\hat{\mathbf{f}}(\mathbf{x}(k), \mathbf{W}_f^*) - \hat{\mathbf{f}}(\mathbf{x}(k), \hat{\mathbf{W}}_f) &= \mathbf{W}_f^{*T}\varphi(\bar{\mathbf{x}}(k)) - \hat{\mathbf{W}}_f^T\varphi(\bar{\mathbf{x}}(k)) \\ &= \tilde{\mathbf{W}}_f^T\varphi(\bar{\mathbf{x}}(k)) \end{aligned} \tag{23}$$

$$\hat{\mathbf{g}}(\mathbf{W}_g^*) - \hat{\mathbf{g}}(\hat{\mathbf{W}}_g) = \mathbf{W}_g^{*T} - \hat{\mathbf{W}}_g^T = \tilde{\mathbf{W}}_g^T \tag{24}$$

$$\begin{aligned}\hat{\boldsymbol{\delta}}(\mathbf{S}(k), \mathbf{W}_\delta^*) - \hat{\boldsymbol{\delta}}(\mathbf{S}(k), \hat{\mathbf{W}}_\delta) &= (\mathbf{W}_\delta^{*T} - \hat{\mathbf{W}}_\delta^T)\boldsymbol{\Psi}(\bar{\mathbf{S}}(k)) \\ &= \tilde{\mathbf{W}}_\delta^T\boldsymbol{\Psi}(\bar{\mathbf{S}}(k))\end{aligned} \tag{25}$$

where $\tilde{\mathbf{w}}_f$, $\tilde{\mathbf{w}}_g$ and $\tilde{\mathbf{w}}_\delta$ are defined as

$$\tilde{\mathbf{W}}_f^T = \mathbf{W}_f^{*T} - \hat{\mathbf{W}}_f^T; \tilde{\mathbf{W}}_g^T = \mathbf{W}_g^{*T} - \hat{\mathbf{W}}_g^T; \tilde{\mathbf{W}}_\delta^T = \mathbf{W}_\delta^{*T} - \hat{\mathbf{W}}_\delta^T \tag{26}$$

being $\mathbf{W}_f^*$, $\mathbf{W}_g^*$ and $\mathbf{W}_\delta^*$ the optimal constant weigh values.

From (22) and taking into account (23), (24), (25) and (26), the following expression is reached:

$$\begin{aligned}\Delta\mathbf{S}(k) &= -\mathbf{A}\mathbf{S}(k)T_0 + \tilde{\mathbf{W}}_g^T\mathbf{u}(k) + \tilde{\mathbf{W}}_f^T\varphi(\bar{\mathbf{x}}(k)) \\ &\quad +\tilde{\mathbf{W}}_\delta^T\mathbf{\Psi}(\bar{\mathbf{S}}(k)) + \boldsymbol{\varepsilon} - \boldsymbol{\eta}T_0 sign(\mathbf{S}(k)) + \mathbf{u}_\Delta \end{aligned} \quad (27)$$

Remark 1. *From (27), the minimum approximation error ε is bounded by $\|\boldsymbol{\varepsilon}\| \leq \varepsilon_{\max} = \|\boldsymbol{\eta}\||T_0|$.*

Remark 2. *The error approximation weight vectors $\tilde{\mathbf{W}}_{fi}$, $\tilde{\mathbf{W}}_{gi}$ and $\tilde{\mathbf{W}}_{\delta i}$ are bounded [14] by $\bar{\mathbf{W}}_f = sup_{k\in\Re^+} \|\tilde{\mathbf{W}}_{fi}(k)\|$, $\bar{\mathbf{W}}_g = sup_{k\in\Re^+}\|\tilde{\mathbf{W}}_{gi}(k)\|$, $\bar{\mathbf{W}}_\delta = sup_{k\in\Re^+} \|\tilde{\mathbf{W}}_{\delta i}(k)\|$.*

Theorem. *Considering the uncertain nonlinear system defined by (4). Then, the controller proposed in (16) ensures the convergence of tracking error to zero when applying the following weights adaptation laws:*

$$\Delta\tilde{\mathbf{W}}_{fi} = -\gamma_1 S_i\varphi(\bar{\mathbf{x}}(k)) \quad (28)$$

$$\Delta\tilde{\mathbf{W}}_{gi} = -\gamma_2 S_i\mathbf{u}(k) \quad (29)$$

$$\Delta\tilde{\mathbf{W}}_{\delta i} = -\gamma_3 S_i\mathbf{\Psi}(\bar{\mathbf{S}}(k)) \quad (30)$$

Proof: To prove the convergence of the proposed control technique is necessary to define a positive-definite function (Lyapunov's function candidate) L as

$$\begin{aligned}L &= \frac{1}{2}\sum_{i=1}^{2}[S_i^2(k) + \gamma_1^{-1}(\tilde{\mathbf{W}}_{fi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) \\ &\quad +\gamma_2^{-1}(\tilde{\mathbf{W}}_{gi}^T(k-1)\tilde{\mathbf{W}}_{gi}(k-1)) \\ &\quad +\gamma_3^{-1}(\tilde{\mathbf{W}}_{\delta i}^T(k-1)\tilde{\mathbf{W}}_{\delta i}(k-1))]\end{aligned} \quad (31)$$

Now, the discrete difference $\Delta L(k)$ may be represented as follows:

$$\begin{aligned}\Delta L &= \sum_{i=1}^{2}\{(S_i^2(k+1) - S_i^2(k)) + \dots \\ &\quad +\gamma_1^{-1}(\tilde{\mathbf{W}}_{fi}^T(k)\tilde{\mathbf{W}}_{fi}(k) - \tilde{\mathbf{W}}_{fi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) \\ &\quad +\gamma_2^{-1}(\tilde{\mathbf{W}}_{gi}^T(k)\tilde{\mathbf{W}}_{gi}(k) - \tilde{\mathbf{W}}_{gi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) \\ &\quad +\gamma_3^{-1}(\tilde{\mathbf{W}}_{\delta i}^T(k)\tilde{\mathbf{W}}_{\delta i}(k) - \tilde{\mathbf{W}}_{\delta i}^T(k-1)\tilde{\mathbf{W}}_{\delta i}(k-1))\}\end{aligned} \quad (32)$$

Now, the discrete difference ΔL can then be written as follows:

$$\begin{aligned}\Delta L &= \sum_{i=1}^{2}\{(S_i^2(k+1) - S_i^2(k)) \\ &\quad +\gamma_1^{-1}(\tilde{\mathbf{W}}_{fi}^T(k)\tilde{\mathbf{W}}_{fi}(k) - \tilde{\mathbf{W}}_{fi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) + \dots \\ &\quad +\gamma_2^{-1}(\tilde{\mathbf{W}}_{gi}^T(k)\tilde{\mathbf{W}}_{gi}(k) - \tilde{\mathbf{W}}_{gi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) \\ &\quad +\gamma_3^{-1}(\tilde{\mathbf{W}}_{\delta i}^T(k)\tilde{\mathbf{W}}_{\delta i}(k) - \tilde{\mathbf{W}}_{\delta i}^T(k-1)\tilde{\mathbf{W}}_{\delta i}(k-1))\}\end{aligned} \quad (33)$$

For the purpose of analysis, it is convenient to define the following variables $\Delta\boldsymbol{W}_{fi}$, $\Delta\boldsymbol{W}_{gi}$ and $\Delta\boldsymbol{W}_{\delta i}$ in vector form:

$$\begin{cases}\Delta\mathbf{W}_{fi} = \gamma_1^{-1}(\tilde{\mathbf{W}}_{fi}^T(k)\tilde{\mathbf{W}}_{fi}(k) - \tilde{\mathbf{W}}_{fi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) \\ \Delta\mathbf{W}_{gi} = \gamma_2^{-1}(\tilde{\mathbf{W}}_{gi}^T(k)\tilde{\mathbf{W}}_{gi}(k) - \tilde{\mathbf{W}}_{gi}^T(k-1)\tilde{\mathbf{W}}_{fi}(k-1)) \\ \Delta\mathbf{W}_{\delta i} = \gamma_3^{-1}(\tilde{\mathbf{W}}_{\delta i}^T(k)\tilde{\mathbf{W}}_{\delta i}(k) - \tilde{\mathbf{W}}_{\delta i}^T(k-1)\tilde{\mathbf{W}}_{\delta i}(k-1))\end{cases} \quad (34)$$

redefining the expression for (33) gives

$$\begin{aligned}\Delta L &= \sum_{i=1}^{2}\{(S_i^2(k+1) - S_i^2(k)) + \Delta\mathbf{W}_{fi} + \Delta\mathbf{W}_{gi} \\ &\quad +\Delta\mathbf{W}_{\delta i}\} = \sum_{i=1}^{2}\{((S_i(k) + \Delta S_i(k))^2 - S_i^2(k)) \\ &\quad +\Delta\mathbf{W}_{fi} + \Delta\mathbf{W}_{gi} + \Delta\mathbf{W}_{\delta i}\} = \dots \\ &= \sum_{i=1}^{2}\{(2S_i(k)\Delta S_i(k) + \Delta S_i^2(k)) + \Delta\mathbf{W}_{fi} \\ &\quad +\Delta\mathbf{W}_{gi} + \Delta\mathbf{W}_{\delta i}\}\end{aligned} \quad (35)$$

It follows using (27) in (35) gives

$$\begin{aligned}\Delta L &= \sum_{i=1}^{2}\{2(-\alpha_i T_0 S_i^2(k) + S_i(k)\tilde{\mathbf{W}}_{gi}^T\mathbf{u}(k) \\ &\quad +S_i\tilde{\mathbf{W}}_{fi}^T\varphi(\bar{\mathbf{x}}(k)) + S_i\tilde{\mathbf{W}}_{\delta i}^T\mathbf{\Psi}(\bar{\mathbf{S}}(k)) + \dots \\ &\quad +S_i\varepsilon_i - \boldsymbol{\eta}_i T_0 S_i sign(S_i(k))) + \Delta S_i^2(k) \\ &\quad +\Delta\mathbf{W}_{fi} + \Delta\mathbf{W}_{gi} + \Delta\mathbf{W}_{\delta i} + S_i\mathbf{u}_{\Delta i}\}\end{aligned} \quad (36)$$

From the expressions in (34), we get $\Delta\boldsymbol{W}_{fi}$ as

$$\begin{aligned}\Delta\mathbf{W}_{fi} &= \gamma_1^{-1}(\tilde{\mathbf{W}}_{fi}^T(k)\tilde{\mathbf{W}}_{fi}(k) - [\tilde{\mathbf{W}}_{fi}(k) - \Delta\tilde{\mathbf{W}}_{fi}(k)]^T \\ &\quad \times[\tilde{\mathbf{W}}_{fi}(k) - \Delta\tilde{\mathbf{W}}_{fi}(k)]) = \dots \\ &= 2\gamma_1^{-1}(\tilde{\mathbf{W}}_{fi}^T(k)\Delta\tilde{\mathbf{W}}_{fi}(k)) \\ &\quad -\gamma_1^{-1}(\Delta\tilde{\mathbf{W}}_{fi}^T(k)\Delta\tilde{\mathbf{W}}_{fi}(k))\end{aligned} \quad (37)$$

Similarly, $(\Delta\boldsymbol{W}_{gi} = 2\gamma_2^{-1}(\tilde{\mathbf{W}}_{gi}^{-1}(k)\Delta\tilde{\mathbf{W}}_{gi}(k)) - (\Delta\tilde{\mathbf{W}}_{gi}^T(k)\Delta\tilde{\mathbf{W}}_{gi}(k))$ and $(\Delta\boldsymbol{W}_{\delta i} = 2\gamma_2^{-1}(\tilde{\mathbf{W}}_{\delta i}^{-1}(k)\Delta\tilde{\mathbf{W}}_{\delta i}(k)) - (\Delta\tilde{\mathbf{W}}_{\delta i}^T(k)\Delta\tilde{\mathbf{W}}_{\delta i}(k))$.

Combining $\Delta\boldsymbol{W}_{fi}$, $\Delta\boldsymbol{W}_{gi}$ and $\Delta\boldsymbol{W}_{\delta i}$ in (36) we obtain that

$$\begin{aligned}\Delta L &= \sum_{i=1}^{2}\{-2\alpha_i S_i^2(k)T_0 + \Delta S_i^2(k) + 2S_i\varepsilon_i + 2\tilde{\mathbf{W}}_{fi}^T(k) \\ &\quad \times(S_i\varphi(\bar{\mathbf{x}}(k)) + 2\gamma_1^{-1}\Delta\tilde{\mathbf{W}}_{fi}(k)) - \dots \\ &\quad -2\gamma_1^{-1}(\Delta\tilde{\mathbf{W}}_{fi}^T(k)\Delta\tilde{\mathbf{W}}_{fi}(k)) + 2\tilde{\mathbf{W}}_{gi}^T(k)(S_i(k)\mathbf{u}(k) \\ &\quad +\gamma_2^{-1}\Delta\tilde{\mathbf{W}}_{gi}(k)) - \dots - 2\gamma_2^{-1}\left(\Delta\tilde{\mathbf{W}}_{gi}^T(k)\,\Delta\tilde{\mathbf{W}}_{gi}(k)\right) \\ &\quad +2\tilde{\mathbf{W}}_{\delta i}^T(k)\left(S_i\mathbf{\Psi}\left(\bar{\mathbf{S}}(k)\right) + \gamma_3^{-1}\Delta\tilde{\mathbf{W}}_{\delta i}(k)\right) - \dots \\ &\quad -2\gamma_3^{-1}(\Delta\tilde{\mathbf{W}}_{\delta i}^T(k)\Delta\tilde{\mathbf{W}}_{\delta i}(k)) - 2\boldsymbol{\eta}_i T_0 S_i sign(S_i(k)) \\ &\quad +S_i\mathbf{u}_{\Delta i}\}\end{aligned} \quad (38)$$

Replacing the learning rules (28), (29) and (30) in $\Delta\tilde{\mathbf{W}}_{fi}$, $\Delta\tilde{\mathbf{W}}_{gi}$ and $\Delta\tilde{\mathbf{W}}_{\delta i}$ of (38)

$$\begin{aligned}\Delta L = & \sum_{i=1}^{2}\{-2\alpha_i T_0 S_i^2(k) + \Delta S_i^2(k) + 2S_i\varepsilon_i - 2\gamma_1^{-1} \\ & \times(\Delta\tilde{\mathbf{W}}_{fi}^T(k)\Delta\tilde{\mathbf{W}}_{fi}(k)) - \ldots - 2\gamma_2^{-1}(\Delta\tilde{\mathbf{W}}_{gi}^T(k) \\ & \Delta\tilde{\mathbf{W}}_{gi}(k)) - 2\gamma_3^{-1}(\Delta\tilde{\mathbf{W}}_{\delta i}^T(k)\Delta\tilde{\mathbf{W}}_{\delta i}(k)) \\ & -2\boldsymbol{\eta}_i T_0|S_i(k)| + S_i u_{\Delta i}\}\end{aligned} \tag{39}$$

From (27) we get

$$\begin{aligned}|\Delta S_i(k)| \leq & |\alpha_i T_0 S_i(k)| + \left\|\tilde{\mathbf{W}}_{gi}^T\right\| \|\mathbf{u}\| + \|\boldsymbol{\eta}\| |T_0| \\ & + \left\|\tilde{\mathbf{W}}_{fi}^T\right\| \|\varphi(\bar{\mathbf{x}}(k))\| + \left\|\tilde{\mathbf{W}}_{\delta i}^T\right\| \|\boldsymbol{\Psi}(\bar{\mathbf{S}}(k))\| \\ & + |\varepsilon_i| + |u_{\Delta i}|\end{aligned} \tag{40}$$

Now, considering the boundedness of $\|\boldsymbol{\varphi}(\bar{\mathbf{x}}(k))\|$ and $\|\boldsymbol{\psi}(\bar{\boldsymbol{S}}(k))\| \leq 1$ and Remarks 1 and 2. From these considerations all terms in (40) are bounded, and recalling α_0 as $\alpha_0 = \|\eta\|T_0 + \|\tilde{\mathbf{W}}_{fi}^T\|\|\boldsymbol{\varphi}(\bar{x}) + \|\tilde{\mathbf{W}}_{gi}^T\|\|\mathbf{u}(k)\| + \|\tilde{\mathbf{W}}_{\delta i}^T\|\|\boldsymbol{\psi}(\bar{\boldsymbol{S}})\| + |\varepsilon_i|$, (40) can be rewritten as follows:

$$|\Delta S_i(k)| \leq |\alpha_i T_0 S_i(k)| + |\alpha_0| + |u_{\Delta i}| \tag{41}$$

Now, both sides of (41) are squaring,

$$\begin{aligned}|\Delta S_i(k)|^2 \leq & \left[|\alpha_i T_0 S_i(k)| + |\alpha_0| + |u_{\Delta i}|\right]^2 = |\alpha_i T_0 S_i(k)|^2 \\ & + 2|\alpha_i T_0 S_i(k)||\alpha_0| + 2|\alpha_i T_0 S_i(k)||u_{\Delta i}| \\ & + 2|\alpha_0||u_{\Delta i}| + |\alpha_0|^2 + |u_{\Delta i}|^2\end{aligned} \tag{42}$$

Now considering the term $2|u_\Delta||S_i(k)|$, adding and subtracting it in (42) and rearranging:

$$\begin{aligned}|\Delta S_i(k)|^2 \leq & \left[|\alpha_i T_0 S_i(k)| + |\alpha_0|\right]^2 + 2|\alpha_i T_0 S_i(k)||u_{\Delta i}| \\ & + 2|\alpha_0||u_{\Delta i}| + 2|S_i(k)||u_{\Delta i}| \\ & - 2|S_i(k)||u_{\Delta i}| + |u_{\Delta i}|^2\end{aligned} \tag{43}$$

Rewriting and rearranging (43)

$$|\Delta S_i(k)|^2 \leq \left(\vartheta_2(k) + \vartheta_1(k)|u_{\Delta i}| + |u_{\Delta i}|^2\right) - 2|u_{\Delta i}||S_i(k)| \tag{44}$$

where $\vartheta_2 = (|\alpha_i T_0 S_i(k)| + |\alpha_0|)^2$ and $\vartheta_1 = 2|\alpha_i T_0 S_i(k)| + 2|\alpha_0| + 2|S_i(k)|$, and (44) is bounded by

$$|\Delta S_i(k)|^2 \leq \left[-\vartheta_2(k) + \left(\vartheta_1^2(k) - 4\vartheta_2(k)\right)^{1/2}\right]|S_i(k)| \tag{45}$$

From (45), the term $u_{\Delta i}$ is chosen as more suitable as

$$\begin{aligned}2S_i(k)u_{\Delta i} = & 2S_i(k)\Delta_i sign(S_i(k)) \\ = & -\left[-\vartheta_2(k) + \left(\vartheta_1^2(k) - 4\vartheta_2(k)\right)^{1/2}\right]|S_i(k)|\end{aligned} \tag{46}$$

With the last considerations, it is easily demonstrated that:

$$\begin{aligned}\Delta L = & \sum_{i=1}^{2}\Big[-2\alpha_i T_0 S_i^2(k) - 2\gamma_1^{-1}\left(\Delta\tilde{\mathbf{W}}_{fi}^T(k)\Delta\tilde{\mathbf{W}}_{fi}(k)\right) \\ & - 2\gamma_2^{-1}\left(\Delta\tilde{\mathbf{W}}_{gi}^T(k)\Delta\tilde{\mathbf{W}}_{gi}(k)\right) - \ldots \\ & -2\gamma_3^{-1}\left(\Delta\tilde{\mathbf{W}}_{\delta i}^T(k)\Delta\tilde{\mathbf{W}}_{\delta i}(k)\right) + 2S_i\varepsilon_i - 2\boldsymbol{\eta}_i T_0|S_i(k)|\Big]\end{aligned} \tag{47}$$

From Remark 1, (47) can be represented by:

$$\begin{aligned}\Delta L = & \sum_{i=1}^{2}\Big[-2k_{di}T_0 S_i^2(k) - 2\gamma_1^{-1}\left(\Delta\tilde{\mathbf{W}}_{fi}^T(k)\Delta\tilde{\mathbf{W}}_{fi}(k)\right) \\ & - 2\gamma_2^{-1}\left(\Delta\tilde{\mathbf{W}}_{gi}^T(k)\Delta\tilde{\mathbf{W}}_{gi}(k)\right) \\ & -2\gamma_3^{-1}\left(\Delta\tilde{\mathbf{W}}_{\delta i}^T(k)\Delta\tilde{\mathbf{W}}_{\delta i}(k)\right)\Big] < 0\end{aligned} \tag{48}$$

This result demonstrates the convergence of the proposed control law.

The equations describing the adaptation laws are (28)–(30), but a practical implementation of the adaptation laws must be expressed in estimation terms. Then replacing (26) in (28)–(30) leads to:

$$\begin{aligned}\Delta\tilde{\mathbf{W}}_{fi} = & \tilde{\mathbf{W}}_{fi}(k+1) - \tilde{\mathbf{W}}_{fi}(k) = \left(\mathbf{W}_{fi}^{*T} - \hat{\mathbf{W}}_{fi}^T(k+1)\right) \\ & \left(-\hat{\mathbf{W}}_{fi}^{*T} - \hat{\mathbf{W}}_{fi}^T(k)\right) = -\Delta\hat{\mathbf{W}}_{fi}\end{aligned} \tag{49}$$

And the adaptation laws can be expressed as $\Delta\widehat{\mathbf{W}}_{fi}^T(k) = \gamma_1 S_i \boldsymbol{\varphi}(\bar{\mathbf{x}}(k))$, $\Delta\widehat{\mathbf{W}}_{gi}^T(k) = \gamma_2 S_i \boldsymbol{u}(\bar{\mathbf{x}}(k))$ and $\Delta\widehat{\mathbf{W}}_{\delta i}^T(k) = \gamma_3 S_i \boldsymbol{\psi}(\bar{\mathbf{S}}(k))$. These equations represent a practical adaptation law for the neural weights.

5. Experimental Results

For the experiment, a Pioneer 2DX WMR from Active Media Inc. is used (Fig. 1(b)). The Pioneer 2DX has an onboard computer 800 MHz Intel/Pentium III using 512 Mb of RAM onto which the controller was programmed. The WMR position is determined by odometric sensors (Fig. 1(b)).

For adjusting the neural net parameters, the so-called random-value parameters are initialized. At the same time, the robot must follow a pre-defined trajectory (for this case, a polygonal trajectory) where the proposed control

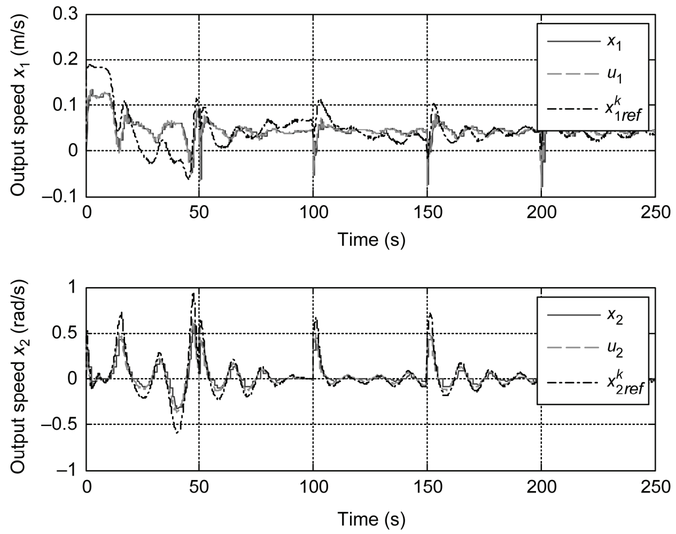


Figure 3. Mobile robot kinematics reference, output velocities and control actions using NA-SMC.

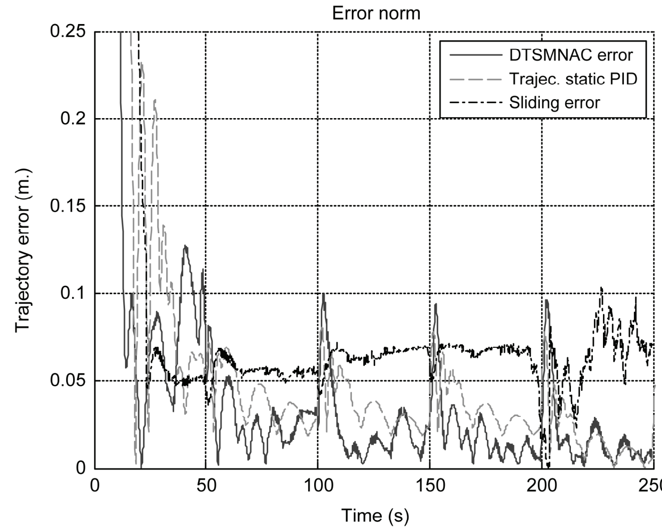


Figure 5. Norm of the trajectory error using an NA-SMC and a static PID control, and also an SMC.

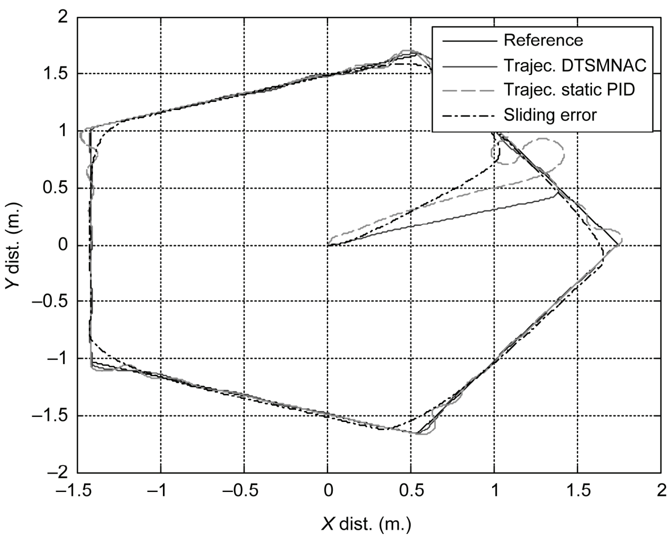


Figure 4. Reference and trajectory followed by the WMR using NA-SMC, PID and an SMC controller.

system adjusts its parameters according to the adjustment laws established by the Lyapunov criterion (equations $\Delta\tilde{\mathbf{W}}_f^T$, $\Delta\tilde{\mathbf{W}}_g^T$ and $\Delta\tilde{\mathbf{W}}_\delta^T$) which tend to stabilize and remain constant. In the experiment, the robot starts at $(r_{xref}, r_{yref}) = (0, 0)$ m. The robot must follow a pentagonal trajectory reference (see Fig. 4). The recorded experimental results are presented in Figs. 3–5 (compared with the classical PID controller and SMC controller). The classical PID controller was tuned using a linearized dynamic of the WMR. The response driven by the PID controller is characterized only by a larger value of the tracking error, because of the existing time-varying inertia and loads, and model uncertainties. As the discrete time NA-SMC can compensate for these phenomena by means of learning, the actual response showed a better approach to the desired response. It can be seen that, after switching to the proposed neurocontrol scheme, the kinematic velocity signals are changed by the discrete time NA-SMC, the tracking errors are significantly reduced and the error position closely follows the required trajectories, therefore demonstrating a very good tracking performance by the developed control scheme.

During the experiment, the NA-SMC controller was initialized with trained parameters from previous experiments. Figure 3 presents the references, control actions and output speeds of the NA-SMC controller. Figure 4 shows the trajectory performed by the WMR with both the discrete NA-SMC controller and the kinematic controller, compared with PID tuned at different gains for each output variable [15]. Another comparison was done with a single SMC controller, being the same discrete time NA-SMC where the neuro-adaptive part was turned off. In this experiment, the SMC gain increased, and the results show that the followed trajectory is far in each corner. If the neuro-adaptive is turned on, the error between the reference and the followed trajectory tends to zero. Figure 5 shows the trajectory error norm of the experiment using the proposed discrete time neuro-adaptive SMC controller to follow the desired reference trajectory as compared with the PID and the SMC controller. It can be easily verified that after 50 s, the position error for the discrete time NA-SMC control becomes smaller than that of the SMC controller. This fact can be explained by the learning effect of the neural network.

6. Conclusion

This control technique has been applied for trajectory tracking of WMR with dynamics variations and model uncertainties. The main contribution of this research work is that it does not require previous knowledge of the dynamic structure or its parameters to compute the proposed control law. The control strategy was totally designed in discrete time. Therefore, the stability problems caused by direct implementation in discrete time of a system designed in continuous domain are thus avoided, and the stability analysis was done in discrete time using Lyapunov's criterion and the global asymptotic stability of the control technique is ensured. When matching with the robot dynamics occurs, the overall control system becomes equivalent to a stable dynamic system.

References

[1] F.G. Rossomando, C. Soria, and R. Carelli, Sliding mode neuro adaptive control in trajectory tracking for mobile robots, *Journal of Intelligent & Robotic Systems*, *74*(3–4), 2014, 931–944.

[2] Y. Yamamoto, Identification and control for nonlinear discrete time systems using a new neural network, *Intelligent Systems and Control*, Acta Press, 2007, Proceeding 592, vol. 360592, p. 024.

[3] J. Wang, Z., Qu, M.S. Obeng, and X. Wu, Approximation based adaptive tracking control of uncertain nonholonomic mechanical systems, *Control and Intelligent Systems*, *37*(4), 2009, 204.

[4] J. Ye, Tracking control of a nonholonomic wheeled mobile robot using improved compound cosine function neural networks, *International Journal of Control*, *88*(2), 2015, 364–373.

[5] F.G. Rossomando, C. Soria, and R. Carelli, Autonomous mobile robot navigation using RBF neural compensator, *Control Engineering Practice*, *19*(3), 2010, 215–222.

[6] M. Lopez-Franco, E.N. Sanchez, A.Y. Alanis, C. Lopez-Franco, and N. Arana-Daniel, Discrete-time decentralized inverse optimal neural control combined with sliding mode for mobile robots, *IEEE World Automation Congress (WAC)*, 2014, 496–501, 2014, doi: 10.1109/WAC.2014.6936014.

[7] Y.-J. Liu, L. Tang, S. Tong, and C.L.P. Chen, Adaptive NN controller design for a class of nonlinear MIMO discrete-time systems, *IEEE Transactions on Neural Networks and Learning Systems*, *26*(5), 2015, 1007–1018.

[8] Z. Wang, R. Lu, F. Gao, and D. Liu, An indirect data-driven method for trajectory tracking control of a class of nonlinear discrete time systems, *IEEE Transactions on Industrial Electronics*, *64*(5), 2017, 4121–4129.

[9] F.G. Rossomando and C.M. Soria, Discrete-time sliding mode neuro-adaptive controller for SCARA robot arm, *Neural Computing and Applications*, 2017, *28*(12), 3837–3850.

[10] F.G. Rossomando and C.M. Soria, Adaptive neural sliding mode control in discrete time for a SCARA robot arm, *IEEE Latin America Transactions*, *14*(6), 2016, 2556–2564.

[11] F.G. Rossomando and C.M. Soria, Identification and control of nonlinear dynamics of a mobile robot in discrete time using an adaptive technique based on neural PID, *Neural Computing and Applications*, *26*(5), 2015, 1179–1191.

[12] J.J.E. Slotine and W. Li, *Applied nonlinear control* (Upper Saddle River, NJ: Prentice Hall, 1991).

[13] V. Utkin and J. Shi, Integral sliding mode in systems operating under uncertainty conditions, *Proc. 35th IEEE Conf. on Decision and Control*, Vol. 4, IEEE, 1996, 4591–4596.

[14] Y.J. Liu, S. Li, S. Tong, and C.L.P. Chen, Neural approximation-based adaptive control for a class of nonlinear nonstrict feedback discrete-time systems, *IEEE Transactions on Neural Networks and Learning Systems*, *99*, 1–11 doi: 10.1109/TNNLS.2016.2531089.

[15] K.J. Åström and T. Hägglund, PID Controllers: theory, design, and tuning, *Instrument Society of America*, Research Triangle Park, NC, 2nd edition, 1995.

Biographies

Francisco G. Rossomando from 2002 to 2006, worked on his doctorate degree at the Universidad Federal of Espirito Santo (ES-Brazil), with a thesis on the modelling and control of hot rolling mills. He is currently a researcher at the National Council for Scientific and Technical Research of Argentina (CONICET), at the Universidad Nacional de San Juan (UNSJ).

Carlos Soria graduated in Electrical Engineering from the National University of Tucuman, Argentina, in 1996. On March 2005, he obtained the Ph.D. degree in Control Systems Engineering at the Instituto de Automática, National University of San Juan (UNSJ).

Eduardo O. Freire received the Ph.D. degree in electrical engineering from the Universidade Federal do Espírito Santo, Vitória, Brazil. He is currently an Associate Professor in the Department of Electrical Engineering, Universidade Federal de Sergipe, Brazil. His research is related to robotics, control and artificial intelligence.

Ricardo O. Carelli was born in San Juan, Argentina, in 1952. He graduated in electrical engineering from the National University of San Juan, Argentina, and received the Ph.D. degree in electrical engineering from the National University of Mexico (UNAM). He is currently a full Professor at the National University of San Juan and Senior Researcher (Investigador Principal) of the National Council for Scientific and Technical Research (CONICET, Argentina). He is a Director of the Instituto de Automática, National University of San Juan. He also coordinates the Ph.D. and Master Programs in Control Engineering at the same university.