Adaptive Neural Network Identification for Robust Multivariable Systems

Felipe Osorio-Arteaga, Eduardo Giraldo

Abstract—**This paper proposes a robust identification and control-based on a neural network method for a Twin Rotor Multivariable System (TRMS) using a recursive adaptive training algorithm. The algorithm is based on a recursive least squares approach with an additional steepest descent stage. An Adaline neural network is used for modeling the system, and a robust structure is selected based on a linear auto-regressive structure with exogenous inputs (ARX) related to the estimation error. The identification is performed online and the system is controlled under a polynomial structure by pole placement with a dead-beat strategy. The method is evaluated in terms of estimation and tracking error in the presence of external additive disturbances, parametric disturbances, and sinusoidal reference signals. The Root-Mean Square Error (RMSE) is used to evaluate the estimation performance and the Integral-Time Absolute Error (ITAE) is used to evaluate the tracking performance. As a result, a novel robust controller based on a neural network is designed where the best results are obtained for a training recursive least squares algorithm with an additional steepest descent stage.**

Index Terms—**Neural Network, Adaline, robust identification, multivariable system, Twin Rotor.**

I. Introduction

Enhancing the control of real multivariable physical systems holds significant importance in the field of engineering, due to the nonlinearity in their region of operation, uncertainties in the system parameters, actuator constraints, response delay times, and the presence of external disturbances. To tackle these complexities, several authors have studied different adaptive control approaches that estimate the controller gains from the input and output measurements and solve some of these difficulties [1]. In [2], an adaptive predictive control method based on the ARX-Laguerre mathematical model for multivariable systems is implemented that guarantees a simple recursive representation, however, it is quite sensitive to the signal-to-noise ratio. In [3], an adaptive control multivariable strategy is proposed based on a polynomial structure where the exogenous inputs are considered in order to obtain the robustness of the identified model.

In [4], an adaptive observer for a class of multivariable nonlinear systems is presented considering unknown parameters of the state and output equations but obtaining a continuous-time model. In [5], an adaptive fuzzy dynamic fuzzy surface control for multivariable nonlinear systems is designed, although it improves the system performance, work must be done on the selection of the design parameters. [6] presents a sliding plane control for nonlinear systems with the use of smooth functions for decreasing *chattering* and [7] implements an adaptive neural controller with fewer learning parameters designed for a class of nonlinear systems but both only applied to scalar systems. In [8], a model-based polynomial control is designed for a nonlinear buck converter, however, the training of the model is developed offline. In [9], an adaptive strategy for a multivariable microgrid system is proposed by considering a deterministic approach in state space, and in [10], a linear state-space identification is applied for control of a Twin Rotor MIMO System (TRMS) by state feedback.

Neural networks have been widely used in machine learning as a complement to the control of complex systems in different research fields such as stabilization or control of nonlinear systems [11]. In [12] an adaptive control with backward step based on neural networks using a neural network to guarantee controller tracking is presented, in [13] the use of neural networks with reinforcement learning and Liapunov functions for the control of nonlinear systems is proposed, in [14] a discrete adaptive dynamic decoupled controller is designed for a multivariable nonlinear system using neural networks for the convergence to zero of the tracking error, and in [1] a constrained adaptive controller with backward step capable of guaranteeing asymptotic stability is shown, however, they do not take into account external disturbances.

In this work, a robust identification and control method based on a neural network structure is proposed. The robustness of the model is achieved by considering exogenous inputs related to the estimation error. The identification is performed online by adding a steepest descent algorithm to the recursive least squares algorithm. The proposed controller is designed based on the weights of the robust neural network and is also computed online by considering a dead-beat strategy for pole placement. The proposed approach is evaluated over a TRMS for simulation, where the root mean square error is used to evaluate the estimation error, and the integral time absolute error is used to evaluate the tracking error. The system is evaluated under sinusoidal reference signals in terms of estimation and tracking error, and by considering external disturbances and parametric disturbances. The paper is organized as follows: in section II are presented the theoretical framework for the TRMS, the neural network identification stage, and the control design. In section III is presented the evaluation of the proposed approach over the TRMS simulation in the presence of external and parametric disturbances and

Manuscript received June 22, 2023; revised November 10, 2023.

Felipe Osorio-Arteaga is an adjunct Professor at the Department of Electrical Technology, Universidad Tecnológica de Pereira, Pereira, Colombia. Research Group in Automatic Control. E-mail: pipeoso28@utp.edu.co.

Eduardo Giraldo is a Full Professor at the Department of Electrical Engineering, Universidad Tecnológica de Pereira, Pereira, Colombia. Research group in Automatic Control. E-mail: egiraldos@utp.edu.co.

sinusoidal reference signals. And finally, in section IV are presented the conclusions and final remarks.

II. Theoretical Framework

A. Adaline Neural Network

It consists of a neuron with a linear activation function. Since its development in 1960, it has been used in a wide range of both linear and nonlinear applications. Since it has a linear activation function, it can be represented as a neuron with no activation function, as shown in Figure 1.

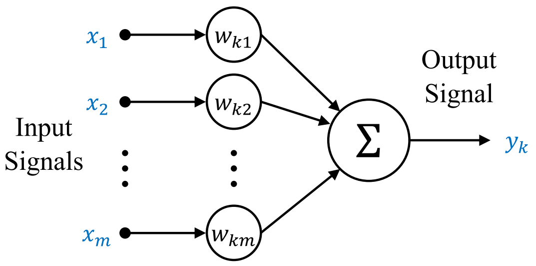

Fig. 1. Lineal Neural Network (Adaline).

The mathematical model of an Adaline neuron, without bias:

$$y = \sum_{j=1}^{m} w_j x_j = \mathbf{x}^T \mathbf{w} \tag{1}$$

Consider a single layer Adaline neural network with one layer:

$$\mathbf{y} = \mathbf{x}^T \mathbf{W} \tag{2}$$

B. Steepest Descent Algorithm

It is a classical optimization approach widely used for minimizing or maximizing functions. Applied to the learning of an Adaline network, the update of the weights is in the opposite direction of the gradient of the cost function E and is given by the equation:

$$\mathbf{W}_{k+1} = \mathbf{W}_k - \eta \, \nabla E_{(\mathbf{W})}\big|_{\mathbf{W}_k} \tag{3}$$

where η is known as the learning rate, which must be positive and must have a sufficiently small value for the convergence to the optimal solution will be smooth and non-oscillatory.

C. Least Squares Algorithm

The least squares algorithm is given by an error function as:

$$E_{(\mathbf{W})} = \frac{1}{2}\sum_j e_j^2 \tag{4}$$

$$= \frac{1}{2}\sum_j (d_j - \mathbf{x}^T \mathbf{w}_j)^2 \tag{5}$$

$$= \frac{1}{2}(\mathbf{d} - \mathbf{x}^T\mathbf{W})(\mathbf{d} - \mathbf{x}^T\mathbf{W})^T \tag{6}$$

The derivative of the error function with respect to the weights is obtained as:

$$\frac{\partial E_{(\mathbf{W})}}{\partial \mathbf{W}} = -\mathbf{x}\mathbf{d} + \mathbf{x}\mathbf{x}^T\mathbf{W} = -\mathbf{x}(\mathbf{d} - \mathbf{x}^T\mathbf{W}) = 0 \tag{7}$$

Therefore, the optimal value for the weight vector of the neural network is obtained as follows:

$$\mathbf{W}^* = (\mathbf{x}\mathbf{x}^T)^{-1}\mathbf{x}\mathbf{d} \tag{8}$$

where $\frac{\partial E_{(\mathbf{W})}}{\partial \mathbf{W}}$ is the instantaneous gradient estimate. As a result, the weights update equation from (3) is given by:

$$\mathbf{W}_{k+1} = \mathbf{W}_k - \eta \left.\frac{\partial E_{(\mathbf{W})}}{\partial \mathbf{W}}\right|_{\mathbf{W}_k} \tag{9}$$

$$= \mathbf{W}_k + \eta \mathbf{x}(\mathbf{d} - \mathbf{x}^T\mathbf{W}_k) \tag{10}$$

The aforementioned method is also known as the stochastic gradient. It is worth noting that in the least squares method, the weight of the net follows a stochastic trajectory, in contrast with the steepest descent method where a definite trajectory is followed. This behavior makes the least squares method more appropriate for some applications [15].

The least squares algorithm is also known as the delta learning algorithm, which when applied to nonlinear activation functions turns into the *Backpropagation* learning algorithm [16].

D. Recursive least squares algorithm

In order to obtain an online training algorithm for the neural network, a recursive version of the least squares algorithm is presented in [17]. This is performed by using the solution obtained in (8), as follows:

$$\mathbf{W}_k = \left(\mathbf{X}_k\mathbf{X}_k^T\right)^{-1}\mathbf{X}_k\mathbf{D}_k = \mathbf{P}_k\mathbf{F}_k \tag{11}$$

The matrices $\mathbf{P}_k$ and $\mathbf{F}_k$ are defined as:

$$\mathbf{P}_k = \left(\mathbf{X}_k\mathbf{X}_k^T\right)^{-1} \qquad \mathbf{F}_k = \mathbf{X}_k\mathbf{D}_k \tag{12}$$

By considering that $\mathbf{X}_k$ is $\mathbf{X}_{k-1}$ and $\mathbf{x}_k$, as follows:

$$\mathbf{X}_k\mathbf{X}_k^T = \mathbf{X}_{k-1}\mathbf{X}_{k-1}^T + \mathbf{x}_k\mathbf{x}_k^T \tag{13}$$

Equation (13) can be rewritten as:

$$\mathbf{P}_k^{-1} = \mathbf{P}_{k-1}^{-1} + \mathbf{x}_k\mathbf{x}_k^T \tag{14}$$

And by using the matrix-inversion-lemma (*Woodbury* equivalence), the following result is obtained:

$$\begin{aligned}(\mathbf{A}+\mathbf{BCD})^{-1} &= \mathbf{A}^{-1} - \\ &\mathbf{A}^{-1}\mathbf{B}\left(\mathbf{C}^{-1}+\mathbf{D}\mathbf{A}^{-1}\mathbf{B}\right)^{-1}\mathbf{D}\mathbf{A}^{-1}\end{aligned} \tag{15}$$

where

$$\mathbf{A} = \mathbf{P}_{k+1}^{-1}$$
$$\mathbf{B} = \mathbf{x}_k$$
$$\mathbf{C} = 1$$
$$\mathbf{D} = \mathbf{x}_k^T$$

From (14), is given that:

$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{P}_{k-1}\mathbf{x}_k\left(1 + \mathbf{x}_k^T\mathbf{P}_{k-1}\mathbf{x}_k\right)^{-1}\mathbf{x}_k^T\mathbf{P}_{k-1} \tag{16}$$

If $\mathbf{K}_k$ is defined as:

$$\mathbf{K}_k = \mathbf{P}_{k-1}\mathbf{x}_k\left(1 + \mathbf{x}_k^T\mathbf{P}_{k-1}\mathbf{x}_k\right)^{-1} \tag{17}$$

From (16), $\mathbf{P}_k$ can be obtained as follows:

$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{K}_k \mathbf{x}_k^T \mathbf{P}_{k-1} \tag{18}$$

Similarly for $\mathbf{D}_k$, the following equivalence can be defined:

$$\mathbf{X}_k \mathbf{D}_k = \mathbf{X}_{k-1} \mathbf{D}_{k-1} + \mathbf{x}_k \mathbf{d}_k \tag{19}$$

Where (19) can be rewritten as:

$$\mathbf{F}_k = \mathbf{F}_{k-1} + \mathbf{x}_k \mathbf{d}_k \tag{20}$$

And by clearing $\mathbf{F}_k$ from (11), the following equation is obtained:

$$\mathbf{F}_{k-1} = \mathbf{P}_{k-1}^{-1} \mathbf{W}_{k-1} \tag{21}$$

where $\mathbf{e}_k$ is defined as:

$$\mathbf{e}_k = \mathbf{d}_k - \mathbf{x}_k^T \mathbf{W}_{k-1} \tag{22}$$

In addition, by clearing $\mathbf{d}_k$ from (22) and replacing (20) and (21) in (11):

$$\mathbf{W}_k = \mathbf{P}_k \left(\mathbf{P}_{k-1}^{-1} \mathbf{W}_{k-1} + \mathbf{x}_k \left(\mathbf{x}_k^T \mathbf{W}_{k-1} + \mathbf{e}_k \right) \right) \tag{23}$$

By clearing $\mathbf{P}_{k-1}^{-1}$ from (14) and replacing in (23), the following equation is obtained for weights update:

$$\mathbf{W}_k = \mathbf{W}_{k-1} + \mathbf{P}_k \mathbf{x}_k \mathbf{e}_k \tag{24}$$

By considering $\mathbf{K}_k$ from (17), the following equation is obtained

$$\mathbf{K}_k = \mathbf{P}_{k-1} \mathbf{x}_k - \mathbf{K}_k \mathbf{x}_k^T \mathbf{P}_{k-1} \mathbf{x}_k \tag{25}$$

$$\mathbf{K}_k = \left(\mathbf{P}_{k-1} - \mathbf{K}_k \mathbf{x}_k^T \mathbf{P}_{k-1} \right) \mathbf{x}_k \qquad \mathbf{K}_k = \mathbf{P}_k \mathbf{x}_k \tag{26}$$

Therefore, equation (24) can be rewritten as:

$$\mathbf{W}_k = \mathbf{W}_{k-1} + \mathbf{K}_k \mathbf{e}_k \tag{27}$$

E. Robust neural network based polynomial control

Consider a neural network Adaline, which represents the dynamics of a physical system, linear or nonlinear, including some possible noise dynamics, some nonlinear characteristics, or parameter variation. From the equation (2), it is defined:

$$\mathbf{x} = [\mathbf{y}_{k-1}, \ldots, \mathbf{y}_{k-n}, \mathbf{u}_{k-1}, \ldots, \mathbf{u}_{k-n}, \mathbf{e}_{k-1}, \ldots, \mathbf{e}_{k-m}]$$

$$\mathbf{W} = [\mathbf{W}_{\mathbf{a_1}}, \ldots, \mathbf{W}_{\mathbf{a_n}}, \mathbf{W}_{\mathbf{b_1}}, \ldots, \mathbf{W}_{\mathbf{b_n}}, \mathbf{W}_{\mathbf{c_1}}, \ldots, \mathbf{W}_{\mathbf{c_m}}]$$

Where m is the number of samples of the estimation error $\mathbf{e}$ and n is the number of samples for the input $\mathbf{u}$ and output $\mathbf{y}$. From the knowledge of the neural network weights $\mathbf{W}$, it is possible to design a polynomial controller with direct-loop tracking gain that allows tracking constant reference signals. Defining the control algorithm as an Adaline neural network:

$$\mathbf{u} = \mathbf{x_f}^T \mathbf{W_g} \tag{28}$$

$$\mathbf{x_f} = [\mathbf{u}_{k-1}, \mathbf{u}_{k-2}, \ldots, \mathbf{u}_{k-n}, \mathbf{e}_{k-1}, \mathbf{e}_{k-2}, \ldots, \mathbf{e}_{k-n}]$$

$$\mathbf{W_g} = [\mathbf{W}_{\mathbf{L_1}}, \mathbf{W}_{\mathbf{L_2}}, \ldots, \mathbf{W}_{\mathbf{L_n}}, \mathbf{W}_{\mathbf{P_1}}, \mathbf{W}_{\mathbf{P_2}}, \ldots, \mathbf{W}_{\mathbf{P_n}}]$$

Where

$$\mathbf{e}_k = \mathbf{r}_k - \mathbf{y}_k \qquad \mathbf{r}_k = \mathbf{k}_r \mathbf{y}_k^* \tag{29}$$

The neural network weights of the controller $\mathbf{W_g}$ are calculated in an intermediate step with the update of the neural network weights $\mathbf{W}$. Defining:

$$\mathbf{W_a} = [\mathbf{W}_{\mathbf{a_1}}, \ldots, \mathbf{W}_{\mathbf{a_n}}] \qquad \mathbf{W_b} = [\mathbf{W}_{\mathbf{b_1}}, \ldots, \mathbf{W}_{\mathbf{b_n}}]$$

$$\mathbf{W_L} = [\mathbf{W}_{\mathbf{L_1}}, \ldots, \mathbf{W}_{\mathbf{L_n}}] \qquad \mathbf{W_P} = [\mathbf{W}_{\mathbf{P_1}}, \ldots, \mathbf{W}_{\mathbf{P_n}}]$$

$$\mathbf{W_a^*} = \left[\mathbf{W}_{\mathbf{a_1}}^*, \ldots, \mathbf{W}_{\mathbf{a_{2n}}}^* \right]$$

The following polynomial equation arises:

$$\mathbf{W}_{\mathbf{a}(z^{-1})} \mathbf{W}_{\mathbf{L}(z^{-1})} + \mathbf{W}_{\mathbf{b}(z^{-1})} \mathbf{W}_{\mathbf{P}(z^{-1})} = \mathbf{W}_{\mathbf{a}(z^{-1})}^* \tag{30}$$

Which can be organized in a matrix form:

$$\mathbf{M} \begin{bmatrix} \mathbf{W}_{\mathbf{L_1}} \\ \mathbf{W}_{\mathbf{L_2}} \\ \cdot \\ \mathbf{W}_{\mathbf{L_n}} \\ \mathbf{W}_{\mathbf{P_1}} \\ \mathbf{W}_{\mathbf{P_2}} \\ \cdot \\ \mathbf{W}_{\mathbf{P_n}} \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{\mathbf{a_1}}^* \\ \mathbf{W}_{\mathbf{a_2}}^* \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{W}_{\mathbf{a_{2n}}}^* \end{bmatrix} \tag{31}$$

$$\mathbf{M} = \begin{bmatrix} \mathbf{I} & & & & & \\ \mathbf{W}_{\mathbf{a_1}} & & & \mathbf{W}_{\mathbf{b_1}} & & \\ \mathbf{W}_{\mathbf{a_2}} & & & \mathbf{W}_{\mathbf{b_2}} & & \\ \cdot & \cdot & & \cdot & & \\ \cdot & \cdot\ \cdot & & \cdot & \cdot & \\ \cdot & \cdot\ \cdot & \mathbf{I} & \cdot & \cdot\ \cdot & \\ \mathbf{W}_{\mathbf{a_n}} & \cdot\ \cdot & \mathbf{W}_{\mathbf{a_1}} & \mathbf{W}_{\mathbf{b_n}} & \cdot\ \cdot & \mathbf{W}_{\mathbf{b_1}} \\ & \cdot\ \cdot & \mathbf{W}_{\mathbf{a_2}} & & \cdot\ \cdot & \mathbf{W}_{\mathbf{b_2}} \\ & \cdot\ \cdot & \cdot & & \cdot\ \cdot & \cdot \\ & \cdot\ \cdot & \cdot & & \cdot\ \cdot & \cdot \\ & \cdot & \cdot & & \cdot & \cdot \\ & & \mathbf{W}_{\mathbf{a_n}} & & & \mathbf{W}_{\mathbf{b_n}} \end{bmatrix} \tag{32}$$

Where $\mathbf{W_a^*}$ can be rewritten as follows:

$$\begin{bmatrix} \mathbf{W}_{\mathbf{a_1}}^* \\ \mathbf{W}_{\mathbf{a_2}}^* \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{W}_{\mathbf{a_{2n}}}^* \end{bmatrix} = \begin{bmatrix} \alpha_1 - \mathbf{W}_{\mathbf{a_1}} \\ \alpha_2 - \mathbf{W}_{\mathbf{a_2}} \\ \cdot \\ \alpha_{n_1} - \mathbf{W}_{\mathbf{a_n}} \\ \alpha_{n_1+1} \\ \alpha_{n_1+2} \\ \cdot \\ \alpha_{n_1+n_2} \end{bmatrix} \tag{33}$$

where α_i are the coefficients of the polynomial which contains the poles of the closed-loop system, giving as solutions the values of $\mathbf{W_P}$ and $\mathbf{W_L}$ [18]. The value of the tracking gain $\mathbf{k}_r$ is calculated from the closed-loop transfer function formed by $\mathbf{W_a}$, $\mathbf{W_b}$, $\mathbf{W_P}$ and $\mathbf{W_L}$ as shown:

$$\mathbf{k}_r = \mathbf{I} + \frac{\left(\mathbf{I} + \mathbf{W}_{\mathbf{a_1}} \cdots + \mathbf{W}_{\mathbf{a_n}}\right)\left(\mathbf{I} + \mathbf{W}_{\mathbf{P_1}} + \cdots + \mathbf{W}_{\mathbf{P_n}}\right)}{\left(\mathbf{W}_{\mathbf{b_1}} + \cdots + \mathbf{W}_{\mathbf{b_n}}\right)\left(\mathbf{W}_{\mathbf{L_1}} + \cdots + \mathbf{W}_{\mathbf{L_n}}\right)} \tag{34}$$

A block diagram representation of the proposed robust identification and control can be visualized in Figure 2.

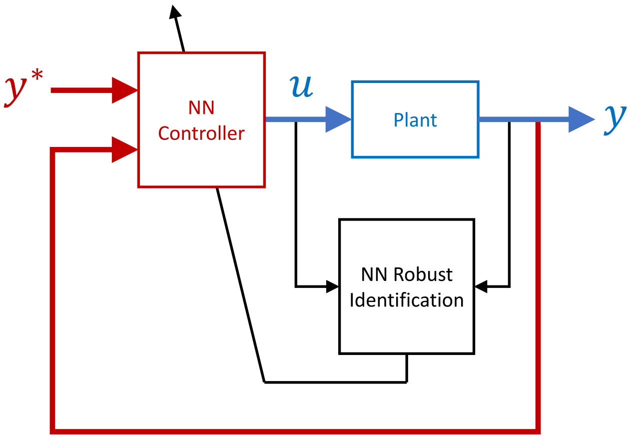

Fig. 2. Robust identification and control based neural networks.

III. RESULTS

In order to evaluate the performance of the proposed approach, an evaluation over a nonlinear multivariable model is considered. In this case, a TRMS is used for validation. Three stages are considered, model identification, identification and control based on robust neural network, and identification and control based on robust neural network under parametric disturbances.

A. TRMS Mathematical model

The TRMS has two inputs, the armature winding voltage of the DC motors that move the propellers, and two outputs, the elevation (*pitch*) and direction (*yaw*) angles. From Figure 3, the moment equations for vertical and horizontal motion are:

$$\mathrm{I}_1\ddot{\psi} = \mathrm{M}_1 - \mathrm{M}_{\mathrm{FG}} - \mathrm{M}_{\mathrm{B}\psi} - \mathrm{M}_{\mathrm{G}} \tag{35}$$

$$\mathrm{I}_2\ddot{\varphi} = \mathrm{M}_2 - \mathrm{M}_{\mathrm{B}\phi} - \mathrm{M}_{\mathrm{R}} \tag{36}$$

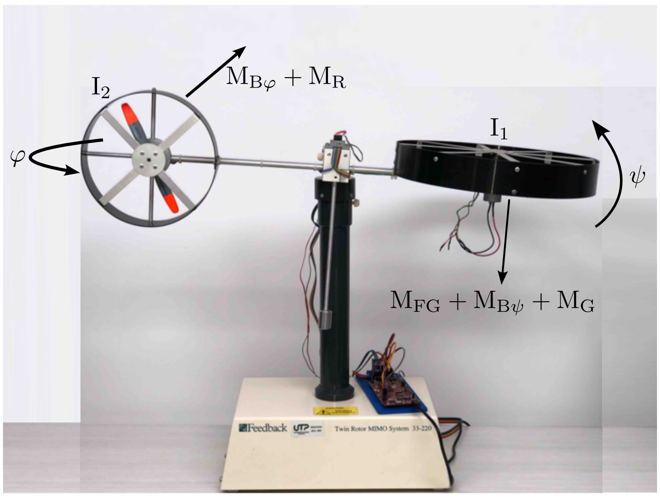

Fig. 3. TRMS variables.

Where M_1 y M_2 are caused by DC motors:

$$\mathrm{M}_1 = \mathrm{a}_1\tau_1^2 - \mathrm{b}_1\tau_1 \qquad \mathrm{M}_2 = \mathrm{a}_2\tau_2^2 - \mathrm{b}_2\tau_2 \tag{37}$$

M_{FG} is the gravitational momentum:

$$\mathrm{M}_{\mathrm{FG}} = \mathrm{M}_{\mathrm{g}} \sin\psi \tag{38}$$

$\mathrm{M}_{\mathrm{B}\psi}$ y $\mathrm{M}_{\mathrm{B}\varphi}$ are due to the frictional forces for each of the outputs:

$$\mathrm{M}_{\mathrm{B}\psi} = \mathrm{B}_{1\psi}\dot{\psi} + \mathrm{B}_{2\psi}\mathrm{sign}\left(\dot{\psi}\right) \tag{39}$$

$$\mathrm{M}_{\mathrm{B}\varphi} = \mathrm{B}_{1\varphi}\dot{\varphi} + \mathrm{B}_{2\varphi}\mathrm{sign}\left(\dot{\varphi}\right) \tag{40}$$

M_{G} is the gyroscopic momentum:

$$\mathrm{M}_{\mathrm{G}} = \mathrm{K}_{\mathrm{gy}}\mathrm{M}_1\dot{\varphi}\cos\psi \tag{41}$$

And M_{R} is the cross-reacting moment:

$$\mathrm{M}_{\mathrm{R}} = \frac{\mathrm{k}_{\mathrm{c}}\left(\mathrm{T}_{\mathrm{o}}\mathrm{s}+1\right)}{\left(\mathrm{T}_{\mathrm{p}}\mathrm{s}+1\right)}\tau_1 \tag{42}$$

Where τ_1 and τ_2 are the torques of the DC motors, which relate the above equations to the inputs:

$$\tau_1 = \frac{\mathrm{k}_1}{\mathrm{T}_{11}\mathrm{s}+\mathrm{T}_{10}}\mathrm{u}_1 \qquad \tau_2 = \frac{\mathrm{k}_2}{\mathrm{T}_{21}\mathrm{s}+\mathrm{T}_{20}}\mathrm{u}_2 \tag{43}$$

The experimental values of the parameters of the above equations are in [19].

B. Model identification based on robust neural networks

To obtain a model for the TRMS, a high order Adaline neural network is considered. As a result, the nonlinear dynamics are modeled by using a higher-order linear model with time-varying parameters. Thus, the neural network is designed based on a 8-th order model. Figure 4 shows the real outputs of the system, as well as the estimated outputs by using the neural network without robustness. The recursive least squares algorithm is used to perform an online training of the weights of the neural network.

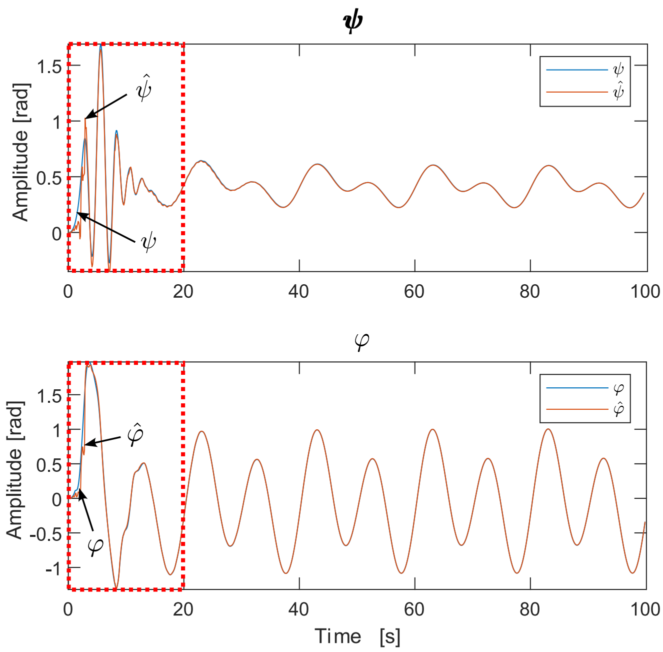

Fig. 4. Real and estimated outputs of the TRMS by using a neural network without robustness.

A detailed view of the first 20 seconds of Fig. 4 is shown in Fig. 5.

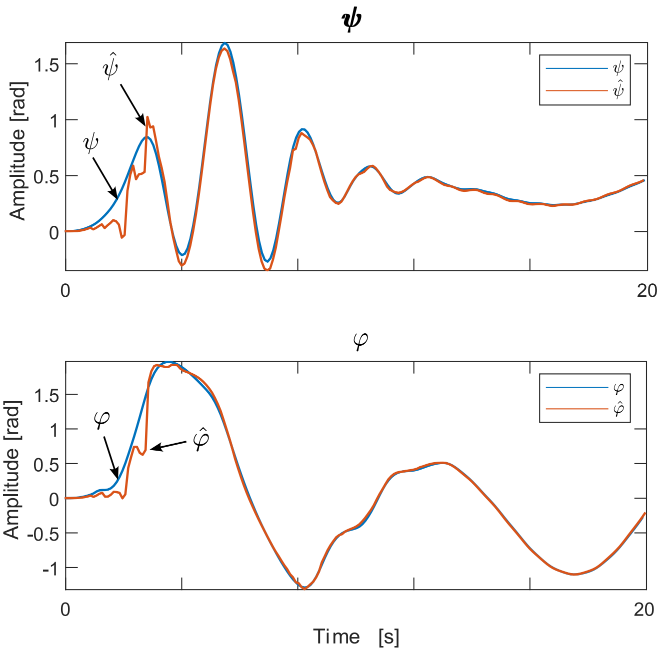

Fig. 5. Detailed view of the first 20 seconds of real and estimated outputs of the TRMS from Fig. 4.

To generate a robust model of the system, an exogenous input by considering a second-order model is included in the estimation. To this end, the estimation error defined by $\mathbf{e_{est}} = \mathbf{y} - \hat{\mathbf{y}}$ is considered as the exogenous input, resulting in a robust neural network. In Fig. 6 are shown the real and estimated outputs of the TRMS by considering an additive stochastic disturbance with zero mean and 10% amplitude (in terms of the output signal). The recursive least squares algorithm is also used for online training of the robust neural network weights.

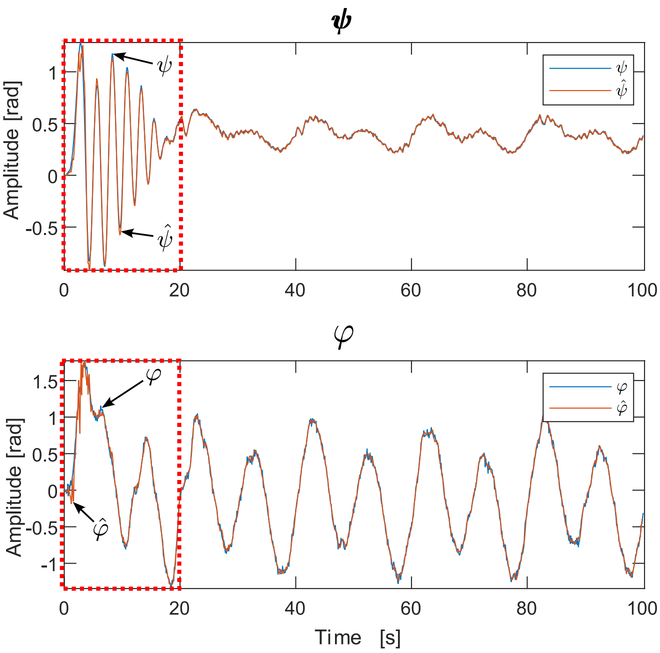

Fig. 6. Real and estimated outputs of the TRMS by considering a robust neural network with the recursive least squares algorithm.

A detailed view of the first 20 seconds of Fig. 6 is shown in Fig. 7.

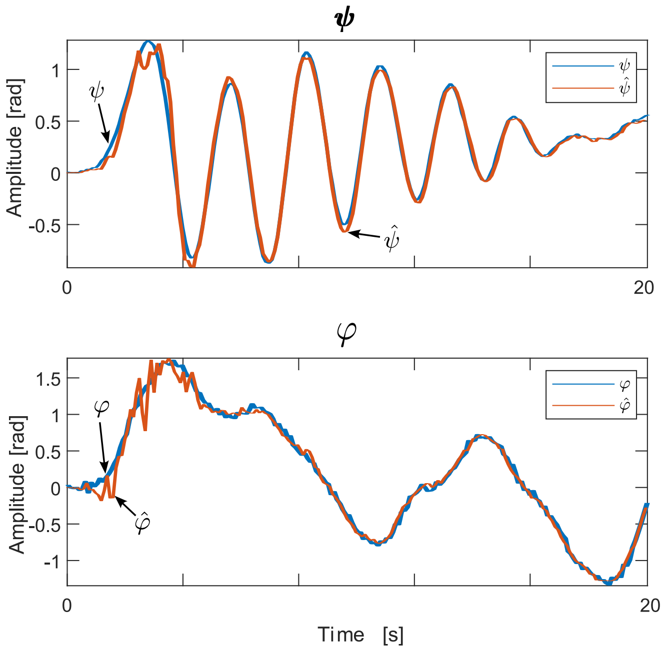

Fig. 7. Detailed view of the first 20 seconds of real and estimated outputs of the TRMS by considering a robust neural network with the recursive least squares algorithm from Fig. 6.

Fig. 8 shows the time evolution of the weights of the robust neural network related to the exogenous inputs model.

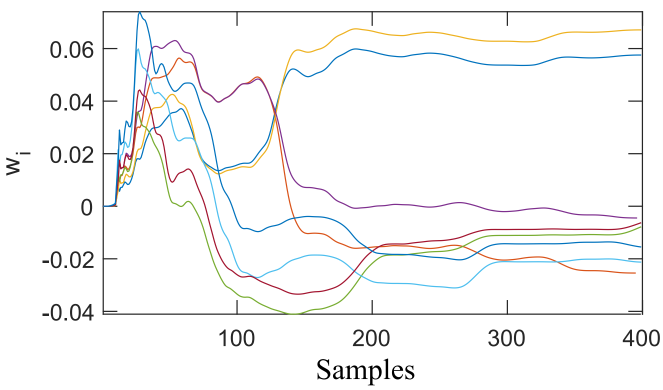

Fig. 8. Robust neural network weights related to the exogenous input e.g. estimation error.

An additional training method for the robust neural network is considered to improve the performance of the recursive least squares algorithm. This is achieved by using the steepest descent method. It is worth noting that this method is added as a second stage of the recursive least squares algorithm. The real and estimated outputs of the TRMS by considering an additive stochastic disturbance with zero mean and 10% amplitude (in terms of the output signal) are shown in Fig. 9 by using the recursive least squares method with steepest descent stage. A detailed view of the first 20 seconds of Fig. 9 is shown in Fig. 10.

To perform a quantitative comparison of the algorithms, the root mean squared error (RMSE) is used, as presented in (44). In Table I are presented the obtained results for each

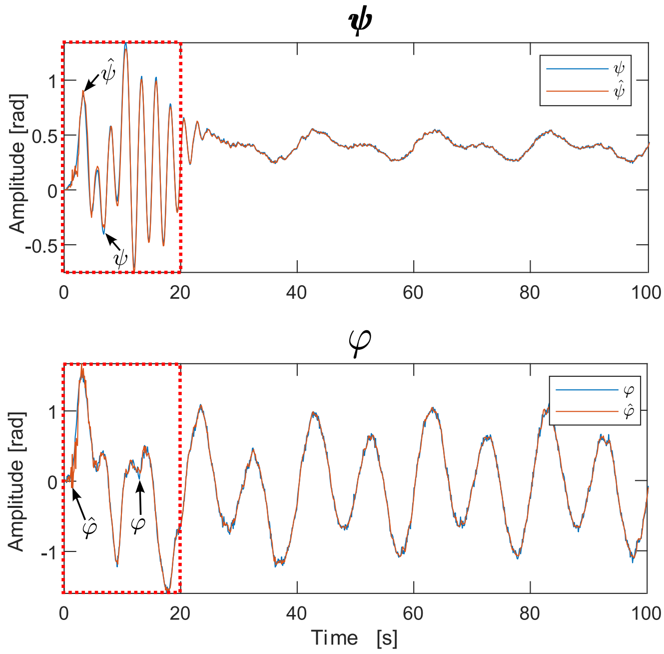

Fig. 9. Real and estimated output of the TRMS by considering a robust neural network trained by using recursive least squares with steepest descent.

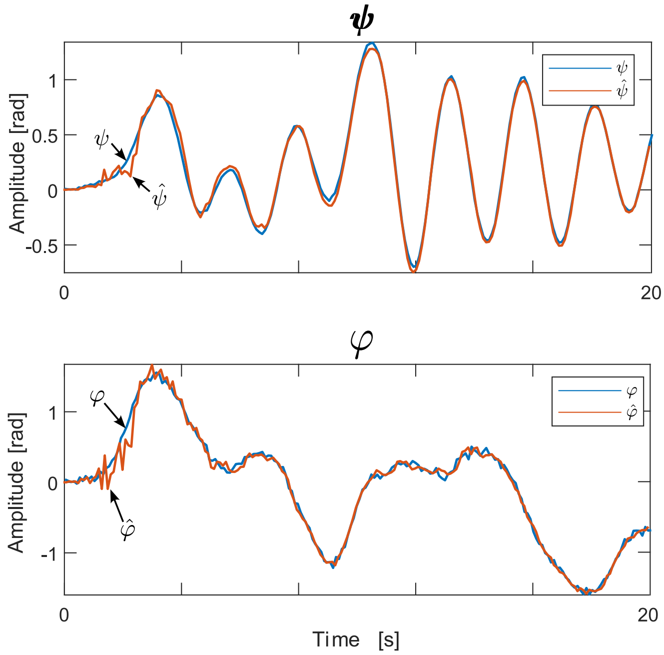

Fig. 10. Detailed view of the first 20 seconds of real and estimated output of the TRMS by considering a robust neural network trained by using recursive least squares with steepest descent from Fig. 9.

output signal by using each identification method and model, for a time window of 100 s.

$$RMSE = \sqrt{\frac{\sum_{j=1}^{n}(x_j - \hat{x_j})^2}{n}} \tag{44}$$

TABLE I
RMSE COMPARISON OF THE IDENTIFICATION ALGORITHMS.

RMSE	Pitch ψ	Yaw φ
Robust recursive least squares	1.0577	1.9058
Robust recursive least squares + steepest descent	0.7197	1.8732

From Table I it can be observed the similarities among the estimation methods. It can be observed that for the robust neural network model the best result is obtained by using the recursive least squares algorithm with steepest descent for both angles: pitch and yaw.

C. Identification and control based on robust neural networks

Once the TRMS is adequately identified by the robust neural network, a polynomial controller is defined in the same order as the robust neural network. Since the TRMS is identified online, each time an instant is obtained a robust model, and therefore the controller is also calculated. In this case, the closed-loop design is performed by considering a dead-beat strategy (poles at the origin of the complex plane) [18] and control signals saturated in the range ± 2.5, according to the allowed range of the manufacturer [19].

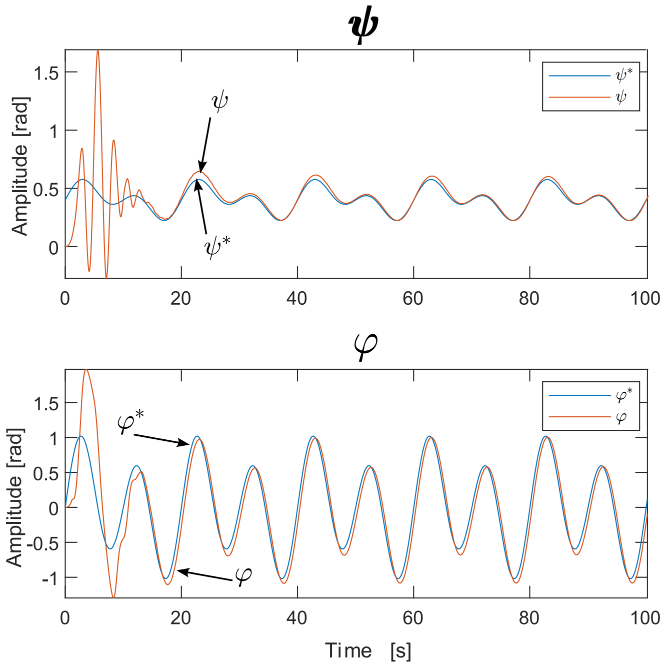

Fig. 11. Tracking response of the identification and control of the TRMS without robustness (references: ϕ^* and φ^*, outputs: ϕ and φ).

In Fig. 11 it can be seen the tracking response (references and outputs) of the identification and control based on neural networks without exogenous inputs related to the robustness of the model. In addition, in Fig. 12 it is shown the tracking response (references and outputs) of the identification and control of the robust neural networks. It is worth noting that the weights of the neural networks are adequately adjusted during the first 20 s of the test. This is performed by considering an additive disturbance of zero means and 10% noise level (in terms of the amplitude of the reference signal). It is worth mentioning that the references are time-varying defined as a sum of sinusoidal signals.

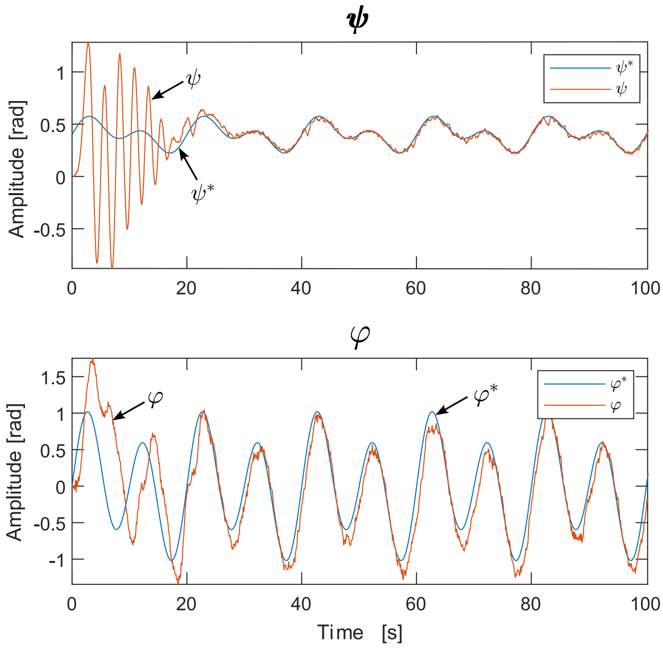

Fig. 12. Tracking response of the identification and control of the TRMS with a robust neural network (references: ϕ^* and φ^*, outputs: ϕ and φ).

Figure 13 shows the tracking response of the identification and control by using the robust neural network with the recursive least squares method and the steepest descent stage. Figure 14 shows a PID tracking response with an additive disturbance in each output *Feedback*.

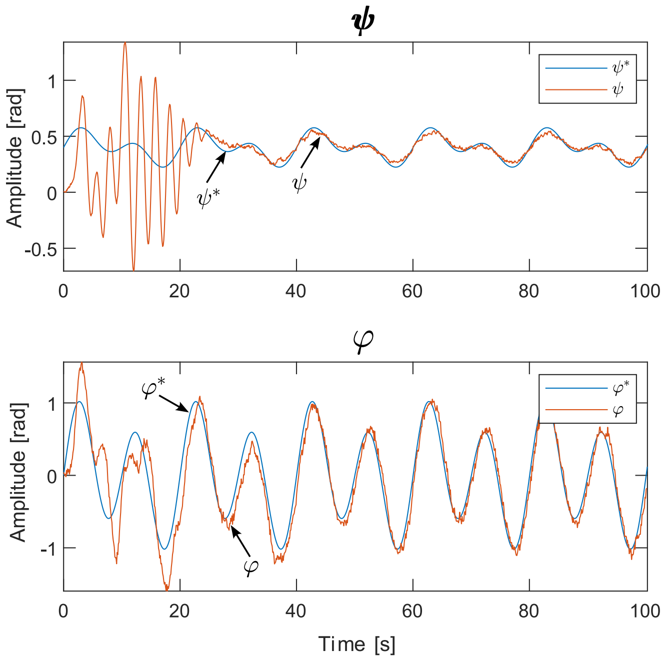

Fig. 13. Tracking response identification and control of the TRMS with a robust neural network by using recursive least squares with steepest descent stage (references: ϕ^* and φ^*, outputs: ϕ and φ).

A comparison analysis is performed in terms of the integral time absolute error criterion (ITAE) as described in (45). Table II shows the obtained results for each signal output by considering the reference tracking with each method of identification and control. This analysis is performed by using a time window of 100 s.

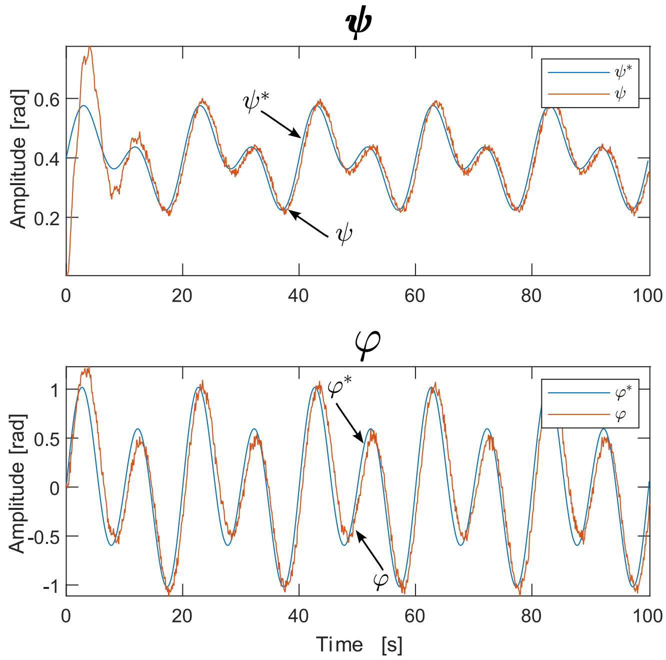

Fig. 14. Tracking response of the TRMS by using a PID controller (references: ϕ^* and φ^*, outputs: ϕ and φ).

$$ITAE = \sum_{j=1}^{n} kh \left| \frac{e_j + e_{j-1}}{2} \right| \tag{45}$$

TABLE II
ITAE COMPARISON OF THE IDENTIFICATION AND CONTROL METHODS USING THE ROBUST NEURAL NETWORK STRUCTURE

ITAE	Pitch ψ	Yaw φ
Robust recursive least squares	1361.7	8496.1
Robust recursive least squares + steepest descent	2185.3	6251
PID	1214.6	8691.5

According to the results shown in Table II, for the yaw angle, it can be seen that all the methods show a higher ITAE value in comparison to the yaw angle. This behavior is consistent with the results shown in Fig. 11, Fig. 12, Fig. 13 and Fig. **??** where a persistent tracking error is shown. In terms of the yaw angle, the robust recursive least squares algorithm with the steepest descent shows the lower ITAE value.

D. Identification and control based on robust neural networks with parametric disturbances

The final analysis to evaluate the performance of the proposed identification and control based on a robust neural network is the evaluation under parametric disturbances.

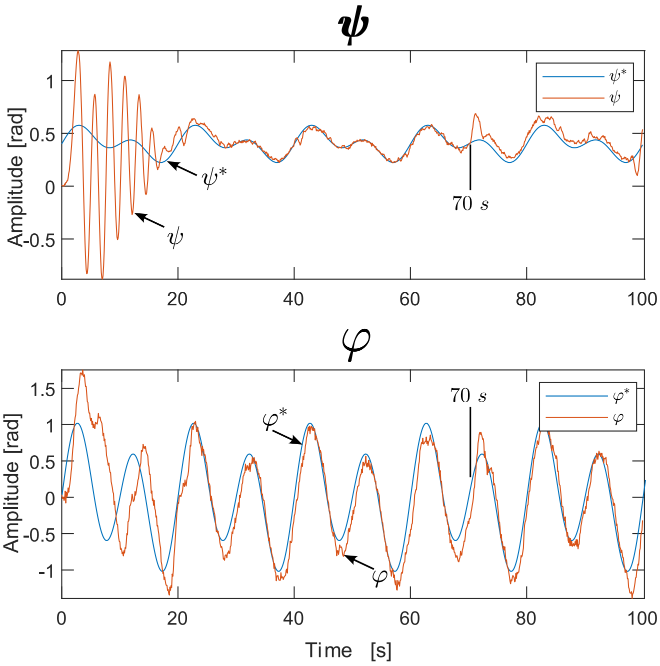

Fig. 15. Tracking response identification and control of the TRMS with a robust neural network with a parametric disturbance of $\mathrm{M_g}$ from 0.32 to 0.16 (references: ϕ^* and φ^*, outputs: ϕ and φ).

In Fig. 15 it is shown the tracking response of the identification and control based on robust neural networks, where the parameter $\mathrm{M_g}$ changes 50% of its value. According to (35), this parameter is directly related to the ψ angle. In Fig. 15 it is shown that the parametric disturbance is applied at time instant 70 s. It can be seen that the tracking error is increased during 30 s. Since the weights of the robust neural network are trained online, the model of the TRMS is adequately updated, and therefore the tracking error is successfully reduced.

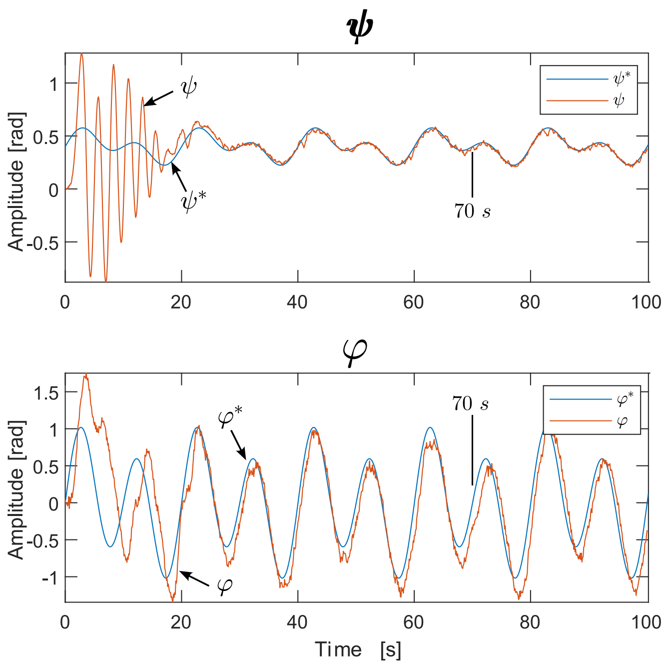

Fig. 16. Tracking response identification and control of the TRMS with a robust neural network with a parametric disturbance of $\mathrm{B}_{1\psi}$ from 0.1 to 0.2 (references: ϕ^* and φ^*, outputs: ϕ and φ).

In Fig. 16 it is shown the tracking response of the identification and control based on robust neural networks, where the parameter $\mathrm{B}_{1\psi}$ changes 100% of its value. According to (36), this parameter is directly related to the φ angle. In Fig. 16 it is shown that the parametric disturbance is applied at time instant 70 s. It can be seen that the tracking error is increased during 30 s but only for the φ angle. Since the weights of the robust neural network are trained online, the model of the TRMS is adequately updated, and therefore the tracking error is successfully reduced.

IV. Conclusions

In this work, an Adaline neural network structure for the identification of a nonlinear MIMO system is proposed. A robust for the neural network is proposed by considering as exogenous inputs the estimation error. Three methods for online identification of the system are analyzed to obtain the neural network training algorithm: the recursive least squares algorithm, and the recursive least squares algorithm with the steepest descent stage. It can be seen that by considering the RMSE performance criterion, the lower estimation error is achieved by the recursive least squares algorithm with the steepest descent stage. This behavior is validated by considering additive noise disturbances. It is worth noting that the robust neural network is trained at each time sample, therefore obtaining a time-varying adapted model.

In addition, a polynomial multivariable control is also applied based on the robust neural network, where the parameters of the controller are computed online at each time sample by using a dead-beat pole placement strategy. The performance of the proposed method is evaluated by considering the tracking performance of the method under the ITAE criterion. This behavior is validated under additive disturbances where the recursive least squares algorithm with the steepest descent stage achieves the lower tracking error.

The proposed approach is also evaluated by considering parametric disturbances. Two disturbances are considered which are directly related to the pitch and yaw angles. It can be seen that the proposed approach effectively updates the weights of the robust neural network since the training is performed online. And, once the model is identified adequately, the tracking error is effectively diminished.

In future works, a multi-layer robust neural network will be proposed where the closed-loop control must be designed according to the neural network structure. This work will be developed by considering the online training structure which effectively tracks any parameter variation of the model to be controlled.

References

[1] M. Chen, S. S. Ge, and B. Ren, “Adaptive tracking control of uncertain mimo nonlinear systems with input constraints,” *Automatica*, vol. 47, pp. 452–465, 3 2011.

[2] I. B. Abdelwahed, A. Mbarek, and K. Bouzrara, “Adaptive mpc based on mimo arx-laguerre model,” *ISA Transactions*, vol. 67, pp. 330–347, 3 2017.

[3] F. Osorio-Arteaga, J. J. Marulanda-Durango, and E. Giraldo, “Multivariable adaptive control of time-varying systems,” *IAENG International Journal of Computer Science*, vol. 47, no. 4, pp. 605–612, 2020.
[4] L. Zhao, X. Li, and P. Li, “Adaptive observer design for a class of mimo nonlinear systems,” in *10th World Congress on Intelligent Control and Automation*. Institute of Electrical and Electronics Engineers Inc., 2012, pp. 2198–2203.
[5] T. Yoshimura, “Adaptive fuzzy dynamic surface control for a class of stochastic mimo discrete-time nonlinear pure-feedback systems with full state constraints,” *International Journal of Systems Science*, vol. 49, pp. 3037–3047, 11 2018.
[6] J. E. Slotine, “Sliding controller desing for non-linear systems,” *International Journal of Control*, vol. 40, pp. 421–434, 1984.
[7] Y. J. Liu, J. Li, S. Tong, and C. L. Chen, “Neural network control-based adaptive learning design for nonlinear systems with full-state constraints,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, pp. 1562–1571, 7 2016.
[8] E. Giraldo, “Adaptive model-based control design of a synchronous buck converter,” *Engineering Letters*, vol. 30, no. 4, pp. 1364–1371, 2022.
[9] J. S. Velez-Ramirez, M. Bueno-Lopez, and E. Giraldo, “Adaptive control approach of microgrids,” *IAENG International Journal of Applied Mathematics*, vol. 50, no. 4, pp. 802–810, 2020.
[10] Y. Martinez-Armero, J. B. Renteria-Mena, and E. Giraldo, “Robust embedded control applied to a twin rotor mimo system,” *Engineering Letters*, vol. 31, no. 1, pp. 230–238, 2023.
[11] Y. Yang, T. Wang, J. P. Woolard, and W. Xiang, “Guaranteed approximation error estimation of neural networks and model modification,” *Neural Networks*, vol. 151, pp. 61–69, 7 2022.
[12] Y. Q. Han, “Adaptive tracking control of nonlinear systems with dynamic uncertainties using neural network,” *International Journal of Systems Science*, vol. 49, pp. 1391–1402, 5 2018.
[13] R. Rego and F. Araújo, “Nonlinear control system with reinforcement learning and neural networks based lyapunov functions,” *IEEE Latin and America Transactions*, vol. 19, p. 1253, 2021.
[14] Y. Fu and T. Chai, “Neural-network-based nonlinear adaptive dynamical decoupling control,” *IEEE Transactions on Neural Networks*, vol. 18, pp. 921–925, 5 2007.
[15] S. S. Haykin, *Neural Networks and Learning Machines*. Pearson Education Limited, 2009.
[16] V. Kecman, *Learning and soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*. Te MIT Press, 2001.
[17] S. S. Haykin, *Adaptive filter theory*. Pearson Education Limited, 2014.
[18] G. C. Goodwin and K. S. Sin, *Adaptive filtering prediction and control*. Prentice-Hall, 1984.
[19] *Twin Rotor MIMO System Control Experiments*, Feedback Instruments Limited, 2013.