Neural network ARMAX model for a Furuta pendulum

Modelo de red neuronal ARMAX para un péndulo de Furuta

David Acosta Villamil[1]* Jose Noguera Polania[2]
Jovanny Pacheco Bolivar[1] Marco Sanjuan Mejia[1]

Recibido 18 de julio de 2019, aceptado 1 de junio de 2021
Received: July 18, 2019 Accepted: June 1, 2021

ABSTRACT

The rotational inverted pendulum or Furuta Pendulum is a mechatronic system used by control engineers to explore a various dynamic modeling and control schemes. Due to its nonlinear nature, open-loop instability, and because it is an under-actuated system (more degrees of freedom than actuators), which is the basis for the design of vehicles such as the Segway, the self-balancing scooter, hoverboard, or self-balancing board, among others. The authors present a model for the Furuta Pendulum using the equations of Euler-Lagrange and the methodology to identify a black-box model by training an NNARMAX (Neural Network Auto-Regressive Moving Average with exogenous inputs). The results show that two interconnected MISO-NNARMAX estimates 10-step-ahead predictions accurately for the horizontal and vertical angles.

Keywords: Neural networks, Furuta pendulum, system identification, NN-ARMAX modeling.

RESUMEN

El péndulo invertido rotacional o péndulo de Furuta es un sistema mecatrónico usado por los ingenieros de control para explorar una variedad de modelos dinámicos y sistemas de control debido a su naturaleza no lineal, inestabilidad en lazo abierto y porque es un sistema que posee mas grados de libertad que actuadores, por lo que es la base para el diseño de vehículos como el segway, la tabla de dos ruedas autoequlibrada, aeropatín o tabla flotante entre otros. Los autores presentan un modelo para el péndulo de Furuta usando las ecuaciones de Euler-Lagrange y una metodología para identificar un modelo de caja negra entrenando una NNARMAX (por sus siglas en inglés, Red Neuronal para un Modelo Autoregresivo de Media Movil con Entrada Externa). Los resultados muestran que dos modelos MISO-NNARMAX interconectados estiman de forma precisa predicciones de 10 pasos adelante para los angulos vertical y horizontal.

Palabras clave: Redes neuronales, péndulo de Furuta, identificación de sistemas, modelo NN-ARMAX.

[1] Universidad del Norte. Departamento de Ingeniería Mecánica. Barranquilla, Colombia.
E-mail: villamilr@uninorte.edu.co; jfnoguera@uninorte.edu.co; jpacheco@uninorte.edu.co; msanjuan@uninorte.edu.co
[2] Universidad Cooperativa de Colombia. Facultad de Ingeniería. Campus Santa Marta. Colombia.
E-mail: jose.noguerap@campusucc.edu.co
* Autor de correspondencia: villamilr@uninorte.edu.co

INTRODUCTION

The main objective of this project is to identify a neural network model of a nonlinear mechatronic system, the Furuta pendulum; several aspects of the mechanism must be kept under consideration, aspects which make this specific system to be used vastly in research dealing with modeling and control strategies for mechatronic systems [1] but there is no information about a Neural Network Armax Model applied to this mechatronic system. There are several works on Furuta Pendulum; the authors in [2] study anti-swing controllers for the nominal and disturbed system, they propose a combination of collocated partial feedback linearization and hierarchical sliding mode control inside a two-loop controller. The authors in [3] developed a control for the Furuta pendulum utilizing on–off-type cold gas thrusters as the actuators and an observer-based state feedback controller. Balancing robotic systems is an important topic in robotics [4, 5].

In this way, the most classic is the inverted pendulum problem [6] and variations like the Furuta pendulum [7, 8], and the one or two reaction wheel pendulum [4, 9] where there is a reaction torque on the wheels to balancing the pendulum. The authors in [10] also developed a 3-D reaction wheel cube inverted pendulum, known as Cubli, previously worked by the authors in [11].

All the work developed in the balance of robotic systems, taking the theory behind the inverted pendulum and inverted rotary pendulum, has allowed its use in space systems and satellites [12, 13, 14], in motorcycle systems stabilization [15], in bipedal robot systems [16], in the famous conveyance known like segway [17, 18, 19] and [20], this is a two wheels self-balancing system.

The first aspect to be considered is that the Furuta pendulum is a challenging stabilization problem due to its unstable dynamics, nonlinearities, and especially the fact that it is an under-actuated system [21]. The Lagrange method has had broad use in developing dynamic models as observed in [22] and the literature contained within. The present work develops two mathematical models of the Furuta pendulum; the first is the analytical model based on the equations of motion, and the second is a black-box model, where there is extensive research in neural networks. They are bioinspiration for the development of engineering and used for parameter estimation [23]. Therefore, they will be used to adjust the ARMAX model. The paper is organized as follows: Section 2 describes the Furuta pendulum, the equations of motion are obtained analytically using the Lagrangian formulation. Also, the structure and training method for the neural network model is developed. Section 3 presents the neural network ARMAX model by training and validating the parameters with experimental data. Finally, section 4 shows some conclusions.

METODOLOGY

The Furuta pendulum setup

A simple description of the Furuta pendulum is that of a rigid rod of length L1, connected at the end with another rod of length L2, and attached to the end of this last rod; there is a point mass m.

The first rod is driven by an electric motor of Torque Tm in a plane which we might call the horizontal plane since its most of the time perpendicular to our standing up position. The second rod can spin around the point of contact of the first rod only in a plane perpendicular to the horizontal plane, see Figure 1. The pendulum designed by the Reliability Engineering at the Hamburg University of Technology is actuated with a Maxon RE36 70 W-brushed DC motor, which can provide a torque of 77,1 mNm with a maximum nominal current of 1,82 A. A current controller drives the motor with a Maxon LSC 4-Q-DC servo amplifier. An encoder of reference Avaco-HEDS-5540#A06 mounted in the axis is used to measure the motor angle and has a resolution of 500 Pulses per Revolution (PPR). The pendulum angle β is measured by an incremental encoder reference Kubler-2400, with

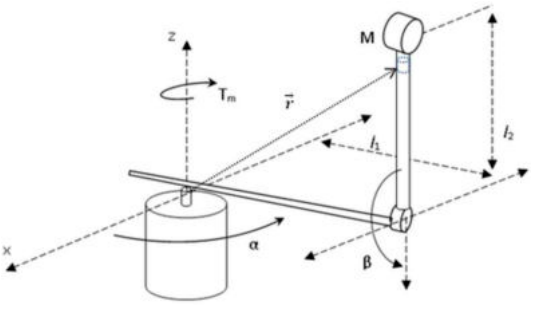

Figure 1. Furuta pendulum.

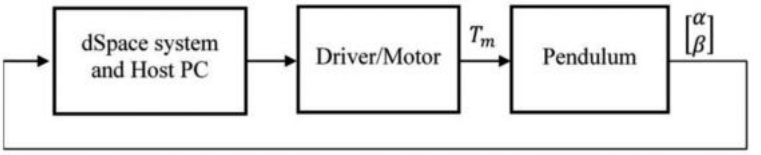

Figure 2. TUHH Furuta pendulum.

a resolution of 1024 PPR. The pendulum control is performed using hardware from dSpace, specifically the CLP1103 Connector Panel, a DS1103 Control Card installed in the Host-PC, and the software dSpace ControlDesk. The measured angles are transmitted through this interface as well as the current controller setpoints. The controller for the pendulum is implemented in Simulink, and the execution in real-time using ControlDesk. From an inspection of the mechanical system, the system has a stable position or stable equilibrium state when $\alpha=0$ and $\beta=\mp\pi$, with the pendulum hanging down and in the absence of movement. Within control theory, it is the objective for a controller to stabilize the pendulum at the upright state given by $\beta = 0$ and any horizontal angle, as shown in Figure 2.

Equations of motion using the Lagrangian method

The Euler-Lagrange (E-L) method takes advantage of the principle of stationary action and is used in this section to obtain the equations of motion for the Furuta pendulum. First, the Lagrangian is defined as the kinetic energy, T, minus the system's potential energy, U, expressed in generalized coordinates, L = K-U. The E-L equations take the following form eq (1):

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right)=\frac{\partial L}{\partial q_i}+Q \tag{1}$$

Where q_i represents the generalized coordinates, and the last term stands for the generalized forces, i.e., the projection of the active forces onto the generalized coordinates direction. For a point r of the pendulum in the coordinate system shown in Figure 1, the position can be expressed as eq (2):

$$\vec{r}=\begin{bmatrix} r_1\cos(\alpha)+r_2\sin(\alpha)\sin(\beta)\\ r_1\sin(\alpha)+r_2\cos(\alpha)\sin(\beta)\\ r_2\cos(\beta)\end{bmatrix}, \tag{2}$$

$$\begin{cases} r_2=0 & 0\le r_1<l_1\\ r_1=l_1 & 0\le r_2<l_2\end{cases}$$

Where r_1 and r_2 are the distances for the first and second link, respectively, measured from the center of rotation. The velocity is obtained with the time derivative eq (3):

$$\vec{v}=\frac{d\vec{r}}{dt}= \begin{bmatrix} -r_1\dot{\alpha}\cos(\alpha)\sin(\beta)+r_2\dot{\alpha}\cos(\alpha)\sin(\beta)+r_2\dot{\beta}\sin(\alpha)\cos(\beta)\\ r_1\dot{\alpha}\cos(\alpha)+r_2\dot{\alpha}\sin(\alpha)\sin(\beta)-r_2\dot{\beta}\cos(\alpha)\cos(\beta)\\ r_2\dot{\beta}\sin(\beta)\end{bmatrix}. \tag{3}$$

The kinetic energy is obtained by adding the contributions of the motor-encoder pair, both links, and the mass eq (4):

$$K_s=K_{m-e}+K_{l_1}+K_{l_2}+K_m. \tag{4}$$

Each of these terms is calculated using the following definition for the kinetic energy eq (5):

$$K=\frac{1}{2}\int \vec{v}\cdot\vec{v}\ dm, \tag{5}$$

And the squared velocity term as eq (6):

$$v(r_1,r_2)^2=\vec{v}\cdot\vec{v}=\left(r_1^2+r_2^2\sin^2(\beta)\right)\dot{\alpha}^2 -2r_1r_2\dot{\alpha}\dot{\beta}\cos(\beta)+r_2^2\dot{\beta}^2. \tag{6}$$

For the motor and encoder eq (7):

$$K_{m-e}=\frac{1}{2}(J_{m-e})\dot{\alpha}^2 \tag{7}$$

For the first link eq (8):

$$K_{l_1}=\frac{1}{2}\int_0^{l_1} r_1^2\dot{\alpha}^2\frac{m_{l_1}}{l_1}ds=\frac{1}{6}m_{l_1}l_1^2\dot{\alpha}^2 \tag{8}$$

For the second link eq (9)- eq (11):

$$K_{l_2}=\frac{1}{2}\int_0^{l_2}\left[l_1^2+r_2^2\sin^2(\beta)\right]\dot{\alpha}^2 -2l_1r_2\dot{\alpha}\dot{\beta}\cos(\beta)+r_2^2\Big]\frac{m_{l_2}}{l_2}ds, \tag{9}$$

$$k_{l_2} = \frac{1}{2}\frac{m_{l_2}}{l_2}\left[\left(l_1^2 l_2 + \frac{l_2^3}{3}\sin^2(\beta)\right)\dot{\alpha}^2 - l_1 l_2^2 \dot{\alpha}\dot{\beta}\cos(\beta) + \frac{l_2^3}{3}\dot{\beta}\cos(\beta) + \frac{l_2^3}{3}\dot{\beta}^2\right], \tag{10}$$

$$K_{l_2} = \frac{1}{2}m_{l_2}\left[\left(l_1^2 + \frac{l_2^2}{3}\sin^2(\beta)\right)\dot{\alpha}^2 - l_1 l_2 \dot{\alpha}\dot{\beta}\cos(\beta) + \frac{l_2^2}{3}\dot{\beta}^2\right], \tag{11}$$

and for the mass eq (12):

$$K_M = \frac{1}{2}M + \left[\left(-l_1^2 + l_2^2\sin^2(\beta)\right)\dot{\alpha}^2 - 2l_1 l_2 \dot{\alpha}\dot{\beta}\cos(\beta) + l_2^2\dot{\beta}^2\right]. \tag{12}$$

The potential energy is obtained by adding the contributions of the motor-encoder pair, both links and the mass eq (13):

$$U_s = U_{m-e} + U_{l_1} + U_{l_2} + U_M, \tag{13}$$

Each of these terms is calculated using the following definition for the potential energy eq (14):

$$U = g\int \vec{r}_z dm. \tag{14}$$

For the motor and encoder eq (15):

$$u_{m-e} - 0, \tag{15}$$

For the first link eq (16):

$$U_{l_1} = 0, \tag{16}$$

For the second link eq (17):

$$U_{l_2} = \frac{1}{2}m_{l_2} g l_2 \cos(\beta), \tag{17}$$

And for the mass eq (18):

$$U_m = Mgl_2\cos(\beta). \tag{18}$$

For the Lagrangian of the system, $L_s = K_s - U_s$, the E-L equations are eq (19)- eq (20):

$$\frac{d}{dt}\left(\frac{\partial L_s}{\partial \dot{\alpha}}\right) = \frac{\partial L_s}{\partial \alpha} - T_m - b_1\dot{\alpha}, \tag{19}$$

And

$$\frac{d}{dt}\left(\frac{\partial L_s}{\partial \dot{\beta}}\right) - b_2\dot{\beta}, \tag{20}$$

where the terms b_1 and b_2 account for a viscous friction model. Solving the differentials concerning velocities and angles for the Lagrangian eq (21):

$$L_s = (J_{m-e})\dot{\alpha}^2 + \frac{1}{6}m_{l_1} l_1^2 \dot{\alpha}^2 + \frac{1}{2}m_{l_2}\left(l_1^2 + \frac{l_2^2}{3}\sin^2(\beta)\right)\dot{\alpha}^2 - l_1 l_2 \dot{\alpha}\dot{\beta}\cos(\beta) + \frac{l_2^2}{3}\dot{\beta}^2\Big] + \frac{1}{2}M\left[\left(l_1^2 + l_2^2\sin^2(\beta)\right)\dot{\alpha}^2 - 2l_1 l_2\dot{\alpha}\dot{\beta}\cos(\beta) + l_2^2\dot{\beta}^2\right] - \frac{1}{2}m_{l_2} g l_2\cos(\beta) - Mgl_2\cos(\beta), \tag{21}$$

And defining eq (22)-(25):

$$\alpha = J_{m-e} + \frac{1}{3}m_{l_1} l_1^2 + m_{l_2} l_1^2 + Ml_1^2, \tag{22}$$

$$b = m_{l_2}\frac{l_2^2}{3} + Ml_2^2, \tag{23}$$

$$c = \frac{1}{2}m_{l_2} l_1 l_2 + Ml_1 l_2, \tag{24}$$

$$d = \frac{1}{2}m_{l_2} g l_2 + Mgl_2. \tag{25}$$

Replacing eq (22)- eq (25) in the L-E equations yields the following equation of motion in matrix form eq (26):

$$\begin{bmatrix} a + b\sin^2(\beta) & -c\cos(\beta) \\ -c\cos & (\beta)b \end{bmatrix}\begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} 2b\sin(\beta)\cos(\beta)\dot{\beta} & c\dot{\beta}\sin(\beta) \\ -b\sin(\beta)\cos(\beta)\dot{\alpha} & 0 \end{bmatrix}\begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} 0 \\ -d\sin(\beta) \end{bmatrix} = \begin{bmatrix} T_m - b_1\dot{\alpha} \\ -b_2\dot{\beta} \end{bmatrix}, \tag{26}$$

Or eq (27):

$$D\begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + C\begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} + g = F, \tag{27}$$

Which can be arranged in the following form to implement it as an S-Function in Simulink (28):

$$\begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} = D^{-1}\left(F - C\begin{bmatrix} \dot{\alpha} \\ \dot{\beta} \end{bmatrix} - g\right), \tag{28}$$

Or eq (29):

$$\frac{d}{dt}\left(\begin{bmatrix}\dot{\alpha}\\ \dot{\beta}\end{bmatrix}\right)=\frac{1}{\left(a+b\sin^2(\beta)\right)}\cdot\begin{bmatrix}b & c\cos(\beta)\\ c\cos(\beta) & a+b\sin^2(\beta)\end{bmatrix}\cdot \left[\begin{bmatrix}T_m-b_1\dot{\alpha}\\ -b_2\beta\end{bmatrix}-\begin{bmatrix}2b\sin(\beta)\cos(\beta)\dot{\beta} & c\dot{\beta}\sin(\beta)\\ -b\sin(\beta)\cos(\beta)\dot{\alpha} & 0\end{bmatrix}\begin{bmatrix}\dot{\alpha}\\ \dot{\beta}\end{bmatrix}-\begin{bmatrix}0\\ -d\sin(\beta)\end{bmatrix}\right]. \quad (29)$$

The values in Table 1 can be used to simulate the Furuta pendulum. These values were obtained experimentally and with the manufacturer's specifications [1].

From a superficial inspection of the equations, nonlinearities become evident; analyzing these nonlinearities escapes the scope of this work. If the reader wants to investigate the phase portraits of a controlled Furuta pendulum, [2] is an informative source.

System identification

The usefulness of mathematical models in science is unprecedented. The increase in computational power and the expansion of common knowledge have brought new ways to attempt to grasp the essence of what is perceived and be able to study it and take advantage of this knowledge. The basic idea behind the System Identification Theory is that a pattern is inferred based on observations of a system, and a mathematical model of the dynamic system can be found. The theory has been applied to biological systems, population growth, market prediction, or what is of interest in this work, mechatronic systems. A system is such an object that upon an input produces an output that is measured and known at specific instants of time. Furthermore, the term dynamic describes that the actual output is dependent not only on previous inputs but also on past states of the system. Ljung, one of the most read researchers in the area, defines three primary entities in the system identification procedure: A data set, a Model Structure, and a rule to assess the candidate models [24]. The first one refers to the input-output data recorded from an experiment on the system. The set of models is the definition of the mathematical precepts that every candidate follows, i.e., a domain in which the desired solution lies; and finally, the rule is the way to identify the best model for that system.

Table 1. Specifications adopted for the simulated Furuta pendulum.

Symbol	Value	Units
J_{m-e}	6.75 x 10^{-6}	$kg\ m^2$
m_{l_1}	0,27	kg
l_1	0.205	m
m_{l_2}	0.09	kg
l_2	0.208	m
M	0.05	kg
g	9.81	$^{m}/_{s_2}$
b_1	0.003	$N^{ms}/_{rad}$
b_2	0.00088	$N^{ms}/_{ra}$

The field of Identification of Linear systems is particularly advanced, but the identification of nonlinear systems is still an area of active research. Considering that most, if not all, real systems are nonlinear, there has been a growing interest in the field of nonlinear system identification. Initially, researchers stressed the connection between biological neural networks and the idea of learning. What the pioneers sought was, as stated by [25], "A computer program that was able to learn from experience". As the NN research progressed, it has shown interesting features to map nonlinear behavior from observations.

Overview of the neural network theory

A neural network is an interconnected set of individual neurons, which take a finite number of inputs and weights them; then, it applies a function over the summation to give the output (see Figure 3). The

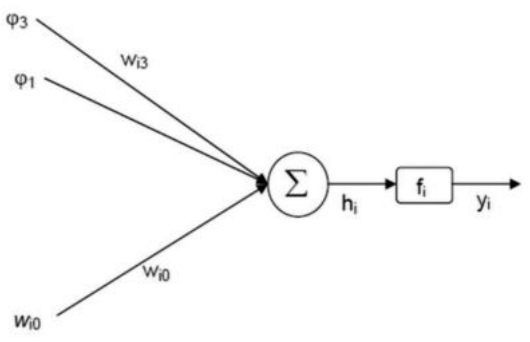

Figure 3. Neuron model.

Multilayer Perceptron (MLP) is a structure of neural network vastly used in the literature, in which layers of neurons are organized, letting the result of the first layer be the input of the subsequent hidden layers until the outer layer. In a two-layer perceptron Sig-Lin NN, the input layer uses a sigmoid averaging function (f_{sig}) and the second layer uses a linear averaging function (F_{lin}), which is the architecture utilized in this work.

In [25], the output of a fully connected MLP is expressed as follows eq (30):

$$\hat{y}_i(t) = g_i[\varphi,\theta] = F_{lin} \left[\sum_{j=1}^{n_h} W_{i,j} fsig\left(\sum_{l=1}^{n_\varphi} w_{j,l}\varphi_l + w_{j,0}\right) + W_{i,0}\right] \quad (30)$$

Where is a parameter vector containing the weights and biases for both layers to be found (this can be defined as $\theta = \left[w_{j,l}\ w_{j,0}\ w_{i,j}\ w_{i,0}\right]^T$); W for the output layer and w for the input layer; n_h and n_φ are the number of neurons in the output and input layer respectively. The inputs are defined by φ, and the output is $\hat{y}_i(t)$. For further detailed concepts on the theory behind the equation (30) and Neural Networks, the reader can consult [25].

The Levenverg-Marquardt algorithm

The problem of identifying the optimal weights of the neural network for a specific set of Input-Output Data is to find a mapping function from the data set to set of candidate models, to find a model which can provide predictions of future outputs which are in some sense close to the real outputs of the system. In formal terms, according to [25], the mapping function is defined by eq (31):

$$Z^N = \left\{\left[u(t), y(t)\right], T = 1,\ldots,N\right\}$$
$$y(t) = \hat{y}(t|\theta) + e(t) = g[t,\theta] + e(t) \quad (31)$$
$$Z^N \rightarrow \hat{\theta}$$

To measure the closeness of the predicted and the real outputs, a mean square criterion is often used eq (32):

$$V_N(\theta, Z^N) = \frac{1}{2N}\sum_{t=1}^{N}\left[y(t) - \hat{y}(t)\right]^2 \quad (32)$$

To solve this optimization problem for this specific work, the Levenberg-Marquardt Algorithm is employed, in which a Gauss-Newton method is used with a slight modification of the search direction. The problem and update rule for the determination of the weights is given by the following equations eq (33)- eq (34), for a more detailed approach look [26].

$$\theta^{(i+1)} = \arg\min_{\theta} L^{(i)}(\theta)$$
$$Subject \left|\theta - \theta^{(i)}\right| \le \delta^{(i)} \quad (33)$$

$$\theta^{(i+1)} = \theta^{(i)} + f^{(i)}$$
$$\left[R\left(\theta^{(i)}\right) + \lambda^{(i)} I\right] f^{(i)} = -G\left|\left(\theta^{(i)}\right)\right. \quad (34)$$

One of the problems found when training an NN is the bias/variance dilemma, which states that the bias error; due to insufficient model structure decreases as more weights are added to the structure. However, the variance error grows in the other direction since the function approximated by the network deviated from the real function. One way to face this problem is using an additional term in the optimization criterion in this case eq (35):

$$W_N(\theta, Z^N) = \frac{1}{2N}\sum_{t=1}^{N}\left[y(t) - \hat{y}(t)\right]^2 + \frac{1}{2N}\theta^T D\theta \quad (35)$$

Where D is often selected as D = d I. The term d represents the weight decay, and again the choice of this tuning knob is grossly heuristic.

It is important to consider that training a Multiple Input Multiple Output (MIMO) system is much more involved. Considerations of the covariance matrix have to be included to treat the model as a whole; nonetheless, a Multiple Input Single Output (SIMO) model was obtained in this work. For the Furuta pendulum, two neural networks will be trained, each of which will use two different control inputs, being one of them the Torque signal and the other one being the past values of the other output, respectively, in other words, for the Neural network that obtains the predictions for the horizontal angle eq (36):

$$u(t) = \begin{bmatrix} T_m(t) \\ \beta(t) \end{bmatrix},$$
$$y(t) = \left[\hat{\alpha}(t)\right], \quad (36)$$
$$\varepsilon_\alpha(t-1) = \left[\alpha(t-1) - \hat{\alpha}(t-1)\right].$$

And a Neural network that obtains the predictions for the vertical angle eq (37):

$$u(t)=\begin{bmatrix} T_m(t) \\ \alpha(t) \end{bmatrix}, y(t)=\left[\beta(t)\right], \quad \varepsilon_\beta(t-1)=\left[\beta(t-1)-\hat{\beta}(t-1)\right]. \tag{37}$$

Once both models are identified (eq (36) and eq(37)), they are combined into a SIMO system.

Validation, examining correlations

In this stage, a validation of the model is performed using a test dataset, which was not included in the training set. This procedure aims to ensure that the Neural Network can simulate the system when new data is presented. One of the tests carried out consists of visualizing the network's predictions either for the next value or several time steps ahead, also known as k-step ahead prediction. The other test is to guarantee a small correlation between the prediction errors and the control input and output signal respectively. If the NN captures all the system's dynamic behavior, then there should not be any correlation.

Application to Furuta pendulum model

This work follows a black-box modeling paradigm, where the estimation is based purely on information retrieved from the system. Nonetheless, the order of the differential motion equations was used to determine the number of neurons on the network. The methodology followed to achieve the model estimation was the one described in [25]. The experiment was the procedure with which the Set of Training and Evaluation Data were retrieved from the system; it would be approached in the next section. The Model Structure selected was an NNARMAX Model. Consequently, a mixed Estimation and Validation iteration was initiated until a satisfactory Neural Network model of the pendulum dynamics was obtained.

The following sections describe and discuss the procedure briefly. In [27], a detailed description of a common framework is given for many model structures and other Nonlinear Black-Box Modelling considerations. Two different experimental setups are arranged, and both correspond to a direct approach scheme, where the output of the process (in this case, the angles) and the input Torque are used, ignoring any possible feedback or reference signal; thus, being able to disregard the complexity of the regulator used in the closed-loop case. A schematic of this approach is shown in Figure 4 and Figure 5.

Figure 4 depicts the flow diagram of the closed-loop experiment. The nonlinear controller derived by Otto

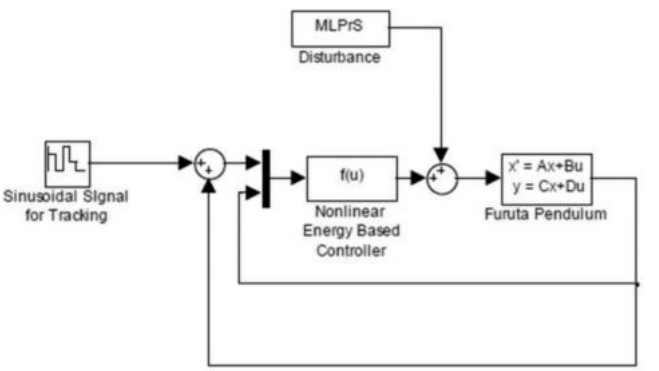

Figure 4. Closed loop experiment scheme.

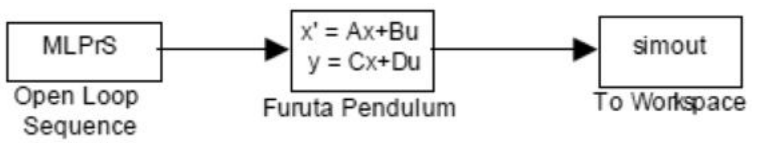

Figure 5. Open-loop experiment scheme.

is used to control the pendulum [1]. This controller has two different modes. The first one is a catching mode, in which the controller attempts to take the pendulum from the equilibrium state and increase its energy until it reaches an angular threshold, in this case just dependent on an angle. Once this energy is achieved, a linear controller takes over and maintains the pendulum on the unstable equilibrium position of $\beta = 0$. The other mode is the tracking mode; in this case, a reference signal is applied to either of the controlled angles. In this experiment, only the vertical angle was forced to follow a sinusoidal signal once stabilized. A perturbation signal was added to the Control Input to broaden the explored regions. The authors generated a sequence of multilevel pseudo-random signals (MLPrS) and used it during some parts of the closed-loop experiment. Figure 5 shows the open-loop diagram. In this case, the input to the Torque was given by a Sequence of MLPrS.

These Experiments were carried out continuously and produced different sets of Data, which was examined to form the definite Training Data for the neural network. The final Experiment for Data Acquisition was automated on Control Desk by a Python Script.

It is worth mentioning at this point that an independent identification of each input-output combination is carried out, obtaining a SISO nonlinear model of each output referenced to the Input Torque. Afterward, both neural networks would be combined into a Single Input Multiple Output NNARMAX model.

Finally, there is one last consideration that will have a profound effect on the control system's performance, the sampling Time Ts. The data is to be retrieved and sent to the system to decide with which frequency. The sampling time determines how often data is retrieved and sent to the system. Employing a series of heuristic sequences of steps with different amplitude and aperiodic triggering, the frequency range was explored between 1 and 6 Hz. This range is where the natural frequencies of the pendulum are believed to lie. During this test, it was inferred that a sampling frequency of 50 Hz, i.e., Ts = 0.02 s, was enough for the neural network model to model the system's dynamics.

Open and closed loop experiments

As stated, experiments have the difficult task of obtaining rich and enough data for the neural network to be able to learn or grasp the system's behavior. The signals from the pendulum must cover the entire operating range, which is unfeasible since it would have to reach every possible position and velocity values. Thus, the training signal must be defined to extract as much information as possible by exciting the system. For this purpose, the authors conducted a mixture of open and closed-loop experiments, in which heuristic changes were applied to the torque signal to obtain the measurement data.

Open-loop experiments

The signal applied should be persistently exciting, i.e., the signal $T_m(t)$ is persistently exciting of order n, if its spectrum $\Phi_u(w)$ is different from zero on at least n points in the interval $-\pi < w \leq \pi$. One of the signals commonly used to identify nonlinear systems is a Multi-Level Pseudo-Random Sequence (MLPrS). The problem of applying this signal to the pendulum is that direct supervision must always take place during the experiment because it might lead to unstable behavior that could affect the integrity of the mechanical system. Obtaining information from the entire operation range is dubiously possible with a finite dataset.

Closed-loop experiments

In this case, the reference signal is zero, the controller uses a Lyapunov-based nonlinear Energy controller to swing up the pendulum, and then a linear controller catches and stabilizes the system near the unstable equilibrium point. Once the system is stabilized, a reference signal might be applied to the vertical angle.

Training and validation data obtained

In Figure 6, the Torque signal T_m is shown, including the open-loop and closed-loop data. The ranges that exhibit a high-frequency signal correspond to the closed-loop experiment, whereas the other regions contain the open-loop data. The open-loop data collected between 0 and 50 sec, used an MLPrS signal as Torque input to the system. Then the closed-loop controller was activated, raising the system's energy until the linear controller would catch and stabilize it, between 50 and 100 sec. Subsequently, a sinusoidal reference signal to the Torque was given to the closed-loop system using the linear controller, as seen between 100 and 115 sec, then after a brief stability period, a sinusoidal but noisy signal was fed to *t*.

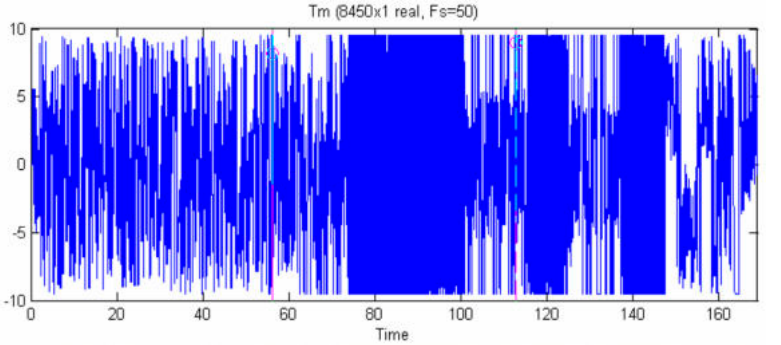

Figure 6. Input torque T_m (t).

Figure 7 shows the data obtained from the experiment. This data would be extensively manipulated within the training of the neural network. The green line corresponds to the vertical Angle β. It is interesting to try to observe the behavior of the pendulum by reading the data; at the beginning, there are some oscillations around the stable origin, then suddenly and short after 60 s the pendulum turned once through the unstable equilibrium state, and then once the closed-loop was enabled the pendulum is stabilized at $\beta = 0$. Afterward, a series of open and closed-loop signals are applied until obtaining a complete set of samples.

Model structure definition and training of the neural network

In the introduction, the authors stated that the model structure to be used in the project is the NNARMAX architecture. First, a small description of the linear ARMAX model structure is given, and then the Neural Network Architecture is presented.

Armax model structure

A linear differential equation exists for a given linear system that describes the input-output relations between considering past values of both and a description of the disturbance entering the system.

In Figure 8, the linear case is depicted in the following considerations eq (38):

$$u(k) = T_m(k),$$
$$y(k) = \begin{bmatrix} \alpha(k) \\ \beta(k) \end{bmatrix}. \tag{38}$$

The disturbance e(k) is assumed to be a white noise signal filtered through H(z-1), which in this

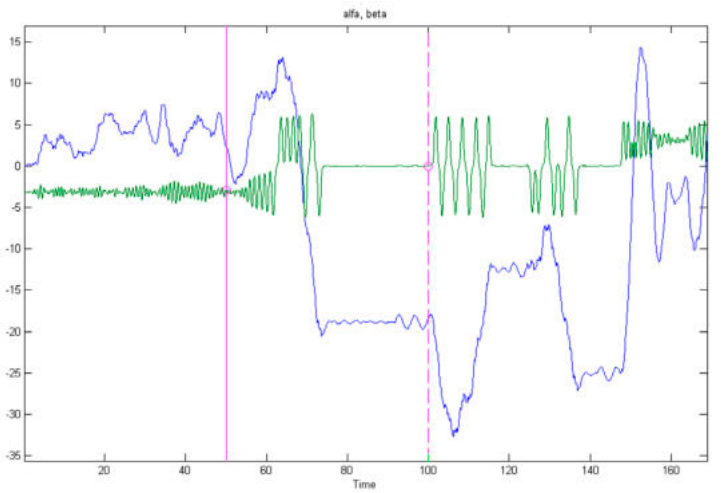

Figure 7. Measured output angles.

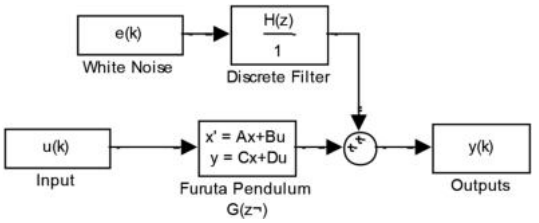

Figure 8. ARMAX structure.

case would represent a model for this disturbance. Using the fact that the expectation of e(k) is zero because it is Gaussian distributed and uncorrelated white noise, a model for predicting future outputs might be encountered.

If the actual output of the system is given by eq (39):

$$y(z) = G\left(z^{-1}\right)u(z) + H\left(z^{-1}\right)e(z), \tag{39}$$

Where eq (40):

$$G\left(z^{-1}\right) = z^{-d}\frac{B\left(z^{-1}\right)}{A\left(z^{-1}\right)}, \qquad H\left(z^{-1}\right) = \frac{C\left(z^{-1}\right)}{D\left(z^{-1}\right)}. \tag{40}$$

Now, the value of represents the time delay of the system, which in this work is assumed to be $d = 1$. The transfer function $H(z^{-1})$ and its inverse are assumed to be stable. The polynomials *A, C* and *D* are monic, i.e., $A(0) = C(0) = D(0) = 1$. Furthermore, the transfer function $G(z^{-1})$ is strictly proper $G(0) = 0$; and the noise model $H(z^{-1})$ is biproper $H(0) = 1$.

Considering the abovementioned assumptions, the error can be defined as follows eq (41):

$$e(k) = y(k) - \hat{y}\left(k\middle|k-1\right), \tag{41}$$

Where $\hat{y}\left(k\middle|k-1\right)$ stands for the predicted output at time k, given values up to time k-1. To close the linear case discussion, the authors introduce the predictor form. Now the predicted output at time k is expressed in terms of past values of the control Input, output variables, as well as the transfer functions H and G, which can be better appreciated in the equation eq (42):

$$\hat{y}\left(y\middle|k-1\right) = H^{-1}Gu(k) + \left(1 - H^{-1}\right)y(k). \tag{42}$$

Now expanding the polynomials in time, the predictor form is encountered eq (43)- eq (45):

$$\hat{y}\left(t\middle|\theta\right) = \varphi^{T}(t,\theta)\theta \tag{43}$$

$$\varphi(t,\theta) = \left[-y(t-1),\ldots,-y(t-na),u(t-d),\right.$$
$$u(t-d-1)\ldots,u(t-d-nb),$$
$$\left.e\left(t-1\middle|\theta\right),\ldots e\left(t-nc\middle|\theta\right)\right]^{T} \tag{44}$$

$$\theta = \left[a_1,\ldots,a_{na}b_1,\ldots,b_{nb},c_1,\ldots,c_{nc}\right] \tag{45}$$

Equation (43) is known as the predictor form, where θ is a parameter vector containing the coefficients of the different polynomials and vector φ is the regression vector containing data from time up to $t = k - 1$.

NNARMAX model structure

The model structure for a NNARMAX is shown in Figure 9.

The number of past values is already introduced from input-output signals which are used in this project. In this case, $na = nb = 4$, the only value that was changed from this initial model structure, shown in Figure 9, once the network during training was the number of past values used in the error signal. Initially, $nc = 4$was selected but as it will be explored in the next section, using more repressors from the error signal caused poor performance of the learning process from the neural network.

With the Levenberg-Marquardt algorithm, a neural network of two layers can map any nonlinear function

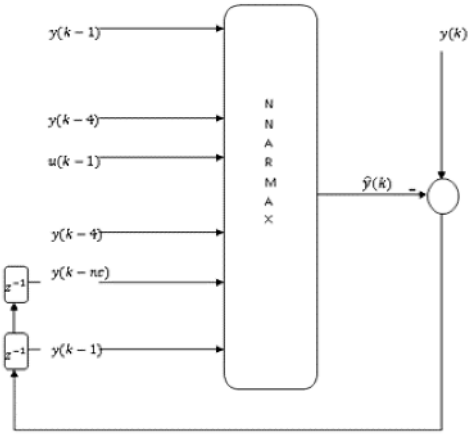

Figure 9. NNARMAX Structure.

provided the necessary set of training data. In this work, a slight modification of the model structure, taken from Werner [28] (see Figure 10), is used to identify the parameters during the training stage of the C polynomial that represents a noise model. This network performs a feedback-like procedure by using past values of the error.

Once the model structure is selected, the number of repressors used in the regression vector should be defined. For SISO cases, it is useful to apply a systematic procedure to obtain the necessary information from the lag space to be able to make a better and more informed selection of the number of repressors used as inputs to the NN, see the paper of [29]. Nevertheless, the extension of this procedure for the MIMO case is beyond the scope of this work. Therefore, a simple evaluation of the differential equations of the dynamic system was used to determine and expect that four past values would be enough to capture the essential pendulum's dynamics.

RESULTS

Training and validation of the NNARMAX for α and β

Once the model structure is selected and the training signals are obtained, the training procedure follows. The Department of Mathematical Modelling of the Technical University of Denmark released a free toolbox for Matlab, written by Nörgaard, in which the algorithms are implemented to conduct NN-based nonlinear system identification. In this project, the toolbox was used to identify two SISO NNs, one for each output.

The data used in the training phase is scaled so that it has zero mean and unitary variance. Then, the Data is divided; a part will be used solely for training purposes while the second part will be used for validating the obtained model. The parameters that define the selected model structure are the number of neurons of each layer and consequently the number of regressors. When the network was trained using 4 past values of the prediction error for the regressor vector, the learning process was unsuccessful; the training algorithm did not converge to the expected global minima. The training was successful only when the last value of the prediction error was used as input to the neural network.

Once the model structure is defined, the parameters used by the training algorithm are selected through the following command:

trparms = settrain (trparms, 'maxiter', 300, 'D', 1e-3,'skip', 10, 'infolevel',1, 'repeat', 100);

A maximum of 300 iterations is forced, with a weight decay factor of 0.001; the 'skip' parameter was set to be 10, indicating that the first 10 samples are not used for training. The IGLS parameter (last function argument), on the other hand, is set at 100, meaning that the procedure to estimate the covariance matrix using an iterated generalized least squares method is repeated 100 times. When the training stage has finished, the weights are rescaled, so that the resulting network can work with unscaled data.

Results of the training

After the training session, the results returned by the toolbox are shown for each output in Figures 11-14. Figure 11 shows the results for the one-step-ahead

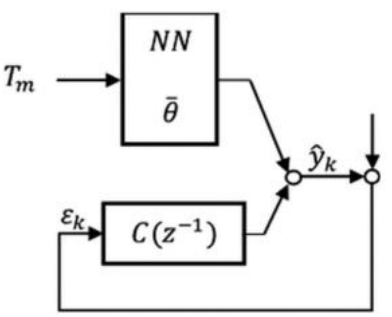

Figure 10. Modified NNARMAX model structure.

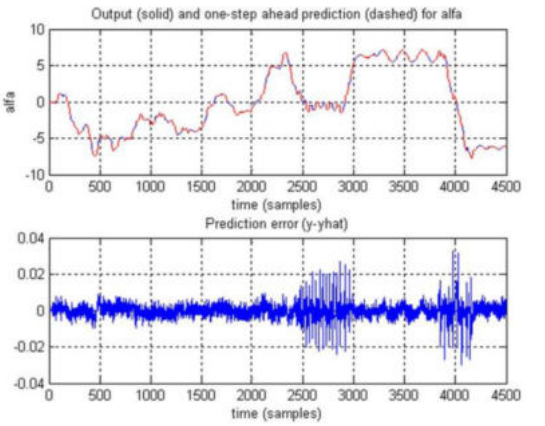

Figure 11. Results of the training session for alfa.

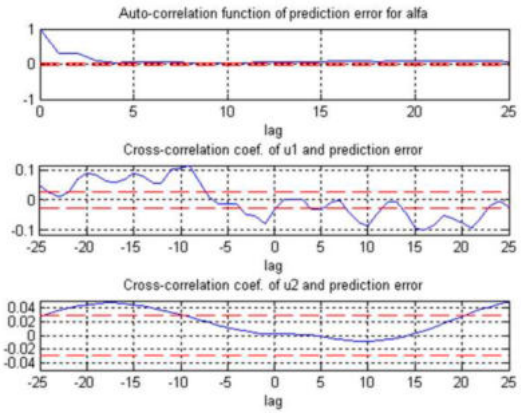

Figure 12. Results of the training session for alfa.

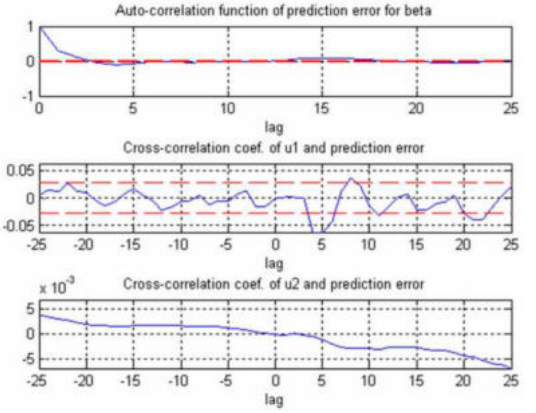

Figure 14. Results of the training session for beta.

prediction of the horizontal angle, and it follows almost an identical path as the observed output. The prediction error is smaller than 0.02 rad for the whole range, which is acceptable. Figure 12 presents three different correlation functions; the first one looks for patterns in the prediction error, the second one shows that the correlation between the torque and the predicted horizontal angle is below 0.1 for all the range, and the authors consider the performance satisfactory.

Finally, the correlation between the output for the vertical angle and the predicted output for the horizontal angle is below 0.05, which is also acceptable. Figure 13 and Figure 14 can undergo a similar analysis, and thus the authors consider that the estimated model adequately captures the system's dynamics.

Validation of the neural network

To fully accept the estimated model, it is necessary to consider that the closed-loop system would be expected to track a periodic signal in the frequency range of 2 to 4 Hz, which implies that the NN must be able to predict enough steps in the future for the dynamics of the system are considered. Consequently, the NN should be able to predict the next 10-time steps accurately. For this purpose, a 10-step-ahead prediction is obtained from each network (see Figure 15 and Figure 16). The authors consider the performance satisfactory; thus, the model is accepted.

Final SIMO – NNARMAX model

To clarify how the complete NN is assembled, consider Figure 17; each network depends on a set of regressors, which are the same for both

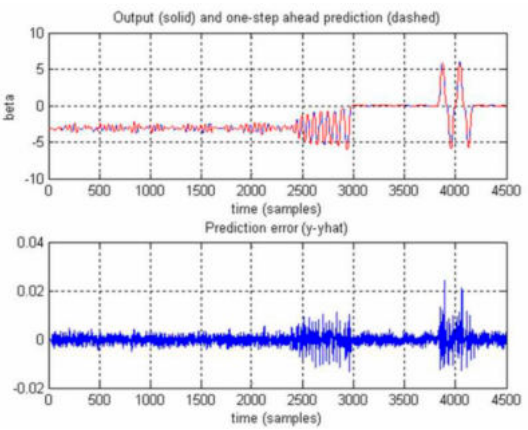

Figure 13. Results of the training session for beta.

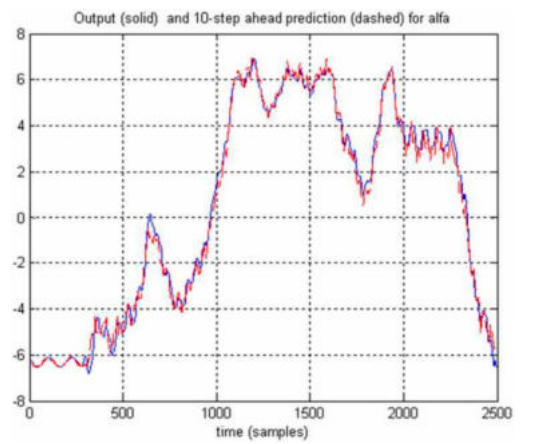

Figure 15. 10-step-ahead prediction of alfa.

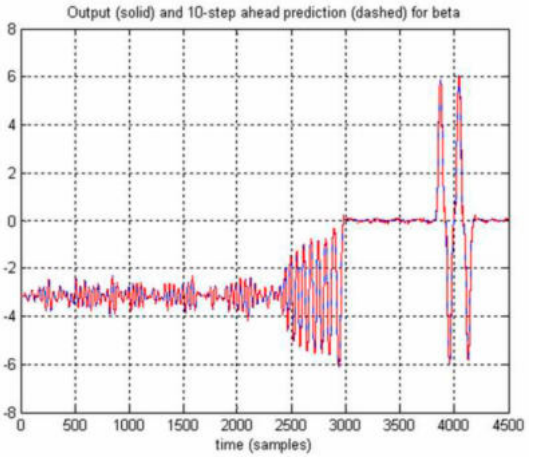

Figure 16. 10-step-ahead prediction of beta.

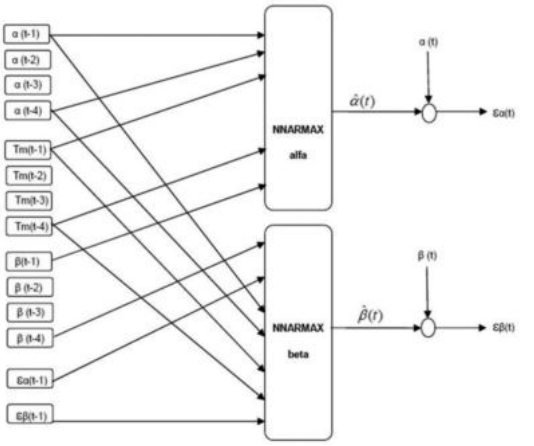

Figure 17. Structure of the identified NNARMAX Model.

networks. The result of arranging the obtained NN for each angular output in parallel forms the SIMO NNARMAX model, with twice as many neurons in the input layer and two neurons in the output layer. Such a model is suitable for controller design purposes.

CONCLUSION

An appropriate model of the Furuta Pendulum using two interconnected MISO-NNARMAX that behave as a SIMO-NNARMAX has been found and estimates accurately 10-step-ahead predictions for the horizontal and vertical angles.

The Identification of nonlinear dynamic systems using an ARX model is more straightforward and less involved as that of an ARMAX model. Furthermore, an adequate study of the frequency range on which data from the system must be retrieved might reduce the size of the data needed for training. The model structure selection is one of the major problems when using neural networks; considering that there is no systematic approach to select the appropriate number of regressors, hints might be available on the differential equations of motion.

In the training stage of the NNARMAX network, the first 4 past values of the prediction error were used as inputs to the NN; the absence of convergence in the training algorithm led the author to believe that a programming error might have occurred or that maybe some unstable issues with the ARMAX model had to be taken into consideration. The possibility that the input signal was not exciting enough the system's dynamics was disregard because a good approximation was found using an NNARX. After several training attempts, a modification on the model structure reducing the number of regressors from four to one lead to convergence of the estimated model. This conclusion might have been obtained by noticing that the network tried to identify higher frequencies, leading to poor performance.

The solution of multivariable systems is considerably more involved than that of a SISO system. Using the NNARMAX model or the linearized ARMAX model obtained from the NN, many controller strategies can be pursued for the Furuta pendulum, including pole placement, optimal control, and minimum variance.

ACKNOWLEDGEMENTS

This work was supported by Universidad del Norte, Contrato de Becario Nº UN-OJ-2017-38173, Contrato de Becario Nº UN-OJ-2017-38125, and Colciencias Convocatoria Nº 727-2015. Also, the experimental data w as obtained in the Institute for Reliability Engineering at the Hamburg University of Technology where the Scholarship E06M103212CO from the Alban Program of the European-Union funded D. Acosta during that time.

REFERENCES

[1] H. Otto. "Entwicklung eines bewertungssystems zur bestimmung der güte verschiedener regelungstrategien am Furuta pendel unter einsatz von Matlab und Simulink". Master Thesis, Institute for Reliability Engineering. Technische Universität Hamburg-Harburg. Hamburg. 2008.

[2] M. Aminsafaee and M. Shafiei. "A robust approach to stabilization of 2-DOF underactuated mechanical systems". Robotica. Vol. 38 Nº 12, pp. 2221-2238. 2020.

[3] Y. Silik and U. Yaman. "Control of rotary inverted pendulum by using on-off type of cold gas thrusters". Actuators. Vol. 9 Nº 95, pp. 1-17. 2020.

[4] G. Neves, B. Angélico and C. Agulhari. "Robust $\mathcal{H}$ 2 controller with parametric uncertainties applied to a reaction wheel unicycle". International Journal of Control. Vol. 93 Nº 10, pp. 2431-2441. 2020.

[5] C. Ott, B. Henze, G. Hettich, T. Seyde, M. Roa, V. Lippi and T. Mergner. "Good posture, good balance: Comparison of bioinspired and model-based approaches for posture control of humanoid robots". IEEE Robotics & Automation Magazine. Vol. 23 Nº 1, pp. 22-33. 2016.

[6] S. Mori, H. Nishihara and K. Furuta. "Control of unstable mechanical system control of pendulum". International Journal of Control. Vol. 23 Nº 5, pp. 673-692. 1976.

[7] J. Aracil, J. Acosta and F. Gordillo. "A nonlinear hybrid controller for swinging-up and stabilizing the Furuta pendulum". Control Engineering Practice. Vol. 21 Nº 8, pp. 989-993. 2013.

[8] K. Furuta, M. Yamakita and S. Kobayashi. "Swing up control of inverted pendulum". International conference on industrial electronics, control and instrumentation (IECON '91). Vol. 3, pp. 2193-2198. 1991.

[9] F. Jepsen, A. Soborg, A. Pedersen and Z. Yang. "Development and control of an inverted pendulum driven by a reaction wheel". International conference on mechatronics and automation (ICMA), pp. 2829-2834. 2009.

[10] M. Muehlebach and R. D'Andrea. "Nonlinear analysis and control of a reaction-wheel-based 3-D inverted pendulum". IEEE Transactions on Control Systems Technology. Vol. 25 Nº 1, pp. 235-246. 2017.

[11] J. Mayr, F. Spanlang and H. Gattringer. "Mechatronic design of a selfbalancing three-dimensional inertia wheel pendulum". Mechatronics. Vol. 30, pp. 1-10. 2015.

[12] M. Chou, C. Liaw, S. Chien, F. Shieh, J. Tsai and H. Chang. "Robust current and torque controls for PMSMdriven satellite reaction wheel". IEEE Transactions on Aerospace and Electronic Systems. Vol. 47 Nº 1, pp. 58-74. 2011.

[13] J. Jin, S. Ko and C. Ryoo. "Fault tolerant control for satellites with four reaction wheels". Control Engineering Practice. Vol. 16 Nº 10, pp. 1250-1258. 2008.

[14] H. Zhou, D. Wang, B. Wu and E. Poh. "Time-optimal reorientation for rigid satellite with reaction wheels". International Journal of Control. Vol. 85 Nº 10, pp. 1452-1463. 2012.

[15] A. Tanos, T. Steffen and G. Mavros. "Improving lateral stability of a motorcycle via assistive control of a reaction wheel". UKACC international conference on control (CONTROL), pp. 80-85. 2014.

[16] T. Brown and J. Schmiedeler. "Reaction wheel actuation for improving planar biped walking efficiency". IEEE Transactions on Robotics. Vol. 32 Nº 5, pp. 1290-1297. 2016.

[17] H. Bin, L. Zhen and L. Feng. "The kinematics model of a twowheeled self-balancing autonomous mobile robot and its simulation". 2nd international conference on computer engineering and applications (ICCEA). Vol. 2, pp. 64-68. 2010.

[18] G. Raffo, M. Ortega, V. Madero and F. Rubio. "Two wheeled self-balanced pendulum workspace improvement via underactuated robust nonlinear control". Control Engineering Practice. Vol. 44, pp. 231-242. 2015.

[19] S. Kim and S. Kwon. "Nonlinear optimal control design for underactuated two-wheeled inverted pendulum mobile platform". IEEE/ASME Transactions on Mechatronics. Vol. 22 Nº 6, pp. 2803-2808. 2017.

[20] J. Kralev, T. Slavov and P. Petkov. "Design and experimental evaluation of robust controllers for a two-wheeled robot". International Journal of Control. Vol. 89 Nº 11, pp. 2201-2226. 2016.

[21] A. Shiriaev, L. Freidovich, A. Robertsson and R. Johansson. "Virtual-constraints-based design of stable oscillations of Furuta pendulum". 45th IEEE Conference on Decision & Control. IEEE Xplore. San Diego, USA, pp. 6144-6149. 2006.

[22] J. Noguera, O. García y C. Robles. "Modeling and control of a robot manipulator using standard control techniques and H infinity". Revista Espacios. Vol. 38 Nº 58, pp. 25-38. 2017.

[23] J. Noguera, N. Portillo y L. Hernández. "Redes neuronales, bioinspiración para el desarrollo de la ingeniería". Ingeniare. Vol. 17, pp. 117-131. 2014.

[24] L. Ljung. "System identification, theory for the user". Prentice Hall. 1st edition. New Jersey, USA. 1999.

[25] M. Nörgaard. "Neural networks for modelling and control of dynamic systems". Springer-Verlag. 1st edition. 2003.

[26] D. Marquardt. "An algorithm for least-squares estimation of nonlinear parameters". Journal of the Society for Industrial and Applied Mathematics. Vol. 11 Nº 2, pp. 431-441. 1963.

[27] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Deylon, P. Glorennec, H. Hjalmarsson and A. Juditsky. "Nonlinear black-box modeling in system identification: A unified overview". Automatica.Vol. 31 N° 12, pp. 1691-1724. 1995. DOI: 10.1016/0005-1098(95)00120-8

[28] H. Werner. "Neural and genetic computing for control engineering". Lecture Notes, Hamburg: Technische Universität Hamburg-Harburg. Hamburg. 2007.

[29] X. He and H. Asada. "A new method for identifying orders of input-output models for nonlinear dynamic systems". American Control Conference IEEE Xplore, pp. 2520-2523. San Francisco, USA. 1993.