# Recurrent neural identification on Xilinx system generator using V7 FPGA for a 2DOF robot manipulator

Ulises Dávalos G.[1], Carlos E. Castañeda[2]
University Center of Los Lagos
University of Guadalajara
Lagos de Moreno, Jalisco México
email: [1]udavalos@cualtos.udg.mx,
[2]ccastaneda@lagos.udg.mx
Tel: +52 (474) 7424314

P. Esquivel.[3], F. Jurado Z.[4]
Technological Institute of la Laguna,
Torreón Coahuila, México
email: [3]pesquivelpr@conacyt.mx,
fjurado@itlalaguna.edu.mx.

Onofre A. Morfín[5]
Autonomous University of Ciudad
Juárez Chihuahua, México
email: [5]omorfin@uacj.mx

***Abstract*—This paper describes an identification process for a class of discrete-time nonlinear systems, which includes the Xilinx system generator software and the process is implemented in a Virtex 7 (V7) field programmable gate array (FPGA). This procedure consists of programming a discrete-time nonlinear plant where the dynamics of this plant is reproduced by a discrete-time recurrent high order neural network (RHONN). The neural network is trained on-line with the extended Kalman filter algorithm where the associated state and measurement noises covariance matrices are composed by the coupled variance between the plant states. Additionally, a sliding window-based method for dynamical modeling of nonstationary systems is presented in order to improve the neural identification process. This identification process is implemented on a Virtex 7 (V7) FPGA using Xilinx system generator software where are programed in this FPGA: the discrete-time dynamics of the two degrees of freedom (2DOF) robot manipulator, the RHONN, the extended Kalman filter (EKF) training algorithm and the sliding window-based method. The obtained results from the FPGA are compared with the results obtained from Matlab/SImulink in order to validate the identification process for the present proposal.**

***Index Terms*—Virtex 7 FPGA, extended KF algorithm, two degrees of freedom robot manipulator.**

## I. Introduction

Many efforts have been carried out in order to reduce the identification error in the process of reproducing the plant behavior, which, in some cases, it is necessary to derive control algorithms in continuous and discrete time. For the particular case when Artificial Neural Networks are used (ANN) to identify the plant dynamics, few information of this plant is required, and to achieve control results. For example, different authors use a RHONN to identify the behavior of a 2DOF vertical robot manipulator in continuous time [1]. The EKF algorithm is used on an ANN to get a good identification process, as it is explained in [2], where the contribution is supported from the training algorithm using a Decentralized Neural Identification in discrete-time; and in [3], where the advantage is for a DC motor system getting control results via simulation in Matlab/Simulink in order to obtain fast learning convergence provided by the EKF. All these works use the EKF algorithm for obtaining good response with excellent results, due to the fact that this algorithm becomes a good identification process. Some authors have been working with neural networks to identify parameters with good results [4]; some of them integrates different kind of on-line identification algorithms as a feed back correction, learning mechanism and predictive controllers [5]; and some that use feed back correction with a Back propagation (BP) algorithm [6] to identify non-linear systems for plants that have many aggressive non-linearities being very hard in the identification process [7]-[11]. Regarding identification process for a 2DOF robot manipulator, for instance using fuzzy logic [12], and where a neural network based direct adaptive dynamic inversion control method is proposed for manipulator system's characteristics of nonlinear time-varying, multivariable, strong coupling and basic control law is designed by nonlinear dynamic inversion method.

On the other hand, the industry has been working for many years with implementations to obtain the best performance of digital circuits. Those implementations have advantages like parallel computing, real-time implementations, reconfiguration process, and low relative coast of implementation. Because of this, many companies use the technology of Field Programmable Gates Array (FPGA); this environment give us the opportunity to solve different problems in the industry and researches about the embedded systems. The FPGA has been in the market with some technologies just like de Complex Programmable Logic Device (CPLD), and digital signal processor (DSP) to mention a few of this systems capable to process digital signals. Using FPGA advantages over the other technologies makes some researchers that have selected these devices let obtain with excellent results from the approximate answers in real-time as in [13]. In [14], a network controlled by a neural network implemented on a FPGA is presented, where favourable results via simulations, and calculating radiation patterns of antenna arrays are obtained using the BP training algorithm. Additionally, some research studies have used a methodology to program nonlinear systems in Xilinx System Generator, as it is explained in [14]-[17]. From this methodology it is possible to design in Xilinx blocks a single source code used to implement a system on a trainer device, where the FPGA is integrated with additional peripherals ready to use.

The main goal of this paper is the elaboration of an useful structure for a non-linear dynamic system, such as a dynamical model of a 2DOF robot manipulator implemented in an FPGA, which constitutes an important novelty regarding the previous work shown in [3]. A RHONN is trained with the extended Kalman filter algorithm for obtaining best results in the identification process. In addition, the embedded system computes on-line the synaptic weights of the neural network identifier within a V7 FPGA. The design process is created in discrete-time with the NN programed in the V7 FPGA using Xilinx System Generator Software as well as using sliding window where the identification error is minimized. This procedure is implemented on a re-configurable device FPGA integrated on a V7 trainer device. It is important to remark that an impotant work for the accurate tracking control of robot manipulator was presented in [18], same that is adopted on this work. Then, some physics variables have been obtained in an experimental evaluation way to drive robots [19]. So, some basics system parameters from this research are used in a 2DOF non-linear system. This dynamic model is useful just because the model is easy to translate to a computed systems, with an advantage that the model is based on Fourier finite harmonic series working with a sine and cosine waves. This model has some aspects that are important to consider, for example the viscous friction and the which are specified in [19]-[21]. As mentioned before, in the present proposed, a RHONN trained with the extended Kalman filter implemented in an FPGA, is shown; also, the identification error is reduced by means of the associated state and measurement noises covariance matrices are composed by the coupled variance between the plant states. This represents a novelty regarding the previous work presented in [25].

This paper is organized as follows: section II describes the nonlinear plant used in this paper, the structure of the NN implemented, the training algorithm and the sliding window which are programed in the V7 FPGA; in order to validate the present proposal, in section III it is shown the real-time results obtained in the V7 FPGA and the comparison with the ones obtained via simulation in Matlab/Simulink; some discussion is explained in section IV; and finally, conclusions are presented at the end of the paper in section V.

## II. Proposed Methodolgy

In this section, we present the main theoretical fundamentals in order to explain the proposed methodology.

### A. Two degrees of freedom robot manipulator model

In order to show the present proposal, we describe a serial $n$-link rigid robot, which is shown in [19] and [22] as:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}\left(\mathbf{q},\dot{\mathbf{q}}\right)\dot{\mathbf{q}} + \boldsymbol{g}(\mathbf{q}) + \boldsymbol{f}\left(\dot{\mathbf{q}}\right) = \boldsymbol{\tau}, \qquad (1)$$

where, for our particular case, where we use a 2DOF robot manipulator, $\mathbf{q}$ is the $2 \times 1$ vector of joint positions, $\dot{\mathbf{q}}$ is the $2 \times 1$ vector of joint velocities, $\boldsymbol{\tau}$ is the $2 \times 1$ vector of applied torques, $\mathbf{M}(\mathbf{q})$ is the $2 \times 2$ symmetric positive definite manipulator inertia matrix, $\mathbf{C}\left(\mathbf{q},\dot{\mathbf{q}}\right)$ is the $2 \times 2$ matrix of centripetal and Coriolis torques, and $\boldsymbol{g}(\mathbf{q})$ is the $2 \times 1$ vector of gravitational torques. Then, system (1) can be represented in state space as:

$$\begin{aligned}\dot{\mathbf{x}_1} &= \mathbf{x}_2\\ \dot{\mathbf{x}_2} &= -\mathbf{M}^{-1}\left[\mathbf{C}\mathbf{x}_2 + \boldsymbol{g}(\mathbf{x}_1) + \boldsymbol{f}\left(\mathbf{x}_2\right)\right] + \mathbf{M}^{-1}\boldsymbol{\tau},\\ \mathbf{y} &= \mathbf{x}_1\end{aligned} \qquad (2)$$

where $\mathbf{x}_1 = \left[q_1 \quad q_2\right]^{\top}$ is the output of the system, $\mathbf{x}_2 = \left[\dot{q}_1 \quad \dot{q}_2\right]^{\top}$, $\mathbf{M} = \mathbf{M}\left(\mathbf{q},\dot{\mathbf{q}}\right) \in \mathbb{R}^2$, $\mathbf{C} = \mathbf{C}\left(\mathbf{q},\dot{\mathbf{q}}\right) \in \mathbb{R}^2$, $\boldsymbol{g}\left(q_1,q_2\right) \in \mathbb{R}^{2\times 1}$, $\boldsymbol{f}\left(\dot{q_1},\dot{q_2}\right) \in \mathbb{R}^{2\times 1}$, and $\boldsymbol{\tau} = \left[\tau_1 \quad \tau_2\right]^{\top}$ is the torque vector as the input signal, the symbol $\top$ indicates transposed vector. Using the Euler method, model (2) can be represented in discrete-time as:

$$\begin{aligned}\mathbf{x}_1(k+1) &= \mathbf{x}_1(k) + T_s\mathbf{x}_2(k)\\ \mathbf{x}_2(k+1) &= \mathbf{x}_2(k) - T_s\mathbf{M}^{-1}[\mathbf{C}\mathbf{x}_2(k) + \boldsymbol{g}(\mathbf{x}_1(k))\\ &\quad + \boldsymbol{f}\left(\mathbf{x}_2(k)\right)] + \mathbf{M}^{-1}\boldsymbol{\tau}(\boldsymbol{k})\\ \mathbf{y}(k) &= \mathbf{x}_1(k)\end{aligned}, \qquad (3)$$

with $\mathbf{x}_1(0) = \left[0 \quad 0\right]^{\top}$ and $\mathbf{x}_2(0) = \left[0 \quad 0\right]^{\top}$, where $k \in Z \cup 0$ is the time index with $Z$ as the set of non-negative numbers and $T_s$ is the sampling time. We use the Euler method due to it represents a good alternative for discretizing nonlinear plants when the sampling rate is very low [23]. It is important to remark that system (2) is discretized due to the fact that the resulting system (3) is programmed by blocks in the V7 FPGA using Xilinx system generator (see Figure 1). In order for simulating system (3) in the V7 FPGA, in this paper we use the model parameters of the 2DOF robot manipulator presented in [19].

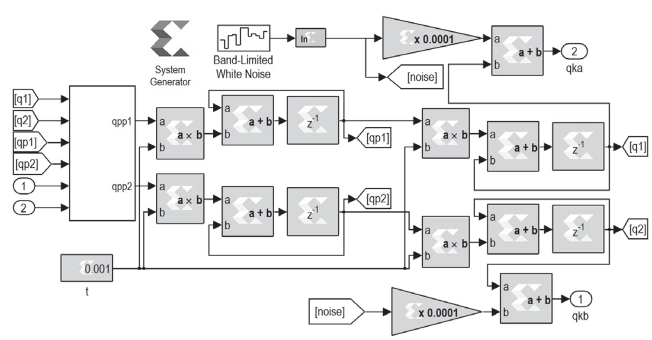


Fig. 1. Discretized model of the robot 2DOF on Xilinx system generator blocks.

### B. Sequential Arquitecture

In this paper, the designing of the 2DOF robot manipulator (3) on Xilinx system generator blocks is explained as follows: we use basics arithmetical, integrators, comparators, selectors and constants blocks running at 0.001 second. These blocks represent a VHDL or Verilog Code inside Xilinx system generator and in the same time the behavior of a synchronous digital circuit; so, when the design is compiled, all the circuits depend on a master clock frequency which is specified in the design. This clock frequency represents some FPGA's work frequencies letting be a training device capable to reconfigure the V7 FPGA and then process the information.

The second internal level of the discretized 2DOF robot manipulator dynamical model design as sequential arquitecture is represented on Figure 2 for each joint of equation (3); as well as the neural network identificator of this second level is presented on Figure 2. This neural network identificator is explained in subsection II-C.

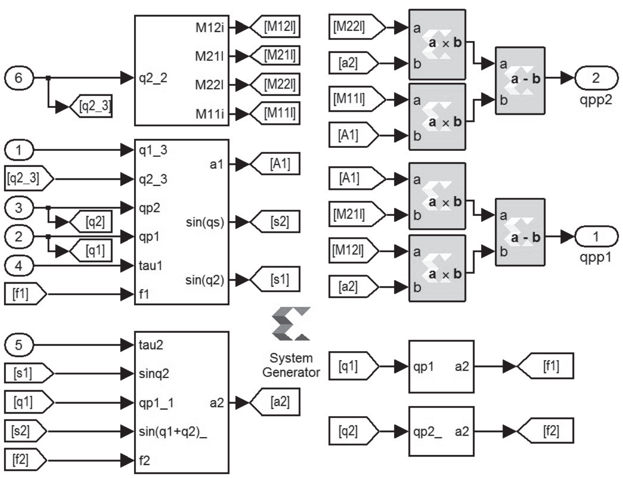


Fig. 2. Second level internal of the discretized 2DOF robot manipulator dynamical model.

programmed in the V7 FPGA; $\bar{w}_1^i$ and $\bar{w}_2^i$ are constant values [25]; $u^i(k)$ represents the $i-$th input signal; $y(k)$ is the output of the system. Finally, $S(\cdot)$ is the activation function defined as in [25] as:

$$F(\chi(k)) = \frac{1}{(1 + e^{-\beta x^i(k)})}, \tag{5}$$

where $x^i(k)$ represents the $i$-th plant state. Figure 3 shows the system (4) programmed in Xilinx system generator, as well as the activation function (5) is shown in Figure 4. Note that the $\chi(k)$ $i,j$-th neuron state of system (4) identifies its corresponding $x(k)i$-th plant state (3) of each joint.

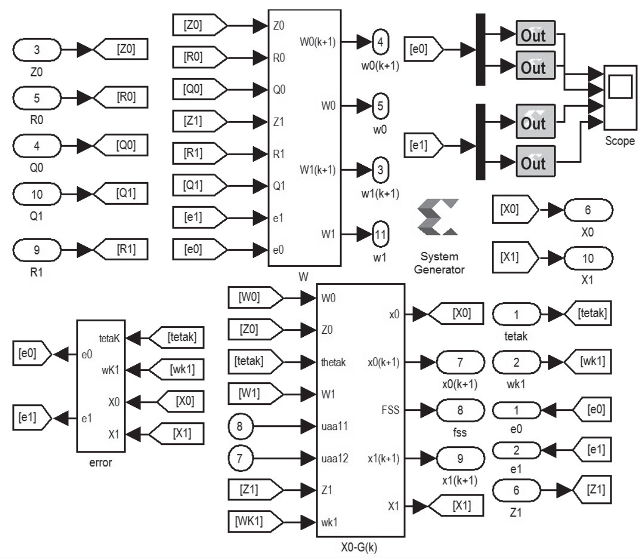


Fig. 3. Neural network Identificator on Xilinx system generator.

## C. Neural Network implementation

In order to reproduce the behavior of the 2DOF discrete-time dynamic model (3), we use a Recurrent High Order Neural Network (RHONN), as it is done in [3]. It is important to remark that the mathematical expression of model (3) is developed on Xilinx system generator in dicrete-time, and it is also validated with Matlab/Simulink. Then, the RHONN proposed for identifying model (3) has the next structure [1], [2] and [24]:

$$\begin{aligned} \chi_1^i(k+1) &= w_1^i(k)S(\chi_1^i(k)) + \bar{w}_1^i\chi_2^i(k) \\ \chi_2^i(k+1) &= w_2^i(k)S(\chi_1^i(k)) + w_3^i(k)S(\chi_2^i(k)) \\ &\quad + \bar{w}_2^i u^i(k) \\ y^i(k) &= \chi_1^i(k) \end{aligned}, \tag{4}$$

where $i = 1, 2$; $\chi_1^i(k)$ and $\chi_2^i(k)$ are the states of the $i-$th neural network, which identify the behavior of the $i-$th plant states, position $x_1^i(k)$ and velocity $x_2^i(k)$, respectively of each joint of system (3). It is important to mention that two neural networks structures are used, one for each joint of system (3) ( $i = 1$ and $i = 2$). $w_1^i(k)$, $w_2^i(k)$ and $w_3^i(k)$ are the synaptic weights calculated on line using the extended Kalman filter

## D. Training Algorithm

In this paper, we use a RHONN in order to obtain an identification process on a V7 FPGA due to the fact that this

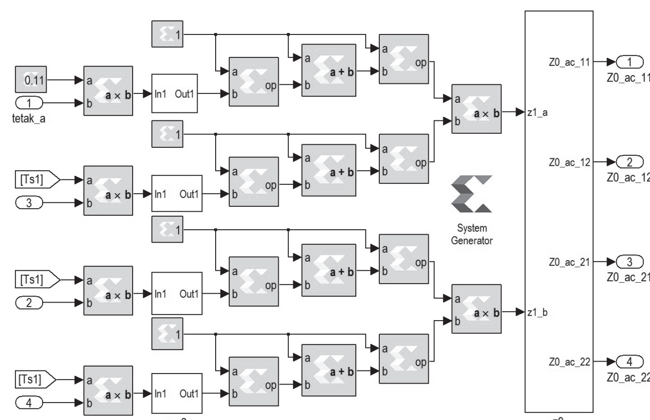


Fig. 4. Activation function $S(\cdot)$ on Xilinx system generator.

procedure represents an efficient identification alternative. The Kalman filtering (KF) estimates the states of a linear system with additive state and output white noises [26], [27]. The neural network mapping is nonlinear system, therefore it is necessary the use of extended KF algorithm [1]. In this paper, we use the extended KF algorithm as it is described in [3]:

$$\begin{aligned} w_j^i(k+1) &= w_j^i(k) + \eta_j^i K_j^i(k) e_j^i(k) \\ K_j^i(k) &= P_j^i(k) H_j^i(k) M_j^i(k) \\ P_j^i(k+1) &= P_j^i(k) - K_j^i(k) H_j^{iT}(k) P_j^i(k) + Q_j^i(k) \end{aligned}, \tag{6}$$

where $i = 1, 2$ is the number of joints, $j =$ is the number of states of each joint, and with:

$$\begin{aligned} M_j^i(k) &= [R_j^i(k) + H_j^{iT}(k) P_j^i(k) H_j^i(k)]^{-1} \\ e_j^i(k) &= [x_j^i(k) - \chi_j^i(k)] \end{aligned}, \tag{7}$$

where $K_j^i(k) \in R^{L_j^i \times m}$ is the gain matrix of Kalman; $L_j^i$ is the respective number of neural network weights; $\chi_j^i(k) \in R^m$ is the $i$-th neuron of the $j$-th plant state; $x_j^i(k) \in R^m$ is $j$-th plant state of the $i$-th joint; $\eta_j^i$ is the learning parameter; $w_j^i(k) \in R^{L_j^i}$ is the synaptic weight vector calculated on line by the V7 FPGA; $Q_j^i(k) \in R^{L_j^i \times L_j^i}$ is the measurement covariance matrix; $R_j^i(k) \in R^{m \times m}$ is the state noise covariance matrix; $e_j^i(k)$ is the prediction identification error; $H_j^i(k) \in R^{L_j^i \times m}$ is a matrix in which each input $H_j^i(k)$ is the derivative of $j$-th neural network state as it is explained in [25]. It is important to mention that the covariance matrices $Q_j^i(k)$, $R_j^i(k)$, and $P_j^i(k)$ are calculated as in [25] as follows: for the time-varying learning algorithm proposed in this paper $P_j^i(k)$, $Q_j^i(k)$ and $R_j^i(k)$ are initialized as diagonal matrices with random entries; $Q_j^i(k)$ and $R_j^i(k)$ are composed of a time-varying coupled covariance between the $j$-th plant state, which allows identification of interactions associated between plant state and to help in the neural convergence. The time-varying formulation implemented in V7 FPGA is the one as proposed in [3] and requires the efficient computation of $Q_j^i(k)$ and $R_j^i(k)$ in time-varying form such that minimizes the identification error:

$$\begin{aligned} &min_{Q_j^i(k)} \left( x_j^i(k) - \chi_j^i(k) \right) \\ &min_{R_j^i(k)} \left( y_j^i(k) - \bar{y}_j^i(k) \right) \end{aligned}, \tag{8}$$

This can be done such that minimizes the variance ($\sigma$):

$$\begin{aligned} \sigma(x(k)) &= \left[ E\left( [x(k) - x_m(k)]^2 \right) \right]^{1/2} \\ \sigma(y(k)) &= \left[ E\left( [y(k) - y_m(k)]^2 \right) \right]^{1/2} \end{aligned}, \tag{9}$$

where $x_m(k) = E(x(k))$ and $y_m(k) = E(y(k))$ are expressed in terms of the recursive expectation value, $E(\bullet)$, which represents the instantaneous mean value of the signal. For the proposed formulation, we consider the state and output to be available for measurements [25].

To get the best identification process, the algorithm calculates the prediction covariance matrices with a recurrent process feeding on-line. In this case, the procedure for minimizing the identification error is programmed in the Xilinx system genrator which includes the calculation of the covariance matrices $Q_j^i(k)$, $R_j^i(k)$, and $P_j^i(k)$ on-line in the V7 FPGA. This is one of the most important contributions of this paper.

## III. Implementation and Simulation Results

The training of the RHONN is performed on-line using the extended KF in series-parallel configuration [2] and it is designed and programmed on Xilinx system generator which is shown in Figure 5 using the neural block identification scheme presented in Figure 6 (this scheme is similar used in the previous work presented in [25]).

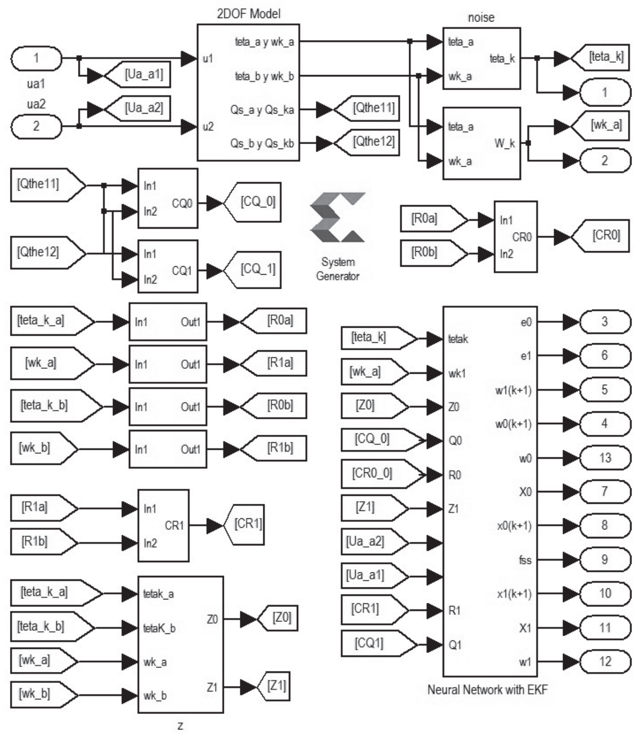


Fig. 5. RHONN architecture designed on system generator Xilinx blocks.

Simulation results are presented as follows: in Figure 7 it is shown the identification results for the first plant state of joint 1, where $x_1^1(k)$ and $\chi_1^1(k)$ correspond to plant state and neural newtork identificator state, respectively, obtainted in the V7 FPGA; $\phi_1^1(k)$ is the identification results validated in Matlab/Simulink. Figure 8 displays the identification error $e_1^1(k)$ (V7 FPGA) and $\hat{e}_1^1(k)$ (Matlab/Simulink). In the same way, the identification results for the first plant state of joint 2 is presented in Figure 9, where $x_2^1(k)$ and $\chi_2^1(k)$ correspond to plant state and the neural network identificator state, respectively, obtainted in V7 FPGA; $\phi_2^1(k)$ is the identification results validated in Matlab/Simulink. Figure 10 displays the identification error $e_2^1(k)$ (V7 FPGA) and $\hat{e}_2^1(k)$ (Matlab/Simulink). For simplicity, only state 1 of each joint in the estimation process are presented.

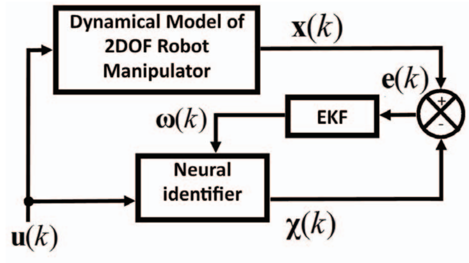


Fig. 6. Neural block identification scheme.

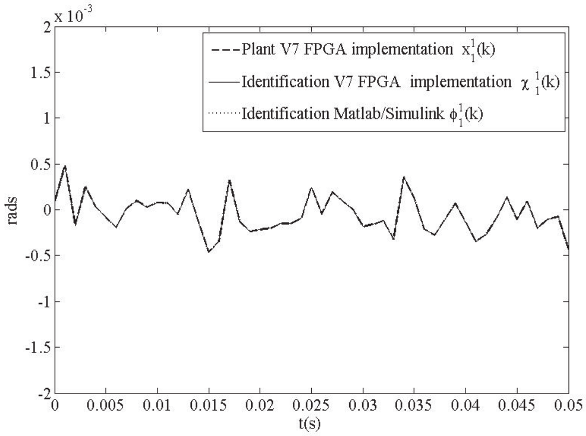


Fig. 7. Identification of the 2DOF robot manipulator for the first plant state of joint 1.

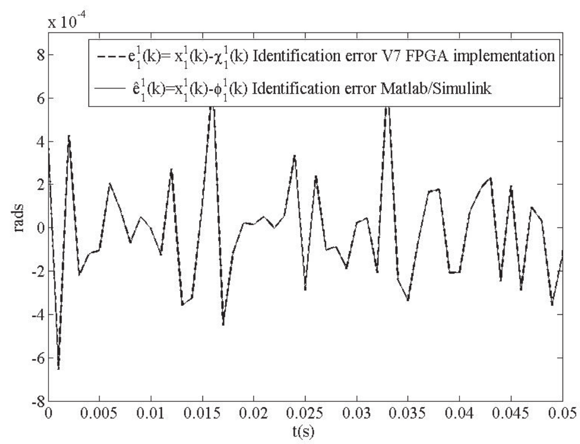


Fig. 8. Identification error for the first plant state of joint 1.

The design of the RHONN architecture on system generator Xilinx blocks shown on Figure 5 is summarized on Figure 6 as a block scheme which shows how the extended KF identification is interconnected with the plant and the neural network identifier and how the prediction error identification is obtained. The results of the identification process shown on Figures 7 and 9 are validated with the 2DOF robot manipulator implemented on a training device Board VC7 Evaluation kit. This board has a V7 FPGA embedded with its parallel architecture. It allows the using a lot of on-line arithmetical operations. On the other hand, through the floating point variables, it is possible a real-time implementation of this kind of dynamical mathematical models, making a successfully hardware parallel processing. It is important to mention that the initial conditions of the identification process shown in Figures 7 and 9 are the same for the plant V7 FPGA, identification V7 FPGA and the validation in Matlab/Simulink. Additionally, the identification error of this RHONN is developed on Xilinx system generator blocks which is implemented on a training device Board VC7 Evaluation kit, which is validated with Matlab/Simulink in discrete-time in Figures 7 and 9. These figures show the process with step time equal to 0.001 second with short time interval from 0 to 0.05 second and show how the identification is done. In Table I, it is shown a comparative analysis of hardware resource utilization between the programming of 2DOF robot manipulator and programming the RHONN 2DOF robot manipulator. This table presents the logic circuits used to the hardware Co-simulation on Xilinx system generator for the plant 2DOF Manipulator with an optimization goal for the implementation on a V7 FPGA. Additionally, Table I displays that the results obtained of RHONN 2DOF, does not increase significantly in comparison with the hardware resource utilization in 2DOF. This result allows the application of a control algorithm which will be implemented in future research.

It is important to mention that the clock frequency of the V7 FPGA is 50 MHz; the interface of the hardware Co-simulation is made with a processor Intel(R) Core(TM) i5-2400 CPU at 3.10 GHz.

In order to validate the corresponding identification process, in Table II it is shown the root mean square error which represents the difference between the original state space variable of angular position of each joint with the ones programmed in

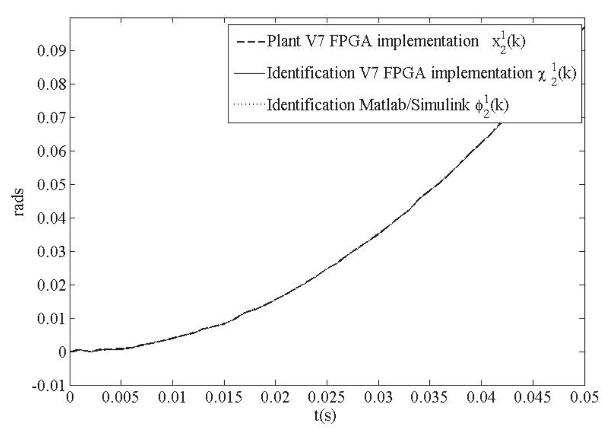


Fig. 9. Identification of the 2DOF robot manipulator for the first plant state of joint 2.

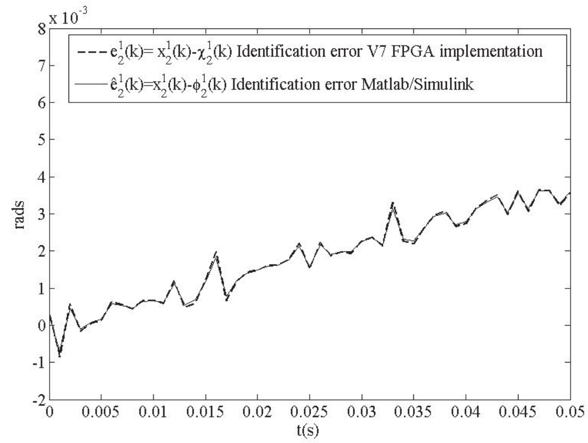


Fig. 10. Identification error for the first plant state of joint 2.

TABLE I
HARDWARE RESOURCE UTILIZATION OF LOGIC CIRCUITS USED TO THE HARDWARE CO-SIMULATION ON XILINX SYSTEM GENERATOR.

| **Device utilization summary** | **2DOF robot manipulator** | | **RHONN 2DOF robot manipulator with EKF** | |
|---|---|---|---|---|
| | **Used** | **Percentage used (%)** | **Used** | **Percentage used (%)** |
| Slices | 47347 | 7 | 62754 | 9 |
| FFs | 343 | 2 | 1983 | 13 |
| BRAMs | 0 | 0 | 0 | 0 |
| LUTs | 168860 | 27 | 242464 | 39 |
| IOBs | 38 | 5 | 76 | 10 |
| Mults/DSP48s | 1507 | 53 | 1818 | 64 |
| TBUFs | 0 | 0 | 0 | 0 |

V7 FPGA and in Matlab/Simulink, respecively. These results show that the identification process is very accurate.

TABLE II
RMS ERROR.

| **Angular Position** | **Identification V7 FPGA** | **Identification Matlab/Simulink** |
|---|---|---|
| Joint 1 | 0.001076958 | 0.001074930 |
| Joint 2 | 0.037816551 | 0.037808345 |

## IV. DISCUSSION

In the work, the authors of the method here presented have implemented a procedure based on the programming dynamical discrete-time systems on a V7 FPGA. The procedure here implemented constitutes a very important step for such that the method can be implemented in future research. Such researches as the implementation of close-loop control in a 2DOF robot manipulator. It is important to mention that the procedure for obtaining system (2) is not considered in this paper.

## V. CONCLUSIONS

This paper has presented an identification process using a V7 FPGA for a class of discrete-time nonlinear systems. This process has shown the implementation of a RHONN as a neural identifier which includes the Xilinx system generator identification. Also, the procedure consists of programming a discrete-time nonlinear plant where the dynamics of this plant is reproduced by a discrete-time RHONN. The neural network is trained on-line with the extended Kalman filter algorithm where the associated state and measurement noises covariance matrices are composed by the coupled variance between the plant states. Additionally, a sliding window-based method for dynamical modeling of nonstationary systems is presented in order to improve the neural identification procedure. This identification process is implemented on a V7 FPGA using Xilinx system generator software where both, a discrete-time 2DOF robot manipulator dynamics and the RHONN are programmed in the V7 FPGA. The validation of the present identification process is done by means of comparing the results obtained from the V7 FPGA using Xilinx system generator with the ones obtained using Matlab/Simulink.


## ACKNOWLEDGMENT

This paper was supported in part by Consejo Nacional de Ciencia y Tecnología (México) under the retention program no. 120489 and by Cátedras CONACYT, and by Tecnológico Nacional de México/Instituto Tecnológico de la Laguna under project no. 1982. The authors also would like to thank to Centro Unversitario de los Lagos and Centro Universitario de los Altos for facilitating the realisation of the present paper.


## REFERENCES


[1] F. Jurado, M. Flores, V. Santibanez, M. A. Llama, and C. E. Castañeda, "Continuous-Time Neural Identification for a 2 DOF Vertical Robot Manipulator", In *Electronics, Robotics and Automotive Mechanics Conference (CERMA), IEEE*, pp. 77–82, 2011.

[2] R. Garcia-Hernandez, E. Sanchez, A. Loukianov, E. Bayro-Corrochano, and V. Santibanez, "Discrete-Time Decentralized Neural Identification and Control for a 2 DOF Robot Manipulator", In *Proc. of the International Control Conference, UKACC*, 2008.

[3] C. E. Castañeda and P. Esquivel, "Decentralized neural identifier and control for nonlinear systems based on extended Kalman filter", *Neural Networks*, vol. 31, pp. 81–87, 2012.

[4] J. Zhang, Y. Li, and C. Wu, "Neural network identification and control for nonlinear dynamic systems with time delay", In *Control Conference (CCC), 34th Chinese*, pp. 3524–3529, 2015.

[5] W. Yongyu and S. Qu, "Predictive Functional Control of Power Kites for High Altitude Wind Energy Generation Based on Hybrid Neural Network", In *Control Conference (CCC), 34th Chinese*, pp. 7866–7870, 2015.

[6] K. J. Nidhil Wilfred, S. Sreeraj, B. Vijay, and V. Bagyaveereswaran, "System Identification Using Artificial Neural Network", In *Circuit, Power and Computing Technologies (ICCPCT), International Conference on*, pp.1-4, 2015.

[7] G. Zhang and Y. Shen, "Novel Conditions on Exponential Stability of a Class of Delayed Neural Networks with State-dependent Switching", *Neural Networks*, vol. 71, pp. 55–61, 2015.

[8] L. Chen, R. Wu, J. Cao, and J. Liu, "Stability and Synchronization of Memristor-based Fractional-order Delayed Neural Networks", *Neural Networks*, vol. 71, pp. 37–44, 2015.

[9] W. Zhang, C. Li, J. Cao, T. Huang, and M. Xiao, "Synchronization of Neural Networks with Stochastic Perturbation via Aperiodically Intermittent Control", *Neural Networks*, vol. 71, pp. 105–111, 2015.

[10] H. Wu and X. Gu, "Towards Dropout Training for Convolutional Neural Networks", *Neural Networks*, vol. 71, pp. 1–10, 2015.

[11] L. Ma, J. W. Minett, T. Blu, and W. Wang, "Resting State EEG-Based Biometrics for Individual Identification Using Convolutional Neural Networks", In *Engineering in Medicine and Biology Society (EMBC), 37th Annual International Conference of the IEEE*, pp. 2848–2851, 2015.

[12] L. Wang, Y. Chen, S. Li, and C. Liu, "2DOF Manipulator Tracking Control Based on Fuzzy CMAC Neural Network Dynamic Inversion", In *Intelligent Computing and Integrated Systems (ICISS) International Conference on*, pp. 315–318, 2010.

[13] E. Monmasson, L. Idkhajine, M. N. Cirstea, I. Bahri, A. Tisan, and M. W. Naouar, "FPGAs in Industrial Control Applications", *Industrial Informatics, IEEE Transactions on*, vol. 7, no. 2, pp. 224–243, 2011.

[14] J. L. Fournier, D. Titz, R. Ferrero, C. Luxey, E. Dekneuvel, and G. Jacquemod, "Phased Array Antenna Controlled by Neural Network FPGA", In *Antennas and Propagation Conference (LAPC), Loughborough*, pp. 1–5, 2011.

[15] M. Bahoura and H. Ezzaidi, "FPGA-Implementation of a Sequential Adaptive Noise Canceller using Xilinx System Generator", In *Microelectronics (ICM), International Conference on*, pp. 213–216, 2009.

[16] M. Bahoura and H. Ezzaidi, "FPGA-Implementation of Wavelet-based Denoising Technique to Remove Power-Line Interference from ECG Signal", In *Information Technology and Applications in Biomedicine (ITAB), 10th IEEE International Conference on*, pp. 1–4, 2010.

[17] M. Bahoura and C. W. Park, "FPGA-Implementation of an Adaptive Neural Network for RF Power Amplifier Modeling", In *New Circuits and Systems Conference (NEWCAS), IEEE 9th International*, pp. 29–32, 2011.

[18] J. J. E. Slotine, "The robust control of robot manipulators", *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 49–64, 1985.

[19] F. Reyes and R. Kelly, 'Experimental evaluation of identification schemes on a direct drive robot", *Robotica*, vol. 15, no. 05, pp. 563–571, 1997.

[20] M. M. Olsen, J. Swevers, and W. Verdonck, "Maximum Likelihood Identification of a Dynamic Robot Model: Implementation Issues", *International Journal of Robotic Research*, vol. 21, no. 2, pp. 89–96, 2002.

[21] J. Swevers, W. Verdonck, and J. De Schutter, "Dynamic Model Identification for Industrial Robots", *Control Systems, IEEE*, vol. 27, no. 5, pp. 58–71, 2007.

[22] M. W. Spong and M. Vidyasagar, *Robot Dynamics and Control*, John Wiley and Sons, N.Y., 1989.

[23] N. Kazantzis and C. Kravaris, "Time discretization of nonlinear control systems via Taylor methods", *Computer and Chemical Engineering*, vol. 23, no. 6, pp. 763–784, 1999.

[24] S. Alvarez, C. E. Castañeda, and F. Jurado, "Neural identification and control using high order sliding modes", In *Proceedings of the Latin American Control Conference (CLCA) and Red Peruana de Control y Automatizacin (REPCA)*, pp. 23–26, 2012.

[25] C. E. Castañeda, A. G. Loukianov, E. N. Sanchez, and B. Castillo-Toledo, "Discrete-time neural sliding-mode block control for a dc motor with controlled flux", *Industrial Electronics, IEEE Transactions on*, vol. 59. no. 2, pp. 1194–1207, 2012.

[26] R. Grover and P. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, Jhonn Wiley and Sons, Inc., N. Y., 1992.

[27] S. Haikin, *Kalman Filtering and Neural Networks*, Jhonn Wiley and Sons, Inc, N. Y., 2001.