Discrete-Time Neural Identifier for Linear Induction Motors

Alma Y. Alanis[1], Jorge Rivera[1], Eduardo Rangel[1] and Gustavo Hernandez[1]

Abstract— This paper focusses on a discrete-time neural identifier applied to a Linear Induction Motor (LIM) model, whose model is assumed to be unknown. This neural identifier is robust in presence of external and internal uncertainties. The proposed scheme is based on a discrete-time recurrent high order neural network (RHONN) trained with a novel algorithm based on extended Kalman filter (EKF) and particle swarm optimization (PSO), using an off-line series-parallel configuration. Experimental results are included in order to illustrate the applicability of the proposed scheme.

Keywords— **Linear Induction Motor, Recurrent high order neural networks, Kalman filtering learning, Discrete-time nonlinear systems, Neural identifier.**

I. Introduction

Linear induction motors (LIM) is a special electrical machine, in which the electrical energy is converted directly into mechanical energy of translatory motion. Strongest interest on these machines raised in the early 1970, however, in the late 1970, the research intensity and number of publications dropped. After 1980, LIM found their first noticeable applications in, among others, transportation industry, automation, and home appliances [8], [4]. LIM has many excellent performance features such as high-starting thrust force, elimination of gears between motor and motion devices, reduction of mechanical loses and the size of motion devices, high speed operation, silence, and so on [4], [20]. The driving principles of the LIM are similar to the traditional rotary induction motor (RIM), but its control characteristics are more complicated than the RIM, and the parameters are time varying due to the change of operating conditions, such as speed, temperature, and rail configuration.

Modern control systems usually require detailed knowledge about the system to be controlled; such knowledge should be represented in terms of differential or difference equations. This mathematical description of the dynamic system is named as the model. There can be different motives for establishing mathematical descriptions of dynamic systems, such as: simulation, prediction, fault detection, and control system design. In this sense, basically there are two ways to obtain a model; it can be derived in a deductive manner using physics laws, or it can be inferred from a set of data collected during a practical experiment. The first method can be simple, but in many cases is excessively time-consuming; it would be unrealistic or impossible to obtain an accurate model in this way. The second method, which is commonly referred as system identification [22], could be a useful short cut for deriving mathematical models. Although system identification not always results in an accurate model, a satisfactory one can be often obtained with reasonable efforts. The main drawback is the requirement to conduct a practical experiment, which brings the system through its range of operation [6].

Due to their nonlinear modeling characteristics, neural networks have been successful applied in control systems, pattern classification, pattern recognition, and identification problems. The best well-known training approach for recurrent neural networks (RNN) is the back propagation through time [10]. However, it is a first order gradient descent method, and hence its learning speed could be very slow. Another well-known training algorithm is the Levenberg–Marquardt one [10]; its principal disadvantage is that is not guarantee it will find the global minimum and its learning speed could be slow too, this depends on the initialization. In past years, Extended Kalman Filter (EKF) based algorithms has been introduced to train neural networks [1]. With the EKF based algorithm, the learning convergence is improved [10]. The EKF training of neural networks, both feedforward and recurrent ones, has proven to be reliable for many applications [10]. However, EKF training requires the heuristic selection of some design parameters which not always are an easy task [1].

On the other hand Particle Swarm Optimization (PSO) technique, which is based on the behavior of a flock of birds or school of fish, is a type of evolutionary computing technique [14]. It has been shown that the PSO training algorithm takes fewer computations and is faster than the BP algorithm for neural networks to achieve the same performance [14].

In this paper a recurrent high order neural network (RHONN) is used to design the proposed neural identifier for nonlinear systems, whose mathematical model is assumed to be unknown. The learning algorithm for the RHONN is implemented using an Extended Kalman Filter with particle swarm optimization (EKF-PSO) based algorithm. We consider a class of Multi-Input Multi-Output (MIMO) discrete-time nonlinear system, for which we develop a neural identifier [16]; then this identifier is applied to a discrete-time unknown nonlinear system. This identifier is based on a recurrent high

[1] CUCEI, Universidad de Guadalajara, Av. Revolucion 1500, Col. Olimpica, C.P. 44430, Guadalajara, Jalisco, Mexico, e-mail: almayalanis@gmail.com

order neural network (RHONN) [17], which identify the model of the unknown plant dynamics. The applicability of these schemes is illustrated via experimental for a Linear Induction Motor (LIM).

II. Preliminaries

Through this paper, we use k as the sampling step, $k \in \mathbb{N}$, $|\bullet|$ as the absolute value and, $\|\bullet\|$ as the Euclidian norm for vectors and as any adequate norm for matrices.

Consider a MIMO nonlinear system:

$$\chi(k+1) = F(\chi(k), u(k)) \quad (1)$$
$$y(k) = h(x(k)) \quad (2)$$

where $\chi \in \Re^n$,$u \in \Re^m$, and $F \in \Re^n \times \Re^m \rightarrow \Re^n$ is a nonlinear function.

A. The EKF Training Algorithm

It is known, that Kalman filtering (KF) estimates the state of a linear system with state and output additive white noises [9]. For KF-based neural network training, the network weights become the states to be estimated. In this case the error between the neural network output and the measured plant output can be considered as the additive white noise [10]. Although the white noise assumption is seldom satisfied, the developed algorithm has proven to be efficient in real applications [1], [10]. Due to the fact that the neural network mapping is nonlinear, an EKF-type is required [17]. The training goal is to find the weight values which minimize the prediction error. In this paper, we use an EKF-based training algorithm described by

$$\begin{aligned} w_i(k+1) &= w_i(k) + \eta_i K_i(k) e(k) \quad (3)\\ K_i(k) &= P_i(k) H_i(k) M_i(k) \\ P_i(k+1) &= P_i(k) - K_i(k) H_i^{\top}(k) P_i(k) + Q_i(k) \\ i &= 1, \cdots, n \end{aligned}$$

with

$$M_i(k) = \left[R_i(k) + H_i^{\top}(k) P_i(k) H_i(k)\right]^{-1} \quad (4)$$
$$e(k) = y(k) - \widehat{y}(k) \quad (5)$$

where $e(k) \in \Re^p$ is the output estimation error and $P_i(k) \in \Re^{L_i \times L_i}$ is the weight estimation error covariance matrix at step k, $w_i \in \Re^{L_i}$ is the weight (state) vector, L_i is the respective number of neural network weights, $y \in \Re^p$ is the plant output, $\widehat{y} \in \Re^p$ is the NN output, n is the number of states, $K_i \in \Re^{L_i \times p}$ is the Kalman gain matrix, $Q_i \in \Re^{L_i \times L_i}$ is the NN weight estimation noise covariance matrix, $R_i \in \Re^{p \times p}$ is the error noise covariance, and $H_i \in \Re^{L_i \times p}$ is a matrix, in which each entry (H_{i_j}) is the derivative of the ith neural output with respect to ijth neural network weight (w_{i_j}), given as follows:

$$H_{i_j}(k) = \left[\frac{\partial y(k)}{\partial w_{i_j}(k)}\right]^{\top} \quad (6)$$

where $i = 1, ..., n$ and $j = 1, ..., L_i$. Usually P_i, Q_i and R_i are initialized as diagonal matrices, with entries $P_i(0)$, $Q_i(0)$ and $R_i(0)$, respectively. Due to typically the entries $P_i(0)$, $Q_i(0)$ and $R_i(0)$ are defined heuristically, in this paper we propose the use of a PSO algorithm in order to compute off-line such entries to improve the EKF training algorithm, as follows.

B. PSO improvement for EKF Training Algorithm

Particle swarm optimization (PSO) is a swarm intelligence technique developed by Kennedy and Eberhart in 1995 [13]. In fact, natural flocking and swarm behavior of birds and insects inspired him to PSO. This technique has been used in several optimization and engineering problems ([14], [21]). In the basic PSO technique proposed by Kennedy and Eberhart [13], great number of particles moves around in a multi-dimensional space and each particle memorizes its position vector and velocity vector as well as the time at which the particle has acquired the best fitness. Furthermore, related particles can share data at the best-fitness time. The velocity of each particle is updated with the best positions acquired for all particles over iterations and the best positions are acquired by the related particles over generations [23].

To improve the performance of the basic PSO algorithm, some new versions of it have been proposed. At first, the concept of an inertia weight was developed to better control exploration and exploitation in [14], [19], [23]. Then, the research done by Clerc [5] indicated that using a constriction factor may be necessary to insure convergence of the particle swarm algorithm. After these two important modifications in the basic PSO, the multiphase particle swarm optimization (MPSO), the particle swarm optimization with Gaussian mutation, the quantum particle swarm optimization, a modified PSO with increasing inertia weight schedule, the Gaussian particle swarm optimization (GPSO) and the guaranteed convergence PSO (GCPSO) were introduced in [2], respectively.

In this paper the algorithm proposed in [14] is used in order to determine the design parameters for the EFK-Learning algorithm. Initially a set of random solutions or a set of particles are considered. A random velocity is given to each particle and they are flown through the problem space. Each particle has memory which is used to keep track of the previous best position and corresponding fitness. The best value of the position of each individual is stored as p_{id}. In other words, p_{id} is the best position acquired by an individual particle during the course of its movement within the swarm. It has another value called the p_{gd}, which is the best value of all the particles p_{id} in the swarm. The basic concept of the PSO technique lies in accelerating each particle towards its p_{id} and p_{gd} locations at each time step. The PSO algorithm used in this paper is defined as follows [14]:

1. Initialize a population of particles with random posi-

tions and velocities in the problem space.
2. For each particle, evaluate the desired optimization fitness function.
3. Compare the particles fitness evaluation with the particles p_{id} if current value is better than the p_{id} then set p_{id} value equal to the current location.
4. Compare the best fitness evaluation with the population's overall previous best. If the current value is better than the p_{gd}, then set p_{gd} to the particle's array and index value.
5. Update the particle's velocity and position as follows: The velocity of the ith particle of d dimension is given by:

$$\begin{aligned} v_{id}\left(k+1\right) &= c_0 v_{id}\left(k\right) + c_1 \operatorname{rand}_1 \left(p_{id}\left(k\right) - x_{id}\left(k\right)\right) \\ &\quad + c_2 \operatorname{rand}_2 \left(p_{gd}\left(k\right) - x_{id}\left(k\right)\right) \end{aligned}$$

The position vector of the ith particle of d dimension is updated as follows:

$$x_{id}\left(k+1\right) = x_{id}\left(k\right) + v_{id}\left(k\right)$$

where c_0 is the inertia weight, c_1 is the cognition acceleration constant and c_2 is the social acceleration constant.
6. Repeat the step 2 until a criterion is met, usually a sufficiently good fitness or a maximum number of iterations or epochs.

In case the velocity of the particle exceeds V_{max} (the maximum velocity for the particles) then it is reduced to V_{max}. Thus, the resolution and fitness of search depends on the V_{max}. If V_{max} is too high, then particles will move in larger steps and so the solution reached may not be the as good as expected. If V_{max} is too low, then particles will take a long time to reach the desired solution [14]. Due the above explained PSO are very suitable models of noisy problems, as the one we are considering.

Since PSO has shown good results in optimization problems [14] it will be used to optimize the values for Kalman's filter covariance matrices instead of heuristic solutions. For this purpose, each particle will represent one of the Kalman's covariance entries.

III. Neural Identification

In this section, we consider the problem to identify the nonlinear system

$$\chi\left(k+1\right) = F\left(\chi\left(k\right), u\left(k\right)\right) \tag{7}$$

where $\chi \in \Re^n$, $u \in \Re^m$ and $F \in \Re^n \times \Re^m \rightarrow \Re^n$ is nonlinear function. To identify the system (7), we use a RHONN defined as:

$$x_i(k+1) = w_i^{\top} z_i(x(k), u(k)), \quad i = 1, \cdots, n \tag{8}$$

where x_i $(i = 1, 2, \cdots, n)$ is the state of the i-th neuron, L_i is the respective number of higer-order connections, $\{I_1, I_2, \cdots, I_{L_i}\}$ is a collection of non-ordered subsets of $\{1, 2, \cdots, n+m\}$, n is the state dimension, m is the number of external inputs, w_i $(i = 1, 2, \cdots, n)$ is the respective on-line adapted weight vector.

Consider the problem to approximate the general discrete-time nonlinear system (7), by the following discrete-time RHONN series-parallel representation [18]:

$$\chi_i\left(k+1\right) = {w_i^*}^{\top} z_i\left(x(k), u(k)\right) + \epsilon_{z_i}, \quad i = 1, \cdots, n \tag{9}$$

where χ_i is the i-th plant state, ϵ_{z_i} is a bounded approximation error, which can be reduced by increasing the number of the adjustable weights [18]. Assume that there exists an ideal weights vector w_i^* such that $\|\epsilon_{z_i}\|$ can be minimized on a compact set $\Omega_{z_i} \subset \Re^{L_i}$ The ideal weight vector w_i^* is an artificial quantity required for analytical purpose [18]. In general, it is assumed that this vector exists and is constant but unknown and w_i is its estimated. The RHONN is trained with a EKF-PSO algorithm as defined above.

IV. Linear Induction Motor Application

In this section, we apply the above developed scheme to identify a three-phase linear induction motor. It is important to note that the proposed scheme is developed assuming that the plant model, parameters as well as external disturbances (load torque) are unknown.

A. Motor Model

In order to illustrate the applicability of the proposed scheme, in this section, the proposed neural identifier is applied to the $\alpha - \beta$ model of a LIM discretized by the Euler technique, which is considered unknown, [3], [15], [12] as follows

$$\begin{aligned}
q_m(k+1) &= q_m(k) + v(k)T \\
v(k+1) &= (1 - K_2 T)v(k) - k_1 T \lambda_{r\alpha}(k)\rho_1 i_{s\alpha}(k) \\
&\quad - k_1 T \lambda_{r\beta}(k)\rho_2 i_{s\alpha}(k) \\
&\quad + k_1 T \lambda_{r\alpha}(k)\rho_2 i_{s\beta}(k) \\
&\quad - k_1 T \lambda_{r\beta}(k)\rho_1 i_{s\beta}(k) - k_3 T F_L \\
\lambda_{r\alpha}(k+1) &= (1 - k_6 T)\lambda_{r\alpha}(k) + k_4 T v(k)\rho_1 i_{s\alpha}(k) \\
&\quad - k_4 T \rho_1 i_{s\alpha}(k) + k_5 T \rho_2 i_{s\alpha}(k) \\
&\quad + k_4 T \rho_2 i_{s\beta}(k) - k_4 T v(k)\rho_2 i_{s\beta}(k) \\
&\quad + k_5 T \rho_1 i_{s\beta}(k) \\
\lambda_{r\beta}(k+1) &= (1 - k_6 T)\lambda_{r\beta}(k) + k_4 T v(k)\rho_2 i_{s\alpha}(k) \\
&\quad - k_4 T \rho_2 i_{s\alpha}(k) - k_5 T \rho_1 i_{s\alpha}(k) \\
&\quad + k_4 T \rho_1 i_{s\beta}(k) + k_4 T v(k)\rho_1 i_{s\beta}(k) \\
&\quad + k_5 T \rho_2 i_{s\beta}(k) \\
i_{s\alpha}(k+1) &= (1 + k_9 T) i_{s\alpha}(k) - k_7 T \lambda_{r\alpha}(k)\rho_2 \\
&\quad - k_8 T \lambda_{r\alpha}(k) v(k)\rho_1 + k_7 T \lambda_{r\beta}(k)\rho_1 \\
&\quad - k_8 T \lambda_{r\beta}(k) v(k)\rho_2 - k_{10} T u_\alpha(k) \\
i_{s\beta}(k+1) &= (1 + k_9 T) i_{s\beta}(k) + k_8 T \lambda_{r\alpha}(k) v(k)\rho_2 \\
&\quad - k_7 T \lambda_{r\alpha}(k)\rho_1 - k_7 T \lambda_{r\beta}(k)\rho_2 \\
&\quad - k_8 T \lambda_{r\beta}(k) v(k)\rho_1 - k_{10} T u_\beta(k)
\end{aligned} \tag{10}$$

with

$$\begin{aligned}
\rho_1 &= \sin(n_p q_m(k)), \quad \rho_2 = \cos(n_p q_m(k)) \\
k_1 &= \frac{n_p L_{sr}}{D_m L_r}, & k_2 &= \frac{R_m}{D_m} \\
k_3 &= \frac{1}{D_m}, & k_4 &= n_p L_{sr} \\
k_5 &= \frac{R_r L_{sr}}{L_r}, & k_6 &= \frac{R_r}{L_r} \\
k_7 &= \frac{L_{sr} R_r}{L_r(L_{sr}^2 - L_s L_r)}, & k_8 &= \frac{L_{sr} n_p}{L_{sr}^2 - L_s L_r} \\
k_9 &= \frac{L_r^2 R_s + L_{sr}^2 R_r}{L_r(L_{sr}^2 - L_s L_r)}, & k_{10} &= \frac{L_r}{L_{sr}^2 - L_s L_r}
\end{aligned}$$

where $q_m(k)$ is the position, $v(k)$ is the linear velocity, $\lambda_{r\alpha}(k)$ and $\lambda_{r\beta}(k)$ are the α and β secondary flux components, respectively, $i_{s\alpha}(k)$ and $i_{s\beta}(k)$ are the α and β primary current components, respectively, $u_{s\alpha}(k)$ and $u_{s\beta}(k)$ are the α and β primary voltage components, respectively, R_s is the winding resistance per phase, R_r is the secondary resistance per phase, L_{sr} is the magnetizing inductance per phase, L_s is the primary inductance per phase, L_r is the secondary inductance per phase, F_L is the load disturbance, D_m is the viscous friction and iron-loss coefficient and n_p is the number of poles pairs, T is the sampling period [3].

B. Neural Identifier Design

The neural identifier proposed is designed as follows:

$$\begin{aligned}
x_1(k+1) &= w_{11}(k)\, S(v(k)) + w_{12}(k)\, S(q_m(k)) \\
x_2(k+1) &= w_{21}(k)\, S(v(k))^2 + w_{22}(k)\, S(\lambda_{r\alpha}(k))^2 \\
&\quad + w_{23}(k)\, S(\lambda_{r\beta}(k))^{15} \\
x_3(k+1) &= w_{31}(k)\, S(v(k))^2 + w_{32}(k)\, S(\lambda_{r\alpha}(k))^2 \\
&\quad + w_{33}(k)\, S(\lambda_{r\beta})^{15} \\
x_4(k+1) &= w_{41}(k)\, S(v(k))^2 + w_{42}(k)\, S(\lambda_{r\alpha}(k))^2 \\
&\quad + w_{43}(k)\, S(\lambda_{r\beta}(k))^{15} \\
x_5(k+1) &= w_{51}(k)\, S(v(k))^2 + w_{52}(k)\, S(\lambda_{r\alpha}(k))^2 \\
&\quad + w_{53}(k)\, S(\lambda_{r\beta}(k))^2 + w_{54}(k)\, S(i_{s\alpha}(k))^3 \\
&\quad + 0.02178 u_\alpha(k) \\
x_6(k+1) &= w_{61}(k)\, S(v(k))^2 + w_{62}(k)\, S(\lambda_{r\alpha}(k))^2 \\
&\quad + w_{63}(k)\, S(\lambda_{r\beta}(k))^2 \\
&\quad + w_{64}(k)\, S(i_{s\beta}(k))^3 + 0.02178 u_\beta(k)
\end{aligned} \tag{11}$$

where $S(x(k)) = \alpha \tan h(\beta x(k)) + \gamma$, $\hat{x}_1(k)$ to identify $q_m(k)$, $\hat{x}_2(k)$ to identify $v(k)$, $\hat{x}_3(k)$ to identify $\psi_\alpha(k)$, $\hat{x}_4(k)$ to identify $\psi_\beta(k)$, $\hat{x}_5(k)$ to identify $i_\alpha(k)$ and $\hat{x}_6(k)$ to identify $i_\beta(k)$. For this application only the fluxes are considered unmeasurable. The training is performed offline, using a series-parallel configuration as shown in Fig. 1. Both the NN and LIM states are initialized randomly. The associated covariances matrices are computed using

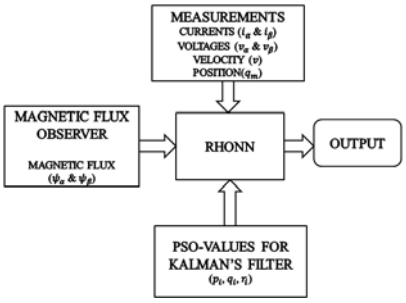

Fig. 1. LIM Identification scheme

the PSO algorithm and the RHONN weights are updated with the EKF as in (3). The input signals u_α and u_β are selected as chirp functions.

B.1 Reduced order nonlinear observer

The last control algorithm requires the full state measurement assumption [3]. However, rotor fluxes measurement is a difficult task. Here, a reduced order nonlinear observer is designed for fluxes on the basis of rotor speed and currents measurements. The flux dynamics in (10). Therefore, the following observer is used [11]:

$$\begin{aligned}
\widetilde{\Lambda}(k+1) &= \widetilde{\Lambda}(k) - k_6 T \widetilde{\Lambda}(k) - k_4 T \Theta^T J I_s(k) \\
&\quad + k_4 T \Theta^T J I_s(k) v(k) \\
&\quad + k_4 T \Theta^T I_s(k)
\end{aligned} \tag{12}$$

where

$$\begin{aligned}
\Lambda(k) &= \begin{bmatrix} \lambda_{r\alpha}(k) \\ \lambda_{r\beta}(k) \end{bmatrix} \\
I_s(k) &= \begin{bmatrix} i_{s\alpha}(k) \\ i_{s\beta}(k) \end{bmatrix} \\
\Theta(k) &= \begin{bmatrix} \cos(n_p q_m(k)) & -\sin(n_p q_m(k)) \\ \sin(n_p q_m(k)) & \cos(n_p q_m(k)) \end{bmatrix} \\
J &= \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}
\end{aligned}$$

The stability proof for (12) is presented in [11].

C. Experimental Results

The proposed scheme is depicted in Fig. 1. The experiments are performed using a benchmark whose schematic representation is depicted in Fig. 2. Fig. 3 shows the experimental benchmark for the LIM

The methodology used to implement experimental identifier is as follows: 1) To validate and to test this algorithm via simulation in Matlab/Simulink, using a plant model and their respective parameters; 2) To download the validated identifier to the DS1104 board; 3) To replace the simulated model state variable values by the induction motor measurements (current and angular position, acquired through the DS1104 board

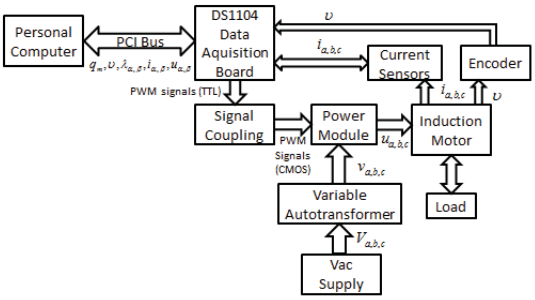

Fig. 2. Schematic representation of the prototype to be identified.

Fig. 3. Linear induction motor prototype

A/D ports, and calculated (fluxes) state variables values; 4) To send back through the DS1104 board, the input signals (voltages) defined as chirp signals 5) To process the input signals through the Space Vector Pulse Width Modulation (SVPWM) power stage and 6) To apply the SVPWM output to the induction motor.

The experimental results are presented as follows: Fig. 4 and Fig. 5 display the identification performance for the position and linear velocity, respectively; Fig. 6 and Fig. 7 present the identification performance for the fluxes in phase α and β, respectively. Figs 8 and 9 portray the identification performance for currents in phase α and β, respectively. Finally Fig. 10 shows the identification errors.

V. Conclusions

This paper has presented the application of recurrent high order neural networks to the identification of discrete-time nonlinear systems. The training of the neural networks was performed on-line using an ex-

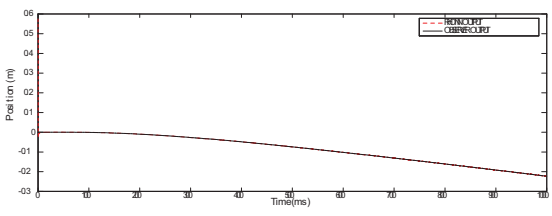

Fig. 4. Position identification

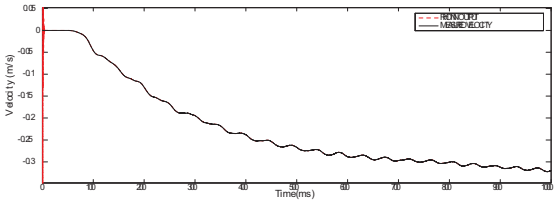

Fig. 5. Linear velocity identification

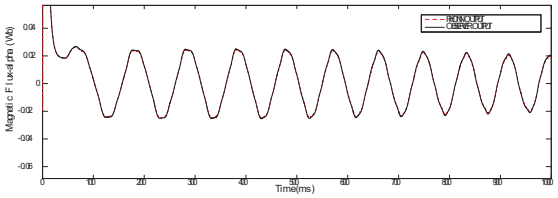

Fig. 6. Alpha flux identification

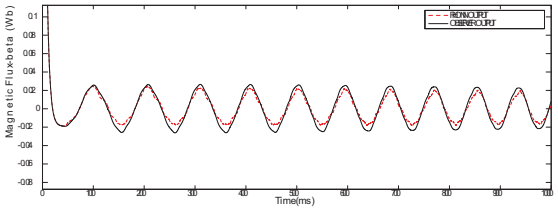

Fig. 7. Beta flux identification

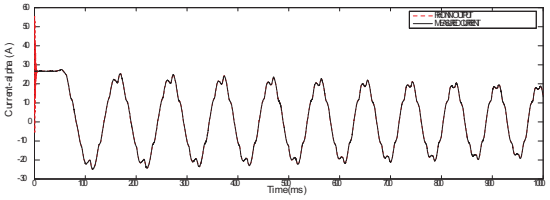

Fig. 8. Alpha current identification

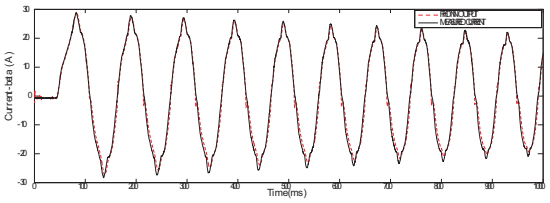

Fig. 9. Beta current identification

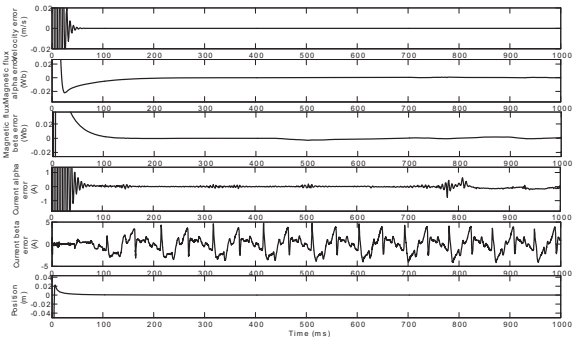

Fig. 10. Identification errors

tended Kalman filter improved with a PSO algorithm. Experimental results illustrate the applicability of the proposed identification methodology for the off-line identification of a three phase induction motor. Researches are being pursued to develop the on-line EKF-PSO neural identification algorithm.

Acknowledgments

The authors thank the support of SEP Mexico, through Project PROMEP/103.5/11/6290 and CONACYT Mexico, through Project 103191Y.

References

[1] A. Y. Alanis, E. N. Sanchez and A. G. Loukianov, "Discrete time adaptive backstepping nonlinear control via high order neural networks", IEEE Transactions on Neural Networks, vol. 18, no. 4, pp. 1185-1195, 2007.

[2] B. Al-kazemi, C. K. Mohan, "Multi-phase generalization of the particle swarm optimization algorithm", *Proceedings of the 2002 Congress on Evolutionary Computation*, vol. 1, pp. 489-494, 2002.

[3] V. H. Benitez Block Neural Control: Application to a Linear Induction Motor, in Spanish, Master Thesis, Cinvestav, Unidad Guadalajara, Mexico, 2002.

[4] I. Boldea. and S. A. Nasar, Linear Electric Actuators and Generators. Cambridge University Press., Cambridge, England, 1997.

[5] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization", *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1951-1957, 1999.

[6] J. A. Farrell and M. M. Polycarpou, Adaptive approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches, John Wiley and Sons, New York, USA, 2006.

[7] S. S. Ge, J. Zhang T. H. and Lee, "Adaptive neural network control for a class of MIMO nonlinear systems with disturbances in discrete-time", IEEE Transactions on Systems, Man and Cybernetics, Part B, vol. 34, no. 4, 2004.

[8] J. F. Gieras. Linear Inductions Drives. Oxford University Press., Oxford, England, 1994.

[9] R. Grover and P. Y. C. Hwang, Introduction to Random Signals and Applied Kalman Filtering, Second edition, John Wiley and Sons, New York, USA, 1992.

[10] S. Haykin, Kalman Filtering and Neural Networks, John Wiley and Sons, New York, USA, 2001.

[11] M. Hernandez, E. N. Sanchez, A. G. loukianov, "Discrete-time Neural Network Control for a linear Induction Motor", Proceedings of the IEEE Multi-Conference on Systems and Control, San Antonio, Texas, September 2008.

[12] N. Kazantzis, C. Kravaris, "Time-discretization of nonlinear control systems via Taylor methods", Computer and Chemical Engineering, vol. 23, pp 763-784, 1999.

[13] J. Kennedy, R.C. Eberhart, "Particle swarm optimization", *Proceedings of the IEEE International Joint Conference on Neural Networks*, pp. 1942-1948, 1995.

[14] R. Kiran, S. R. Jetti and G. K. Venayagamoorthy "Online Training of a Generalized Neuron with Particle Swarm Optimization", *Proceedings of the 2006 International Joint Conference on Neural Networks,* Vancouver, BC, Canada, July, 2006.

[15] A. G. Loukianov, J. Rivera and J. M. Cañedo, "Discrete time sliding mode control of an induction motor", Proceedings IFAC'02, Barcelone, Spain, July, 2002.

[16] A. S. Poznyak, E. N. Sanchez and W. Yu, Differential Neural Networks for Robust Nonlinear Control, World Scientific, Singapore, 2001.

[17] L. J. Ricalde and E. N. Sanchez, "Inverse optimal nonlinear high order recurrent neural observer", International Joint Conference on Neural Networks IJCNN 05, Montreal, Canada, 2005.

[18] G. A. Rovithakis and M. A. Chistodoulou, Adaptive Control with Recurrent High -Order Neural Networks, Springer Verlag, New York, USA, 2000.

[19] Y. Shi, R.C. Eberhart, "A modified particle swarm optimizer", *Proceedings of the IEEE Int. Conf. Evol. Comput*, pp. 69-73, 1998.

[20] I. Takahashi and Y. Ide, "Decoupling control of thrust and attractive force of a LIM using a space vector control inverter", IEEE Trans. Ind. Applicat., vol. 29, pp 161-167, Jan/Feb. 1993.

[21] R. L. Welch, S. M. Ruf ng and G. K. Venayagamoorthy, "Comparison of Feedforward and Feedback Neural Network Architectures for Short Term Wind Speed Prediction", *Proceedings of the IEEE International Joint Conference on Neural Networks*, Atlanta, Georgia, USA, pp. 3335-3340, June 2009.

[22] W. Yu and X. Li, "Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms", Information Sciences, no. 158, pp. 131-147, 2004.

[23] S.-H. Zahiri, S.-A. Seyedin, , *Journal of the Franklin Institute,* vol. 344, pp. 362-376, 2007.