A Robust PMSM Speed Control using an Artificial and a Recurrent Neural Network

Flah Aymen, Ben Hamed Mouna, Farhat Mayssa, and Sbita lassâd

National School of Engineering of Gabes, University of Gabes, Tunisia

Email: flahaymening@yahoo.fr

Abstract – In this paper, the robust PMSM control theme is achieved using the field oriented vector control strategy. The neural network technique is used to ameliorate the global PMSM control function, where the recurrent and the artificial neural network are applied and compared as a speed controller in the field oriented vector control strategy. The effectiveness of these controllers was discussed and demonstrated in MATLAB simulation results for a variable speed with a load application. To study the two neural controllers, the same run conditions, the same architecture and database learning are used for the two neural models, where especially, three layers and the back-propagation learning algorithm are applied.

Keywords – PMSM, Recurrent and artificial neural network, Speed control, Field oriented vector control

1. Introduction

The permanent magnet synchronous motor (PMSM) becomes most used in many applications like robotics, aerospace actuators, industrial applications and especially in the transport section, where most of the electric vehicle based on the PMSM, for her larger torque inertia ratio and power density compared with the induction motors for the same output capacity. Well for the same performances of the induction motors, the PMSM is smaller and lower in weight that makes it preferable for high performance applications [1].

The PMSM speed control is the target needed in the most applications, where many strategies like field oriented vector control appeared to assume the motor drive. Most authors applied this technique that base especially on three controllers, one for the speed control loop and two for the current, however the researches is not stopped to ameliorate the global performance and thanks to the development in automation technology [2], high performance electrical servos are most used.

To deal with this high performance servo requirements, it has become a necessity to develop an intelligent controllers to overcome the influence of nonlinear plant behavior motor’s parameters tune varying, the motor in a wide speeds range, and to keep on high performances of the overall system unchanged [3], [4], [5].

With the necessity for an adaptive controller, neural network can be applied to control and identify the nonlinear systems since they approximate any desired degree of accuracy with a wide range of nonlinear model [6], [7], [8].

The neural network appeared firstly in 1950 and her learning possibility, was used in many applications like identification, classifier and system control. Many configurations are presented, like the artificial and the recurrent neural network.

Each one of them is characterized by a specially interconnection between the neurons in the network, where in the artificial model only a connection between the neurons in the input and hidden layer and between the hidden and the output layer, however in the recurrent one the neurons can change information between the others neurons in the other layers, they even and the neurons in the same layer, however with this additional connections the recurrent neural network prove its effectiveness in many applications especially in the control problems[9], for her precession learning and especially for her dynamic mapping, where it can capture the dynamic response of a system without delay caused by a external feedback because the recurrent neural has an internal feedback loop[9].

In this paper the neural network is used in the field oriented vector control loop as a robust speed drive and a comparative study between two neural speed controllers, with the artificial and recurrent neural network, is applied.

However with the Artificial Neural Network Speed Controller (ANSC) or the Recurrent Neural Network Speed Controller (RNSC), the unknown reference quadrature stator current can be generated to control the non linear plant dynamics by presenting a suitable set of input and output patterns generated by the system, but in the field oriented vector control the reference direct control current must be also given, however an field weakening algorithm was used in order to touch a wide speed but our application the PMSM will be running in the rated speed range so the reference direct stator current is equal to zero.

This paper is organized as follows. The PMSM model is described in section II. Section III presents the Artificial Neural Network Speed Controller. Section IV describes the Recurrent Neural Network Speed Controller. In last section we compare the speed results using the two types of neural controller.

2. The PMSM model and field oriented vector control strategy

The dynamic properties of the PMSM can be described by a set of nonlinear differential equations linking the stator and rotor currents and voltages with the mechanical quantities torque, speed, and angular position [1], [3].

The voltage expressions:

\[ v_d = R_i_d + L_d \frac{di_d}{dt} - \omega L_q i_q \]

\[ v_q = R_i_q + L_q \frac{di_q}{dt} + \omega L_d i_d + \frac{d\omega}{dt} \]

(1)
The torque expression can be written like presented in (2) where the first and the second expression are described respectively for the saliency and the non saliency model:

\[
T_P = \frac{3}{2} \left( \phi_L i_d + (L_d - L_q) i_d i_q \right)
\]

\[
T_{eq} = \frac{3}{2} \left( \phi_L i_d \right)
\]

Hence the electric torque depends only on the quadrature axis current in the non saliency model.

The flux expression is presented in (3).

\[
\phi_d = L_d i_d + \phi
\]

\[
\phi_q = L_q i_q
\]

The equation for the motor dynamics is:

\[
(T_e - T_i) = \frac{3}{2} \left( J \frac{d\omega}{dt} + f \omega \right)
\]

The field oriented vector control was appeared as an induction motor control strategy, needed especially three hall current sensor, a modulation method like PWM or SVPWM modulator and a power stage, Inverter. The rotor angular velocity is obtained using a speed encoder or observer.

Effectively the speed value response will be compared with the desired speed value, then through a speed regulator a reference quadrature current will be generated, this one will be compared with the real quadrature current after a Park transformation to generate the quadrature reference voltage and with the direct reference voltage the modulator generated the switching command for the inverter [10].

3. The Artificial Neural Network Speed Controller-ANSC

The artificial neural network was used as a PMSM speed controller to provide a real time quadrature current signal. Effectively the artificial neural speed controller ANSC compose three layers, an input, hidden and an output layer, each neuron in the hidden layer is connected to the other in the input and the output layers [11].

This connections is characterized respectively by a weights, \( w_l \) and \( w_o \), like presented if figure (1).

![Diagram](image)

Figure 1. Structure of an artificial neural network.

The mathematical expressions define the relation between the input and the output signals are given in (5), (6) and (7).

- The output layer:

\[
i_{qa}(k) = f\left(\sum_{j=1}^{nc} w_{ij}^o(k)y_j(k)\right)
\]

- The output hidden layer:

\[
y_i(k) = f\left(S_j(k)\right) = \frac{1}{1 + \exp(-S_j(k))}
\]

- The input hidden layer:

\[
S_j(k) = \sum_{i=1}^{ne} w_{ji}^l(k)x_j(k)
\]

Where \( nc, ne \) are respectively the number of neurons in the hidden and the output layer.

The data vector \( x_l \) represent the input ANSC signals is given in (8).

\[
x_l = [i_q^*(k-1) \ \epsilon_y(k) \ \epsilon_y(k-1)]^T
\]

An activation function defines the each neuron in the ANSC and the sigmoid function is used in this application.

To achieve the desired input/output relationship, adjusting weights between the output layer and the hidden layer, and the input layer and the hidden layer is required and the back propagation algorithm is used for updating this weights, by minimized an error function defined in (9).

\[
E_{na}(k) = \frac{1}{2} \left[ i_q^*(k) - i_{qa}(k) \right]^2 = \frac{1}{2} e_{na}^2(k)
\]

Where \( i_q^*(k) \) is the desired target, \( i_{qa}(k) \) is the output signal from the ANSC and \( e_{na}(k) \) is the error between them.

The weights updating is calculated using the gradient method, the expressions (10) and (11) define the formula.

\[
\frac{\partial \varepsilon_{na}(k)}{\partial w_{j,i}^l(K)} = \left( i_q^*(k) - i_{qa}(k) \right) y_j(k)
\]

\[
\frac{\partial \varepsilon_{na}(k)}{\partial w_{j,i}^o(K)} = -e_{na}(k) w_{i,j}^o(k)x_j(k)
\]
The Recurrent Neural Network Speed Controller - RNNSC

The recurrent neural network is an extended version of the artificial neural network, characterized by the connections existed between the neurons in the hidden layer, effectively the output of these neurons will be transmitted to the output neurons and themselves in the same layer, like presented in figure (2).

![Figure 2. Structure of the recurrent neural network.](image)

The RNNSC compose three layered network structure, input, hidden and output layer. The mathematical operations describe the general relations [9] are:

1. The equation (14) expresses the input signal for the hidden layer, the equation (15) illustrates the output signal of the hidden layer and equation (16) is the output signal from the output layer that represents the RNNSC output.

2. In equation (14), \( I_i \) is the input vector that regroups the general information must be given to the RNNSC to generate the desired output, in this application this vector is illustrated in (17).

3. Four weights characterize the RNNSC, \( v_D \), \( v_L \), \( v_I \) and \( v_O \) represent respectively the weight between the same neuron in the hidden layer, between two successive neurons in the hidden layer, the weight from the input layer to the hidden layer and the weight from the hidden layer to the output layer. Each neuron in the hidden and the output layer is characterized by an activation function \( f(\cdot) \) that used as a sigmoid type.

4. Like presented in the introduction section, each neuron blocs oblige a learning phase, that equivalent to a minimization of the error between the desired and the neuron output. In this application this error function is appeared in (18) and the minimization algorithm, is the back-propagation algorithm where its steps figured in the figure (3).

\[
E(k) = \frac{1}{2} \left[ I_i^* (k) - i_{nr}^n (k) \right]^2 = \frac{1}{2} e_{nr}^2 (k)
\]  

On running this algorithm all weights characterized the RNNSC will be updated in each step learning and the weights update equations are in (19) to (22).

\[
v_j^o (k+1) = v_j^o (k) + \Delta v_j^o (K) = v_j^o (k) - \eta E_j^o (K) \frac{\partial E_j^c (K)}{\partial v_j^o (K)}
\]  

\[
v_{j,i}^l (k+1) = v_{j,i}^l (k) + \Delta v_{j,i}^l (K) = v_{j,i}^l (k) - \eta E_j^l (K) \frac{\partial E_j^c (K)}{\partial v_{j,i}^l (K)}
\]  

\[
v_{j,j-1}^l (k+1) = v_{j,j-1}^l (k) + \Delta v_{j,j-1}^l (K) = v_{j,j-1}^l (k) - \eta E_j^l (K) \frac{\partial E_j^c (K)}{\partial v_{j,j-1}^l (K)}
\]  

\[
v_{j,j}^d (k+1) = v_{j,j}^d (k) + \Delta v_{j,j}^d (K) = v_{j,j}^d (k) - \eta E_j^d (K) \frac{\partial E_j^c (K)}{\partial v_{j,j}^d (K)}
\]  

The gradient expressions are:

\[
\frac{\partial E_j^c (K)}{\partial v_j^o (K)} = -e_{nr} (k) Z_j (k)
\]  

\[
\frac{\partial E_j^c (K)}{\partial v_{j,i}^l (K)} = -e_{nr} (k) v_j^o (k) G_{j,i} (k)
\]  

\[
\frac{\partial E_j^c (K)}{\partial v_{j,j}^d (K)} = -e_{nr} (k) v_j^o (k) F_{j,j} (k)
\]  

And

\[
F_{j,j} (k) = Z_j (k) \left[ 1 - Z_j (k) \right]
\]  

\[
G_{j,i} (k) = Z_j (k) \left[ 1 - Z_j (k) \right] \times \left[ I_i (k) + v_{j,j}^d (K) G_{j,i} (k) (k - 1) \right]
\]  

\[
H_{j,j-1} (k) = Z_j (k) \left[ 1 - Z_j (k) \right] \times Z_j (k) \left[ 1 - Z_j (k) \right] \times \left[ Z_{j,j-1} (k) \right] \times \left[ 1 - Z_j (k) \right] \times \left[ 1 - Z_j (k) \right]
\]
\( \eta_{nr}^{i}, \eta_{nr}^{D}, \eta_{nr}^{L}, \eta_{nr}^{O} \) are the learning rates characterize the RNSC.

5. Simulation Results

In this part we show and discuss the simulations results illustrated in the Matlab Simulink environment.

The non salient permanent magnet synchronous motor PMSM parameters of used in this work are: the direct and the quadrature inductance \( L_d = L_q = 0.0115 \text{mH} \), the stator resistance \( R_s = 3.5 \Omega \), the permanent magnet flux \( \Phi = 0.178 \text{Wb} \), the inertia coefficient \( J = 0.000139 \), the pole number \( P = 6 \).

As described previously, the neuron technique is used as a speed controller in the field oriented vector control strategy to remote the PMSM speed, however a comparison between two neural speed controllers is achieved in this work and the same database, architecture, training patterns, input vector are given for the two models.

The neural architecture, for the two models is:
- 3 neurons in the input layer,
- 3 neurons in the hidden layer
- 1 neuron in the output layer
- All the learning rates in the two model is 0.1
- The training pattern is 100 epoch.

To beginning the learning phase a database regroup an information on the motor must be given to the two neural controllers, however this information given are obtained for an variable speed like presented in figure (5.a) and regroup current, figure (5.b) and error information.

Figure (6) show the windows given by the Matlab environment at starting the learning step, where the structure, the neural model performance, the learning function, the maximum training patterns, the neural architecture and the error learning rates for the two neural models are given and it's clear that the two neural controller's performance is accepted,
however with the RNSC the learning time is bigger than by the ANSC, that present an recurrent neural disadvantage.

The learning strategy is given in figure (3), effectively after evaluating the input data each neural controller start the learning phase by executing the mathematical expressions define the RNSC end the ANSC and figured in figure(3).

The global control scheme presented in figure (), illustrate the PMSM control loop, where the three reference signals are generated from the inverter. This one need six reference signals obtained from a PWM bloc. However a neural speed controller generate a reference quadrature current and compared with the real one to deliver the input error for the PI current to generate the reference quadrature voltage, the direct one is obtained from another PI current needed the error between the reference and the real direct current. A field weakening bloc used to generate this direct stator current.

The results illustrated in the figure (7.a) show the behavior of speed for the two case, with RNSC and with ANSC, however a variable reference speed is given started from 0 rpm to 1000 rpm then decrease to 500 rpm at t=1s, then return to 1000 rpm at t=2s. To verify the global system stability a load torque was applied at t= 0.5 s.
The figure (7.c) and (7.b) show respectively a speed zoom at t=0.5 the time when a load torque is applied and at t=0s, where it’s clear the effectiveness of the neural network in the speed control loop, however with the ANSC some perturbation in the translation from an target to another and when a load torque applied appeared, unlike with the RNSC this perturbation is eliminated, that demonstrate the recurrent neural network superiority.

6. Conclusion

In this paper a comparative study between the recurrent and the artificial neural network as a PMSM speed controller in the field oriented vector control was applied. The same neural configuration, the same database and the same training patterns are used for the two models, effectively the neural speed controller generate a reference quadrature stator current then used in the field oriented vector control strategy to control the motor. A simulation study based on MATLAB Simulink is used and the results figured confirm system robustness with the two neural models and the superiority of the recurrent neural network as a speed controller in the field oriented vector control strategy.

Appendix

\[ R_s \]: stator resistance,
\[ L_d, L_q \]: Direct and quadrature stator inductance,
\[ v_d, v_q \]: Direct and quadrature stator voltage,
\[ i_d, i_q \]: Direct and quadrature stator current,
\[ \omega \]: Real speed output
\[ n_p \]: Pôles number,
\[ \phi_d, \phi_q \]: Direct and quadrature stator flux,
\[ e_{na}, e_{nr} \]: learning error in the artificial and the recurrent neural network
\[ i_d^a, i_q^a \]: The artificial and the recurrent neural output
\[ \eta_a, \eta_r \]: the RNSC and ANSC learning rates

References