Laboratory Implementation of an Artificial Neural Network for Online Tuning of a Genetic Algorithm Based PI Controller for IPMSM Drive

M. Nasir Uddin[+], *Member IEEE*, Casey Butt*, M. A. Abido[#] *Member IEEE*, M. A. Rahman*, *Fellow, IEEE*

+ Modeling, Automation and Control Engineering Lab.
Dept. of Electrical Engineering
Lakehead University
Thunder Bay, Ontario P7B 5E1, Canada

*Faculty of Engineering & Applied Science
Memorial University of Newfoundland
St. John's, Newfoundland A1B 3X5
Canada

Electrical Engineering Department
King Fahd University of Petroleum and Minerals
Dhahran, 31261, Saudi Arabia.

Abstract

This paper presents an artificial neural network (ANN) for online tuning of a genetic based proportional-integral (PI) controller for interior permanent magnet synchronous motor (IPMSM) drive. The proposed controller is developed for accurate speed control of the IPMSM drive under system disturbances. In this work, initially different operating conditions are obtained based on motor dynamics incorporating various uncertainties. At each operating condition genetic algorithm (GA) is used to optimize PI controller parameters in a closed loop vector control scheme. In the optimization procedure a performance index is developed to reflect the minimum speed deviation, minimum settling time and zero steady-state error. A radial basis function network (RBFN) is utilized for online tuning of the PI controller parameters to ensure optimum drive performance under different disturbances. The proposed controller is successfully implemented in laboratory using a digital signal processor board DS1102 for a 1 hp IPMSM. The efficacy of the proposed controller is verified by simulation as well as experimental results at different dynamic operating conditions.

Keywords: Interior permanent magnet motor, artificial neural network, genetic algorithm, PI controller, digital signal processor and vector control.

1. Introduction

Recent developments in microprocessors, high speed power electronic devices, magnetic materials and control algorithms have enabled modern ac motor drives to face challenging high efficiency and high performance requirements in the industrial sector. Among ac drives, the interior permanent magnet synchronous motor (IPMSM) has been becoming popular owing to its high torque to current ratio, large power to weight ratio, high efficiency, high power factor and robustness [1]. These features are due to the incorporation of high energy rare-earth alloys such as Neodymium-Iron-Boron in its construction. Especially, the interior permanent magnet synchronous motor (IPMSM) which has magnets buried in the rotor core exhibit certain good properties, such as, mechanically robust rotor construction, a rotor physically non-saliency and small effective air gap. The rotors of these machines have a complex geometry to ensure optimal use of the expensive permanent magnet material while maintaining a high magnetic field in the air-gap.

Fast and accurate speed response, quick recovery of speed from any disturbances and insensitivity to parameter variations are some of the important criteria of the high performance drive systems used in robotics, rolling mills, machine tools etc. In order to achieve high performance, the vector control of IPMSM drive is employed [2]. However, the controller design of such system plays crucial role in the system performance. The decoupling characteristics of vector controlled IPMSM are adversely affected by the parameter changes in the motor. Traditionally, control issues are handled by the conventional PI controller and various adaptive controllers such as model reference adaptive controller, sliding mode controller, variable structure controller have been widely utilized as speed controllers in the IPMSM drive. However, the difficulties of obtaining the exact d-q axis reactance parameters of the IPMSM leads to cumbersome design approach for these controllers. Moreover, the conventional fixed gain PI controller is very sensitive to step change of command speed, parameter variations and load disturbance [3]. Again, precise speed control of an IPMSM drive becomes a complex issue due to nonlinear coupling among its winding currents and the rotor speed as well as the nonlinearity present in the electromagnetic developed torque due to magnetic saturation of the rotor core [4]. Therefore, there exists a need to tune the PI controller parameters online to ensure optimum drive performance over a wide range of operating conditions. In the present work, as an artificial neural network (ANN) a radial basis function network (RBFN) is utilized for this purpose. The ability of ANN to handle nonlinear system uncertainties such as step change in command speed, load impact, saturation and parameter variations is well-known [5]. Over the last decade, ANN are used in modeling and control techniques for many real industrial processes [3,6-11]. Most of these works are dealt with dc motor [3,6-8]. Recently, some works are reported on ANN based permanent magnet synchronous motor drive [9-11]. However, the simulation and experimental results published in these works [9-11] are not very much satisfactory in terms of disturbance rejection such insensitive to load variations, parameter variations etc. In this work, GA is used to optimize the parameters of ANN structure, which added a new feature to get a predictive and better performance.

This paper presents the detailed laboratory implementation and performance investigation of a novel speed control scheme using an ANN for online tuning of a genetic based PI controller for IPMSM drive. In developing the proposed controller, the PI controller parameters are optimized by GA at all possible operating conditions in a closed loop vector control scheme. In the optimization procedure a performance index is developed to reflect the minimum speed deviation, minimum settling time and zero steady-state error. Then a radial basis function network (RBFN) is utilized for online tuning of the PI controller parameters to ensure optimum drive performance under different disturbances and operating conditions. The laboratory implementation of the proposed controller is done using a digital signal processor (DSP) board DS 1102 for a 1 hp IPMSM. The efficacy of the proposed controller is verified by simulation as well as experimental results at different dynamic operating conditions.

2. Motor Dynamics

The mathematical model of an IPMSM drive can be described by the following equations in a synchronously rotating rotor d-q reference frame as,

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} R + pL_d & -P\omega_r L_q \\ P\omega_r L_d & R + pL_q \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} 0 \\ P\omega_r \psi_f \end{bmatrix} \tag{1}$$

$$T_e = T_L + J_m p\omega_r + B_m \omega_r \tag{2}$$

$$T_e = \frac{3P}{2}\left(\psi_f i_q + (L_d - L_q) i_d i_q\right) \tag{3}$$

where, v_d, v_q = d- and q-axis stator voltages;

i_d, i_q = d- and q-axis stator currents;

R = stator per phase resistance;

L_d, L_q = d- and q-axis stator inductances;

T_e, T_L = electromagnetic and load torques;

J_m = moment of inertia of the motor and load;

B_m = friction coefficient of the motor;

P = number of poles of the motor;

w_r = rotor speed in angular frequency;

p = differential operator (=d/dt);

y_f = rotor magnetic flux linking the stator.

3. Control Principle

According to the motor model given in equations (1-3), it can be seen that the speed control can be achieved by controlling the q-axis component v_q of the supply voltage as long as the d-axis current i_d is maintained at zero. This results in the electromagnetic torque being directly proportional to the current i_q. Since $i_d = 0$, the d-axis flux linkage depends only on the rotor permanent magnets. The resultant IPMSM model can be represented as,

$$pi_q = \frac{1}{L_q}\left(v_q - Ri_q - Pw_r y_f\right) \tag{4}$$

$$v_d = -w_r L_q i_q \tag{5}$$

$$T_e = T_L + J_m p w_r + B_m w_r \tag{6}$$

$$T_e = \frac{3P}{2}\left(y_f i_q\right) \tag{7}$$

In the proposed approach, various operating conditions are generated randomly taking into account different drive uncertainties based on motor dynamics given by Eqs.(4)-(7) and the machine parameters given in Table-I..

Table-I: Machine parameters

Motor rated power	3-phase, 1 hp
Rated voltage	208 V
Rated current	3 A
Rated frequency	60 Hz
Pole pair number (P)	2
d-axis inductance, L_d	42.44 mH
q-axis inductance, L_q	79.57 mH
Stator resistance, R	1.93 Ω
Motor inertia, J_m	0.003 kgm^2
Friction coefficient, B_m	0.001 Nm/rad/sec
Magnetic flux constant, y_f	0.311 volts/rad/sec

At each operating condition genetic algorithm (GA) is used to optimize PI controller parameters in a closed loop vector control scheme. In optimization procedure a performance index is developed to reflect the minimum speed deviation, minimum settling time and zero steady-state error. A radial basis function network (RBFN) is utilized for online tuning of the PI controller parameters to ensure optimum drive performance under different disturbances. In the following sections, the GA and RBFN are briefly described.

4. Genetic Algorithm

Genetic algorithms are exploratory search and optimization procedures that were devised on the principles of natural evolution and population genetics [12]. Unlike other optimization techniques, GA work with a population of individuals represented by bit strings and modify the population with random search and competition. The advantages of GA over other traditional optimization techniques can be summarized as follows:

- GA search the problem space using a population of trials representing possible solutions to the problem, not a single point, i.e. GA have implicit parallelism. This property ensures GA to be less susceptible to getting trapped on local minima.
- GA use a performance index assessment to guide the search in the problem space.
- GA use probabilistic rules to make decisions.

In general, GA include operations such as reproduction, crossover, and mutation. Reproduction is a process in which a new generation of population is formed by selecting the fittest individuals in the current population. Crossover is the most dominant operator in GA. It is responsible for producing new offsprings by selecting two strings and exchanging portions of their structures. The new offsprings may replace the weaker individuals in the population. Mutation is a local operator, which is applied with a very low probability. Its function is to alter the value of a random position in a string.

A. Real-Coded Genetic Algorithm (RCGA)

Due to difficulties of binary representation when dealing with continuous search space with large dimension, the proposed approach has been implemented using real-coded genetic algorithm (RCGA). A decision variable x_i is represented by a real number within its lower limit a_i and upper limit b_i, i.e. $x_i \in [a_i, b_i]$.
The RCGA crossover and mutation operators are described as follows:

A.1. Crossover

A blend crossover operator has been employed in this study. This operator starts by choosing randomly a number from the interval $[x_i - a(y_i - x_i), y_i + a(y_i - x_i)]$, where x_i and y_i are the i^{th} parameter values of the parent solutions and $x_i < y_i$. To ensure the balance between exploitation and exploration of the search space, $a = 0.5$ is selected.

A.2. Mutation

The non-uniform mutation operator has been employed in this study. In this operator, the new value x_i' of the parameter x_i after mutation at generation t is given as

$$x_i' = \begin{cases} x_i + \Delta(t, b_i - x_i) & \text{if } t = 0 \\ x_i - \Delta(t, x_i - a_i) & \text{if } t = 1 \end{cases} \tag{8}$$

and; $$\Delta(t, y) = y(1 - r^{(1-\frac{t}{g_{\max}})^b}) \tag{9}$$

where t is a binary random number, r is a random number $r \in [0,1]$, g_{max} is the maximum number of generations, and b is a positive constant chosen arbitrarily. In this study, $b = 5$ was selected. This operator gives a value $x_i' \in [a_i, b_i]$ such that the probability of returning a value close to x_i increases as the algorithm advances. This makes uniform search in the initial stages where t is small and very locally at the later stages.

For the optimal settings of PI controller parameters, following quadratic performance index J is considered:

$$J = \sum_{k=1}^{L} [kT_s \Delta w(k)]^2 \tag{10}$$

In the above index, the speed deviation $\Delta\omega(k)$ is weighted by the respective time kT_s. The index J is selected because it reflects small settling time, small steady state error, and small overshoots. The tuning parameters are adjusted so as to minimize the index J.

B. The Computational Flow

Applying GA to the problem of the optimal design of PI controller involves repetitively performing the following two basic steps:
1. The objective function value must be calculated for each of the strings in the current population. To do this, the FLC parameters must be decoded from each string in the population and the system is simulated to obtain the objective function value. 2. GA operations are applied to produce the next generation of the strings.
These two steps are repeated from one generation to another until the population has converged. The computational flow of the optimization problem can be shown in Fig. 1.

5. Radial Basis Function Network (RBFN)

Like most feed forward networks, RBFN has three layers, namely, an input layer, a hidden layer, and an output layer [13]. A schematic diagram of the specific RBFN with 2 inputs and 2 outputs is given in Fig.2. The hidden layer nodes are the RBF units. Each node in this layer contains a parameter vector called a center. The node calculates the Euclidean distance between the center and the network input vector, and passes the result through a nonlinear function $\Phi(.)$. The output layer is essentially a set of linear combiners. For a general n-input and m-output RBFN structure, the *i*th

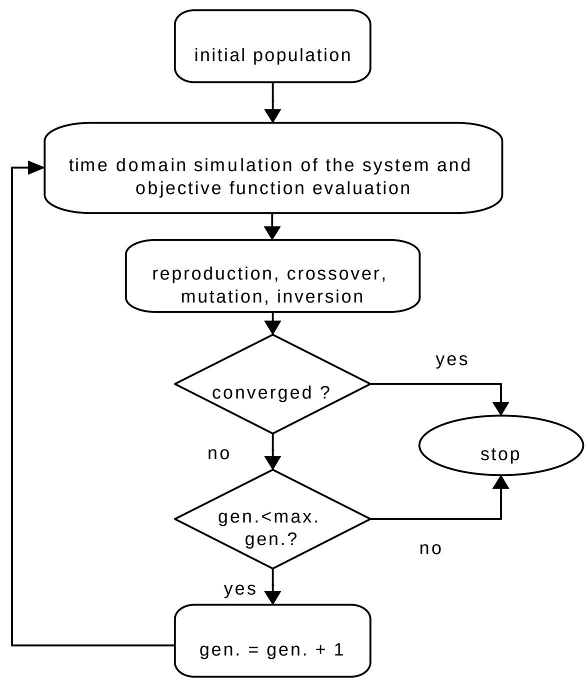


Fig.1. Computational flow chart

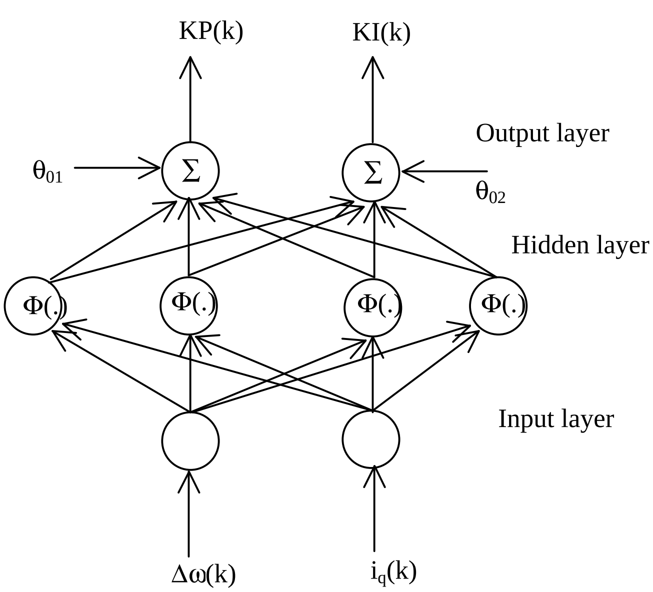


Fig. 2. Schematic diagram of RBFN

output y_i due to input vector $\boldsymbol{x}$, $\boldsymbol{x}=[x_1, \ldots, x_n]^T$, can be expressed as,

$$y_i = \theta_{0i} + \sum_{j=1}^{M_s} \theta_{ji} \Phi\left(\|\boldsymbol{x} - \boldsymbol{c}_j\|, \sigma_j\right) \tag{11}$$

where M_s is the number of hidden units, $\boldsymbol{c}_j$ and σ_j are the center and the width of the jth hidden unit respectively, θ_{ji} represents the weight between the jth hidden unit and the ith output unit, and θ_{0i} is the bias term of the ith output unit. In this study, $\Phi(.)$ is chosen to be gaussian activation function, that is,

$$\Phi(z, \sigma) = \exp(-z^2 / 2\sigma^2) \tag{12}$$

For learning purpose the orthogonal least square (OLS) method is used in this work [13].

6. Laboratory Implementation

The block diagram of the closed loop vector control scheme of IPMSM incorporating the proposed RBFN controller is shown in Fig. 3. The drive is experimentally implemented using digital signal processor (DSP) board DS1102 through both hardware and software [14].

A. Hardware Implementation

The DSP board is installed in a PC with uninterrupted communication capabilities through dual-port memory. The hardware schematic for real-time implementation of the proposed IPMSM drive is shown in Fig.4. The DS1102 board is based on a Texas Instrument (TI) TMS320C31, 32-bit floating point digital signal processor. The DSP has been supplemented by a set of on-board peripherals used in digital control systems, such as A/D, D/A converters and incremental encoder interfaces. The DS 1102 is also equipped with a TI TMS320P14, 16-bit micro controller DSP that acts as a slave processor and is sued for some special purposes. In this work, slave processor is used for digital I/O configuration. . In this work, slave processor is used for digital I/O configuration. The actual motor currents are measured by the Hall-effect sensors which have good frequency response and fed to the DSP board through A/D converter. As the motor neutral is isolated, only two phase currents are fed back and the other phase current is calculated from them. The rotor position is measured by an optical incremental encoder which is mounted at the rotor shaft end. Then it is fed to the DSP board through encoder interface. The encoder generates 4096 pulses per revolution. By using a 4-fold pulse multiplication the number of pulses is increased to 4x4096 in order to get better resolution. A 24-bit position counter is used to count the encoder pulses and is read by a calling function in the software.

The motor speed is calculated from the rotor position by backward difference interpolation. A digital moving average filter is used to remove the noise from the speed signal. The calculated actual motor speed is used to calculate the torque component of the current i_q^* using the FLC algorithm. The command a-b-c phase currents are generated from i_q^* using inverse Park's transformation. In order to implement the vector control algorithm, the hysteresis controller is used as current controllers. The hysteresis

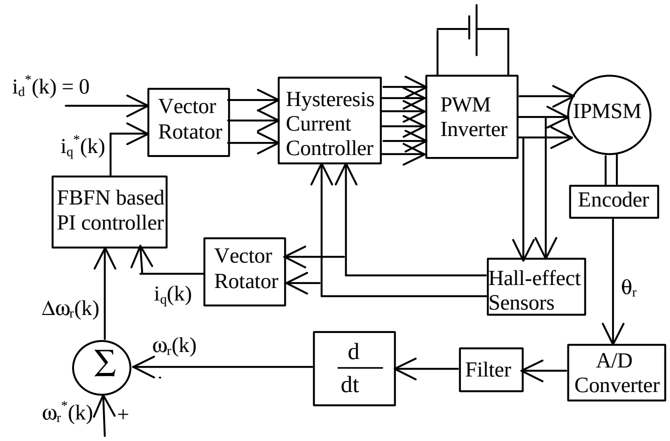


Fig.3. Block diagram of the RBFN based controller for IPMSM drive.

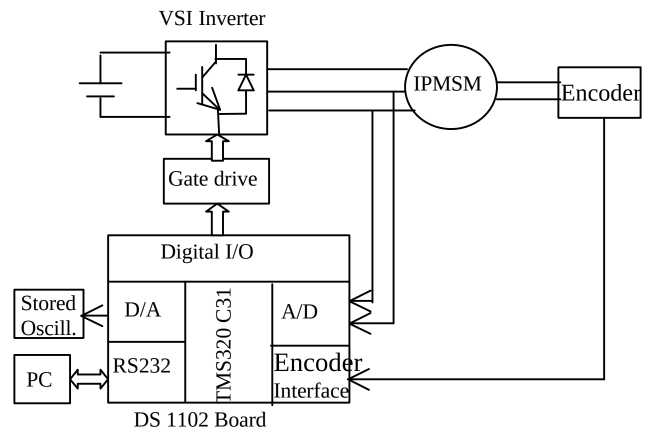


Fig.4. Hardware schematic for real-time implementation.

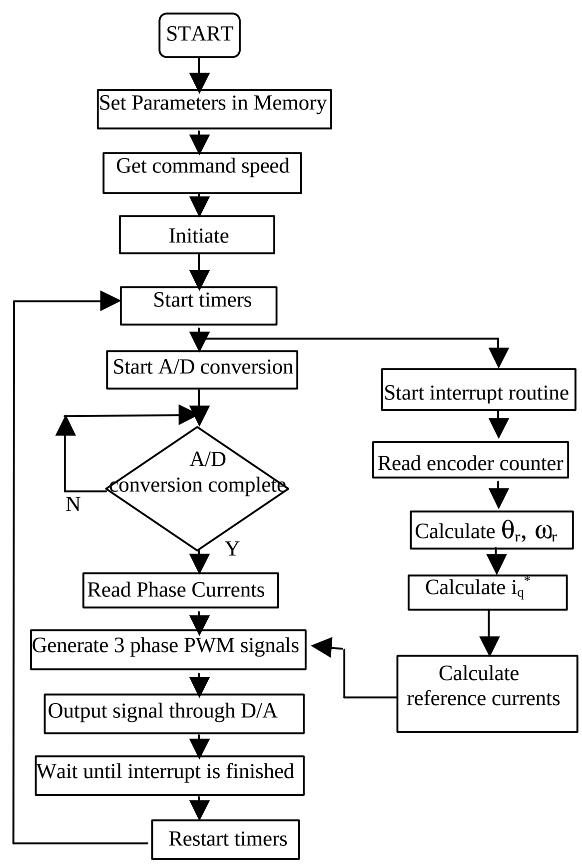


Fig.5. Flow chart of the software for the complete IPMSM drive.

current controller compares the command currents with the corresponding actual motor currents and generates the logic signals, which act as firing pulses for the inverter switches. Thus, these six PWM logic signals are the output of the DSP board and fed to the base drive circuit of the inverter power module. The D/A channels are used to capture the necessary output signals in digital storage oscilloscope.

B. Software Implementation

The complete IPMSM drive is implemented through software by developing a program in high level ANSI ‘C’ programming language. The programming algorithm is summarized by the flow chart shown in Fig. 5. The program is compiled by the TI ‘C’ compiler and then the program is downloaded to the DSP controller board. The sampling frequency for experimental implementation of the proposed IPMSM drive system is 6.67 kHz.

7. Simulation and Experimental Results

Several tests were performed to evaluate the performance of the proposed RBFN based IPMSM drive system both in simulation and experiment. The speed, stator current and torque responses are observed under different operating conditions such as sudden change in command speed, step change in load, parameter variations, etc. Sample results are presented below. The complete drive has been simulated using Matlab/Simulink [15].

The simulated motor speed and current responses are shown in Figs. 6(a)-(c) to see the starting performance as well as the response with a load disturbance of the drive. The drive system is started at a constant load of 1 Nm with the speed reference set at 1800 rpm (188.5 rad/sec). It can be seen from Fig. 5(a) that the actual speed converges to the reference value within 0.1 seconds without any overshoot and undershoot and with zero steady-state error. At t=0.3 seconds, a load torque of 2 Nm is applied to the motor shaft in a stepwise manner. The actual speed does not change during the disturbance while the stator current swiftly reaches to its new value corresponding to the load applied. This shows the capability of new controller to start from standstill condition to the rated speed as well as to reject the disturbance.

The experimental starting performance including speed, torque and stator current i_a are shown in Figs. 7(a)and 7(b), respectively. It is shown that the proposed drive is also capable of following the command speed very quickly with zero steady-state error and without any overshoot or undershoot in a real-time situation. Another experimental speed and the corresponding torque responses are shown in Fig. 8(a) & 8(b), respectively, for a step change in load torque. It is found that the drive is insensitive with load disturbance. Figure 8 shows an experimental speed response for a sudden change in command speed. It is evident that the proposed drive can adapt itself with speed disturbance. Fig.10 shows another speed response with doubled stator resistance. The resistances are inserted externally to the stator. It is shown in this figure that the drive is also insensitive with parameter variations.

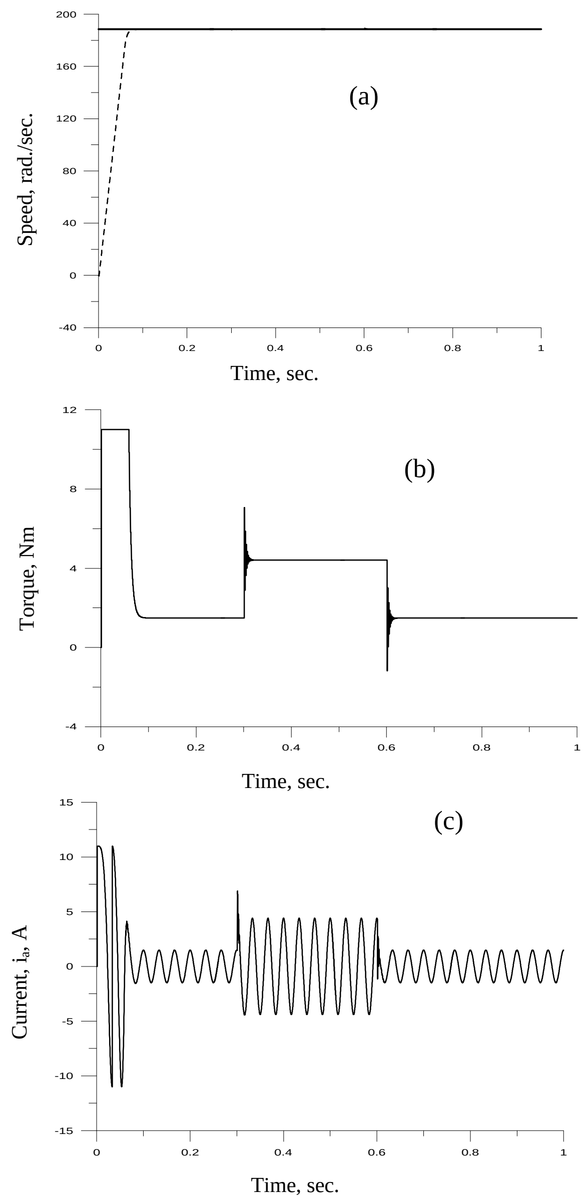
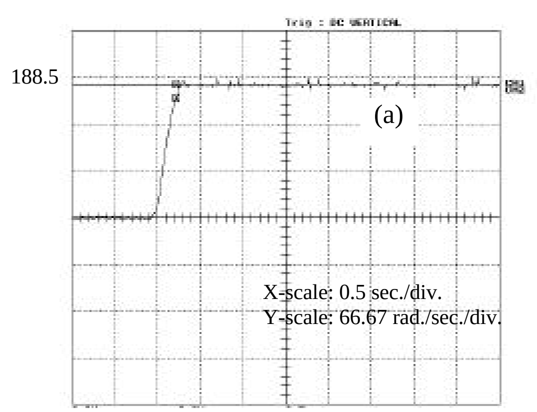


Fig.6. Simulated starting responses of the drive: (a) speed, (b) torque and (c) current, i_a.



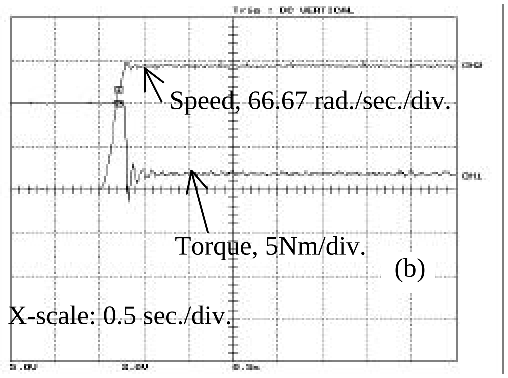


Fig.7. Experimental starting responses of the drive: (a) speed and, (b) torque.

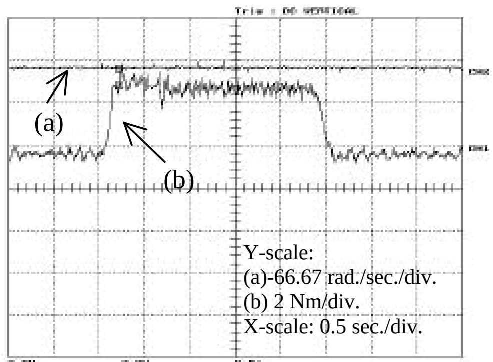


Fig.8. Experimental responses of the drive for a step increase in load: (a) speed, (b) torque.

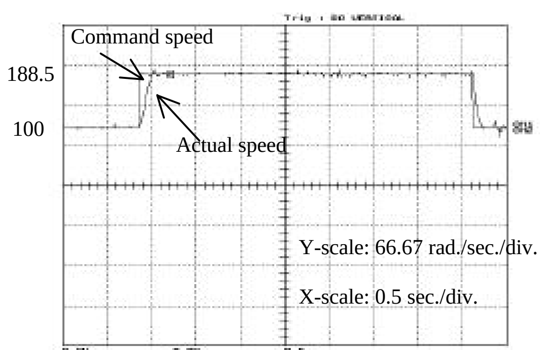


Fig.9. Experimental speed response of the drive for a step change in speed.

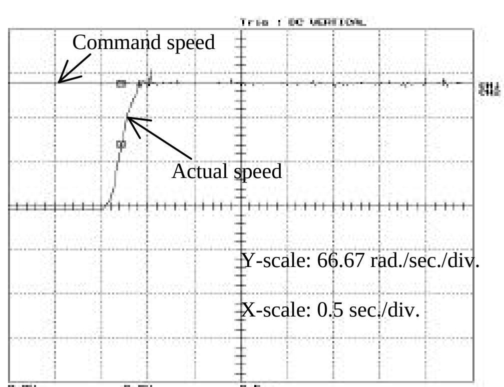


Fig.10. Experimental speed response of the drive with doubled stator resistance.

8. Conclusions

A novel speed control technique for IPMSM using a neural network for on-line tuning of the parameters of a genetic based PI controller has been presented in this paper. The closed loop vector control of IPMSM drive incorporating the proposed RBFN based tuned PI controller has been successfully implemented in real-time for a laboratory 1 hp interior type permanent magnet motor. The PI controller parameters have been optimized off-line using GA based on a performance index to reflect the minimum settling time, minimum overshoot/undershoot and zero steady-state error. Based on the optimized operating conditions and control parameters the RBFN structure has been developed and trained on line to tune the PI controller parameters. The validity of the proposed control technique has been established both in simulation and experiment at different operating conditions. There is a close agreement between simulation and experimental results. The drive has been found robust in terms of quick response and disturbance rejection for a wide range of operating conditions.

9. Reference

[1] G. R. Slemon, "Electric Machines and drives", Addison-Wesley Publication Company, 1992, pp. 503-511.

[2] F. Blaschke, *The Principle of Filed Orientation as Applied to The New Transvector Closed-Loop Control System for Rotating-Field Machines*, Siemens Review, Vol. 34, No.3, pp. 217- 220, May 1972.

[3] M. A. Rahman and M. A. Hoque, "On-Line Self -Tuning ANN Based Speed Control of a PM DC Motor", *IEEE/ASME Trans. on Mechatronics*, vol. 2, No. 3, Sept. 1997, pp. 169-178.

[4] M. A. Rahman and Ping Zhou, "Field circuit analysis of brushless permanent magnet synchronous motors", *IEEE Trans. on Ind. Elect.*, vol.43, No.2, April 1996, pp. 256-267.

[5] X. Cui and K. G. Shin, "Direct control and coordination using neural networks", *IEEE Trans. on Man and Cybernatics,* vol. 23, no. 3, 1993, pp. 686-697.

[6] S. Weerasooriya and M. A. El-Sarkawi, "Identification and control of a DC motor using back propagation neural networks", *IEEE Trans. on Energy Conversion*, vol. 6, no. 4, Dec. 1991, pp. 663-669.

[7] S. Weerasooriya and M. A. El-Sarkawi, "Laboratory implementation of a Neural Network trajectory controller for a dc motor", *IEEE Trans. on Energy Conversion*, vol. 8, March 1993, pp. 107-113.

[8] F. M. Khouly, A.S. A. Gaffar, A. A. Mohammed and A. M. Sharaf, "Artificial Intelligent Speed Control Strategies for Permanent Magnet DC Motor Drives", *IEEE/IAS Annual Meeting Conference Record*, 1994, pp. 379-385.

[9] M. A. El-Sarkawi, A. A. El-Samahy and M. L. El-Syed, "High Performance Drive of dc Brushless Motors Using Neural Network", *IEEE Trans. on Energy Conversion*, vol. 9, no. 2, June 1994, pp. 317-322.

[10] M. A. Rahman and M. A. Hoque, "On-Line Adaptive Artificial Neural Network Based Vector Control of Permanent Magnet Synchronous Motors", *IEEE Trans. on Energy Conversion*, vol. 13, no. 4, 1998, pp. 311-318.

[11] Y. Yi, D. M. Vilathgamuwa, and M. A. Rahman, "A new artificial neural network controller for an interior permanent magnet motor drive", *IEEE/IAS Annual Meeting*, Chicago, 2001, pp. 1115-1120.

[12] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley, 1989.

[13] M. A. Abido and Y.L. Abdel-Magid, "Online identification of synchronous machines using radial basis function networks", IEEE Transactions on Power Systems, vol. 12, no. 4, Nov. 1997, pp. 1500-1506.

[14] *dSPACE*, "Digital Signal Processing and Control Engineering, Manual Guide, GmbH, Paderborn, Germany, 1996.

[16] Matlab, Simulink User Guide, The Math Works Inc., 1997.