GENETIC PI CONTROLLER FOR A PERMANENT MAGNET SYNCHRONOUS MOTOR

Cetin ELMAS
celmas@gazi.edu.tr
Gazi University,
Technical Education Faculty
Department of Electrical Education,
Ankara, TURKEY

M. Ali AKCAYOL
akcay@gazi.edu.tr
Gazi University,
Faculty of Engineering and architecting
Department of Industrial Engineering,
Ankara, TURKEY

ABSTRACT: Permanent magnet synchronous motors (PMSM's) is often used in electrical drives because of their simple structures, ease of maintenance and high efficiency. However, PMSM drive systems have a nonlinear characteristic arised from motor dynamics and load characteristics. To overcome with this problem, a Genetic PI speed controller for the drive system is proposed in this paper. Firstly, a mathematical model of a PMSM fed by three phase PWM inverter is developed. Then, the Genetic PI is designed and adapted to the drive system. Besides, to illustrate performance of proposed controller, a conventional proportional integral (conventional PI) controller is used to speed control of PMSM. Simulation is realized by proposed control strategies and simulation results are represented.

As an intelligent control technology the GA can give robust adaptive response of a drive with nonlinearity, parameter variation and load disturbance effect. In this paper, the Genetic PI speed controller was applied to the speed loop by replacing the conventional PI speed controller. The Genetic PI controller software was implemented using C++ Builder on a PC. Both the conventional and Genetic PI controller for the PMSM is implemented by using a TMS320F240 digital signal processor. The Genetic PI controller has superiority over conventional PI control. In addition, the results show that the Genetic PI controller is also less sensitive to the parameter variations and disturbances.

I. INTRODUCTION

Permanent magnet synchronous motors are of great interest especially for industrial applications in low-medium power range, since it has superior features such as compact size, high torque/weight ratio, high torque/inertia ratio and absence of rotor losses [1]. However, the performance of the PMSM is very sensitive to parameter variations and is spoiled due to external load disturbances in the system. To overcome with these problems several control strategies such as fuzzy logic control [2], sliding mode control [3], artificial neural network [4] have been proposed for speed and position control of PMSM.

Genetic algorithm can be used easily used in the control of systems that an exact mathematical model of system cannot be obtained at all [5]. Combining artificial neural network into genetic algorithm becomes an attractive field for many researchers. Genetic PI control is suitable for control of systems consisting uncertainties and nonlinearities [6] and gives smooth dynamic response.

In this paper, Genetic PI speed controller is designed for speed control of permanent magnet synchronous motor. Artificial neural network is used to adjust input and output parameters of membership functions in genetic algorithm control. For this reason, a Genetic PI network with four layer is designed. The back propagation learning algorithm is used for training this network. The proposed Genetic PI and conventional PI controller separately is simulated and simulation results are compared with each other. Simulation results show that the presented controller is reliable, effective. in the speed control of PMSM.

Various intelligent control and modeling strategies have been proposed for control and modeling of the PMSM, such as fuzzy modeling and control [1], modeling by neural networks [2]. Recent literature has also explored the potentials of the genetic algorithm (GA) for motor drive applications [3, 4, 5]. As an intelligent control technology the GA can give robust adaptive response of a drive with nonlinearity, parameter variation and load disturbance effect.

The conventional controller design (i.e. proportional-integral (conventional PI) controller) is based on mathematical model of the plant, which may often be unknown, ill-defined, nonlinear, complex and multivariable with parameter variation. Thus, the conventional PI controller is not an all-purpose solution for any motor drive applications.

In this paper, the Genetic PI speed controller was applied to the speed loop by replacing the conventional PI speed controller. An automatic tuning process using the improved iterative GA is used to optimize the conventional PI parameters. Only two parameters are sought but tuning is complicated by a significant nonlinearity caused by saturation of the speed controller. This means that optimum controller settings depend on the form of the required speed demand.

The results of applying the Genetic PI controller to the PMSM were compared to those obtained by the application of a conventional PI controller. The Genetic PI control provided better response than the conventional PI control in terms of accuracy, and insensitivity to changes in operating conditions. The software was developed using C++ Builder.

II. MODELING OF PMSM DRIVE SYSTEM

The configuration of PMSM drive system is given in Fig. 1. The drive system are composed of speed controller (Genetic PI or conventional PI), a current regulator, a hysteresis band current controller, a three phase PWM inverter and a position encoder.

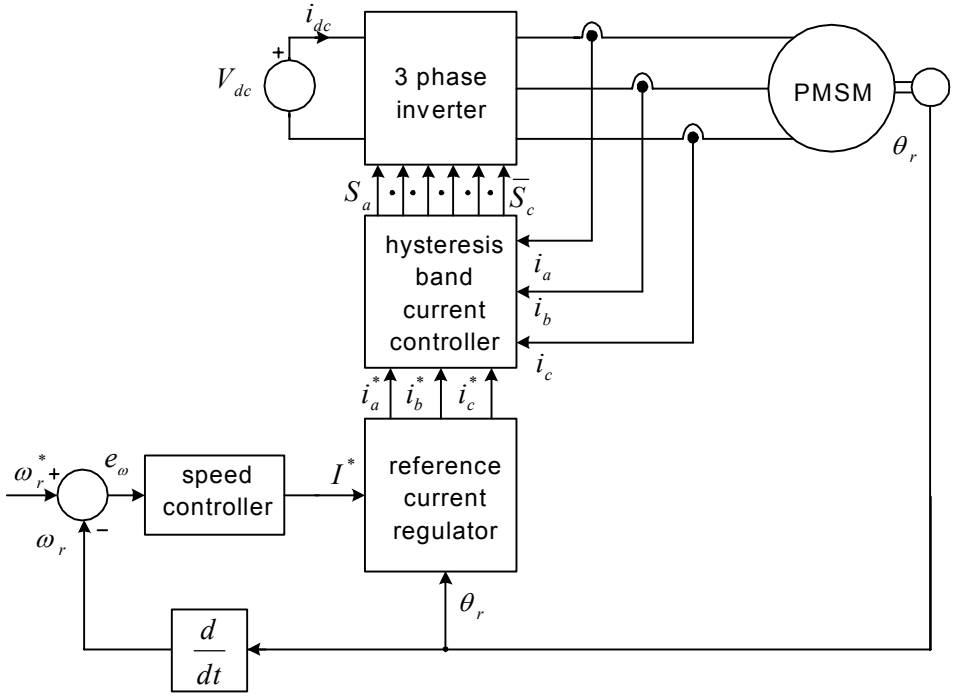


Figure 1. Block diagram of PMSM speed drive system

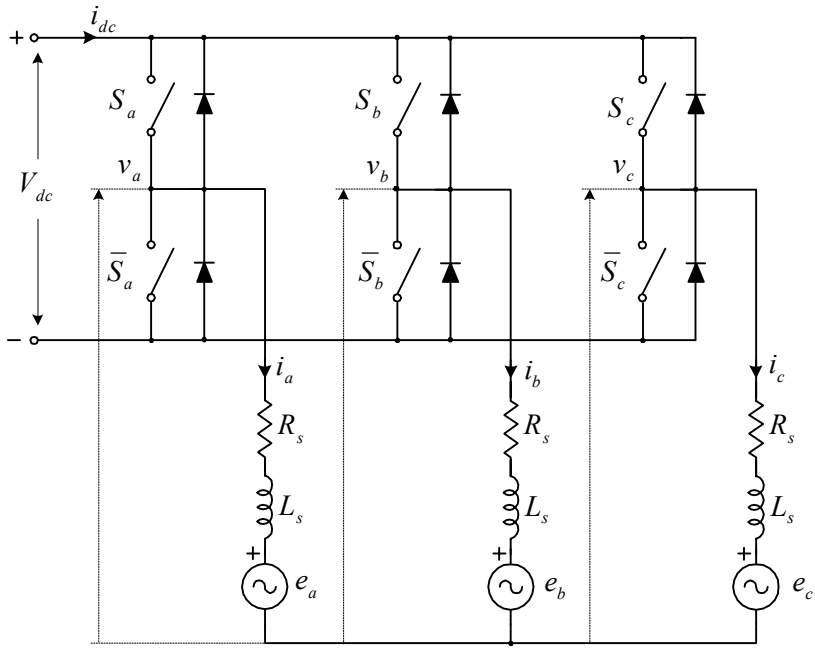


Figure 2. Equivalent circuit of PMSM and inverter

θ_r is rotor position, ω_r actual speed and i_a^*, i_b^*, i_c^* reference phase currents. e_ω speed error is difference between ω_r^* reference speed and ω_r actual speed. Using e_ω speed error, the speed controller generates I^* called as reference current or control current.

Equivalent circuit of PMSM and three phase full bridge inverter are given in Fig. 2. In Fig. 2, v_a, v_b, v_c are the phase voltages, e_a, e_b, e_c are the back emfs, R_s is the stator winding resistance, L_s is the synchronous inductance and i_a, i_b, i_c are the phase currents, V_{dc} is DC link voltage and i_{dc} is DC link current. Using this equivalent circuit, the model of motor is obtained as follows.

The stator voltage equations of PMSM in matrix form can be represented as, The statements of phase currents in Eq. (1) can be shown in state-space form as in Eq. (2).

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} L_s & 0 & 0 \\ 0 & L_s & 0 \\ 0 & 0 & L_s \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix} \tag{1}$$

$$\frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = \begin{bmatrix} L_s & 0 & 0 \\ 0 & L_s & 0 \\ 0 & 0 & L_s \end{bmatrix}^{-1} \left\{ \begin{bmatrix} -R_s & 0 & 0 \\ 0 & -R_s & 0 \\ 0 & 0 & -R_s \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} - \begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix} + \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} \right\} \tag{2}$$

The rotor speed and electrical torque can be written as,

$$\frac{d}{dt}\omega_r = \frac{p}{2}\left(T_e - T_L - B\left(\frac{2}{p}\right)\omega_r \right) \Big/ J \tag{3}$$

$$T_e = K_t I^* \tag{4}$$

where, K_t is $-\frac{3p}{4}\lambda_f$, and λ_f is the flux due to the permanent magnet rotor. The function of hysteresis band current controller is given in Eq. (5).

$$h_x = \begin{cases} 1 & if \quad i_x^* - i_x \le 0.5 h_{rb} \\ 0 & if \quad i_x^* - i_x \ge -0.5 h_{rb} \end{cases} \tag{5}$$

where, x represents respectively, a,b,c, h_x represents the functions of hysteresis band current controller h_a, h_b, h_c, h_{rb} is the range of band hysteresis band current controller. Using the obtained functions of hysteresis band current controller, Eq. (6) is obtained as follows,

$$\frac{d}{dt}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = \begin{bmatrix} L_s & 0 & 0 \\ 0 & L_s & 0 \\ 0 & 0 & L_s \end{bmatrix}^{-1} \left\{ \begin{bmatrix} -R_s & 0 & 0 \\ 0 & -R_s & 0 \\ 0 & 0 & -R_s \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} - \begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix} + \begin{bmatrix} \frac{(2h_a - h_b - h_c)}{3} \\ \frac{(-h_a + 2h_b - h_c)}{3} \\ \frac{(-h_a - h_b + 2h_c)}{3} \end{bmatrix} [V_{dc}] \right\} \quad (6)$$

1. GENETIC ALGORITHMS

The synthesis of a control system includes both the controller selection and the adjustment of its parameters. In some cases, the type of controller might be conventional PI. In this case, the tuning problem must be satisfactorily solved. To improve limitations of conventional PI controller especially when applied to high order systems, we propose Genetic PI controller for the BDCM. The structures of the proposed controller were motivated by the problems of conventional PI controllers that they generally give inevitable overshoot when one tries to reduce rise time of response especially when a system of order higher than one is under consideration. Since the undesirable characteristics of the conventional PI controller are caused by integrating operation of the controller, even though the integrator itself is introduced to overcome steady state error in response, we propose Genetic PI controller that clear out integrated quantities according to situation. The Genetic PI gives reduced rise time as well as small overshoot.

This initial tuning has been tested for the system and a qualitative tuning has also been established.

Abstract

This paper investigates the use of genetic algorithm (GA) for tuning the gains of the conventional PI. Such problems are very hard in general, and GA offers a useful and successful alternative to existing techniques. Thus Genetic Algorithm has made possible the establishment of intelligent control.

GA is a group of methods which solve problems using algorithms inspired by the processes of neo-Darwinian evolutionary theory. In a GA, the performance of a set of candidate solutions to a problem (called `chromosomes') are evaluated and ordered, then new candidate solutions are produced by selecting candidates as `parents' and applying mutation or crossover operators which combine bits of two parents to produce one or

more children. The new set of candidates is then evaluated, and this cycle continues until an adequate solution is found.

A GA can be seen as an unusual kind of search strategy. In a GA, there is a set of candidate solutions to a problem; typically this set is initially filled with random possible solutions, not necessarily all distinct. Each candidate is typically (though not in all GAs) an ordered fixed-length array of values (called `alleles') for attributes (`genes'). Each gene is regarded as atomic in what follows; the set of alleles for that gene is the set of values that the gene can possibly take. Thus, in building a GA for a specific problem the first task is to decide how to represent possible solutions. Assuming we have thus decided on such a representation, a GA usually proceeds in the following way:

Initialisation: A set of candidate solutions is randomly generated. For example, if the problem is to maximise a function of x, y and z then the initial step may be to generate a collection of random triples (xi,yi,zi) if that is the chosen representation.

-Now iterate through the following steps, until some termination criterion is met (such as no improvement in the best solution so far after some specified time, or until a solution has been found whose fitness is better than a given `adequate' value). The process alters the set repeatedly; each set is commonly called a generation.

1. Evaluation. Using some predefined problem-specific measure of fitness, we evaluate every member of the current set as to how good a solution to the problem it is. The measure is called the candidate's fitness, and the idea is that fitter candidates are in some way closer to being one of the solutions being sought. However, GAs do not require that fitness is a perfect measure of quality; they can to some modest extent tolerate a fitness measure in which the fitter of some pairs of candidates is also the poorer as a solution.

2. Selection. Select pairs of candidate solutions from the current generation to be used for breeding. This may be done entirely randomly, or stochastically based on fitness, or in other ways (but usually based on fitness, such that fitter individuals have more chance of being chosen).

3. Breeding. Produce new individuals by using genetic operators on the individuals chosen in the selection step. There are two main kinds of operators:

{ Recombination: A new individual is produced by recombining features of a pair of `parent' solutions.

{ Mutation: A new individual is produced by slightly altering an existing one.

The idea of recombination is that useful components of the members of a breeding pair may combine successfully to produce an individual better than both parents; if the ofispring is poor it will just have lower chance of selection later on. In any event, features of the parents appear in difierent combinations in the ofispring. Mutation, on the other hand, serves to allow local hill-climbing, as well introduce variation which cannot be introduced by recombination.

4. Population update. The set is altered, typically by choosing to remove some or all of the individuals in the existing generation (usually beginning with the least fit) and replacing these with individuals produced in the breeding step. The new population thus produced becomes the current generation.

The GA is an optimization routine based on the principles of Darwinian Theory and natural genetics. Since the inception of the GA concept by Holland in 1975 it has been useful in solving a wide variety of problem. In the use of the GA, there are two important aspects;

- chromosome coding
- defining the evaluation criteria.

The GA performs a parallel search of a parameter space by using genetic operators to manipulate a set of encoded chromosome which represents system parameters. The operation of the GA changes slightly depending on the base of the numbers to apply the genetic operators (*crossover, mutation, reproduction, elitism*). Traditionally GA's have been designed to operate over binary numbers (*0,1*) and more recently there have been several decimal numbers (*0,..,9*). The fitness of each of the members of the population is calculated using a fitness function that characterizes how well each particular member solves the given problem [6].

The GA has found application in the area of the automatic tuning process for conventional and intelligent controllers. Same research has been conducted using genetic algorithms to help on-line or off line control systems [7, 8]. It has primarily been utilized as an off-line technique for performing a directed search for the optimal solution to a problem. In this paper, the GA is used on-line in real-time controller implementation to adaptively search through a population of controllers and determine the member most fit to be implemented over a given sampling period.

In the Genetic PI controller tuning, each chromosome has a genes as a possible proportional and integral gain values. Chromosome fitness is evolved during evolution using the **i**ntegral with respect to **t**ime of the **a**bsolute speed **e**rror

(ITAE) This criteria balances error size and duration, and avoids positive and negative errors canceling. Best tuning is associated with smallest ITAE.

2. THE GENETIC conventional PI CONTROLLER FOR THE PMSM

The Genetic PI controller for the PMSM drives is shown in Fig. 1. The GA uses the principles of evolution and genetics to select and adapt the controller parameters (K_p and K_i). The controller parameters are coded by decimal numbers in chromosome. The candidate controllers of the Genetic PI controller are defined as members of the population. During time step, each member of the population is evaluated on how well it minimizes the ITAE.

For each member of the population, the GA computes the speed error (e_ω) and change in the speed error (ce_ω). The output variable of controller is change in the reference current ($\Delta_i(k)$). The e_ω and ce_ω are defined as:

$$e_\omega(k) = \omega^* - \omega(k), \tag{6}$$

$$ce_\omega(k) = e_\omega(k) - e_\omega(k+1) \tag{7}$$

where ω^* is the reference speed. Also, $.(k)$ and $.(k+1)$ denote actual and previous values, respectively.

In this application, feedback signals are the position θ and the phase currents $i_{a,b,c,d}$ and the position signal is used to calculate the speed. The switching signal generator is used to control turn-on angle θ_{on}, turn-off angle θ_{off}, and pulse width modulation duty cycle.

The steps for speed control are summarized as follows:

a) Sample the speed signal of the PMSM

b) Calculate the speed error and change in speed error.

c) Chose the number of digits to represent each controller parameter K_p and K_i. Chose crossover probability (p_c) and mutation probability (p_m).

d) Generate an initial population of K_p and K_i gains (we make a random selection) Initialize sample time T and set time t.

e) Generate $\Delta i(k)$, for each population member C_i, $i=1,2,...n$ using the conventional PI control laws.($\Delta i(k) = K_p.e_\omega(k) + K_i.e_\omega(k).T$)

f) Assign fitness to each element of the population C_i, $i=1,2,3,....n$,

$$p_1 = e_\omega(k) \tag{8}$$

$$p_2 = \Delta i(k) \tag{9}$$

$$F = \frac{1}{(\alpha_1 . p_1^2 + \alpha_2 . p_2^2)} \tag{10}$$

g) Produce the next generation using GA operators and let $t:=t+T$ go to step (d)

h) The maximally fit C_i becomes C^* and send the change of control action ($i^*(k)$) to control the drive.

Where $i^*(k)$ is the inferred change of reference current by the controller at the kth sampling time and defined as

$$i^*(k) = i^*(k+1) + \Delta i(k) \tag{11}$$

where, $i^*(k+1)$ is the previous reference current.

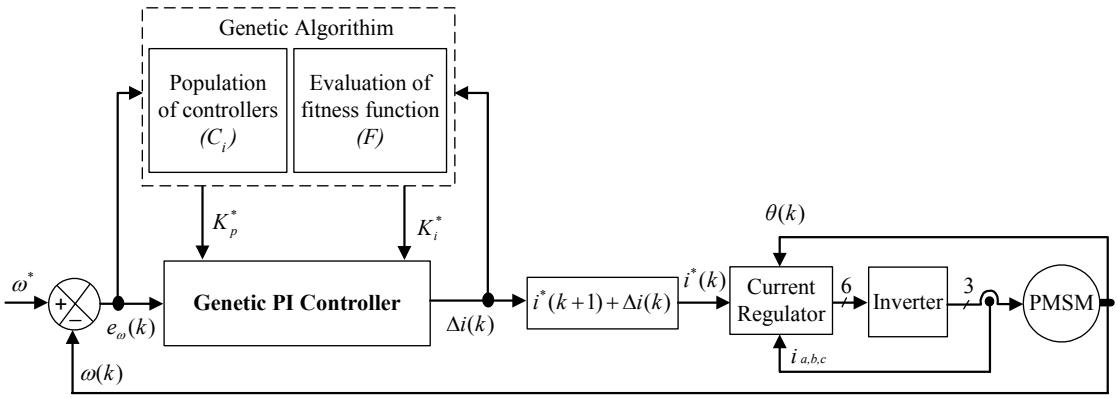


Figure 1. The Genetic PI controller for the PMSM drives

THE EXPERIMENTAL SETUP SHEME OF THE PMSM DRIVE

The Genetic PI speed controller and the conventional PI speed controller for the PMSM is implemented by using a TMS320F240 DSP EVM. It is 16-bit fixed point and has 50 ns instruction cycle. It is able to perform, in an efficient way for the Genetic PI algorithms, in order to obtain the best performance. It has PWM outputs allow managing directly power devices and high frequency PWM controls. Three half bridge IGBT modules (CM75DU-12H) are used for three phase voltage source inverter. The experimental setup scheme is shown in Figure 5.

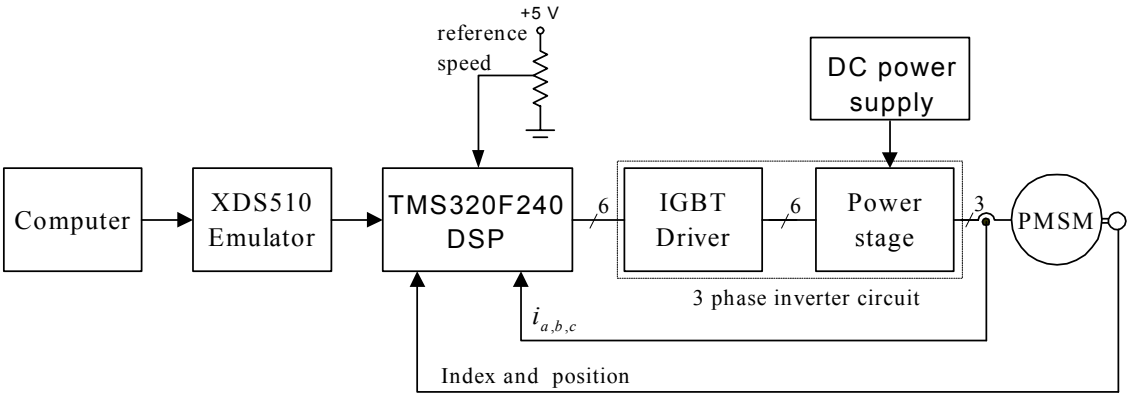
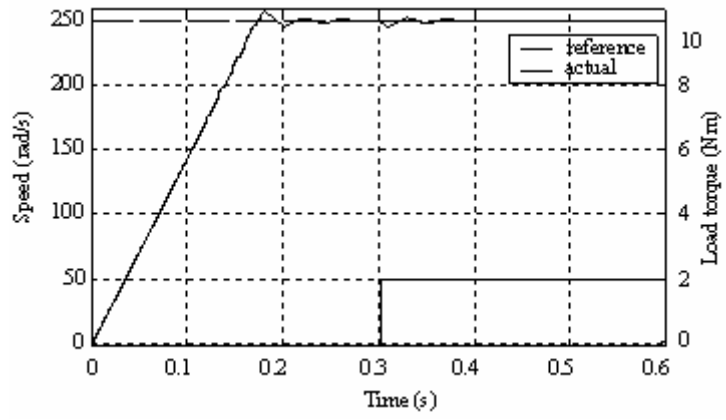


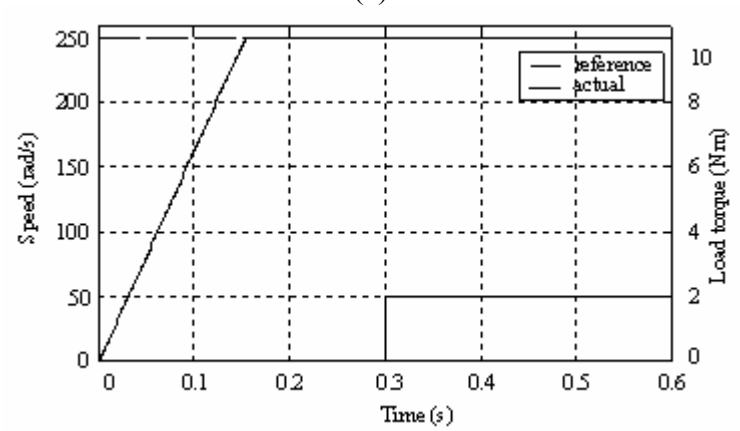
Figure 5. The experimental setup scheme of the PMSM drive

V. SIMULATION RESULTS

The Genetic PI and conventional PI controller separately is simulated and simulation results are compared with each other. For different case such as step, variable speed, variable load, simulation results are obtained. The obtained results are given in Fig. 6-12



(a)



(b)

Fig.6 Step speed response ($t = [0.2,0.4]s$, $T_L = 2Nm$) a) Genetic PI b) conventional PI

In Fig.6, reference speed is 250 rad/s. At *t=0.2-0.4s*, *2Nm* load torque is applied in system. In Fig.6, the time interval applied load torque is zoomed and given in Fig. 7.

Fig. 7(a) shows speed response for Genetic PI and Fig. 7(b) shows speed response for conventional PI controller. In speed response for conventional PI controller, oscillation is exist. It is obvious that the performance of Genetic PI as speed response is better than the conventional PI controller for this case.

In Fig. 8, reference speed is 250 rad/s for *t=0-0.*3s time interval, reference speed is 500 rad/s for *t>0.*3s time interval and load torque is zero. As Fig. 8(a) and Fig. 8(b), the performance of Genetic PI is better than the conventional PI controller for this case.

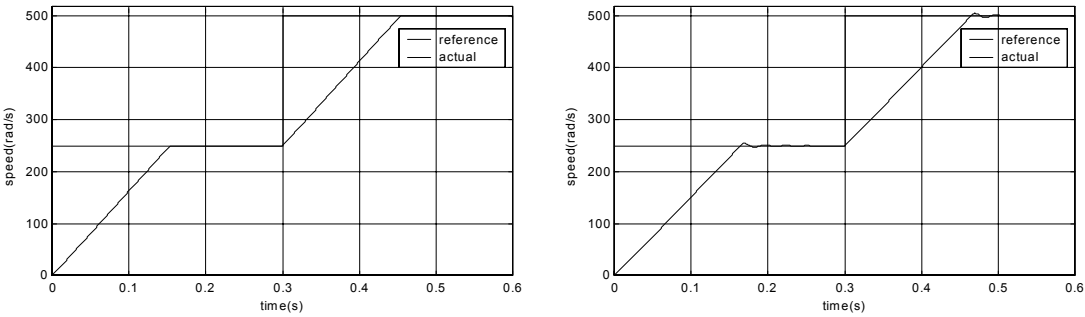
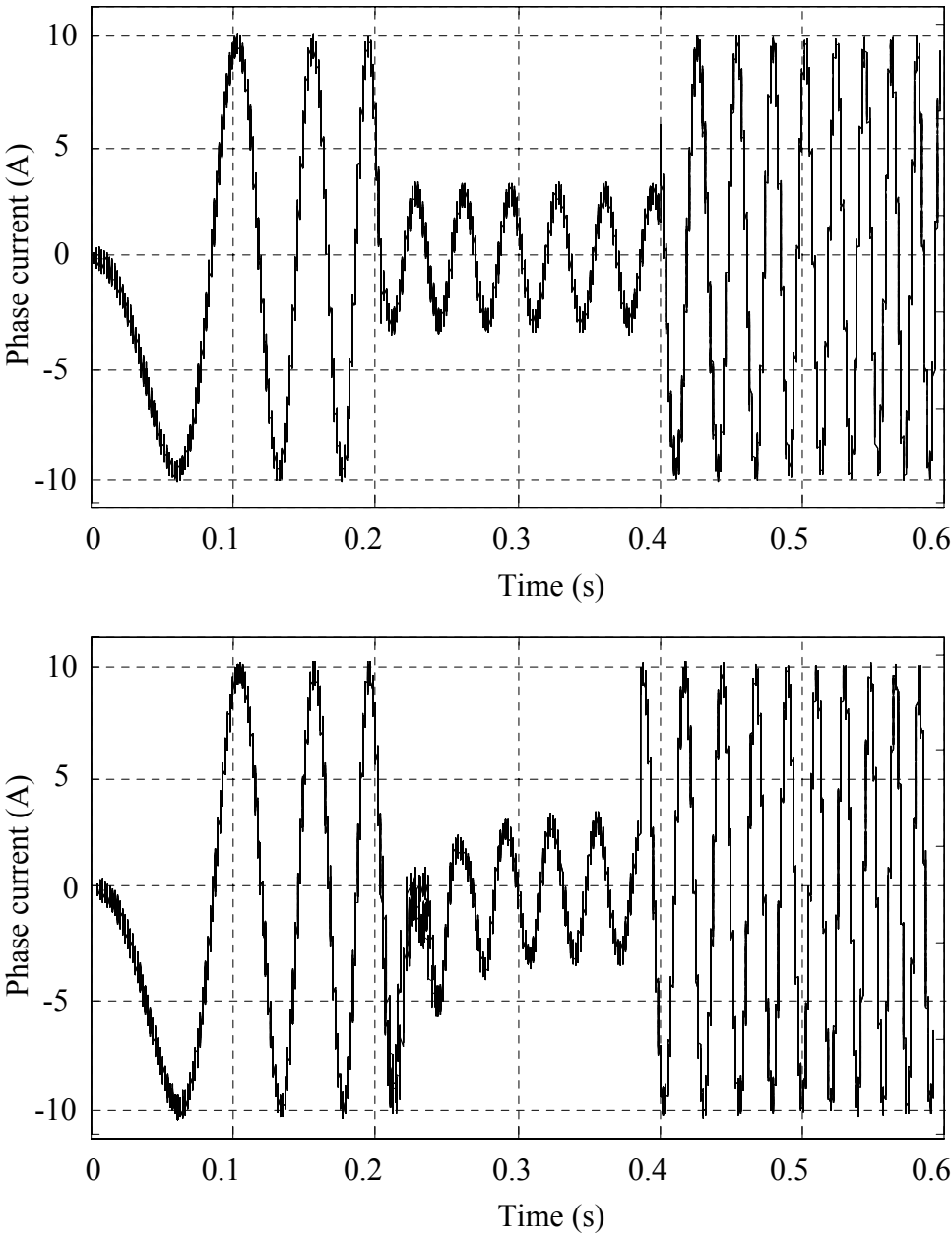


Figure 9. Speed response for variable reference speed ($T_L = 1Nm$)a) Genetic PI b) conventional PI

In Fig. 9, reference speed is 250 rad/s for *t=0-0.*3s time interval, reference speed is 500 rad/s for *t>0.*3s time interval and load torque is 1Nm. As Fig. 9(a) and Fig. 9(b), the performance of Genetic PI is better than the conventional PI controller for this case. As seen from Fig. 8 and Fig. 9, the motor speed reaches to the 250 rad/s at 0.1 s for $T_L = 0Nm$ and at 0.15 s for $T_L = 1Nm$.



(a) (b)

Figure 10. A graphics of phase current ($T_L = 1Nm$) a) Genetic PI b) conventional PI

In Fig. 10 shows a graphics of phase current for the case given in Fig. 9 Fig. 10(a) shows a graphics of phase current for Genetic PI and Fig. 10(b) shows a graphics of phase current for conventional PI controller. The obtained graphics of phase current for Genetic PI is more smooth than for conventional PI controller.

VI. CONCLUSIONS

In this paper, a Genetic PI speed controller for PMSM drive system is presented.. a mathematical model of a PMSM fed by there phase PWM inverter is realized, then the

Genetic PI speed controller for speed control of PMSM are designed. The Genetic PI and conventional PI controller are designed and simulated individually and results are given. From the simulation results, it is clear that the designed the Genetic PI controller has better speed response than conventional PI controller.

Appendix: Motor parameters

$I_m = 10A$, $V_{dc} = 180V$, $R_s = 1.6\Omega$, $L_s = 0.0025H$, $\lambda_{af} = 0.111V/rad/s$,

$J = 0.00245Nms^2$

REFERENCES

1. G. R. Slemon, Electrical machines for variable-frequency drives, Proceeding of IEEE, Vol.82, No.8, pp. 1123-1139, 1994.
2 M. A. Akcayol, A. Cetin, and C. Elmas, An Educational Tool for Genetic algorithm-Controlled BDCM,IEEE Transactions On Education, Vol. 45, No. 1, pp. 33-42, 2002.
3 J.P. Karunadasa and A.C. Renfrew, Design and implementation of microprocessor based sliding mode controller for brushless servomotor, IEE Proceedings-B, Vol.138, No.6, pp.345-363, 1991.
4. 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, pp. 311-318, 1998.
5. C.C. Lee, Genetic algorithm in control systems: genetic algorithm controller, Part I, Part II, IEEE Trans. on Syst., Man, and Cyber., Vol. 20, No.2, pp. 404-435, 1990.
6. C.T. Lin and C.S.G. Lee, Neural-network-based genetic algorithm control and decision system, IEEE Trans. On Comp., Vol.40,No. 12, pp.1320-1336, 1991.

6. SIMULATION AND EXPERIMENTAL RESULTS

To explore the effectiveness of proposed technique, both computer simulation and practical experimental work have been carried out. The Genetic PI controller software was implemented using C++ Builder on a PC. The motor parameters are given in the Appendix.

The results of applying the Genetic PI controller to the PMSM were compared to those obtained by the application of a conventional PI controller. The output of the controller is applied to a chopper that controls the duty cycle and as a result, the applied voltage to the phase winding. The conventional PI controller gains are tuned manually for different working conditions. To improve the performance for Genetic PI controller, the members of the population are defined. Each of candidate controllers is defined by a fourteen digit numbers. The first seven digits describe the proportional gain and last seven digits describe the integral gains. Limits of the controller gains are needed to start. In practice, these limits could be defined by experience, using off-line simulation or by simplified control theory. In this study, the ranges of the proportional (*0-10*) and integral (*500-1000*) gain have been set exceptionally wide to illustrate the operation of the GA to maximum effect. A population size of *20* chromosomes and chromosome length 14

gene. Is initially created and evolved throughout *10* generations. The crossover and mutation rate are *0.9* and *0.1* respectively.

Figure 3 and Figure 4 show the speed response of the PMSM with conventional PI and Genetic PI controller respectively. Simulation traces in Figure 3 and Figure 4 shows how the system follows the reference speed profile. While motor working at stand-still, a 10 Nm load torque was applied between 1-2 ms. After 2 ms the load was removed.

As seen from figures, all results indicate Genetic PI controller has superiority over conventional PI control. In addition, the results show that the Genetic PI controller is also less sensitive to the parameter variations and disturbances. Genetic PI controller has also less overshoot and ripple than that of the result obtained by conventional PI controller.

Experimental results are shown in Figure 5 and Figure 6. As seen from figures, there is generally good agreement between simulation and experimental results.

Figure 3. Speed response of the PMSM with conventional PI controller under step load torque

Figure 4. Speed response of the PMSM with Genetic PI controller under step load torque

Figure 5. Experimental speed response of the PMSM with conventional PI controller under step load torque (vert. 1 V/div, horz. 0.5 s/div)

Figure 6. Experimental speed response of the PMSM with Genetic PI controller under step load torque (vert. 1 V/div, horz. 0.5 s/div)

7. CONCLUSIONS

An application of genetic algorithm to control of the PMSM drive was presented in this paper. The Genetic PI controller has superiority over the conventional PI control. Genetic PI controller provided a better response than the conventional PI control. The Genetic PI controller shows robustness over a wide range of operations. In addition, the results show that the Genetic PI controller is also less sensitive to the parameter variations and disturbances. Genetic PI controller has also less overshoot and ripple than that of the result obtained by conventional PI controller.

8. REFERENCES

1. W.G. da Silva, P.P. Acarnley and J.W. Finch, "Application of genetic algorithms to the online tuning of electric drive speed controllers" Trans. IEEE,IE-47, No. 1,pp. 217-219,2000.

2. J. Wang and Y. Zhao, "The optimization of conventional PI controller parameters using genetic algorithm in the DC speed control system", Proceedings of the 3rd World Congress on Intelligent Control and Automation, pp. 545-548, 2000

3. C., Vlachos, at al, "Genetic approach to decentralized conventional PI controller tuning for multivariable processes", IEE Proceedings Control Theory and Applications, Vol. 146, No. 1, pp58-64, 1999

4. D. E., Goldberg, Genetic Algorithms in Search, Optimization and Motor Learning, Reissue, Addison-Wesley Publishing Company, 1989.

5. K., Kristinsson, and G.,Dumont, "System identification and control using genetic algorithm, IEEE Trans. on Man. and Cybernetics,Vol.22, No.5, pp. 1033-1046, 1992

6. W.K., Lennon, and K.M., Passino, "Genetic adaptive identification and control", Engineering Applications of Artificial Intelligence, Vol.12, No.2,pp.185-200., 1999.

7. Texas Instruments, TMS320F240 DSP Controllers Evaluation Module, July, 1999.