SERBIAN JOURNAL OF ELECTRICAL ENGINEERING
Vol. 2, No. 1, May 2005, 93 - 116

High – Performance Adaptive Intelligent Direct Torque Control Schemes for Induction Motor Drives

M. Vasudevan[1], R. Arumugam[2], S.Paramasivam[3]

Abstract: This paper presents a detailed comparison between viable adaptive intelligent torque control strategies of induction motor, emphasizing advantages and disadvantages. The scope of this paper is to choose an adaptive intelligent controller for induction motor drive proposed for high performance applications. Induction motors are characterized by complex, highly non-linear, time varying dynamics, inaccessibility of some states and output for measurements and hence can be considered as a challenging engineering problem. The advent of torque and flux control techniques have partially solved induction motor control problems, because they are sensitive to drive parameter variations and performance may deteriorate if conventional controllers are used. Intelligent controllers are considered as potential candidates for such an application. In this paper, the performance of the various sensorless intelligent Direct Torque Control (DTC) techniques of Induction motor such as neural network, fuzzy and genetic algorithm based torque controllers are evaluated. Adaptive intelligent techniques are applied to achieve high performance decoupled flux and torque control. This paper contributes:

i) Development of Neural network algorithm for state selection in DTC;

ii) Development of new algorithm for state selection using Genetic algorithm principle; and

iii) Development of Fuzzy based DTC. Simulations have been performed using the trained state selector neural network instead of conventional DTC and Fuzzy controller instead of conventional DTC controller.

The results show agreement with those of the conventional DTC.

Keywords: Direct Torque Control, Induction Motor, Intelligent Control, Fuzzy, Neural Networks and Genetic Algorithm.

[1]Department of Electrical and Electronics Engineering, Anna University, Chennai-600025, India; E-mail address: vasumame@yahoo.com
[2]Department of Electrical and Electronics Engineering, Anna University, Chennai-600025, India
[3]Department of Electrical and Electronics Engineering, Anna University, Chennai-600025, India

1 Introduction

Direct Torque Control (DTC) of pulse - width - modulated inverter fed induction motor drive is receiving wide attention in the recent years [1, 2]. Fig. 1 shows the basic configuration for the direct torque controlled induction motor drive. The scheme uses stator flux vector and torque estimators on a PWM – inverter-fed drive. The stator flux amplitude ψ_s^* and torque T_e^* are the command signal and which are compared with the estimated ψ_s and T_e values, respectively, giving instantaneous flux error E_ψ and torque error E_{Te} as shown in Fig. 1.

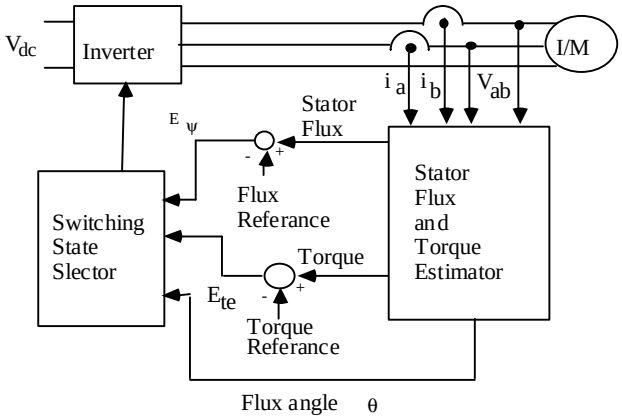

Fig. 1 - *Basic configuration of DTC scheme.*

In the conventional scheme, the flux error E_ψ and torque error E_{Te} signals are delivered to two hysteresis comparators. The corresponding digitalized output variables and the stator flux position sector create a digital word, which selects the appropriate voltage vector from the switching table. Selection of voltage vector is also depending upon the sector in which the stator flux positioned [3]. Thus, the selection table generates pulses S_a, S_b, S_c to control the power switches in the inverter. Fig.2 shows the pulses S_a, S_b, S_c generated when the position of stator flux is in sector 1 [4].

The expression for the developed torque of an induction motor is given by (1).

$$T = \frac{N_p M \Psi_s \Psi_r \sin\delta}{\sigma L_s L_r}, \tag{1}$$

where: $\sigma = \dfrac{1-M^2}{L_s L_r}$ and Ψ_s is stator flux.

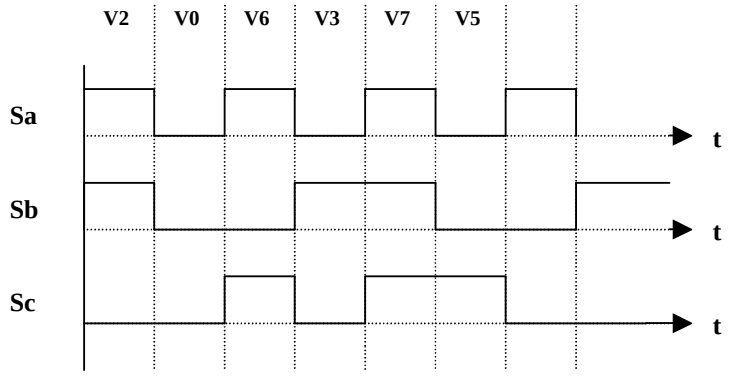

Fig. 2 – *Generation of Pulses for PWM inverter when flux vector lies on sector 1.*

Under normal operating conditions, the amplitude of the working flux is kept constant at the maximum value. Hence the developed torque is proportional to the sine of the torque angle 'δ' between stator and rotor fluxes, and can be controlled by suitably changing the angle 'δ'. Since the time constant of rotor current is large compared to stator, the stator flux is accelerated or decelerated with respect to the rotor flux to change the torque angle. Stator flux is a computational quantity, which is obtained using the stator-measured current ' I_s' and voltage 'V_s'.

$$\psi_S = \int_0^t (V_S - I_S R_S)\,\mathrm{d}t\,. \tag{2}$$

In general, conventional DTC scheme has the following disadvantages [5]:

i) Variable switching frequency
ii) Violence of polarity consistency rules
iii) Current and torque distortions caused by the sector changes
iv) Starting and low - speed operation problems
v) High sampling frequency needed for digital implementation of hysteresis comparators.

Introducing adaptive controllers instead of conventional hysteresis controllers can eliminate all the above difficulties. In this paper, viable intelligent controllers in DTC scheme are discussed to improve the performance in low speed operations and to minimize the torque ripple. Intelligent controls using expert systems, fuzzy logic, neural networks and genetic algorithms have been recently recognized as important tools to enhance the performance of the power electronic systems [6, 8]. The combination of intelligent control with adaptive and robust control appears today the most promising research accomplishment in

the drive control area and in the meantime, as the best approach for the optimal exploitation of intelligent control prerogatives and practical realization of adaptive and robust ac motor drives. In this paper, detailed investigations on viable intelligent torque control schemes are carried out by simulation and the results are compared.

2 Neural Network Controllers for DTC Scheme

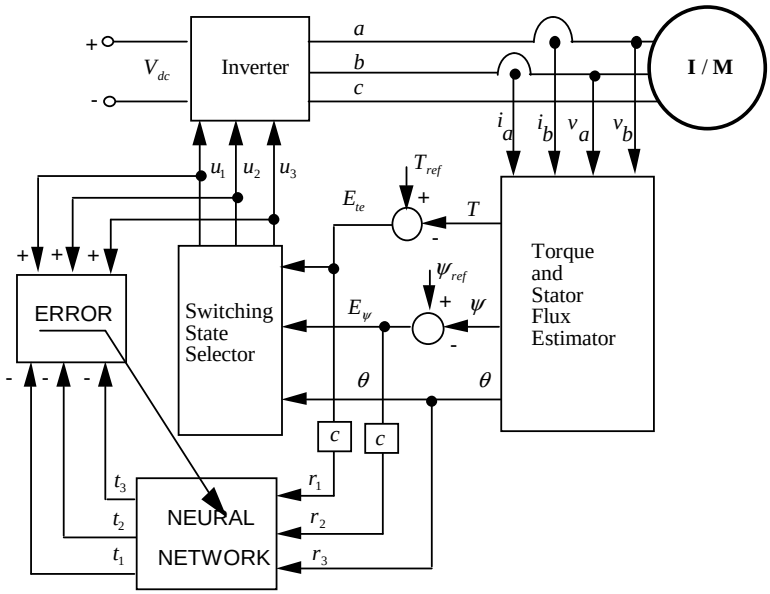

Fig. 3 – *Schematic of DTC using Neural-Network controller.*

A neural network is a machine like human brain with properties of learning capability and generalization. They require a lot of training to understand the model of the plant. The basic property of this network is that it enables approximation of complicated nonlinear functions [8, 11]. In direct torque control scheme, neural network is used as a sector selector. The direct torque neuro controller is shown in Fig. 3. In this control strategy, torque and flux errors are multiplied by the constant value '*c*' and which are given as inputs along with the flux position information to the neural network controller. Output of the controller is compared with the previous switching states of inverter. Artificial Neural Network (ANN) offers inherent advantages over other conventional DTC schemes for induction motors, namely:

i) Reduction of the complexity of the controller;

ii) Reduction of the effects of motor parameter variations, particularly in the stator-flux estimation;

iii) Improvement of controller time response, i.e., the ANN controller uses only parallel processing of sums, products by constant gains, and a set of well known non-linear functions so that no time- consuming sequential integrations routines are required;

iv) Improvement of drive robustness – ANN's are fault tolerant and can extract useful information from noisy signals.

3 Principles of Artificial Neural Networks

Feed forward artificial neural networks are universal approximators of nonlinear functions [8, 10]. As such, the ANN's use a dense interconnection of neurons that correspond to computing nodes. Each node performs the multiplication of its input signals by constant weights, sums up the results, and maps the sum to a nonlinear function; the result is then transferred to its output. The structure of neuron is shown in Fig. 4 and the mathematical model of a neuron is given by

$$y = \varphi\left(\sum_{i=1}^{N} \omega_i x_i - b\right), \tag{3}$$

where, $x_i = (x_1, x_2, \ldots, x_N)$ are inputs from the previous layer neurons, $\omega_i = (\omega_1, \omega_2, \ldots, \omega_N)$ are the corresponding weights and '*b*' is the bias of the neuron.

For a logarithmic sigmoidal activation function, the output is given by

$$y = \frac{1}{1 + \exp\left(\sum_{i=1}^{N} \omega_i x_i - b\right)}. \tag{4}$$

A feed forward neural network is organized in layers: an input layer, one or more hidden layers, and an output layer. No computation is performed in the input layer and the signals are directly supplied to the first hidden layer through input layer. Hidden and output neurons generally have a sigmoidal activation function. The knowledge in an ANN is acquired through a learning algorithm, which performs the adaptation of weights of the network iteratively until the error between the target vectors and output of network falls below a certain error goal. The most popular learning algorithm for multi-layer networks is the back propagation algorithm, which consists of a forward and backward action. In the first, the signals are propagated through the network layer by layer. An output vector is thus generated and subtracted from the desired output vector. The resultant error vector is propagated backward in the network and serves to adjust

the weights in order to minimize the output error. The back propagation training algorithm and its variants are implemented by many general – purpose software packages such as the neural-network toolbox from MATLAB [13, 14] and the neural-network development systems described in [12]. The time required to train an ANN depends on the size of the training data set and training algorithm. An improved version of back propagation algorithm with adaptive learning rate is proposed and which permits a reduction of the number of iterations. Fig. 5 shows the proposed neural network for DTC scheme in which, input, output and hidden layers are shown. The error signals and stator flux angle are given to input layer. Switching state information is taken from the output layer.

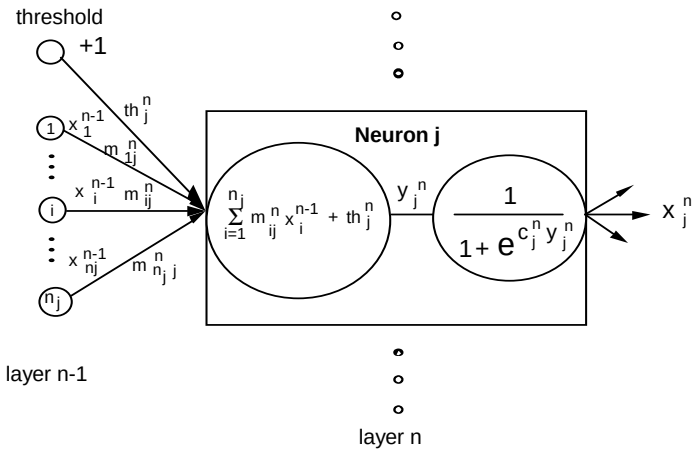

Fig. 4 – *Structure of Neuron.*

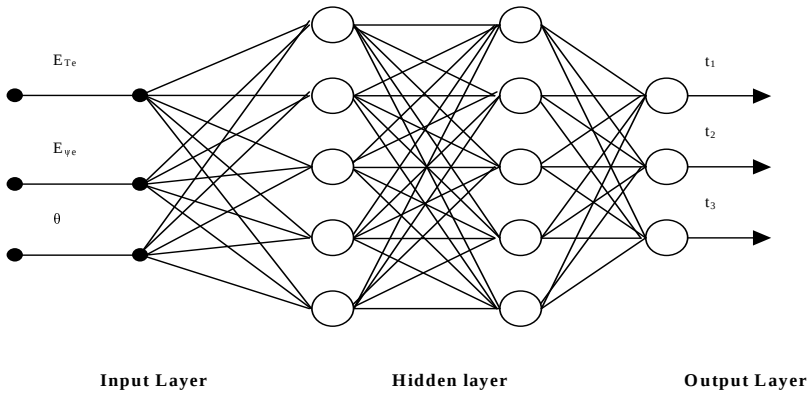

Fig. 5 – *Structure of Neural network proposed for DTC scheme.*

4 DTC Using Genetic Algorithm

Genetic algorithms are stochastic global search algorithms. They mimic processes observed in natural evolution and use a vocabulary borrowed from the natural genetic [15]. A GA considers individuals in a population quite often called strings or chromosomes and must have the following components:

i) A genetic representation for potential solution encoded as strings or chromosomes;
ii) A way to create an initial population of potential solutions;
iii) An evaluation function for rating solutions in terms of their fitness;
iv) Genetic operator that alter the composition of children;
v) Values for various parameters that the genetic algorithm uses (population size, probabilities of applying genetic operators, etc.).

Given these five components, a genetic algorithm is constructed as follows:

i) Initialize a population of chromosomes;
ii) Evaluate each chromosome in the population;
iii) Select chromosomes in the population as parent chromosomes to reproduce;
iv) Apply the genetic operators to the parent chromosomes to produce children;
v) Evaluate the new chromosomes and insert them into the population;
vi) If the termination condition is satisfied, stop and return the best chromosome. If not go to step (iii).

For executing genetic algorithm to train the neural networks, detailed procedures were followed. Fig. 8 shows the flowchart to execute a genetic algorithm. It gives an algorithm to select best chromosome from the total population of chromosomes. To select best chromosome, parent selection is prominent. Steps for parent selection are summarized as follows:

i) Selection of parents for reproduction is stochastic;
ii) Selection of parents with higher fitness value;
iii) Roulette wheel technique for parent selection. A roulette wheel shown in Fig. 7 has slots, which are sized according to the fitness of each chromosome;
iv) Selection process is to spin the roulette wheel.

In Fig. 6, f_1, f_2, f_3, f_4, f_5 are fitness of chromosomes 1, 2, 3, 4 and 5, respectively. Pop represents the total population size; that is, if total number of chromosomes is 50, population size is also 50. Therefore,

$$f_{pop} = f_{50} = \text{ Fitness of } 50^{\text{th}} \text{ chromosome.} \tag{5}$$

Total fitness is given by $F = \text{Sum}$ of the fitness of the population,

$$F = \sum_{j=1}^{pop} eval_j \ . \tag{6}$$

Probability function for each chromosome is

$$p_i = eval_i / F \ , \quad i = 1, 2, 3, \ldots, pop \ . \tag{7}$$

Accumulative probability function for each chromosome is

$$q_i = \sum_{j=1}^{i} p_i \ , \quad i = 1, 2, 3, \ldots, pop \ . \tag{8}$$

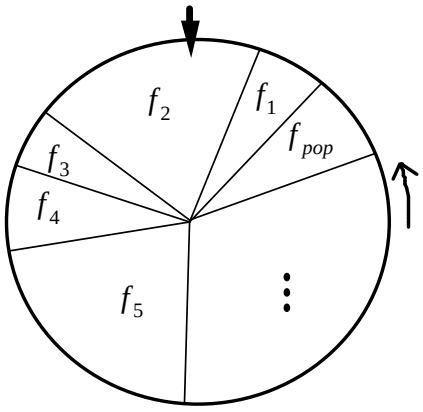

Fig. 6 – *Roulette Wheel.*

5 Neural Networks Trained by Genetic Algorithms

In neural networks, genetic algorithms are used to determine the weights and threshold values. Fig. 7 shows the structure of neural networks trained using GA [15]. The respective error vectors between the state selector of conventional DTC and the neural networks outputs are e_1, e_2, e_3. To achieve minimum value of performance index, the groups of threshold values and weights have to be determined.

Performance index $E(W)$ can be given by:

$$E(W)=\frac{1}{2}\sum_{j=1}^{N} e^{T}(j)\Lambda e(j), \tag{9}$$

where: $e^{T}=[e_1 \quad e_2 \quad e_3]^{T}$ is error vector;

Λ is symmetric positive definite matrix; and

N is sample size.

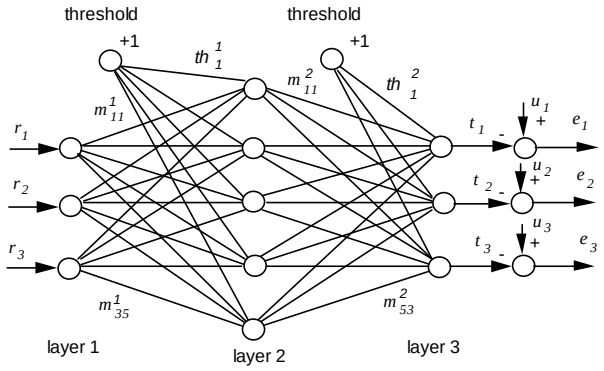

Fig. 7 – *Structure of neural networks trained using GA.*

Implementation of the genetic algorithm described in this paper has three stages:

i) Fitness evaluation

ii) Selector

iii) Breeding

The genetic operators used in this work are quite different from the classical ones used in [15].

The main differences between the proposed work and existing work are described as follows:

i) The real valued space are dealt in this paper, where a solution is coded as a vector with floating point type components

ii) Some genetic operators are non-uniform, that is, their action depends on the age f the population.

The contents of the algorithm are listed below:

5.1 Chromosome Encoding

Let the total number of thresholds and weights of the neural network shown in Fig. 7, be packed in the n-dimensional vector *W*,

$$W = \left[th_1^2 m_{11}^2 \ldots m_{51}^2 \ldots th_5^1 m_{15}^1 \ldots m_{35}^1\right] = \left[w_1 w_2 \ldots w_{38}\right]$$

Here, the weights vector *W* as a chromosome (individual). In other words, each chromosome vector is coded as a vector of floating point numbers of the same length as the solution vector. Each element is initially selected as to be within the desired domain.

5.2 Evaluation Function

The evaluation function for chromosomes '*w*' is

$$eval(W) = \frac{100}{1 + E(W)}, \tag{9a}$$

where, the chromosome vector *W* is a real weights vector, and *E*(*W*) is defined by equation (9). The evaluation function is used to rating a chromosomes in terms of their "fitness". The higher fitness of chromosome will perform better.

5.3 Genetic Operators

In this paper, both binary and floating point encoding are used as genetic operators to train the neural networks in DTC technique.

The binary operators are one point crossover, two points crossover and bit mutation. The operators used for floating point encoding are different from classical ones. They work in a real valued space. However, because of intuitive similarities, they are divided into the standard classes of mutation and crossover. Mutation groups used in this paper are Uniform mutation (UM), Non-Uniform Mutation (NUM) and Non- Uniform Arithmetical Mutation (NUAM). Crossover groups are Two-Points Crossover (TPC) and Two-Points Arithmetical Crossover (TPAC).

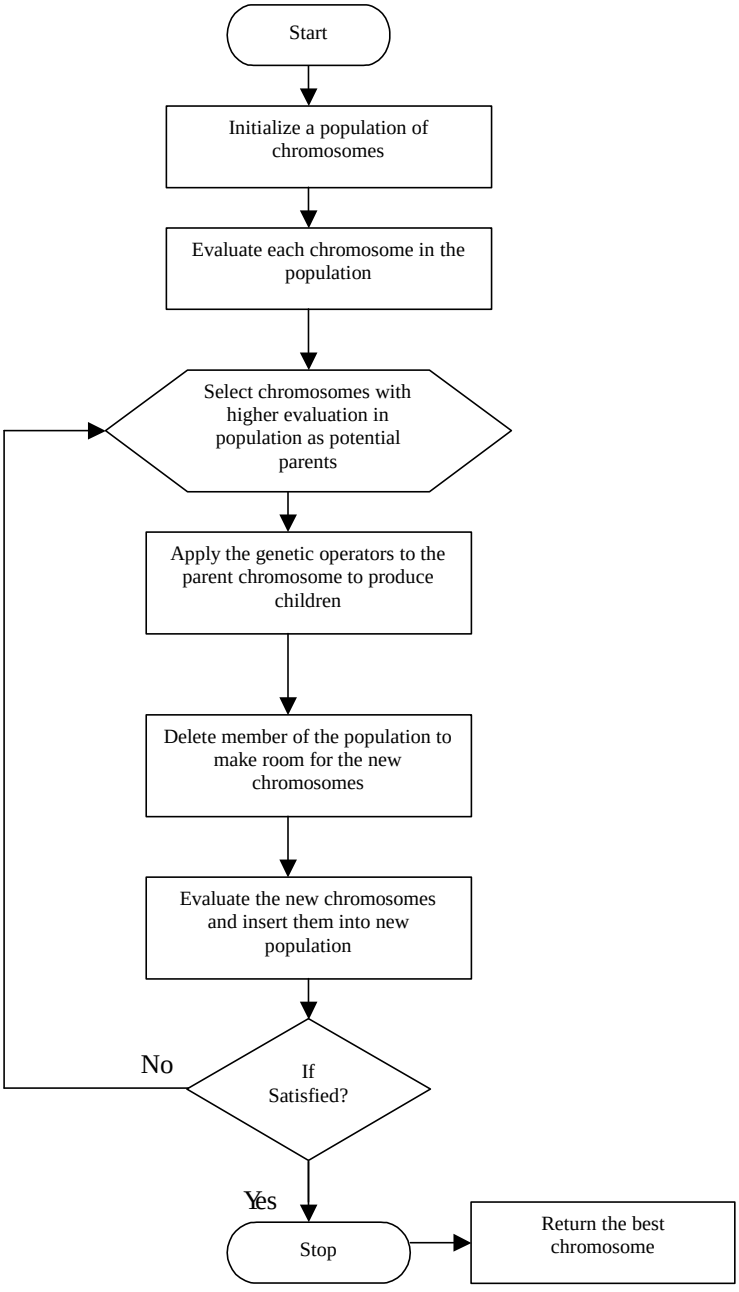

Fig. 8 – *Flowchart for execution of a Genetic Algorithm.*

6 Fuzzy Logic Direct Torque Control of Induction Motor

In DTC induction motor drive, there are torque and flux ripples because none of the inverter states is able to generate the exact voltage value required to make zero both the torque electromagnetic error and the stator flux error [6, 7]. The suggested technique is based on applying switching state to the inverter and the selected active state just enough time to achieve the torque and flux references values. A null state is selected for the remaining switching period, which won't almost change both the torque and the flux. Therefore, the switching state has to be determined based on the values of torque error, flux error and stator flux angle. Exact value of stator flux angle (θ) determines where stator flux lies [8].

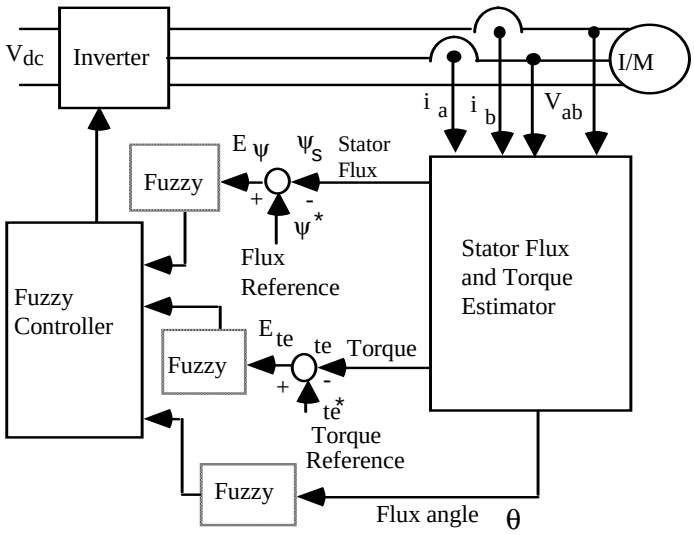

Fig. 9 – *Schematic of fuzzy logic DTC.*

The schematic of fuzzy logic direct torque control scheme for induction motor drive is shown in Fig. 9. The fuzzy output of torque, flux errors and stator flux angle are given as input variables to fuzzy controller and output variable obtained from the fuzzy controller is switching state of the inverter. Switching state of the inverter is a crisp value. The input variables membership functions are shown in Fig. 10.

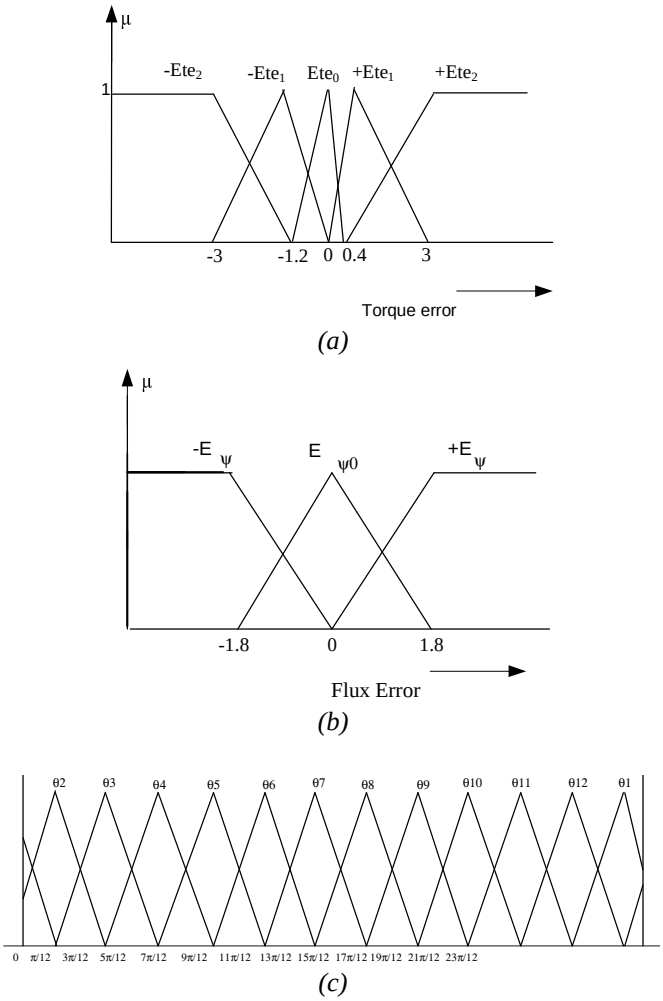

Fig. 10 – *Membership distributions for input variables (a) Torque error (b) Flux error and (c) Stator flux angle.*

7 Fuzzy Rules for Direct Torque Control Scheme

To improve the performance of classical DTC scheme, Fuzzy rules have been developed. In the **Table 1**, '1' represents the upper limb switches and '0' represents the lower limb switches of the inverter. Switching states of the inverter varies from V_0 to V_7. From this table it is concluded that, V_0=V_7 and which are null states. That is, V_0 and V_7 are zero vectors. The fuzzy system comprises 12 groups of rules and each of which contains 15 rules. Each group represents the respective stator flux angle θ. For example, rules are shown in

Table 2 for stator flux angle θ_1, θ_2 and θ_3. For every combination of inputs and outputs, one rule can be applied. Totally, there are twelve-stator flux angles from θ_1 to θ_{12} and 180 rules are formed. With the help of them, corresponding switching state of the inverter is selected.

Table 1

Switching States of Voltage Vectors

States	$\mathbf{u_1}$	$\mathbf{u_2}$	$\mathbf{u_3}$
V_0	0	0	0
V_1	1	0	0
V_2	1	1	0
V_3	0	1	0
V_4	0	1	1
V_5	0	0	1
V_6	1	0	1
V_7	1	1	1

Table 2

Fuzzy Rules Developed for Direct Torque Control Technique

	θ_1			θ_2			θ_3		
E_ψ / E_{te}	P	Z	N	P	Z	N	P	Z	N
PL	V_1	V_2	V_2	V_2	V_2	V_3	V_2	V_3	V_3
PS	V_2	V_2	V_3	V_2	V_3	V_3	V_3	V_3	V_4
ZE	V_0	V_0	V_0	V_0	V_0	V_0	V_0	V_0	V_0
NL	V_6	V_0	V_4	V_6	V_0	V_5	V_1	V_0	V_5
NS	V_6	V_5	V_5	V_6	V_6	V_5	V_1	V_6	V_6

From the rules, fuzzy inference equations are given as

$$\alpha_i = \min\left(\mu A_i(E_\psi), \mu B_i(E_{te}), \mu C_i(\theta)\right), \tag{10}$$

$$\mu Ni'(n) = \min\left(\alpha i, \mu N_i(n)\right), \tag{11}$$

$$\mu N(n) = \max \sum_{i=1}^{180} \mu Ni'(n). \tag{12}$$

8 Simulation Procedures

A 1kW induction motor was used for simulation. The parameters of the machine were determined experimentally and are given in the Appendix. For the simulation of the viable torque control schemes, Voltage source inverter (VSI) was employed. The simulations were carried out using MATLAB / SIMULINK technical package described in [13, 14].

8.1 Direct Torque Neural Network Controller

The neural network is trained using the MATLAB neural-networks toolbox. This network consists of a three layer neural – network with three input nodes connected to five log sigmoid neurons and three pure output nodes connected to five log sigmoid neurons (3-5-3) shown in Fig. 5. The training strategy consists the parallel recursive error prediction was chosen as a learning technique for simulation purposes to update the weights of the neural network. The algorithm was chosen because of its learning speed, robustness and high learning capability. This algorithm is so powerful when complicated and nonlinear functions are to be learned by the neural network [9]. The neural network structure mentioned previously was simulated using this algorithm and using the hyperbolic tangent function

$$S(x) = \tanh\left(\frac{1}{2}cx\right) = \frac{1-e^{-cx}}{1+e^{-cx}}. \tag{13}$$

as the nonlinearity in the transfer functions of the hidden and output layers. The parameter *'c'* was fixed to one for all the cases. Small values of *'c'* are found to give larger weights and vice versa.

Simulation results were determined using an electromagnetic torque and stator flux commands of 2.5Nm and 0.85Wb respectively. The switching frequency of the inverter used by the simulations was 10kHz while the frequency of the neural network was 100Hz. The neural network frequency was chosen to give the plant enough time to stabilize its output. The data used to train the neural network have been determined by direct simulation of DTC using a sampling frequency of 100Hz.

8.2 DTC Using Genetic Algorithm

Neural network trained with genetic algorithm is implemented in such a way that the total number of thresholds and weights of the neural network be packed in n - dimensional vector 'w' as given in equation (14).

$$w = \left[th_1^{\,2}\, m_{11}^{\,2} \cdots m_{51}^{\,2} \cdots th_5^{\,1}\, m_{15}^{\,1} \cdots m_{35}^{\,1}\right], \tag{14}$$

where: *th* is threshold vector, *m* is weight vector and n = 38.

To represent the values of weights *w*, binary encoding or floating point encoding is used as a chromosome. Genetic operators used for binary representation are one point crossover, two-point crossover and bit mutation and for floating point representation are two point arithmetical crossover, uniform mutation, non-uniform mutation and non- uniform arithmetical mutation. **Table 3** shows the parameters used for simulation:

Table 3

Parameters used for Genetic Algorithm based DTC

Parameters used	Binary representation	Floating point representation
Number of chromosomes	30	100
Crossover probability	0.8	0.9
Mutation probability	0.005	0.008

In binary encoding algorithm, Lower number of chromosomes was used than floating point encoding algorithm. The performance of the system is affected if number of chromosomes reduced. To improve the performance and to overcome this drawback, the best member of each generation must be copied into the succeeding generation. Crossover probability can be chosen from 0.5 to 0.9. Convergence rate becomes slower with the higher crossover probability values. Convergence rate should be in high bias level. Mutation rate taken for simulation as shown in **Table 2** will make the convergence faster. In floating point-encoding algorithm, non-uniform mutation and non-uniform arithmetic mutation operators were introduced to prevent premature convergence. Fine tuning capabilities of genetic algorithm were achieved by using these operators and performance of the algorithm was also improved.

8.3 Direct Torque Fuzzy Logic Controller

Direct torque control of induction motor using fuzzy logic was also simulated using the MATLAB / SIMULINK package. Membership functions were chosen and simulations were carried out. Only for three flux angle positions, rules were given in **Table 2**. Similarly, rules could be formed for another nine flux angle positions and totally for twelve positions, rules were written and membership functions were formed. Simulations include all the possible rules and total number of rules found is 180.

9 Results and Discussions

As described earlier, 1kW induction motor was used for simulation and results were obtained. Switching frequency of the inverter taken for simulation

was 10KHz. There fore, the sampling time taken for simulation was 0.1ms. Torque and flux reference values taken were 2.5Nm and 0.5Wb when torque and flux hysteresis values are 0.5Nm and 0.02Wb respectively. Fig. 11 shows the actual torque developed in induction motor using conventional DTC. Referring to the Fig.11, torque rises from 0 to 2.5Nm in 10ms and then oscillates around the reference value in a narrow band.

9.1 DTC using Neural Network

The algorithm used to train the neural network is back propagation with momentum factor. The time taken to train the neural network using this algorithm is 2000s. The simulations that have been performed in this paper were obtained using a trained state selector neural network. The desired outputs are taken from the outputs of the conventional DTC. Thus, the training time is basically the time used in the simulation by the conventional DTC with the induction motor. All training algorithms were used to train the 3-5-3 neural-network structure using sigmoids. The torque and phase currents for the first half-second of simulation using a state selector neural network trained by the back propagation algorithm are shown in Fig. 12 and Fig. 13 respectively. The temperature coefficient of all the neurons was fixed to one, which gives reasonable weight magnitudes. An increase in the learning rate produces a faster learning, but a certain point it could become unstable, in the sense that the performance index begins oscillating around some local minimum, which make the weights not settle to their final values. A small learning rate is convenient even though it requires more training time in order to get a safety weights convergence.The results of the simulations given by back propagation are almost the same given by the conventional DTC, which shows that the neural network has been fully trained.

9.2 DTC using Genetic Algorithm

9.2.1 Binary representation

In binary representation, elitist strategy is used to fix the potential source of loss by copying the best member of each generation into the succeeding generation. The crossover rates of 0.5, 06, 0.7 and 0.9 in the problem are tried, the results show that convergence rate is slower with the high crossover rate, maximum fitness values never get as high as with the setting of 0.8. In addition, mutation rates of 0.1, 0.05, 0.01, 0.001 and 0.0001 in the problem are tried. Figs. 14 and 15 show the actual torque developed using DTC by neural network trained with genetic algorithm in which Fig. 14 represents binary coding. The results showed that the low mutation rate lead to poorer solutions but faster convergence. The higher mutation rate allows better solutions to be found, but it prohibits convergence to a high bias level. These results also showed that the GA procedure is not highly sensitive to parameter changes. Fig. 16 exhibits the

step function of the developed torque in induction motor using neural network trained with genetic algorithm using binary coding representation.

9.2.2 Floating point representation

In floating point representation, the genetic operators needs careful designing to preserve the constraint. There is no such problem in the binary representation, but the design of the operators is rather simple. In this paper, the property of convex space is used in designing the operators.This property indicates that for any two points w_i + wj (E [L, U]), the linear configuration a w_i +(1-a) w_j (E [L, U]), where a = (E [0,1]). If only ordinary crossovers are used for the resulting offspring, the premature convergence cannot be avoided since the population size is finite. Using the non-uniform arithmetical crossover, new points of population can be obtained which are much helpful to prevent premature convergence. Both NUM (Non Uniform Mutation) and NUAM (Non Uniform Arithmetical Mutation) are the operators responsible for the finite tuning capabilities of the genetic algorithm. These two operators initially search the space uniformly and then locally at later steps. It should be mentioned here that when using NUM some elements of the solution often lay on the boundary of the search space, this is not the case with using NUAM. Fig.15 represents the torque developed in induction motor with DTC using floating point coding representations.

9.2.3 Comparison of binary and floating point representations

From the detailed investigations, it is observed that the floating point representation provides a lot of advantages compared with the binary representation. It is capable of representing large domains, while the binary representation must sacrifice precision with an increase in domain size, given fixed binary length. The precision of the floating point representation depends on the underlying machine, but generally much better than that of the binary representation. In addition, in the floating point representation it is much easier to design special tools for handling non-trivial constraints [8]. The floating point representation may greatly improve a performance of genetic algorithms on numerical problems. Fig. 16 shows the locus of the stator flux and it is noticed that flux follows a circular shape. The components of stator fluxes in stationary reference frame are sinusoidal and 90º phase displacement to each other.

9.3 DTC using Fuzzy Logic Controller

In DTC using fuzzy logic, calculated flux error, torque error and flux angle are taken as inputs and switching states to the inverter are outputs. As described earlier, membership functions were chosen and rules were formed. Fuzzy logic controllers especially used in induction motor for low speed operation. At low speed operation, ripple contents are more. Here, fuzzy control is applied to minimize the ripple at low speed region of induction motor. Fig. 18 is the torque

developed by fuzzy controller and which is compared with the conventional DTC technique. From this result, it is observed that, at fuzzy logic DTC, torque easily attains steady state value at the earlier stage itself when compared to the conventional DTC technique. Initial stator flux rise at fuzzy logic control is shown in Fig.19. From this figure also, it is observed that, the time taken to reach the steady state value of flux is less using fuzzy logic DTC than the conventional DTC.

An index error has been used to quantify the error in both the stator flux and torque responses. This index is the integral of the square error (IE2), which is computed by means of the square error instead of just the error. Errors obtained in control schemes have been compared with each other. The error comparison is shown in **Table 4**.

Table 4

Errors obtained in various control strategies

Index Error (EI)		**Classical DTC**		**DTC_NN**	
$T=a*T_n$	**$\omega=b*\omega_n$**	**Flux**	**Torque**	**Flux**	**Torque**
a = 100%	b = 10%	$2.53\ 10^{-3}$	0.189	$2.2\ 10^{-3}$	0.165
a = 50%	b = 50%	$2.57\ 10^{-3}$	0.068	$0.53\ 10^{-3}$	0.025
a = 10%	b = 10%	$7.46\ 10^{-3}$	0.0367	$1.58\ 10^{-3}$	0.0014
a = 100%	b = 100%	$2.46\ 10^{-3}$	0.297	$2.1\ 10^{-3}$	0.263
DTC_NN_GA		**DTC_ Fuzzy**			
Flux	**Torque**	**Flux**	**Torque**		
$1.97\ 10^{-3}$	0.156	$2.74\ 10^{-3}$	0.169		
$0.68\ 10^{-3}$	0.023	$0.88\ 10^{-3}$	0.033		
$5.68\ 10^{-3}$	0.0015	$0.14\ 10^{-3}$	0.00135		
$2.33\ 10^{-3}$	0.31	$2.55\ 10^{-3}$	0.251		

T - Actual torque;T_n - Nominal torque = 5Nm;

ω - Actual motor speed; ω_n - Nominal motor speed = 1420r.p.m.

From the **Table 4**, it is realized that the index errors for flux and torque have been calculated for the different values of torque and speed in terms of their respective nominal values.

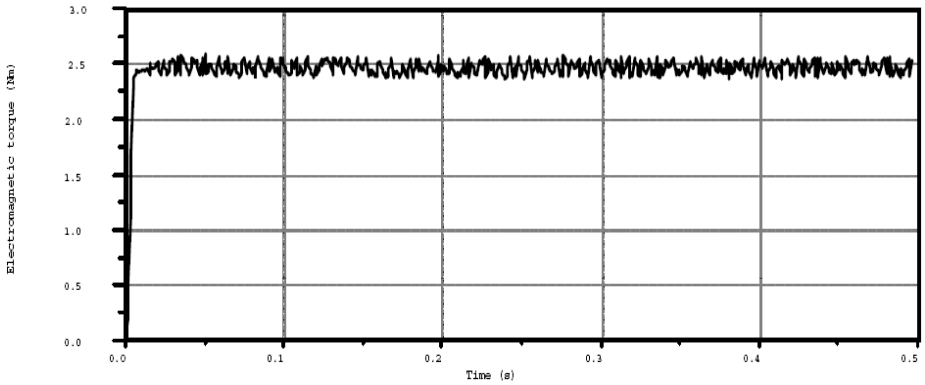

Fig. 11 – *Torque developed in conventional DTC.*

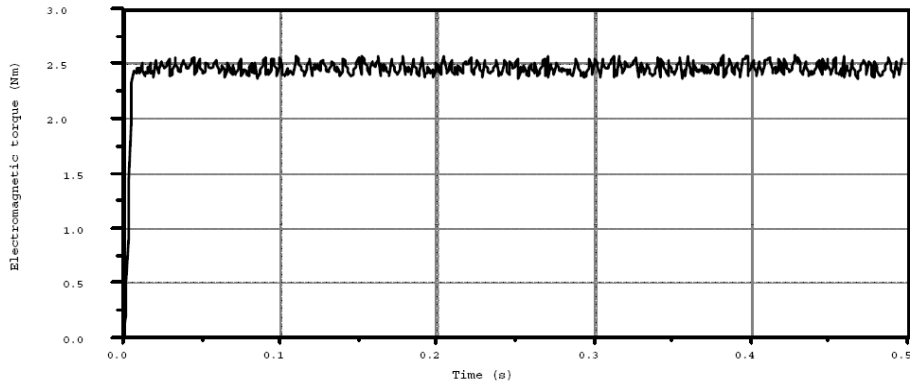

Fig. 12 – *Torque developed in DTC using neural network.*

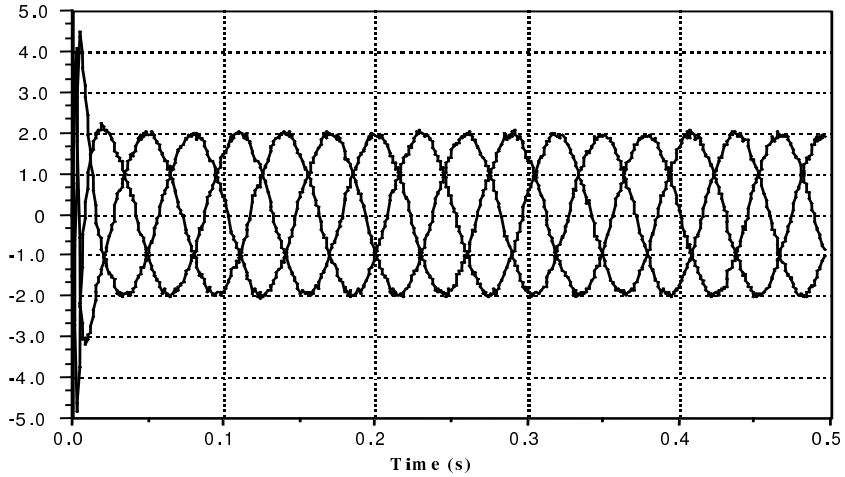

Fig. 13 – *Stator current in DTC using neural network.*

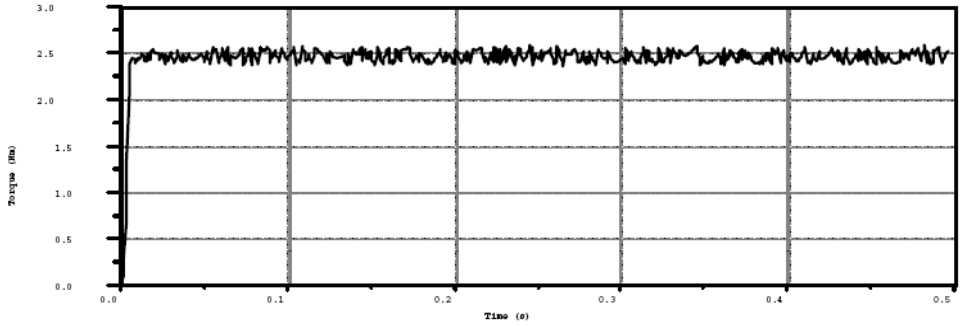

Fig. 14 – *Torque developed in DTC using neural network trained with genetic algorithm (Binary coding representation).*

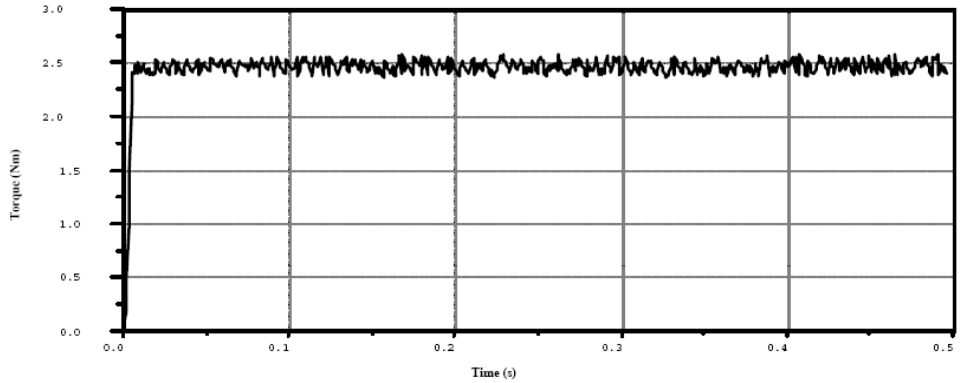

Fig. 15 – *Torque developed in DTC using neural network trained with genetic algorithm (Floating-point coding representation).*

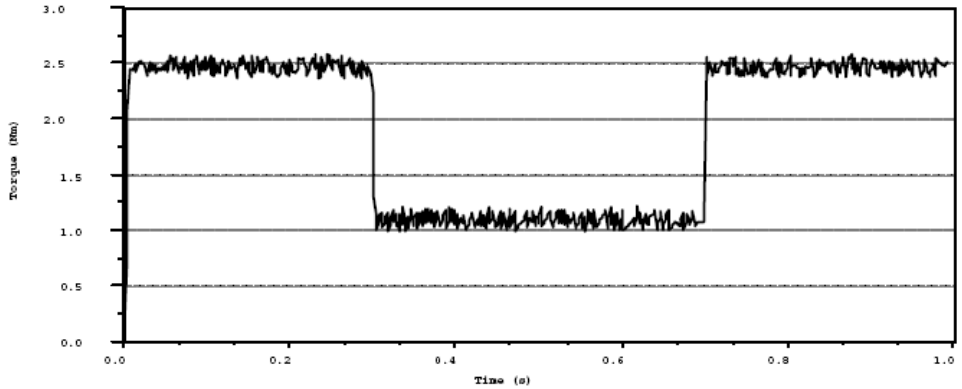

Fig. 16 – *Torque step response using genetic algorithm (Binary coding representation).*

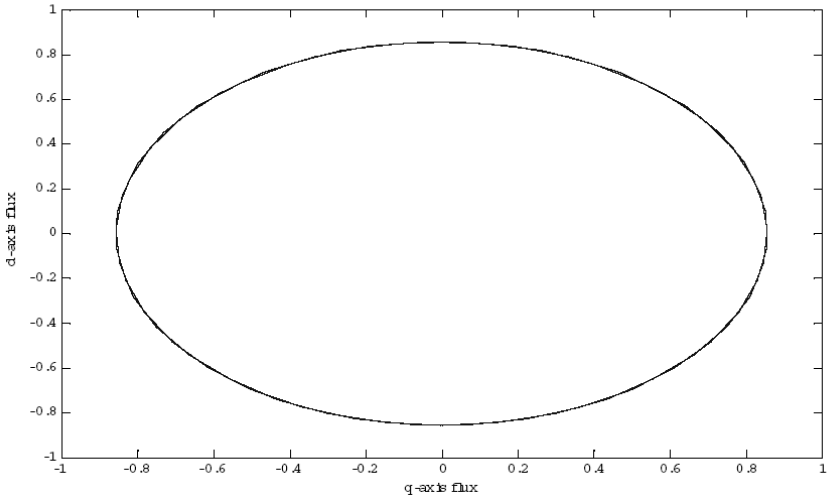

Fig. 17 – *Locus of the stator fluxes in the stationary reference frame.*

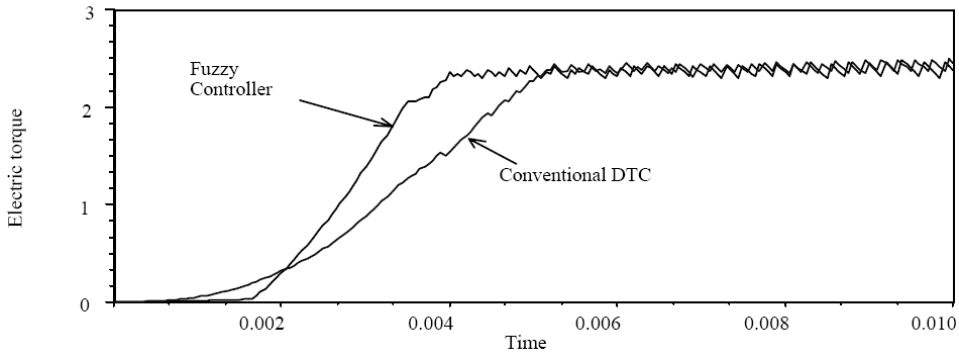

Fig. 18 – *Torque developed in conventional DTC and DTC using fuzzy logic.*

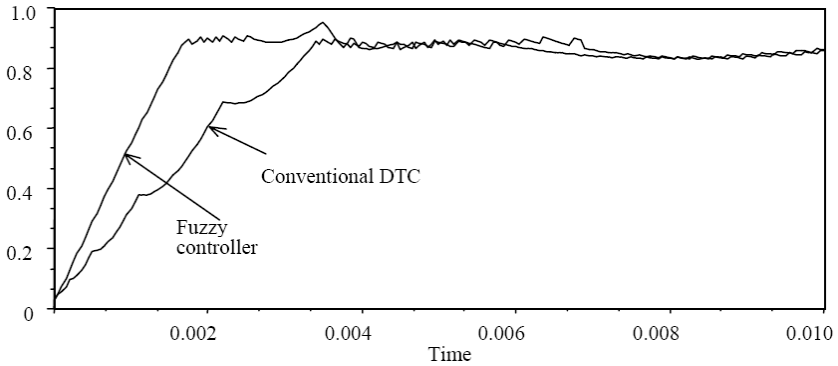

Fig. 19 – *Initial stator flux rise.*

10 Conclusion

Three different intelligent torque control schemes such as direct torque neuro controller, direct torque neuro controller trained with genetic algorithm and direct torque fuzzy controller have been evaluated for induction motor control and which have been compared with the conventional direct torque control technique.

Table 5
Features of Adaptive controllers

Sl.No.	Control Strategies	Advantages	Limitations
1.	DTC using Neural Network	1.Many training methods such as Back propagation algorithm, parallel recursive method, Kalman filter method and adaptive neuron model methods are available. 2.The results obtained are very close to conventional DTC.	1.Training time required is more. 2.Affected by parameters of the machine changes.
2.	DTC using Genetic Algorithm (Binary Representation)	1.It is not highly sensitive to parameter of the machine changes. 2.Gives precise results.	1.Accuracy is affected when domain size increases. 2.Difficult to design for handling non-trivial constraints.
3.	DTC using Genetic Algorithm (Floating point Representation)	1. It is also used to improve the performance on numerical problems. 2.Capable of representing quite large domains. 3.In this representation, it is easier to design special tools for handling non-trivial constraints.	1. In floating point representation, the genetic operators needs careful designing to preserve the constraint.
4.	DTC using Fuzzy Logic	1.Fuzzy logic does the resistance compensation in DTC at low speed region. 2. Provides more accuracy	1.Many rules are required to provide accuracy. 2.Computational time required is high.

Since the conventional DTC presents some disadvantages such as difficulties in torque and flux control at very low speed, high current and torque ripple, variable switching frequency behavior, high noise level at low speed and lack of direct current control, an adaptive torque controller must be proposed for high performance applications. In this paper, three various adaptive intelligent torque controllers have been proposed and results were compared. Among all these three adaptive controllers, genetic algorithm based direct torque neuro controller shows better response. By using this controller, parameters of induction motor are also be tuned and parameter variations are also be much reduced. When

compare to other adaptive controllers precise results have been obtained using genetic algorithm based direct torque neuro controller. The individual advantages and limitations of each scheme is presented in **Table 5**.

11 Appendix

Rating	1kW	Rr	8.38 Ω	ω_{nom}	1420 r.p.m.
P	4	L_m	0.7014 H	T_{nom}	6.7 Nm
Rs	7.23 Ω	L_s=L_r	0.0391 H	J	0.006 kgm^2

12 References

[1] B. J. A. Krose, P. P. van der Smagt: An Introduction to Neural Networks, The University of Amsterdam, Netherlands, Sept. 1991.

[2] P. Vas: Sensorless Vector and Direct Torque Control, Oxford University Press, 1998.

[3] The MATLAB compilers users guide in Math works handbook Math works 1994.

[4] J. William: Introduction to MATLAB 6 for Engineers, Palm III, McGraw – Hill International Edition, 2001.

[5] A. Ba-Razzouk, A. Cheriti, G. Olivier, P. Sicard: Field–Oriented Control of Induction Motors Using Neural – Network Decouplers, IEEE Transact. on Power Electronics,Vol. 12, No.4, pp.752-763, July 1997.

[6] B. K. Bose: Expert System, Fuzzy Logic, and Neural Network Applications in Power Electronics and Motion Control: Proceedings. IEEE, vol.82, pp. 1303-1323, Aug. 1994.

[7] C. Lascu, I. Boldea, F. Blaabjerg: A Modified Direct Torque Control for Induction Motor Sensor less Drive, IEEE Transactions on Industry Applications, Vol.36, No. 1, pp. 122-130, January / February 2000.

[8] G. Buja, D. Casadei, G. Serra: DTC – Based Strategies for Induction Motor Drives, IEEE-IAS Annual Meeting 2002, pp. 1506-1516.

[9] G. Griva, T. G. Habetler: Performance Evaluation of a Direct Torque Controlled Drive in the Continuous PWM-Square Wave Transition Region, IEEE Transactions on Power Electronics, Vol. 10, July 1995. pp. 464- 471.

[10] J. H. Lee, C. Kim, M. J. Youn: A Dead-Beat Type Digital Controller for the Direct Torque Control of an Induction Motor, IEEE Transactions on Power Electronics, Vol. 17, No.5, pp.739-746, September 2002.

[11] K.S. Narendra, K. Parthasarathy: Identification and Control of Dynamical Systems using Neural Networks, IEEE Transactions on Neural Networks, Vol. 1, pp. 4-27, March 1990.

[12] M. Elbuluk, X. Wang: Neural Network Control of Induction Machines using Genetic Algorithm Training, Annual Conference Record, IEEE IAS society ,2004.

[13] P. Z. Grawbowski, M. P. Kazmierkowski, B. K.Bose, F. Blaabjerg: A Simple Direct-Torque Neuro Fuzzy Control of PWM- Inverter- Fed Induction motor Drive, IEEE transactions on Industrial Electronics, Vol.47, pp.863-870., August 2000.

[14] T. G. Habetler, F. Profumo, M. Pastorelli, L. M. Tolbert: Direct Torque Control of Induction Machines Using Space Vector Modulation, IEEE Transactions on Industry Applications, Vol. 28, Sept/Oct 1992. pp. 1045- 1053.

[15] T. G. Habetler, F. Profumo, M. Pastorelli: Direct Torque Control of Induction Machines Over a Wide Speed Range, IEEE-IAS Annual Meeting, Conf. Rec. 1992, pp.600-606.