# Design of a Fuzzy Controller Using a Genetic Algorithm for Stator Flux Estimation

Mehmet KARAKOSE Mehmet KAYA Erhan AKIN

Firat University Department of Computer Engineering, 23119,Elazig, TURKEY
e-mail:{mkarakose,mekaya,eakin}@firat.edu.tr

**Abstract:** Performance of the field orientation in induction motors depends on the accurate estimation of the flux vector. The voltage model used for field orientation has in the flux calculation process an open integration problem, which is generally solved with a feedback loop. In this paper, a new method is developed for the feedback loop of the integrator. The method, as apart from studies in the literature, uses a fuzzy controller determined membership functions using a genetic algorithm (GA). For this purpose, a fuzzy controller is designed and tested on various motors of different power ratings. The proposed method is simulated by using MATLAB-SIMULINK and implemented on an experimental system using a TMS320C31 digital signal processor.

## 1 Introduction

The performance of the vector control is related to the accuracy of the resultant flux phase and magnitude information of the induction motor. There are two common methods to estimate the stator or rotor flux vector of the induction motor. These are called the current model and the voltage model. The current model does not contain an open integration, but requires the rotor parameters information and motor speed measurement during the flux calculation process. In this method, the flux vector is also estimated at standstill due to the lack of an open integration process. On the other hand, the voltage model is sensitive to stator parameters and requires voltage and current measurements, but it also does not require speed measurement. In sensorless vector control, the voltage model is most preferred. There are various methods of stator flux estimation using the voltage model. Rotor flux can also be estimated by using stator flux and motor parameters. The stator flux equations in the stator reference frame known as the voltage model are as follows:

$$\psi_{s\alpha} = \int (u_{s\alpha} - i_{s\alpha} R_s)\, dt \tag{1a}$$

$$\psi_{s\beta} = \int (u_{s\beta} - i_{s\beta} R_s)\, dt \tag{1b}$$

By using Equation 1a-1b, $\psi_{s\alpha}$ and $\psi_{s\beta}$ can be estimated. The angle of the stator flux orientation can be calculated by using these equations. This angle is used for axes transformation between $\alpha\beta$ *to* $dq$ and called as transformation angle $\theta_s$.

$$\theta_s = arctg \frac{\psi_{s\beta}}{\psi_{s\alpha}} \tag{2}$$

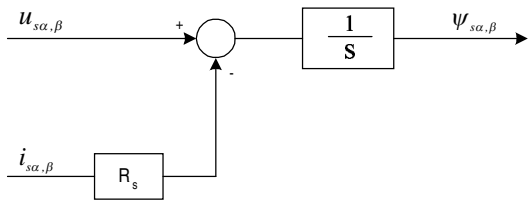



**Fig. 1.** Block diagram of the voltage model

The block diagram in Fig. 1 shows the flux calculation process described in Equations 1a-1b. In this process, the integrator has no feedback. This open integration process causes drift in the integration. This integration process can be analog or digital. Digital flux integration has integration method and integration step size, errors in the current and voltage measurements, variations in the stator resistance due to temperature and

V.N. Alexandrov et al. (Eds.): ICCS 2001, LNCS 2074, pp. 272–280, 2001.

frequency of the current, initial value of the integrator, error introduced by the finite bit length of the processor, execution time of the commands etc.

It is clear from the above open integrator that the performance of the voltage model depends on the stator resistance. This resistance gains further importance in low-speed operation where the voltage drop on this resistance becomes significant. At zero speed, it is not possible to estimate the stator flux by using this algorithm.

Numerous papers exist in the literature on the integration problem, especially sensorless type direct vector-controlled drives. For example, Ohtani [1] proposed to add a feedback to the integration algorithm to solve the problem. In Ohtani's study, a steady-state analysis of the issue is presented. In [2] another approach is studied. This is based on two-phase voltage and current measurements, and stability analysis is made under dynamic conditions. In this study, flux magnitude and flux derivative are used in the feedback loop. However, the gain of the flux derivative feedback is a function of the load and speed. Therefore, the performance of the system depends on the correct value of these feedback gains. A study by Patel [3] employs a cascade connected and automatically adjusted low-pass filter instead of the integrator. An algorithm proposed by Akın [4] eliminates the DC component of the integrator output, but it only works in the steady state.

Some of the algorithms mentioned here are a $\delta$ feedback integration algorithm to solve the stability problem of the integrator and a PI feedback integration algorithm. The block diagram and transfer function of the $\delta$ feedback integrator are given in Fig. 2 and Equation 3 respectively.

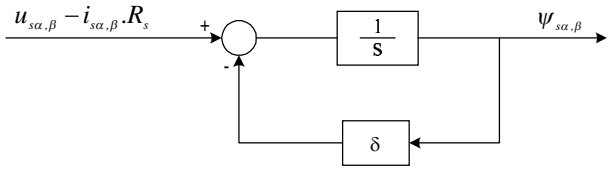



**Fig. 2.** Block diagram of δ feedback integrator

$$F(s) = \frac{1}{s+\delta} \tag{3}$$

In this integrator, the $\delta$ feedback path is used for stability of the integrator. When the frequency is high *(jw>>δ),* the integrator function behaves well. However, when *jw* becomes comparable to δ, both magnitude and phase errors appear and affect the performance both at steady state and in transient conditions. The $\delta$ feedback integrator, shown in Fig. 2, eliminates offset substantially. However, integration output still involves phase shift and error in magnitude [5].

In this study, a fuzzy controller is developed for the feedback loop of the integrator. This fuzzy controller has the advantages of robustness, ease of design and good transient response.

One of the main problems of a fuzzy controller design is the process of determining membership functions. Usually such a determination is obtained from human experts. However, the approach adopted for acquiring the shape of any particular membership function often depends on the application. For most fuzzy logic control problems, the membership functions are assumed to be linear and usually triangular in shape. So, typically the sets that describe various factors of importance in the application and the issues to be determined are the parameters that define the triangles. These parameters are usually based on the control engineer's experience and/or are generated automatically. However, for many other applications, triangular membership functions are not appropriate as they do not represent accurately the linguistic terms being modelled, and the shape of the membership functions have to be elicited directly from the expert, by a statistical approach or by automatic generation of the shapes.

GA was employed first by Karr [6] in determination of membership functions. Karr has applied GA to the design of a fuzzy logic controller (FLC) for the cart pole problem. Meredith [7] has applied GA to the fine tuning of membership functions in a FLC for a helicopter. Initial guesses for the membership functions are made by a control engineer, and the GA adjusts the defining parameters by through use in order to minimize the movement of a hovering helicopter. Again, triangular membership functions are used. Lee and Takagi [8] have also tackled the cart problem. They have taken a holistic approach by using GA to design the whole system determining the optimal number of rules as well as the membership function, which are again triangular. Recently, Arslan and Kaya [9] have proposed a new method for determination of fuzzy logic membership functions using GAs.

## 2 A Survey of Fuzzy Controller and Genetic Algorithms

Fuzzy logic is a technology based on engineering experience and observations. In fuzzy logic, an exact mathematical model is not necessary, because linguistic variables are used in fuzzy logic to define system behavior rapidly. Fuzzy logic is a very recent technology relative to conventional controllers; its areas of application are increasing very quickly. Fuzzy PID, fuzzy PI, fuzzy PD and fuzzy mixed controllers are fuzzy controller design approaches, but unlike conventional controllers the focus is not in the modeling [10].

Some of the problems, such as stability and performance, are encountered both in fuzzy controllers and conventional controllers. Unlike conventional control design, where mathematical models are used to solve these problems, fuzzy controller design involves IF-THEN rules defined by an expert to tackle these problems.

There are two methods that are commonly used to design fuzzy controllers: trial and error method and the theoretical method. In trial and error, IF-THEN rules are defined by using expert knowledge and experience. Then, these rules are applied to the actual system. Unlike the theoretical approach where the parameters are adjusted to guarantee the desired performance, in the fuzzy method the IF-THEN rules are modified until the desired performance is achieved. In practice, both methods can be used to obtain better performance [11].

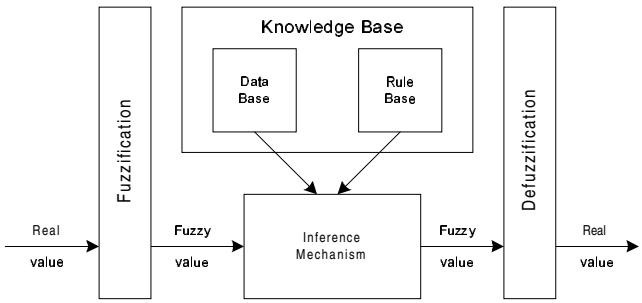



**Fig. 3.** Block diagram of fuzzy control architecture

The fuzzy controller has four components as shown in Fig. 3. These are:

a- Fuzzifier: The input values are scaled and grouped into fuzzy sets. In other words, the input values labeled and transformed into linguistic variables.

b- Inference mechanism: The inference mechanism uses a database and a rule base. The database involves membership functions that are used by the inference mechanism to make fuzzy decisions.

b- Rule Base: Rule base is a set of IF-THEN rules defined by an expert. The inference mechanism uses these rules.

c- Defuzzifier: The linguistic variables manipulated by the inference mechanism are converted back to real values.

In a fuzzy controller design, the knowledge and observations of an expert are more important than the underlying mathematical model. This expert knowledge and observation is used while the system is being designed. This kind of approach provides an opportunity to easily embed experience into a controller, which has been gained over a long time. However, it is not possible to obtain automation during controller design.

A GA is an iterative procedure that consists of a constant-size population of individuals, each one represented by a finite string of symbols, known as the genome, encoding a possible solution in a given problem space. This space, referred to as the search space, comprises all possible solutions to the problem at hand. Generally speaking, the GA is applied to spaces that are too large to be exhaustively searched. The standard GA proceeds as follows: an initial population of individuals is generated at random or heuristically. Every evolutionary step, known as a generation, the individuals in the current population are decoded and evaluated according to some predefined quality criterion, referred to as the fitness function. To form a new population individuals are selected according to their fitness. Many selection procedures are currently in use one of the simplest being Holland's original fitness-proportionate selection [12]. Selection alone cannot introduce any new individuals into the population. These are generated by genetically inspired operators, of which the best known are crossover and mutation.

## 3 Design of a Fuzzy Controller Using Genetic Algorithms

In this paper, a fuzzy controller is used in the feedback loop of the integrator in the voltage model as shown in Fig. 4. The proposed fuzzy controller is based on rules and can be adopted to different machines easily. The membership functions of the fuzzy controller used are determined using GAs. Unlike conventional controllers, fuzzy controllers are less sensitive to sensor errors and small variations of the parameters.

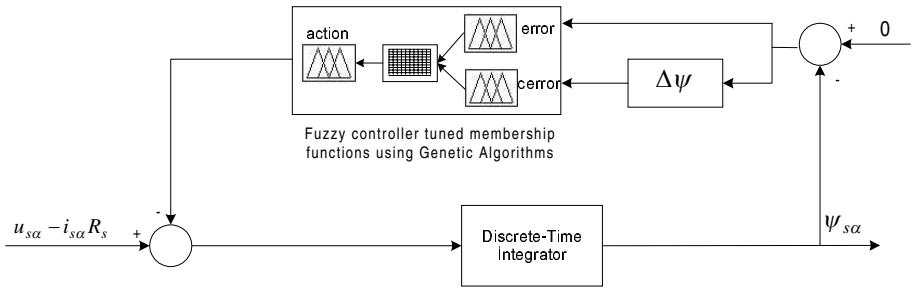
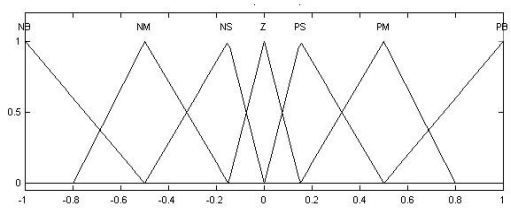
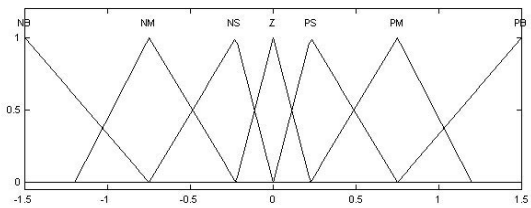



**Fig. 4.** Block diagram of the fuzzy controller for stator flux estimation

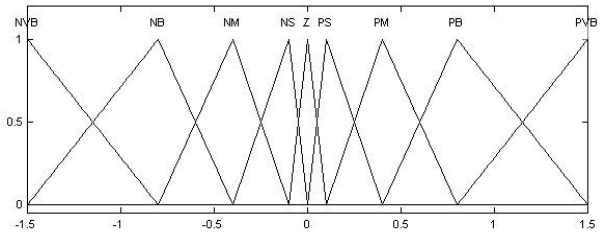
As shown in Fig. 4, the core of the estimation of the stator flux is the discrete integration of the difference between $(u_{s\alpha} - i_{s\alpha}.R_s)$ and the feedback signal. $R_s$ is assumed constant during the simulation. In this figure, first the flux is compared to zero reference in the feedback loop. Next, this difference and the derivative of the difference are given as inputs to the fuzzy logic controller tuned membership functions using GAs. Each variable of the fuzzy controller is represented by using 7 membership functions at the input, as shown in Fig. 5a-5b, and 9 membership functions at the output in Fig. 5c. Initially, the base values and intersection points are chosen randomly. The ranges of the input and output variables are assumed to be [-1,1], [-1.5, 1.5] and [-1.5, 1.5], respectively. The fuzzy rule base which resulted in the most efficient process for this fuzzy controller is as shown in Fig. 5d. For better precision, the number of membership functions can be increased at the expense of computational cost.




**(5a)**




**(5b)**




**(5c)**

| | | **error** | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | **NB** | **NM** | **NS** | **Z** | **PS** | **PM** | **PB** |
| **cerror** | **NB** | NVB | NVB | NVB | NB | NM | NS | Z |
| | **NM** | NVB | NVB | NB | NM | NS | Z | PS |
| | **NS** | NVB | NB | NS | NS | Z | PS | PM |
| | **Z** | NB | NM | NS | Z | PS | PM | PB |
| | **PS** | NM | NS | Z | PS | PM | PB | PVB |
| | **PM** | NS | Z | PS | PM | PB | PVB | PVB |
| | **PB** | Z | PS | PM | PB | PVB | PVB | PVB |

**(5d)**

**Fig. 5.** Membership functions, rule table and surface viewer of fuzzy controller **(a).** Initial membership functions of input variable "*error*" **(b).** Initial membership functions of input variable "*cerror*" **(c).** Initial membership functions of output variable "*action*" **(d).** Rule table of fuzzy controller.

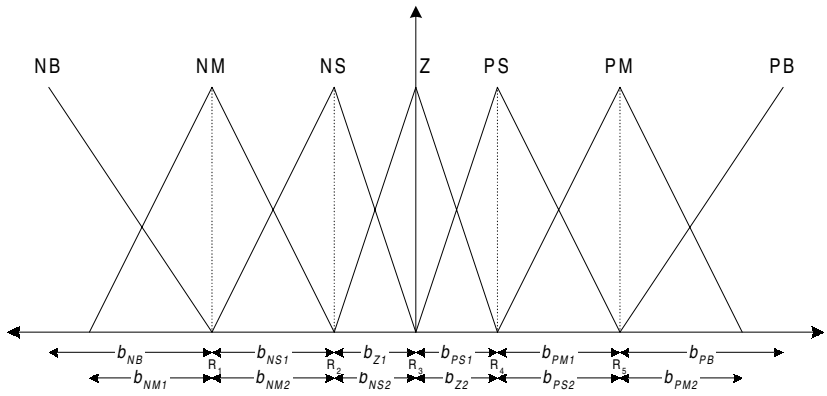



**Fig. 6.** The base lengths of the membership functions for input variable *"error"*

The goal expected from the GA is to find the base lengths and intersection points of triangles corresponding to the input data. Each base length has minimum and maximum values available. For example, the GA searches base value $b_{NB}$ between the minimum value of *error* (i.e. -1) and the maximum value of *ac* (i.e. 1). The search intervals of some the base values and intersection points for variable *error* are as follows:

$b_{NB}$: -1, 1 $(min(error) - max(error))$
$R_1$: -1, 1 $(min(error) - max(error))$
$b_{NM1}$: -1, $R_1$ $(min(error) - R_1)$
$b_{NM2}$: $R_1$, 1 $(R_1 - max(error))$

These base values and intersection points must be reflected in the definite range of the system, because their values depend on bit length. This is formulated as follows.

$$b = b_{min} + \frac{d}{(2^L - 1)}(b_{max} - b_{min}) \tag{4}$$

where d is the decimal value of a gene, L is the length of this gene, $b_{min}$ is the minimum value of the area reflected, and $b_{max}$ is the maximum value of that area.

Two different gene pools are created in the GA process. While one of these pools codes the base lengths and intersection points of the input variables for the fuzzy system, the other pool does that of the output. Each chromosome in a gene pool consists of the values of the bases and the intersection points. The length of chromosomes in each pool depends on the definite ranges of the input and output variables. For example, the chromosome encoding the base lengths and the intersection points for the input variables of the fuzzy system consists of genes in the form $b_{NB}b_{NM1}R_1b_{NM2}....b_{PM1}R_5b_{PM2}b_{PB}$.

The bit length of each gene may be different. In this study, when the bit length is chosen care has been taken that the sensitivity should be between 0.2 and 0.3. For example, because the range is –1, 1, if the gene of $b_{NB}$ is represented by 3bits, a sensitivity of 0.28 is achieved. $\left(\frac{1}{2^3 - 1} * (1 - (-1)) = 0.28\right)$

This value is equal to the decimal value of a change in the smallest bit of gene $b_{NB}$.
The GA process in each pool, therefore, includes the following steps:

(i) Specify string length $l_s$ and population size N,
(ii) Evaluate each chromosome with respect to the fitness function,
(iii) Perform selection, crossover and mutation,
(iv) If not (end-test) go to step (ii), otherwise stop and return the best chromosome.

The fitness function of this procedure is calculated as follows:

$$Fitness\ function = Max.\ error - Total\ error \tag{5}$$

The maximum error is made large enough to prevent the value of the fitness function from being negative. The maximum error is found as follows:

$$\sum_{i=1}^{9}\left(action_{GA_i} - 1.5\right)^2 \tag{6}$$

From the equation above, the maximum error is equal to 20.25. Total error is calculated as follows:

$$\sum_{i=1}^{9}\left(action_i - action_{GA_i}\right)^2 \tag{7}$$

where $action_{GA_i}$ is the output found by GA in current cycle and $action_i$ is the output obtained in previous cycle.

To find the desired results, first of all, corresponding membership functions are found for the i[th] values of the input variables. Then, it is determined whether or not these two membership functions may be put in a rule. If there is a rule between these two membership functions, the grades of the membership functions are calculated and analysed to determine if the rule contains AND or OR. If these two membership functions are ANDed, outputs are determined from lower grades of membership functions, but if two membership functions are ORed, outputs are determined from higher grades of membership functions.

If there is more than one membership function intersecting with the i[th] input of any input variable, then the outputs of these membership functions are both evaluated, and the one which has less error is used.

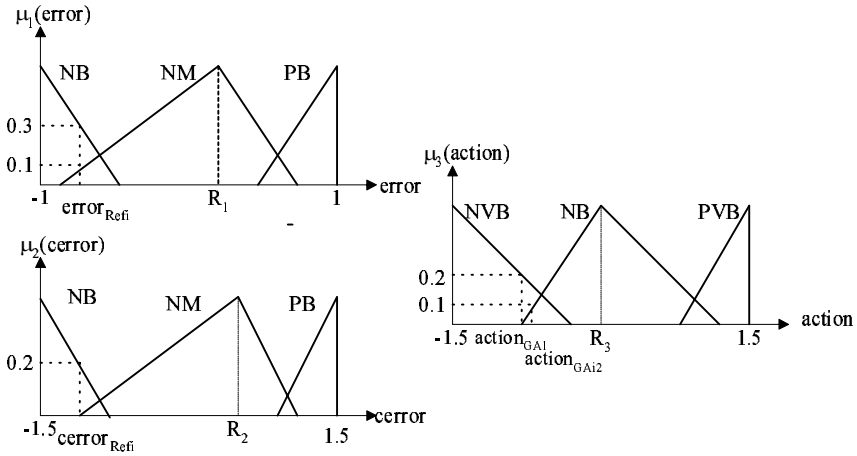



**Fig. 7.** The method of finding appropriate outputs of GA for the i[th] inputs

The i[th] inputs intersect with *NB* and *NM* for *error* and with *NB* for *dd*. After obtaining the intersection situation from Fig. 4, grades of membership are determined for each membership function. Then, from the rule bases that were obtained from both *small* and *slightly dirty*, the outputs are calculated from *short* and *warm*. While doing calculations, $\mu_2$ is taken as 0.2, because the rule involves AND ($\mu_2 < \mu_1$). Since there is also a rule between *medium* and *slightly dirty,* the outputs are calculated for $\mu_1$=0.1. The outputs are calculated from both rules, and the one which has less error compared to the desired output is used. This situation is depicted for 3 membership functions in Fig. 7. In other words,

$$if\ [(\ action_i - action_{GA_{i1}}\ )^2 < (\ action_i - action_{GA_{i2}}\ )^2\ ]\ then$$

$$action_{GA_i} = action_{GA_{i1}} \quad else \quad action_{GA_i} = wt_{GA_{i2}}$$

Two important points should be noted here. First, if the intersection between membership function and reference input occurs on the left-hand side of the intersection point ($R_1$) in a non-right triangle, and the output occurs in the range of non-right triangle for the rule, then the output is also taken from the left-hand side of the intersection point. The same rule is valid for the right hand side as well. The second important point is that if there is no rule between any two membership functions for input variables. However, The rule base used here has complete rules.

## 4 Experimental Setup

To verify the proposed compensation algorithm, an experimental setup with an induction motor drive has been constructed. Fig. 8 shows the block diagram of the drive system where a conventional field oriented control is implemented. An IGBT inverter, which is controlled by a digital hysteresis control algorithm, is used

to drive the motor. The controller board is a DS1102 from dSPACE GmbH. The processor on the controller board is a Texas Instruments TMS320C31 32-bit floating-point processor with a 60 ns instruction cycle. The DS1102 is also equipped with a four channel analog-to-digital converter (two 16-bit and two 12-bit channels), a four channel 12-bit DAC and two incremental encoders.

In this experimental setup, LEM sensors are used to measure two-phase currents. The voltage information is obtained from the inverter switching position including dead time effects. The vector control algorithm has an execution cycle time of 35 μs. The new algorithm with the fuzzy controller consumes an additional 80 μs of execution time. In this experimental setup the speed control is implemented by using speed estimation. However, the speed estimation algorithm has been checked with encoder output for confirmation.

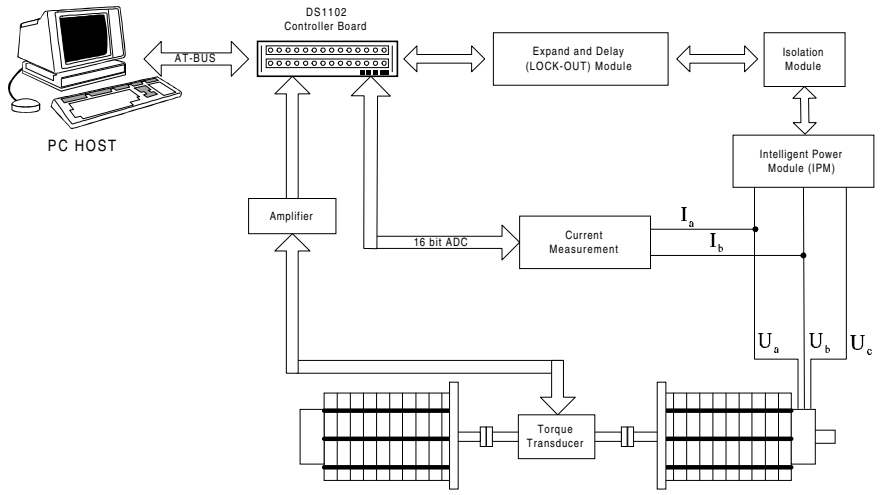



**Fig. 8.** Block diagram of the experimental setup

## 5 Simulation and Experimental Results

Various simulations were carried out by using MATLAB/SIMULINK to assess the performance of the integrator with a fuzzy controller on the feedback. The fuzzy controller used for estimating the flux is developed by the MATLAB Fuzzy Toolbox. Simulations are performed to investigate transient state and steady state performance of the proposed flux estimator.

In the simulations, we tested the fuzzy controller by using two induction motors of different power rating. The stator flux waveform of the fuzzy-controlled for a 3-HP motor is given in Fig. 9. In Fig. 10 experimental result of the fuzzy controller is given. The difference between these results is negligible. The same simulations mentioned above are also performed for a 500-HP motor as shown Fig. 11.

Experimental results mentioned above are obtained by TRACE31 software. Experiments and simulations are also performed in different torque reference and load conditions for the three motors used in the previous simulations. In these simulations the fuzzy controller performed better than other algorithm in most of the torque load conditions.

The membership functions of the fuzzy controller are determined by using GA in offline run. Although the GA decreases a little the performance of the system, the user's effect disappears on the system.

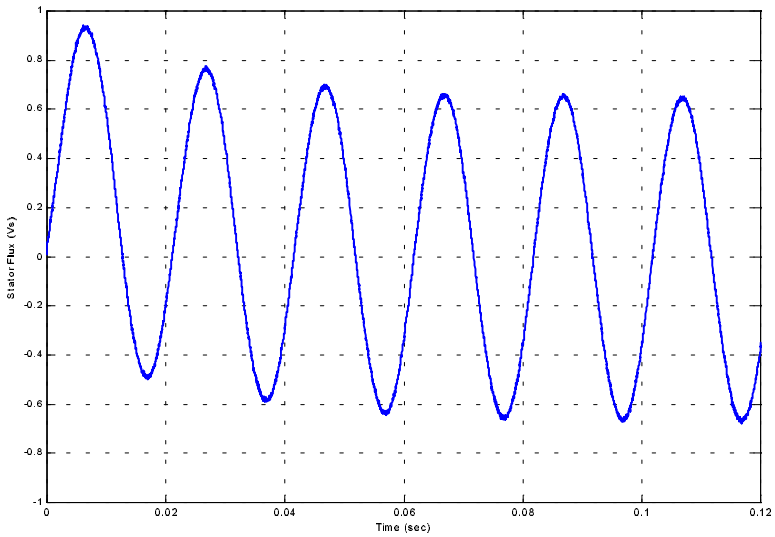



**Fig. 9.** Simulation results of the fuzzy controller for 3 hp motor.

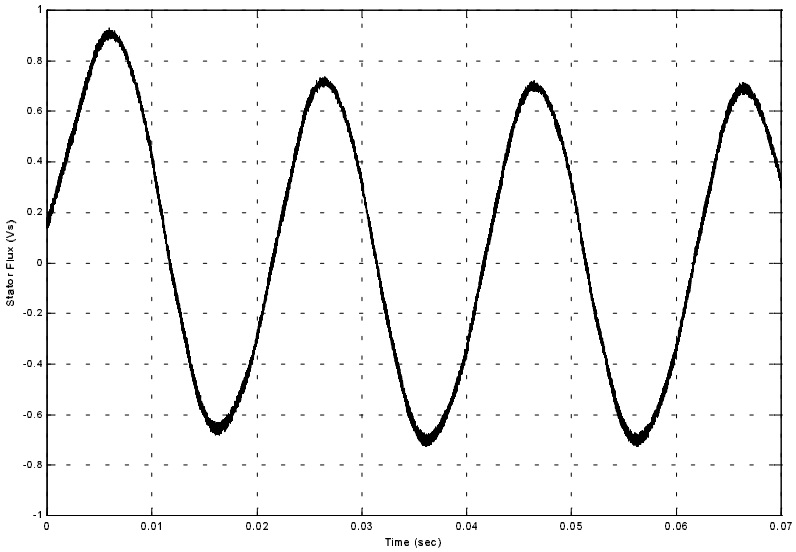



**Fig. 10.** Experimental results of the fuzzy controller for 3 hp motor.

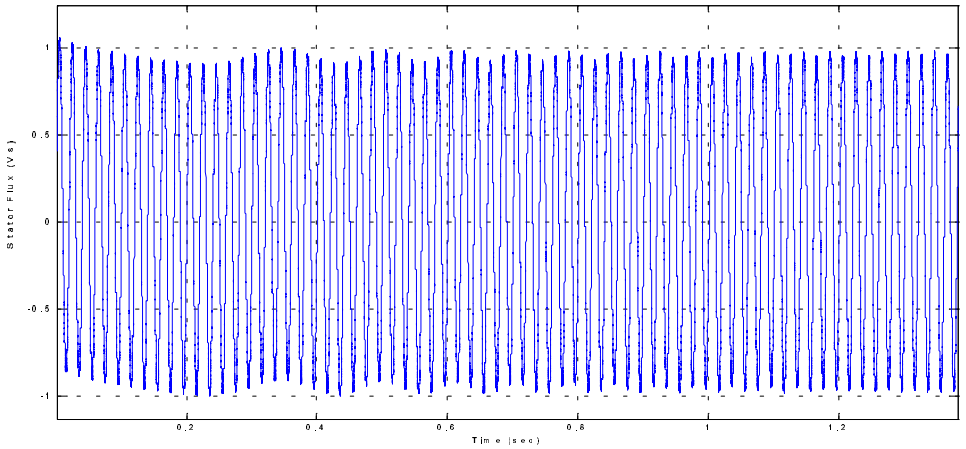



**Fig. 11.** Simulation results of the fuzzy controller for 500 hp motor.

## 6 Conclusions

In this paper a digital integrator employing a fuzzy controller feedback is presented to calculate the stator flux vector. The membership functions of the fuzzy controller have been determined by using GA. Implementation of the proposed algorithm has been performed using a TMS320C31 processor. The most important advantage of this new system is to provide a robust structure and simple design. Moreover, the proposed fuzzy controller method has a shorter settling time then other integration methods.

## References

1. T. Ohtani, N. Takada, K. Tanaka, "Vector Control of Induction Motor without Shaft Encoder", IEEE Transactions on Industry Applications, vol. 28, no. 1, pp. 157-164, January/February, 1992.
2. Jos van der Burgt, The Voltage/Current Model in Field-Oriented AC Drives at Very Low Flux Frequencies, PhD Thesis, 1996.
3. B.K. Bose, N.R. Patel, "A Programmable Cascaded Low-Pass Filter-Based Flux Synthesis for a Stator Flux-Oriented Vector-Controlled Induction Motor Drive", IEEE Transactions on Industrial Electronics, vol. 44, no. 1, pp. 140-143, February, 1997.
4. Akın E., A New Method for Rotor Flux Orientation of Induction Motor via Stator Fluxes, Firat University, PhD Thesis, 1994.
5. E. Akın, H. Can, H.B. Ertan, Y. Üçtuğ, "Comparison of Integration Algoritms for Vector Control", ICEM98, Vol. 3, pp. 1626-1631, September, 2-4, 1998, İstanbul, Turkey.
6. C. L. Karr, “Design of an Adaptive Fuzzy Controller Using a Genetic Algorithm Proc. of the 4th Intl. Conf. on Genetic Algorithms”, 1991.
7. D. L. Meredith, C. L. Karr and K. Krishna Kamur, “The use of genetic algorithms in the design of fuzzy logic controllers”, 3rd workshop on Neural Network WNN’92, 1992.
8. M. A. Lee and H. Takagi, “Integrating Design Stages of Fuzzy Systems Using Genetic Algorithms”, Second IEEE Intl. Conference on Fuzzy Systems, 1993.
9. A. Arslan and M. Kaya, “Determination of fuzzy logic membership functions using genetic algorithms”, Fuzzy Sets and Systems, vol:118 no:2, pp:297-306, 2001.
10. B. Hu, G.K.I. Mann, R.G. Gosine, "New Methodology for Analytical and Optimal Design of Fuzzy PID Controllers", IEEE Trans. On Fuzzy Systems, vol. 7, no. 5, 521-539, October 1999.
11. K.M. Passino, S. Yurkovich, Fuzzy Control, Addison-Wesley, 1998.
12. J. H. Holland, “Adaptation in Natural and Artificial Systems” Ann Arbor, MI: Univ. Mich. Press, 1975.
13. F. Herrara, M. Lozano and J. L. Verdegay, “Tuning fuzzy logic controllers by genetic algorithms”, International Journal of Approximate Reasoning, 12(3/):299-315, 1995.