Journal of Engineering Science and Technology
Vol. 1, No. 1 (2006) 59-75

ADAPTIVE FUZZY LOGIC SPEED CONTROLLER WITH TORQUE ADAPTED GAINS FUNCTION FOR PMSM DRIVE

MUTASIM NOUR, SHIREEN Y.M. TOO
School of Electrical and Electronic Engineering, Faculty of Engineering,
The University of Nottingham Malaysia Campus
43500 Semenyih, Malaysia
Email: mutasim.nour@nottingham.edu.my

Abstract

This paper presents a 15 rules-base Function Torque Adapted Gain Fuzzy Inference System (FTAGFIS) adaptive speed controller for the Permanent Magnet Synchronous Motor (PMSM). The proposed controller was developed using Adaptive Neuro Fuzzy Inference System (ANFIS). This expanded version of the FIS not only suggests the effectiveness of using the ANFIS to develop an adaptive speed controller for motors but also the effectiveness of the methodology that was employed to train the FIS. The simulation results of the developed FTAGFIS show superior speed tracking and recovery performance from motor parameter variations, load torque and speed disturbances, which indicates the ability of the PMSM to self- adapt itself to different operating conditions. The simulations were carried out in the MATLAB® environment.

Keywords: PMSM, Adaptive Speed Controller, Adapted Gains, ANFIS, Neural Network, Fuzzy Logic Control.

1. Introduction

Drive systems are widely used in industrial applications such as machine tools, servo, robots, aerials, x-y tables, computer equipments, textile machines, electric vehicles

Nomenclatures

v_q	Quadrature axis voltage [V]
v_d	Direct axis voltage [V]
L_q	Quadrature axis inductance [H]
L_d	Direct axis inductance [H]
i_q	Quadrature axis stator current [A]
i_d	Direct axis stator current [A]
R	Stator resistance per phase [Ohm]
Ψ_f	Rotor flux linkage [Wb]
ω_m	Mechanical angular rotor speed [rad/s]
P	Number of pole pairs of the motor [ ]
T_e	Developed electromagnetic torque [N.m]
T_L	Load torque applied to the motor [N.m]
J_m	Moment of inertia [kg.m^2]
B_m	Friction coefficient [N.m/rad/s]
K_b	Induced emf constant [V.s/rad]
k_t	Torque constant [N.m/A)

and ship propulsion [1]. These industrial drive applications are generally classified into constant- speed and variable- speed drives. Although a majority of the current variable speed drive applications use dc motors, they are progressively being replaced by ac motor drives, particularly by the PMSM. This is because the construction of the PMSM incorporates high energy rare-earth alloys, which are responsible for the motor's advantageous features such as high torque to current ratio, large power to weight (or volume) ratio, high efficiency, high power factor and robustness [2- 4]. It is due to these very features that the PMSM is fast becoming popular in high performance applications as compared to other types of ac motor drives [2- 4].

However, high performance motor drives require fast and accurate speed response, quick recovery of speed from any disturbances and sensitivity to motor parameter variations [2, 4]. Motor parameters change due to saturation and/or heating that occur during running [5]. Besides that, the motor is subjected to speed and load torque disturbance effects that occur during running. This causes the motor's actual response to deviate from its intended (or desired) response. Though the conventional PI controllers have been widely utilized as speed controllers in the PMSM drives, the fixed gains of these type of controllers is responsible for making the high performance drive systems very sensitive to motor parameter variations, load torque and speed disturbances [1, 2, 6, 7]. For the motor drive to be capable of continuously running at its desired response under different operating conditions (speed command tracking, speed command and load torque disturbances, parameter variation), its speed controller has to be capable of self-tuning, or adapting, its gains automatically online in response to these different operating conditions. These types of controllers are known as adaptive controllers. The core function of an adaptive speed controller is to

drive the speed error between the command and actual speed of the motor to zero under any load disturbances or parameters variations.

There are many types of adaptive control techniques that exist today to assist control designers to develop adaptive speed controllers. Among them are the plant model- based Model Referencing Adaptive Control (MRAC) and Sliding Mode Control techniques as well as AI-based techniques such as Fuzzy and Neural Control [1]. Recent literature has paid much attention to the potential of fuzzy control in machine drive applications [5-6]. This is because it has the advantages of providing robust performance for both linear and nonlinear plant functions, and convenience as it does not require knowledge of the plant's mathematical model [1-2, 13]. However, the qualitative design of the fuzzy logic controller (FLC) is entirely heuristic, and thus difficult to obtain a systematic design as it is based on one's experiences and expert knowledge about the process being controlled [5]. Besides that, its input and output scaling gains are determined by trial and error, and has to be varied to tune the FLC for the desired performance, which altogether makes its design a time- consuming task [4-5]. To make the FLC self-adapting towards varying operating conditions, papers such as [6] and [13] have proposed that an additional FLC be included into the control algorithm. This entails more rules and instructions, and thus requiring more memory and time to execute.

This paper presents an adaptive FLC for the PMSM that is capable of controlling the speed of the motor to track any arbitrary selected reference position and speed despite motor parameter variations, load torque and speed disturbances. The design of the proposed controller incorporates a function known as FTAG that automatically adjusts the output gain of the FLC according to the load applied onto the motor. The paper is organized as follows: in Section 2, the motor dynamics of the PMSM is reviewed; in Section 3, the concept of the ANFIS technique is briefly explained; in Section 4, the development process of FTAG FIS is presented; in Section 5, a speed control performance comparison is made between developed FTAG FIS and the conventional PI Controller; and finally, in Section 6, concluding remarks end the paper.

2. Motor Dynamics of the PMSM

Upon employing the vector control scheme, the dynamic model of the PMSM can be represented mathematically by the following equations in the d-q axis synchronously rotating rotor reference frame (under sinusoidal stator excitation) as [2]

$$\begin{bmatrix} v_q \\ v_d \end{bmatrix} = \begin{bmatrix} R + \dfrac{dL_q}{dt} & P\omega_m L_d \\ -P\omega_m L_q & R + \dfrac{dL_d}{dt} \end{bmatrix} \begin{bmatrix} i_q \\ i_d \end{bmatrix} + \begin{bmatrix} P\omega_r \psi_f \\ 0 \end{bmatrix} \tag{1}$$

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

$$T_e = J_m \frac{d\omega_m}{dt} + B_m \omega_m + T_L \tag{3}$$

where v_q and v_d are the d, q axis voltages, L_d and L_q are the d, q axis inductances, i_d and i_q are the d, q axis stator currents, respectively; R is the stator resistance per phase, Ψ_f is the rotor flux linkage, ω_m is the mechanical angular rotor speed, P is the number of pole pairs of the motor, T_e is the developed electromagnetic torque, T_L is the load torque applied to the motor, J_m is the moment of inertia and B_m is the motor friction coefficient.

As the permanent magnet supplies constant rotor flux to the motor, the magnetizing current $i_d = 0$, and Ψ_f is constant [1]. Upon simplifying equations (1) to (3) above and taking the Laplace transform of ω_m and i_q, the dynamic PMSM model can be modelled by the following transfer functions,

$$\omega_m(s) = \frac{k_t i_q(s) - T_L(s)}{J_m s + B_m} \tag{4}$$

$$i_q(s) = \frac{v_q(s) - K_b \omega_m(s)}{R + sL_q} \tag{5}$$

which is equivalent to a separately excited dc motor drive. K_b and k_t represent the induced emf constant and torque constant. This is the reference model of the PMSM. The full list of numerical values of the motor parameters is given in the Appendix.

3. Concept of ANFIS

The Adaptive Neuro- Fuzzy System, or ANFIS, was proposed by Roger Jang from the Tsing Hua University, Taiwan, back in 1993. According to Jang, the ANFIS is a neural network that is functionally equal to a Sugeno type inference model [7]. The ANFIS is a hybrid intelligent system that synergies the advantages offered by Artificial Neural Network (ANN) and Fuzzy Logic (FL) technology into one system. Thus, to understand the concept of ANFIS, one needs to understand the concept of ANN and FL.

3.1 Fuzzy Logic (FL)

The general methodology of reasoning in FL is by the IF…THEN rules. From Fig. 1 below, a fuzzy inference system (or FIS) basically consists of a formulation of the mapping from a given input set to an output set using fuzzy logic [1]. This mapping process consists of five main steps.

The first step in the mapping process is to fuzzify the crisp input variables, X and Y, which is done by mapping them to the membership functions (MFs) of the fuzzy variables. The second step in the mapping process is the application of the fuzzy operator (AND, OR, NOT) in the IF (antecedent) part of the rule, followed by the

implication from the antecedent to the consequent (THEN part of the rule) [1]. As the ANFIS is based on the Sugeno type inference model, only the Sugeno implication method will be explained here. The Sugeno method can be defined in two ways: the zero-order and the first-order method. In the Sugeno zero-order method, the fuzzy output MFs are only constants; as in the fuzzy output, Z_n = constant, K_n, as shown in Fig. 1. In the Sugeno first-order method, the fuzzy output MFs have linear relations with the inputs; as in the fuzzy output, Z_n, is mathematically expressed as,

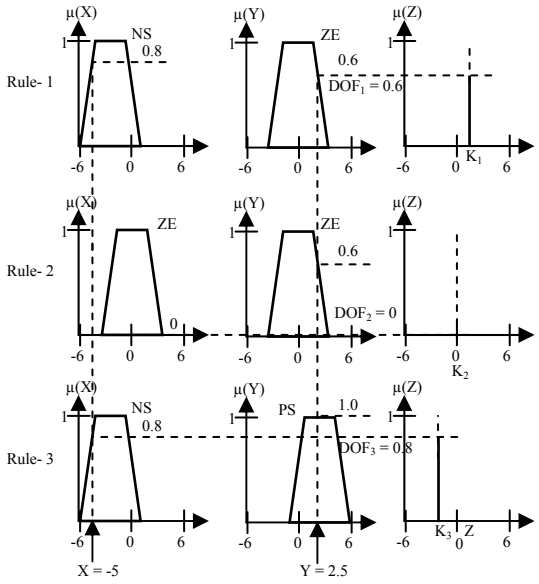


Fig. 1. A two-input, three-rule fuzzy system using Sugeno (zero- order) method.

$$Z_n = A_{0n} + A_{1n} \cdot X + A_{2n} \cdot Y \tag{6}$$

where all the A's are constants. The fourth step in the mapping process is the aggregation of the consequents across all the rules [1], where the total fuzzy output, Z_f, is the union (OR) of all the individual fuzzy outputs, Z_n. The final step in the mapping process is the defuzzification of the total fuzzy output, Z_f, to Z_0, where Z_0 is the crisp output value of the FIS as a result of the crisp input values X and Y. The defuzzification formula for a Sugeno FIS type is,

$$Z_o = \frac{\sum_{n=1}^{n=N} Z_n . DOF_n}{\sum_{n=1}^{n=N} DOF_n} \tag{7}$$

where n denotes the rule number; N, the total number of rules in the rule base; and DOF, the Degree of Fulfilment.

3.2 Artificial Neural Network (ANN)

An ANN is a model that is made to simulate the biological nervous system of the human brain [8]. It is trained, not programmed such as the rule- based FLC, to learn by example from the sets of input-output training data fed into it. The ANN results from the interconnection of artificial neurons via weights that act like gains. A non-linear activation function contributes to the non-linear transfer characteristics of a neuron, which permits non-linear input-output mapping in a neural network (NN). The learning (or training) process of the network has been illustrated simply in Fig. 2 [2], where the training algorithm adjusts the values of the weights until the network output matches the target.

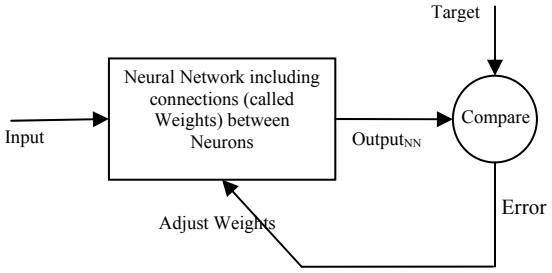


Fig. 2. Simplified schematic of the ANN training process.

3.3 Adaptive Neuro- Fuzzy Inference System (ANFIS)

Though the MFs of the FIS have to be manually adjusted by trial-and-error, the FIS acts like a white box, meaning that the control designers are able to understand how the controller reached its solution. On the other hand, the NN can learn, but acts as a black box as to how it had reached to a particular solution. By employing the NN approach to develop the parameters of the FIS, the FIS is given the ability to learn from a given set of training data, just like an ANN. At the same, the solutions mapped out onto the FIS can be explained in linguistic terms. This learning process of the FIS is illustrated in Fig. 3 [1], where the parameters of the MFs and Sugeno output functions, f_1 and f_2, are adjusted by the Backpropagation (BP) algorithm (one of many training algorithms) until the FIS output matches the target. A more detailed explanation of ANFIS can be found in [1], [7] and [9-11].

4. Development of FTAG FIS

This section summarizes the methodology that was adopted to develop the adaptive FTAG FIS speed control algorithm for the PMSM, which can be subdivided into 4 phases.

4.1 First Phase of Algorithm Development: Reference PMSM Model

An extensive experimentation was initially carried out to determine the best method that generates the most appropriate training data from which the FIS can learn from, the best initial FIS architecture parameters that has the best learning ability, and the most appropriate training method to train the FIS. The adaptive speed controller that was developed at this stage was based on the reference PMSM model that was derived in (4) and (5). Theoretically, from equations (1)- (5), the control principle between the dynamic and reference PMSM model is the same, thus, the developed controller for the reference model should be a good approximate to that required by its dynamic counterpart.

The input-output training data that was used to train the FIS was generated from the PMSM reference model. The speed error, e (rad/s), and the rate of change in speed error, ce (rad/s), was collected to be the input training data whilst the command torque current, I_q* (A), was the output training data. Many training sessions was carried out using the MATLAB ANFIS Editor GUI to determine the type, number and range of training data that would facilitate the most efficient training of the FIS to be an adaptive speed controller. For the type of data, it was found that training the FIS with data that were generated at both the No-Load (NL) and Full-Load (FL) condition of the motor is sufficient for the FIS to generalize for all the other load conditions in between. The number of training data used should also be in proportion to the number of FIS parameters being estimated by the ANFIS.

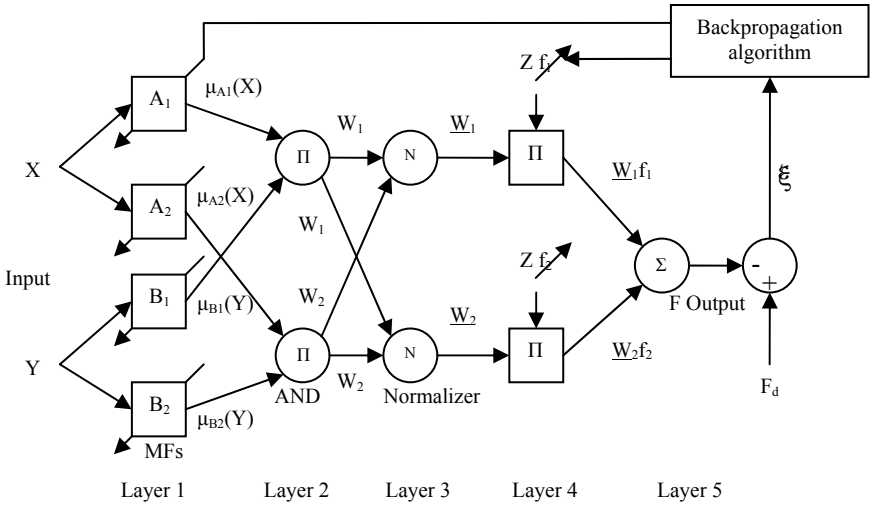


Fig. 3. Corresponding ANFIS structure for a Sugeno fuzzy inference system.

The MATLAB FIS Editor GUI was used to build the initial FIS architecture for the ANFIS training. After numerous testing, it was found that the following initial FIS architecture resulted in the most effective adaptive speed control performance; a two input, Sugeno first-order type FIS that uses 5 and 3 bell type MFs to define the e and ce input, respectively, and 15 Sugeno linear output MF to define I_q*, hence, a 15-rules base.

From here on, this will be referred to as the 5- 3- 15 FIS architecture. Throughout the development of the adaptive speed control algorithm, the hybrid FIS model parameter optimisation method was employed to adjust the parameters of the MFs, which a combination of least squares estimation (LSE) and BP.

The findings at this stage led to a successful development of an adaptive speed controller for the reference PMSM model, referred to as FIS refPMSM. Its robustness has been indicated in Fig. 4.

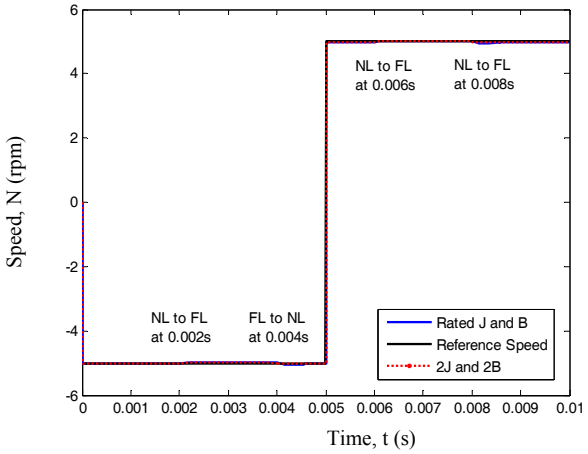
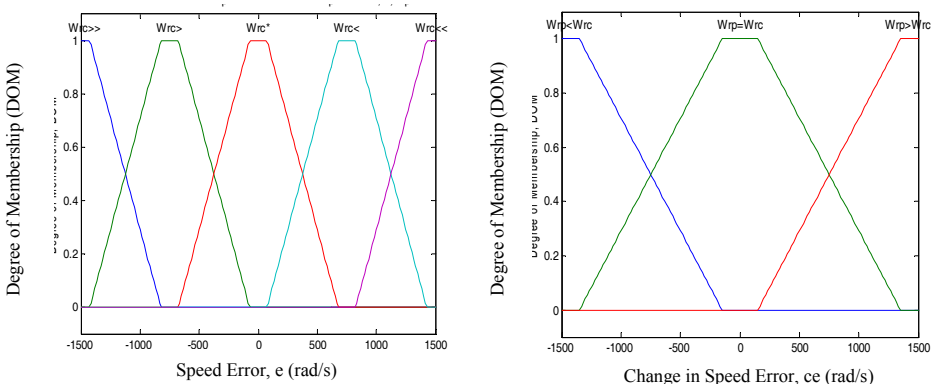


Fig. 4. The resulting speed control performance of the developed adaptive speed control algorithm, refPMSM, for the reference model, at low speeds of ±5 rpm, and unity input- output gains.

4.1 First-Mid Phase of Algorithm Development: Dynamic PMSM Model at NL Condition

In the previous section, it was mentioned that the adaptive speed control algorithm for the dynamic PMSM model could be developed using its reference model. However, this was not the case. This was probably due to the complicated dynamics of the actual PMSM that was not properly represented by the nature of the training data generated from the reference PMSM model, and thus, was not properly mapped by the FIS upon training it.

Therefore it is required to generate the training data directly from the dynamic PMSM model to train the FIS. However, it was very difficult to generate good training data from the random ±1 rad/s step speed commands that was applied onto the dynamic PMSM model as was done previously with its reference model (i.e. random ±1 rpm step speed changes), even at NL condition. This problem was resolved by generating 10 sets of training data from10 step speed changes; ±1 rad/s, ±10 rad/s, ±100 rad/s, ±314 rad/s, and ±628 rad/s (motor's rated speed). This was to ensure that the entire range of the motor speed was covered. This led to the development of an adaptive FLC for the NL condition of the dynamic PMSM model.



(a) (b)

Fig. 5. The initial parameters of the input trapezoidal MFs that were used to define the (a) speed error, e, and (b) change in speed error, ce.

Table 1. The 15- rules base that was used, where wm: actual motor speed, wm*: desired motor speed, ep: previous speed error, e: current speed error, GR: Greatly Reduce, R: Reduce, NC: No Change, I: Increase, GI: Greatly Increase.

ce (rad/s) / e (rad/s)	ep < e	ep = e	ep > e
wm >> wm*	GR1	GR2	GR3
wm > wm*	R1	R2	R3
wm = wm*	NC1	NC2	NC3
wm < wm*	I1	I2	I3
wm << wm*	GI1	GI2	GI3

The 5- 3- 15 architecture that was described previously was used, except that the bell MFs were replaced with trapezoidal MFs that were uniformly distributed across the input range, as shown in Fig. 5. Table 1 shows the 15- rules base that was used.

Here is an example of a rule taken from the table:

IF wm is very much slower than wm*, i.e. e is wm << wm*,

AND e is the same as ep, i.e. ce is ep = e,

THEN let I_q* = GI2, so that the actual motor speed can catch up with the command speed.

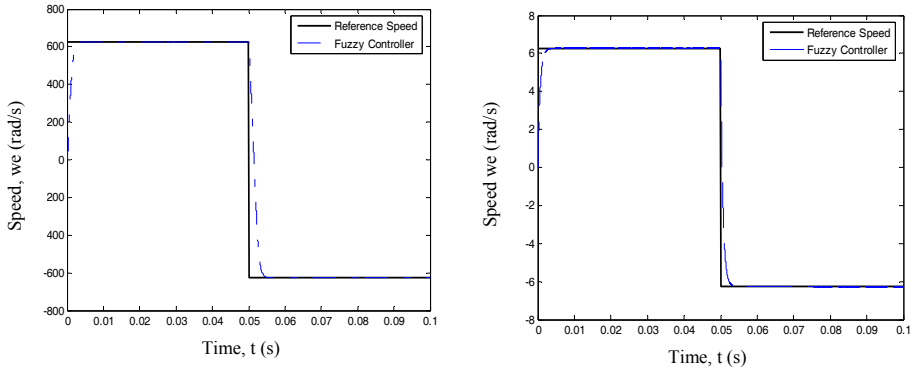


Fig. 6. Reversing transient of (a) rated and (b) 1% rated reference speed settings (at NL condition). Unity output gain.

Upon training the FIS at NL condition, a FLC that is not only able to adapt to variations in the motor parameters, but also respond effectively to small and large reverse transients was successfully developed. This is referred to as FIS dyNLtra, and its superiority at the motor's NL condition has been illustrated in Fig. 6. However, the developed algorithm was unable to track the command speed at situations when there were load torque disturbances of any kind as it was trained only with NL data.

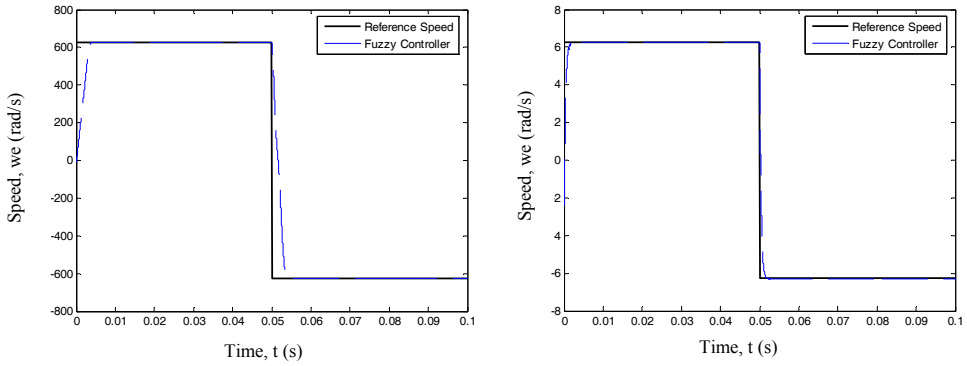
4.2 Second-Mid Phase of Algorithm Development: Dynamic PMSM Model at FL Condition

In this phase, a FIS was trained with training data that was generated from the dynamic PMSM model at FL condition.

In generating the training data from the dynamic PMSM model at FL condition, it was found that the methodology that was used previously at the NL condition could not be employed. After trying out several methods, it was finally decided to generate

the data sets from only the positive speed operating mode of the PMSM. This was based on the assumption that the FIS would be able to learn from its ANFIS training to adapt to the negative speed reference settings. The procedure from here on is the same as before. The 5- 3- 15 FIS architecture as before was used as the initial FIS architecture.

Upon training the FIS at FL condition, a FLC that has good speed tracking performance at both low and high speeds, even with variations in the motor inertia, J, inductance, L, and resistance, R, was developed. It has also proven itself to be able to respond effectively to reverse transients during high and low speeds as shown in Fig. 7. This FIS is referred to as dyFLtra. However, as before, the developed algorithm was unable to adapt itself to any other load torque applications aside from FL.



(a) (b)

Fig. 7. Reversing transient of (a) rated and (b) 1% rated reference speed settings (at FL condition), with a constant output gain of 0.645.

4.3 Final Phase of Algorithm Development: FTAG FIS

The final phase of the algorithm development involves training the FIS with both the NL and FL training data that were generated previously to develop a fuzzy speed controller that will be adaptive towards all possible operating conditions of the PMSM, including during load torque disturbances. This resulted in the self-adapting, speed controller, FTAG FIS algorithm.

The NL and FL data sets that were generated previously were combined together into a data pool. The data sets were stacked on top of the other, alternating between the NL and FL data sets; in a manner such that the speeds were grouped in a descending order to facilitate the training process of the FIS.

For the initial FIS architecture, the 5- 3- 15 FIS architecture was used as before. The FIS that was trained at NL and FL condition is referred to by the name nftrap. Upon simulating the FISnftrap, it was observed that the amount of SSE increases as

the amount of load torque applied decreases. Based on observation and tuning, a list of output gains, G_{ob}, was generated for the different load torque applications.

To convert the non-linear characteristic of the load torque, T_L, into a linear one, the output gains, G_{oa}, for each load torque value were approximated such that the gains were equally displaced from one another by 0.173 for every load change of 0.1 Nm. Thus, the following linear relationship between T_L, and FLC output gain, G_{oa}, could be derived:

$$Approx.OutputGain = f(LoadTorque) \tag{10}$$

$$G_{oa} = m \times T_L \tag{11}$$

where the linear slope, m, between T_L and G_{oa} is a constant 0.173. However, for accuracy, the value of m was represented as a fraction of 1.04/0.6. The load torque, T_L, can be estimated from equation (4). The complete configuration of the developed adaptive controller is illustrated in Fig. 8, which is called the Function Torque Adapted Gains Fuzzy Inference System (FTAG FIS) because the output gain of FLC nftrap is adjusted accordingly to the amount of T_L applied onto the PMSM. The FTAG FIS has been tested for loads as low as 0.05 Nm (to approx. the motor's NL condition), and had exhibited good speed tracking performance, even at unity input gains.

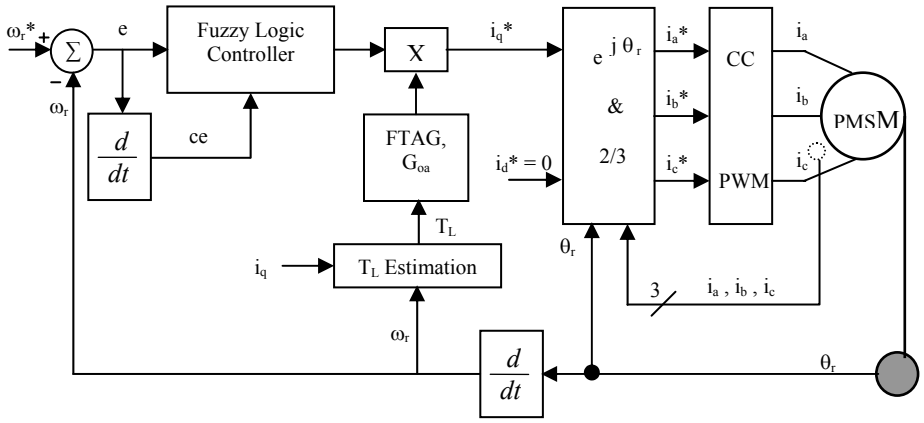


Fig. 8. The configuration of the FTAG FIS speed control algorithm.

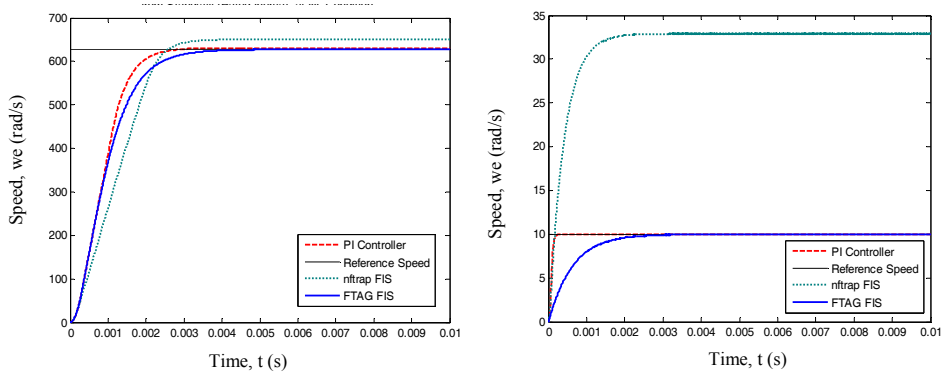
5. Comparison Between Simulated Speed Responses of Developed FTAG FIS and PI Controller

In this section, a thorough comparison between the developed adaptive speed control algorithm, FTAG FIS, the FIS nftrap, and the conventional PI controller is made to observe the robustness of the developed FTAG FIS. The following set of benchmark tests was used as a guide to evaluate the controllers' performance:

- Large step speed command from standstill, under rated inertia condition and with increases inertia;
- Small reference speed change, with rated and with increased inertia;
- Step load torque application;
- Reversing transient.

The above benchmark tests were proposed by Z. Ibrahim and E.Levi in [12]. Unlike many papers that only carry out their comparisons at a single transient or single operating point, these benchmark tests are performed at both high and low reference speed settings; 628 rad/s and 10 rad/s, respectively, to justify the validity of the evaluation and comparison of the speed controllers.

The FIS controller nftrap was simulated with a constant output gain of 0.645 to illustrate the effect of not employing an adaptable gain such as that used in the FTAG FIS algorithm. As for the PI controller, the initial K_p and K_i gains that were used were the same gains that were used to generate the training data to train the FIS dyNLtra and nftrap, both of which were used to develop the FTAG FIS algorithm.



(a) (b)

Fig. 9. Response to (a) 628 rad/s and (b) 10 rad/s step speed command from standstill (rated inertia J), at NL condition.

From the simulated results in Fig. 9 to Fig. 12, one may conclude that the PI controller has better speed control performance than the developed FTAG FIS algorithm; with faster speed tracking response at both low and high speeds, even

during motor parameter variations (where motor inertia J is twice the rated value) and sudden speed changes (where a step 10% speed reference reduction from the command speed). However, it should be highlighted to the reader that the PI controller gains that were used at rated and low speeds had to be manually tuned each time. The reason the three controllers' speed performance was tested at rated and 10 rad/s reference speed setting is because the optimum PI gains for these speeds were already known. At any other (lower) reference speeds, the optimum PI gains would have to be found by trial and error. Besides that, it is also desired to illustrate the competitive advantage that FTAG FIS has over the PI controller at operating conditions from which the FIS' training data were generated. Though the PI controller has the fastest transient time among the three, the developed FTAG FIS is only about 1 to 5ms slower than the PI controller, with negligible SSE as well.

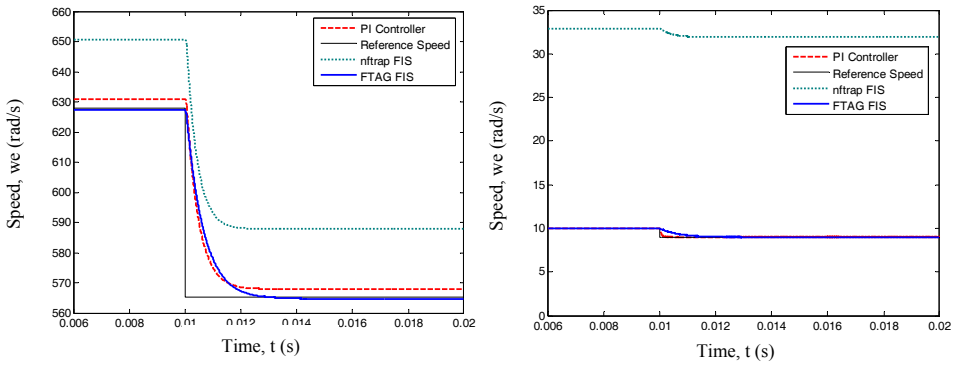


Fig. 10. Response to step 10% speed reference reduction from (a) 628 rad/s and (b) 10 rad/s (rated inertia J), at NL Condition.

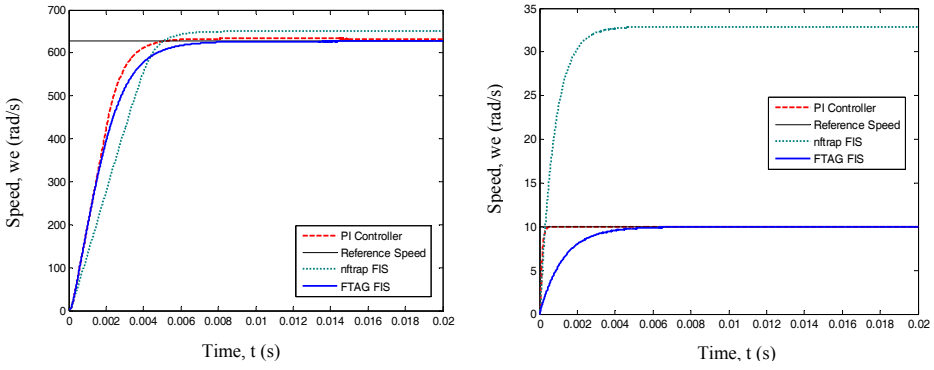


Fig. 11. Response to (a) 628 rad/s and (b) 10 rad/s step speed command from standstill (increased inertia J), at NL Condition.

The superiority of the developed FTAG FIS algorithm over the conventional PI controller and even the FIS nftrap can be observed in its response to sudden load torque disturbances and reversing transients in Fig. 13 and 14, respectively. From Fig. 13, it is clear to see that FTAG FIS was able to adapt itself efficiently from a NL to a FL condition, without requiring any user intervention. Though the PI controller can be adjusted to provide a similar or even possibly better load rejection transient than FTAG FIS, the point that is desired to be made here is that FTAG FIS will automatically adjust itself to any operating condition that it is in to ensure that the motor will track the reference speed setting in a short recovery time with negligible SSE. The FTAG FIS also yields a rapid speed reversal with no overshoots. On the other hand, the PI controller is characterized with over and undershoots that are significantly large and harmful to the motor at low speeds. In short, the developed FTAG FIS offers superior speed tracking performance, speed recovery from sudden load torque and speed changes, and speed reversals at both low and high speeds, even during variations in the motor parameters.

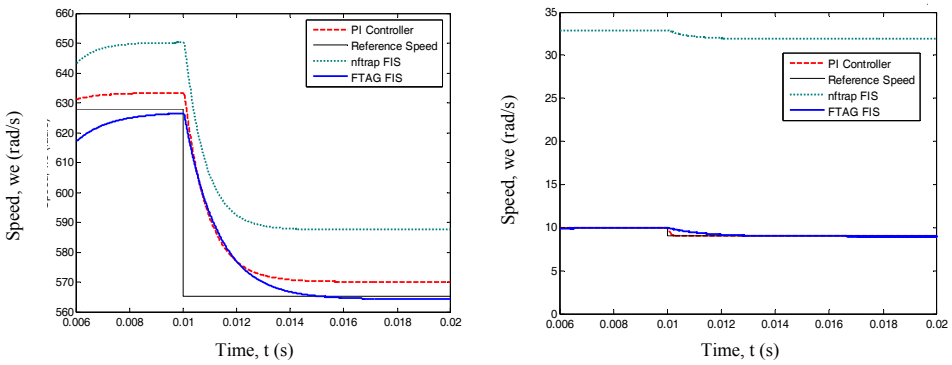


Fig. 12. Response to step 10% speed reference reduction from (a) 628 rad/s and (b) 10 rad/s (increased inertia J), at NL Condition.

6. Conclusion

This paper has presented a successful development of FTAG FIS, an adaptive speed controller for the PMSM using the Adaptive Neuro-Fuzzy Inference System. The developed control algorithm has been verified through simulations to be not only more superior to the conventional PI controllers, especially during load torque disturbances, but also to be robust during speed changes, speed disturbances, as well as variations in the motor parameters. The control algorithm design methodology that has been presented in this paper shows its effectiveness in eliminating the need for the control designer to adjust the input and output gains of the fuzzy controller manually by trial and error. On top of that, it has disguised the non-linear characteristic of the load torque into one that is linearly related to the output gain of the developed fuzzy

controller. The FTAG FIS algorithm guarantees the closed loop performance of the PMSM even when the motor is subjected to significant and unpredictable motor parameter variations and load torque disturbances at both low and high speeds. An outstanding advantage of the FTAG FIS algorithm for the PMSM is that it uses only 15 rules in its rule base, which is less than half the number of rules that is being used in many other fuzzy controllers that have been proposed in the literature. With fewer rules, the FTAG FIS is not only more easily implemented, but will have shorter execution time. Suggestions for further work include a real time implementation of the algorithm onto the motor to verify whether there is a close agreement between the theoretical and experimental results and modifying the current FTAG FIS algorithm to incorporate one, and not two, FIS.

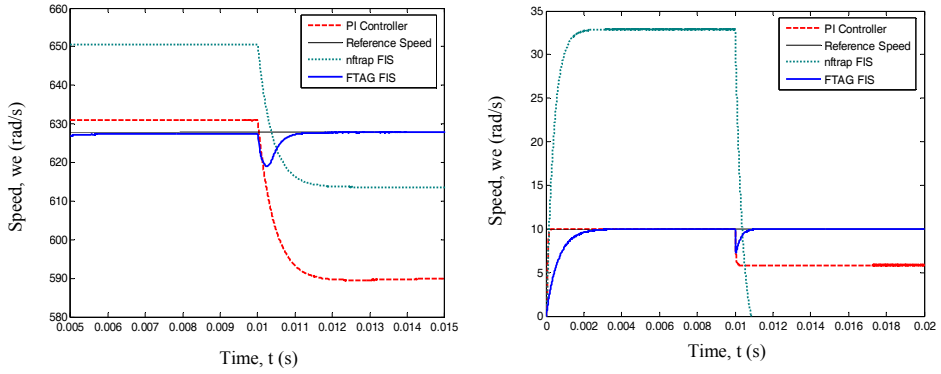


Fig. 13. Response to step load torque applications at (a) 628 rad/s and (b) 10 rad/s reference speed settings, from NL to FL Condition.

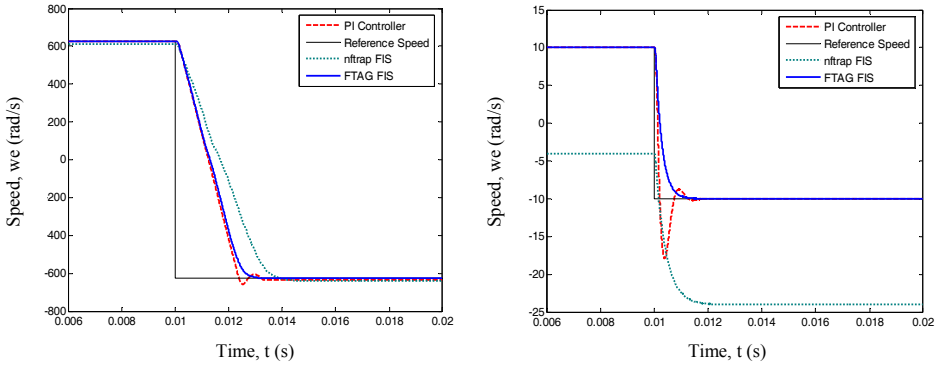


Fig. 14. Reversing transient (a) 628 rad/s and (b) 10 rad/s reference speed settings, at Rated Conditions.

Appendix: Machine Parameters

230V, 377W, 1.9A, 6000 rpm, 4 pole- pairs, L = 5 mH, R = 3.1 Ω, Ψ_f = 0.19 volts/rad/s, J_m = 0.251 Kgcm^2, and B_m = 3.6 x 10^{-5} Nm/rad/s.

References

1. Bose, B.K. (2002). *Modern Power Electronics and AC Drives*, Upper Saddle River, NJ: Prentice- Hall PTR.
2. Uddin, M. N. and Rahman, M.A. (1999). Fuzzy Logic Based Speed Control of an IPM Synchronous Motor Drive. *Proceedings of the 1999 IEEE Canadian Conference on Electrical and Computer Engineering Shaw Conference Centre, Edmonton, Alberta, Canada*, May 9-12, 1259-1264.
3. Rahman, M.A., Vilathgamuwa, D.M., Uddin, M. N. and Tseng, K.J. (2003). Nonlinear Control of Interior Permanent- Magnet Synchronous Motor. *IEEE Transactions on Industry Applications.* 39 (2), 408-416.
4. Rahman, M.A., Uddin, M.N. and Abido, M.A. (2004). Development and Implementation of a Hybrid Intelligent Controller for Interior Permanent- Magnet Synchronous Motor Drives. *IEEE Transactions on Industry Applications.* 40(1), 68-76.
5. Heber, B., Xu, L. and Tang, Y. (1997). Fuzzy Logic Enhanced Speed Control of an Indirect Field- Oriented Induction Machine Drive. *IEEE Transactions on Power Electronics.* 12(5), 772-778.
6. Zhen, L. and Xu, L. (2000). Fuzzy Learning Enhanced Speed Control of an Indirect Field- Oriented Induction Machine Drive. *IEEE Transactions on Control Systems Technology*. 8(5), 270-278.
7. Negnevitsky, M. (2002). *Artificial Intelligence: A Guide to Intelligent Systems*, 1st ed., Addison Wesley, Pearson Education Limited.
8. Math Works Inc. (2004). *Neural Network Toolbox For Use with Matlab® User's Guide Version 4*.
9. Zilouchian, A. and Jamshidi, M. (2001). *Intelligent Control Systems Using Soft Computing Methodologies*, Boca Raton, FL: CRC Press.
10. Lin, C.T. and George, C.S. (1996). *NEURAL FUZZY SYSTEMS: A Neuro- Fuzzy Synergism to Intelligent Systems*, Upper Saddle River, NJ: Prentice- Hall PTR.
11. Math Works Inc. (1998). *Fuzzy Logic Toolbox For Use with Matlab® User's Guide Version 2*.
12. Ibrahim, Z. and Levi, E. (2000). A Comparative Analysis of Fuzzy Logic Speed Control in High Performance AC Drives Using Experimental Approach. *IEEE*, 1217-1224.
13. Kouzi, K., Mokrani, L. and Nait- Said M.S. (2003) A Fuzzy Logic Controller with Fuzzy Adapted Gains Based on Indirect Vector Control for Induction Motor Drive. *Journal of Electrical Engineering,* 3, 43-49.