Design and Implementation of Fuzzy Logic system for DC motor Speed Control

Dr. Maan M. Shaker
Electrical Engineering
Technical College
Mosul, Iraq
Khalil@presto.ac.uk

Yaareb M.B. Ismeal Al-khashab
Computer Technology Engineering
State Commission for Dams & Reservoirs
Mosul, Iraq
alkhashab60@gmail.com

***Abstract*— In this paper an integrated electronic system has been designed, constructed and tested. The system utilizes an interface card through the parallel port in addition to some auxiliary circuits to perform fuzzy control operations for DC motor speed control with load and no load. Software is written using (C++ language Ver. 3.1) to display the image as control panel for different types of both conventional and fuzzy control. The main task of the software is to simulate: first, how to find out the correct parameters for fuzzy logic controller (membership's function, rules and scaling factor). Second, how to evaluate the gain factors (K_P, K_I and K_D) by Ziegler-Nichols method. When executing any type of control process the efficiency is estimated by drawing the relative speed response for this control.**

I. INTRODUCTION

Almost every industry and household has motors used in their equipment or appliances. Motors that are often controlled by computers have also become an essential part of many motion control systems. Major problems in applying a conventional control algorithm in a speed controller are the effects of nonlinearity in a DC motor. The nonlinear characteristics of a DC motor such as dead zone, saturation and friction could degrade the performance of conventional controllers [1-3]**.** Many advanced model-based control methods such as variable-structure control [4] and model reference adaptive control [5] have been developed to reduce these effects. However, the performance of these methods depends on the accuracy of system models and parameters. Generally, an accurate non-linear model of an actual DC motor is difficult to find, and parameter values obtained from system identification may be only approximated values. Emerging intelligent techniques have been developed and extensively used to improve or to replace conventional control techniques because these techniques do not require a precise model. One of intelligent techniques, fuzzy logic developed by Zadeh [6, 7] is applied for controller design in many applications [8, 9]. A fuzzy logic controller (FLC) was proved analytically to be equivalent to a nonlinear controller when a nonlinear defuzzification method is used [10]. Also, the results from the comparisons of conventional and fuzzy logic control techniques in the form of the FLC and fuzzy compensator [11-14] showed that fuzzy logic can reduce the effects of nonlinearity in a DC motor and improve the performance of a controller. The primary goal of control engineering is to distill and apply knowledge about how to control a process so that the resulting control system will reliably and safely achieve high-performance operation. In this thesis, it will be shown how FL provides a methodology to represent and implement the knowledge about how to control a process. Indeed the best way to really learn fuzzy control is to design your own fuzzy controller for DC motor, and simulate the fuzzy control system to evaluate its performance. Initially, it is recommended to code this fuzzy controller in a high-level language (C++) after having acquired a firm understanding of the fuzzy controller's operation, which it can take shortcuts by using (or designing your own) CAD package for fuzzy control system(FCS).

In this paper, the FLC is implemented to control DC motor speed of fuzzy logic system. Heuristic knowledge **is** applied to define fuzzy membership functions and rules. The membership functions and rules are modified after initially design the software to simulate the fuzzy control(FC). The necessary hardware interface circuit along with the required software is given. The results from real time experiments with load and no load conditions are also included[15].

II. THEORY

To control DC motors, a variable voltage must be generated and applied to power the motor. There are two main sources of voltage supply for a motor: the linear power supply, and the switching power supply. The linear power supply technique uses power transistors acting in their linear

mode to provide a smooth and continuously adjustable output voltage depending upon load drive requirements. In this mode, the transistors are used as variable resistors, conducting according to the level of the input signal applied. This method of control wastes power when the transistors are used to drop voltage and in doing so dissipates power. Switching power supply techniques use transistors to switch the voltage through to the motor in a series of pulses that applies an 'average' voltage to the motor. Two switching methods are commonly used, pulse-width modulation (PWM) and pulse-frequency modulation (PFM). Pulse-width modulation is the more common method used for motor control. This method uses a constant period between sequential pulses, while the width of the pulses is altered (duty cycle) to change the effective or average voltage applied to the motor. Pulse-frequency modulation varies the frequency of pulses having a constant pulse width to control the average voltage applied to the motor [16]. The most common means of controlling the voltage applied to DC motors is through the use of H-bridge circuit as shown in Fig.1 Two switches are closed at a time to switch the DC voltage through the motor. When switches designated as SW1 and SW4 are closed and switches SW2 and SW3 are open, as shown in the diagram, the motor will rotate in one direction. Swapping the positions of the open and close switches to make SW2 and SW3 closed and SW1 and SW4 open will reverse the flow of current through the bridge and reverse the direction of motor rotation. The average voltage delivered to the motor can be controlled using pulse-width modulation applied to the switches. In practice, the switch function is performed using power transistors in place of each switch. As the switch is opened, a large voltage is generated across the motor winding, known as back-emf. This voltage must be limited to avoid damage of the active switching transistors. Damage is prevented by fitting diodes across the transistors that clamp the voltage to much lower levels that will not cause damage.

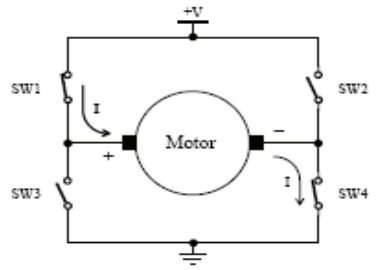

Fig. 1 Motor control using an H-bridge

III. CONTROLLER DESIGN

The main steps of the proposed system are:

1. Design a suitable hardware to achieve the task.
2. Identify the types of the two inputs and output for FC. The inputs were chosen, in this paper, error (e) and change of error (Δe), but the output, in all types of control, is the control output (C) or armature voltage, except in the PI like FC; the output is the change of control output (ΔC), as the following equations:

$$e(t)=set\ point - actual\ speed \quad (1)$$

$$\Delta e = e(t) - e(t\text{-}1) \quad (2)$$

Where e(t) is the error, Δe is change of error, e(t-1) is previous error.

$$\Delta C=C(t) - C(t\text{-}1) \quad (3)$$

Where ΔC is change of control output, C(t) is control output, C(t-1) is previous control output.

3. Evaluate the universe of discourse of the inputs and output depending on the DC motor characteristics.
4. Normalize the universe of discourse with (±1) for two inputs and output.
5. Determine the number of MFs and its shape for inputs and output, in this paper trapezoidal MF has been chosen because it is easy to represent, and it is possible to convert it to a triangle. The number of MF is fixed to seven (NB, NM, NS, ZE, PS, PM and PB) to include all the possible cases. Each MF means the linguistic variable of the inputs and output, for example (NB) means that the actual speed is more than the set point (reference speed) according to the relation which was mentioned in step (2) (NM) and (NS) has the same effect of (NB) but with the less value, (ZE) means that the actual speed is equal to the reference speed (PB) and others means that the opposite of the (NB) state.
6. Initialization process: Initialize all the MFs and the rules to a specific value moreover, initialize the scaling factor for inputs and output, all the values above considered as a starting point,
7. Fuzzification process: Convert the crisp inputs into linguistic variable according to the range of each MF.
8. Inference engine process: The method for inference engine which has been used during this work is Mamdani method using the (AND) fuzzy operator to format all the rules.
9. Defuzzification process: The center of area (COA) method is used in this work, because it is easily comfortable with the DC motor in additional to being easy computed and represented.
10. Tuning process: the priority of the tuning process is as in the following steps:
    - Change the scaling factor for the output (g_2), and monitor the speed response until it reaches the desired speed.
    - Change the scaling factor for the inputs (g_0, g1), and monitor the speed response until it reaches the desired speed.
    - Change the shape of MFs and monitor the speed response until it reaches the desired speed.

- Change the rules table and monitor the speed response until it reaches the desired speed, and so on.

These four priorities above are repeatedly changed until the exact speed response is obtained.

The proposed system shown in Fig. 2 consists of a clocking source which can be a crystal generating suitable frequency that inserted to the speed counter. The counter is used for counting speed of the signal that comes from the tachometer north pulse (Np). The frequency-type tachometer (encoder) is used here. The control device could be a microprocessor or a microcontroller. The system is connected to a PC through its parallel port, and the input is the data that represents the speed of the DC motor from the counter.

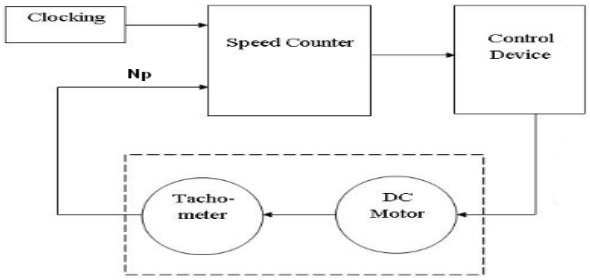

Fig. 2 Block diagram of the proposed system

The form of the output has two signals A and B(connected to SW1,SW2,SW3 and SW4) that represent the direction and the value of desired speed according to the width of the pulse. As shown in Table. 1.

TABLE 1. Truth table for the four switches

SW1	SW2	SW3	SW4	Motor direction
On	Off	Off	On	Clockwise
Off	On	On	Off	Counter Clockwise
On	Off	On	Off	No movement (brake)
Off	On	Off	On	No movement (brake)
On	On	Off	Off	Short circuit
Off	Off	On	On	Short circuit
Off	Off	Off	Off	No movement (motor freewheels)

The basic configuration of the FLC consists of four main components (fuzzification, knowledge base, inference engine, and defuzzification). Each component have several stages. The first stage of the fuzzy controller operation is fuzzification and the last one is defuzzification. On both stages, the MFs which describe different values of the linguistic variables (or labels) are applied. To choose MFs, first of all, one needs to consider the universe of discourse for all the linguistic variables applied to the rules formulation.

To specify the universe of discourse, one must firstly determine the applicable range for a characteristic variable in the context of the system designed. The range you select should be carefully considered. Because of this situation, it is usually desirable and often necessary to scale, or normalize the universe of discourse of an input/output variable. Normalization means applying the standard range of (±1) for the universe of discourse both for the inputs and the outputs.

In the case of the normalized universe, an appropriate choice of specific operating areas requires scaling factors. An input scaling factor transforms a crisp input into a normalized input in order to keep its value within the universe. An output scaling factor provides a transformation of the defuzzified crisp output from the normalized universe of the controller output into an actual physical output.

To design FLC, it must be taken into consideration how to find the methods that represent all the components above with high flexibility to change and adjust at any time to get the proper results that verify the purpose of the designed FLC. Using a PC to design the FLC is very useful in this case study, since it is possible to change all parameters (types of membership function (MF), coordinate of each MF, universe of discourse, scaling factor, rules) at any time with different values. In this case exploiting time is necessary to be estimated when using different methods of conventional control (P,PD, PI, and PID), and moreover, it should be fixed out for three methods of fuzzy control (FC): (1) PD-like FC, (2) PI-like FC, (3) PID-like FC) to compare with them. The proposed method allows correcting and choosing the applicable gains factor of the conventional control (K_P, K_I, and K_D) and in to adjust the scaling factor of the FC (g_0, g_1, and g_2). In all control methods, the speed response could be drawn (speed indicator) for each method to measure the efficiency of control. At the same time it is possible to change the gain and scaling factor to get the proper speed response with less steady state error, rise time and settling time. On the other hand, when designing fuzzy logic microcontroller there are several points that must be considered to determine the final shape of the MF that could be practically applied to the proposed system design and the coordinates of each MF, as well as limiting the range of each MF and choosing the final rules that achieve the desired output. It is not a practical way to work on microcontroller directly because any change in the conditions mentioned above will oblige to drag the microcontroller from the circuit designed and put it in the programmer device to burn the old program then to be replaced by the new program, and then return it into the circuit designed. The other reason is that there is no available software for this specific work. For this reason a suggestion for a flexible simulator must be designed to easily change the MF and rules at any time, to determine the final shape of each MF with proper coordinates and rules to obtain the required speed. The whole proposed system consists of four main parts as shown in Fig. 3.

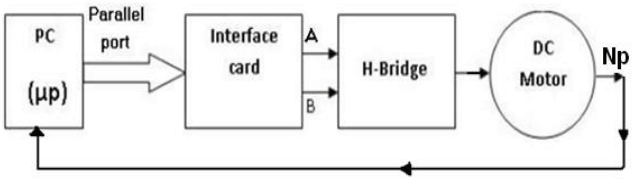

Fig. 3. Block diagram of the proposed hardware system

1. *The PC:*

Any pc supported operating system MS-DOS or WINDOWS.

2. *Design the interfacing card:*

The main reason for designing an interface card is reading the speed that is counted in the counter in addition to generating the two signals (A and B) sending to H-Bridge to rotate the DC motor in two directions with the value of speed at which they were read. The block diagram is shown in Fig. 4.

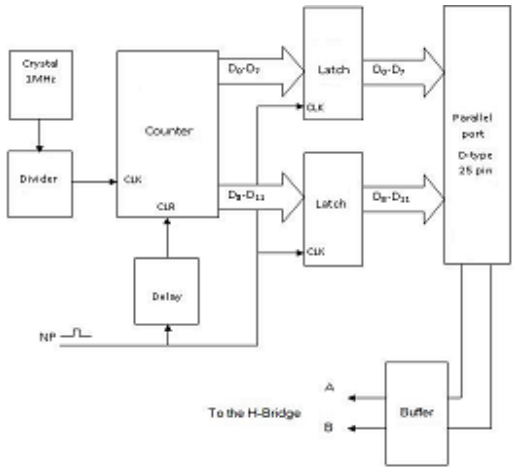

Fig. .4. Block diagram of an interface card by parallel port

The frequency-type tachometer is used in this paper, this means the output of the tachometer produces pulses (Np). The Np is the desired signals were used to feed the counter and the two latches to clear the counter in the raising edge of the Np and operating as (CLK) to the two latches. To perform synchronization, a delay time circuit will be needed in the way of Np between the counter and the two latches. In this case the Np is delayed and connected to the CLR pin of the counter, where the Np is directly connected to the CLK of the two latches.

The essential task for the interface card now is to read (12 bit) information from the two latches by making the base (378H) pins (2-9) of the parallel port adapter as the LSB inputs in additional to (base+1) pins (10,11,12,13) as the MSB of the inputs, which represents the speed of the DC motor.

3. *H-bridge DC motor:*

The H-bridge is a device designed to control an electric motor by accepting control signals from some control device, and to translate them into a higher voltage, high current signals required to drive the motor forwards or backwards at various speeds. Normally an H-bridge will avoid the illegal short circuit states by employing extra components that control the main power switches. The H-Bridge used in this paper is shown in Fig. 5. and it employs such a circuit with small signal transistors being used to control the power darlington transistors. The transistors are wired together in such a way that two input wires can be used to control the device. It is obvious that the purpose of the motor driver is to convert the output of the controller into correct inputs to the motor. There are two logic level compatible inputs, A and B, and two outputs send to the motor. If input A is brought high, and input B is brought low, the motor goes in one direction. If input B is driven, the opposite happens and the motor runs in the opposite direction. If

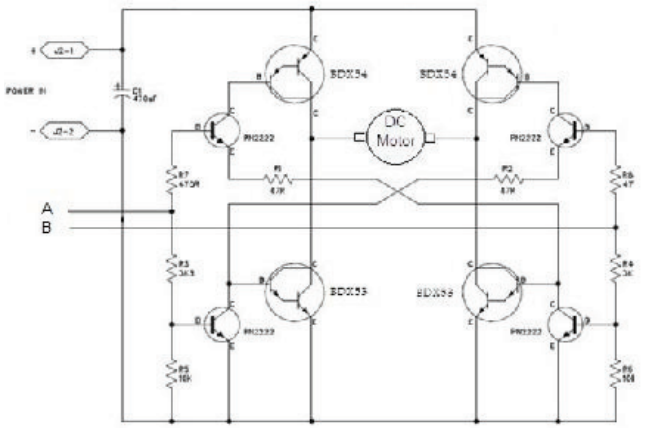

Fig. 5. The proposed H-bridge circuit

both inputs are low, the motor is not derived and can freely "coast", and the circuit consumes no power. If both inputs are brought high, the motor is shorted and breaking occurs. The truth table for this operation is shown in Table 2.

TABLE .2 Truth table for DC motor driver

Input		Output
A	***B***	
0	0	no power
1	0	run in one direction
0	1	run in opposite direction
1	1	Short Circuit

In all available programs which were written using MATLAB, there is a problem since the programmer suffers from representing rules because the programs were written by using the expression "IF– THEN" rule and this means that at performing any change in the rules it is necessary to get back to the source program and performing the operation of compilation and linking then running the programs another time. For this reason, a new basic ruler is demanded in this work to present the fuzzy tables and the MF. The MF as an array is described as three-dimensional array with [3*7*4] dimensions where (3) means two inputs (error and change of error) and one output, (7) means number of MFs negative big (NB), negative medium (NM), negative small (NS), zero (ZE), positive small (PS), positive medium (PM), positive big (PB) and (4) is the coordinates of trapezoidal (X-coordinate of four bounded points) which constitutes MF. For describing fuzzy rule tables an array [7*7] is required. The maximum number of rules in this array is (49). To fill an array with numbers (0 – 7), each linguistic variable is represented as fuzzy number to reduce the arithmetic operation. Table. 3 explains this operation. The program stores the values of the final two arrays when terminating the program in text file and it can re-read them at input again. When giving control order to the DC motor by using FC, the program uses the final values that are existed in these arrays to apply the FC. To perform this matter it is inevitable to use a language which has the ability to deal with the arrays in a very quick way besides it should save high flexibility in addressing mode and hence the programming language C++ Ver. (3.1) is used [15]. The software has ability to store and re-read all the variables. The main program in fact is consisting of many subprograms integrated to form one program. Each subprogram represents the type of control that is used. The core of the program has many functions that may change its parameters and show these changes on the control panel.

TABLE. 3. Array of two inputs and output

No Rule	PB	PM	PS	ZE	NS	NM	NB
7	6	5	4	3	2	1	0

When the program is running the control panel is displayed as shown in Fig. 6.

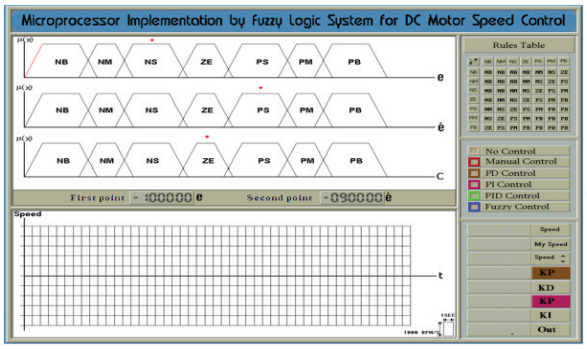

Fig. 6. Control panel of the main program

The features of the control panel are:

- Rules table consists of (49) rules with flexibility to change any rule at any time, and also it is possible that there is no rule at all.
- Seven MFs (NB, NM, NS, ZE, PS, PM, PB) were displayed for the first input (e) and the second input (Δe or ce) in additional to the output with the range (±1). The red line indicates to the first line of trapezoidal MF (NB) of the first input (e), at the same time the X-coordinate for this line is displayed (i.e. first and second points) with possibility to move the red line to any one of the MFs inside the first input or to the next input (Δe) or the output (C). The new MF may be changed again and again until the desired MFs is verified.
- The control panel with different types of control (PD, PI, PID, PD like FC, PI like FC, and PID like FC) is displayed. It allows activating the required type of control that has been selected with flexibility to change the gain and scaling factor for all control.
- The speed response is drawn for any control type that has been selected.

IV. PRACTICAL RESULTS

- *Manual control:*
which is designed to help in checking all the parts of hardware which were designed, it is the same as P-control but the value of (K_P) is equal to one. The speed of the DC motor can be controlled from the keyboard. The main purpose of the MC is to test all the designed hardware by giving the value of the desired speed to the MC, then monitoring and tracking the DC motor to converge this speed. If this operator is being executed, so the hardware is working properly. The speed response for this test control is shown in Fig. 7.

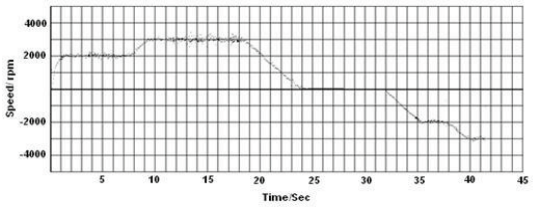

Fig .7. Speed response of the manual control

- *Conventional control:*
To find the gain factors (K_p, K_I and K_D) without a mathematical model, the Ziegler-Nichols method is used to calculate the gain factors. The first step is to find the critical gain by applying the P-control and increasing the gain factor from (0) to the critical value K_{cr} until the output exhibits sustained oscillations. The second step is to determine the

value of P_{cr} from the generated pulses in the first step by measuring the time between peak to peak. The software evaluates the gain factors using try and error by changing the values of the gain factors and monitoring the speed response, until it reaches the convinced value. The speed response for all conventional controls used in this work with load is shown in Fig. (8,9 and 10).

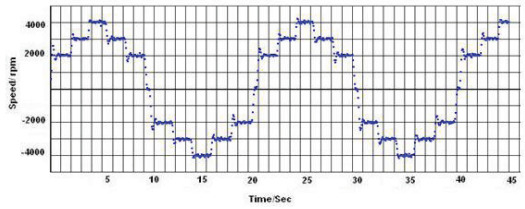

Fig. 8. Speed response for the PD control

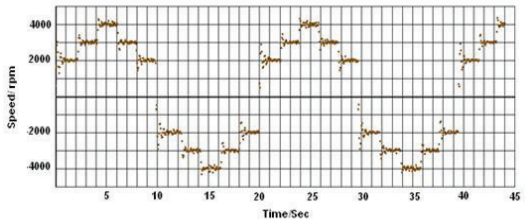

Fig. 9. Speed response for the PI control

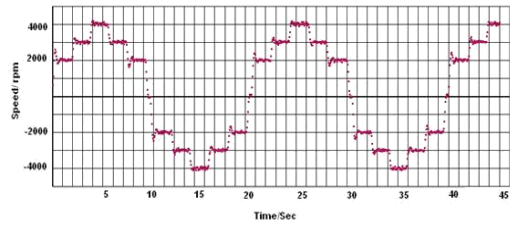

Fig. 10. Speed response for the PID control

- *Fuzzy control:*
  The FC requires more tuning during the simulation process. The tuning is used for rules, MFs and scaling factors. Moreover, the universe of discourse should be normalized for two inputs and output. For this reason, initial value for all above functions should be used, and is considered as a starting point. The initial MFs and rules were shown in Fig. 11, and Table. 4.

From the determined starting point the operation of tuning is occurred by using the try and error method. Executing the program many times will help to see the speed response which is considered as an indicator of the appropriate results by changing the values of the mentioned functions. When completing the tuning, the final shape of the MFs and final rules were as shown in Fig. 12 and Table. 5.

Fig. (13,14 and 15) shows the speed response to all types of FC with load.

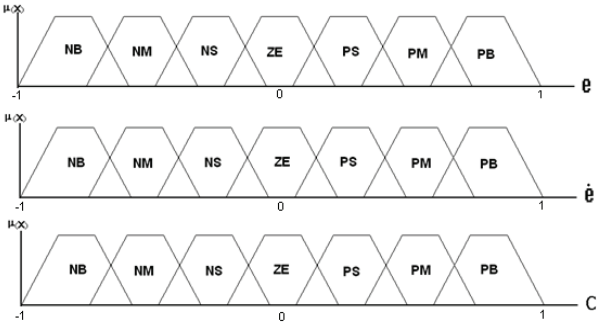

Fig. 11. Initial MF

TABLE .4 Initial rules

e / ce	NB	NM	NS	ZE	PS	PM	PB
NB	NB	NB	NB	NB	NM	NS	ZE
NM	NB	NB	NB	NM	NS	ZE	PS
NS	NB	NB	NM	NS	ZE	PS	PM
ZE	NB	NM	NS	ZE	PS	PM	PB
PS	NM	NS	ZE	PS	PM	PB	PB
PM	NS	ZE	PS	PM	PB	PB	PB
PB	ZE	PS	PM	PB	PB	PB	PB

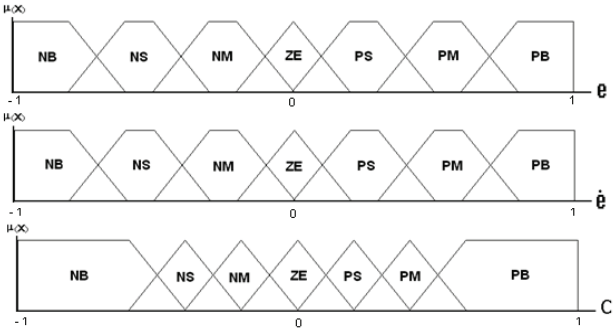

Fig .12 Final MFs

Table .5 Final rules

e / ce	NB	NM	NS	ZE	PS	PM	PB
NB	PB	PB	PM	PM	PS	PS	ZE
NM	PB	PB	PM	PM	PS	ZE	NS
NS	PB	PB	PM	PS	ZE	NS	NM
ZE	PB	PM	PS	ZE	NS	NM	NB
PS	PM	PS	ZE	NS	NM	NB	NB
PM	PS	ZE	NS	NM	NM	NB	NB
PB	ZE	NS	NS	NM	NM	NB	NB

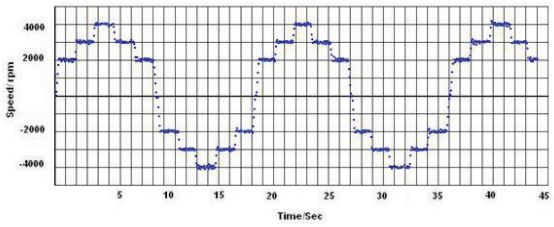

Fig. 13. Speed response for PD-like FC with load

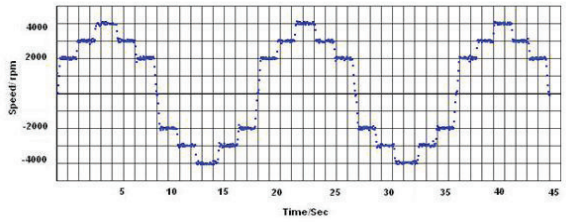

Fig. 14. Speed response for PI-like FC with load

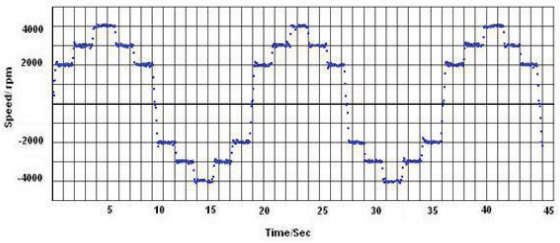

Fig. 15. Speed response for PID-like FC with load

V. CONCLUSION

In this paper FLC system is designed to control the speed of any DC motor and implemented. The design dramatically reduced the hardware to the least possible and on the other hand made the system software rather a complex one. The software combined the simulation of all types of control (i.e. conventional types of control as well as FC). By using this software, the user can find out the correct values of the control parameters for all control types in the conventional and FC. Moreover there is a capability to enhance the system performance by altering the MFs and the fuzzy rules in order to obtain the optimal result (by monitoring the motor speed response).

The FLC is implemented to evaluate the proposed system, and to display the speed response drawing for all control types used in this paper. In FLC the corresponding values of all parameters (over shot, settling time, rise time, steady state) are very small in comparison with the parameters of the conventional control. That means the FC has higher stability, reaching the desired speed in a shorter time, but requires more tuning than conventional control, because FC contain more computation and more parameters (MFs, rules and additional scaling factor in the inputs and output) unlike conventional control (gain factors for inputs only).Fig. (16,17 and 18) shows the comparison between the conventional control and the FC.

To achieve better performance and less utilization of available hardware resources, the work had been built to reduce the total number of used rules and to eliminate those which are not effective on the system.

From experience, accurate results can be achieved if the relationship between the FC & conventional control parameters were computed.

For more reliability a comparison between practical results and those which were produced by simulation (MATLAB 7.0 and *fuzzy* TECH 5.54d) are achieved .

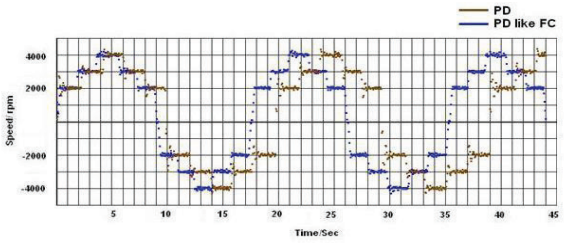

Fig. 16. Comparison between PD & PD-like FC with load

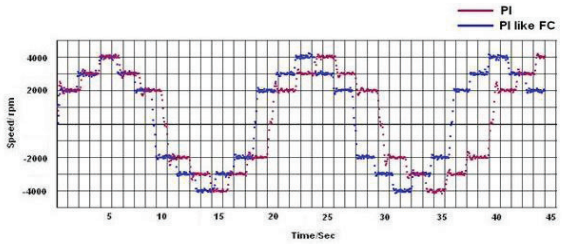

Fig. 17. Comparison between PI & PI-like FC with load

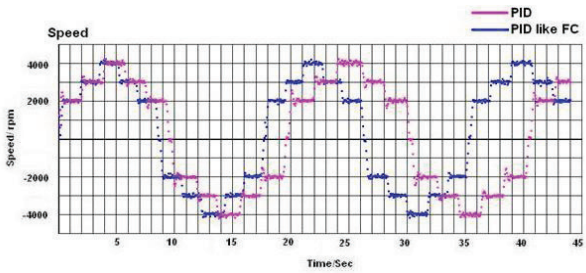

Fig. 18. Comparison between PID & PID-like FC with load

REFERENCES

[1] B. J. Chalmers, "Influence of saturation in brushless permanent-magnet motor drives," IEEE Proc. B, Electr. Power Appl., vol. 139, no. 1, pp. 51-52, 1992.

[2] C. T. Johnson and R.D. Lorenz, "Experimental identification of friction and its compensation in precise, position controlled mechanisms," IEEE Trans. Ind. Applicat., vol. 28, no. 6, pp. 1392-1398, 1992.

[3] C. Canudas, K. J. Astrom, and K. Braun, "Adaptive friction compensation in DC-motor drives," IEEE J. Robot., Automat., vol. RA-3, no. 6, pp. 681-685, 1987.

[4] J. Y. Hung, W. Gao, and J. C. Hung, "Variable structure control: A survey," IEEE Trans. Ind. Electron., vol. 40, no. [SI H. Butler, *G.* Honderd, and J. V. Amerongen, "Model reference adaptive control of a direct-drive DC motor," IEEE Mag. Contr. Sys., vol. 9, no. 1, pp. 80-84, 1989.

[6] L. A. Zadeh, "Fuzzy sets,'' Informat. Control, vol. 8, pp. 338-353,1996.

[7] L. A. Zadeh, "Outline of a new approach to the analysis complex systems and decision processes," IEEE Trans. Syst. Man Cybem., vol. SMC-3, pp. 28-44, 1973. 1. pp. 2-22, 1993. 338-353,1965.

[8] M. Chow and H. Tram, "Application of fuzzy logic technology for spatial load forecasting," IEEE Trans. Power Syst., vol. 12, no. 3, pp. 1360-1366, 1997.

[9] M. Chow, J. Zhu and H. Tram, "Application of fuzzy multi-objective decision making in spatial load forecasting," IEEE Trans. Power Syst., vol. 13, no. 3, pp. 1185 - 1190, 1998.

[10] H. Ying, W. Siler, and J. J. Buckley, "Fuzzy control theory: A nonlinear case," Automatica, vol. 26, no. 3, pp. 1185 - 1190, 1998. 513-520, 1990.

[11] M. Chow and A. Menozzi, "On the comparison of emerging and conventional techniques for DC motor control," Proc. IECON, pp. 1008-1013, 1992.

[12] J.-H. Kim, J.-H. Park, S.-W. Lee, and E. K. P. Chong, "A two-layered fuzzy logic controller for systems with dead zones," IEEE Trans. Ind. Electron., vol. 41. no. 2, pp. 155-162,1994

[13] J.-H. Kim, K.-C. Kim, and E. K. P. Chong, "Fuzzy precompensated PID controllers," LEEE Trans. Contr. *S*ys. Tech., vol. 2, no. 4, pp. 406-411, 1994.

[14] J. T. Teeter, M. Chow, and J. J. Brickley Jr., "A novel fuzzy friction compensation approach to improve the performance of a DC motor control system," IEEE Trans. Ind. Electron., vol. 43, no. 1, pp. 113-120, 1996.

[15] Y. Tipsuwan and M.Y. Chow "Fuzzy logic microcontroller implementation for DC motor speed control" IEEE Trans. pp. 1271-1276, 1999.

[16] J. Katupitiya, K. Bentley "Interfacing with C++ Programming Real-World Applications" School of Mechanical and Manufacturing Engineering The University of New South Wales Sydney NSW 2052, Australia, pp 189-201, 2006.