Fuzzy Logic Microcontroller Implementation for DC Motor Speed Control

Yadyum Tipauwan
IEEE Student Member
ytipauw@unity.ncsu.edu

Mo-Yuen Chow
IEEE Senior Member
chow@eos.ncsu.edu

Department of Electrical and Computer Engineering
Box 7911, North Carolina State University
Raleigh, NC 27695-7911, USA

ABSTRACT: This paper described an alternative method to implement a fuzzy logic speed controller for a DC motor using a fuzzy logic microcontroller. The design, implementation, and experimental results on load and no-load conditions are presented. The controller can be implemented by using only a small amount of components and easily improved to be an adaptive fuzzy controller. The controller also provides high performance with compact size and low cost.

I. INTRODUCTION

DC motors used in many applications such as steel rolling mills, electric trains, and robotic manipulators require speed controllers to perform tasks. 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 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 PI 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 a FLC [11, 12] and fuzzy compensator [13, 14] showed fuzzy logic can reduce the effects of nonlinearity in a DC motor and improve the performance of a controller.

A FLC has been implemented on many platforms such as digital signal processor (DSP) [15], PC [16], or off-the-shelf microcontroller [17]. These platforms have different advantages and disadvantages. The FLC developed on DSP or PC can quickly process fuzzy computation to generate control efforts, but the physical size of the system may too big and quite expensive for a small DC motor application.

On the other hands, using an off-the-shelf microcontroller to implement a FLC is inexpensive and the physical size of the system is small, but the FLC requires longer processing time. One way to improve the response time in microcontroller implementation approach is to use a look-up table, but this method needs much more memory to store a table.

An alternative method to implement a FLC is using a fuzzy logic chip. The fuzzy logic chip is first developed in 1985 by Togai and Watanabe [18]. It has been developed and improved continuously to be a commercial fuzzy logic microcontroller by many companies. The main feature of this chip is its capability in hardware level to execute fuzzy computation. Fuzzy rules and membership functions are defined and stored in RAM or ROM by specific formats that make a designer’s job easier. This feature could reduce developing time and bypass the need of a high speed yet expensive system to develop a FLC. The designer can also utilize other features included in a fuzzy logic microcontroller to reduce the size and to improve the system performance.

In this paper, a FLC is implemented for DC motor speed control on a fuzzy logic microcontroller. Heuristic knowledge is applied to define fuzzy membership functions and rules. The membership functions and rules are modified after initially borrowing the knowledge from a PI controller developed from a simple linear model [11, 14]. The hardware interface circuit and software algorithm are described. The results from real-time experiments with load and no-load conditions are also included in this paper.

II. SYSTEM DESCRIPTION

A simple DC motor linear model is firstly used for the motor controller design as shown in equation (1).

\[
\begin{bmatrix}
R & K & 1 \\
L & L & 0 \\
K & J & 0
\end{bmatrix}
\begin{bmatrix}
u \\
I \\
\theta
\end{bmatrix}
+ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}
\] (1)

where control \( u \) is the armature voltage of the dc motor, state variable \( x = [i, \omega]^T \), where \( i \) and \( \omega \) are the armature current and shaft rotational speed, respectively, \( d \) is the disturbance on the DC motor system. All motor
parameters in this paper are obtained by standard system
identification as shown in Table 1.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_a$</td>
<td>Armature resistor</td>
<td>4.67 $\Omega$</td>
</tr>
<tr>
<td>$L_a$</td>
<td>Armature inductance</td>
<td>170e-3 $H$</td>
</tr>
<tr>
<td>$J$</td>
<td>Moment of inertia</td>
<td>42.6e-6 Kg-m²</td>
</tr>
<tr>
<td>$J_f$</td>
<td>Viscous-friction coefficient</td>
<td>47.3e-6N-m-rad/sec</td>
</tr>
<tr>
<td>$K_t$</td>
<td>Torque constant</td>
<td>14.7e-2N-m/A</td>
</tr>
<tr>
<td>$K_r$</td>
<td>Back-EMF constant</td>
<td>14.7e-3 V-sec/rad</td>
</tr>
</tbody>
</table>

An experiment, as shown in Fig 1, is set up to identify
the characteristic of the motor by measuring the real-time
data of shaft rotational speed and armature voltage by
decreasing the armature voltage from the rated voltage to
zero. The experimental result shows the actual motor in this
paper has several nonlinear characteristics including
deadzone, friction, and saturation, as shown in Fig 1. The
lowest armature voltage to overcome deadzone and static
friction for starting the motor is 4.5 V, while at least 4 V
armature voltage can sustain the rotation. The effect of
saturation obviously happens when the armature voltage is
higher than 11.3 V. The method to reduce these effects will
be discussed in section III.

![Fig 1. DC motor steady-state input-output relationship.](image)

### Experimental setup

The armature voltage for driving the motor spans
between -15 V and +15V. When the motor rotates, the
tachometer attached on the motor will generate the velocity
signal in +/- 13.5 V range. The limited maximum and
minimum voltage levels represent the highest angular
velocity at 500 rad/s in different direction.

### III. CONTROLLER DESIGN

While conventional controllers depend on the accuracy
of the system model and parameters, FLCs use a different
approach to control the DC motor speed. Instead of using a
system model, the operation of a FLC is based on heuristic
knowledge and linguistic description to perform a task. The
effects from inaccurate parameters and models are reduced
because a FLC does not require a system model. However,
building a FLC from the ground-up may not provide good
results or sometime even a worse result than a conventional
controller if there is not enough knowledge of the system.
Therefore, in this paper, the result from a PI controller is
initially borrowed as a-priori knowledge in the design
process. The performance of the FLC is then improved by
adjusting the rules and membership functions. These design
procedures are described as follows.

**Procedure 1: Defining inputs, outputs, and universe of discourse**

To apply heuristic knowledge in the FLC, inputs,
outputs and universe of discourse are defined first. The
inputs are the error ($E$) between the reference ($\omega_a$) and
actual speed ($\omega$), and the change in error ($CE$). The output
is the change in armature voltage ($CU$). The inputs and
output illustrated in Fig 2, are described by:

$$
E = \omega(k) - \omega_a(k)
$$

$$
CE = \omega(k) - \omega(k-1)
$$

$$
CU = u(k) - u(k-1)
$$

where $k$ is the time index.

![Fig 2. Block diagram of the FLC.](image)

As mentioned in section II, the maximum range of the
DC motor angular velocity that will not damage the motor
is +/- 500 rad/s. The possible error in the range is between
-1000 rad/s and 1000 rad/s. Therefore, the universe of
discourse of $E$ is defined to span between -1000 rad/s and
+1000 rad/s. The universe of discourse of the change in error
is based on the experiment data from the PI controller
design included in procedure 2, which gives the range of
ear change is +/- 5.5 rad/s. For the change in armature
voltage, the minimum and maximum defined value are -1.5
V and +1.5V respectively.

**Procedure 2: Defining fuzzy membership functions and rules**

To perform fuzzy computation, the inputs and outputs
must be converted from numerical or "crisp" value into
linguistic forms. The terms such as "Small" and "Big" are
used to quantize the inputs and outputs values to linguistic
values. In this paper, the linguistic terms that used to
represent the input and output values are defined by seven
fuzzy variables as shown in Table 2.

Fuzzy membership functions are used as tools to
convert crisp values to linguistic terms. A fuzzy
membership function can contain several fuzzy sets
depending on how many linguistic terms are used. Each
fuzzy set represents one linguistic term. In this paper, seven
fuzzy sets are obtained by applying the seven linguistic
terms. The number for indicating how much a crisp value
can be a member in each fuzzy set is called a degree of
membership. One crisp value can be converted to be
"partly" in many fuzzy sets, but the membership degree in
each fuzzy set may be different.

In order to define fuzzy membership function, designers
choose many different shapes based on their preference
or experience. The popular shapes are triangular and
TABLE 2
FUZZY LINGUISTIC TERMS

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB</td>
<td>Positive Big</td>
</tr>
<tr>
<td>PM</td>
<td>Positive Medium</td>
</tr>
<tr>
<td>PS</td>
<td>Positive Small</td>
</tr>
<tr>
<td>ZE</td>
<td>Zero</td>
</tr>
<tr>
<td>NS</td>
<td>Negative Small</td>
</tr>
<tr>
<td>NM</td>
<td>Negative Medium</td>
</tr>
<tr>
<td>NB</td>
<td>Negative Big</td>
</tr>
</tbody>
</table>

trapezoidal because these shapes are easy to represent designer's ideas and require low computation time. For performing fine-tuning to improve the efficient of the controller, the adjacent of each fuzzy set value should overlap about 25% [19]. The initial membership functions are illustrated in Fig 3.

![Membership Functions](image)

Fig 3. Initial membership functions.

Instead of using mathematical formulas, a FLC uses fuzzy rules to make a decision and generate the control effort. The rules are in the form of IF-THEN statements. For example, IF the error (E) is equal to positive big (PB) and the change in error (CE) is equal to positive medium (PM) THEN the change in armature voltage (CU) is negative medium (NM). The matters in defining rules are how many rules should be used and how to determine the relation in IF-THEN statements. Actually, the solutions are based on the experience of a designer or the previous knowledge of the system. The critical point is if there is not sufficient knowledge applied in the design, the result could be drastically bad. Therefore, in this paper, the knowledge from a PI controller is borrowed first to help define rules. The velocity transfer function and PI controller equation are [11]

\[
G_v(s) = \frac{a(s)}{e_v(s)} = \frac{K}{JLs^2 + (JL_e + JR_e)s + (JR_e + KK_v)}
\]

(5)

\[
a(k) = a(k-1) + \left( K_v \frac{K_T}{2} - K_v \right) (k - 1) \quad \text{where } K_v = 0.12 \text{ and } K_T = 0.264
\]

(6)

As observed from the PI control surface, the initial rules are constructed as showed in Table 3. Because the FLC uses the knowledge from the PI controller, the performance obtained from the FLC is similar to the PI controller. The efficiency can be improved by adjusting the membership functions and rules in procedure 3.

TABLE 3
INITIAL RULES

<table>
<thead>
<tr>
<th>$E$</th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PS</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>ZE</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
<td>PB</td>
</tr>
<tr>
<td>NS</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
<td>PS</td>
<td>PM</td>
</tr>
<tr>
<td>NM</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>NS</td>
<td>ZE</td>
</tr>
<tr>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NM</td>
<td>NS</td>
<td>ZE</td>
</tr>
</tbody>
</table>

To send out the armature voltage output, the output in the form of fuzzy sets must be converted to a crisp value. This process is called defuzzification. In this paper, the center of gravity method is chosen. The formula of this method is

\[
z = \frac{\sum_{i=1}^{n} S_i F_i}{\sum_{i=1}^{n} F_i}
\]

(7)

where $z$ is the output from defuzzification, $S_i$ is the specific position at $i$th fuzzy set, and $F_i$ is the membership degree at the position [10].

Procedure 3: Adjusting fuzzy membership functions and rules

In order to improve the performance of the FLC, the rules and membership functions are adjusted. The membership functions are adjusted by making the area of membership function near ZE region narrower to produce finer control resolution. On the other hands, making the area far from ZE region wider gives faster control response. Also, the performance can be improved by changing the severity of the rules [11]. After adjusting the membership functions and rules, the final membership functions and rules are obtained as shown in Fig 4, and Table 4, respectively.

IV. CONTROLLER IMPLEMENTATION

A. Hardware Aspects

The fuzzy microcontroller, Motorola 68HC812A4 used for the implementation is 16-bit microcontroller with full
TABLE 4
FINAL RULES

<table>
<thead>
<tr>
<th>E</th>
<th>NB</th>
<th>NM</th>
<th>NS</th>
<th>ZE</th>
<th>PS</th>
<th>PM</th>
<th>PB</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PB</td>
<td>NM</td>
<td>NS</td>
<td>NS</td>
<td>NB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PM</td>
<td>NM</td>
<td>NM</td>
<td>NS</td>
<td>NB</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>PS</td>
<td>NB</td>
<td>NM</td>
<td>NM</td>
<td>ZE</td>
<td>PB</td>
<td>PB</td>
<td>PB</td>
</tr>
<tr>
<td>ZE</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>ZE</td>
<td>PM</td>
<td>PM</td>
<td>PM</td>
</tr>
<tr>
<td>NS</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>PS</td>
<td>PM</td>
<td>PM</td>
</tr>
<tr>
<td>NM</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>PS</td>
<td>PS</td>
<td>PS</td>
</tr>
<tr>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>NB</td>
<td>PS</td>
<td>PS</td>
<td>PS</td>
</tr>
</tbody>
</table>

16-bit data paths, 8-channel 16 bit timer, 8-channel 8-bit analog to digital converter (A/D), 4K EEPROM, 1K RAM, and multiple input/output ports. In this paper, 68HC812A4 runs at an internal clock speed of 8 MHz with an external 16 MHz oscillator. The special feature of 68HC812A4 is the fuzzy logic instruction set which can perform a fuzzy logic task.

The complete controller set is composed of Kevin Ross 68HC812A4 microcontroller board, input condition circuit, LCD display, and power supply. The block diagram of the controller, microcontroller board, and the complete setup controller box are shown in Fig 5(a), 5(b), and 5(c), respectively.

The controller receives the actual motor speed input in voltage form from the tachometer attached on the DC motor. This input is preprocessed by the input condition circuit for adjusting voltage offset and gain before converted to digital data and computing E and CE. After the controller executes fuzzy computation, it provides the TTL level PWM output signal to adjust the shaft angular velocity and direction to the reference speed that can be changed by using the keypad. The reference speed and speed error are displayed on LCD display, Optrex DMC-16204. The PWM signal is 3-time amplified by a level shifter circuit before sent to the motor driver. The schematic diagram of input condition and level shifter are shown in Fig 6 and 7 respectively.
the tachometer. The inverting amplifier is used for scaling the +/–13.5V signal from the first stage to the +/–2.5V signal. This signal is buffered by the third stage buffer. This buffer works similarly to the first stage buffer, but input is from the second stage and the output is sent to the fourth stage. Finally, the signal is adjusted to the 0-5V signal that could be applied for A/D unit on the 68HC812A4. In this circuit, the Burr-Brown OPA132 Op-Amp is chosen for the first and second stage because of its ability to handle high voltage. For the third and fourth stage, the Burr-Brown OPA272 is selected because it provides low noise level and inherently low offset voltage.

The level shifter shifts the PWM signals at port F of the microcontroller from the TTL level to the +/- 15V level that is the motor driver input level. For this propose, ICL7667 MOSFET driver is selected.

B. Software Aspects

A speed control and direction control of the motor is used in this paper to illustrate the FLC design. In this implementation, the controller software program is separated to two parts. The first part is the main program running as an infinite loop. This loop contains procedures performing different tasks. To setup the reference speed and direction, the program has to monitor the keypad. When the reference speed and direction are assigned via keypad, the program obtains the current speed from the A/D unit and computes E and CE to apply with the fuzzy computation routine. After the defuzzification process, the duty cycle of the PWM signal and the direction are set via the interrupt service routines, and then the current speed and error shown on the LCD display are updated.

The second part is composed of two timer interrupts. The timers are set to perform as counter. The first timer represents the frequency of the PWM signal and remains constant at 14.5 kHz since initialized. When the timer is expired, the first interrupt service routine is invoked to set the direction of the DC motor and to reload initial value on second timer to set the duty cycle of the PWM signal. The duty cycle and direction of the PWM signal have been determined by the fuzzy logic sequence in the main program. When second timer is expired, the second interrupt service routine is called to turn off the PWM output for starting the next cycle. The flowchart of the operation is shown in Fig 8.

To perform these tasks efficiently, the implemented program on FLC has to operate at high speed to minimize the response time and the size of the program must be small enough to store in on-chip memory. Therefore, the program is coded by using assembly language. The membership functions and rules are stored with the programs on ROM, and then the membership functions are transferred to the RAM for easy modification in future research.

Fig 7. Level shifter circuit.

Fig 8. Flowchart of the algorithms running on the FLC.

V. EXPERIMENTAL RESULTS

In this section, an experiment is set up to demonstrate the performance of the FLC. The controller is tested on different velocity tracking performance both load and no-load conditions with a given speed. The motor load is generated by applying a magnetic load on the rotational disk attached on the motor shaft. This procedure is equivalent to adding friction load to the motor. In this paper, the motor is operated at 93 rads. The voltage signal from the tachometer without load in forward direction is shown in Fig 9(a) and the real-time PWM signal is illustrated in Fig 9(b), respectively.

The results show the FLC could regulate the angular velocity of the DC motor in no-load condition by adapting the duty cycle of the PWM signal. The results of the DC motor operated at the same speed under load condition are shown in Fig 9(c) and 9(d). These results shows even the load is applied, the DC motor can be regulated by the controller and still rotates at the same speed with the duty cycle of the PWM signal extended to maintain the required speed.

VI. CONCLUSION

This paper has demonstrated the implementation of a FLC for the velocity control of a DC motor by using a fuzzy microcontroller. The FLC is easy to implement and requires a small amount of inexpensive components in compact size. The controller showed good velocity tracking performance under load and no-load condition. The size of the FLC can be reduced for a smaller application by removing some redundant components such as LCD.
Fig 9. (a) Shifted-positive PWM output without load, (b) Tachometer output from (a), (c) Shifted-positive PWM output with load, (d) Tachometer output from (b).

display or redesigning the board layout. Because the memberships are stored in a small size of RAM, the controller performance could also be easily improved by adding the feature of neuro-fuzzy or adaptive fuzzy logic algorithms.

ACKNOWLEDGMENT

The authors are grateful to Scott Elliot, Jeff Strang, Daniel Wang and Kevin Ross for their help in this project.

REFERENCES


