Teaching Concepts in Fuzzy Logic Using Low Cost Robots, PDAs, and Custom Software

Abraham L. Howell, Roy T.R. McGrann, and Richard R.Eckert
Binghamton University (SUNY), Mechanical Engineering and Computer Science Department,
Binghamton, NY 13902 abe@abotics, mcgrann@binghamton.edu, reckert@binghamton.edu

Abstract - **Fuzzy logic is a topic traditionally taught in artificial intelligence, machine learning, and robotics courses. Students receive the necessary mathematical and theoretical foundation in lecture format. The final learning experience may require that students create and code their own fuzzy logic application that solves a real world problem. This can be an issue when the target is a Bioengineering course that introduces classical control theory, fuzzy logic, neural networks, genetic algorithms and genetic programming through the use of a low cost robot, Personal Digital Assistant (PDA) handheld computer, and custom PDA software. In this course, the concepts and theories discussed in lecture are reinforced and extended in a corresponding laboratory through the use of wireless robots and PDAs. Fuzzy logic libraries and software modules for laptops and desktop computers are readily available, however, when it comes to handheld computers no such libraries exist. Students are able to spend more time experimenting with different fuzzy logic controllers when a custom fuzzy logic library and PDA graphical user interface are utilized. In this paper we introduce and discuss a unique low cost wireless robot, a custom fuzzy logic library, a custom fuzzy logic GUI for the PDA, and the implementation results for the fuzzy logic section in a newly created Bioengineering course. Diagnostic and summative assessment in the form of a pre-test and post-test was administered for each section of the course, however, only the results for the fuzzy logic section will be provided.**

Index Terms – Bioengineering, fuzzy logic, low cost educational robot, handheld computer.

INTRODUCTION

Fuzzy logic is powerful and useful tool that can benefit students of all disciplines but more so in engineering and science courses. Introducing the mathematical and theoretical foundations of fuzzy logic is typically the first step in teaching fuzzy logic. After learning the basics, students are required to work through various problems by hand. Having students use fuzzy logic to solve a real world problem can be good way to have students extend and reinforce their classroom knowledge. This step can be difficult for students and educators alike. There are numerous issues that can arise when attempting this final step. Students may not have the necessary programming background that is required for the creation and coding of a fuzzy logic application. There are numerous readily available commercial and academic software packages that can alleviate this issue. Users of Matlab® can opt for the Fuzzy Logic module, which provides a rich tool set for creating, editing, and executing fuzzy logic systems (FLS). Similar issues occur when the target-programming platform is a PDA and more specifically a Pocket PC device. Pocket PC users do not have access to readily available software applications and therefore must create their own unique solutions.

This paper focuses on solving the issues associated with introducing and teaching fuzzy logic in the context of a Bioengineering course, BE470, Autonomous Agents. In this course students are exposed to classical control theory, fuzzy logic, neural networks, genetic algorithms, and genetic programming. During the fuzzy logic section students receive the necessary math and fuzzy logic set theory in lecture. Once students have mastered solving various fuzzy logic problems by hand, they are then ready to attend the corresponding laboratory. In lab, students use a low cost, Bluetooth-enabled mobile robot, BIObot, Pocket PC hand-held computer, and custom software to further explore the concepts and theories from lecture. The primary goal of this course is to introduce students to the above-listed topics and then use BIObot and a Pocket PC as a teaching tool that helps to bridge the gap between lecture and the real world. A secondary requirement was to eliminate programming and have the students configure and execute various fuzzy logic control systems using the Pocket PC and an appropriate Graphical User Interface (GUI). Satisfying this requirement dictated the creation of a custom fuzzy logic library and GUI that would allow for the configuration and execution of a fuzzy logic controller in conjunction with BIObot. The low cost robot, custom fuzzy logic library/GUI, and course results will be discussed in this paper.

LOW COST ROBOT, BIOBOT

Shown in figure 1 is BIObot, a low cost autonomous fully programmable robot that can be controlled wirelessly using Bluetooth®, ZigBee™, or 802.11b/g [1]. In this paper we are using Bluetooth wireless since the chosen Pocket PCs contain built-in Bluetooth hardware [2]. The Pocket PC provides all control intelligence, however, it should be noted that BIObot could be programmed directly when the appropriate C, Basic, or assembly level compiler and PIC programmer is used. The Pocket PC sends ASCII character control commands across the wireless Bluetooth Serial Port Protocol and is able to command BIObot to move using open and closed loop motion, retrieve sensor readings and modify internal parameters.

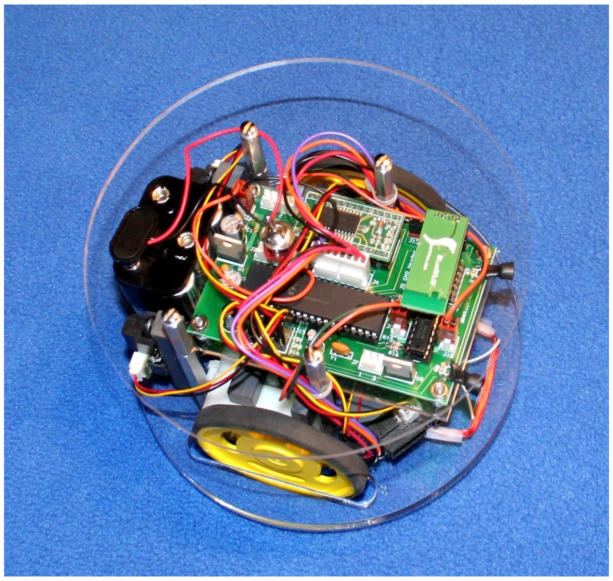
FIGURE 1
BIOBOT ROBOT SHOWN WITH BLUETOOTH

Quadrature wheel encoders, battery voltage level detection, five (5) Sharp GP2D120 Infrared sensors, two (2) light sensors, and Radio Frequency Identification (RFID) comprise the entire sensor array for BIObot. The wheel encoders provide wheel rotation feedback so that BIObot can perform Proportional-Integral-Derivative (PID) velocity and position control. Battery level detection can be used as a notification for battery replacement or charging when rechargeable batteries are utilized. Obstacle avoidance, wall following, and other motion behaviors can be created using the five GP2D120 sensors. Various light-based behaviors can be achieved using the two frontward facing light sensors. A multitude of possibilities exist for the RFID capability. BIObot is able to read and write to 125 kHz passive RFID tags [3]. The approximate read/write range is 0-5 inches.

CUSTOM FUZZY LOGIC LIBRARY

Creating programs for a Pocket PC requires advanced programming skill and knowledge of the .NET Compact Framework [4]. A custom C# fuzzy logic library (Fuzzy_Logic_Lib_PPC.dll) was created so that students would not need to spend time acquiring this knowledge. The library provides an easy-to-use interface for setting up, configuring, and executing fuzzy logic systems within the context of a Visual Studio® created Pocket PC application. To achieve these goals it was necessary to code the entire library from the ground up. Prior to beginning the code writing process, a good functional foundation had to be defined. The library must be able to work in conjunction with a GUI so that students would be sheltered from having to write code. The general flow associated with constructing any fuzzy logic system is to first define the input terminals, input terminal ranges, output terminals, and If-Then Rules [5]. This library must also be capable of accepting input from the GUI. The GUI extracts this information from the end-user. To relax end user interaction and ease library coding, it was decided that comma delimited text would be used to define four different text files.

```
input_terminals.txt
input_terminal_ranges.txt
output_terminals.txt
if_then_rules.txt.
```

Users define the contents of each of these text files and thereby setup and configure their fuzzy logic system. These text files can be edited using the GUI or any readily available text editor program on the Pocket PC. The library is responsible for reading in each file and interpreting the contents, which are then used to setup and configure the fuzzy logic system. Interpreting the input terminal, input terminal ranges, and output terminals text files is a relatively simple task. However, a unique system had to be created for the If-Then Rules. Below is a listing of four example text files for a two-input, two-output fuzzy logic system. This fuzzy logic system will be used to create a run-away behavior. In this run-away behavior the rear left and rear right infrared (IR) sensors will be the two inputs and the left and right wheel velocities are the two outputs to be controlled.

```
Input_terminals.txt
LB,MB,HB
LB,MB,HB

input_terminal_ranges.txt
-10000,30,60
30,60,80
60,80,10000
-10000,30,60
30,60,80
60,80,10000

output_terminals.txt
0,45,90
0,45,90

if_then_rules.txt
a=HB&b=HB~A=90&B=90
a=HB&b=MB~A=90&B=45
a=HB&b=LB~A=90&B=0
a=MB&b=HB~A=45&B=90
a=MB&b=MB~A=45&B=45
a=MB&b=LB~A=45&B=0
a=LB&b=HB~A=0&B=90
a=LB&b=MB~A=0&B=45
a=LB&b=LB~A=0&B=0
```

The linguistic variables used in the above example are low block (LB), medium block (MB), and high block (HB). The first line of linguistics is for the rear left IR sensor and the second line is for the rear right IR sensor. It should be noted that students have the freedom to define their linguistic variables using whichever naming convention they deem appropriate and the library will use these definitions. However, they must remain consistent from this point on otherwise the library will generate errors when the text files are loaded.

Next the input terminal ranges are defined as triangular membership functions. For this example, the ranges correspond to the rear left and rear right IR sensor readings. Each IR sensor returns a varying voltage level that is fed into a dedicated analog to digital capable pin on the robot's PIC16F877A microcontroller. The PIC samples each IR sensor at a fixed frequency to ensure that the most up-to-date reading is available. These readings are converted to an 8-bit digital value, which in turn corresponds to a block distance in inches, centimeters, etc. Even though the reading is 8-bit (0-255), sensor readings generally have a range of 0-150 and this is due to the internal circuitry of the GP2D120 IR sensor. This means that the manner in which a user defines the input terminal ranges will have a large impact on the performance of the system. The first three lines of the input_terminal_ranges.txt file are for the rear left IR and the remaining three define the rear right IR sensor. Each line defines a triangular membership function that corresponds to a linguistic variable. For example, the first line defines a triangular membership function for the rear left IR's LB linguistic variable. The current version of the fuzzy logic library only supports triangular membership functions, but this could easily be changed to accommodate trapezoidal or generalized bell.

From the output terminals file it can be seen that the Fuzzy Singleton method is being used for defuzzification. The Fuzzy Singleton method ensures that calculation complexity and time is kept to a minimum. Calculation time is critical since the Pocket PC is responsible for supplying all control to the robot via the wireless Bluetooth connection. Unfortunately, there is approximately a 120ms time delay that is due to the overhead associated with the wireless Bluetooth communication. As the fuzzy logic calculation time increases the fidelity of robot control decreases. The first line in the output_terminals.txt file corresponds to the left wheel open-loop velocity and the second line is for the right wheel open-loop velocity. The valid range for open-loop wheel velocity is –90 to 90.

Interpreting the contents of the If-Then Rules file requires an understanding of the unique syntax that was created just for this library. All lowercase letters refer to inputs and uppercase letters correspond to system outputs. The "="symbol refers to a logical equal. The "&" symbol refers to the logical AND. Finally, the "~" symbol corresponds to the logical THEN. Additionally, the logical IF is not shown at the beginning of each line, but it is implied. Now the first line of the If-Then Rules file can be interpreted as follows: if input#0 is equal to a high block and input#1 is equal to a high block then output#0 is 90 and output#1 is 90.

Once the four text files are defined and read in by the library, the user can manually supply a set of inputs, which are then used to calculate the corresponding outputs. The library is responsible for any pre and post-processing. This means that the resulting outputs can be fed directly to BIObot and used to control the velocity of the right and left wheel respectively.

CUSTOM FUZZY LOGIC GUI

After creating and defining the four comma delimited text files, students need to be able to pass these files to the fuzzy logic library, map the appropriate BIObot sensors as inputs to the fuzzy logic system, Bluetooth connect/pair with a BIObot, and finally execute the system live to observe how BIObot behaves.

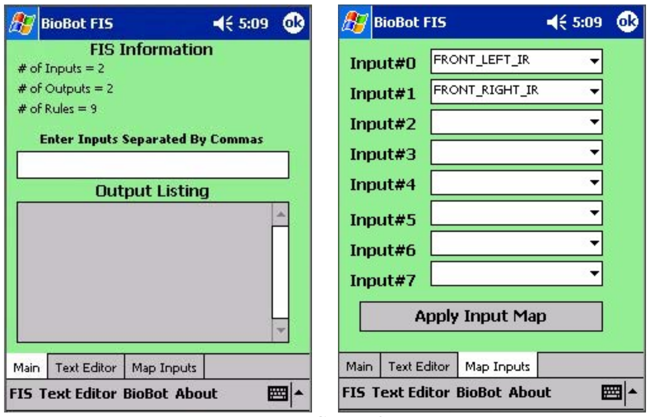

FIGURE 2
SCREENSHOT OF FUZZY LOGIC GUI

A GUI was created to ease student interaction with the library and eliminate the need for student code writing. A screenshot of the Fuzzy Logic GUI can be seen in figure 2. This program is named Fuzzy Logic Inference System (FIS). The four text files can be read into the application by tapping the FIS menu, selecting Read Input Terminals, and finally browsing for the appropriate input_terminals.txt file. Once the files are read by the application, the system will display all related system information such as the number of inputs, outputs and rules. Next, students need to map sensors to inputs so that the application knows which sensor readings need to be passed to the library. After selecting the appropriate sensor for each input from the drop-down boxes, the input map can be applied as shown in figure 2. Finally, the created and configured fuzzy logic controller is ready to serve as BIObot's source of artificial intelligence. After testing their system live, students can quickly go back and make changes to any or all of their configuration files, refresh the library and quickly get back to testing and observing their robot behavior. This simple interaction loop helps to keep the student's focus on exploring and observing the various aspects of fuzzy logic control.

The entire source-code and compiled executables will be made freely available. Contact the authors to request the custom fuzzy logic library and GUI.

COURSE LABORATORIES

Prior to the first fuzzy logic lab, students attend three lectures that provide a mathematical and theoretical background in fuzzy logic. Additionally, the students must also complete a fuzzy logic homework assignment. In the first lab, students are given four fuzzy logic text files that define a run away behavior. The four files are preloaded when the FIS application is installed on the Pocket PC. Students must use the FIS program and these files to configure a fuzzy logic system, connect to BIObot, and finally run the system live on BIObot. Doing this allows the students to become familiar with the FIS program. The next step is to modify the four FIS text files so that BIObot's run away behavior is more intelligent and utilizes the front IR sensor to inhibit the run away behavior. When the front IR sensor observes a high-block situation the run away behavior must be stopped and likewise, BIObot must halt all movement until the high-block condition subsides. This modification requires that all four FIS files receive the appropriate changes and additions.

In the second lab, students create a total of four new fuzzy logic controlled behaviors: light tracking, obstacle avoidance, wall following, and finally a student created behavior. For the light tracking behavior, students are given a flashlight, which is to be used as the light source. It is left up to the students to determine the appropriate sensor values based upon the ambient light conditions in lab. Similar student decisions must be made for the remaining three behaviors. Students are encouraged to use the run away configuration files as a template for their new behaviors since these will require significant modification to achieve the different behaviors, but still reduce some user effort.

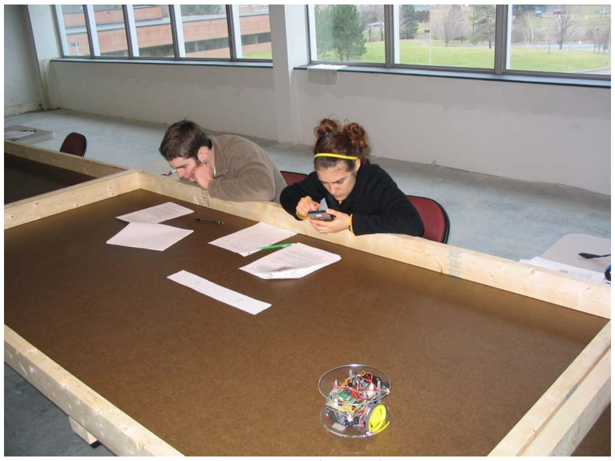
FIGURE 3
ROBOT LAB ENVIRONMENT

Lab groups consist of two to three students and each group is given its own robot, Pocket PC, and a basic four-foot by eight-foot configurable robot environment as shown in figure 3. Various obstacles and other robot environment accessories are available to the students. Students are encouraged to modify their environment as they deem necessary for the specific lab. The robot environments are semi-portable and can be moved from the lab to another room or location. Long distance transportation does require some disassembly and the ability to accommodate a four by eight foot sheet of ¼" thick pressed composite.

FUZZY LOGIC SECTION LEARNING ASSESSMENT RESULTS

A pre-test was administered at the start of the fuzzy logic section, so that the current level of student knowledge with respect to fuzzy logic could be measured. A post-test was administered at the conclusion of this section. Below is a listing of the questions from the pre-test and post-test. The pre-test and post-test contain the same questions.

1. What are fuzzy sets and how do they differ from crisp sets?

2. What is a linguistic variable?

3. What does it mean when we say, "There is nothing fuzzy about fuzzy sets"?

4. What is a characteristic function for crisp and for fuzzy sets?

5. What is the fuzzy singleton method?

A total of nineteen students participated in this course. However, only fifteen students completed the pre-test and post-test for this section of the course. From figure 4 a comparison of post-test and pre-test results reveals a statistically significant increase in student scores for each of the five test questions. This is not surprising since several students mentioned that using BIObot and the PDA in lab helped to solidify the concepts of fuzzy logic from lecture. One student mentioned that using the fuzzy logic GUI to create a run away behavior for the robot helped clarify the process of defining the input linguistic ranges and the If-Then-Rules. With these results one can conclude that the combination of lecture and robot lab contributed to the increase in student scores on the post-test. To isolate the contribution of each requires the administration of a post-test immediately after students attend all the fuzzy logic lectures. A second post-test administered immediately after the BIObot labs would help to provide a measure of the contribution made by lab. This technique was not implemented in this first course offering, but will be addressed in future offerings.

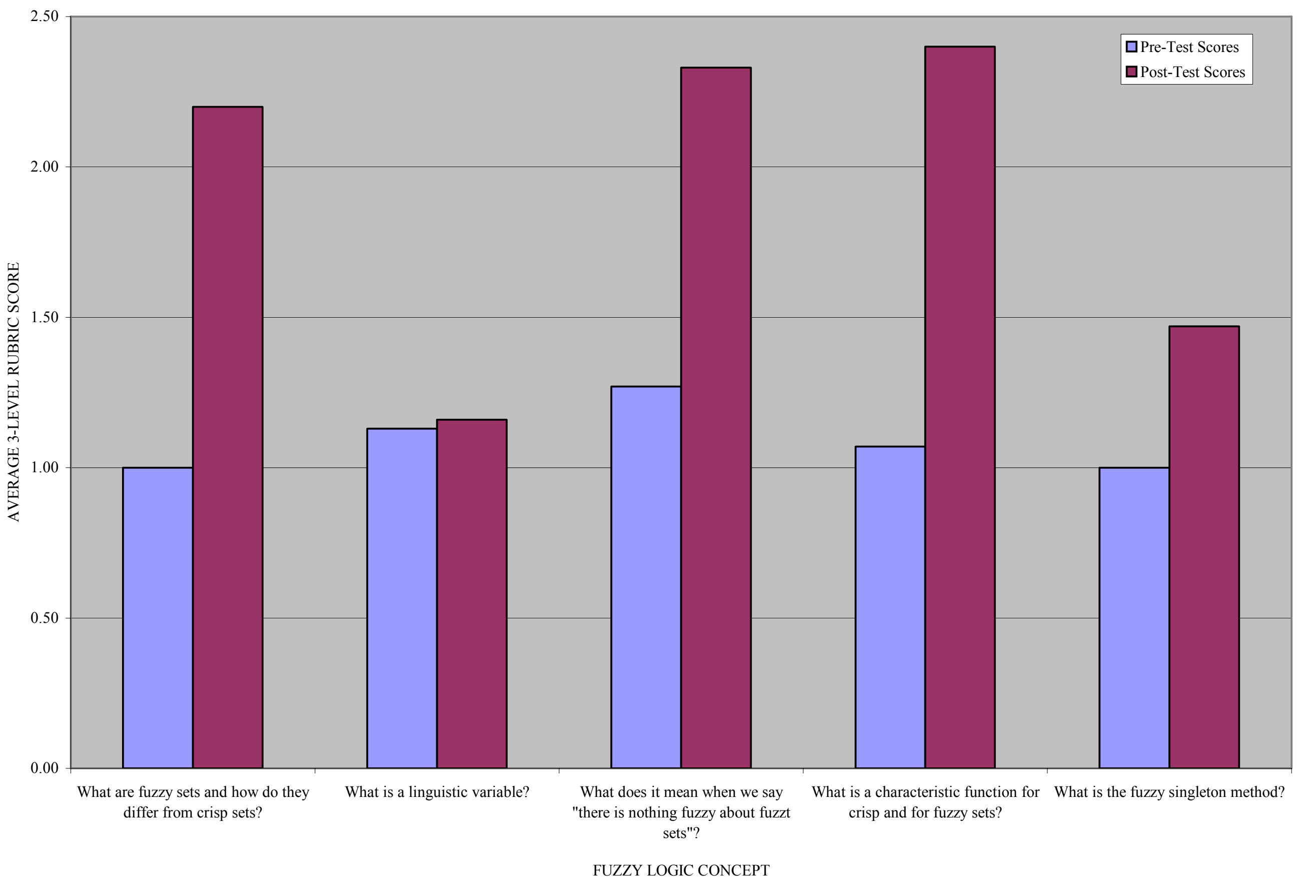

FIGURE 4
PRE-TEST AND POST-TEST SCORES FOR FUZZY LOGIC SECTION.

CONCLUSION

A low cost wireless robot, custom fuzzy logic library, custom fuzzy logic GUI for a PDA, and the implementation results for the fuzzy logic lectures and laboratories in a newly created Bioengineering course have been presented. The combination of a low cost robot and PDA creates a flexible and capable teaching tool. The use of this teaching tool in a Bioengineering course is unique, since the focus is on experimenting with fuzzy logic and not on writing code for the robot and PDA. Results from the pretest and posttest indicate an increase in student scores. One can conclude that the lectures and/or labs contributed to this increase. However, administering one post-test after the fuzzy logic lectures and then a second after the labs would provide additional insight with regard to the contribution from each.

ACKNOWLEDGMENT

This project was funded by NSF CCLI EMD, DUE-0442887.

REFERENCES

[1] Howell, A., Laramee, C., McGrann, R., Way, E., “Autonomous Robots as a Generic Teaching Tool”, Proceedings of ASEE/IEEE Frontiers in Education Conference (FIE), San Diego, CA, 28-31 October 2006, Paper#1674.

[2] https://www.bluetooth.org

[3] http://www.sonmicro.com/125/info125.php

[4] Wigley, A., Wheelwright, S., "Microsoft .NET COMPACT FRAMEWORK", *book, Microsoft Press, 2003.*

[5] Jang, J, S.R., Sun, C.-T., Mizutani, E., “Neuro-Fuzzy AND Soft Computing”, *book*, Prentice-Hall, 1997.