RESULTS OF USING A LOW COST, FLEXIBLE ROBOT IN A MICROCONTROLLERS AND ROBOTICS COURSE

Abraham Howell, Richard Eckert, Roy McGrann
Binghamton University — SUNY

Abstract

This paper discusses the results of using a low cost, flexible robot in a computer science microcontrollers and robotics course. Such a course should introduce students to the fundamentals of microcontrollers and robotics. To achieve this goal, students must understand and interact with a microcontroller at both low and high levels. Additionally, a suitable robot platform must be available for the robotics section of the course, so that students can experiment with the concepts and theoretical material discussed in lecture. Historically, this course made use of a popular microcontroller development kit for the first half and then transitioned to a well-known robotics kit for the second half of the course. A disconnect between the first and second half was created since students were required to learn two different systems. It would be more advantageous if the students worked with a single platform throughout the entire course. This would provide students with additional hands-on interaction and time to reinforce the concepts and theories through direct experimentation with real world hardware. A low cost, flexible mobile robot was integrated into the targeted course through the development of three laboratory modules. Through the lab experiments, students directly interacted with the robot's microcontroller through the use of low-level assembly programming. At the end of the course, students created high-level programs in Visual C# using Microsoft Visual Studio® 2005. Their programs controlled the robot via a wireless Bluetooth® connection and provided high-level intelligence, such as obstacle avoidance and light tracking. The robot's ability to read and write radio-frequency identification (RFID) tags is a unique feature and opens up the realm of possible experiments. The course, low cost robot, three developed laboratory modules, and results of the student evaluations are discussed in this paper.

Overview of Microcontrollers and Robotics Course

Several years ago the Computer Science Department in the Watson School of Engineering and Applied Science at Binghamton University we designed and began to offer an upper-division undergraduate course entitled Microcontrollers and Robotics[1]. This was done in response to the reality that an important application of computer science is that of using embedded microcomputers to control hardware systems. These are ubiquitous in electronic devices found almost everywhere in modern society, and, in particular, in embedded control systems and robots used in industry, science, and defense. Many modern devices – as common as microwave ovens or automobiles, to machines that automate and control the positioning of electronic components on printed circuit boards, to pilot-less airplanes used to spy on and/or deliver weapon systems to potential enemy targets, to robots that search for survivors in mining or other disasters, to something as exotic as the Mars Sojourner Rover robot – use embedded microcontrollers to control hardware. We felt that it was important that computer science students have the opportunity to learn about these devices, how they work, and how to design and program them.

Our course emphasizes those aspects of microcontroller-based control systems and robotics that are most closely related to computer science. These aspects include the following:

- Architectures and instruction sets of microcontrollers
- Interfacing a microcontroller with memory and I/O
- I/O techniques (serial, parallel, interrupt-driven, digital to analog conversion, analog to digital conversion)
- Microcontroller programming languages and techniques
- The use of timers in responding to and controlling real-time situations

The fundamentals learned are then applied in the context of designing, building, and programming autonomous, mobile robots whose motors and sensors are controlled by a microcontroller-based system. The robots are programmed to perform such “intelligent” tasks as following a path, avoiding obstacles, seeking and retrieving objects, and communicating with other robots. Several ideas from the fields of behavior control architectures, computer vision, and robot navigation are presented and applied where appropriate. Robots designed, built, and programmed by students participate in a competition at the end of the course.

The course is divided into two sections: one on microcontrollers and the other on robotics. In the first section students work with Microchip Technology, Inc.'s PIC18F452 microcontroller and an inexpensive trainer called the QwikFlash[2] that contains the microcontroller wired up to several switches, LEDs, a potentiometer, a liquid crystal display (LCD), and other devices. The QwikFlash board can be connected to a QwikProto breadboard where students can build prototype circuits that control many different kinds of hardware devices. In this first part of the course students perform experiments in which they design circuits controlled by programs they create on a PC and upload to the flash memory of the PIC18F452 using a serial link to the PIC. Program upload is facilitated by a QwikBug monitor program that is burned into the PIC and a terminal emulator such as the Tera Term Pro running on the PC. The experiments performed by the students explore the considerable capabilities of the PIC, receive digital input from switches, output to LEDs, display characters and numeric values on the LCD, control motors using pulse width modulation, convert analog input signals from sensors to digital values that can be processed by the microcontroller, and investigate the interrupt capabilities of the PIC. In this first part of the course essentially all of the programming is done at the assembly language level using Microchip's free MPLAB interactive development environment.

Initially the second part of the course was centered on the LEGO Mindstorms Robotic Invention System (RIS), whose microcontoller and other control circuitry is embedded in a large LEGO brick called the RCX[3]. This approach had the advantage of permitting creative designs of different kinds of robots possessing many different capabilities. The programming language used to create most of the robot control programs was a subset of C called NQC (Not Quite C), which contains instructions that make it relatively straightforward to control the RCX's sensor inputs, motor outputs, timers, IR communications, LCD display, and other systems. A tiny version of Java called LEJOS was also used for some of the RCX program development toward the end of the course. This free software consists mainly of a Virtual Machine for the execution of Java byte code on the RCX microcontroller, an API for RCX programming on top of this virtual machine, and additional software tools. These tools include a LEJOS vision system that can control a digital camera mounted on the RCX and that is connected to a PC, a communications package permitting IR communication between the RCX and the PC, a navigation control module, and a subsumption architecture behavior control module.

The transition between the two parts of the course was made by having the students perform an experiment in which they build a robot from LEGO bricks upon which are mounted the QwikFlash and QwikProto boards. We called this robot the "PIC-Brick". In this experiment two LEGO RIS motors controlled by an H-Bridge on the QwikProto board interfaced to the PIC microcontroller make the robot move, and two LEGO RIS touch sensors generate interrupts that can cause the robot to move away from obstructions. Figure 1 is a photograph of one of the student PIC-Bricks. In this experiment, once again the programming is done in PIC assembly language using the MPLAB development system.

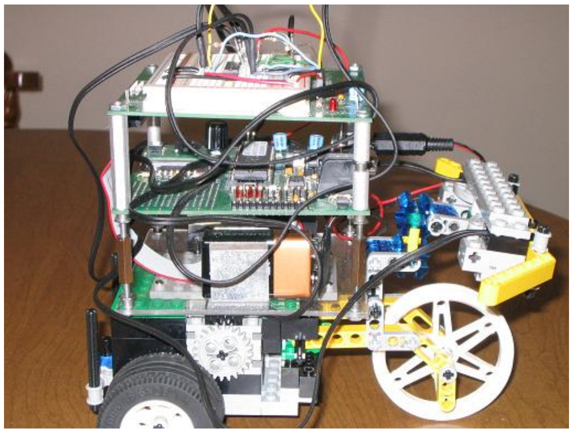

Figure 1. PIC-Brick robot.

Although this two-section organization of the course seemed to work reasonably well, we noticed something of a disconnect between the first and second parts. Both the hardware and the programming language/platforms were different. Ideally we wanted to maintain the low-level nature of the first part of the course because of the insights provided on how microcontrollers really work in the control of hardware devices. But we thought it would be better if somehow the second part of the course would use the same microcontroller mounted on a versatile robot platform, but be programmed at a higher level. That way we could avoid making the students learn so many different hardware and software systems and facilitate the development of powerful control programs. This led to the design and use of BIObot.

Low Cost Robot, BIObot

BIObot is a low cost, fully programmable, autonomous robot that can be controlled wirelessly using Bluetooth® or ZigBee™ or can be programmed locally at the microcontroller level using the appropriate C, Basic, or assembly level compiler and PIC programmer[4]. Figure 2 shows a BIObot with Bluetooth. However, a low-cost XBee ZigBee module can be used to establish large robot networks or control a swarm of robots from one central computer[5]. A Bluetooth or ZigBee-equipped computer can provide all high-level intelligence. By sending control commands across the wireless communication link, a computer is able to command BIObot to move, retrieve sensor readings and modify internal parameters. The brains of BIObot reside in the onboard controller, Autonomous roBot controllEr (A.B.E.)[6]. The A.B.E. board provides a serial based command library, so that built-in functions and parameters can be easily accessed. A PIC18F452 is at the heart of the A.B.E. and operates at 20MHz while executing a specially designed firmware that serves up a serial based command library. All the PIC source code is available and can be modified as needed, however, a copy of the CCS C compiler and a PIC programmer is required[7]. A programming header on the A.B.E. board allows for the connection of an In-Circuit Programmer/Debugger (ICD). When using the CCS C compiler IDE and ICD, users can set break points, monitor or change variables, and step or step-over lines of code. A logging window is provided, so that specified data can be written to the log window or to a spreadsheet compatible file. New programs can be compiled and uploaded to the PIC via in-circuit serial programming when the ICD is used. These features help students to create code quickly and then debug directly on the robot.

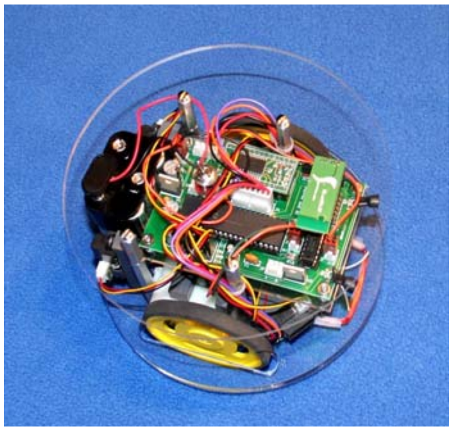

Figure 2. BIObot robot shown with Bluetooth.

BIObot has several different types of sensors that allow it to interact with the environment. The PIC18F452 controls two low cost gear motors using an on-chip H-Bridge and its two hardware PWM channels. The following list comprises the entire sensor array for BIObot.

- Quadrature wheel encoders
- Battery voltage measurement
- (5) Sharp GP2D120 Infrared sensors
- (2) Light sensors
- Radio Frequency Identification (RFID) system

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. Obstacle avoidance, wall following, and other motion behaviors can be achieved using the (5)-GP2D120 sensors. Various light-based behaviors can be created using the frontward facing light sensors. A multitude of possibilities exist for BIObot's RFID capability. Passive 125 kHz RFID tags can be read or written when in the range of 0-5 inches from the antenna.

BIObot Based Laboratory Modules

In the course, Experiment 10, Serial I/O, is the first to leverage BIObot and the QwikFlash PIC18F452 development board. The first half of the course focuses on assembly language programming with the QwikFlash board, which also utilizes the PIC18F452. In this experiment we build upon the student's PIC18F452 assembly knowledge while introducing BIObot for the first time. The purpose of Experiment 10 is for students to learn how to setup serial communication between the PIC18F542 on the QwikFlash board and the PIC18F452 of BIObot. After wiring the two devices together, students are required to write assembly code to communicate serially with BIObot and retrieve all the analog sensor readings, parse the individual readings, and then display a specific sensor reading on the QwikFlash LCD.

A transition from assembly to C# programming occurs in Experiment 11, Wireless Control. A Bluetooth equipped desktop computer with Visual Studio® 2005 is used to develop a C# graphical user interface (GUI) for controlling BIObot. To help with high-level software development, a custom C# .NET library (abe_functions.dll) is available and facilitates serial communication with BIObot's A.B.E. board. Students quickly and easily create code using Microsoft's freely available C# Express. The abe_functions.dll encapsulates BIObot's functionality and manages all serial port communication so the user can focus on creating high-level intelligence for BIObot. After adding the library as a reference to their C# project, students have immediate access to all of the BIObot functions. The student GUIs must control BIObot using open loop velocity, closed loop velocity, or position control. A user should also be able to enter the left and right motor velocities or number of encoder ticks depending upon which control methodology is selected. Additionally, the GUI must be able to retrieve and display the A/D sensor readings and read an in-range RFID tag and display the data in both ASCII and hex format. A custom C# .NET library (abe_functions.dll) is provided so that students only need to add this as a reference to their project.

In the final BIObot experiment, Obstacle Avoidance and Light Tracking, students are required to code obstacle avoidance and light tracking behaviors for BIObot using C#. A user must be able to select from the following behaviors: obstacle avoidance, light tracking, or obstacle avoidance with light tracking. When the last option is selected, the obstacle avoidance must maintain priority over the light tracking behavior. Prior to this, a lecture is provided so that students receive an introduction to the subsumption control architecture, however, it is up to the students to implement this in their C# program.

Student Evaluations

At the conclusion of each BIObot experiment a student evaluation was administered. These evaluations are feedback mechanisms that measure student interest, and the effect of integrating BIObot into the course. The results of these can be used to adjust the corresponding experiment and lecture material so that it is better aligned with the course objectives. The questions for each experiment are listed below, along with a discussion of the students' responses.

Experiment 10 Questions

1. I enjoyed working with the BIObot robot in this lab.
2. BIObot helped to clarify the concepts associated with this lab.
3. It was easy to interface BIObot with the QwikFlash microcontroller board.
4. BIObot helped me to better understand real-world hardware/software interaction.
5. Working with BIObot increased my interest in this lab.
6. BIObot helped me to better understand serial communication.
7. I recommend using BIObot in this lab for future course offerings.
8. I would like to work with BIObot again.

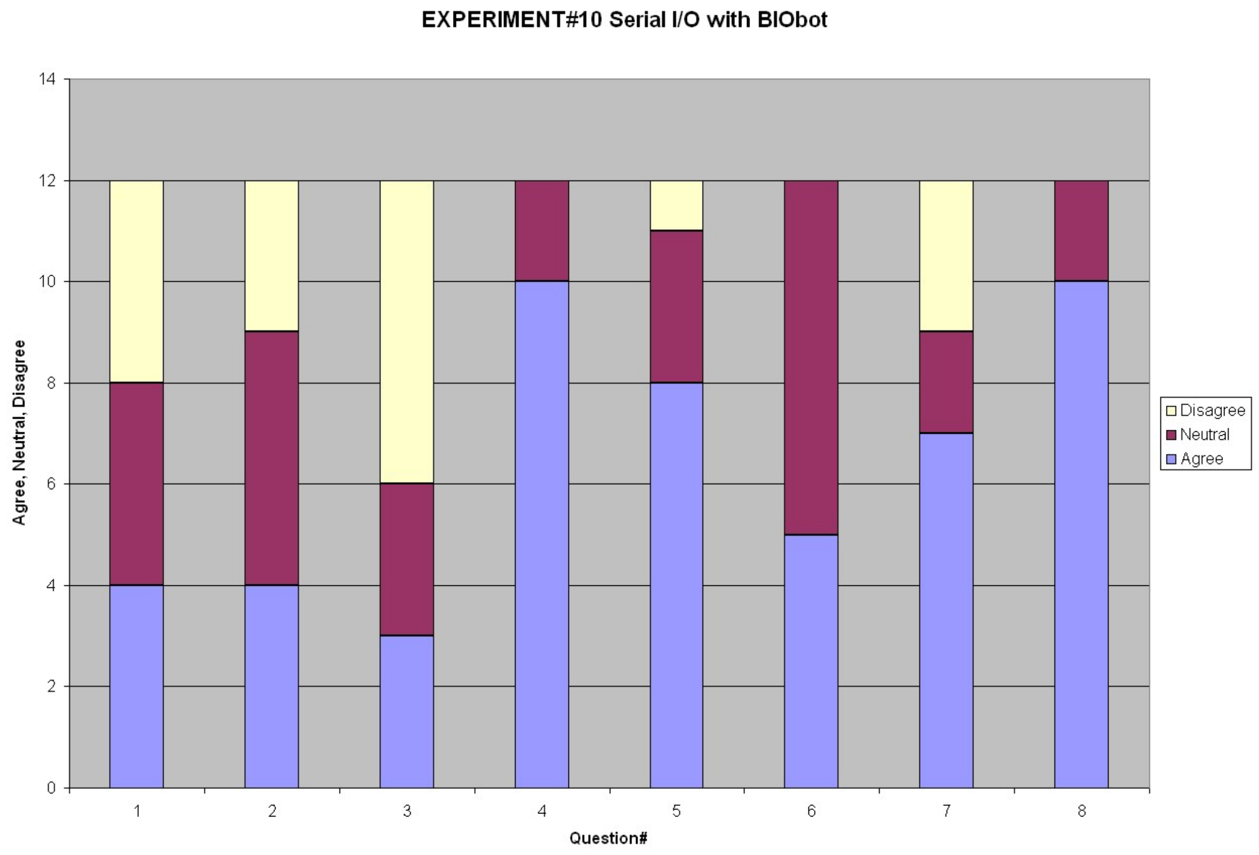

Figure 3. Results from Experiment 10 Student Evaluation.

From figure 3 it can be seen that student agreement with questions 1, 2, and 3 is relatively low in Experiment 10 and this was due to a technical issue surrounding BIObot's UART configuration. Future course offerings will focus more on the details of interfacing a software-UART to a hardware-UART so that this issue is eliminated. However, having this difficulty helped students to realize the importance of real world hardware to software interaction, question 4. Even though students had to work through the experiment with minor technical issues they still recommended that we use BIObot in future course offerings. Students also indicate a strong interest in working with BIObot again.

<u>Experiment 11 Questions</u>

1. I enjoyed working with the BIObot robot in this lab.
2. BIObot helped to clarify the concepts associated with this lab.
3. Connecting BIObot with a desktop computer was not difficult.
4. BIObot helped me to better understand wireless robot control.
5. Working with BIObot increased my interest in this lab.
6. Programming BIObot was not difficult and it helped me to better understand the issues associated with wireless control.
7. I recommend using BIObot in this lab for future course offerings.
8. After working with BIObot, I am more interested in learning about wireless control.

In Experiment 11 a transition from PIC assembly to C# was made. The majority of the students welcomed this transition since the entire first half of the course focuses on assembly. According to figure 4, student responses indicate strong agreement with all eight questions. It is good to see that BIObot is able to pique student interest while also helping to clarify concepts in wireless control. Ten of

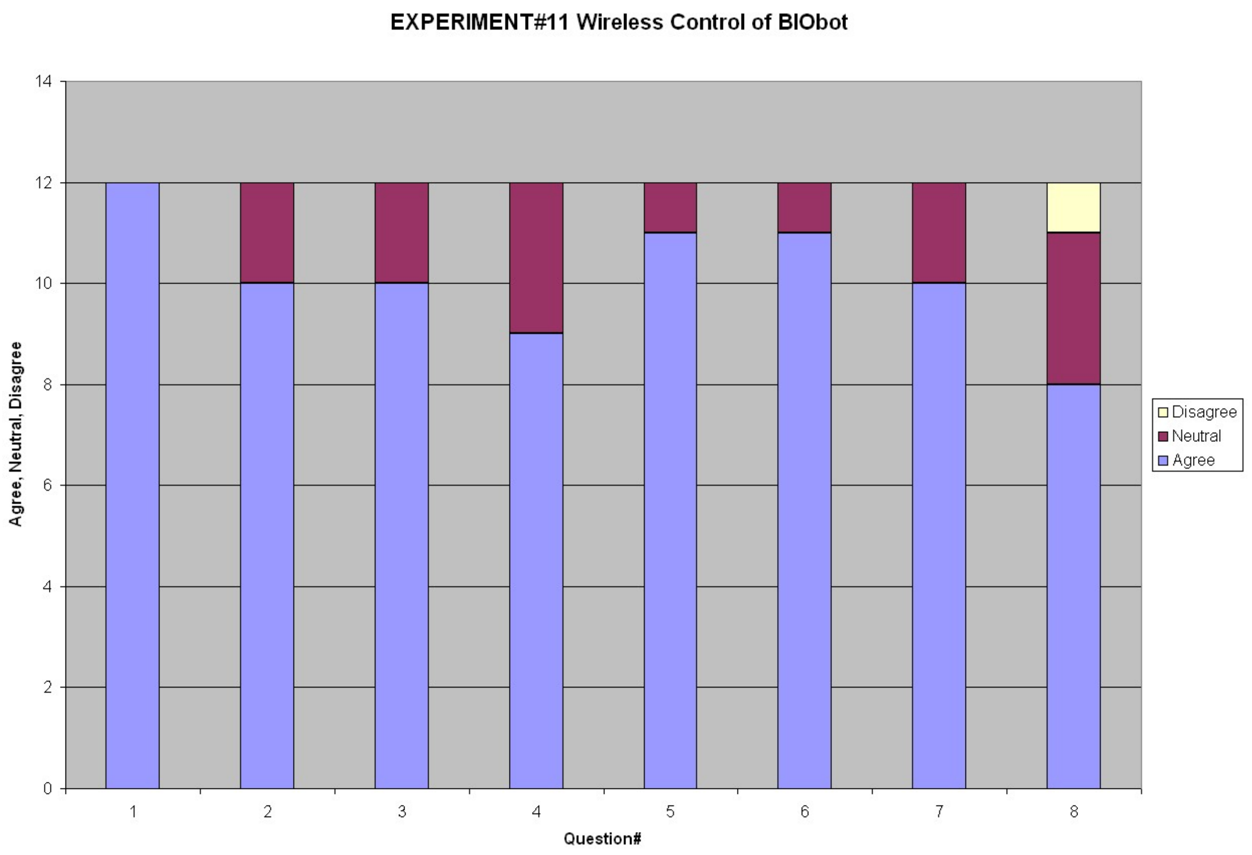

Figure 4. Results from Experiment 11 Student Evaluation.

the students left this experiment with an increased interest in learning more about wireless control.

Experiment 12 Questions

1. I enjoyed working with the BIObot robot in this lab.
2. BIObot helped to clarify the concepts associated with this lab.
3. It was easy to program BIObot for obstacle avoidance.
4. Programming BIObot for two competing behaviors (obstacle avoidance and light tracking) is difficult.
5. Working with BIObot increased my interest in this lab.
6. BIObot helped me to better understand how robots sense objects and navigate in unknown environments.
7. I recommend using BIObot in this lab for future course offerings.
8. BIObot helped me to better understand how to program robots for real-world applications.

Students again enjoyed working with BIObot in Experiment 12 and recommend its use in future course offerings. The responses to question 4 are somewhat puzzling since everyone successfully completed the experiment well within the 3hr time frame and with limited questions.

Conclusions and Future Work

In this paper we present the results of using a flexible low-cost robot in a microcontrollers and robotics course. Previous course offerings leveraged several different teaching platforms, which in turn required that students learn a new toolset each time. By using a flexible robot platform, the students are able to spend less time learning how to use a new tool and instead are

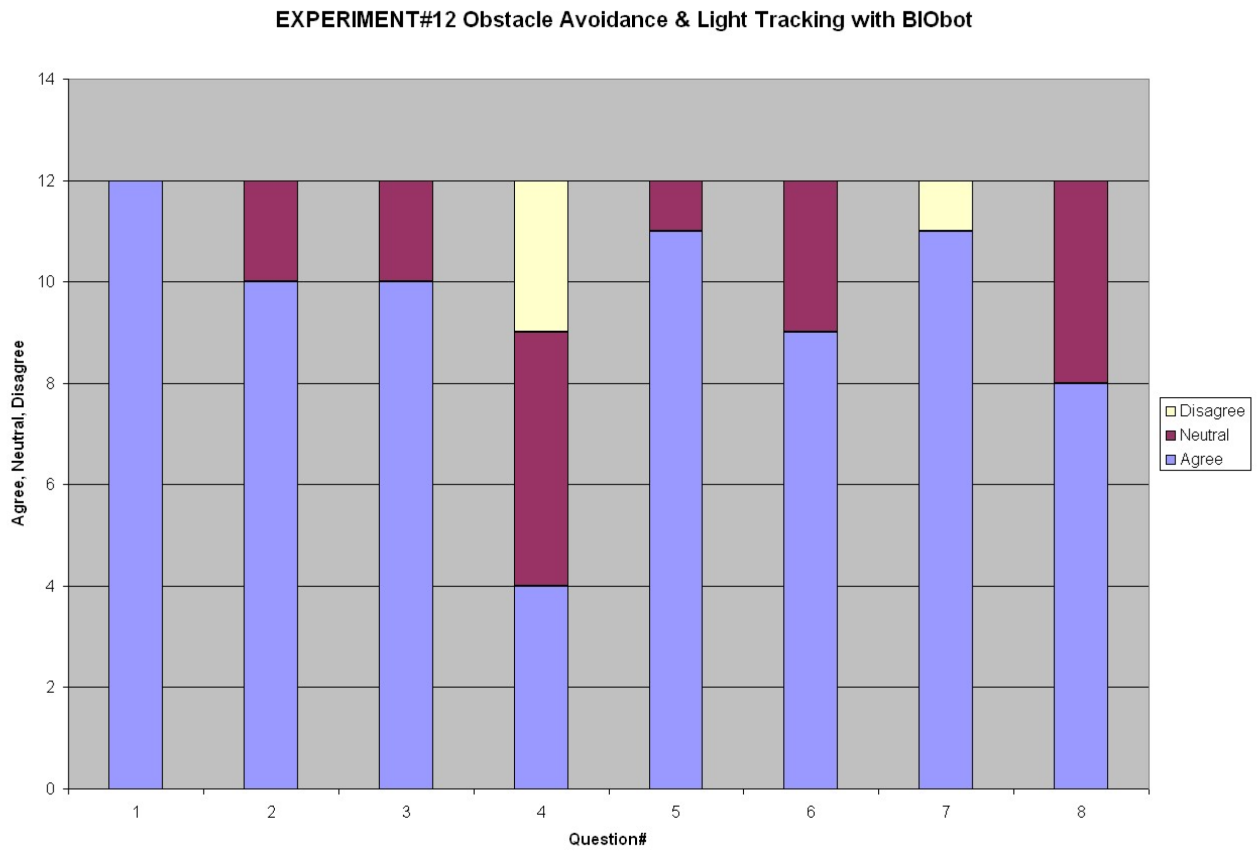

Figure 5. Results from Experiment 12 Student Evaluation.

able to focus their attention on experimenting with the concepts and theories from lecture. To achieve these goals required the use of a flexible robot that can be programmed at the microcontroller level using assembly, but also at a high-level using a desktop or laptop computer. A variety of sensors i.e. light, infrared range, RFID, and wheel encoders, help students to become familiar with different hardware to software interactions and various protocols used in signal acquisition. BIObot satisfies these requirements while helping to positively impact student interest in the lab experiments. Students had an opportunity to work with the RFID feature in Experiment 11 when they created a manual control GUI. In future course offerings the RFID feature can be leveraged in the final project. Programming BIObot to traverse an RFID tag embedded maze is one possible future final project. In this offering we were not able to utilize the CCS PIC C compiler and In-Circuit debugger/programmer. We intend to develop the necessary lecture and lab material for the next course offering. This will help to provide a smoother transition from assembly to C#. Additionally, working with a C compiler and debugger/programmer is beneficial since these are standard tools that are used extensively in industry.

Bibliography

1. CS480, Microcontrollers and Robotics. http://www.cs.binghamton.edu/~reckert/480/480topics.html

2. QwikFlash Development Kit. http://www.microdesignsinc.com/qwikflash/index.htm

3. LEGO Mindstorms Robotics Invention System. http://mindstorms.lego.com/eng/products/ris/risdetails.asp

4. Howell, A., Laramee, C., McGrann, R., and Way, E. "*Autonomous Robots as a Generic Teaching Tool,*" Proceedings of 36th ASEE/IEEE Frontiers in Education Conference (FIE), *San Diego, CA, 28-31 October 2006. Paper #1674.*

5. Howell, A., McGrann, R., and Eckert, R. *Using ZigBee to Control a Swarm of Low Cost RFID Foraging Robots*. 2007 Cybernetics and Information Technologies, Systems and Applications (CITSA) Conference, *Orlando, FL* (July 12-15, 2007).

6. A.B.E. Board. http://www.abotics.com

7. CCS C Compiler. http://www.ccsinfo.com

8. Peatman, J. B., Embedded Design with the PIC18F452 Microcontroller. Pearson Education Inc, New Jersey, 2003.

Biographical Information

Abraham L. Howell is a PhD candidate in the Department of Mechanical Engineering at Binghamton University. His research interests include swarm robotics, artificial intelligence, embedded controllers, and engineering education. The focus of his PhD research is towards the development of a low cost, flexible, open source robot that can be used as a teaching and research tool across the educational spectrum. To promote the development of educational robots, Mr. Howell owns and operates a small robotics company that designs and distributes educational robots and free robot software.

Richard R. Eckert is an Associate Professor in the Department of Computer Science at Binghamton University. His research interests include computer graphics, microcontrollers, robotics, graphical user interfaces, and computer science education.

Roy T.R. McGrann is an Associate Professor in the Department of Mechanical Engineering at Binghamton University. His research interests include design, robotics, and engineering education.