ECE 497: Introduction to Mobile Robotics
Lab # 2
(Obstacle Avoidance/Wall Following)

Purpose:
The purpose of Lab #2 is to implement an obstacle avoidance and wall following behavior for the robot. These behaviors will be created by using feedback from the infrared sensors and encoders on the Traxster robot.

Equipment:
Traxster Robot
MPLAB ICD2 package
9VDC Power Adapter
6'' jumper wires (at least 18)
various lengths of wire
Laptop Computer with MPLAB IDE
Ruler, yard stick or measuring tape
ES203 lab kit
Masking tape

Procedure:
1. *Hardware Setup*
   i. Retrieve the Traxster robot, jumper wires, MPLAB ICD2 and power adapter from the parts cabinet. (Use the **SAME** robot for all of your assignments)
   ii. Connect the battery connector, microcontroller power cable, and MPLAB ICD 2 debugger the same as in Lab #1a. Also, connect the motors to the microcontroller the same as in Lab#1a. Connect the **VCC** and **GROUND** pins on the microcontroller to create the **VCC** and **GROUND** busses. Connect the encoders to the microcontroller the same as in Lab #1b.
   iii. To use the Infrared Sensor Servo Mount, connect it to the microcontroller using the red, yellow, and black cable. Use a short wire and 6" jumper wire to connect the **BLACK** cable to the **GROUND** buss on the microcontroller. Use a short wire and jumper wire to connect the **RED** center wire to the **VCC** buss. Finally insert the connector on the **YELLOW** wire into *RD3* (see Figure 1).

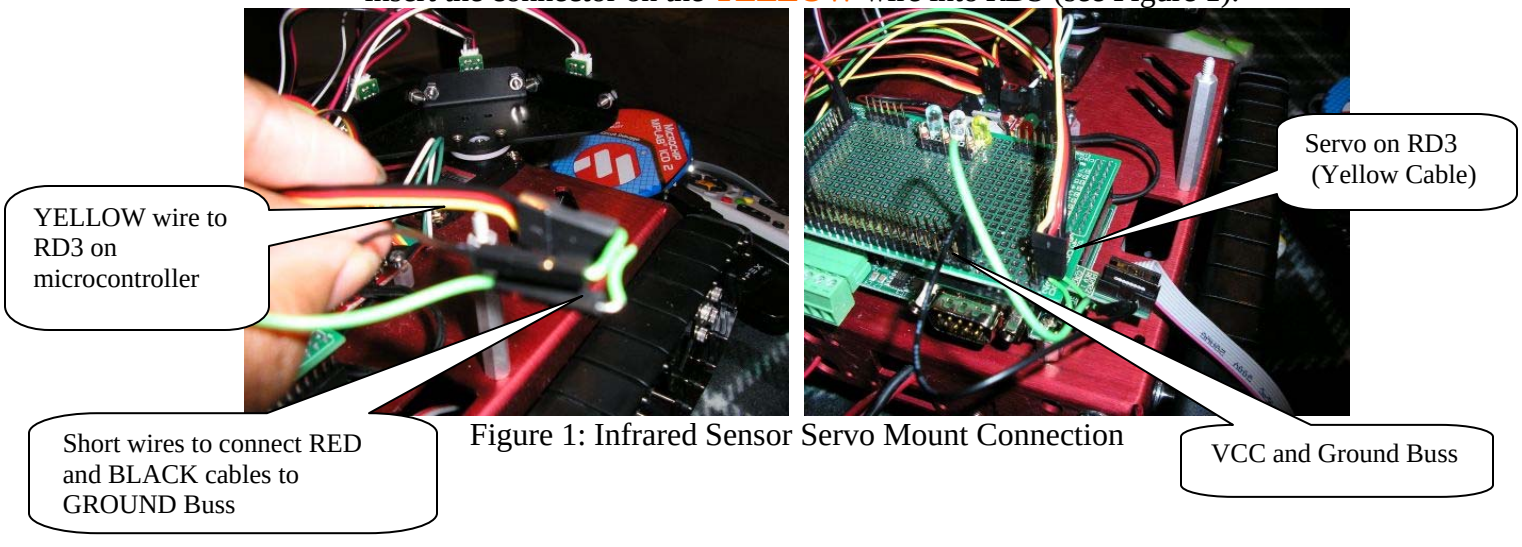

Figure 1: Infrared Sensor Servo Mount Connection

iv. To use the 3 infrared sensors on the front of the robot, connect them to the microcontroller using the 3 sets of red, white and black cables. Use a short wire and a 6" jumper wire to connect the black wire on each cable to the **Ground** buss. Use a short wire and 6" jumper wire to connect the center **RED** wire on all 3 cables to the **VCC** buss (see Figure 1b). Finally, insert the WHITE cable on the robot's left IR sensor into *RA0*, the WHITE cable on the center IR sensor into *RA1*, and the WHITE cable on the robot's right IR sensor into *RA2* (see Figure 2).

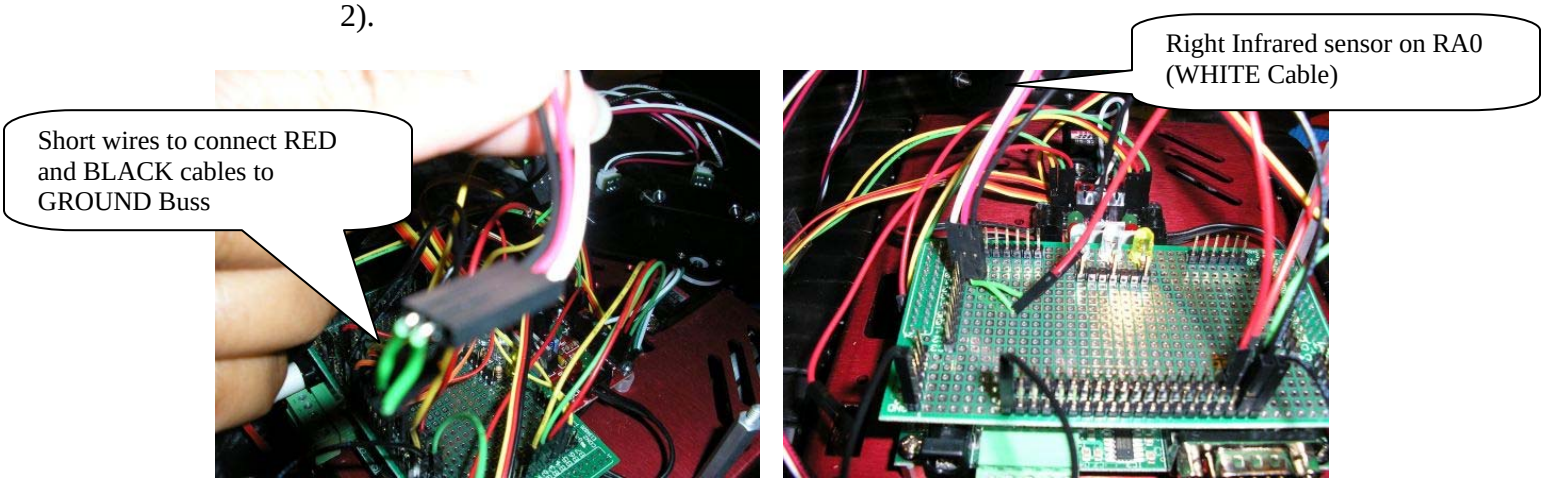

Figure 2: Infrared Sensors Connections

2. *Infrared Sensor Calibration (Linearization)*

a. A/D Conversion

i. Read the documentation on linearizing the Sharp GPD120 Infrared Sensor Data (Linearizing Sharp Range Data). In this section, you will determine a method to linearize the relationship between sensor detection distance and output voltage (see Figure 3).

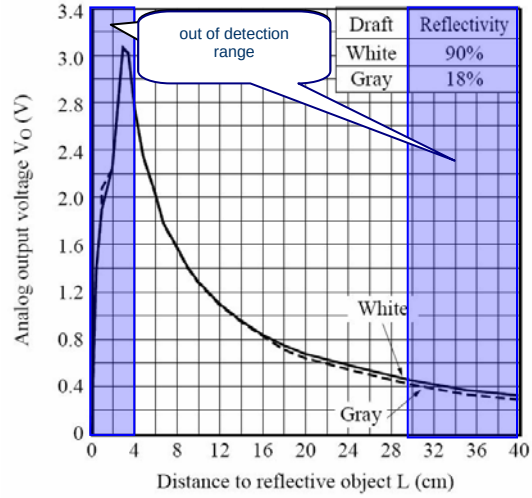

Figure 3: GPD120 Analog Voltage versus detection distance

ii. Download the ***"Infrared(Analog).c"*** and ***"Infrared(Servo).c"*** programs from the Angel Labs folder

iii. Save them to a folder on your laptop for all ECE497 laboratory files

iv. Read the programs and make sure you understand what they do

v. Use the project wizard to create a project (name it ***"Infrared.mcw"***) and include the ***"Infrared(Analog).c"*** file

vi. Insert a Debug point at the *"i = 0"* line at the end of the for loop

vii. Open the Watch window and add the symbols, *Sens1val*, *Sen2val*, and *Sen3val*

viii. In Debug mode, Connect, Build, Program and Run the ***"Infrared(Analog).c"*** program. The program will break at the *"i=0"* line and the digital representations of the sensor voltages should be evident in the Watch window

ix. Select Sensor 3 (left sensor) and place a vertical sheet of paper or other reflective object (preferably white) in front of the sensor at the distances specified on Table 1. Record the sensor value and complete this column of the table. *(hint: DO NOT obsess with getting **perfectly logarithmic** data, just get an acceptable approximation)*

x. Next, leave the power to the microcontroller ON but remove Sensor 3 from RA2 and use the Digital Multimeter to measure the voltage on the white wire with respect to ground for all of the distances in Table 1. *(hint: DO NOT obsess with getting **perfectly logarithmic** data, just get an acceptable approximation)*

Table 1: GPD120 Linearization Data

Range, R Distance (cm)	**Sensor 3 Value (A/D 10-bit conversion)**	**Digital Multimeter Sensor 3 voltage, V**
4		
5		
6		
7		
8		
9		
10		
12		
14		
16		
18		
20		
25		
30		

x. Create **2** x-y scatter plots of the data in Excel
   A. Analog voltage, V (y-axis) versus distance, cm (x-axis)
   B. Range, cm (y - axis) versus A2D conversion (x - axis)

xi. Determine the value of k that will most closely linearize both of these plots and create an x-y scatter plot of 1/(R + k) versus A2D conversion

(see Linearizing Sharp Range Data). *(hint: DO NOT obsess with getting **perfectly linear** data, just get an acceptable approximation)*

xii. Add a trend line with an equation in Excel to get an equation for the line in step xi. $(1 / (R + k) = m * V_{A2D} + b)$

xii. Rearranging the equation, $R = (1 / (m * V_{A2D} + b)) - k$ or $R = (m' / (V_{A2D} + b')) - k$, where m' = 1/m and b' = b/m. Use the 2nd equation in order to eliminate the floating point math.

viii. Provide the data table, Excel scatter plots and the linearizing function with the value of k in the final laboratory report

b. Servo Integration

i.. The ***"Infrared(Servo).c"*** program also has a main function and should be run separately from the ***"Infrared(Analog).c"*** program (perhaps in a dummy project) so that you can see what it does

ii. Eventually, you will have to modify both files and include them in the same project. The final project (***"Infrared.mcw"***) will have one main function to control the Servo position and acquisition of the Analog Infrared Sensor data at some predefined intervals.

3. *Obstacle Avoidance*

i. Using the concepts presented in Pre-Lab 2, write a program ***("AvoidObstacle.c")*** so that the robot will move in a given trajectory until it encounters an obstacle

ii. The robot should then navigate around the obstacle and continue along the provided path. Make use of the 3 LEDS on the microcontroller to indicate robot state or the detection of an obstacle.

iii. Your program should include a method to keep track of the robot's 'state' similar to the pre-lab

iii. Your program should provide a method to get the robot 'unstuck' if it approaches any local minima points

iv. Your program should be as modular as possible with multiple subroutines and behaviors that will be integrated in subsequent programs

v. Devise a method to test and confirm that your program works correctly and present the results in the laboratory report

4. *Wall Following*

i. Using the concepts presented in Pre-Lab 2, write a program ***("WallFollow.c")*** so that the robot detects a wall and follows it at a 5 inch distance using feedback control from infrared sensors and possibly the wheel encoders

ii. Your program should have a technique to get the robot "unstuck" from corners

iii. Your program should also have a method to turn corners to maintain contact with the wall

iv. Your program should have a method to keep track of the robot's current state similar to the pre-lab

iv. Your program should be as modular as possible with multiple subroutines and behaviors that will be integrated in subsequent programs

v. Devise a method to test and confirm that your program works correctly and present the results in the laboratory report.

5. *Integration of Behaviors*
   i. Now that you have wall following and obstacle avoidance behaviors, integrate those two programs to test the following scenario
   ii. If the robot is following a wall and encounters and obstacle, write a program so that the robot navigates around the obstacle and continues to follow the wall (see Figure 4).
   iii. Devise a method to test and confirm that your program works correctly and present the results in the laboratory report.

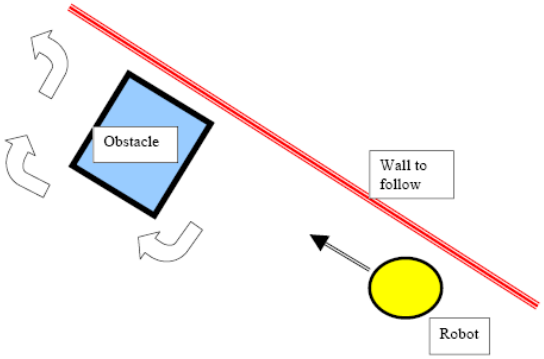

Figure 4: Wall Following and Obstacle Avoidance Integration

A sampling of questions and observations to discuss in your laboratory report:

1. What formula did you use to linearize the robot's infrared sensors?
2. How did you use the servo mount to get a full range of sensory coverage?
3. Were there any obstacles that the robot could not detect??
4. Were there any situations when the infrared sensors did not give you reliable data?
5. What was the general plan you used to implement the Obstacle Avoidance and Wall Follow behaviors?
6. How did you keep track of the robot's states in the program?
7. What strategy did you use to get the robot 'unstuck' from corners?
8. What strategy did you use to program the robot to turn corners?
9. How did you confirm that the programs worked correctly and what type of errors did you get?
10. How would you modify the code you wrote to detect objects with a giving IR signal and navigate to the object in the robot's environment (Lab 3)?
11. Were there any problems that you encountered in the implementation of this lab? Discuss how you addressed or resolved them.

The typewritten team laboratory report is due two weeks from today at the beginning of lab. It **MUST** follow the format provided by the instructor. *(See sample lab report on Angel course website)*