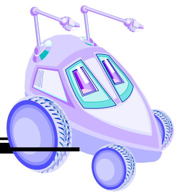

# ECE 497: Introduction to Mobile Robotics
# Lab # 1b
# (Feedback Control)

Purpose:

The purpose of Lab #1b is to continue the motor control experiment on the Traxster robot. In order to correct the error accumulation found in Lab#1a, a position feedback position controller will be implemented. The encoder attached to the input shaft of the wheel motors will be used to provide the feedback position to the controller for the *Circle* and *Square* programs. The results of this controller will be compared to the results of Lab #1a.

Equipment:

Traxster Robot
MPLAB ICD2 package
9VDC Power Adapter
4 - 6'' jumper wires (3 red, 2 yellow, 2 green, 1 black)
Laptop Computer with MPLAB IDE
Watch with second hand
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 (Table 1).

Table 1: H-Bridge and Microcontroller Wiring

| Jumper Wire | PIC16F877A Microcontroller Bit | H-Bridge terminals | Description |
|---|---|---|---|
| Yellow Wire | RB0 | E1-2 | Right motor enable |
| Green Wire | RB1 | I2 | Right motor forward |
| Red Wire | RB2 | I1 | Right motor reverse |
| Yellow Wire | RD0 | E3-4 | Left motor enable |
| Green Wire | RD1 | I3 | Left motor forward |
| Red Wire | RD2 | I4 | Left motor reverse |

   iii. To provide power to the encoders from the microcontroller, use the 2 rows of pins in back of the microcontroller as a +5V and ground buss. To make the busses, connect the black wire from the "GND" pin on the microcontroller to the farthest buss on the back on the microcontroller. Connect the "VCC" pin from the microcontroller to the buss in front of the ground buss (see Figure 1).

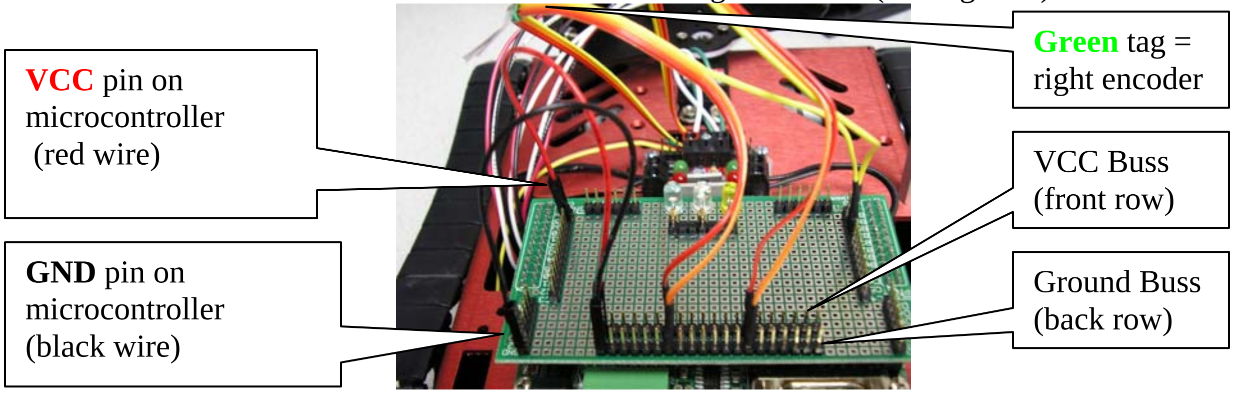
**VCC** pin on microcontroller (red wire)
**GND** pin on microcontroller (black wire)
**Green** tag = right encoder
VCC Buss (front row)
Ground Buss (back row)

Figure 1: Buss Setup

iv. Please note that the right encoder has the green tag and tape on it. Connect the '**red**' wires from both encoders to the "VCC" (front buss) on the microcontroller. Connect the '**orange**' wires from both encoders to the "GND" (back buss) on the microcontroller (see Figure 2).

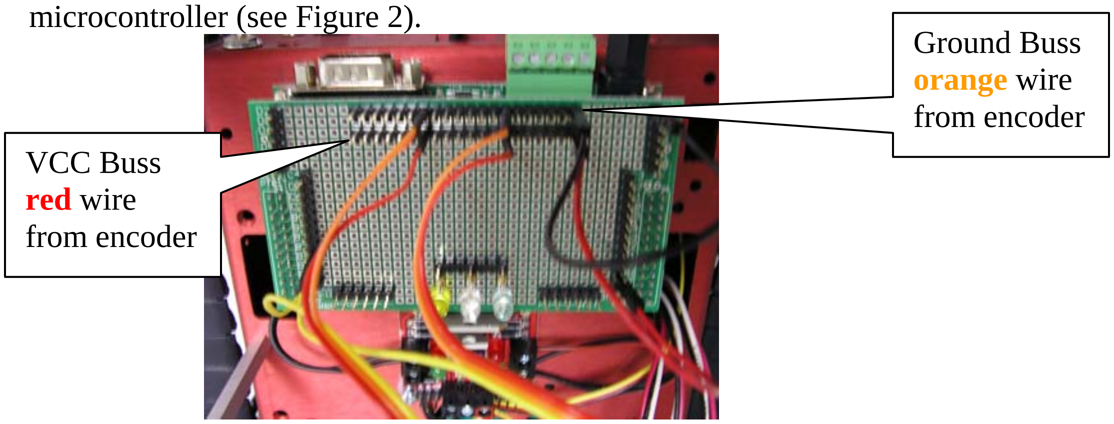
Ground Buss **orange** wire from encoder
VCC Buss **red** wire from encoder

Figure 2: Encoder VCC and Ground

v. Connect the 'yellow' wire from the right encoder to RB4 on the microcontroller. Connect the 'yellow' wire from the left encoder to RB5 on the microcontroller (see Figure 3).

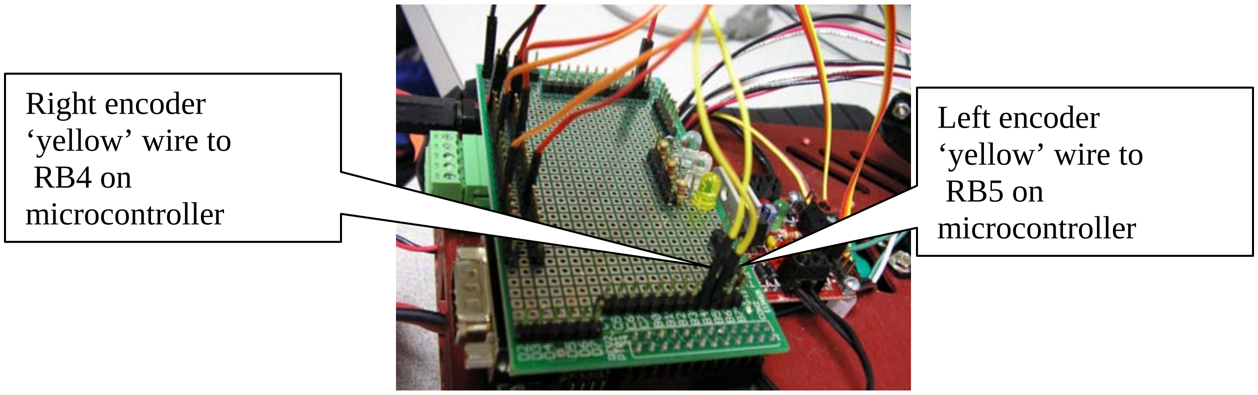
Right encoder 'yellow' wire to RB4 on microcontroller
Left encoder 'yellow' wire to RB5 on microcontroller

Figure 3: Encoder to Microcontroller Connection

2. *Encoder Calibration*
   - i. Track distance
     - a. Download the code for ***"Encoder(2).c"*** from the Angel course website Lab folder
     - b. Save it into a folder on your laptop for all ECE497 laboratory files
     - c. Read the program and make sure you understand what it does
     - d. Use the project wizard to create a project (name it ***"Encoder"***) and include the ***"Encoder(2).c"*** file
     - e. Modify the code to drive the robot forward for a certain number of encoder counts
     - f. Measure the distance that the robot travels in this number of encoder counts.
     - g. Find a relationship between the number of inches and encoder pulses. Call this variable $K_{de}$ (in/pulse)
   - ii. Spin Angle ($v_1 = -v_2$)
     - a. Modify the code in ***"Encoder(2).c"*** to rotate the robot about its center
     - b. Determine the number of encoder pulses it takes for the robot to make one full revolution
     - c. Use this value to find a relationship between the spin angle and the number of encoder pulses. Call this variable $K_{se}$ (°/pulse)
   - iii. Turn Angle ($|v_1| \neq |v_2|$)
     - a. Modify the code in ***"Encoder(2).c"*** to turn the robot by running the tracks at different velocities (similar to ***"Circle.c"*** )
     - b. Determine the number of encoder pulses for each track that allows the robot to make one complete revolution
     - b. Use the values $p\Delta = p_1 - p_2$, $K_{de}$, and the circumference of the circle transcribed by the robot's spin angle to determine a relationship between the turn angle and wheel encoder pulse difference (see Lecture 2). Call this variable $K_{te}$ (°/pulse)

3. *Circle and Square Tests*
   - i. Now that you have relationships between robot distance, spin and turn angles and the encoder pulse counts use this information to modify your "***Circle.c"*** and ***"Square.c"*** code. Modify the code to implement a proportional feedback position controller. Save these files as ***"Circle2.c"*** and ***"Square2.c".*** *Make sure your code is properly commented and all behaviors are written as subroutines (i.e. MoveCircle(int radius), MakeSquare(int sides)). See "Encoder(2).c" for an example of properly commented code with subroutines.*
   - ii. Mark the starting point for the robot with masking tape. Run the program one time and measure the radius of the circle and the sides of the square. Calculate the % error for both with respect to a 2 foot radius and 2 foot side.

iii. Finally, run each program 5 times and create an x-y scatter plot of the end positions compared to the starting point for inclusion in your lab report. Additionally, include the mean and standard deviation of these values.

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

1. What is $K_{de}$ (in/pulse)?
2. What is $K_{se}$ (°/pulse)?
3. What is $K_{te}$ (°/pulse)? What formula did you use to calculate this value?
4. What gain did you use for the proportional controller and what was the result of the various gains tried? How did you arrive at the final gain to use?
5. Compare and contrast the Circle and Square results for the proportional feedback control versus the open loop control. What was the difference in % error? What was the difference in the spread of the data?
6. How could you use PD or PID control to correct for any errors still evident in the robot position?

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)*