Worksheet 1
Parallax Robot Odometry

Learning Outcome 2

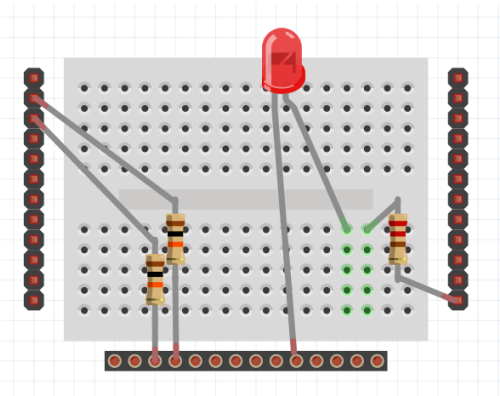
Book Chapter 1

Purpose.

To investigate how wheel encoders can help locate the position of a Parallax robot in the plane.

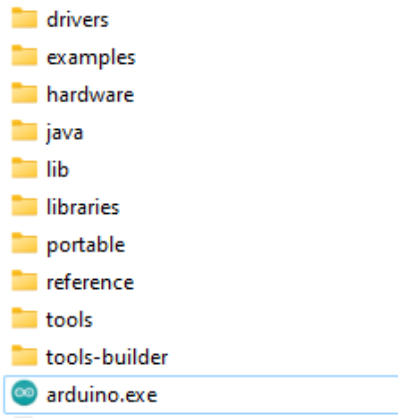
1. Preparation.

(a) The wheel encoders will be connected for you. But you must add two 'pull-up' resistors of value 10k between pin 10 and +5V and pin 11 and +5V. The diagram on the right will help.



(b) Now connect digital pin 10 to pin 2, and pin 11 to pin 3. This is because the encoder connexions on the Parallax pcb come in on pins 10 and 11, but the Arduino Mega has interrupts on pins 2 and 3.

(c) Download and unzip the folder **portable** and paste into the Arduino folder on your machine, so that the folder structure appears as shown on the right.



2. Manual Check of Encoder Function

Here we shall rotate the wheels by hand and observe the encoder pulses using the Serial Monitor.

(a) Load the sketch **CBP_2403_R_Encoder_Test_1** and have a glance over the code. Note that there are two ISRs (Interrupt Service Routines) on lines 52 – 54 and lines 56-58. These are attached to the left encoder pin (line 42) and the right encoder pin (line 43), so they receive the respective wheel pulses.

(b) Fire up the Serial Monitor and turn each wheel by hand. You should see the values of **countL** and **countR** racking up.

(c) Estimate the number of pulses for one complete revolution of a wheel.

(d) Is there any difference between rotating the wheel clockwise or anticlockwise? What could this mean for any application using encoders?

3. Driving Straight for a given distance

(a) Load up the sketch **CBP_2403_R_Encoder_Test_2**. This will make both wheels rotate one revolution. Check that this happens.

(b) Modify the code so that the wheels rotate **N** revolutions. Place the robot in the arena and note its trajectory which should be a straight line. Mark the start and end positions of both wheels. What do you find?

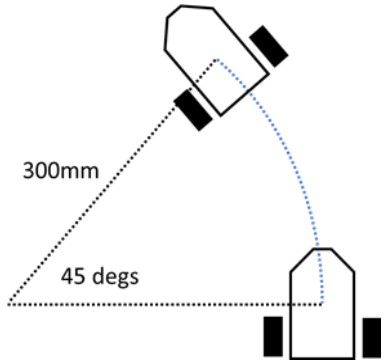
(c) Now let's get the robot to move a **desiredDist** forwards (in mm). Open the sketch **CBP_2403_R_Encoder_Test_2a** and look for the line where you can specify your desiredDist. Now add a line of code to calculate **nL**, the number of left motor steps. Hint: You will need to use the constant **dx** which is the distance moved in one encoder step. Upload and run the robot and measure the actual distance covered.

(d) You will probably find the robot does not go straight, so let's attempt to correct it. If the robot veers to the right, then we need to increase **driveR** and *vice versa*. Measure the distance each wheel moves, **measuredL** and **measuredR**. The correction can be made by the following,

```
driveR = driveL * measuredL / measuredR;
```

Open up **CBP_2403_R_Encoder_Test_2b** and complete the code and test.

4. Driving along an Arc



An arc is specified by its radius and the angle through which the robot body rotates, an example is shown in the diagram on the right.

(a) The sketch **CBP_2403_R_ArcTest_0** will make the robot move along an arc. All you need to do is to specify the **desRadius** (in mm) and **desDegrees** (angle in degrees). Have a look at the code and try to relate this to the theory.

(b) Run the robot, mark the start and end locations of both wheels and measure the radius and angle of the robot's trajectory. You will likely find some errors which we must now correct.

(c) The correction is calculated using the Octave script **Calc_Arc_Correction_Parallax.m** , download this. It will ask you to enter your desired radius and angle, and also your measured radius and angle. It will then calculate the corrected numbers of encoder steps as floats, **nLf, nRf**. Note these down.

(d) Now open the Arduino sketch **CBP_2403_R_ArcTest_1** and insert your float values, over-writing the ones you will find there. Now run your robot again and all should be well, maybe.

Takeaway. You should have found that odometry is not trivial and can be very frustrating. As Fred Martin said, "Real Robots Don't Drive Straight" and he argues, as we do, for feedback to be included in robot drive situations.