Comp2403 Line Following

C.B.Price September 2021

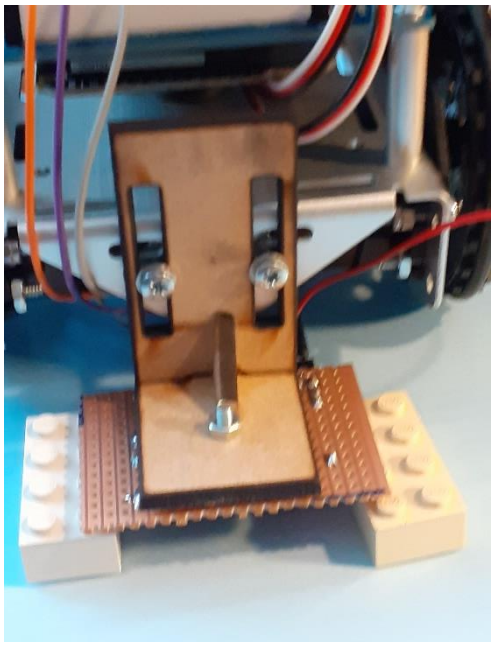
Purpose	(i) To mount and investigate the line sensor (ii) To use the line sensor to follow a line (iii) To investigate the 'PID' Control algorithm
Files Required	Arduino Sketchbook and Octave scripts on the web-pages.
ILO Contribution	LOs 1 & 3
Send to Me	nix
Homework	Read chapters 2 & 3

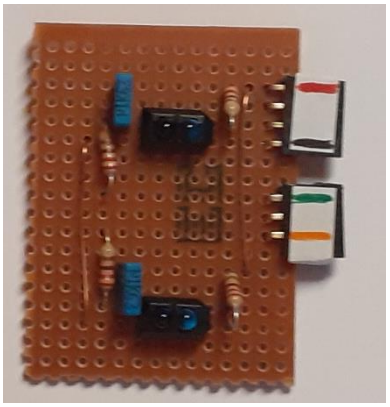
Activities

---

1 Mounting the line detector

(a) Fix the detector to the front of the robot using a couple of screws as shown below left. Make sure the detector components are just above the surface; I used Lego bricks.





(b) Now hook up the detector following the labels on the right. Red to 5V, black to Gnd, the green to pin 5 and orange to pin 4.

(c) Open up the sketch **CBP_R_2403_LineSensor_Test** and add the following lines to get the left and right sensor values

```
senseL = RCTime(5);
senseR = RCTime(4);
```

then calculate the error as explained in class

```
error = ((float)senseL - (float)senseR)/((float)senseL +(float)senseR);
```

---

and finally add code to print everything to the serial monitor thus,

```
cprintf("senseL = %l, senseR = %l, error = %f\n",senseL,senseR,error);
```

(d) Now take the testcard and place the white part under both sensors and note down the readings, they should be similar. Repeat for the black part, again the readings should be similar, but different from the white readings.

Can you explain why one pair of readings is larger? Remember the readings are the times in microseconds needed to discharge the sensor capacitor, and the phototransistor acts as a resistor whose resistance gets smaller the more light enters it.

(e) Now place the testcard as though the robot were following it; place it at the centre of the sensors. Move it to the left and see how the readings change. Now return it to the centre, and move it to the right and see how the readings change.

(f) Now if you are to use the value of **error** to steer the servos, which servo should you add the error to and which should you subtract it from?

---

2 Simple Line Follower

(a) Open up the sketch **CBP_2403_R_Line_Follower** and look where the error is calculated. Now complete these lines of code using `Kp*error` in order to get the robot to turn towards the line when it loses it.

```
driveL = fwd …
driveR = fwd …
```

Remember that **fwd** is the constant drive sent to both servos to get the robot starting off moving forward.

(b) In **setup** set the value of **Kp** to 10.0, upload and see how the robot fairs on the track.

(c) Try increasing **Kp** several times and see what happens. You should find that if this is too large, then the robot will start oscillating, or 'hunting' for the line.

---

3 PID Line Follower

(a) Open up the sketch **CBP_2403_R_Line_Follower_PID**. The code to calculate the error is given. Now add these lines to provide the derivative (the variable **deriv** is already declared)

```
deriv = error - lastError;
lastError = error;
```

(b) Now complete the code for the motor drives using **error, deriv** and also using **Kp** and **Kd**

```
driveL = fwd …
driveR = fwd …
```

(c) The following parameters have been set; **fwd = 30; Kp = 50; Kd = 0;** Run the sketch with these values and observe the robot behaviour.

(d) Now increase **Kp** in stages until you find the robot starts to oscillate on the track. This will probably occur on the curves.

(e) Now increase **Kd** and for each value observe its effect. At some point the oscillations should be almost completely damped out. Typically the value of Kd is less than Kp.

4 Investigation Ideas

(1) There are other variables such as **fwd** which we have not changed. Do the results you found in (3) change if you increase the value of fwd?

(2) How fast can you get the robot to follow a track?

(3) Can the robot traverse gaps in the track?