Comp2403 Hello Parallax Bot

C.B.Price September 2021

Purpose	To learn (i) how to drive the Parallax Bot, (ii) how to use the Ultrasonic distance measurement.
Files Required	Arduino Sketchbook and Octave scripts on the web-pages
ILO Contribution	LOs 1 & 3 
Send to Me	nix
Homework	Read chapters 1 & 2

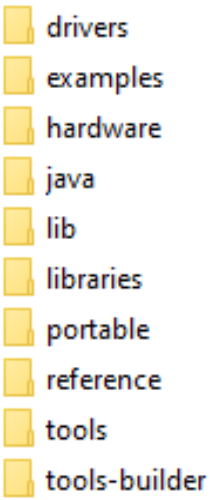
Activities

---

1 Arduino software Initialisation

(a) Download and unzip the Arduino Assets, I suggest you place them on the desktop.

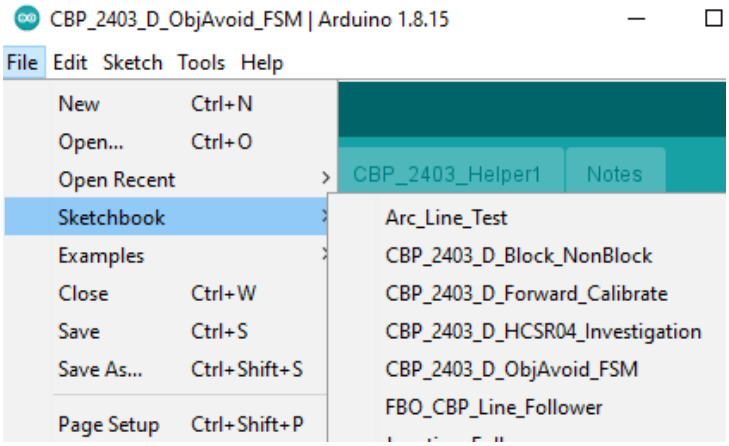
(b) Find where the Arduino application is installed on the C: drive. Open up the folder and any subfolders until you see something like this



If this is your first time, then you will not have the **portable** folder.

(c) Copy the **portable** folder from the unzipped assets into this folder.

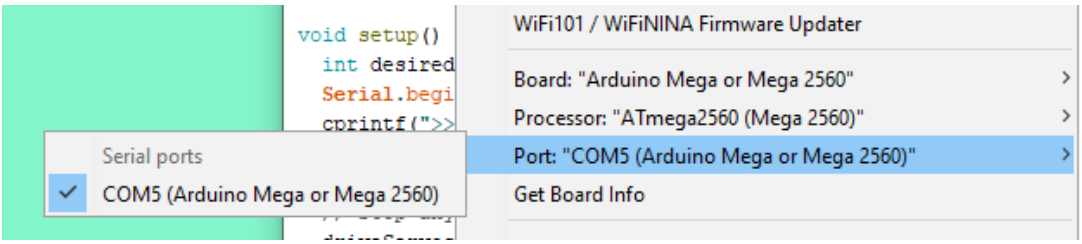
(d) Fire up the Arduino application, and in the menu-bar select File > Sketchbook you should see something like this (it will be different, but there will be some **CBP_2403_** sketches there).



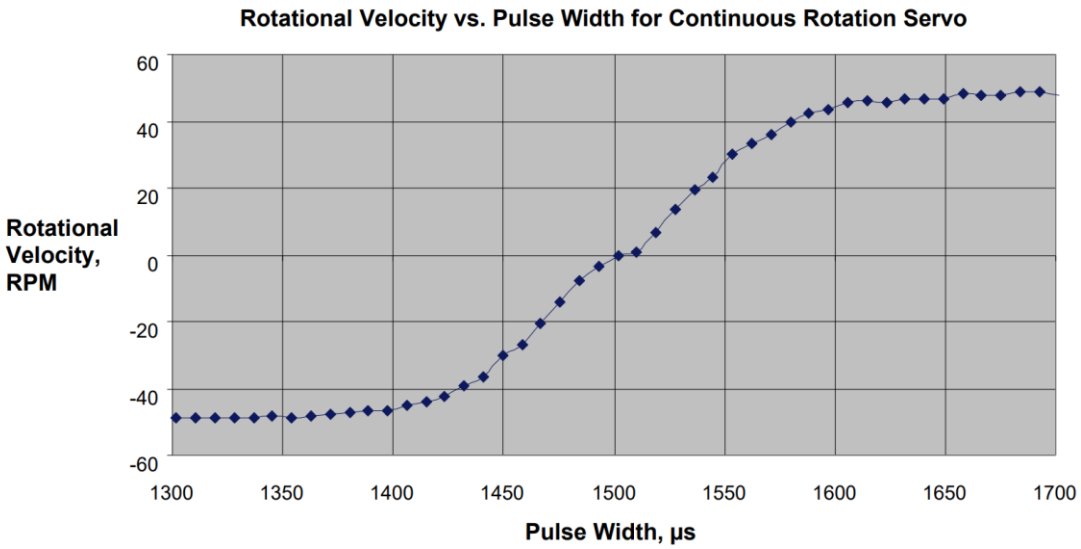
---

2 Driving the Servo Motors

(a) Open the sketch **CBP_2403_R_Simple_Motor_Drive.ino**. Make sure the Arduino is connected to the PC USB port. Check that you have the correct Arduino selected, and the port is hooked up. From the Tools item in the menu-bar you need to see this (though the port number may be different



(b) Now add a couple of lines of code to set **driveL** and **driveR**. Look at the graph below, where the values of drive are along the bottom, and the motor revolutions per minute are up the side. Remember 60 rpm means one revolution of the wheels each second, that's fast. **NOTE:** If you want to get 30 rpm then the drive is 50, **not** 1550, so everything is relative to 1500 (at rest).



(c) Now add the line of code **driveServos(driveL,driveR);** and follow this by a delay of 2 seconds (2000 milliseconds)

delay(2000);

(d) Compile your code  and debug if necessary, then upload to the Arduino  , remove the USB connexion and see what happens.

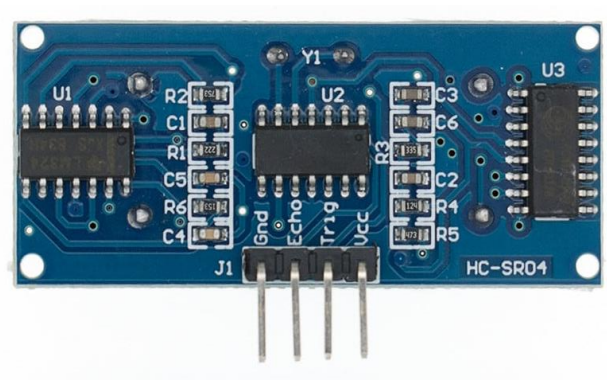
(e) You probably found that the wheels did not stop after 2 seconds. Why not? Well the servos have memory, they store the last driveL and driveR values. So after your 2 sec delay you have to stop the motors explicitly. Add some code to do this.

(f) Investigate the effects of changing **driveL** and **driveR.** Can you get the robot to move backwards? How about in a circular arc? Or spinning about its centre.

(g) Briefly investigate the effects of changing the delay value.

3 Testing the HC-SR04 Ultrasonic distance measurement sensor

Here's the details of the HC-SRO4 sensor. You'll need to use the info on the back (left image).



(a) Insert the sensor into the Parallax Bot protoboard so it is facing forward. Then ***looking at the back labels*** wire-up the following connexions, using the strips near the protoboard.

- Gnd to Arduino GND
- Vcc to Arduino 5V
- Trig to Arduino pin2
- Echo to Arduino pin3

(b) Load up the sketch **CBP_2403_R_HCSR04_Investigation.ino** and find the function **getDistance()**. Now add the following lines of code

digitalWrite(HC_SR04_Tx,LOW);	This will set the Trig low
delayMicroseconds(2);	to hold Trig low for 2 mu-seconds
digitalWrite(HC_SR04_Tx,HIGH);	Now set Trig high
delayMicroseconds(10);	and hold it there for 10 mu-seconds
digitalWrite(HC_SR04_Tx,LOW);	Finally turn it low again

At this point we have sent a 10 microsecond pulse to the HC-SRO4 to make it emit a pulse.

(c) Now add the following lines

duration = pulseIn(HC_SR04_Tx,HIGH);	This gets the time in mu-seconds between the pulse emission from the device transmitter, and the time the reflected pulse is received.
mm = 10*duration / 29 / 2;	This calculates the distance to the object in mm.

Now add this line to send the duration and distance in mm to the Serial monitor.
cprintf("Duration (muSec) = %l, dist (mm) = %f\n",duration,mm);

(d) Compile and upload the sketch and test the sensor out. Point it at your computer screen and check that you get reasonable distance values. Point it at the walls, and at the ceiling

The datasheet suggests a min range of 20mm (2cm) and a max range of 4000mm (4m) and a measuring angle of 15 degrees. But we are using cheapo devices which may have an accuracy of 10%.

You should find you get some 'spurious' readings, especially for large distances, which are often reported as being small.

(e) Add the following code to weed out these pathological cases, and return -1 to the calling code which indicates sensor error

```
if(duration < 600) mm = -1;
if(mm > 4000) mm = -1;
```

Now check that the sensor gives correct values, or -1 on error.