

COSC 4P78 – Lab 4

The goal for this lab is to make a *line-following* robot.

Preliminary Task:

Before doing this week's lab, you'll first need to make sure you're caught up on all of the past weeks' labs. This lab assumes you have a robot that you can make move, as well as turn.

Line-Following Sensor and Testing:

The sensor you've been given is this one:

<http://www.robotshop.com/ca/en/dfrobotshop-rover-line-follower-sensor.html>

That page has links to both the datasheet and (in theory) sample code. However, you won't need sample code. Instead, you're probably better-off just tinkering with it.

The line-follower is an *analog* sensor. This means that, when provided with a standard voltage (in this case, +5V), each of the three IR receivers on it will return a different voltage, depending on the intensity of the infrared light from the emitter reflected off nearby surfaces.

White paper will reflect far more than black paper.

So, what sorts of ranges can you expect? That'll depend on several factors, including precisely how you mount the sensor, and what surface you test it on.

To start, take a quick look at your sensor, as well as the various doodads included with your sensor bracket. For this task, you'll probably just want the spacers (and the screws that can mount them to the robot's underside).

How you choose to actually attach the sensor is up to you. Just be careful, as there are no spares.

Refer to the documentation for specifics, but there are three analog outputs from the sensor; each of these will need to connect to one of the analog/ADC lines of your hackduino.

Then, you'll be using the `analogRead` function at regular intervals to determine what each receiver 'sees'. The returned value will be an `int` from 0 to 1023 (because it uses a 10-bit ADC).

But, if the actual ranges will vary from robot to robot, then how will you know what values to use for your logic?

Serial Communication;

You aren't limited to using your Serial cable *solely* for programming the controller. It can also be used for serial communication with the device when it's running.

This will probably be of use to you:

<https://www.arduino.cc/en/Reference/Serial>

And you might find this of some minimal interest:

<https://www.arduino.cc/en/Reference/Stream>

Some quick tips:

- Put `Serial.begin(9600);` in your `setup()` procedure
- You can use `Serial.print` or `Serial.println` to print values (including automatically rendering an `int` as a character sequence)
- The Arduino IDE has a *Serial Monitor* window for serial communication with the device
 - You *could* also use PuTTY for this, but that's definitely overkill
 - If the byte stream starts out of sync, just reset the controller while the Serial Monitor's open

So, test it out near various patterns, to see what sorts of values each receiver returns for white/black.

Line-Following:

You're not trying to do anything terribly complicated yet, so this should be pretty easy to write.

- Mathematically, there are only 8 possible combinations for the three receivers
 - In practice, there are even fewer

All you need to do is get it working on one of the Mindstorm sheets, but feel free to draw out a fancier course, if you like.