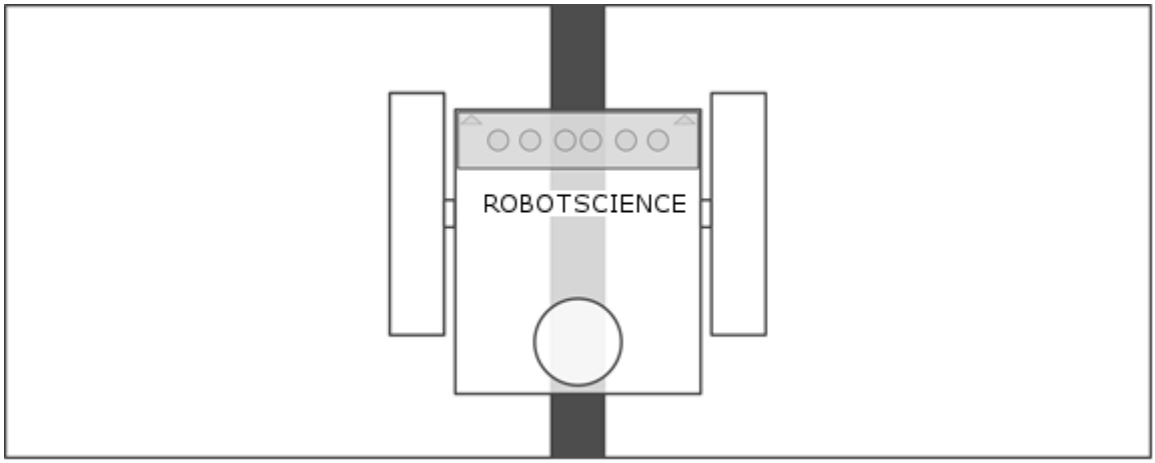


WORKSHOP 11.1: BASICS OF GETTING YOUR ROBOT TO FOLLOW A LINE

It is extremely useful to experiment with practical examples of computer code that simulates what is happening in factories around the world, where automation is important not only to achieve profitability and competitive product pricing, but also plays a major role in delivering consistent quality day in an day out. Where a machine in a factory is able to operate largely without human intervention, and can even make some adjustments and corrections in terms of its own operation, that characteristic is referred to as "autonomy". To create an autonomous system of your own, you can program your robot to follow a line at a much lower cost than the multi-million dollar robotics installations of modern factories. All you need to build your own autonomous system is a flat white surface, and a cheap roll of electrical insulation tape from a hardware store. The basic line following system we teach you in these workshops works well indoors, but won't work reliably outdoors where bright sunlight will overwhelm the sensors.

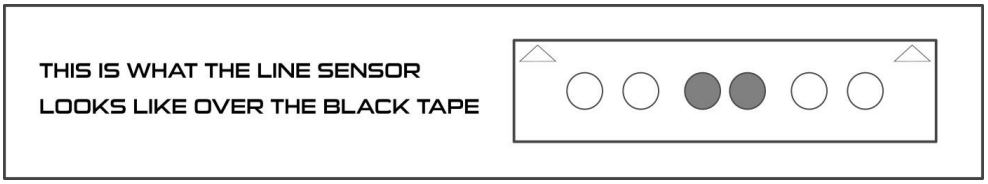
VIEW FROM OVERHEAD THE ROBOT SHOWING THE POSITION OF THE LINE SENSOR



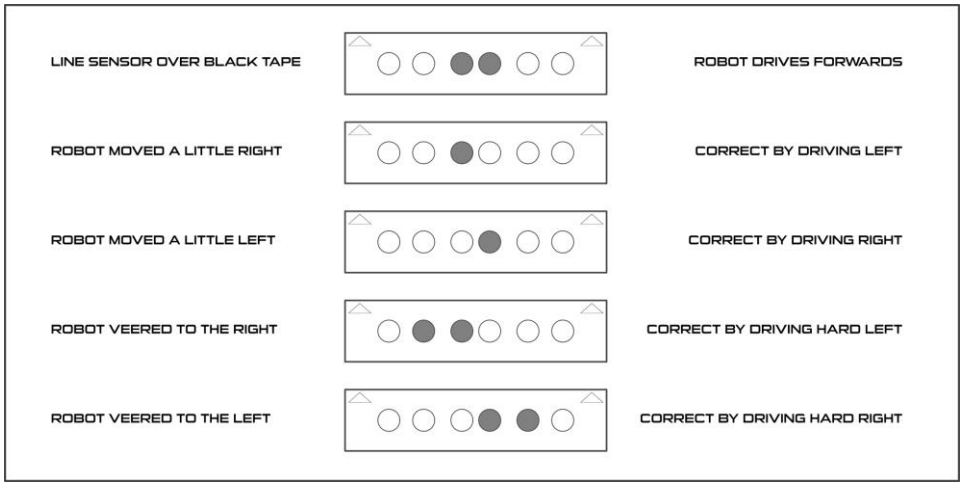
HOW DOES LINE FOLLOWING WORK?

The robot drives forwards along a line until one of the sensors "sees" that the robot has moved to one side, then the robot turns slightly and drives the other way until the other sensor ends up over the black line. In the above drawing the line sensor is over the black line, which means the two middle sensors return a low reading, the other sensors are over white so they return high readings.

In the drawing below we can see the values the robot sensors return when the robot is exactly overhead the line. The white dots represent a photo-transistor that is returning a value that is above the threshold [in other words HIGH] and the grey dots represent a photo-transistor that is returning a value that is below the threshold [in other words LOW].



When one of the middle sensors start reading HIGH that means the robot has moved a little towards the left or right. The following drawings show just a few of the possibilities that arise when the robot moves along a line through a maze.



When the right hand sensor moves over the line the robot must move a little towards the right to correct. When the left hand sensor is over the line the robot must turn towards the left to correct. That is why the robot zig-zags along…

We can follow a line using just two photo transistors on either side of the line with the IF and ELSEIF conditional operators. This system can follow a straight line, even follow curves, but if you have a 90 degree corner the robot will fall off the line.

For this reason a "beginner" maze can be limited to 45 degree turns.

Slight differences in the motors of a robot means it might not track exactly straight, and can pull slightly to one side. As the robot drifts off the line the program receives a HIGH value and it adjusts the motor pulses to slightly change the direction of the robot. Problem is the robot will probably overshoot the line, so when that happens the program will change the pulses to correct the direction of the robot again and try to keep the robot overhead the line in the following code:

```
// REMEMBER TO CHANGE Ax and Ay IN THE CODE TO THE MIDDLE SENSOR PIN NUMBERS
// IF THE PROGRAM DOESN'T WORK TRY SWAPPING Ax AND Ay AROUND

#include <Servo.h>                          // Include servo library

Servo servoLeft;                            // Declare left and right servos
Servo servoRight;

void setup() {
  servoLeft.attach(11);                     // Attach left signal to pin 11
  servoRight.attach(10);                    // Attach right signal to pin 10
}

bool leftIsOverWhite;
bool rightIsOverWhite;

void loop() {
  leftIsOverWhite = analogRead(Ax) > 800;
  rightIsOverWhite = analogRead(Ay) > 800;

  if (leftIsBlack)
  { servoLeft.writeMicroseconds(1500);      // TURNING - LEFT - WHILE DRIVING FWD
    servoRight.writeMicroseconds(1450);
  }
  else if (rightIsBlack)
  { servoLeft.writeMicroseconds(1550);      // TURNING - RIGHT - WHILE DRIVING FWD
    servoRight.writeMicroseconds(1500);
  }
}
```