MultiWingSpan

BBC micro:bit
Following The Lines (PXT)

Introduction

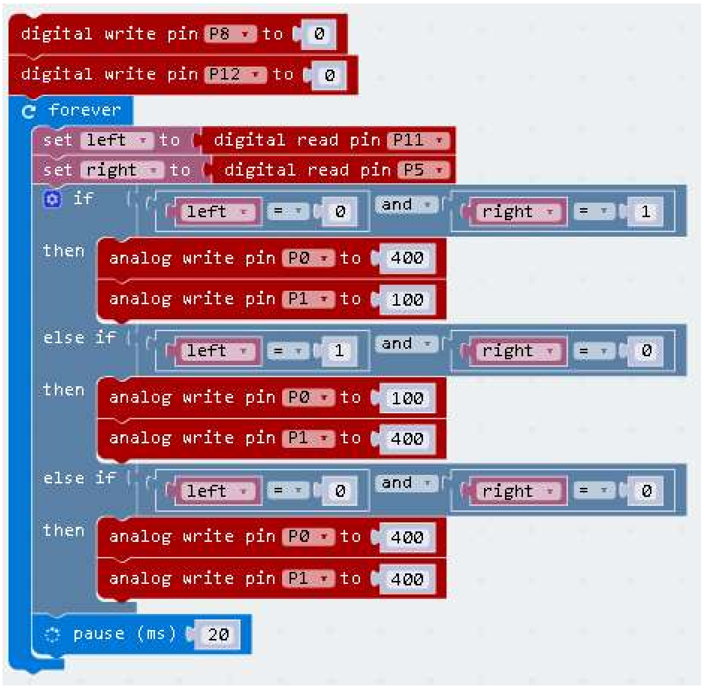
There are two line sensors on the bottom of the Bit:Bot. They are attached to the same pins as the micro:bit buttons. These are pins 5 and 11. You use a read_digital() on these pins to see if the sensor is detecting a black line.

By making frequent adjustments to the speed of the left and right motors, based on the readings from the line sensors, you can make your robot follow a line or drive laps of a cricuit. If the left-hand sensor senses a line, you need to drive the right hand motor more quickly, so that the robot turns to the left. You do the same for the right hand sensor.

The blog entry and the PCB silkscreen contain contradictory information on the line sensors. I am deliberately avoiding being too specific here. It will be useful for you to experiment with how which sensor is on which pin and what values you get from a digital reading for actions in the real world.

Programming

The line-following code below works just about for a Lego Mindstorms track. That has a pretty thick black line and this might need to be tweaked to work more nicely on a different track.



```
digital write pin P8 to 0
digital write pin P12 to 0
forever
    set left to digital read pin P11
    set right to digital read pin P5
    if left = 0 and right = 1
    then analog write pin P0 to 400
         analog write pin P1 to 100
    else if left = 1 and right = 0
    then analog write pin P0 to 100
         analog write pin P1 to 400
    else if left = 0 and right = 0
    then analog write pin P0 to 400
         analog write pin P1 to 400
    pause (ms) 20
```

And the JavaScript for that is,

```
let right = 0
let left = 0
pins.digitalWritePin(DigitalPin.P8, 0)
pins.digitalWritePin(DigitalPin.P12, 0)
basic.forever(() => {
    left = pins.digitalReadPin(DigitalPin.P11)
    right = pins.digitalReadPin(DigitalPin.P5)
    if (left == 0 && right == 1) {
        pins.analogWritePin(AnalogPin.P0, 400)
        pins.analogWritePin(AnalogPin.P1, 100)
    } else if (left == 1 && right == 0) {
        pins.analogWritePin(AnalogPin.P0, 100)
        pins.analogWritePin(AnalogPin.P1, 400)
    } else if (left == 0 && right == 0) {
        pins.analogWritePin(AnalogPin.P0, 400)
        pins.analogWritePin(AnalogPin.P1, 400)
    }
    basic.pause(20)
})
```

Challenges

1. Whatever you want to do later on with the line sensors, it's worth doing a lot more experimentation with how you want to react to reading the lines and how small adjustments make a difference to the overall effect. Try different thicknesses of lines, different amounts of bend on a corner and see what impact that has.
2. Once you have something that can follow a line, you need a good track. Go wild with black masking tape on white paper. Make a complicated route and then work out how to make the best program

to navigate the route without deviating from the course. If you can, race against a buddy's program or robot.

3. Longest track EVER. Or at least the longest one you can make where you are.
4. You can make a track that leads to something you want to bump into. That might be something simple that you want to knock over, or more complicated electronics that sense and react to the collision on a second micro:bit.
5. A little bit of Neopixel action might help you to work out what is going on.
6. You can use the line sensors to find something on an otherwise plain track. Finding a line requires a different approach. You need to think more carefully about how you react to sensor readings.
7. You don't have to follow lines, you can avoid them, or change speed when you cross them. You can do pretty much any movement when you detect them.
8. The truly brave thing to do is to create a maze of lines and write a program that looks methodically for a way out. There are some difficult concepts involved in mazes and how you can encode them for a computer program to understand. You might want to start with writing programming logic for some simple behaviours and a way to detect that escape has taken place. Maze solving is a huge topic.