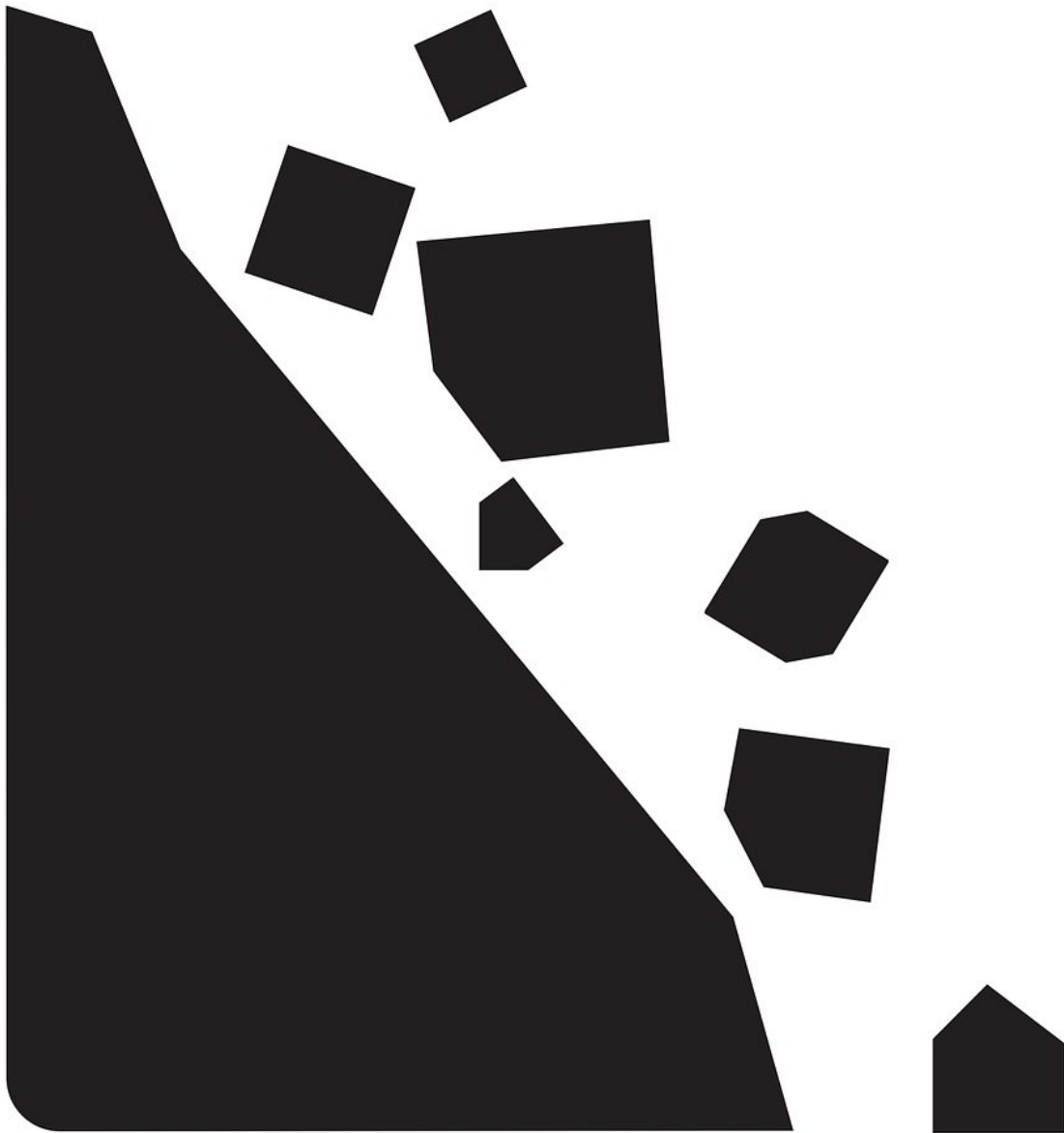


## A - Debris on the Track

Rocks have fallen onto the line for the robot to follow, blocking its path. We need to make the program clever enough to not get stuck!



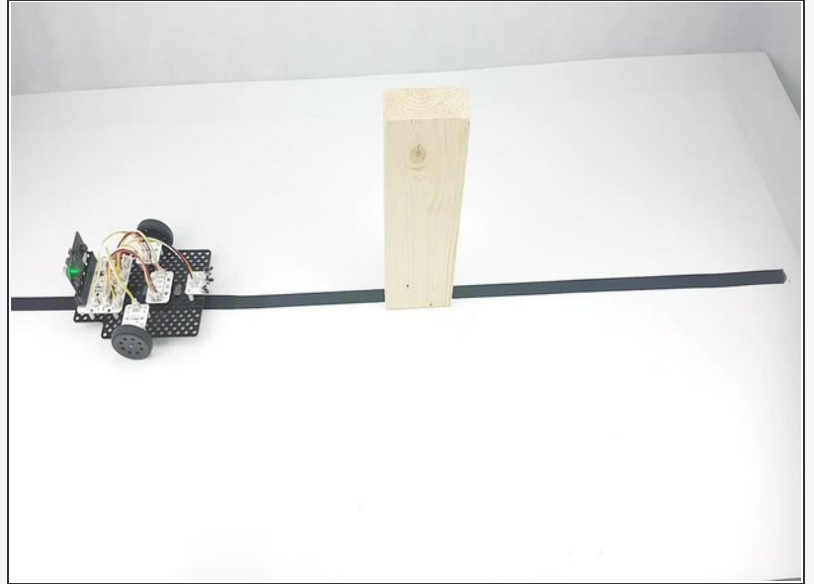
# INTRODUCTION

Rocks have fallen onto the line for the robot to follow, blocking its path. We need to make the program clever enough to not get stuck!

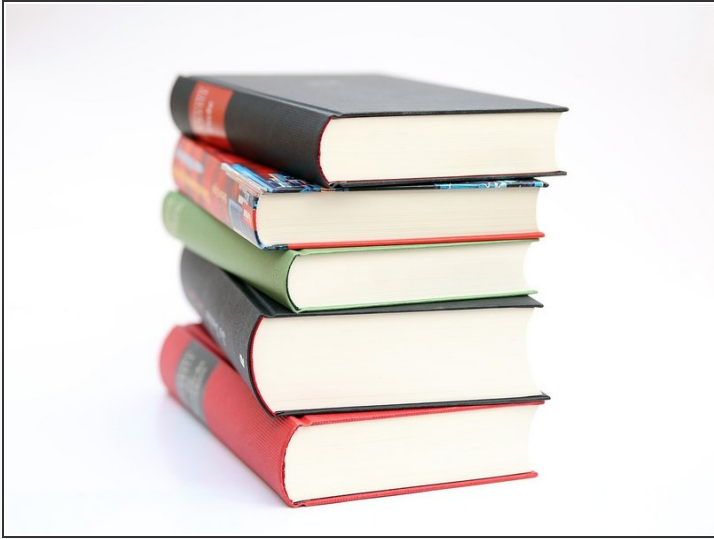
## Step 1

### Obstacle Avoider

- In this lesson we're going to combine the **line follower** program with some **obstacle avoiding** code!
- This should allow the robot to easily drive **around** obstacles, and then **find the line again**.
- Hopefully yours should work like our example in the **video**!



## Make the Obstacle



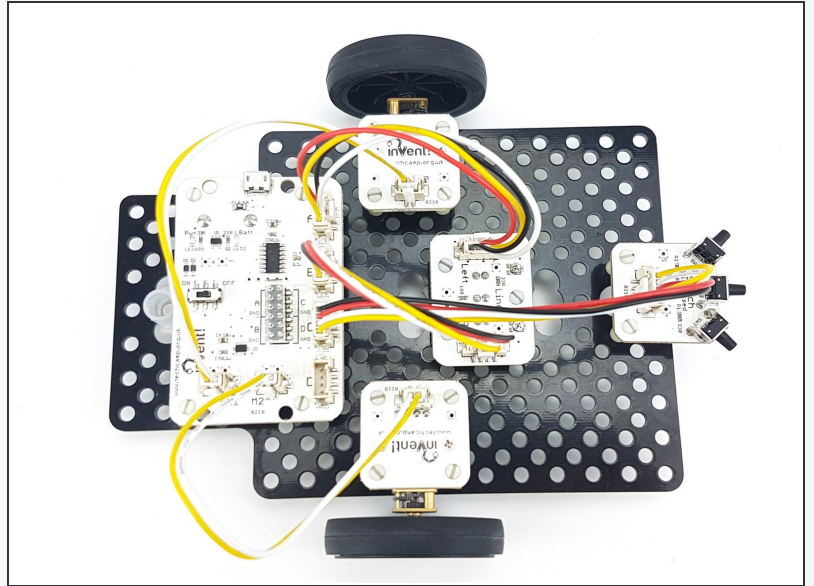
- Let's put an **obstacle** on the track for the robot to avoid.
- You can use anything you like, as long as it is at least **4cm tall**.
- Put it across the track **wherever you like!** However, having a **straight section** of track both sides of the obstacle will make it **much easier**.
- You can use **books**, **wooden blocks**, or **anything else** you have around. You might need to **tape it down** if its very light so the robot doesn't just push it around.



### Step 3

## Setup the Robot

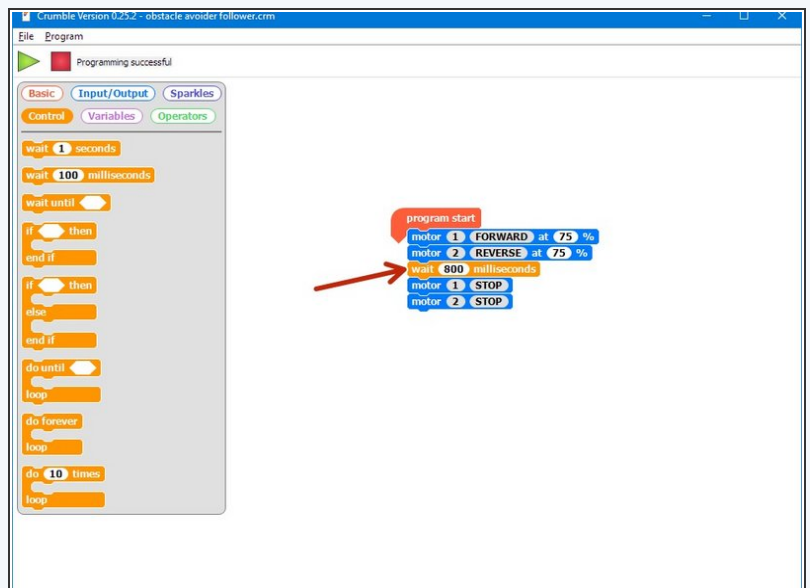
- We're going to use a **switch module** to detect when we hit the obstacle.
- Assemble your robot like the picture
- Make sure the switch is in the **middle**!



### Step 4

## 90 Degree Turn

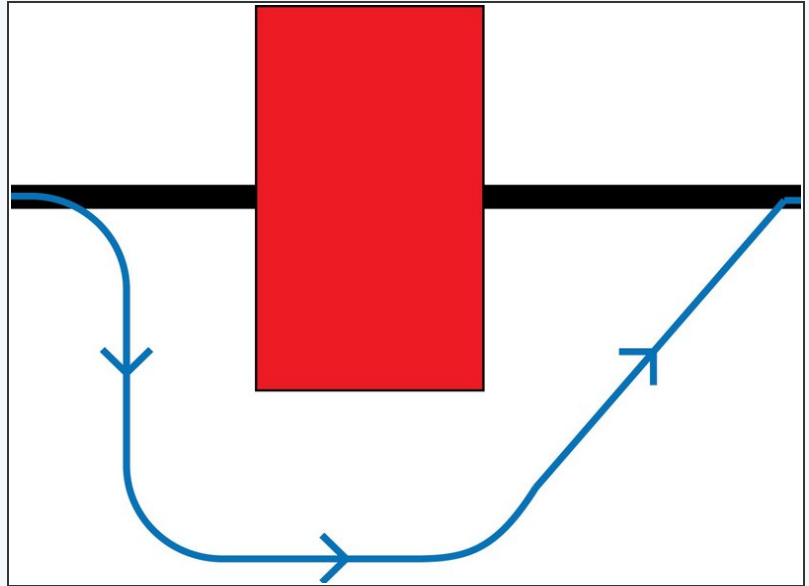
- We're going to need to turn the robot **90 degrees** again to complete this challenge.
- Write the simple program in the picture and work out **how many milliseconds** you need to wait for your robot to turn **90 degrees**.



## Step 5

### Avoiding Path

- We need to decide **how** we want the robot to move so it **avoids the obstacle**.
- You might need to **change** this slightly depending on the **size** and **shape** of your obstacle! When the switch is pressed we need to:
  - **Turn right 90 degrees**
  - Drive **forwards**
  - **Turn left 90 degrees**
  - Drive **forwards**
  - Turn left **45 degrees**
  - Drive forwards until 1 sensor finds the line (**LO**)



## Step 6

### Make the Moves

- Using the **wait time** you worked out earlier for turning 90 degrees, write a simple program to make your robot **drive around the obstacle**.
- Our code is just an example - you will need to **change** all of the times depending on the **size** of your obstacle!
- **Test** your code properly until your robot **reliably** drives around the obstacle

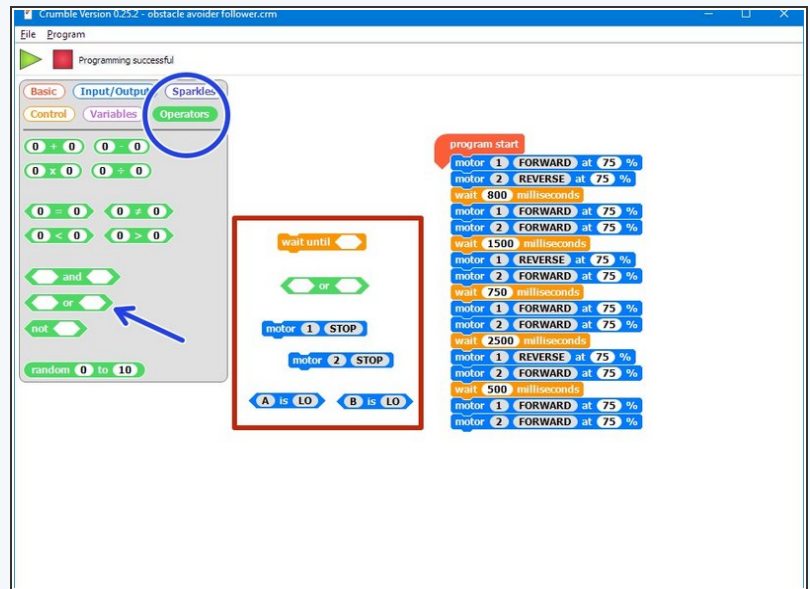


Driving at **45 degrees** to the line after the obstacle is **very important** - if we drive directly at the line, both sensors might find the line at **exactly the same time** and the robot might **drive into the obstacle**!

## Step 7

### Stop on the Line

- After the robot has driven around the box, it should **drive forwards** until **one** of the sensors finds the line again.
- Add a **wait until** block and two more **motor** blocks at the end of your code, so the robot **stops** when it finds the line again.
- You will need to use an **OR block** for this - we want to stop if A is LO **OR** B is LO. You can find it in the operators menu.
- There are some **hint blocks** if you need them!

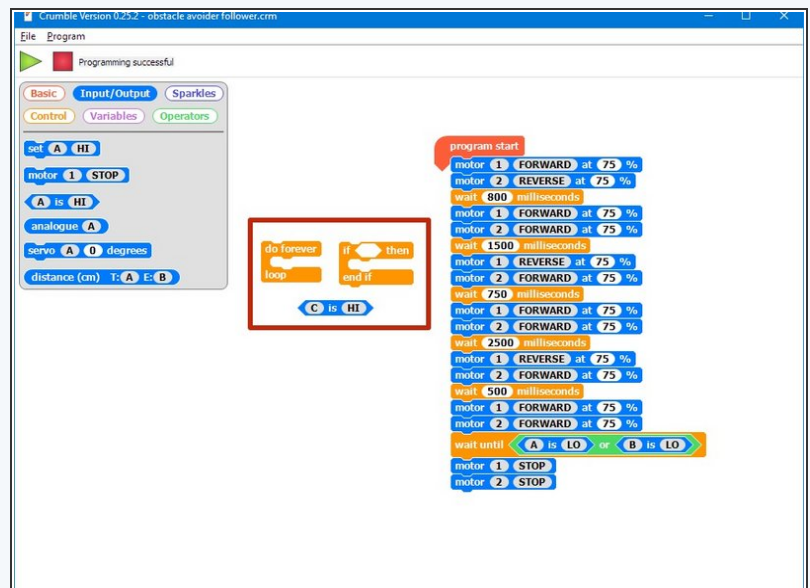


## Step 8

### Add the Switch

- **Almost there!** Let's add an IF block so our sequence is only triggered when the switch is **pressed**.
- There are some **hint blocks** if you need them, but hopefully you are a Pro at using switches by now!

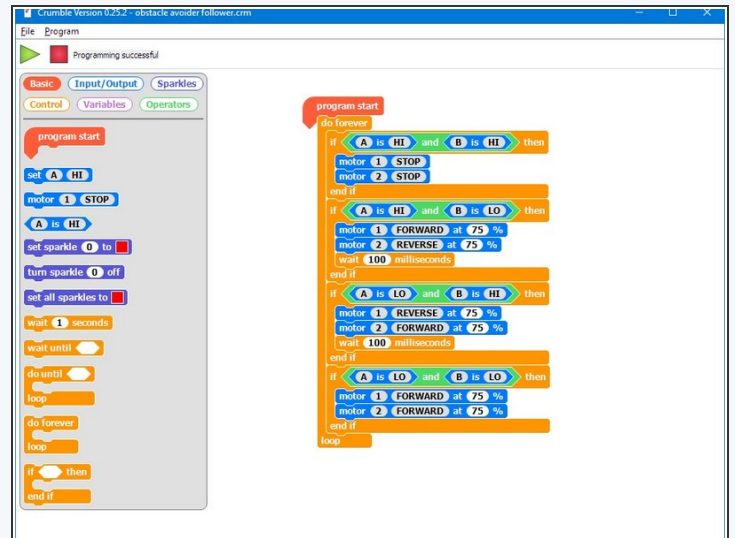
⚠ Don't forget the **do forever loop**.



## Step 9

# Obstacle avoider & line follower

# Challenge!



- Time to combine the **obstacle avoiding** code with our **line follower** code.
- Load up your **2 sensor line follower code** - it should look similar to the one in the picture.
- **Add** your obstacle avoiding sequence to the line follower code, and test it on the track - hopefully it drives round the obstacle and **finds the track again!**
- If it doesn't work, try **adjusting** the code and **keep testing** until it works really well!

## Step 10

### Combined robot with sparkles as well

- If you've finished all that, add your **Sparkle module** back on to the robot, and add the blocks back in to display the **line position** with the red/green Sparkles.
- When the robot is avoiding obstacles, make the Sparkles do something else so we know the robot is **driving round the obstacle**.
- They can do **anything you like**: flashing white, orange or something else entirely - **the more impressive the better!**

**Extension**  
**Challenge!**

