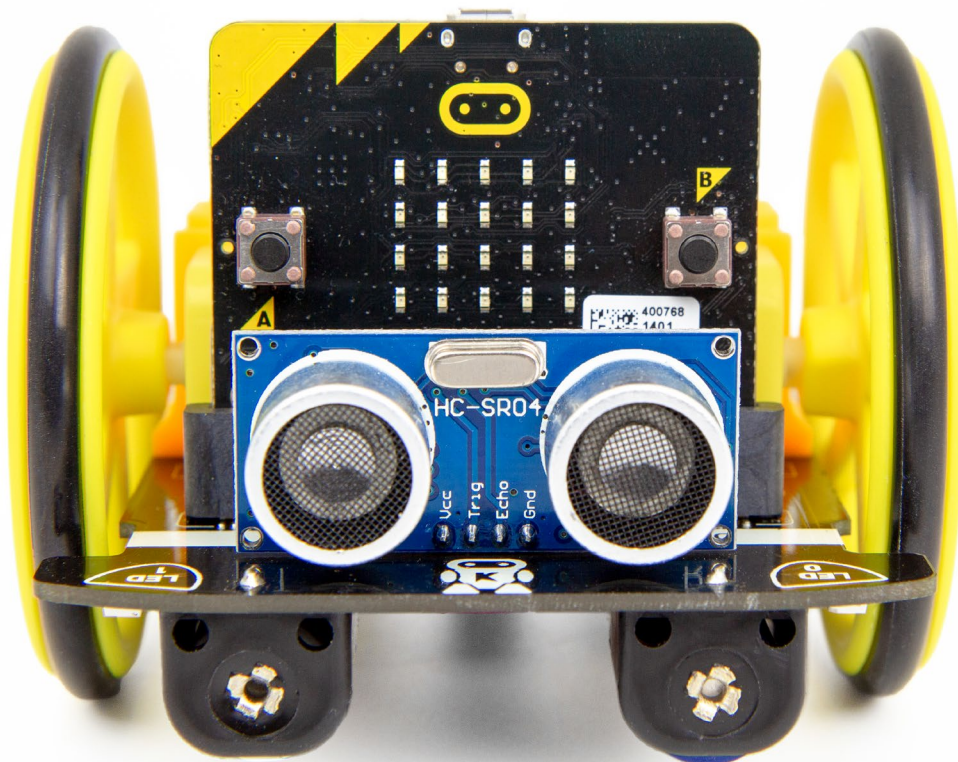


---

# :MOVE MOTOR LINE FOLLOWING TUTORIAL

KITRONIK RESOURCES



## INTRODUCTION

---

Learn how to use the :MOVE Motor's Line Following Sensors to navigate around a marked out track.

## SETTING UP

### EQUIPMENT REQUIRED:

- 1 x BBC micro:bit ([www.kitronik.co.uk/5613](http://www.kitronik.co.uk/5613)),
- 1 x :MOVE Motor Buggy ([www.kitronik.co.uk/5683](http://www.kitronik.co.uk/5683))

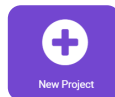
### ADDING IN CUSTOM MAKECODE BLOCKS:

We have made custom coding blocks especially for the :MOVE Motor, which helps to make coding super simple within Microsoft MakeCode.

To add these blocks, follow the steps below:

**STEP 1:** Bring up the MakeCode Block Editor - ([makecode.microbit.org](http://makecode.microbit.org)).

**STEP 2:** Click 'New Project'.

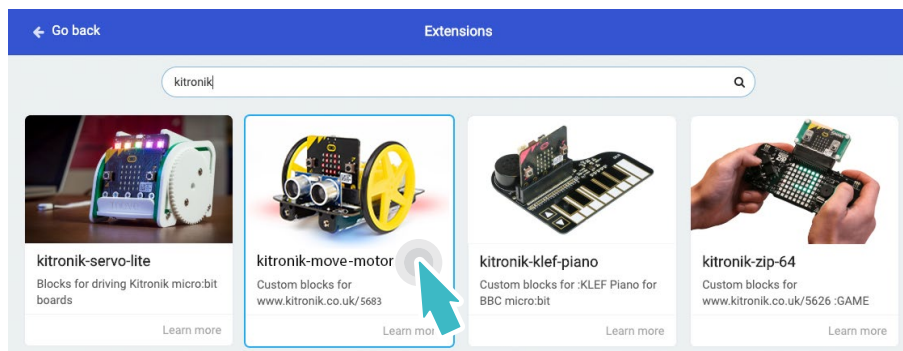
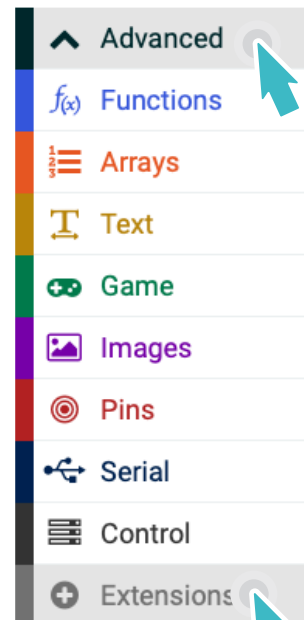


**STEP 3:** In the toolbox towards the left of the screen, select the 'Advanced' section. Additional block categories will appear below.

**STEP 4:** Select 'Extensions'.

**STEP 5:** In the pop up's search bar type 'Kitronik'.

**STEP 6:** Locate & select the 'kitronik-move-motor' box.



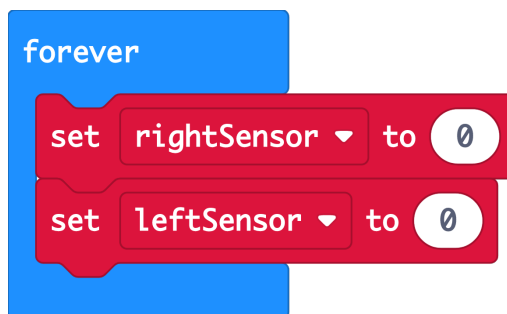
## THE TUTORIAL

**STEP 1:** We're going to create a simple program which continuously checks whether the :MOVE Motor is drifting away from the line it's following, and makes small corrections if it is.

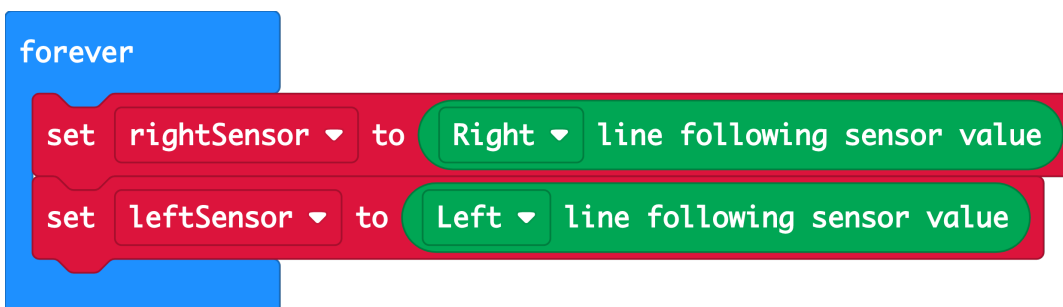
The :MOVE Motor has 2 sensors underneath to allow it to follow a line of contrasting colour to the background. We will start by reading the sensor values.

Create 2 variables, called 'leftSensor' and 'rightSensor'.

Place a 'set leftSensor' and a 'set rightSensor' into the forever loop.



**STEP 2:** Now we need to read the sensors values. From the Sensors section of the 'MOVE Motor' category, drag in a 'Left line following sensor value' block and put it into the 'set leftSensor' statement. Do the same for the 'set rightSensor' but change the drop down to read the 'Right' sensor value



### SENSOR VALUES

Because the sensors on the :MOVE Motor are analogue their value changes depending on what the surface below them is.

We know we are trying to follow a contrasting line, so if the 2 sensors are reading different values one must be on the line and one off.

If they read the same then either they are both sensing on the line - if the line is wide, or they are both sensing the background - if the line is thin.

Because we can compare the sensors against each other it doesn't matter what the sensing value is, only if there is a difference or not.

## :MOVE MOTOR - LINE FOLLOWING TUTORIAL

**STEP 3:** Create a variable called 'sensorDifference' to hold the difference in the sensor values. Set 'sensorDifference' to be 'leftSensor' - 'rightSensor.' Because we only care if the values are different we can use the 'Absolute' block so we don't have to worry about a negative number.

```

forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  
```

**STEP 4:** If the sensors are different then we must be heading off the line, but if they are the same we are still following it. Add an 'if else' block below the 'set sensorDifference' block.

```

forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  if true then
  else
  
```

```

forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  if true then
  else
    show leds
  
```

We will use the 'else' section for when we are still on the line. Add a 'show LEDs' block into the 'else' and draw a straight up arrow to indicate we would be going forward.

## :MOVE MOTOR - LINE FOLLOWING TUTORIAL

**STEP 5:** If we are leaving the line we will need to decide which way, and take corrective action. Into the 'if' condition add a '>' check to see if 'sensorDifference' is greater than 10. We give the difference check a little range to allow for tolerance in the sensing.

```

forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  if sensorDifference > 10 then
  else
    show leds
  
```

Into the first 'if' add an additional 'if else'. We will use this inner 'if else' to decide which way we need to turn to stay on the line.

```

set sensorDifference to absolute of leftSensor - rightSensor
if sensorDifference > 10 then
  if true then
  else
    +
  else
    show leds
  
```

### SENSOR VALUES

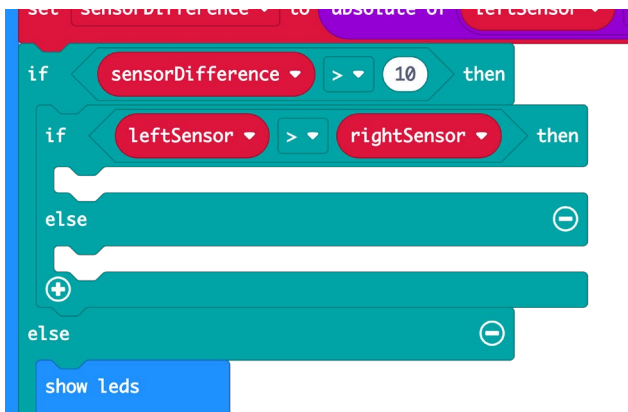
The sensors on the :MOVE Motor give a low value if the surface they are sensing is dark (none reflective) and a high value if the surface is light (reflective).

As we are looking for a dark line on a light background we know that if the left sensor is no longer over the line, but the right sensor is still over the line then the left sensor value will be larger than the right one. We can use this knowledge to work out we need to turn to the right to get back over the line.

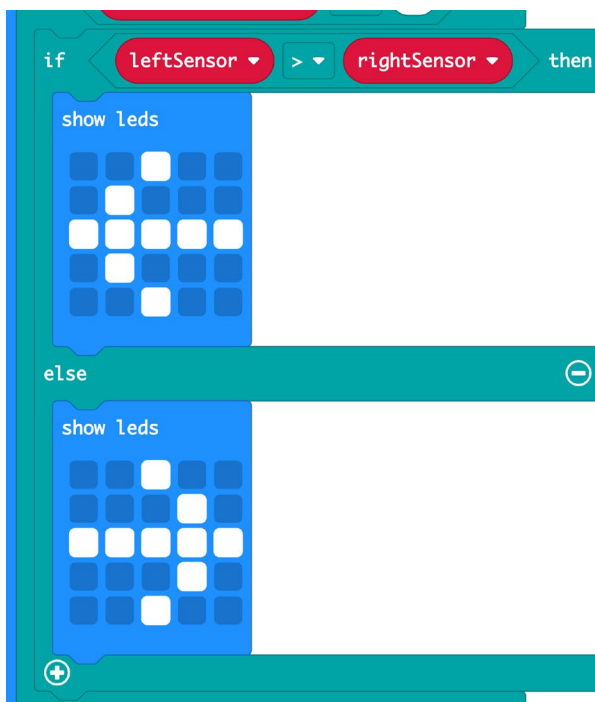
## :MOVE MOTOR - LINE FOLLOWING TUTORIAL

**STEP 6:** Into the inner 'if' condition add a '>' check to see if 'leftSensor' is greater than 'rightSensor'.

This means the left sensor is off the line and the right sensor is on the line.



To get back on the line we need to turn to the right. Add a 'show LEDS' block into the 'if' and draw an arrow to indicate which way to turn. The 'else' needs to turn the other direction, so add the opposite arrow into there .



**STEP 7:** That is the line following algorithm complete. If you have a micro:bit connected, click Download to transfer your code and switch on the :MOVE Motor.

### CHECKING THE LOGIC

Now we can check the logic is correct. Create a continuous track for the :MOVE Motor to drive around and set it on the line. The line should be about 10-20mm wide (black insulation tape is great for this).

We haven't yet put any motor driving commands into the code, but by moving the :MOVE Motor by hand side to side across the line you should see the LED arrows showing which way the logic is going to turn.

## :MOVE MOTOR - LINE FOLLOWING TUTORIAL

### DRIVING THE MOTORS

Next we will replace the arrows with the correct motor driving commands, so the :MOVE Motor can drive itself.

**STEP 8:** The simplest arrow to replace is the straight ahead on one. Change this to a 'Move Forward' block from the 'Motors' section of the 'MOVE Motor' blocks. Set the speed to 30.

```

forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  if sensorDifference > 10 then
    if leftSensor > rightSensor then
      show leds
    else
      show leds
    +
  else
    -
    move Forward at speed 30
    +
  
```

Don't forget to delete the Show LEDs block. Displaying the LEDs at the same time as following the line can cause the code to run too slow to work correctly.

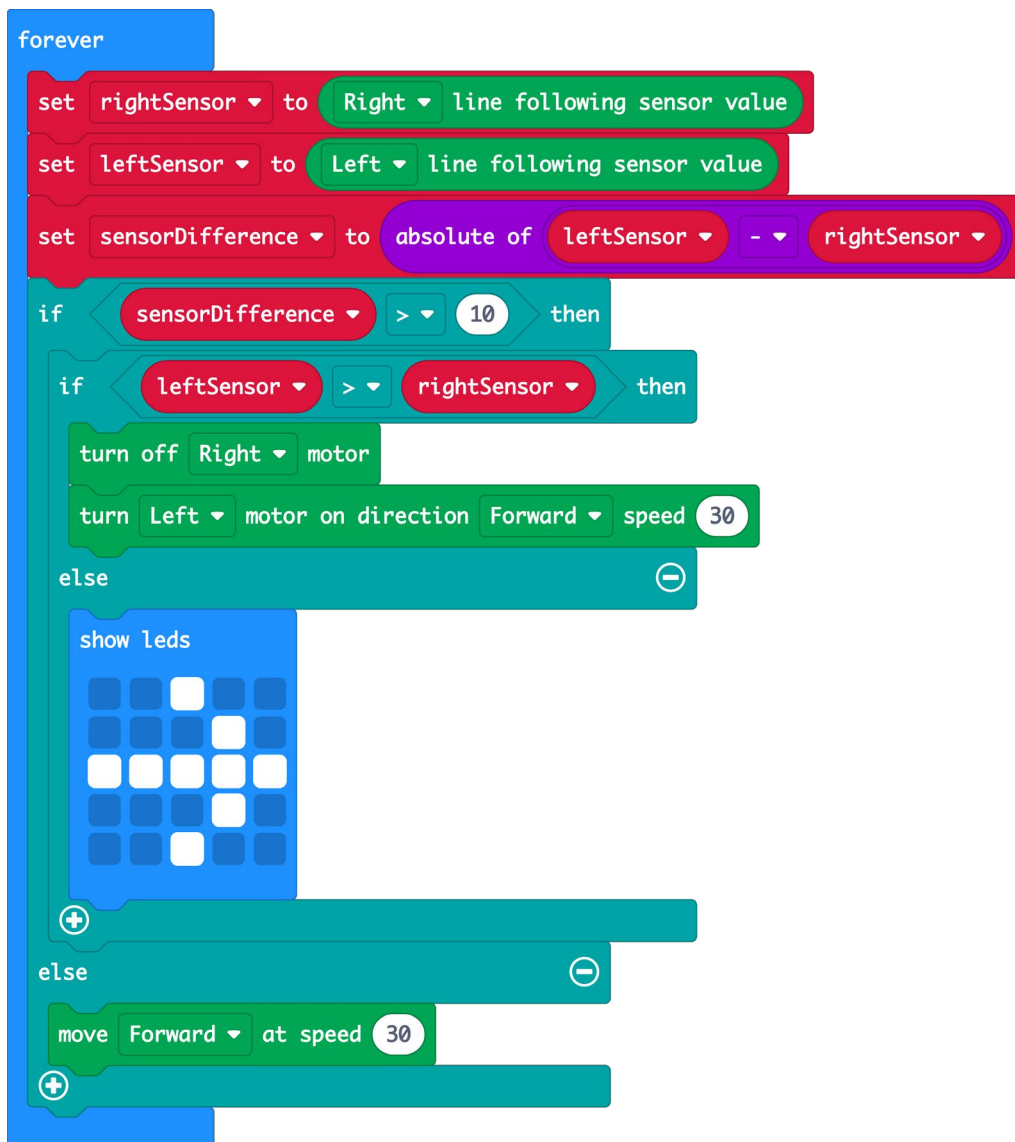
## :MOVE MOTOR - LINE FOLLOWING TUTORIAL

**STEP 9:** Now we need to deal with turning back onto the line.

In the 'if (leftSensor>rightSensor)' we want to turn to the right.

To do this we will stop the right motor and run the left motor. Replace the 'show LEDs' with a 'turn off Left motor' using the block from the 'Move Motor' 'Motors' section.

Change the drop down so that the 'Right' motor is stopped. Add a 'turn Left motor on direction Forward speed 0' block, and set the speed to 30.



```
forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  if sensorDifference > 10 then
    if leftSensor > rightSensor then
      turn off Right motor
      turn Left motor on direction Forward speed 30
    else
      show leds
  else
    move Forward at speed 30
```



## :MOVE MOTOR - LINE FOLLOWING TUTORIAL

Do a similar change to the 'else' section, but turning off the 'Left' motor and running the 'Right' motor at speed 30.

```

forever
  set rightSensor to Right line following sensor value
  set leftSensor to Left line following sensor value
  set sensorDifference to absolute of leftSensor - rightSensor
  if sensorDifference > 10 then
    if leftSensor > rightSensor then
      turn off Right motor
      turn Left motor on direction Forward speed 30
    else
      turn off Left motor
      turn Right motor on direction Forward speed 30
    else
      move Forward at speed 30
  
```

**CODING COMPLETE!** If you have a micro:bit connected, click Download to transfer your code.

Place your :MOVE Motor on the line, and switch it on. It should follow the line. If it doesn't then you might not have a dark enough line, the speed maybe too high, or the batteries might be going flat.

# :MOVE MOTOR - LINE FOLLOWING TUTORIAL

---



For any further queries or support, please visit the Kitronik website: [www.kitronik.co.uk/5683](http://www.kitronik.co.uk/5683)

Or get in touch:

---

**Telephone** +44 (0) 115 970 4243

**Sales email:** [sales@kitronik.co.uk](mailto:sales@kitronik.co.uk)

**Tech Support email:** [support@kitronik.co.uk](mailto:support@kitronik.co.uk)

**Web:** [www.kitronik.co.uk](http://www.kitronik.co.uk)



[kitronik.co.uk/twitter](https://www.kitronik.co.uk/twitter)



[kitronik.co.uk/facebook](https://www.kitronik.co.uk/facebook)



[kitronik.co.uk/youtube](https://www.kitronik.co.uk/youtube)



[kitronik.co.uk/instagram](https://www.kitronik.co.uk/instagram)



Designed & manufactured  
in the UK by 

