

Line Following Lab

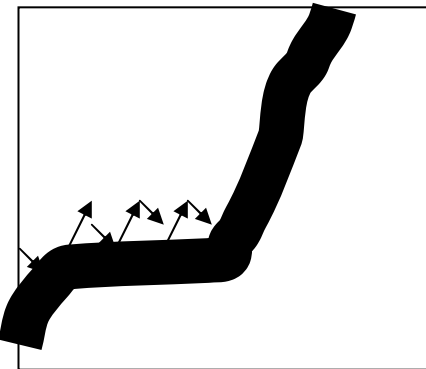
In this Lab we'll explore using NXT programming to get the robot to take light readings for a line/background and then using these readings in a line following program.

The advantage of using automatic light readings taken by the robot is that it removes the necessity for the programmer to manually take light readings, amend a program and download it again every time the light changes, and light values are subject to huge variations especially if you change the venue you are running in (especially in competitions!). You'll also probably find that the robots' readings will be more accurate than when a human takes these readings.

We'll also look at using 2 light sensors to line follow.

To make this explanation easier I'll assume that your line is black (or dark) and the background is white (or lighter than the line). However, these programs will work regardless of line/background colour but you may have to alter your robots motor direction behaviour.

Basic Method - Wiggle Line follower



This is the most basic line reading technique. The robot reads the edge of the tape i.e. the difference between one colour and the contrasting colour. IF the sensor sees light → turn towards dark, IF the sensor sees dark --- → turn towards light. By default the robot will move forward by controlling opposite wheels alternately.

The wiggly line follower has a tendency to waste time 'wiggling', so it's not the fastest method and because it only uses one light sensor it is prone to losing the line on tight corners.

Writing this program



Because we are using automatic light reading and data hubs later program the wiggly line follower using the COMPLETE palette.



Start by selecting a loop and place these icons inside this forever loop.

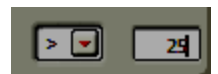
Select a Light sensor icon from the complete palette 'Sensor' menu.



This sensor performs (yellow background) a different function to the wait for light sensor (with the orange background) that you have used previously. It also has a data hub that you can pull down by clicking on the lower left border of the icon's border.

Data hubs allow you to share dynamic (constantly changing, in real time) data between sensors. We'll use the hub later.

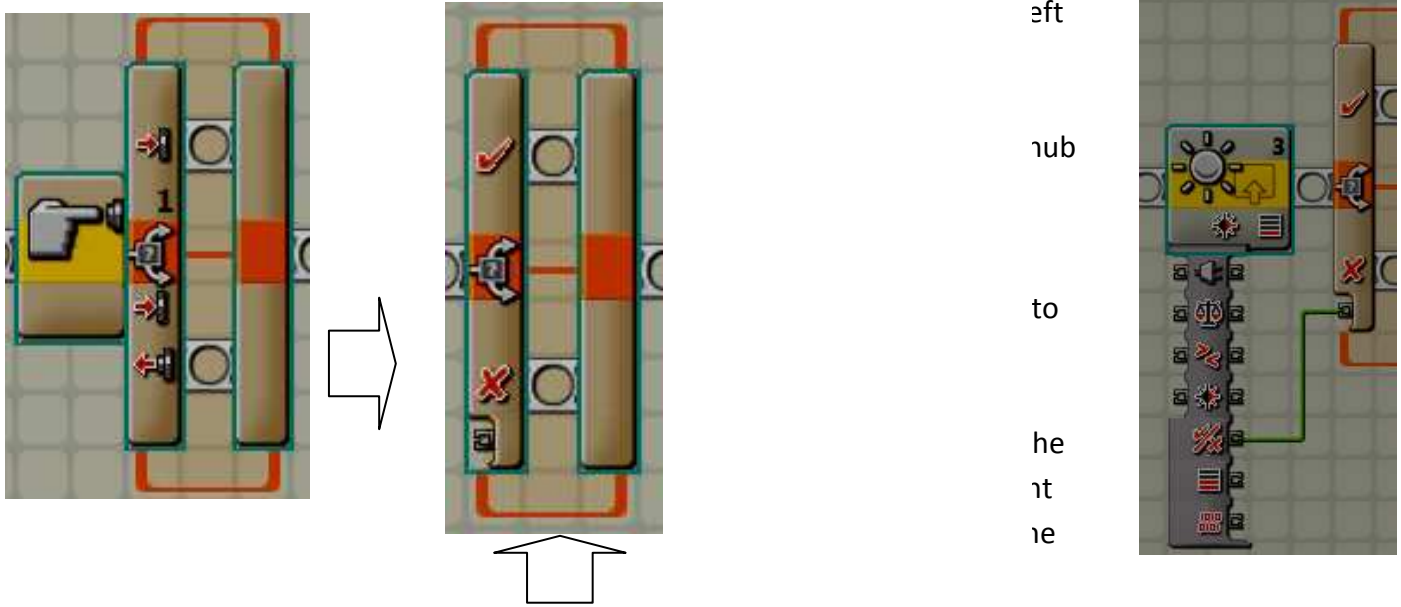
For the moment configure the manually light sensor to read



more than the value of your black line

Now select a 'Switch' block.

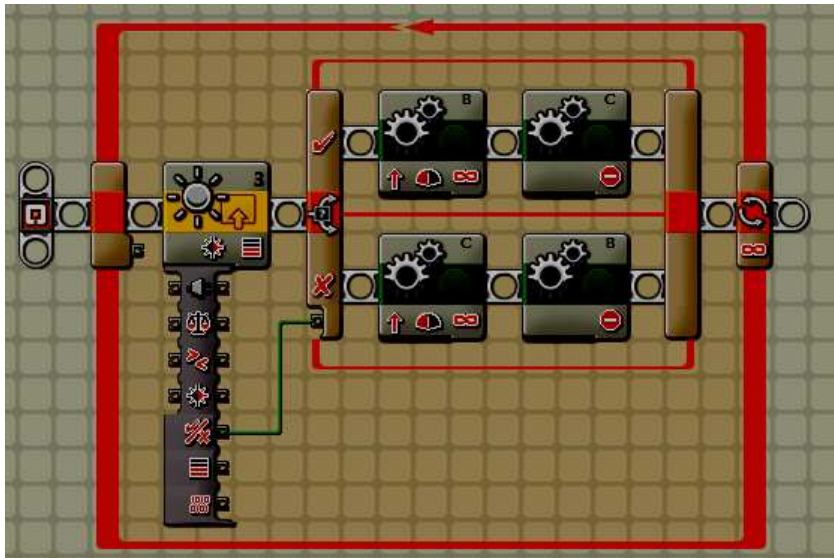
The switch block starts off as a touch sensor switch. Change the touch switch into a 'Logic' switch by using the configuration box. A Logic switch follows the top row if 'Yes' a condition exists or the bottom row if 'No' it doesn't . Our question will be 'Does the light sensor see black?- yes, take the top row, No- take the bottom row. This causes the robot to wiggle along the line



Switch configuration window:

- Control: Value
- Type: Logic
- Display: Flat view
- Conditions:

1.	True
2.	False



Finally, top row of the switch block should have Move blocks - B motor is on, forward power 50, duration unlimited . C motor is off

Bottom row – C motor is on, forward power 50, duration unlimited . B motor is off

Download and test. Change the light reading for the line if teh program doesn't work

Making the robot read the light sensor and use the value in the line reading program

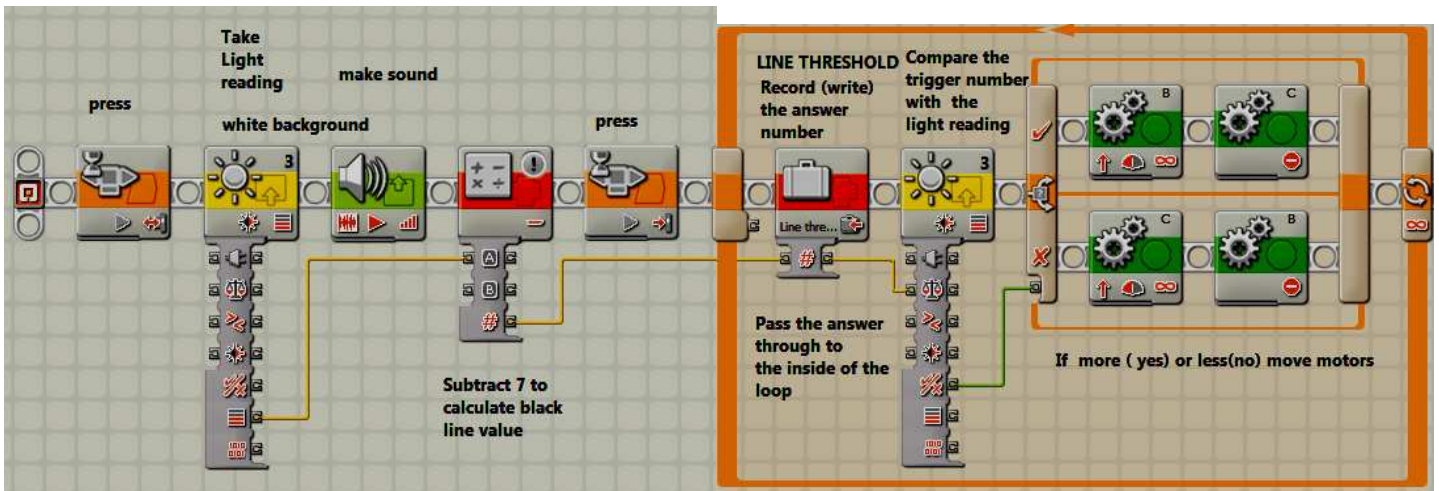
Please download and read Using Variable and Using Data Hubs from my website at <http://www.techxellenttraining.com.au>

Now we will add some programming that causes the robot to take a light reading when you press the NXT bricks right scroll button. Keep the 'wiggle' line following program and we'll add to it.

How it works

You will place the robot on the background colour (white or brighter than the line). Press the right scroll button on the NXT brick and the robot will then make a sound, take the light reading then subtract 7 from this reading. The robot will use this new number as the light reading for the line i.e. assumes that the line will be a minimum of 7 less than the background colour. To do the subtraction we use a math's block. Once the robot has calculated the light reading for the line it will store that number in a variable block. Storing the number locks in the reading so it doesn't change. Then the robot will use this reading (we'll call it a threshold reading) as it moves along the line comparing what it currently 'sees' with the reading (the threshold reading for black) it is 'seeing' in real time and reacting with the appropriate 'wiggle'. The second wait for press of scroll button is there to start the actual movement part of the program after the robot has taken it's reading.

This is Program 1 that you downloaded. It uses a single sensor 'wiggle' to follow the line using an automatic reading. The first 5 icons deal with taking a light reading automatically.

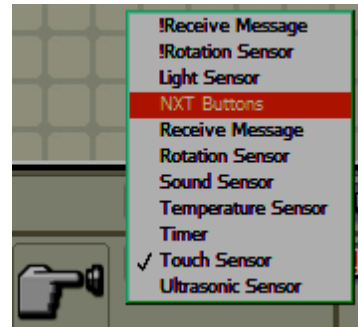
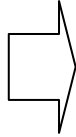


Programming Icons for automatic readings



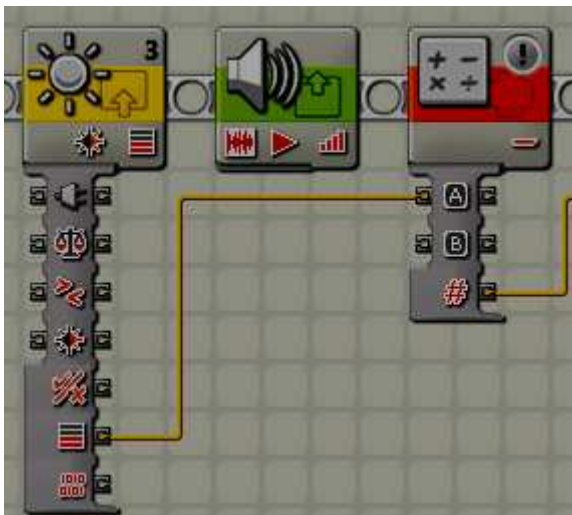
From the 'Wait' icons, select the wait for touch sensor.

Use the configuration box to change the touch sensor to NXT buttons, Right Scroll



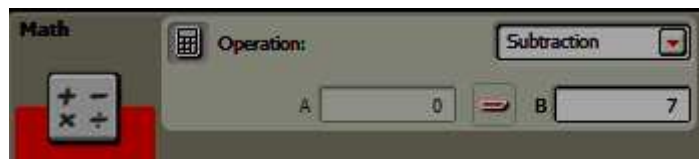
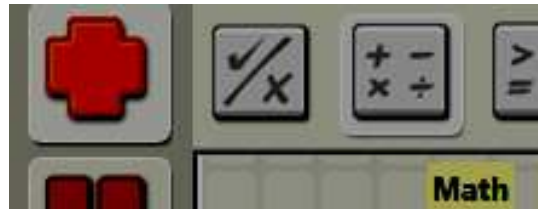
Next we will activate the light sensor to take the light reading.

Select the light sensor(yellow background) , then a play a sound icon.



Then select a Math's block icon.

You will find this icon in the Data sub menu of the Complete palette.



Configure the math's block to subtract 7. This should be low enough to represent the black line .

Now we need to send the light reading that the light sensor take when we press the NXT right scroll button to the math's block so that the robot can calculate the reading we want to use for the black line. Open the light sensors data hub and left mouse click on the Intensity port then go to the maths block and left mouse click on either A or B hub. Now the robot will subtract 7 from whatever light value it see's for the white background, we'll use this new reading to represent the black line. If the program is inconsistent following the line simply subtract more or less from the original white reading.



Follow the math's icon with another wait for the NXT Button right scroll button. This gives you time to place the robot back where you want it to start on the line. The robot won't actually move until you press the scroll button this second time though the program is still running.



Now we'll store the number the robot calculated in a variable block. This locks the reading in so that we can use it.

Select a Variable icon from the

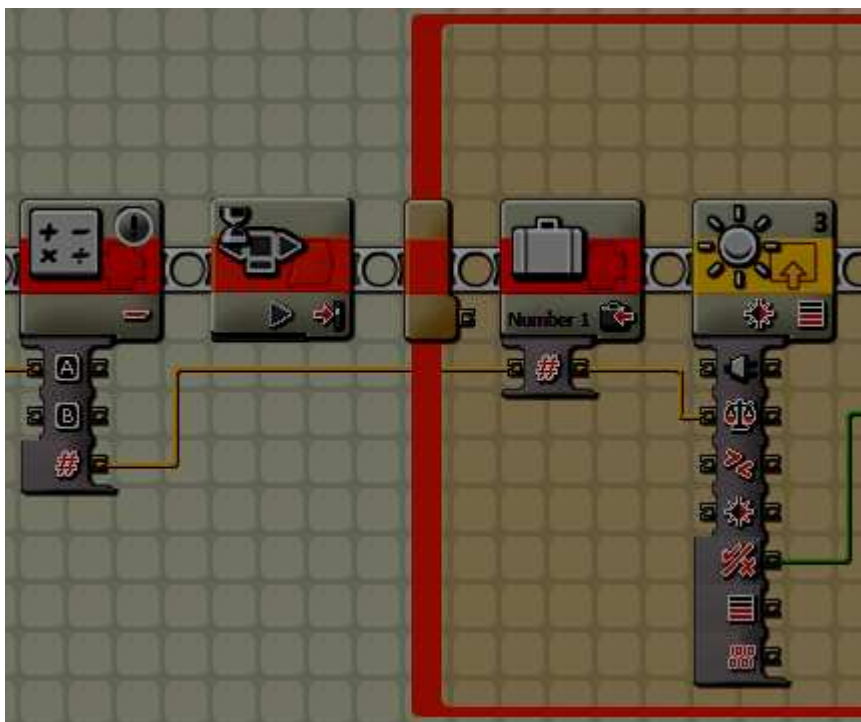
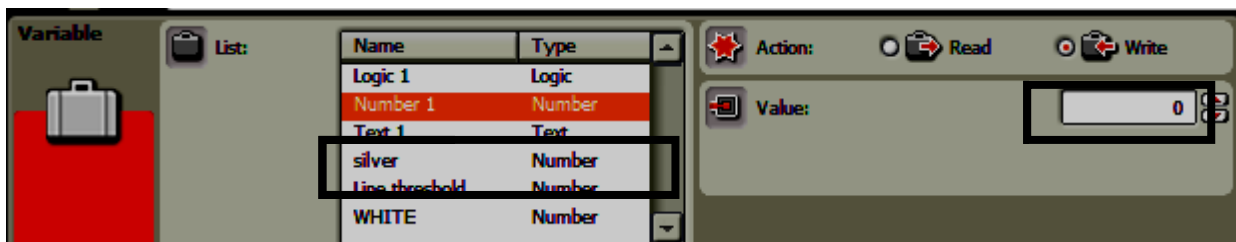
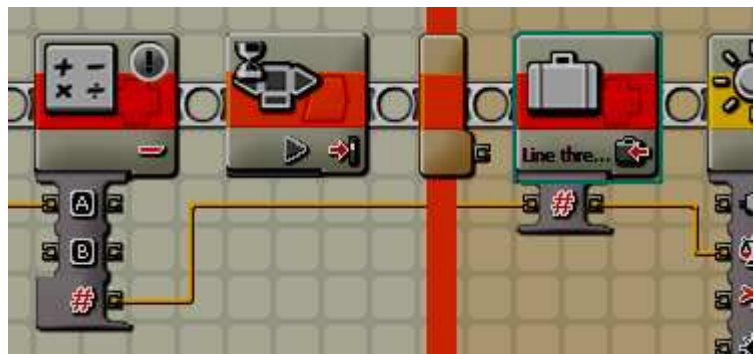
Data sub-menu



Place the variable block inside the loop.



Configure the variable so that it is a Number variable and check Write in the action setting



Connect the data wire from the math's block result hub onto the variable write hub and then the value wire from the variable hub into the Trigger Point port on the light sensors' data hub. Now the light sensor is comparing the number it calculated automatically with the light readings it is getting as it moves along the line and reacting accordingly. There's no need to actually input a light reading for the line following light sensor, even if you do the program will ignore it and use the number plugged

into its' data hub.

Two Light Sensors

Two light sensors will dramatically improve the accuracy of a line follower. Attach two light sensors to your robot, one on port 3 and the other on port 4. Start with the robot centred on the line with a light sensor on either side of the line on the white background Figure 1

Fig. 1

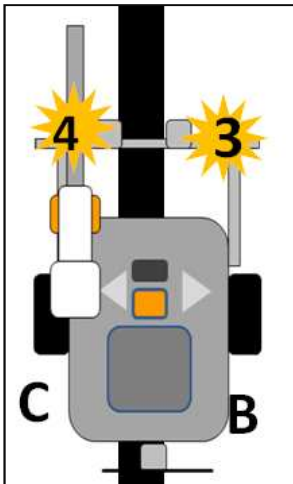


Fig 2.

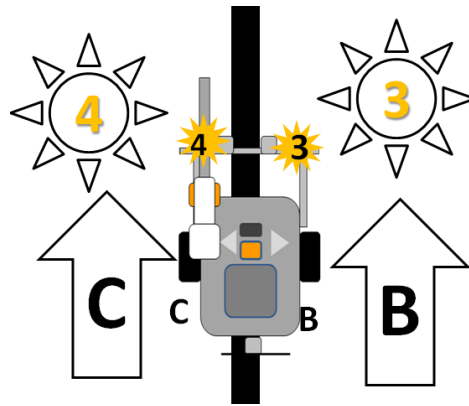


Fig 2 - The program will direct motor C to run forward if the light sensor on port 4 'sees' white. At the same time

The program will direct motor B to run forward if the light sensor on port 3 'sees' white.

This makes the robot run straight forward. BUT....

Fig 3- IF the light sensor on port 4 'sees' black (or dark)

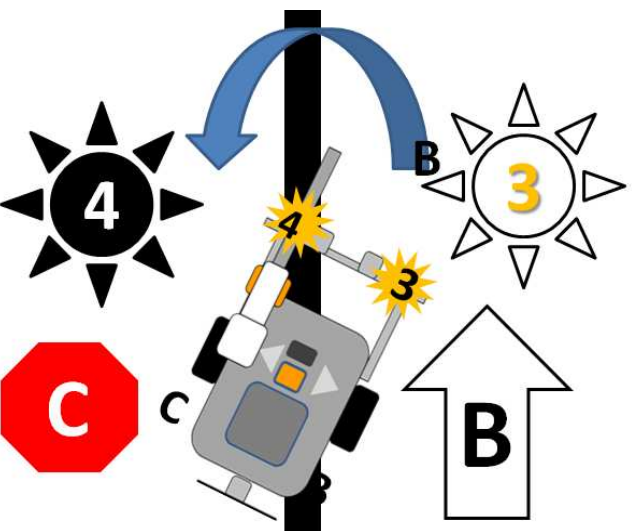
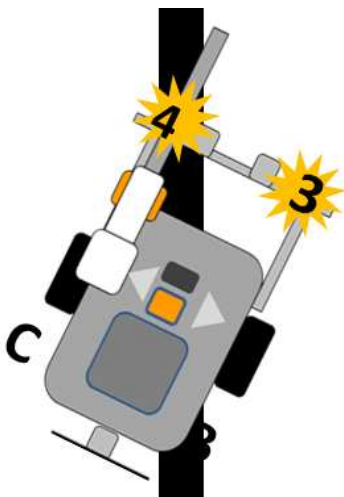


Fig 4 – Then C motor will turn off and B motor Will turn on . This will spin the robot to the left. This will happen until both light sensors 'see' white again (see Fig-1)

Fig 5- IF the light sensor on port 3 'sees' black (or dark)

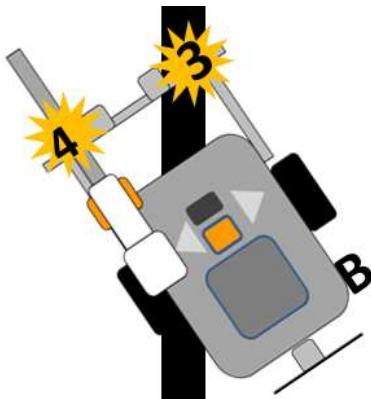
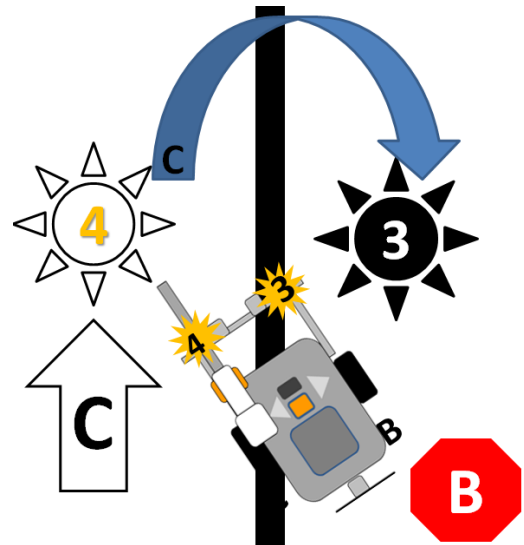


Fig6 – Then B motor will turn off and C motor will turn on . This will spin the robot to the right. This will happen until both light sensors 'see' white again (see Fig-1)



Writing the two light sensor line follower program

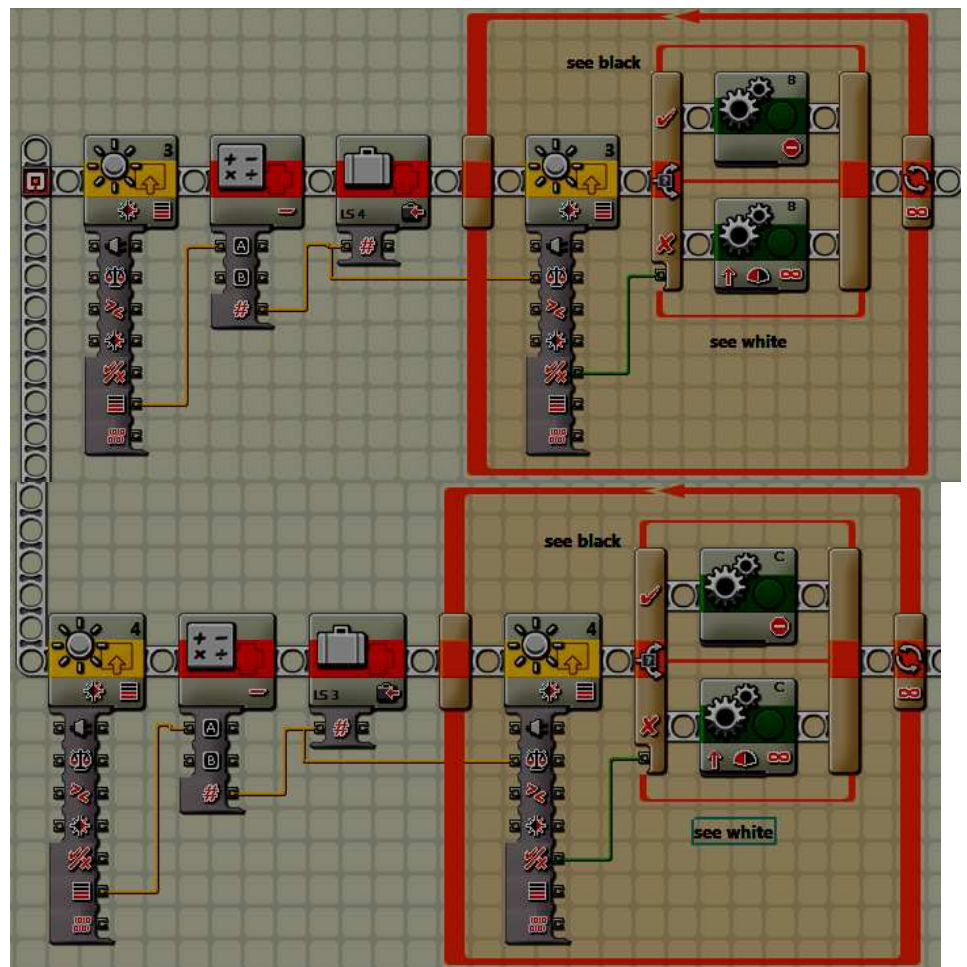
This is program 2 from your download.

This program includes an automatic light reading component but this time no NXT buttons are included to cause this to happen because the robot is intended to commence line following by being positioned with 2 light sensors , 3 and 4, each positioned on the white

background on either side of the line. The robot 'sees' white, takes a light reading for white and then subtract 7 for light sensor 4 and 3.

Light sensor 3 controls motor B and light sensor 4 controls motor c.

If neither light sensor 'sees' the black line the robot will



Using loop for #degrees to control duration of line following

Program 4 from downloads

This program now uses the degrees you logged to control the duration the robot follows the line for

