First LEGO League Programming Skills Exact Control

D

1 – Exact Turns

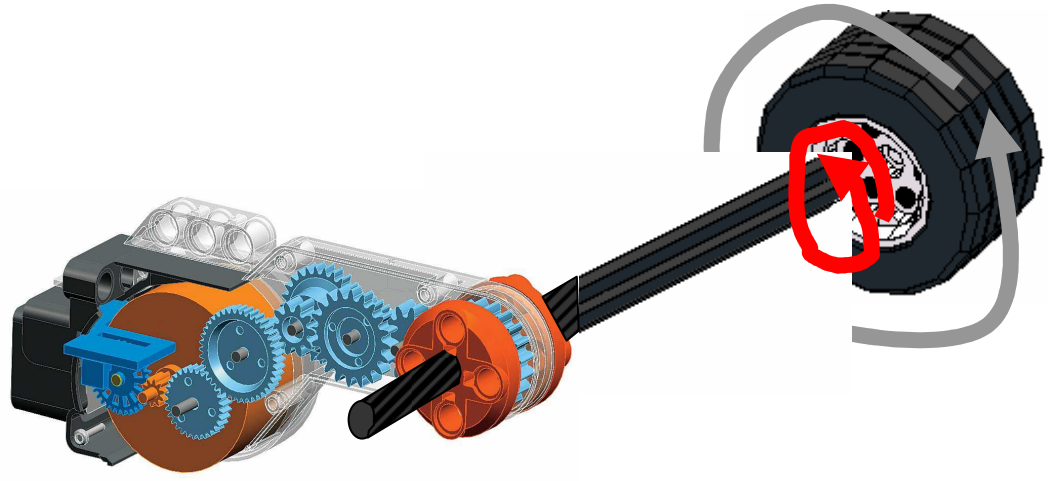
Successfully compleeting any First LEGO League challenges relies foremost on being able to find the object in question by positioning the robot exactly on the mat inside home base (registration) so that it can travel the exact distance required to the object to perform a task.

The LEGO NXT robot provides an in-built sensor that makes this process possible. Inside each motor is a rotation sensor that you, as the programmer, can control.

FOR THE NXT ROBOT MOTOR DEGREES = the number of degrees of rotation of the **AXLE,** in turn this setting will result in the wheel (and tyre) turning and causing the robot to move a specific linear distance .

DEGREES settings will not result in a specific degrees of turn by the physical robot, ie a 90° setting will not result in a 90° turn by the robot, rather a 90°turn of the axle resulting in the whee l rotating 90°and causing the robot to move forward or backwards a specific distance that we can accurately calculate(rotational movement resulting in linear movement, like a trundle wheel) .

The NXT standard tyres circumference measures 18 cm

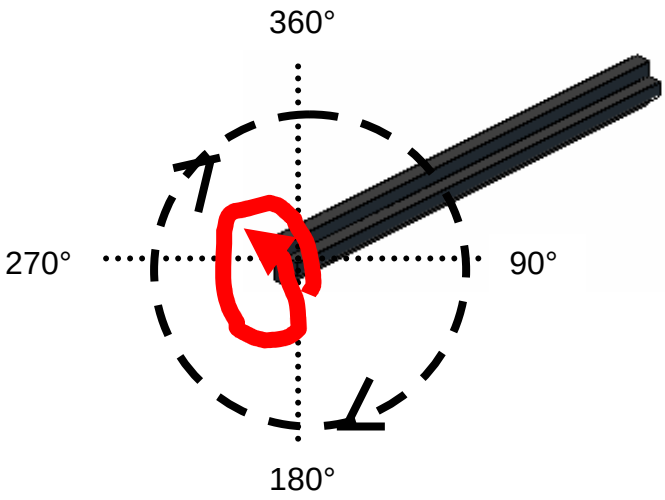


*The distances traveled by the robot when degrees are used as duration settings are dependent on the circumference of the tryes fitted. 18cm is the circumference of the standard balloon style tyres. If you change the wheels/ tyres or treads you will need to repeat these tests. If you change NXT bricks / motors you will need to repeat these tests

DEGREES

D

Both ROTATIONS and DEGREES reference the in-built encoder in the NXT motors . Before you start programming the NXT on a computer you need to experiment to see exactly how many degrees = what distance, you can then use the programming software to control the distance your robot travels exactly



The NXT Programming software illustrating the DURATION : Degrees option

The **DURATION** setting allows you to adjust

HOW LONG the motor will run for this **translates to how far your robot** will actually travel

Wait For (Duration) selections in the pull down menu:

Unlimited – Select this is you are using a switch (IF) block / Wait for block -sensors. The action is limited by the sensor parameters you set following the MOVE icon

Degrees – Degrees of rotation of the axle. This references the in-built sensor in the motors

Rotations – One rotation of the axle. This also references the in-built sensor in the motors.

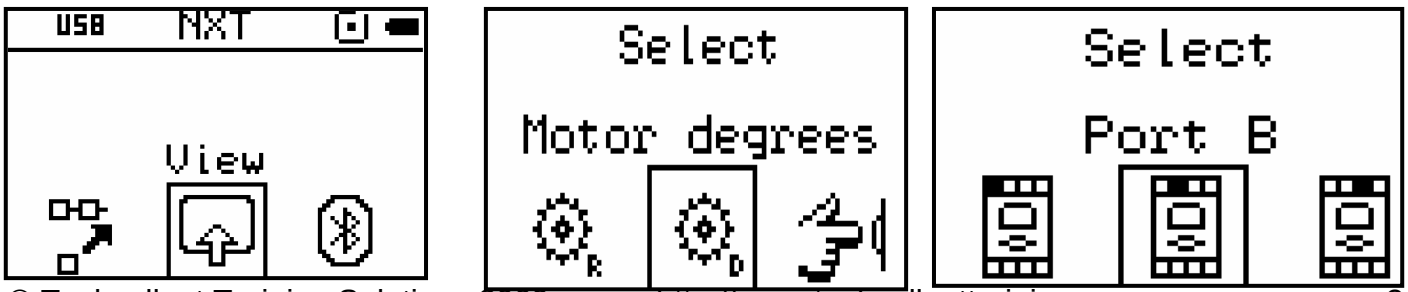
There are several ways of using the degrees sensor to program your robot to move exact distances in straight lines; for turning exact degrees of turn by the physical robot the method is a little more complicated. Here we will look at some of basic reliable methods:

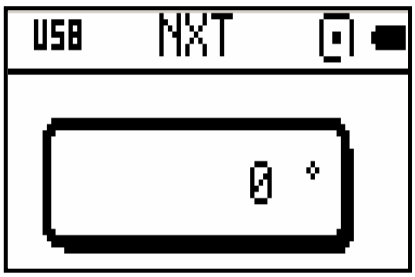
1a Make my robot do an exact turn - Let's sort out the 90°left hand turn first :

Note – This technique works for turns of any size and in either direction (simply reverse the wheel manipulation for the right hand turn. We're using the 90° left hand turn first because 90° turns are possibly the most useful.

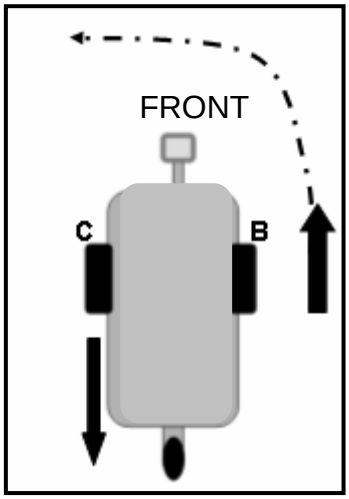
Test 1 – Use the VIEW function on your NXT robot to see how many degrees the axle has to turn to get the robot to physically turn 90°

1 ***2*** ***3***





Reset the degrees sensor to 0 by pressing the orange button



In this test we will be checking a left hand turn – this means that Motor B is the wheel that is moving forwards and Motor C's wheel will be moving backwards causing the robot to spin to the left. Different directions but for the same number of degrees.

(Reveres these directions for a right turn i.e. B backwards , C Forwards)

Make sure that the angle sensor is set to "0" (press the orange button to re-set to zero) Manipulate the B and C wheels of the robot to cause the robot to physically make a turn : Gently place a finger on the top of the B wheel and another finger on the C wheel °. Now push the B wheel forward while pushing the C wheel backwards until the robot has turned 90°. Now read the degrees in the robots VIEW window. This is the number of degrees the B wheel has to drive forward to cause the robot to turn 90°.

Test 1 Results →

Repeat this test at least 5 times. Record all 5 degrees measurements here and then decide which degrees you will use in your program to make the right hand turn .

Test 90°Turn Motor B	
1	
2	
3	
4	
5	
Selected degrees for 90°turn =	

Once you have got the turn correct and decided on the number of degrees turned by the axle for the turn you can use this in your "Square" program. Download and run your program .

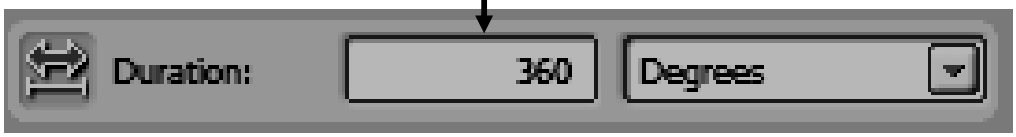
<u>Programming icon hints</u> –

Set the robots motors to spin on the spot by adjusting the steering

Left spin – steering **ALL THE WAY** to the left

Right spin – steering **ALL THE WAY** to the right

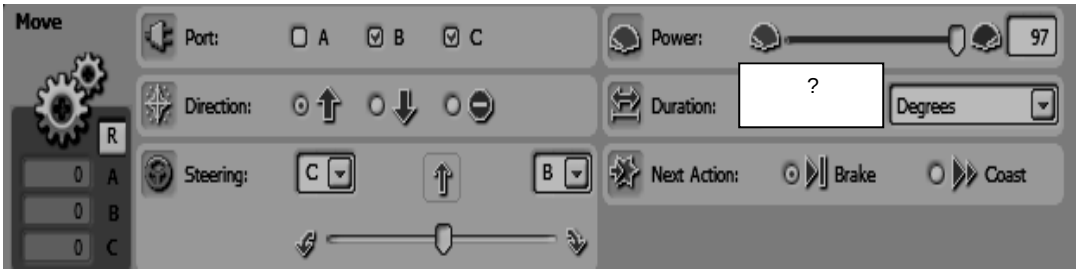
Set the robots motors to run for the number of degrees you have worked out for the right turn by setting the duration to DEGREES and **inputting your number here**



GOING AN EXACT DISTANCE – STRAIGHT

Reading the degrees sensor manually usually leads to errors. There is a far more precice method for turns but it involves quite complicated mathematics. Here we'll investigate using simple mathematics to program our robot to travel exact distances using the degrees sensor.

Programming Illustrated – Travel Straight C & B motors : DURATION DEGREES setting on your MOVE icon



Individual Robot's "Key" method

Test 2 Write a test program that has a MOVE icon straight forward for 900 degrees . Your robot will break straight after this , not coast.

Program this, download into your robot, run the program and measure exactly how far it travels.

Be VERY precise with your measurements , use mm.

Repeat this procedure at least 3 times, record your measurements here and either average or select one distance that you think is the most representative.

Example

A Robot traveled 40 , 41, 40.4, 40.4

--→select 40.45

Test 2 – measurements for 900°	
1	
2	
3	
4	
Add	
Divide by 4 =	distance

Test 2

<u>How far does your robot travel per degree?</u> This will be the key to exact control of your robot.

Complete these calculations on your worksheet

Distance traveled in cm	**cm**
Distance traveled converted to mm	**mm**
Divide the distance traveled in mm by 900 (degrees)	**mm ÷ 900 =**
This answer is the how far your robot travels per degree 	**= mm per °**

In the example where the robot traveled 404.5 mm (40.5cm) in 900° , this robot travels 0.44 per degree

Now we can use this KEY to work out how many degrees we need to program the robot for any exact distance, all we need to know is exactly what the distance is in mm.

We'll use the example robot's 0.44 per degree to show how :

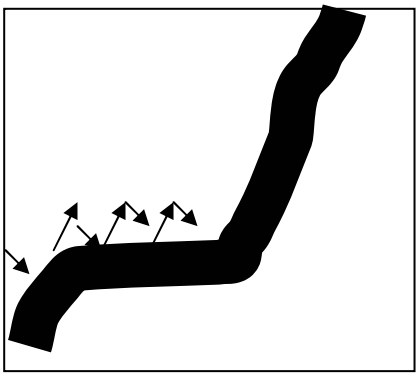
Example robot's key	**= 0.44 mm per °**
Distance I need to travel	= 450mm
Divide distance in mm by robots key	= **1022**
Degrees I need to program this robot for 45cm is	**1022 °**

1 e Use your robot's key to work out how many degrees you need for 45cm distance, do the calculations on your worksheet. Program and download.

My robot's key 	**= mm per °**
Distance I need to travel	= 450mm
Divide distance in mm by robots key	mm ÷ **(key)**
Degrees I need to program this robot for 45cm is	°

Line Following & Line readings

EDGE READING - LINE FOLLOWING

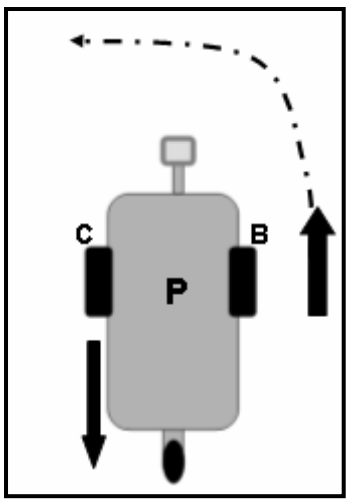


This is the most basic line reading technique. The robot reads the edge of the tape ie. The difference between one colour and the contrasting colour. IF the sensor sees light→ turn towards dark, IF the sensor sees dark---→ turn towards light. By default the robot will move forward by controlling opposite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

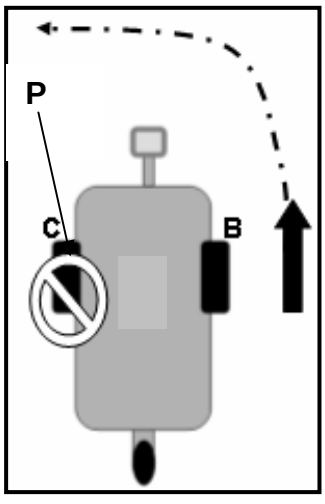
Moving Along the Line

Your robot will need to make a pivot turn to the opposite direction each time the light sensor detects a change in light (line reading colour) The pivot turn moves the robot forward I a "waddling" type motion, the spin turn would simply have the robot spin backwards and forwards in one place

Spin Turn to the left



Doughnut turn to the left



Line / light readings

You must take a light readings for the line and the background and then decide which reading will be the THRESHOLD light reading that causes the robot to “see” the line.

Write down at least 4 readings for the line and 4 readings for the background on your worksheet. Circle the number that you will use for your THRESHOLD

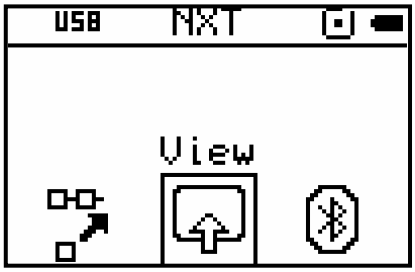
Reading the line light reading
Using the VIEW function on your robot to read the light sensor.

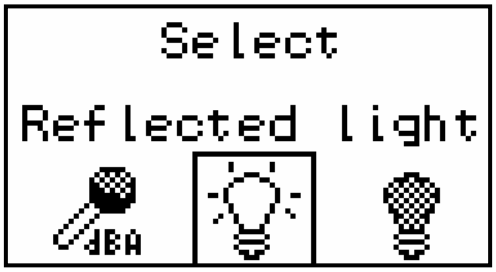
Place the robot on line and take 4 readings from the line, moving the robot each time.

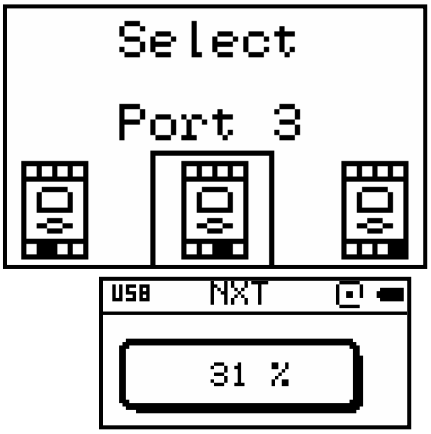
Move the robot around inside background area and take 4 readings from the background area.

Record your readings on your worksheet.

To VIEW the reflected light sensor:







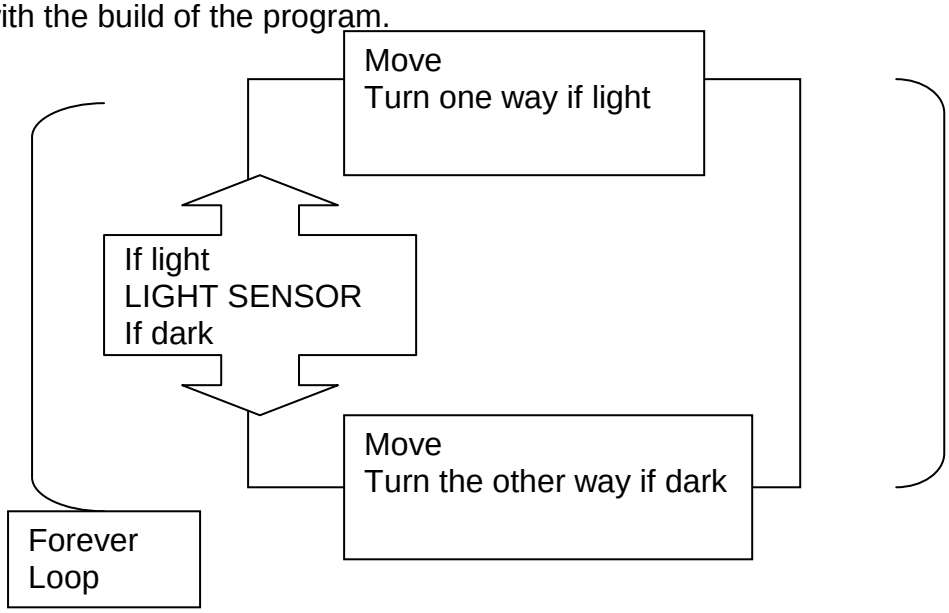
Record your light readings

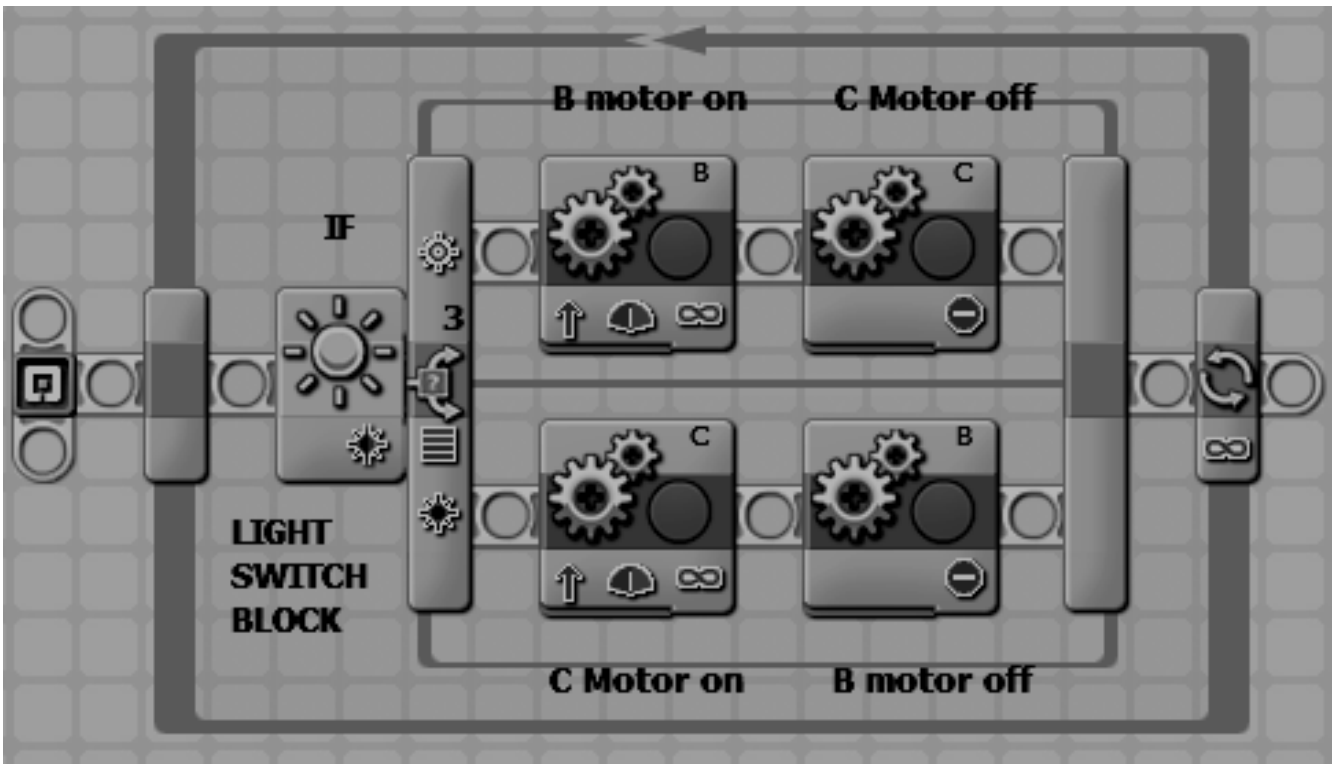
Line Reading	Background Reading

Programming

Use your light sensor threshold reading in your program to get your robot to recognise the line.

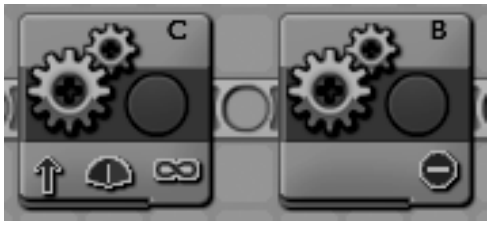
You will also need to use a FOREVER loop and a light sensor switch block. This flowchart should help you with the build of the program.





Programming Icons and settings

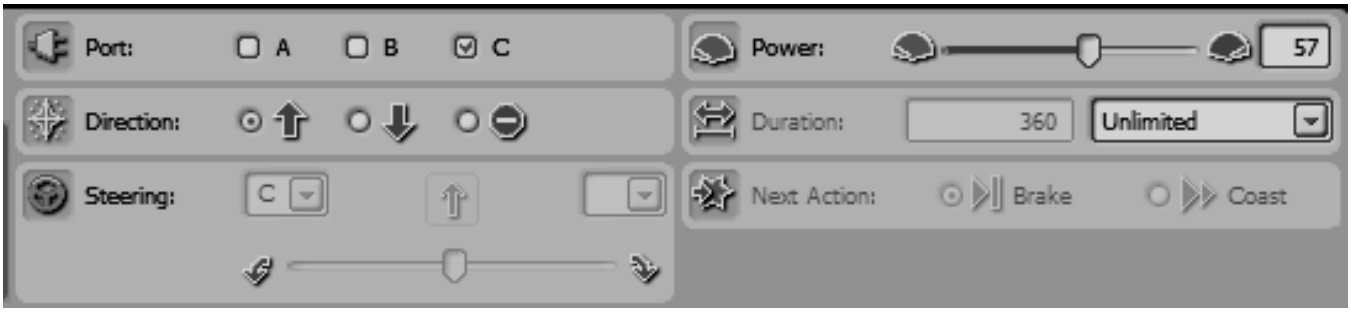
Controlling individual motors so that you can program the pivot turn is easy and can be done in two ways.

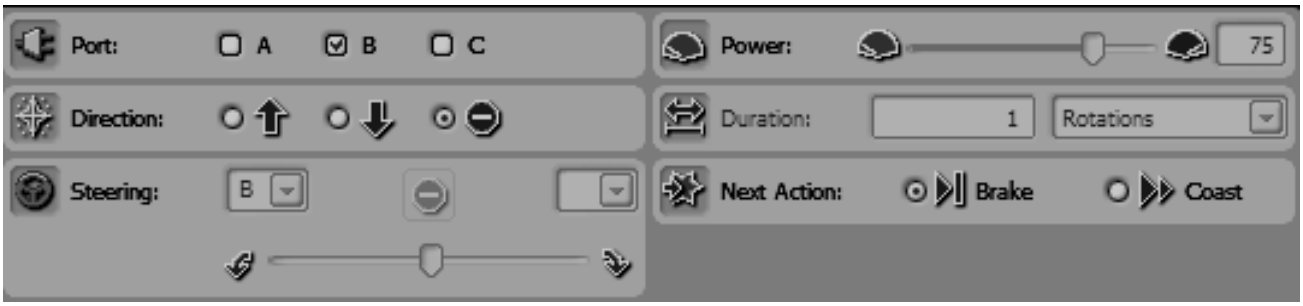


Using the common pallet:

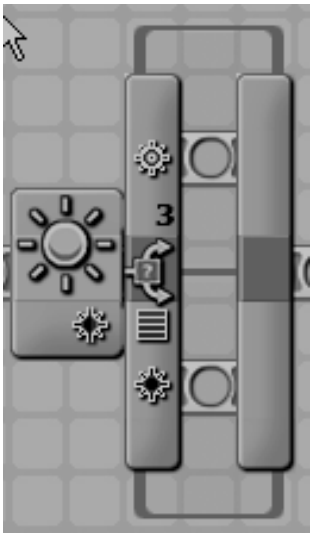
To control the robot's two wheel independently you can use two basic move icons and simply un-check one of the motors but to stop a motor the motor must be checked then stopped.

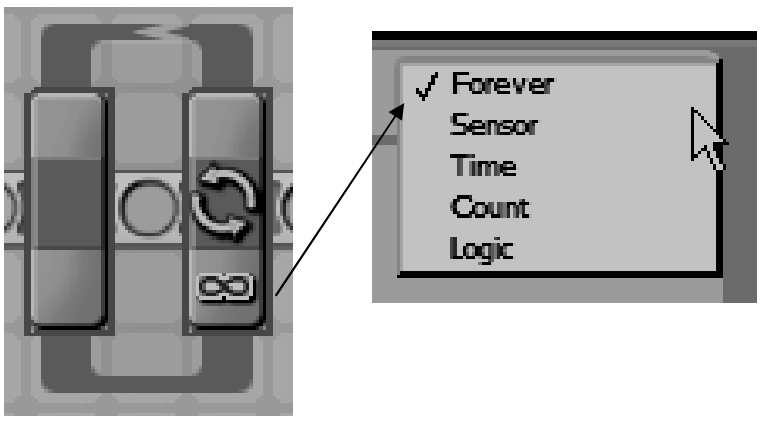
You must set the motor **duration to UNLIMITED** to force the motor to look for the light sensor and you must **slow the motor power down to <50** to help it keep up with the light sensor.





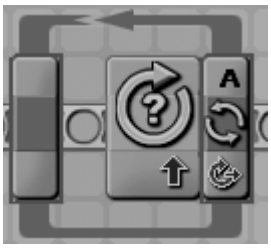
You will also need to use a loop and a light sensor switch block.

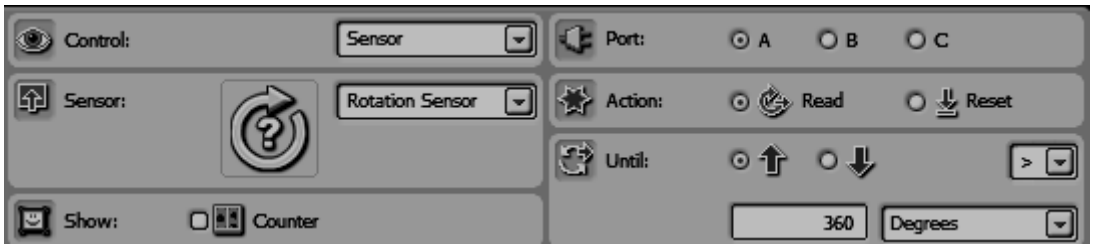




In First LEGO League the loop will be set to control how long the robot follows the line for. You have several options available in the LOOP configuration . Have a look at the SENSOR LOOP options, COUNT and TIME options.

In the sample below the programming inside this loop will only run until >360 degrees are reached then the program will move on to the next set of commands



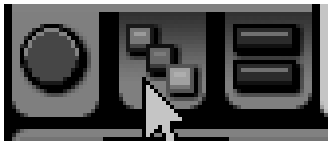


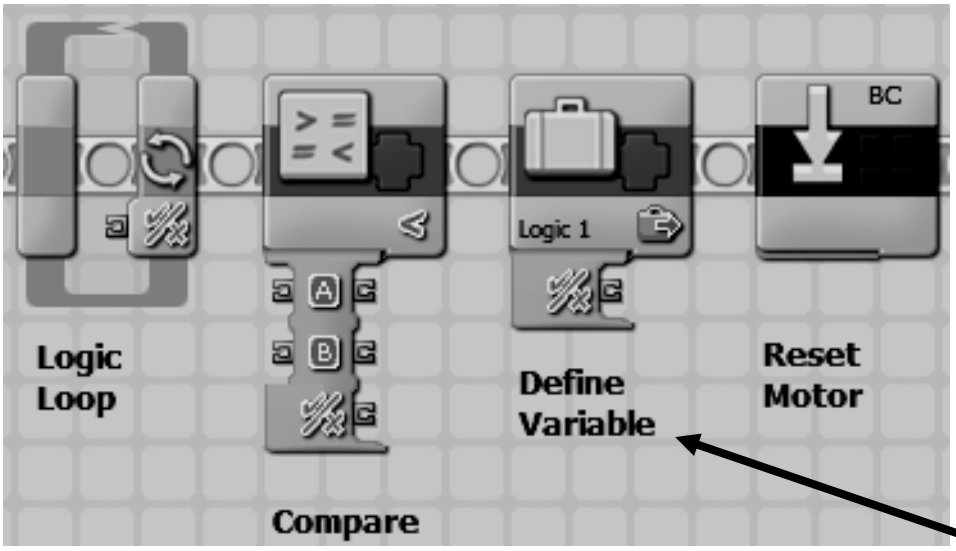
As this is a simple line follower it uses only one light sensor. The side of the line you place the robot on is influenced by the directions the robot turns for the two colours, test the program to see which side is best.

Modifications

Improvements to line following would include:

- Using 2 light sensors, one on each side.
- Using the COMPLETE pallet icons offering more options such as individual motor control, data hubs to dynamically share data and set variables such as where you can program the robot to actually take the light readings itself, mathematically select a threshold and then use that threshold to follow the line. (see next page)





See also Associated Documents on this website:

CIRCLE_ARC_DEGREES.pdf

Calculate degrees.xls

Following pages ; program samples

Automatic Line reader and follower : data hubs, math's blocks, variable

This program uses the right scroll buttons to initiate 2 light sensor readings.

Position the robot inside the base so that when it moves forward the black line leading to the grey loop (and other loops) is on the right of the light sensor. The robot moves forward approximately 2 cm (in degrees) To get it over the base's black line, it then stops and takes a light reading, subtracts 5 (to allow for a little variation) . It will then use that reading as "white" and compare this reading with what the light sensor is seeing as it moves. It will then wiggle along the line according to if it "sees' black or white. The ability of the programming to take and store sensor readings is an extremely useful action that can be used for other tasks such as degrees readings and overcomes the fact that often the weakest link is the accuracy of the human taking manual readings. This type of program also removes the necessity for re programming using a computer when the light conditions change, the robot itself is used to calibrate the readings before each "run".