An introduction to Robotics with LEGO® MINDSTORMS (VII)

A challenge with LEGO MINDSTORMS

Text and images by Koldo

In this article I will present a challenge that must be solved by LEGO® MINDSTORMS robot, programmed with NXT-G, and the necessary steps to reach one of the possible solutions. I say one of the possible solutions, because just like with any problem in engineering, there is always more than one possible solution, some more efficient than others.

Challenge

Inside a circular area, limited by a black line, there are two full 330cc soda cans. You will need to build and program a robot capable of taking the cans out of the circle, after which it will emit a sound and stop.

The available material is a single box of LEGO MINDSTORMS NXT 2.0 which, for reasons undisclosed, contains no other sensors than a single touch sensor and a colour sensor (a light sensor can be used in substitution of the colour sensor).

Where to start

Planning is fundamental if you want to get a good result. Thinking about and determining the steps to take must be the starting point. While doing so, you shouldn't forget that any challenge involving robots consists of two clearly differentiated, though closely related, parts: hardware (the robot itself) and software (the program). Each of these parts conditions the other so in our trial and error process we will have to take both into account.

The tasks that must be completed are the following

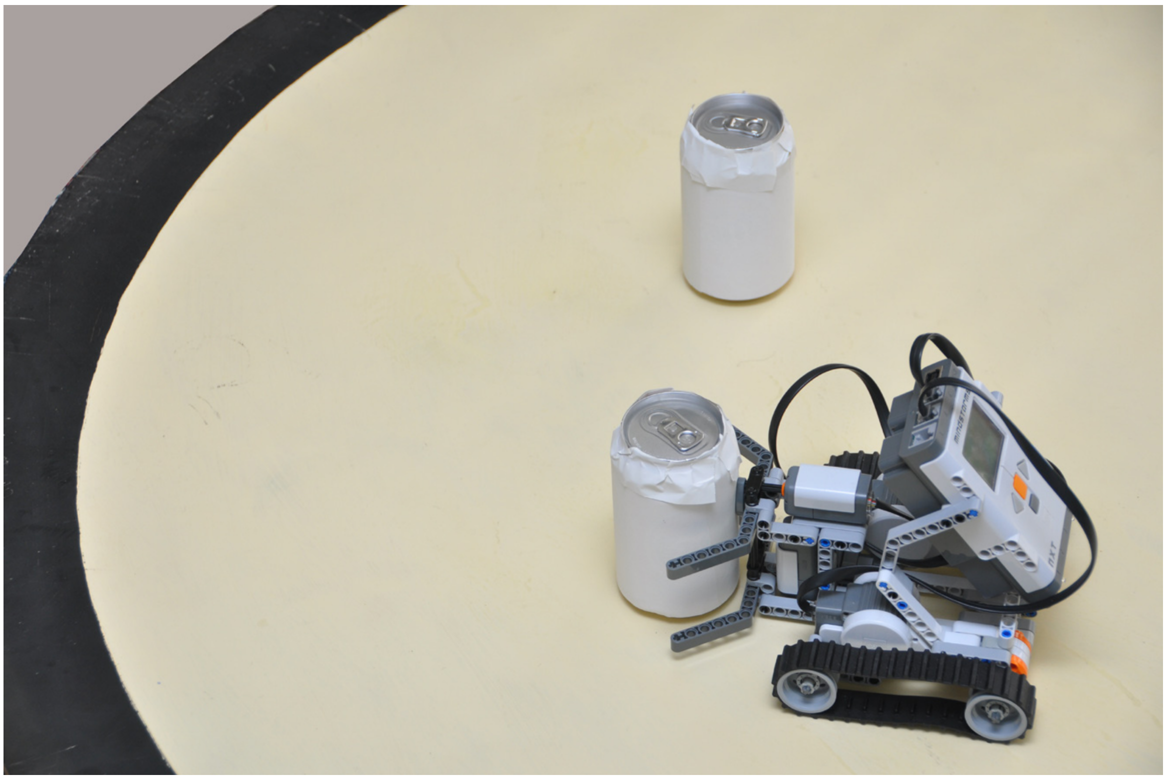
1. Designing and building the robot
2. Writing the program (in the case of NXT-G by combining programming blocks). In order to do this you will first have to write the algorithm and then convert that into the program.
3. Test it – and when the it is capable of carrying out the assigned task:
4. Improve it.

The robot

To start off, you need to keep in mind the restrictions (limitations in building and programming) of this challenge and the abilities the robot needs to have. These are as follows

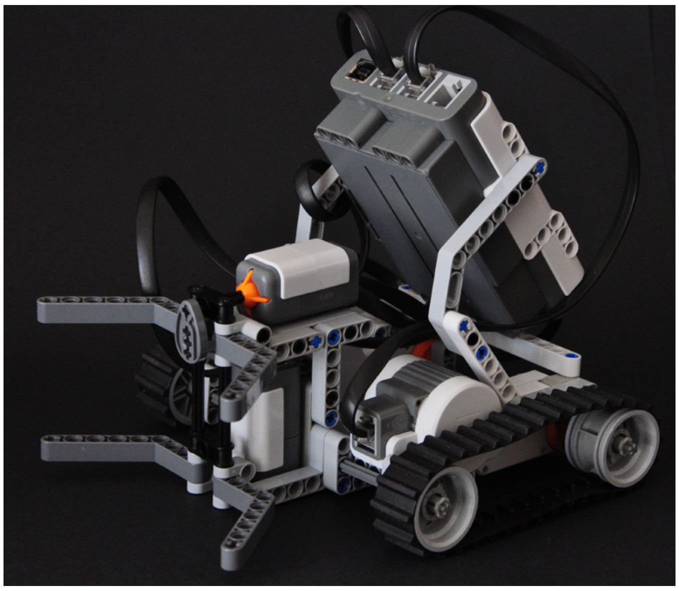
Restrictions

• The robot can only use a touch sensor and a colour/light sensor. There is no restriction to size or number of parts



Abilities

• The robot needs to be able to move and turn in the playing field.
• It should be able to detect the black line that limits this area and stay inside it.
• It needs to be able to detect when it bumps into one of the cans and move it out of the field
• It needs to know when it has taken both cans out of the field.
Once you have built the robot and you start testing it there



will likely be complications that will require modifications. For example, a complication I ran into is related to the colour sensor: while testing my first program I realized the colour sensor detected everything as black. After several tests, sighs and changes I realized the problem was that the sensor was too close to the floor and so there was no space for light to reflect into the sensor. Moving the sensor slightly up solved the problem.

The algorithm

Before you start to combine the code blocks that make up the program, it would be a good idea to write it down in normal language, that is, write down the steps the robot needs to take to accomplish its mission. There isn't one single solution for the task, just like people carry out the same tasks in different ways. So after writing it down, think about alternative ways to reach the same goal in a more efficient way.

In this case the programming tasks are as follows:

1. Create a counter and set it to zero (this will store the number of cans that have been taken out of the field)
2. Advance until the robot detects a black line or an object
    a. If it detects an object, keep advancing until it detects the black line (which means the object is now outside the playing field) and add one to the counter
3. Stop, back up and turn
4. If the counter has reached 2, play a sound and stop. Else repeat the search.

This is only a starting point that may need to be modified or made more specific as you advance in your programming.

The program

You now know which steps the robot needs to take to finish the challenge, but it is neither necessary nor recommendable to start the whole program at once. It is a very useful strategy to break down complex problems into easier ones. In this case you can split the program into two parts and combine those later

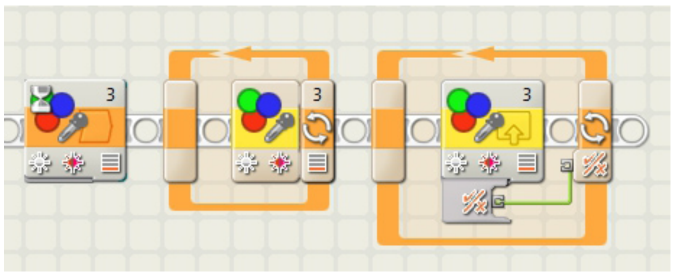
1. The robot moves randomly around the play area without leaving it.
2. The robot pushes the can out of the play area, backs up and plays a sound.

Challenge part 1

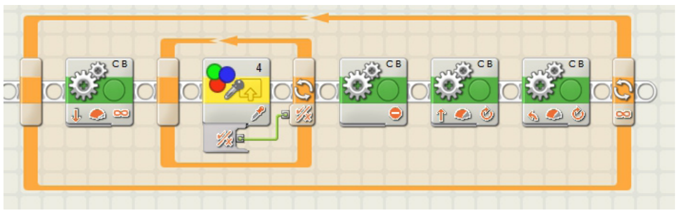
Let's have a look at the algorithm before writing the program:

1. Move forwards until the colour sensor reads black
2. Stop
3. Back up
4. Turn
5. Repeat steps 1-4

To tell the robot it needs to wait until the sensor reads black you can use any of the following three options. Each one will have exactly the same result, but the third option opens the way for combining the parts of the challenge.



The program would be as follows:

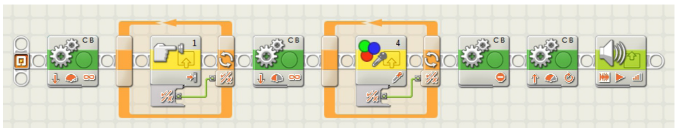


You will need to test it and make sure the behaviour is as expected. Before doing so, read Testing the robot at the end of this article

Challenge part 2

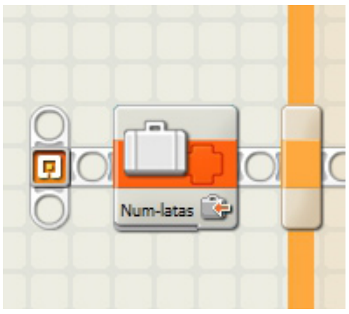
Let's have a look at the second part… To make matters easier, let's suppose the robot is on a collision course with one of the cans (so it doesn't need to look for a can) which requires an easier algorithm, for example like this:

1. Move forwards until you bump into a can
2. Keep moving forwards until you reach the black line at the edge of the play area
3. Stop
4. Back up
5. Play a sound

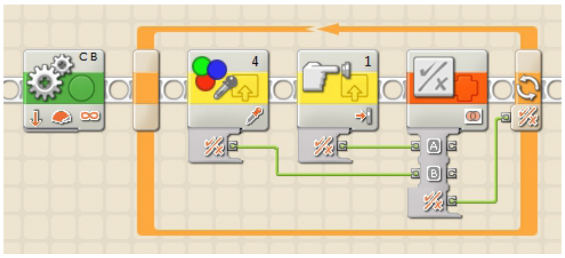


The complete program

If the two challenge parts work as expected it is time to combine them. We'll start creating a variable (represented by a block with a suitcase) in which we can store the number of cans that have been pushed out of the field. The variable is created with the option define variable in the Tools menu and is initialized with value 0 in this way:

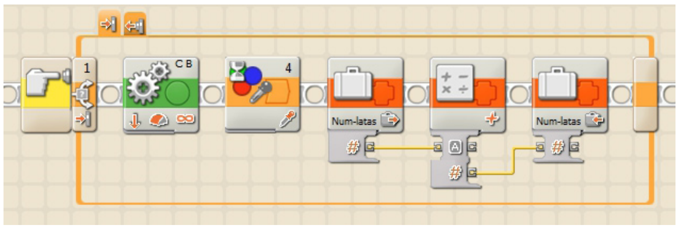


First let's see how you can monitor two sensors at the same time and take decisions based on those readings (step 2 of the algorithm).

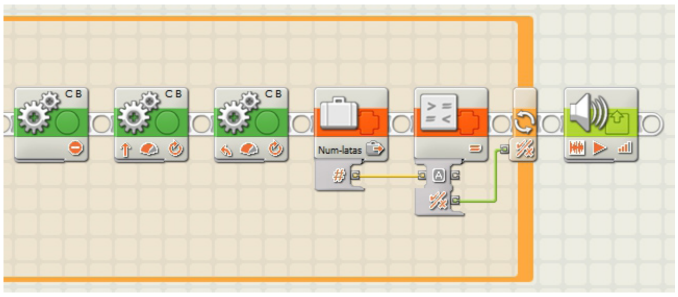


The code fragment in the image begins with a move block to set the robot in motion before starting a loop that is repeated until either the colour sensor reads black or the touch sensor is pressed (makes contact with the can). To this end the loop continuously reads the two sensor and by means of a logical operation (in this case OR), combines both readings, generating a logical value that is true if the can has been touched or the black line has been seen, or both readings are found simultaneously.

The robot needs to know if it has reached the border or has bumped into the can. For this purpose you can use a condition that will only be executed if the touch sensor has been activated. In this case it will push the can until it reaches the black line and then add one to the counter (step 3).



After stopping, backing up and turning (step 4) you need to check how many cans have been pushed out. So the program checks the value of the counter and if it is 2 ends the loop, after which it plays a sound and stops (step 5).



Testing the robot

Now we can test the robot and evaluate the results. If they don't match your expectations you'll need to evaluate if you need to change the robot or the program.

Recommendation! Don't apply too many changes at the same time as that makes it very hard to see which changes improve or deteriorates the behaviour of the robot.

How to continue

If you have come this far you can still make more of the challenge by making the following modifications:

1. After reaching the line the robot backs up and turns, but it always turns the same way. Modify the program to make each turn random (as if it throws the dice to decide how much to turn).
2. You have an ultrasound sensor so your robot doesn't need to move as if it were blind. You may be able to locate the cans without actually touching them.
3. Change the can for something heavier (a bigger can or something similar). This is an interesting exercise to get a feel of what a sumo competition is like with a dead weight.

Final remarks

You can find building instructions of the model in the pictures, the complete program and some more extensions to the challenge at http://lroboticas.net. You can also pose any questions you may have in the associated forum.
#

Lrobotikas.net
Robótica Educativa y Recreativa