An introduction to Robotics with LEGO® MINDSTORMS (VIII)

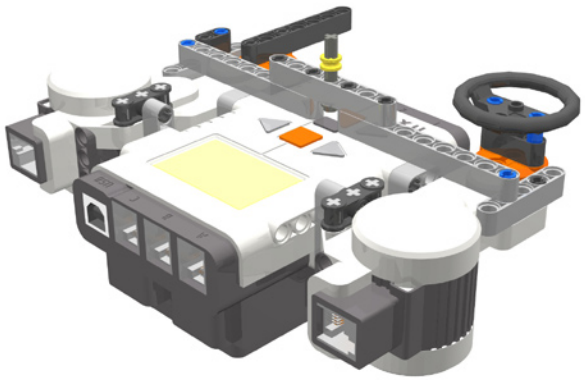
Remote Control with LEGO MINDSTORMS

Text &pictures by Koldo

This is the second article in a series of challenges to be solved using LEGO® MINDSTORMS. It will be extended by means of a longer and more detailed manual which will be available on Lroboticas.net where you can already find the complete version of the challenge published in the previous edition of HBM, and which includes building instructions.

Although we normally consider a robot should be capable of carrying out a task autonomously, at times it may be necessary to control it remotely. There are different communication systems to accomplish this: the classical radio control of the remote controlled cars, infra-red control an even over the Internet.

The following challenge will use Bluetooth communications, which requires two NXTs to carry it out.



Challenge

An interesting challenge is to build and program a remote control that allows you to remote control a robot in the same way you would a remote controlled car, that is to say, with the following options:

· Turning left or right
· Moving forwards or backwards
· Regulating speed

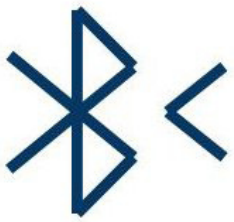
Next, we will see how to build a remote control capable of controlling turning on a mobile robot. The rest is an option for those who want to go a step further.

Bluetooth Communications

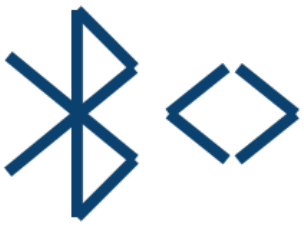
The NXT is equipped with a remote communications system by Bluetooth. The NXT can send and receive text, numerical and logical messages and show a different behaviour depending on the message received.
Up to 4 NXTs can be connected at one time, allowing them to carry out their tasks in a coordinated way. But that does not mean they can communicate freely among all four, there will always be one master who is in command, while the rest are slaves, and can only communicate with each other through the master.

First steps

For two NXTs to be able to communicate they first need to be connected. To do so we need to start by enabling Bluetooth. How do we know if it is enabled? Simply look at the top left part of the NXT screen and if you see the following symbol Bluetooth is activated.



If it is not, we need to use the grey arrows and orange button of the NXT.
After deciding which of the two will be the master we can connect the two NXTs. To do so, we need to look for the slave NXT from the master NXT and connect it to one of the available positions: 1, 2 or 3. Once they are connected, the symbol will look like this.



From now on the two NXTs can communicate with each other. NXT-G 2 allows the NXT to carry out this Bluetooth connection task itself.

Bluetooth Exercises

Before we start with a real project with Bluetooth communication, let's experiment a little. To this end we do not need to build anything, we simply need two NXTs and we can use the buttons as inputs and the screen and speaker as outputs. Below are a number of exercises, two of which will be solved in this article.

1. Create a program that sends a random numerical value between 200 and 400 so the second NXT can reproduce sound in that range.
2. NXT 1 controls the frequency of the sound NXT 2 emits. Pushing the right arrow increases the frequency at intervals of 10 and the left arrow lowers it. NXT 2 emits the sounds with a length of half a second and half second intervals of silence.
3. The same exercise as before, but the orange button switches between raising and lowering the pitch and making the interval longer or shorter in 0.1 second intervals.
4. The screen of one NXT shows the temperature a second NXT registers outside. If you do not have a temperature sensor, you can use a light sensor to register luminosity.

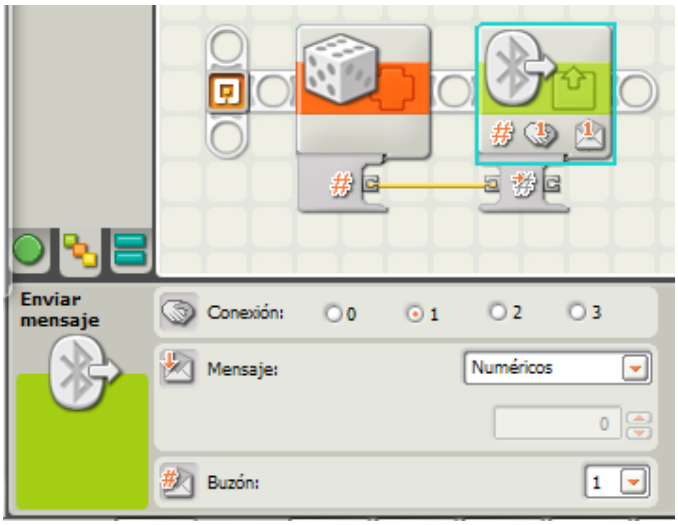
Exercise 1

First, let's have a look at the program for sending the message (NXT master).

The algorithm will be as follows:

1. Choose a random number between 200 and 400
2. Send a message that contains this number

The corresponding program will be as follows:

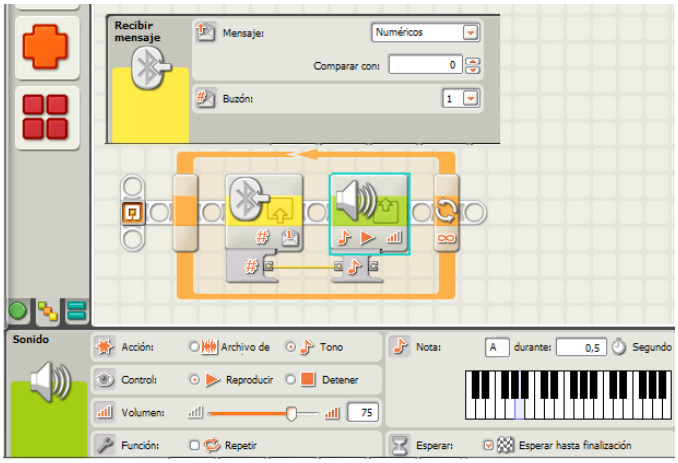


After obtaining a random number between 200 and 400 we will use the **send message** block from the **Action** menu. The message type needs to be adjusted to select a numeric value. You don't need to modify anything else.. The connection number needs to be the same one assigned to the **slave** NXT, in order to identify the recipient. In case the slave needs to send a message to the **master** it has to send it to **Connection 0**.

And now for the receiving NXT (**the slave**)

The receiving NXT must be listening for the message. Let's have a look at the algorithm:

1. Reed inbox 1
2. Play a sound with the frequency indicated in the message
3. Repeat indefinitely



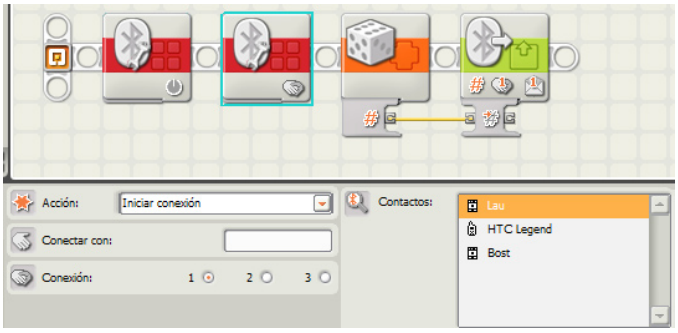
Once both programs have been created we only need to connect both NXTs by Bluetooth, download the programs and run them (for this exercise you need to first start the slave program and then the master).

Improvements

If you have NXT-G 2 it is possible to integrate the Bluetooth connection in the program. The following program adds two new blocks to the previous master program. These are located under Advanced: the Bluetooth Connection block, configured in two different ways. The first one switches on Bluetooth (this block can also be included in the slave program), while the second establishes a connection.

To configure this connection two conditions must be met:

· Both NXTs must have been connected previously so the slave is in the Contacts list of the master
· The master NXT must be on and connected to the PC so the list of contacts appears in the configuration panel as can be seen in the following image.



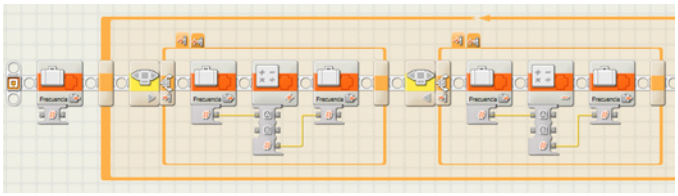
Exercise 2

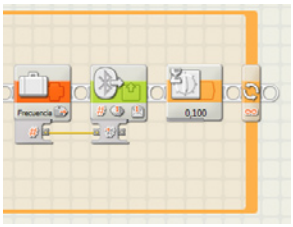
Let's see how we resolve the second exercise as it is very similar to the first one.

First, let's have a look at the algorithm of the program for controlling the sound (master NXT).

1. Create a variable to store the value of the frequency
2. Assign a value of 200 to the variable
3. Continuously repeat the following steps
    a. If the right arrow is pressed, add 10 to the variable Frequency, if not do nothing.
    b. If the left arrow is pressed, subtract 10 from the variable Frequency, if not do nothing.
    c. Send the value of Frequency by Bluetooth
    d. Add a wait of tenth of a second at the end of the loop to limit the effect of a continuously pressed arrow.

The program will be as follows:





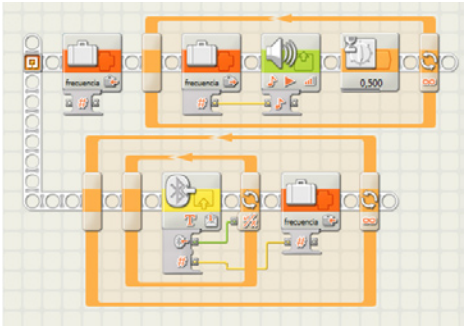
The variable Frequency is created through the option Declare Variables in the Edit menu. In both conditions the False option is left blank.

The program for the slave NXT , the one that reproduces the sound, will have two tasks which are executed simultaneously. The corresponding algorithm is the following:

1. Create a variable to store the value of the frequency
2. Assign a value of 200 to the variable
3. Repeat the following steps
    a. Read the value of the variable Frequency
    b. Play a tone with the frequency that was read during half a second
    c. Wait half a second (silence)

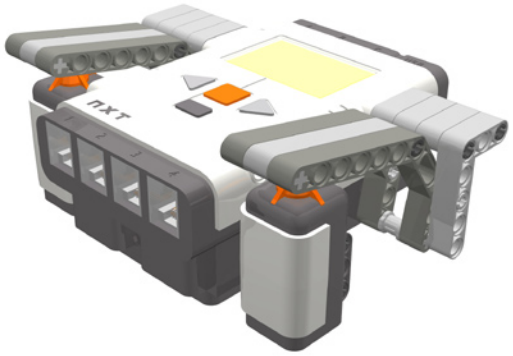
The second task is the one that receives the message and assigns its value to the variable Frequency. The algorithm will be as follows:

1. Repeat the following steps
    a. Wait until you receive a message
    b. Assign the value of the message to the variable Frequency.



The remote

The remote will have two levers that will help control the touch sensors under it.



In fact, for such a simple remote as this you do not really need to build anything, as you could simply use the buttons on the NXT itself. The robot we will control is the one from the challenge in the previous edition of Hispabrick Magazine or any other one with a similar motor configuration.

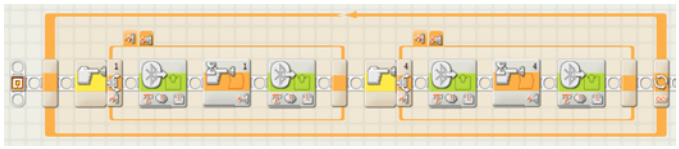
The algorithm for the remote

The remote uses two touch sensors to control the vehicle, the left one connected to port 1 and the right one to port 4. Pressing the left sensor the robot will turn left until you stop pressing and the right one will do the same in the opposite direction.

The algorithm for the remote will be as follows:

1. Repeat the following steps
    a. If Sensor 4 is pressed
        i. Send text message "Right"
        ii. Wait until sensor 4 is no longer pressed
        iii. Send text message "Straight"
    b. If Sensor 1 is pressed
        i. Send text message "Left"
        ii. Wait until sensor 1 is no longer pressed
        iii. Send text message "Straight"

The program will be as follows:



The algorithm for the vehicle will be very similar to the one in exercise 2. It will consist of two tasks, one to control the vehicle and another for receiving and storing the messages.

The first algorithm will be:

1. Create a variable to store the value of the Steering
2. Assign the value "Straight" to the variable Steering so by default the vehicle will go straight.
3. Repeat the following steps
    a. Read the value of the variable Steering
    b. If the value is Straight move forward endlessly
    c. If the value is Right turn right

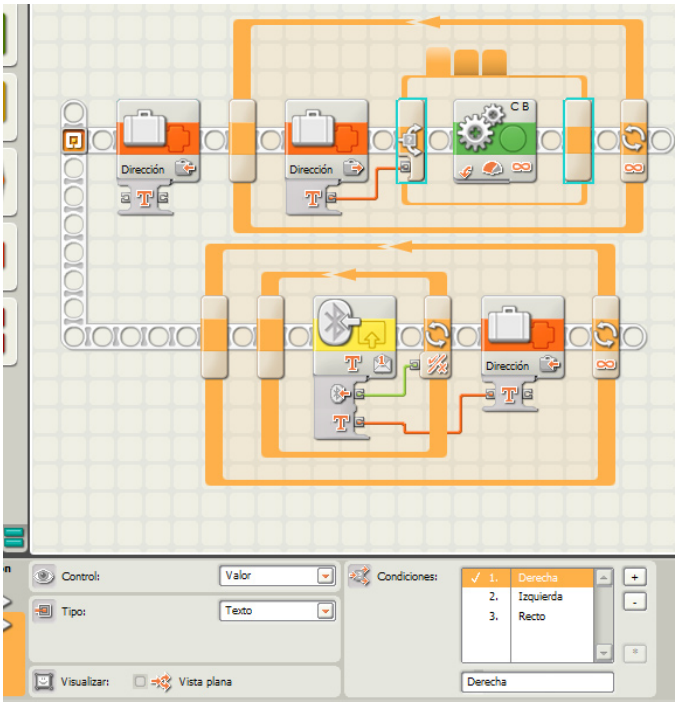
d. If the value is Left turn left

For this last step you can use a switch block to execute one of the operations depending of the value of the variable. In order to be able to do that, you need to uncheck Flat View check box, or else you will only be able to use two options.

For the second task it will be the same as in exercise 2.

1. Repeat the following steps
    a. Wait until you receive a message
    b. Assign the value of the message to the variable Steering.

The program will be like in the following picture:



Improving the program

What can be done to improve the program?

1. Add a function to use the orange button to stop and start the vehicle.
2. A motor can be a very interesting complement to control the turning speed. If you connect a lever to a motor, the turning angle can be used to send a power level to the vehicle.
3. A motor can also be used for steering. Using s steering wheel connected to a motor, the amount of left or right turning can be transformed in the turning speed and direction of the robot.

#