LEGO® MINDSTORMS® NXT Lab 3

This lab session is an introduction to using the touch sensor and ultrasonic distance sensor of the LEGO MINDSTORMS NXT. The first two parts of this exercise will introduce the use of input function blocks to control the Tribot via touch and ultrasonic sensors. You will then follow instructions to complete code that will make the LEGO MINDSTORMS NXT avoid obstacles using both the touch sensor and the ultrasonic distance sensor. You will then create your own programs that will perform a task utilizing what you have learned during the step by step guidelines.

Lab Summary

A. Use the LabVIEW Toolkit for LEGO MINDSTORMS NXT to monitor the touch sensor, the ultrasonic distance sensor, and the sound sensor.
B. Create a program to enable the Tribot to avoid obstacles (i.e., cylinders) in its path.
C. (Challenge) Have the Tribot detect the obstacle, approach the obstacle, and stop in front of the obstacle.

Timeline

It is expected that this lab will take 2 class periods. A/B can likely be completed in one period. Part C should take 1 class period.

Software

- LabVIEW with LEGO MINDSTORMS NXT toolkit

Hardware

- LEGO MINDSTORMS NXT Tribot with the ultrasonic sensor, sound sensor, and touch sensor attached. The ultrasonic sensor should be attached as indicated on page 28 of the guide. The sound sensor should be attaches as on page 18. The touch sensor should be attached in similar manner as the light sensor is attached on page 36 of the guide. You may also want to consider having a 'bumper' attached to the touch sensor to increase the contact area.

Part A: Monitoring the Touch Sensor and Ultrasonic Sensor

1. Open LabVIEW on your computer.
2. Open the rotation sensor VI (**Lab 3 Part A.vi**).
3. Power on the NXT.
4. Using a USB Cable, connect the Tribot to a USB port on your computer
5. Run the VI.
6. Press the Touch Sensor on the Tribot and notice that the state of the sensor changes on the VI front panel.
7. Observe the ultrasonic sensor output. Notice that the sensor observation window values change when you move objects in front of the sensor.

Questions:

i) How far in do you have to press the touch sensor button for it to register on the display?

ii) What is the maximum distance that the ultrasonic sensor measures?

iii) What is the maximum viewing angle of the sensor?

iv) Are there any objects that the ultrasonic sensor cannot detect?
(You may wish to test any objects in question.)

Part B: Detect and Avoid Obstacles

1. Create a new VI.
2. Save the VI as *Lab 3 Part B.vi.*
3. Use the **Ctrl + T** shortcut to tile the front panel and block diagram.

4. Place a **While Loop** on the block diagram.
    a. Select **Structures >> While Loop** from the **NXT Programming** functions palette.
    b. Click and drag the loop to place it on the block diagram.

5. Place a **Read Ultrasound** function inside the while loop.

    a. Select **NXT I/O >> Read Sensor** from the functions palette.
    b. Place the function block inside the while loop.
    c. Select **Read Ultrasonic** in the Polymorphic VI Selector.

6. Place a **Greater?** function block beside the **Read Ultrasound** function block.

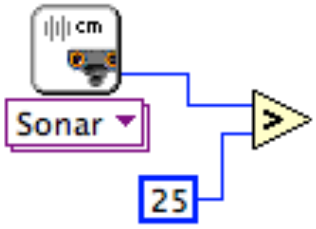
    a. Select **Comparison >> Greater?** from the functions palette.
    b. Place the function block next to the **Read Ultrasound** block.

The **Greater?** function may attempt to wire itself to the **Read Ultrasound** function block automatically when you place it. If it wires itself to an incorrect terminal, simply select the wire and delete it.

c. Wire the **Distance (cm)** terminal of the **Read Ultrasound** function to the **X** terminal of the **Greater?** function.

7. Create a constant of value **25** attached to the **Y** input of the **Greater?** function block.

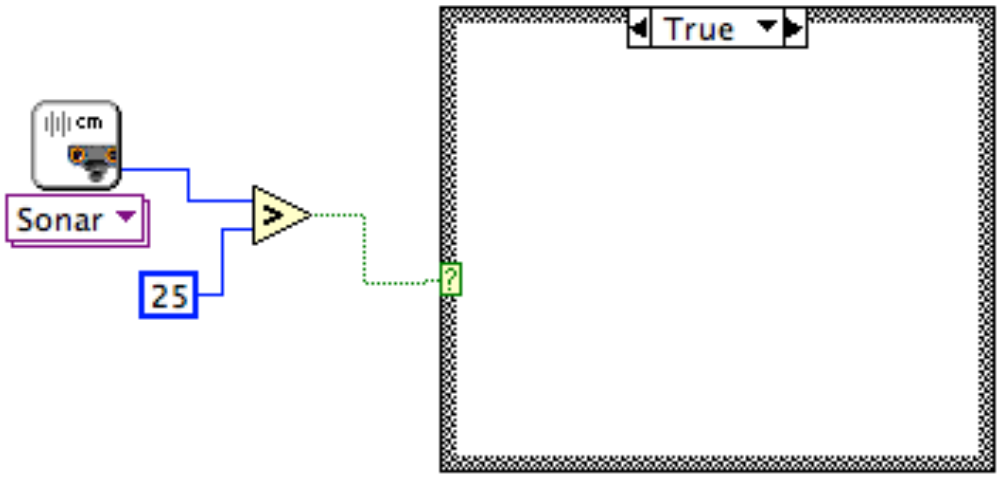
The result of steps 5-7 should resemble the following:



Inside the while loop, the **Greater?** function will continuously compare the distance reading from the ultrasonic sensor with the constant. If the ultrasonic reading is greater than 25 cm, the **Greater?** function will output a **True** value. If the reading is less than 25 cm, the output value will be **False**. To use this output to control the behavior of the Tribot, you will now use a **Case Structure**.

A case structure contains multiple sub diagrams, one of which is executed each time the structure is executed. Which case is executed is determined by the case selector. Case structures can contain many cases, but in this instance, there are only two: **True** and **False**. The false case will execute when the Tribot senses an object within 25 cm of the ultrasonic sensor. Otherwise, the true case will be executed.

8. Create a **Case Structure** inside the while loop.



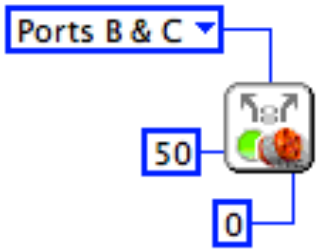
a. Select **Structures >> Case Structure**.
b. Inside the while loop, click and drag in the whitespace to create a case structure.

c. Wire the condition terminal of the case structure to the **X>Y?** terminal of the **Greater?** function.

Ensure you are in the **True** case. The label at the top of the case structure will display the current case. To switch cases, click on the left or right arrow beside the case label. Alternatively, you can right click on the case label and select **Show Case True**.

9. Inside the **True** case, create a function to move the Tribot forward.

   a. Place a **Steering On** function in the case structure. Set the power to **50**, the steering to **0**, and output ports to **Ports B & C**. This presumes that your left and right motors are connected to Ports B and C, respectively.

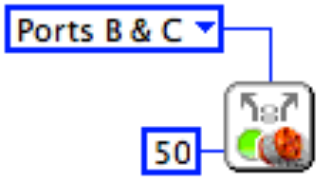


10. Inside the **False** case, create a function to command the Tribot to turn to avoid the obstacle.

    a. Copy the **Steering On** function from the true case and place it in the false case.

To copy code to the clipboard, simply select everything you wish to copy and use the **Ctrl + C** shortcut. When you wish to paste the copied code, left click once where you wish to place the code (the code will be placed with its center at the last click location). Use the **Ctrl + V** shortcut to paste the code.

   b. Use the default steering constant of **100**.

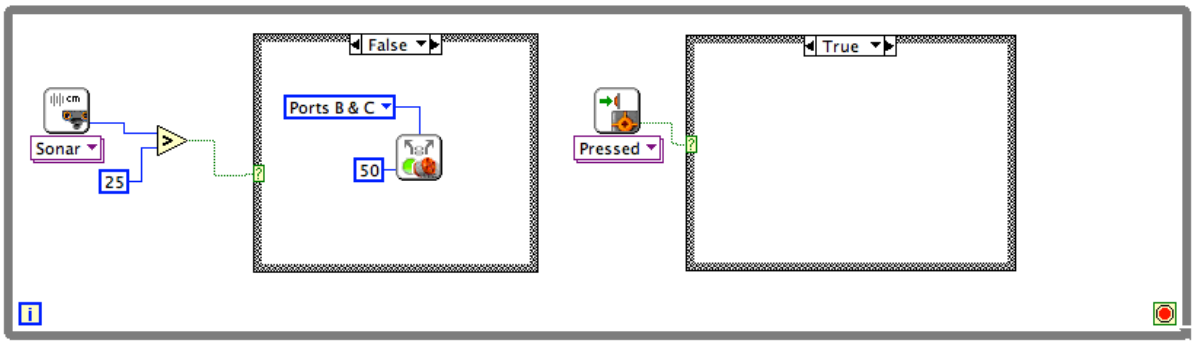


11. Place a **Read Touch (Pressed)** function block inside the while loop to the right of the existing case structure.

    a. Select **NXT I/O >> Read Sensor**
    b. Select **Read Touch >> Pressed** in the **Polymorphic VI Selector.**

12. Place a new case structure to the right of the **Read Touch (Pressed)** function.

Below is a view of the program layout for your reference. You are currently placing the second case structure.

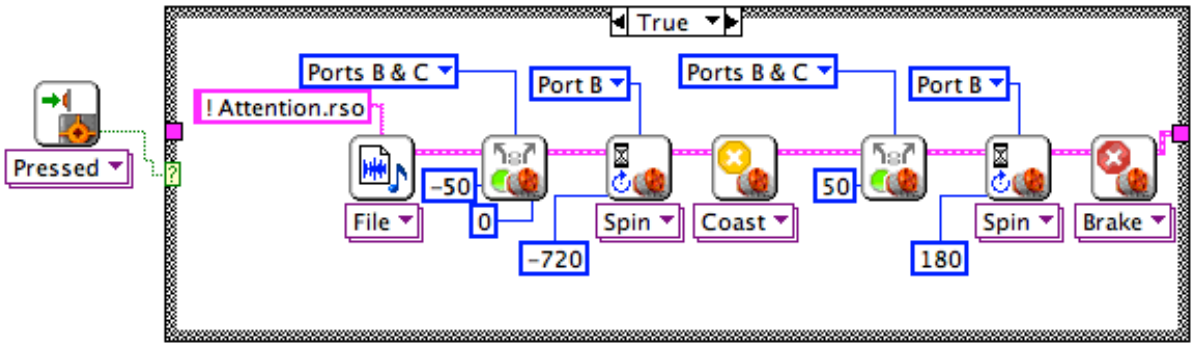


13. Wire the **Yes/No** terminal of the touch sensor to the case selector of the new case structure.
14. Create a **Play Sound File** function block inside the **True** case.

    a. Select **NXT I/O >> Sound Control**.
    b. Select **Play Sound File** in the Polymorphic VI Selector.
    c. Create a constant on the **Filename (*.rso)** terminal with value **! Attention.rso**.

    Note: ! Attention.rso is a system sound file that comes pre-loaded on the NXT. There is no need to copy any files to the device. If, for some reason, the sound file does not exist on the NXT, the code will still run, but the unit will not play a sound. If your NXT does not have this sound file, updating the firmware will reinstall it. Alternatively, you can have your Tribot play a different sound.

    Create a function to move the Tribot away from any obstacles that trigger the bumper.



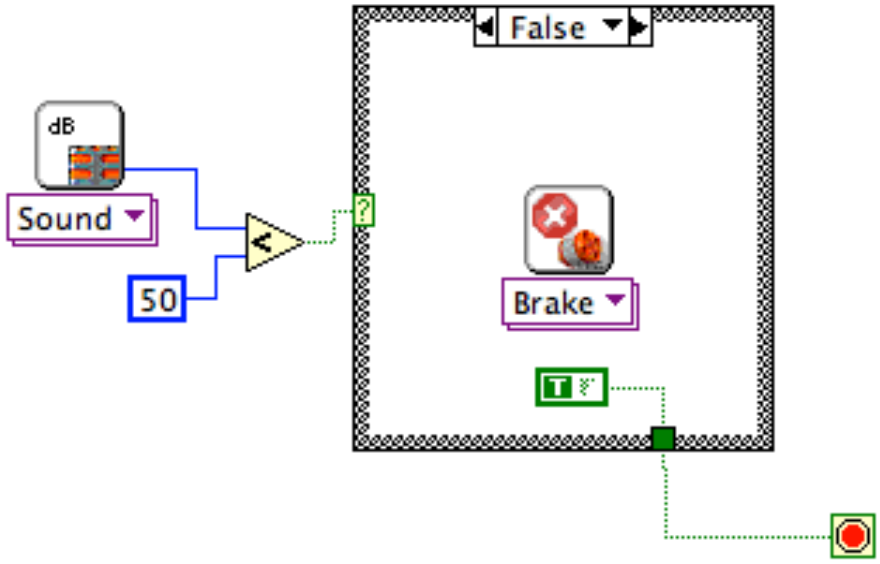
15. Place a **Steering On** function and a **Wait For** function next to the sound file block inside the true case.
    a. Set the steering input of the **Steering On** function to **0**, set the power to **-50**, and the output ports to **Ports B & C**
    b. Set the **Wait For** function to **Wait for Rotation** set the block such that the motors rotate **720** degrees backwards.
    c. Add a **Motor Control** function block next and choose **Motor Off >> Coast**.

    By default, in most cases motors will execute a **Brake** action when they finish executing movement commands. This causes the motors to actively and immediately stop rotating. The **Coast** command simply causes the motors to shut off and coast to a stop. Using the coast command is recommended for executing multiple motor movements in the same direction, allowing the Tribot to move smoothly between movement commands. In this case, the

command allows the Tribot to execute its turn smoothly after reversing away from the obstacle.

16. Place another set of **Steering On**, **Wait For Rotation**, and **Motor Brake** function blocks inside the true case.
    a. Configure the second set of function blocks to rotate the motors **180** degrees with a steering constant of **100** and then **brake**.

    Create a function to shut down the Tribot if the ambient noise level exceeds a certain threshold.



17. Place a **Read Sensor** function block inside the while loop. It should be placed to the right of the second case structure. Refer to the figure above step 13 for visual reference.
    a. Select **NXT I/O >> Read Sensor** from the functions palette.
    b. Choose **Read Sound >> dB** from the Polymorphic VI Selector.

18. Place a **Less?** function block beside the sound sensor block.
19. Wire the **Sound (dB)** terminal of the sound sensor function to the **X** terminal of the **Less?** function.
20. Create a constant of value 50 on the **Y** terminal of the Less? function.
21. Create another case structure within the while loop to the right of the **Read Sound** function.
22. Wire the **X < Y?** terminal of the **Less?** function to the case selector.
23. Right click the **Loop Condition** terminal of the while loop and create a **False** constant.

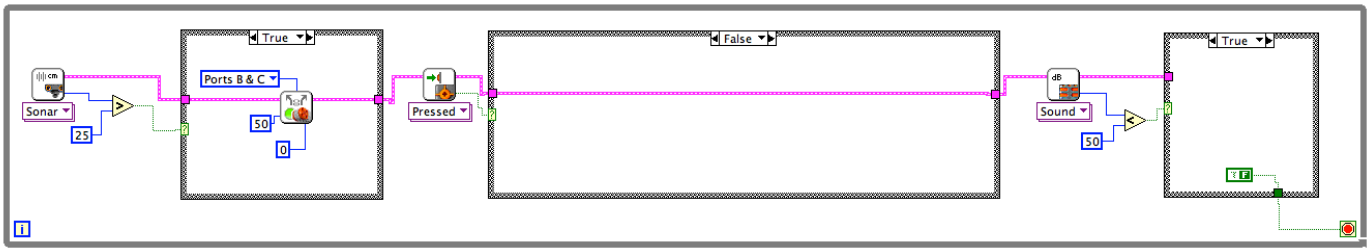
24. Drag the constant inside the **True** case of the case structure and reconnect the wire across the case structure boundary.
25. Switch to the false case and create a **True** constant on the terminal leading to the loop condition.

    Note: When wires pass in or out of a case structure, they use connector tunnels on the border. On a case structure, output tunnels must be connected to a wire in every single case. If the tunnel is correctly wired, it will be a solid color. If it is not wired in one or more cases, it will turn white with a colored border and the VI will not run (the run button will display a broken arrow). Make sure to wire output tunnels in every single case.

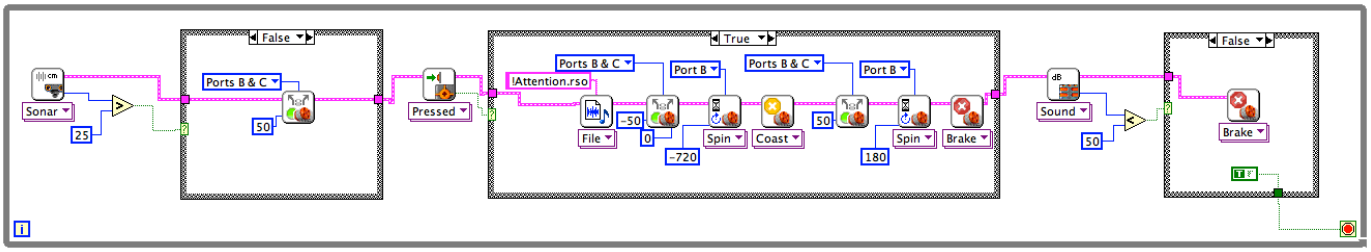
Each time the loop executes, it will poll the microphone input level and compare it with the constant you specified. As long as the ambient noise is below 50%, the true case will be executed and the loop will read a **False** constant on the stop condition and repeat. If the level rises above 50%, the false case will execute and the loop will see a **True** constant connected to the stop condition. This will cause the loop to exit, ending the program.

26. In the false case, place a **Motor Brake** function that will stop all motors before the program ends.
27. Wire all of the NXT blocks together using the **NXT** terminals.

The final program should look like the following:



Normal Cases



Exception Cases

28. Save your work and download the program to the NXT.
29. Run the program.

The Tribot should move forward in a straight line until the Ultrasonic sensor detects an object within 50 cm. When it detects the object, it should turn until the object is no longer detected, and then move forward again.
If the Tribot happens to run into an object that is too short to be detected by the ultrasonic sensor, then the object should depress the touch sensor. This would cause the Tribot to back up, turn left, and resume forward movement. The Tribot should also beep when the touch sensor is depressed.
If the Tribot detects a loud noise, it should stop, reverse direction for a moment, and stop running.

Part C (Challenge): Approach the object and stop just in front of it.

The objective of this challenge is to build on Part B. This time instead of avoiding objects detected by the touch sensor, have the Tribot detect the object, approach it, and stop with its touch sensor pressed against the object.

Good luck!