

LEGO® MINDSTORMS® NXT Lab 1

This lab session is an introduction to the use of motors and rotation sensors for the LEGO MINDSTORMS NXT. The first few parts of this exercise will introduce the use of the movement function blocks to perform simple actions. The final part of the lab is a task to move the robot along a specified path.

Lab Summary

- A. Use the LabVIEW NXT Module to monitor the rotation sensors in the motors
- B. Create a program to move the robot forward, backward, left, and right
- C. Use what you have learned to create your own program that makes the robot move in a specified pattern
- D. Take repeated data to determine robot variability.

Software

- LabVIEW with LEGO MINDSTORMS NXT module

Hardware

- LEGO MINDSTORMS NXT Tribot

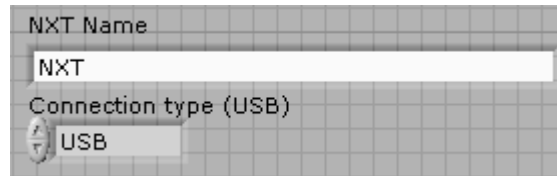
Part A: Monitoring the Rotational Sensors of the Motor

1. Open LabVIEW on your computer.
2. Select **File >> Open** to open the rotation sensor VI.
3. Open **Lab 1 Part A.VI**

The code contained in this VI is fairly complex, and is only intended to display data for the user. Working knowledge of the program is not required for its use.

4. Press the **Enter** button (orange square) on the MINDSTORMS NXT to turn it on.
5. Using a USB Cable, connect the NXT to a USB port on your computer.
6. In the VI front panel, enter the name of the NXT you will be controlling and select the **USB** connection type.

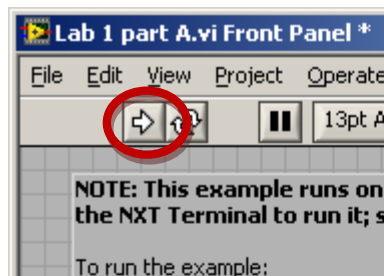
If you are unsure of the NXT name, simply use the default **NXT** value.



7. Make sure that your VI is targeted to run on your computer rather than the NXT. The bottom left hand corner of your front panel should read **Main Application Instance**. If not, then select **File >> Target to Computer**.
8. Run the VI by clicking the **run** command.

The VI will automatically detect the NXT if it is connected.

If the NXT cannot be found, the program will return an error. If this is the case, ensure that the USB cable is properly connected, the NXT is powered on, and the correct NXT name is specified. If the NXT still cannot be detected, ensure that the NXT drivers are properly installed.




9. Watch the chart as you turn the wheels on the Tribot.

There are rotational sensors installed for each motor on the Tribot called encoders. Encoders are devices that send a pulse to the NXT block on the Tribot whenever they detect a small change in the rotation of the wheel. They can also detect the direction of movement.

The VI reads the values from the encoders and translates that value to an angle which it then displays on the chart.

10. Press the **STOP** button when finished.



Press the **STOP** button in the VI rather than the stop  icon on the toolbar. The button inside the VI allows it to finish executing and close properly, while the toolbar control aborts the VI without allowing it to finish executing.

Questions:

(These are important questions to consider and answer, but there's no requirement that you submit your answers.)

- a) What happens when the wheels are rotated backwards?
- b) Approximately how many "ticks" equal one whole revolution of the wheel? (What numerical value does the rotation sensor display on the graph after one revolution?)
- c) Does the number reset to zero when a whole rotation is complete?

Part B: Command the Tribot to Move

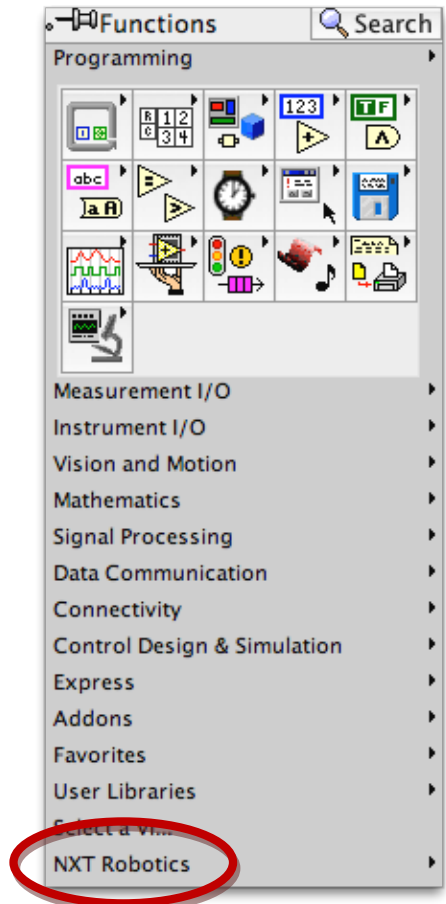
In this part of the exercise, you will program the Tribot to move forward, backward, left, and right. You will also learn some of the basics of LabVIEW in programming and controlling the NXT.

Create a new VI:

1. Open LabVIEW.
2. Select **File >> New NXT VI** from the menu bar at the top of the window.
3. Press the **Ctrl + T** shortcut to tile the front panel and block diagram on the screen.

Program the Tribot:

1. If it is not already active, switch to the block diagram window by using the **Ctrl + E** shortcut.
2. Close any open windows within LabVIEW other than the front panel or block diagram (palette windows, help windows, etc.)
3. Right-click (Ctrl-Click on a Macintosh) in the white space of the block diagram to activate the functions palette. Select **NXT Robotics**.
 - a. If the **NXT Robotics** menu is not visible on the functions palette:
 - Select the pushpin icon at the top left of the functions palette.
 - Select **View >> Change Visible Categories**.
 - Ensure that the box next to **NXT Robotics** is checked
 - Select **OK**.



4. Pin the **NXT Robotics** window by selecting the thumbtack icon.
 - a. Pinning the window allows it to be accessed without navigating the submenus of the functions palette that you just navigated through.



- b. The **NXT Robotics** window provides a complete library of all programming functions supported by the NXT.

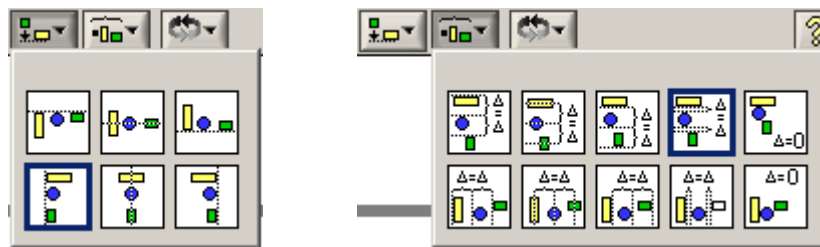
For more information on how to use functions, you can refer to the context help at any time. Press **Ctrl + H** to toggle the context help. Context help contains all labeled inputs and outputs of each function block as well as a description of its function. The context help window will display information about any function that the mouse is placed over, either in the functions palette or the block diagram.

We will now begin placing functions on the block diagram. Before beginning, it is important to understand some good programming practices.

Functions are almost always placed from left to right in sequence. Wires from inputs and outputs should always travel up, down, and/or right. Running connection wires backwards (from right to left) makes diagrams confusing and should be avoided.

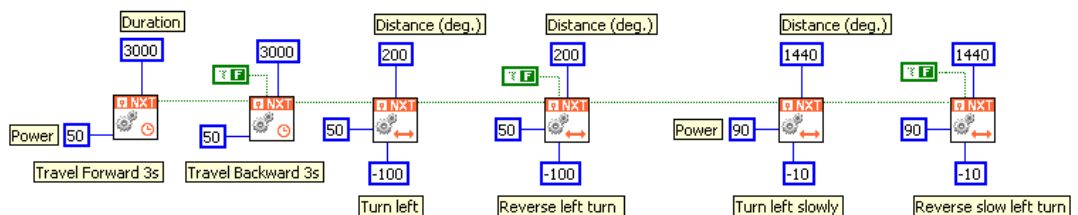
Wires should have as few bends as possible. The cleaner and straighter your wires are, the easier your diagram is to read and comprehend. Always attempt to place and align function blocks to minimize wiring clutter.

In LabVIEW, there is a set of tools at the top of the block diagram to help align objects.



The left hand toolbox allows you to line up all selected objects by different edges or centerlines. The right hand toolbox will distribute the selected function blocks so that they are evenly spaced.

Below is an example of what this program should look like when finished.



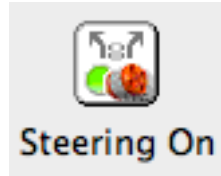
The function blocks are arranged from left to right, and all the wires pass data from left to right, never the other direction. The blocks are aligned, and the code pieces have comments next to them.

To comment your code, simply double-click anywhere in the whitespace of the block diagram. A small pane with a text cursor will appear. Type your comment and click on the block diagram or press **Enter** on the numpad to accept it (pressing **Enter** on the main keyboard will

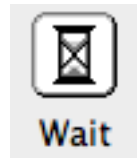
create a new line). Comments can be resized and moved on the block diagram. To edit the comment, double-click on the text.

Now you will create your first program.

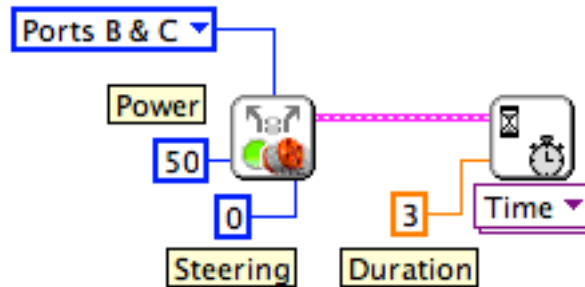
5. Place a **Steering On** function on the block diagram to command the Tribot to move forward along with a **Wait** block to stop the motion.
 - a. Select **NXT I/O >> Complete >> Motors >> Steering On** and click on the block diagram to place the function block.



- b. Next, select **NXT I/O >> Wait** and click on the block diagram to place the function block.



6. Configure the function blocks as shown.

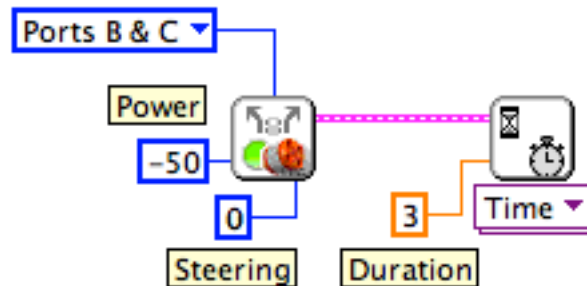


- a. Right click on the **Output Ports** terminal of the **Steering On** function block.
 - b. Select **Create >> Constant** from the context menu.
 - c. Select the option **Ports B & C** in the menu that you just created. These are the ports on the top of the NXT brick that should be connected to your motors.
 - d. Right click on the **Power (75)** terminal of the **Steering On** function block.
 - e. Select **Create >> Constant** from the context menu.
 - f. Enter a value of **50** in the field. This is the power (-100-100) at which the motors will operate.
 - g. Right click on the **Steering (100)** terminal of the **Steering On** function block.
 - h. Select **Create >> Constant** from the context menu.
 - i. Enter a value of **0** in the field. This is the amount of steering (-100-100) that will be used when the robot moves forward
 - j. Right click on the **Time (1 sec)** terminal of the **Wait** function block.
 - k. Enter a value of **3** in the field. This is the time (seconds) that the motors will run before moving to the next function block, which is yet to be inserted.
 - l. Wire the output **NXT** terminal of the **Steering On** function block to the input **NXT** terminal of the **Wait** block to determine the program flow.

This configuration will command the Tribot to move forward for 3 seconds at a power of 50%, and then implement the next function blocks that we will insert.

Notice that a power level of 50% was used instead of 100%. In general, it is good practice to specify conservative power values for motors to help conserve battery life on the Tribot. Use higher power values only when necessary.

7. Place another pair of these functions on the block diagram to command the Tribot to move backwards.



To create these blocks, it is unnecessary to complete all of the steps from parts four and five again. To save time and effort, the existing blocks can be easily copied and modified. Remember this method of code copying; it is a handy time saver.

- a. Click and drag a selection box around the existing blocks and all attached constants.
- b. Hold the **Ctrl** key (or **option** key on the Macintosh) and drag the block to the right. Release the mouse button to place the copied block.

Remember, new function blocks are almost always placed to the right of existing ones.

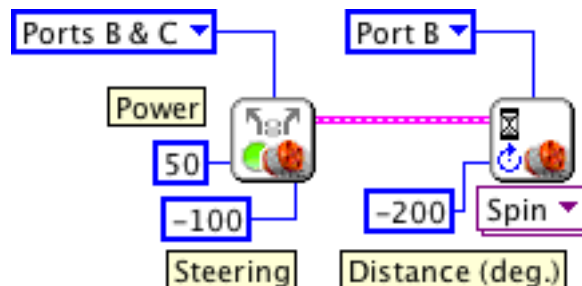
- c. Change the input on the **Power (75)** terminal to be -50. This choice will cause the Tribot to move backwards at 50% power.

You can always refer to the context help if you have trouble finding a terminal.

Once the Tribot completes these blocks, it will be at its original starting position.

Don't worry about connecting the individual function blocks at this time. We will connect everything together when we finish the program in step 13. For now, simply place the blocks in sequence from left to right.

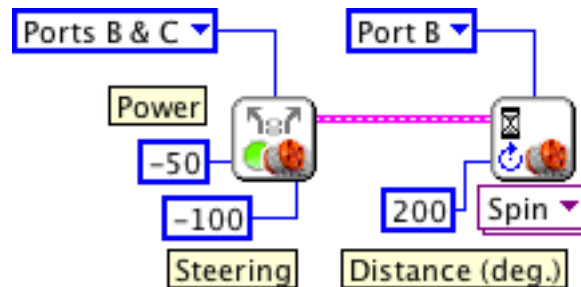
8. Place another pair of these functions on the block diagram to command the Tribot to turn right for a specific distance (i.e., rotation of a wheel).



- Create a constant on the **Power (75)** terminal and enter a value of **50**.
- Create a constant on the **Steering (100)** terminal and enter a value of **-100**.
- Select **Wait for Rotation** on the **Wait** function. The function will then display **Spin**.
- Create a constant on the **Position Change (180)** terminal.
- Enter a constant value of **-200**. This is the angular distance in degrees that the motor will rotate. Note that, because the wheel attached to **Port B** will be rotating backwards, the amount of rotation corresponds to negative degrees.
- Create a constant on the **Output Port (Port A)** terminal of the **Wait** function.
- Select the option **Port B** in the menu that you just created. This will select the motor whose rotational encoder will be used. (In this case the one corresponding to the left wheel.)

The steering constant tells the robot to turn right with the minimum possible turning radius. Turn values can range from -100 to 100, with 0 representing a straight line (no turn) and ± 100 representing a turn with both motors running equally in opposite directions (turning the Tribot on its vertical axis). The motor will rotate 200 degrees before stopping.

- Place another pair of these functions on the block diagram to command the Tribot to return to its starting position.



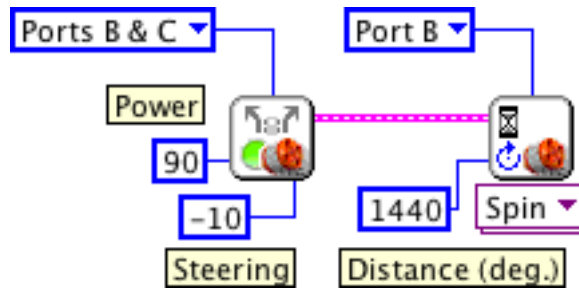
- Select the existing pair of blocks from step 8 and all attached constants by dragging a selection box around them.

You can add objects to the selection (or remove them) individually by holding the **Shift** key while clicking on them.

- Copy the selected items by holding the **Ctrl** key while dragging them to the desired location.
- Create a constant on the **Power (75)** terminal of the new blocks and change the value to **-50**.
- Create a constant on the **Position Change (180)** terminal and change the value to **200**.

These function blocks simply reverse the motor movement of the previous pair of blocks, returning the Tribot to its original position.

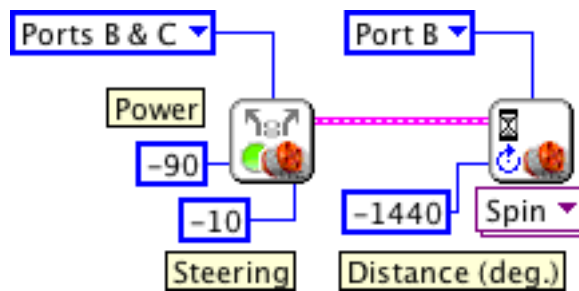
- Place another pair of function blocks to the right of the previous blocks that will command the Tribot to turn at a larger radius than the previous left turn function.



- Create a constant on the **Position Change (180)** terminal with a value of **1440**.
- Create a constant on the **Power (75)** terminal with a value of **90**.
- Create a constant on the **Steering (100)** terminal with a value of **-10**.

Because the steering constant is much closer to 0 than -100, the Tribot will turn to the left while still moving forward. This will result in a much larger turning radius than the previous turn function. The power has been increased to 90% to demonstrate the higher level output capability of the motors.

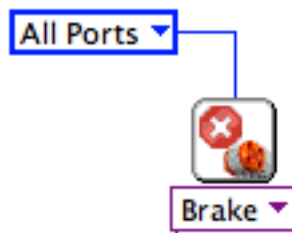
- Place a final pair of function blocks that will command the Tribot to return to its original position.



- Copy the previous pair of function blocks and all attached constants.
- Create a constant on the **Power (75)** terminal and set its value to **-90**.
- Create a constant on the **Position Change (180)** terminal with a value of **-1440**.

- Lastly, we need to stop the Tribot. The **Wait** function blocks enable the **Steering On** function blocks to operate for a certain interval and then allow the next function block to operate. At the end of this sequence of function blocks we want to place a function that will terminate all activity.

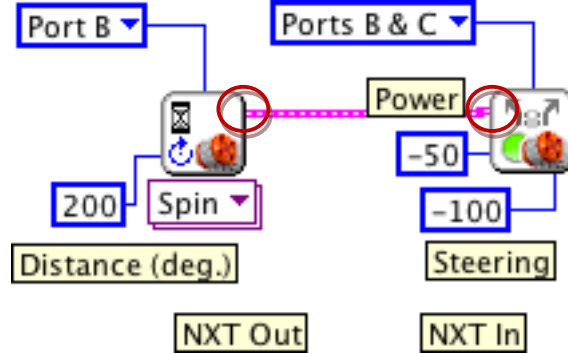
- Select **NXT I/O >> Motor** and click on the block diagram to place the function block.



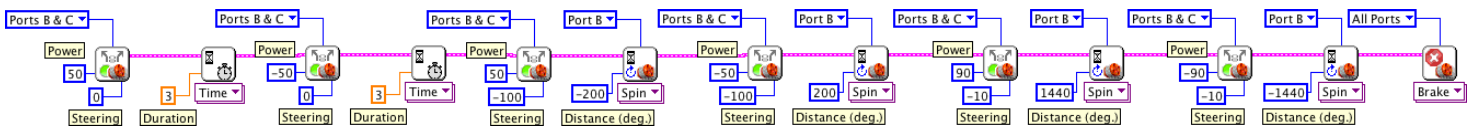
- Select **Motor Off >> Brake** on the **Motor** function. The function will then display **Brake**.
- Create a constant on the **Output Port** terminal and choose the **All Ports** menu item.

Placing this function block at the end of our chain of functions will cause the Tribot to brake to a stop.

13. Connect the function blocks using the **NXT** terminals to complete the program.



- Click on the **NXT out** terminal on the first function block to begin a new wire.
- Click on the **NXT in** terminal of the next function block to connect the wire.
- Repeat until all of the function blocks are connected via the sequence flow terminals.
The finished program should look like the following:



If your program does not look neat, refer back to the programming practices section between steps 4 and 5 for help on cleaning it up.

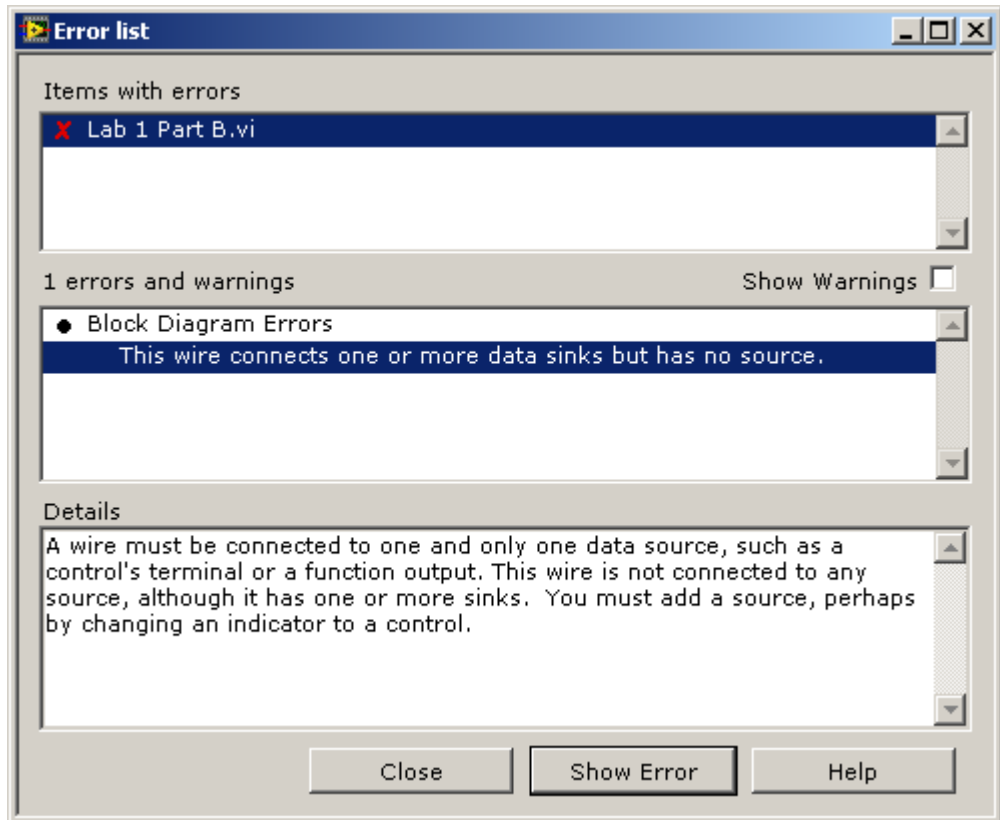
There is an easy way to tell if your program cannot be executed. The run arrow on the toolbar will change from a solid arrow



to a broken arrow.



If you have a broken arrow, then are one or more errors in your code. If your program has errors, you will not be able to download it to the NXT. To locate the errors, click on the broken arrow to display the **Error list** window.

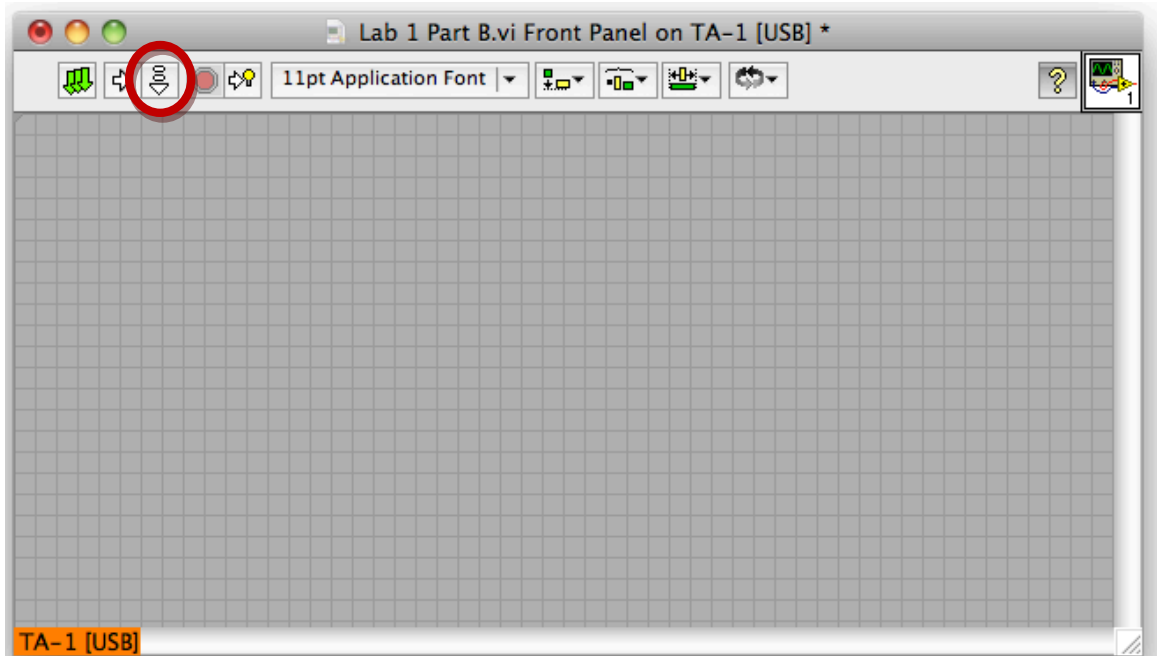


The middle pane displays the errors in the diagram. Below the error, the bottom pane describes the error in further detail and suggests solutions. To locate a specific error, locate it in the middle pane and double click on it. The block diagram will focus on the error and highlight it, making it easy to find and correct.

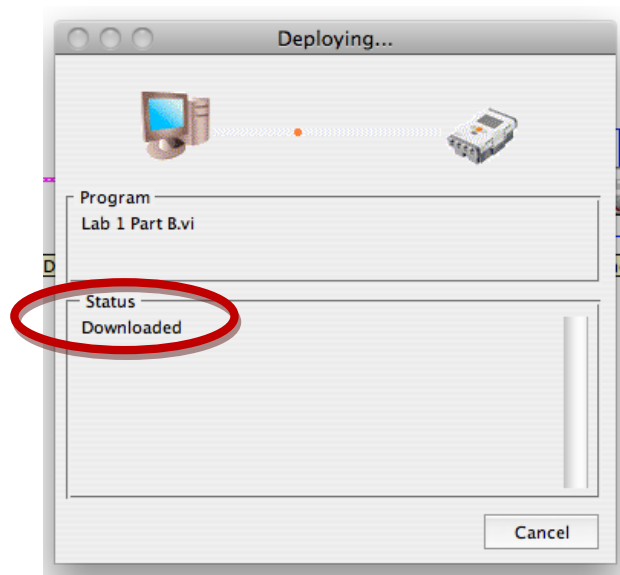
14. Save your program.

- a. In the program menu, select **File >> Save As**.
- b. Click on the **Browse** button, create a new folder named **Exercise 1**.
- c. Type **Lab 1 Part B** as the name of the file.
- d. Click **Save** to save the VI and close the dialog box.

15. Download and run your program on the Tribot.
 - a. Ensure that the NXT is connected and powered on and that the VI is still open.
 - b. The lower left hand corner of the Front Panel should show your NXT's name in orange and also indicate that it is connected by USB. If not, then right-click on the lower left hand corner and select your Tribot's name or choose **Find NXT....**
 - c. Select the **Deploy** option at the top of the Front Panel to place the program on the NXT.



The program will be compiled in NXT format and copied to the device. When the transfer is complete, the “**Deploying ...**” pop-up window will display a **Downloaded** status. The program has now been transferred to the NXT.



16. Once the download completes successfully, disconnect the Tribot from your computer, and place it somewhere on the floor where it has a clear area to run (preferably 3 feet or greater).
17. Run the program on the Tribot.
 - a. On the NXT, go to **My Files >> Software Files**.
 - b. Select **Lab 1 Part B** and select **Run**.

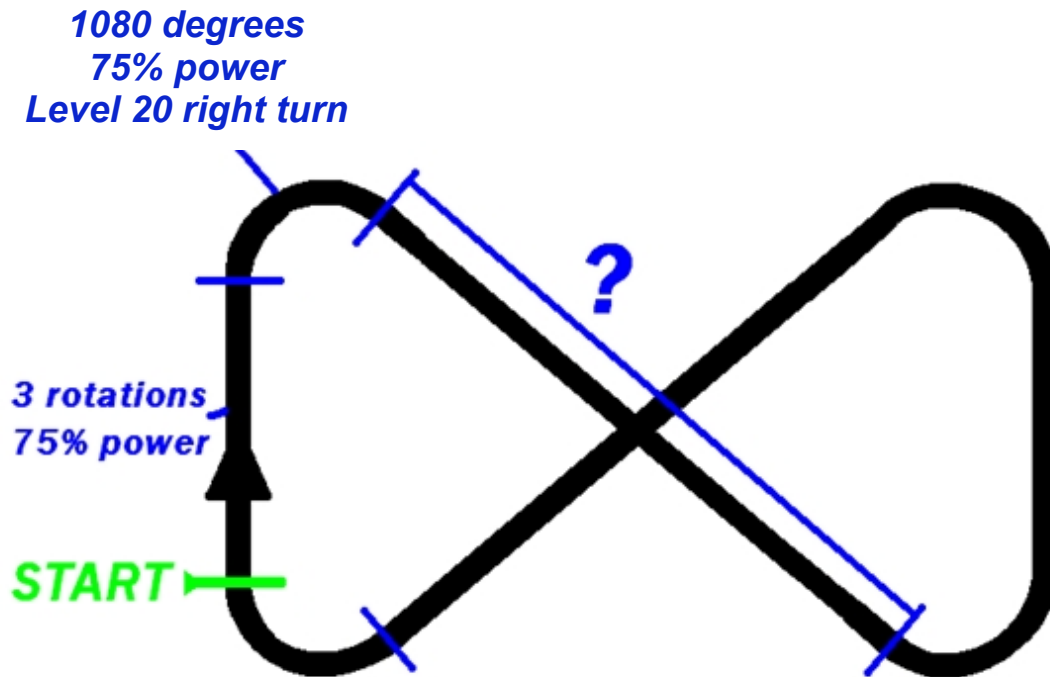
There is a battery indicator in the upper right corner of the NXT brick window that shows the current battery charge. Recharge or replace the batteries if the indicator shows low battery life to ensure that the Tribot moves reliably. The movement accuracy of the Tribot may be reduced if it runs on a low battery.

Questions:

- a) The fourth pair of function blocks can still return the Tribot to its initial position even when the block's **Power (75)** terminal input is set to **+50** (i.e., move forward at half power). How do the other input settings for that block need to be changed in this case?
- b) Why did the Tribot chassis seem to rotate approximately 90 degrees instead of 200 degrees on the first set of left and right turns? Did any part of the Tribot rotate 200 degrees?
- c) Change the last **Wait** function to observe the motor attached to **Port C**. Does the robot still return to the same position as before? If not, how does the stopping point differ and why?

Part C (Challenge): Move the Tribot Along a Path

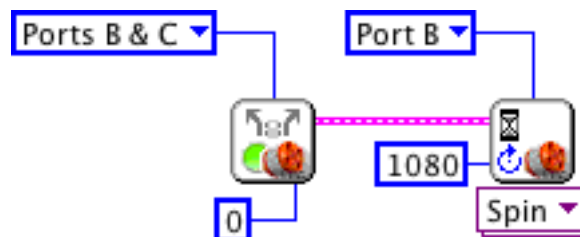
The objective of this challenge is to move the robot in a figure eight pattern as shown in the following image using a series of Sync Distance function blocks.



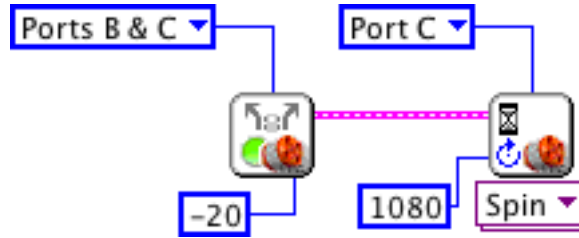
The following criteria must be met to pass the challenge:

- 1) The robot must start with two sets of function blocks that have the following settings:

Function block set # 1:



Function block set # 2:



You should design the remaining sets of functions blocks to complete the figure 8 pattern, keeping the motions as symmetric as possible. Then, terminate your sequence of commands by using a **Motor Brake** function to stop your Tribot.

- 2) **The robot must return to the same location and orientation from where it started.**
- 3) **Once the program has been started on the robot, nothing is allowed to touch the robot until it finishes.**
- 4) **The robot must travel in a figure eight pattern similar to the one shown in the above diagram.**
- 5) **The motion of the Tribot is subject to random factors, causing the Tribot to return to a slightly different position after each run even when the program is unchanged. Each team should use multiple trials to improve their program. Determine an average offset from the starting point for several trials before modifying your code, rather than relying on the outcome of a single trial.**
- 6) **When you are finished, demonstrate your program to your instructor.**

Part D (Conclusion): Repeat Each Experiment 11 Times

For the code that you produced in Part B and Part C, please run each program 11 times in a row with the **SAME** program. Mark the starting point on the floor with a piece of tape. Mark the ending point of the first run with another piece of tape. Run your program 11 times and measure the deviation in the **ENDING** point from the first run. Please make a graph of your results. What are your conclusions about the variability in your robot.

Good luck!