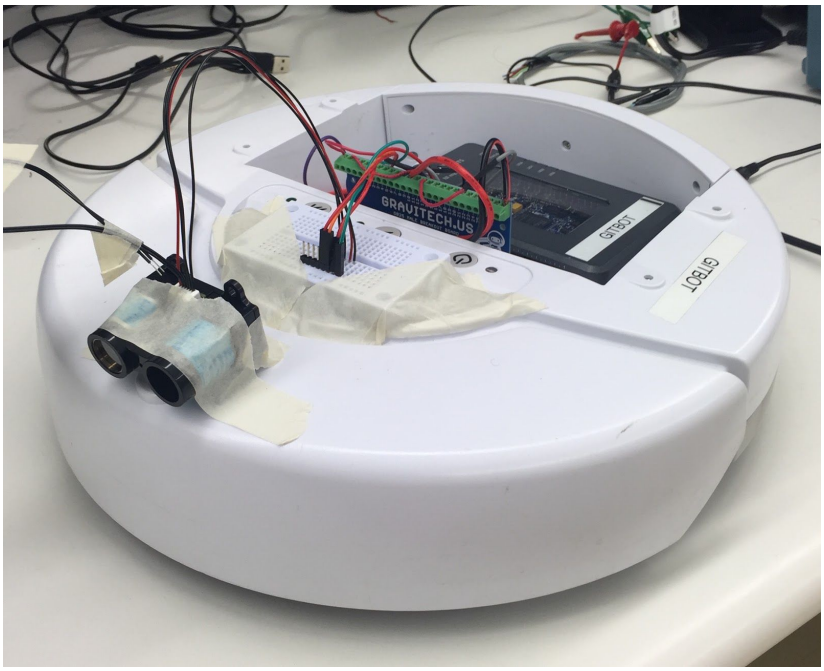
Brian Team (dot4qu) [Mariobots]
Ben Browning (brb7nu) [Mariobots]

ECE 4450 Final Project - IR Sensing

Project Proposal

"Within the demo, our robot would be expected to sense an obstacle without an orange marker and display this fact somehow (via Boolean or otherwise), and change its behavior accordingly. This behavior change could be seen as a slowing of the robot, or beginning its maneuver around the before it comes in contact with it."



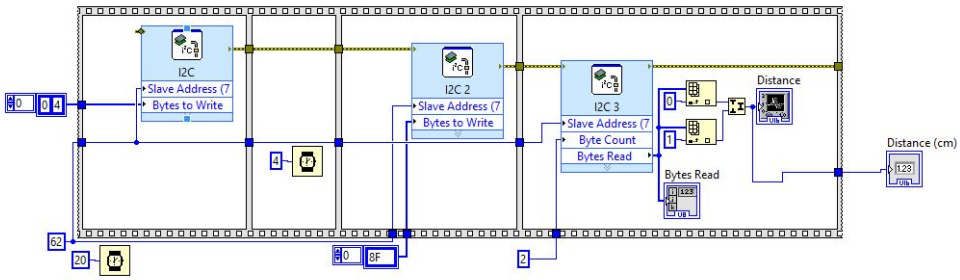
Materials

- PulsedLight LIDAR-Lite v2 "Blue Label" (Model LL-905-PIN-02) Laser Ranging Module
- National Instruments myRIO
- iRobot Create

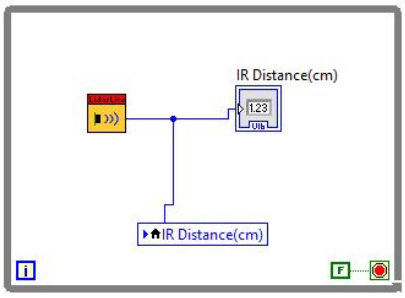
Procedure

1. Refer to the LIDAR-Lite datasheet to determine the slave address of the sensor.
    a. As of 05/4/2016, this information could be found within the LidarLite GitHub, located at https://github.com/PulsedLight3D. The address for the model used in this lab is 0x62.
2. Create a sequence structure in LabVIEW to perform the I2C communication.

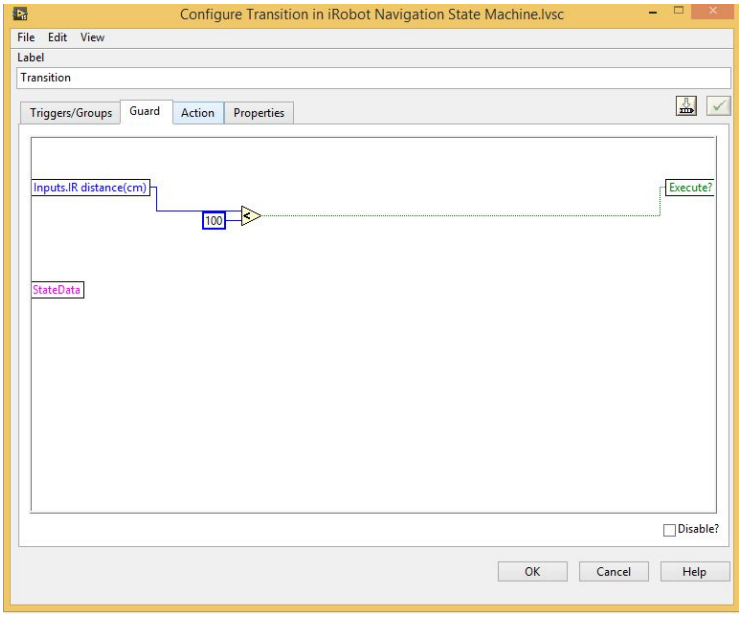
3. The first frame of the sequence resets the IR sensor. To do this, write a value of 0x04 to register 0x00.
4. The second step waits 4 ms to give the sensor time to respond.
5. The third frame writes 0x8f. This tells the sensor which register to read in the next frame.
    a. The reason it is 0x8f is because, as per the datasheet, you change the MSB of the hex number to a 1 if you want to continuous read, and the normal register you read from is 0x0f. This means we're writing 0b10001111 instead of 0b00001111.
6. The fourth frame reads two pieces of data from the ranging register -- one high byte and one low byte. They are in reversed order and must be switched then recombined into the output distance.
    a. This can be displayed as a number (distance in cm) or shown on a graph.



7. Add a new while-loop into the existing iRobot Navigation Target block diagram that contains this subVI. This while loop is completely isolated from the main while iRobot while loop.



8. To transfer data between the concurrent while loops, create a local variable to pass the sensor output distance value into the state diagram.
9. Within the state diagram, create a new state to handle seeing an obstacle.
10. Create a guard running from the Drive state to the new Obstacle state which compares the sensor output distance to a constant threshold. For sensor distances below this threshold, the iRobot will stop.



11. Within the Obstacle state, set the action to set the both wheel speeds to zero.
12. Set the guard from the Obstacle state back to the Drive state to be when the distance variable rises above the same threshold as set in step 10.

Purpose

The purpose of this project was to use the Lidar-Lite v2 IR sensor along with the iRobot to sense objects in front of the robot. We added to our previous Mariobot code to produce the desired functionality. The end result was a line-following, barcode-reading, traffic light-sensing, obstacle detecting robot that would stop if it came within a certain distance of an object in front of it. Although the overall outcome of this project does not seem very significant, the results pave the way for easy communications with many other iRobot peripherals. The I2C setup was by far the hardest part of our design, and can be used in the future to add and interact with many other external devices that communicate through I2C.

This project could have been extended given more time. Instead of coming to a full halt within a specified distance, proportional control could have been utilized. The diminishing distance towards the obstacle would be scaled and subtracted off the overall speed. This would enable the Mariobot to move slower as it became closer to collision, and stop before actually hitting the obstacle.Overall, however, this project implemented the behavior specified within the proposal while further expanding our knowledge of LabVIEW's capabilities with different types of communication.

Photo of the participants