6-9-2022

B2B2: LiDAR 2D Mapping Rover

Laurence Kim

Isaiah Youngblood

Quintion Turner

SANTA CLARA UNIVERSITY

Department of Electrical Engineering

Date: June 9, 2022

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY

Laurence Kim

Isaiah Youngblood

Quintion Turner

ENTITLED

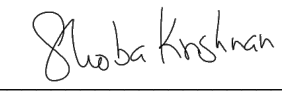
B2B2: LiDAR 2D Mapping Rover

BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF SCIENCE IN ELECTRICAL ENGINEERING

Thesis Advisor: Dr. Maria Kyrarini

Thesis Advisor: Dr. Sally Wood



EE Department Chair

B2B2: LiDAR 2D Mapping Rover

By

Laurence Kim, Quintion Turner, and Isaiah Youngblood

SENIOR DESIGN PROJECT REPORT

Submitted to

the Department of Electrical Engineering

of

SANTA CLARA UNIVERSITY

in Partial Fulfillment of the Requirements

for the degree of

Bachelor of Science in Electrical Engineering

Santa Clara, California

Spring 2022

B2B2: LiDAR 2D Mapping Rover

Laurence Kim

Quinton Turner

Isaiah Youngblood

Department of Electrical Engineering

Santa Clara University

June 9, 2022

ABSTRACT

Autonomous machines are becoming more popular and useful with even self-driving cars being a thing of the present. Most of these machines navigate using cameras and LiDAR which does not detect glass, therefore the machines give misleading results when objects and obstacles are transparent to the wavelengths of the light used. This is problematic in modern building floor plans with glass walls. A solution is to build a ROS system that fuses ultrasonic sensors with LiDAR sensors in order for a robot to navigate in a building that has glass walls. Using both sensors, the final product is a robot that creates a 2D map using Simultaneous Localization and Mapping (SLAM) as well as other pertinent Robotics Operating Systems (ROS) packages. This map enables any mobile robot to pathplan from point A to B on the now created 2D floor plan that incorporates glass and non-glass obstacles. This saves time and energy when compared to a robot that moves from point A to B that has to continuously change paths in the presence of obstacles.

Table of Contents

1 Introduction **1**

1.1 Problem and Background **1**

1.2 Motivation **2**

1.3 Objectives **3**

2 Technologies **5**

2.1 Ridgeback **5**

2.2 ROS **6**

2.3 LiDAR Sensors **7**

2.4 GMap Generation **8**

3 Proposed Systems **10**

3.1 Overview **10**

3.2 Ultrasonic Sensor **10**

3.3 Ultrasonic ROS Package **15**

3.4 LiDAR Sensor **15**

3.5 LiDAR ROS Package **16**

3.6 Gmapping with LiDAR and Ultrasonic Data **16**

4 Experimental Results **19**

4.1 LiDAR Sensor Results **19**

4.2 Ultrasonic Sensor Results **22**

4.3 Gmapping with LiDAR and Ultrasonic Sensor Results **27**

5 Deliverable Analysis and Conclusion **29**

5.1 Performance **29**

5.2 Sustainability Consideration **29**

5.3 Reliability Expectations **30**

5.4 Future work **31**

5.5 Professional Constraints **31**

5.6 Summary and Conclusions **32**

References **34**

List of Figures

1.1 Office Workspace without Glass (left image)vs. Modern Workspace Utilizing Glass Walls (right image) **2**

2.1 Clearpath Ridgeback Robot **5**

2.2 ROS Node-Topic Structure example **7**

2.3 Time of Flight Principle Example **8**

2.4 Rviz Occupancy Grid Example **9**

3.1 Functional Block Diagram **10**

3.2 How an Ultrasonic Sensor Works **11**

3.2 Arduino Mega Board **12**

3.2 HC-SR04 Ping Sensor Wiring Diagram **13**

3.2 Ping Preprocessing Pseudocode **13**

3.2 MB1013 MaxBotix Sensor Wiring Diagram **14**

3.2 Pseudocode to Read and Convert Data **14**

3.3 Publisher, Topic, and Subscriber Node diagram **15**

3.4 UST-Hokuyo LiDAR Sensor **16**

3.5 Gmapping Image **16**

3.6 Ultrasonic Sensor Frame Transformation with respect to the Base Frame **17**

3.6 Rviz simulation: LiDAR and Ultrasonic Sensor Data Fusion (Zoomed in vs. Zoomed Out) **18**

4.1 Rover Moving through Test Environment (4th Floor Hallway) while Controlled with a Joystick **19**

4.1 Laser Data Script Print Output **20**

4.1 Definition of the Laser Scan Parameters **20**

4.1 Picture of Ridgeback at ~Position(X,Y) **21**

4.1 Actual Snapshot of LiDAR 2D Mapping at ~Position(X,Y) **22**

4.2 SCDI Room 4029 **22**

4.2 SCDI Balcony **23**

4.2 Distance vs Sample Number with Ping Sensor **24**

4.2 Distance vs Sample Number with Maxbotix Sensor **25**

4.2 Dual Technology Ceiling Mount Occupancy Sensor **26**

4.2 Ping Sensor Attached to Ridgeback **26**

4.3 Laser Scanner combined with Sonar Range in Rviz and Gazebo **27**

List of Tables

Table 1: Comparison between HC-SR04 Sensor and MB1013 MaxBotix Sensor **11**

1 Introduction

1.1 Problem and Background

With the recent advancement of robots in the past decade, robots have started replacing monotonous tasks through automation. As robot automation has started to gain traction in many fields, the reality of mobile robots has also been introduced. Furthermore, in order for these mobile robots to effectively traverse in an environment, these require more advanced forms of perception technology that accurately detects all forms of obstacles. The advancements in perception technology include robust sensors. Robots need to be able to detect all sorts of obstacles around them and then estimate the location of the objects with respect to the robot, so that they will accurately map a real-human environment.

Just as eyes enable humans to see where they are going, most modern robots use sensors to perceive their environment. Typically, camera and LiDAR sensors are used for obstacle detection. Despite the effectiveness of the camera sensor, it is inadequate in a modern building environment with glass walls/panels (as shown in Figure 1), as the glare or refraction from the glass disrupts the input [20]. Furthermore, LiDAR sensors are limited when trying to detect glass walls, because the near infrared wavelengths (900 nm vs 400–700 nm of everyday light) emitted by the LiDAR sensors pass through all translucent obstacles. The LiDAR will produce accurate measurements from objects on the other side of the glass instead of the glass itself.

Furthermore, robots often use a methodology called Simultaneous Localization and Mapping (SLAM) in order to navigate areas. Utilizing SLAM, a robot creates a two-dimensional (2D) map of the environment with respect to its location, using the LiDAR scanner and the camera sensor [21]. The utility of creating a map of the environment which provides future benefits such as autonomous navigation. Since SLAM typically relies on the LiDAR scanner and the camera, the map generated by the robot will not be accurate in an environment with glass walls and glass doors. This is crucial because glass walls and glass doors are popular in modern architecture.

In brief, a 2D grid map (Gmap) is a digital representation of an environment such as walls and objects within the room [4]. To generate the map in SLAM, you need sensors to

measure how close the robot is to the object. These sensors will omit the glass walls and therefore crash into them when path planning, since there are huge gaps in their mapping. The robot will not recognize where walls and doors should be in a building like the SCDI (Sobrato Discovery and Innovation) building at Santa Clara University. The interior of the SCDI building is shown in Figure 1 (right image). Having a sensor that detects glass walls and glass doors will improve the robot's performance and make it more accurate as it generates the 2D map in this kind of environment.

Figure 1: Office Workspace without Glass (left image) vs. Modern Workspace Utilizing Glass Walls (right image)

1.2 Motivation

As mentioned in the problem statement, LiDAR and camera are both ineffective in mapping modern indoor environments because these sensors are incapable of detecting the glass walls. Therefore, it was necessary to create an additional system that utilized ultrasonic sensors which will improve the robot's performance and reliability.

Ultrasonic sensors reflect sound instead of light to sense objects. This means the robot should be able to use this sensor to sense translucent and reflecting surfaces like glass since the sound waves echo off the glass walls. Ultimately, the combination of LiDAR and ultrasonic sensors creates a robust system that detects all forms of obstacles in a modern building, and therefore accurately creates a 2D map with the corresponding mapping software.

There exist solutions that modify LiDAR Sensors when sensing glass. M. Awais on Improved laser-based navigation for mobile robots [28] used this approach in his project for detecting glass. The author explored only using LiDAR sensors and optimizing the sensor to recognize glass by examining the reflections and refractions of the glass. The author uses probability functions as part of his algorithm in order to understand how the sensor receives weak signals when LiDAR senses these refractions and reflections. Awais uses two complementary filters in his model with the first filter computing the rolling window standard deviation that locates the presence of glass, and the second filter combining measurements of distance and intensity to determine the glass width profile and location. There are other studies like [33] and [34], that attempt to create an algorithm to detect glass using the pre-existing LiDAR and camera sensors as additional software solutions are generally preferred over additional hardware for cost efficiency reasons. However, since there are fewer studies on adding ultrasonic sensors, this project seeks to add knowledge to this area of research.

When it comes to studies that use the same solution, the main issues implementing ultrasonic sensors with LiDAR revolves around range and reliability of the readings. While implementing the ultrasonic sensors, it is imperative to consider its operating range when measuring distances otherwise, the data collected from the sensor will be inaccurate from the actual distance measured [35]. In another project using sensor fusion, the most accurate data collected from the robot was less than a foot away from the glass itself. This is a considerable obstacle going forward [18].

1.3 Objectives

The main objective of the project is to create a 2D map of a human test environment that has glass walls using sensor fusion between LiDAR and additional sensors. One sensor that is effective at mapping the environment is an ultrasonic sensor. Ultrasonic sensors are needed to make a map of the environment with glass because they will detect objects or realize depths that are not visible due to hazards or visibility issues [22]. Therefore, the project implements the LiDAR system with an ultrasonic sensor system to create a robust obstacle detection sensor system. This project has the following three main objectives:

- Get a reliable ultrasonic sensor’s raw data in digital form.

- Integrate the working sensor into the robot's ROS package, a current leading research standard for robotics.
- Use this sensor integration to create an accurate and reliable 2D map.

In order to best tie in the functionality of these sensor systems, navigate the rover, or mobile robot, and accurately output a map, the Robotics Operating System (ROS) was the infrastructure upon which the project was built upon [11]. ROS is an open-source framework with a set of libraries that is meant to create robotics applications. This middleware controls simultaneous communication between the hardware on the robot and software that creates the 2D map. ROS was the most attractive option for the project framework as it is the current leading industry standard for robotics and has vast open source resources that are available. ROS also comes with tools such as Rviz and Gazebo which are monumentally useful in both digitally visualizing and functionally testing the robot during simulation. Only after passing safety and functional tests, will the robot deploy in the real world. The final product of the robot that is able to fuse the LiDAR and ultrasonic sensor systems to accurately map an indoor modern building environment is called B2B2.

2 Technologies

2.1 Ridgeback

The Clearpath Ridgeback Robot is a midsize indoor robot platform (960 x 793 x 311 mm) that uses an omni-drive to move manipulators and payloads of up to 220 lbs with ease. The omnidirectional base provides precision positioning in constrained environments and comes fully integrated with an onboard computer, front and optional rear laser scanners. The Ridgeback, as shown in Figure 2, was chosen as the platform upon which the entire project is built because the robot is ROS-centric and because of the pre-existing open source packages that are available to new developers.

Figure 2: Clearpath Ridgeback Robot

2.2 ROS

The Robot Operating System (ROS) is an open-source framework that helps researchers and developers build and reuse code between robotics applications. Additionally, ROS hosts a global open-source community of engineers, developers and hobbyists who contribute to making robots better, more accessible and available to everyone. The open source resources for development and pre-existing simulation software packages are what made ROS the obvious choice for infrastructure of the project. These simulations involve a multitude of tests regarding functionality of navigation, data collection, and 2D map creation. Running these simulations are essential to safely test the robot because if the robot deploys safely in a digital simulation, the robot is also expected to perform in a similar manner when deployed in a real-world environment. The types of simulation software and their purposes will be further explored later in this section. The official ROS website was the most reliable resource for tutorials and nuances of how to properly function within ROS [11].

One of the fundamental components that enable ROS' modularity and parallel functionality of its modules are the packages. A package is an organization of relevant software such as its source or configuration files regarding a specific functionality within ROS. ROS packages exist to contain relevant software or configuration files and to promote modularity within a workspace, or a folder where a user modifies, builds, and installs packages, so that each functionality is lightweight and reusable. Within the source of the packages are ROS nodes. A ROS node is an executable script that ROS uses to communicate with the scripts, no matter what programming language is used. The typical programming languages are C++ and Python for the scripts. In the project, C++ and Python are the programming languages that were implemented.

ROS nodes are either publishers or subscribers. Publisher is a script that sends the messages of some standard message type to a particular messaging channel, called a topic. On the other spectrum, the subscriber receives the messages whenever a message is published to the topic. Figure 3 shows a visual diagram of how the publisher communicates with the subscriber through the topic.

Before any real world deployment with robots, successful online simulation is a mandatory step; this method embodies ROS Visualization (RViz) and gazebo. Rviz is a graphical interface that allows you to visualize the functionality of the robot, such as the pulse readings of the LiDAR data as seen in Figure 5, using plugins for many kinds of available topics [13].

Gazebo is a software tool that simulates how B2B2 functions in the real world. Unlike Rviz, the user has the option to customize and simulate their own environment on Gazebo for the robot to navigate through [2]. Testing in Gazebo enables risk and danger free testing without having to deploy in a real-world environment. Only after passing all the online simulations which critique the robot's ability safety precaution measures and functionality tests, is the robot allowed to run in a real-world environment. The main safety measure that was tested was its ability to stop instantly with a kill switch. Functionality tests include the reading of the LiDAR sensors, ultrasonic sensors, and expected movement with the joystick.

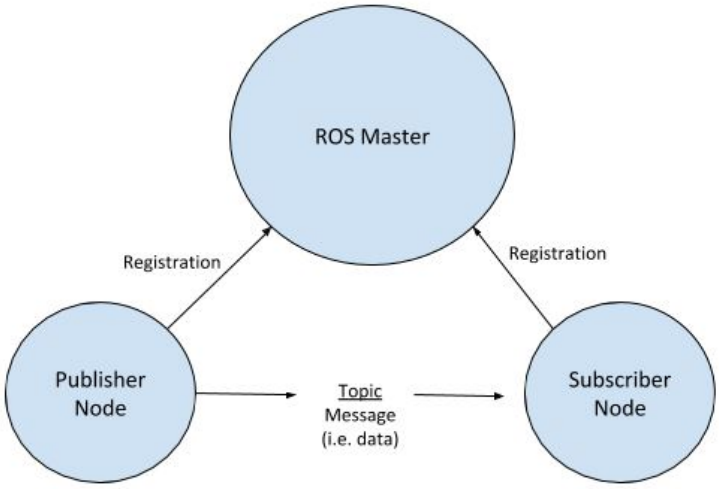


Figure 3: ROS Node-Topic Structure example

2.3 LiDAR Sensors

LiDAR stands for "Light Detection and Ranging", and is commonly used in aircrafts to map the earth. The laser scanner sends a beam of light closer to the infrared wavelength around 0.8 µm to 1.6 µm. Each laser pulse takes a fraction of a second around 4–15 ns in duration. It sends pulses of light toward the direction of its beam angle of 270 degrees [27]. The scanner then receives the light that is scattered and reflected by the objects. This method is called the *Time-of-Flight (TOF) principle* which measures the time it takes for a wave to travel from a source (a time-of-flight sensor) to an object and back, as shown in Figure 4 [26]. The distance (range) between the LiDAR sensor and the object is calculated by multiplying the speed of light

by the time it takes for the light to be sent and received by the sensor. LiDAR sensors these days execute this process with low margins of error especially at close ranges.

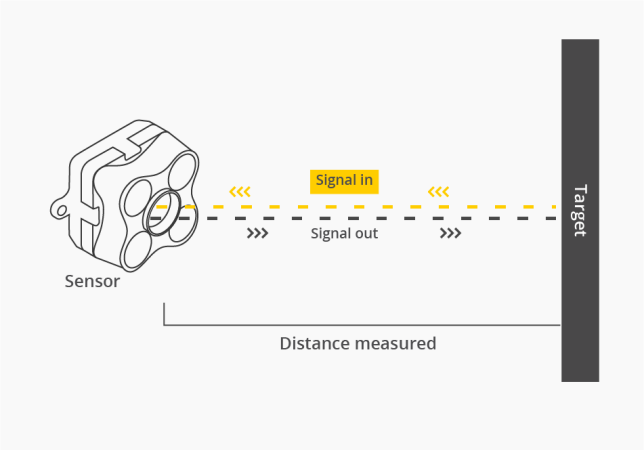


Figure 4: Time of Flight Principle Example

2.4 GMap Generation

Mobile robots that utilize ROS commonly use Gmapping as a way to keep track of their location in a given space. Gmapping is a ROS open source package that receives the LiDAR, odometry, and transform (tf) measurements, in order to create a 2-D occupancy grid similar to the one in Figure 5. Occupancy grids are grids that contain information about the environment that are collected from sensors loaded from prior knowledge or in real-time. Odometry is a package that obtains the distance and direction traveled by the wheels. However, as mentioned by Yousif et al. [29], each present robot localization is incrementally calculated from the last localization calculation which poses "*measurement errors [that] are accumulated over time and cause the estimated robot pose to drift from its actual location*" [29]. Furthermore, the tf package is necessary for the robot to keep track of multiple coordinate frames over time like the frame of the wheels and the frames of the robot. The robot will move as one unit in simulation. Furthermore, the Global frame acts as the main reference frame. The green and red tick mark in the upright quadrant of Figure 5 marks the center of the global frame in that example [17].

To create the map in Gmap, the robot uses Simultaneous Localization and Mapping (SLAM) which is a technique used by the robot to concurrently move, scan, update the map, and localize itself with respect to the global frame. The map is visualized in Rviz so the user

visualizes what the robot "visualizes." In the example in Figure 5, the robot visualizes and records an occupancy grid. The gray area is the area the robot marks as unoccupied free space, the dark grayish blue represents uncharted territory, and the black lines represent occupied boundaries such as walls and furniture sensed by the LiDAR sensors. The red lines that connect to the wall near the robot is the laser data from the robot's current position [13]. Notice that the red lines bleed into what is supposed to be a boundary or wall. This is due to the error associated with the odometry. As there are several calculations of distance traveled, and different surfaces and bumps causing different types of friction, the robot will tend to drift off from its calculated pose [29].

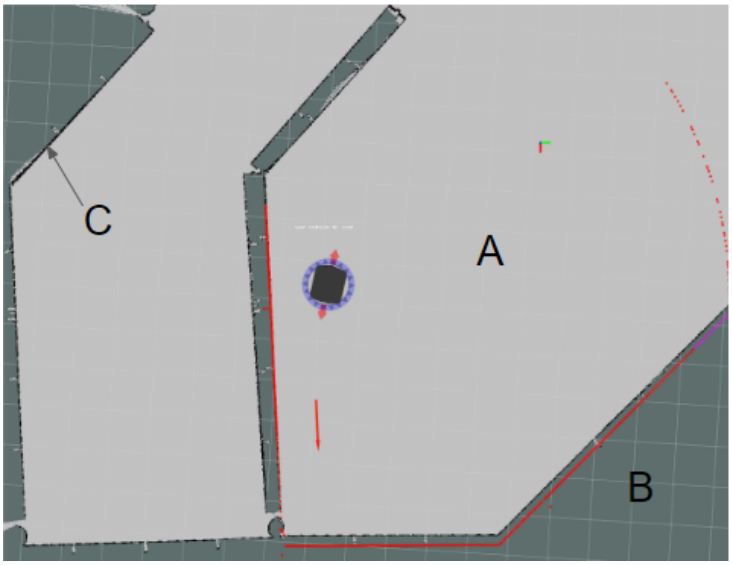


Figure 5: Rviz Occupancy Grid Example

The gray portions (A) is marked as unoccupied free space, the dark grayish blue represents uncharted territory, the blueish gray regions (B) near the edges of the boundaries are all unmarked territory that is yet to be scanned. and the black lines (C) represent occupied boundaries/obstacles by the LiDAR sensors while the red lines layered along the blacklines near the rover are the boundaries that are currently being detected.

3 Proposed Systems

3.1 Overview

This section focuses on the proposed system by discussing the hardware and software designs implemented to integrate an ultrasonic sensor on a robot for generating a 2D map. Figure 6 shows a functional block diagram of the system to combine the data from the ultrasonic sensor and the laser scan data and then create a map using the Gmapping package. Throughout the entirety of this chapter, each subsection will sequentially explain more details about this functional block diagram from the sensors to the ROS packages and the final 2D map.

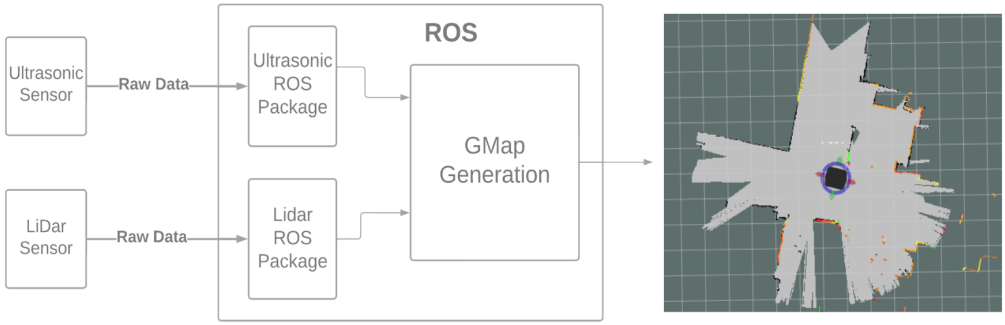


Figure 6: Functional Block Diagram

3.2 Ultrasonic Sensor

As shown in Figure 7, the ultrasonic sensor emits an ultrasonic wave above human hearing, 40kHz, and receives the wave reflected back from the target as long as it is within operating range. Similar to LiDAR sensors, most ultrasonic sensors measure the distance to the target by measuring the time it takes to send and receive the pulse. Most sensors have a transmitting channel and a receiving channel, while there are some that have a single channel called a transducer, which operates like a transmitter and receiver at the same time. The following equation is used to calculate distance:

$$D = (c * t)/2 \quad (1)$$

Where D is the distance between the ultrasound sensor and the object, t is the time between emission and reception, and c is the sonic speed through the medium. The medium is air at room temperature and the sonic speed in air is 343 m/s [23], which is significantly slower than the speed of light that LiDAR sensors transmit. Therefore, compared to LiDAR, ultrasonic sensors have lower ranges and less accurate readings.

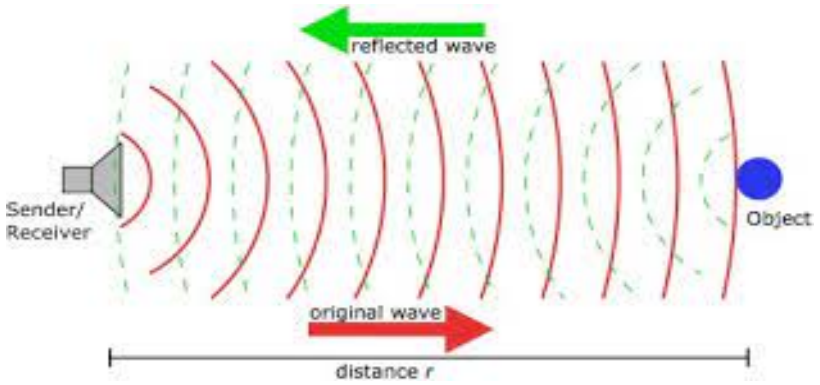


Figure 7: How an Ultrasonic Sensor Works

The two ultrasonic sensors that were implemented into the ultrasonic system were the **HC-SR04 Ping Sensor (ping for short)** [9] and **MB1013 MaxBotix Sensor** [25]. The ping sensor has two separate channels: a transmitter, trigger, and a receiver, echo. The MaxBotix sensor on the other hand has one channel called a transducer. As shown in Table 1, the differences between the two sensors are their operating ranges and effectual angles. The ping sensor detects objects between 2 cm and 400 cm, while the MaxBotix sensor detects objects between 30 cm to 500 cm.

Table 1: Comparison between HC-SR04 Sensor and MB1013 MaxBotix Sensor

Sensor	HC-SR04 (Ping) Sensor	MB1013 MaxBotix Sensor
Power Supply	+5V DC	2.5 V to 5.5 V DC
Effectual Angle	<15°	44°
Ranging Distance	2 cm – 400 cm	30 cm - 500 cm

The ultrasonic sensors are connected to an Arduino Mega 2560, a development electronic board based on the Atmega2560 microcontroller, as shown in Table 1. The board has 54 digital input/output pins, 16 analog inputs, and a reset button. The board comes with a DC power jack to power up the board using a V_{IN} pin on the board. The recommended input power voltage is 7-12 volts. Another alternative to power up the board is to use the USB port and connect it to the computer. Not only is the USB port used for powering up the board, but it is also used for uploading programs. There is no particular reason for this selection besides the fact that the sensor is compatible with the Arduino software and hardware. Any Arduino board with the appropriate number of pins is satisfactory. The Arduino is connected to the robot via a USB cable, which transmits the distance readings from the sensors in real time as the robot is moving through the building, so that the robot will log the readings and eventually convert these logs into a map.

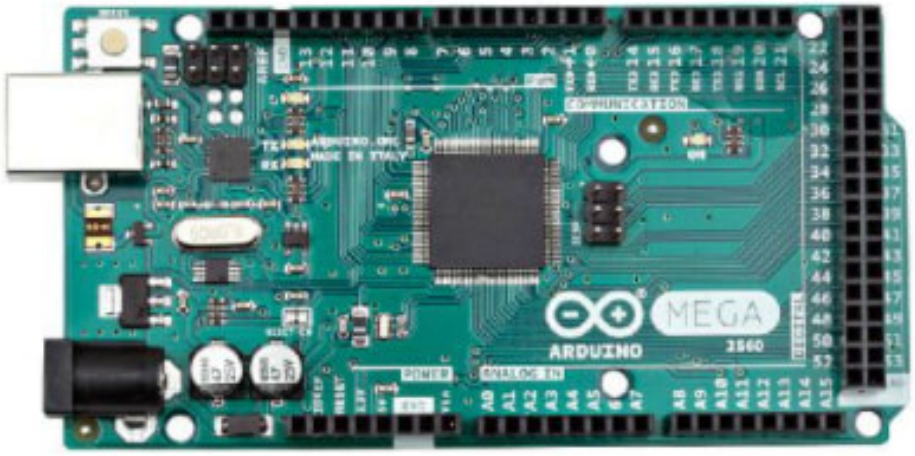


Figure 8: Arduino Mega Board

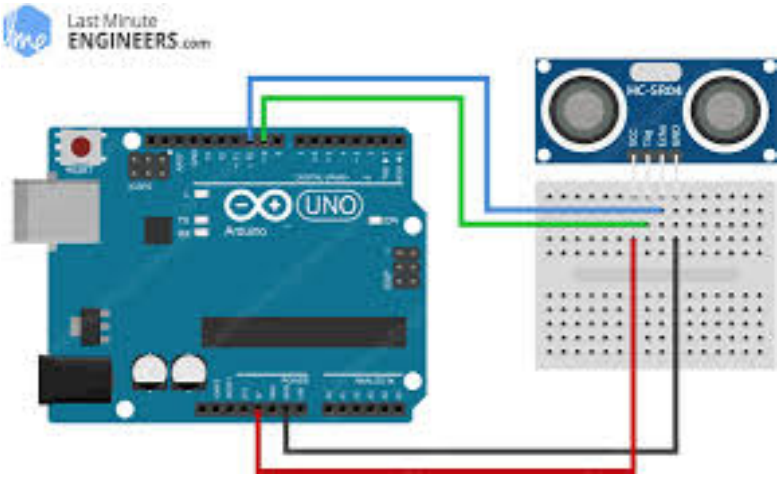


Figure 9: HC-SR04 Ping Sensor Wiring Diagram

Figure 9 shows an example of how to wire the ping sensor to the Arduino board. Starting from the left, is the V_{CC} which is the supply voltage for the ping sensor. The next pin is the Trig pin which sends an ultrasound wave to the object within range. The third pin is the echo pin which receives the ultrasonic wave reflected back from the object within range. Lastly, is the GND pin, or the ground pin, which serves as a common point for the electronic parts to operate correctly. The data the ping sensor collects is the time it takes for the ultrasound wave to reach the desired target and travel back to the sensor. This method is called the *Time of Flight principle* [24]. The appropriate Arduino script file (.ino) was implemented as a ROS package as a means to convert the measured time to distance using Equation (1) which was introduced earlier in this section.

```
function PingPreprocessing(){

    clear trigger PingPreprocessing

    ultrasonic sensor pings a wave

    duration = read the echopin which reads the time it took for the wave to travel

    distance = duration * 0.034 /2 as per our given conversion factor from before

    return distance


}
```

Figure 10: Ping Preprocessing Pseudocode

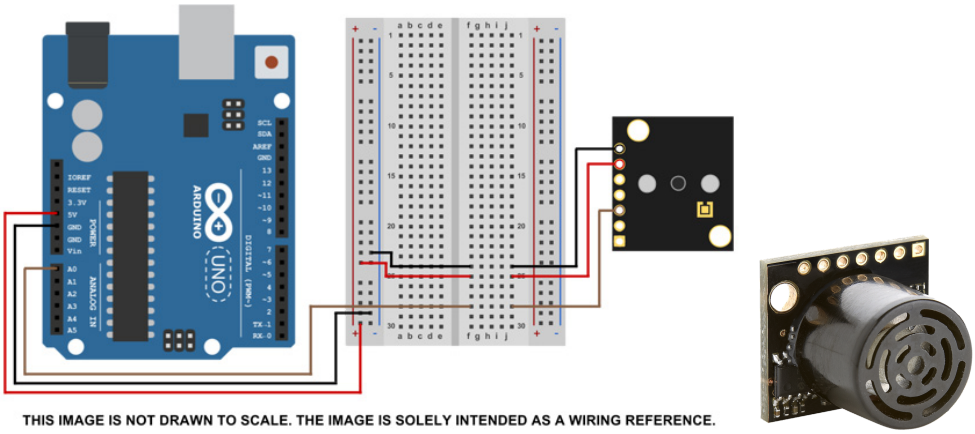


Figure 11: MB1013 MaxBotix Sensor Wiring Diagram

Figure 11 shows the wiring diagram of the MaxBotix sensor. The very top pin is the ground pin and the second pin is the voltage pin. The fifth pin is the analog voltage output, a pin whose voltage corresponds to the distance between the sensor and the object within range. Unlike the Ping sensor, the Maxbotix sensor has a couple other ways of measuring the distance between itself and the object. As shown in Figure 11, the sensor is configured to read analog voltages in bits. Referencing the sensor's datasheet, the conversion factor was used to translate the digital bits read in from the sensor as distance as each 1 bit was equivalent to 5 mm. Implementing the conversion factor into Arduino script (.ino) was a necessary preprocessing method for the Gmapping software because the Gmapping software is unable to work with given bits otherwise. Only when given the distance as a measurement of meters between the object and the sensor, will the Gmapping software be able to localize the robot from the obstacle. As shown in Figure 12 below, the read sensor function takes in the bit values from the sensor, multiplies the bit by 5, which converts it from a unit of bits to mm, then converts the units from mm to m.

```
function dataPreprocessing(){

    "bits" =  reading the analog voltage from the sensor readings in units(bits).

    "millimeters" = multiplying  "bits" by 5 which directly converts the analog voltage bits to millimeter units.

    Convert "millimeters" to "meters" by dividing by 1000

    return meters
}
```

Figure 12: Pseudocode to Read and Convert Data

3.3 Ultrasonic ROS Package

Before collecting the distance data, it was necessary to build a ROS package that contained all of the python and Arduino files called “sensorLoop”. This specific package provides the means to run several scripts and simulations in parallel. Furthermore, each ultrasonic sensor required its own Arduino script to collect the distance data. The Arduino script is the publisher node that transmits the data to the topic called “ultrasound.” Figure 13 shows how the publisher communicates with the subscriber through the topic.

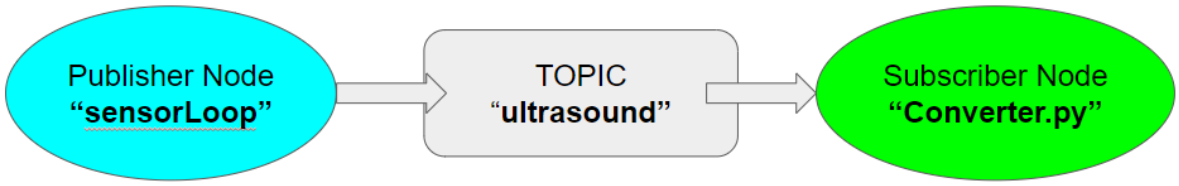


Figure 13: Publisher, Topic, and Subscriber Node diagram

As mentioned, ROS packages are libraries where the scripts, or nodes, are stored for organization and running nodes. As mentioned before, the Ridgeback has two LiDAR sensors: a front sensor and a rear sensor. The front LiDAR sensor was used as the robot moves forward, as it is important to know its environment in front. After creating the ultrasonic sensor system, there exist two independent sensor systems, which need an additional package to fuse the ultrasonic sensors functionality with that of the LiDAR sensors. Only then the robot will detect the multitude of obstacles in a building that has glass walls. Gaining inspiration from simulation tutorials, the robot uses a graphical interface in ROS that enables the user to visualize several messaging channels called Rviz [13]. Ultimately, Rviz acted to visualize both ultrasonic and LiDAR data from the selected topics.

3.4 LiDAR Sensor

The LiDAR sensor that Ridgeback is equipped with is the UST- Hoyuko LiDAR sensor, as shown in Figure 14. The sensor emits pulsed laser beams within a 270° field of view. When the emitted laser beams are reflected back from an object, its distance is measured by applying the TOF principle. The sensor has 1081 measurement steps with a 0.25° pitch. [2]

Figure 14: UST-Hokuyo LiDAR Sensor

3.5 LiDAR ROS Package

Ridgeback offers native ROS and Gazebo integration and is plug-and-play compatible with Clearpath's wide range of robot accessories. Within the Ridgeback package, using the channel to send the raw LiDAR sensor data as parameters for the Gmapping software (section 2.3), creates an image as shown in Figure 15.

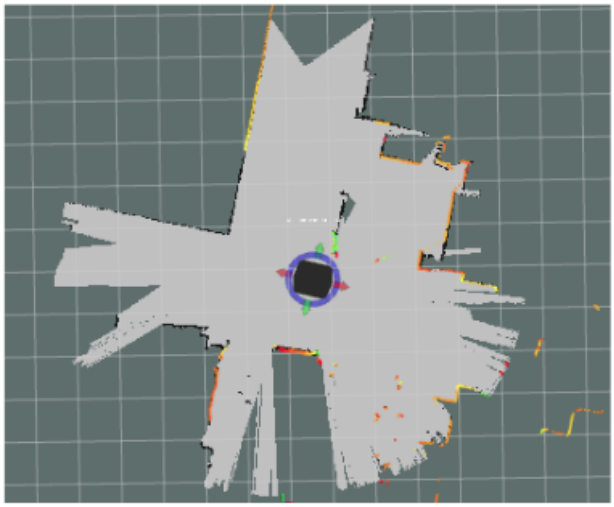


Figure 15: Gmapping Image

3.6 Gmapping with LiDAR and Ultrasonic Data

In order to modify the Gmapping package to effectively map with an additional ultrasonic sensor input, the first orderly process was to understand the functionality of the parameters and functions with the gmapping software. In ROS, a coordinate system is called a frame. Rviz needs to know which frame to use for transforming data from the coordinate frame to global reference frames. In 2D, the coordinates are x and y, while in 3D the coordinates are x, y, and z. Ridgeback has a predefined 3D model along with frames for every part, especially sensors. In Rviz, all

frames of the robot have a common frame of reference called the base frame, which represents the coordinate system of the robot's base. Each part is given a position and an orientation of the frame in relation to the base frame, or a transform. The data collected from the ultrasonic sensor is with respect to the frame of the ultrasonic sensor. However, in navigation, the robot needs to know where the objects are with respect to its base. Therefore, it is important to identify the transformation (${}^{B}_{U}T$) of the ultrasonic sensor frame (U) with respect to the robot base frame (B), as the data collected will contribute to making an accurate map with respect to the robot base frame; see Figure 16 for a visual. Therefore, by knowing the ${}^{B}_{U}T$, this system will compute the ultrasound data with respect to the robot base frame.

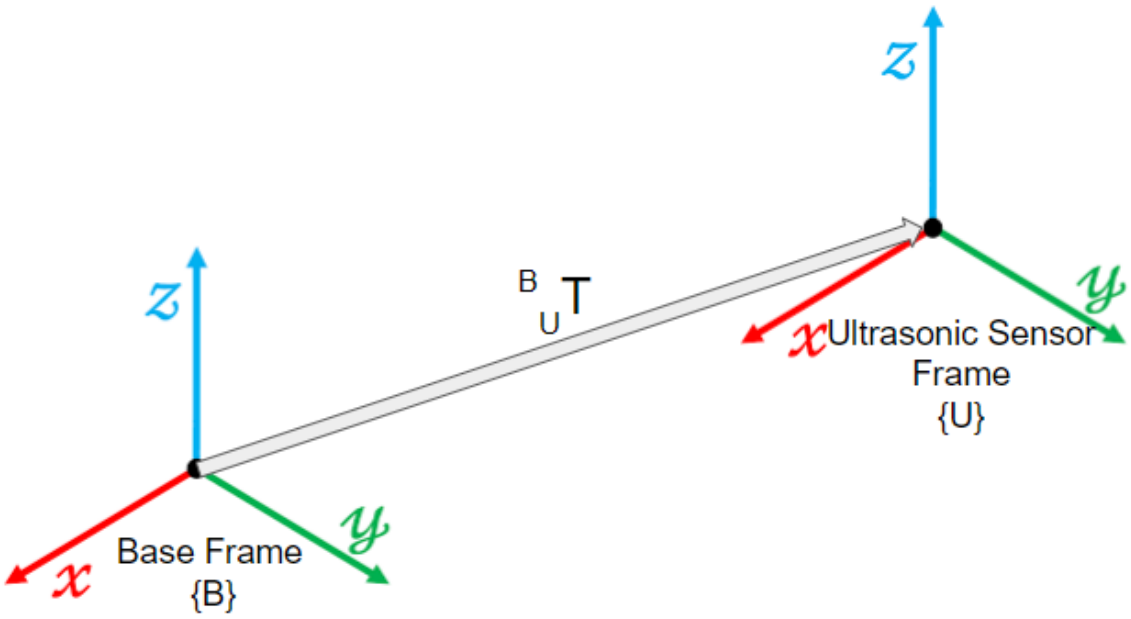


Figure 16: Ultrasonic Sensor Frame Transformation with respect to the Base Frame

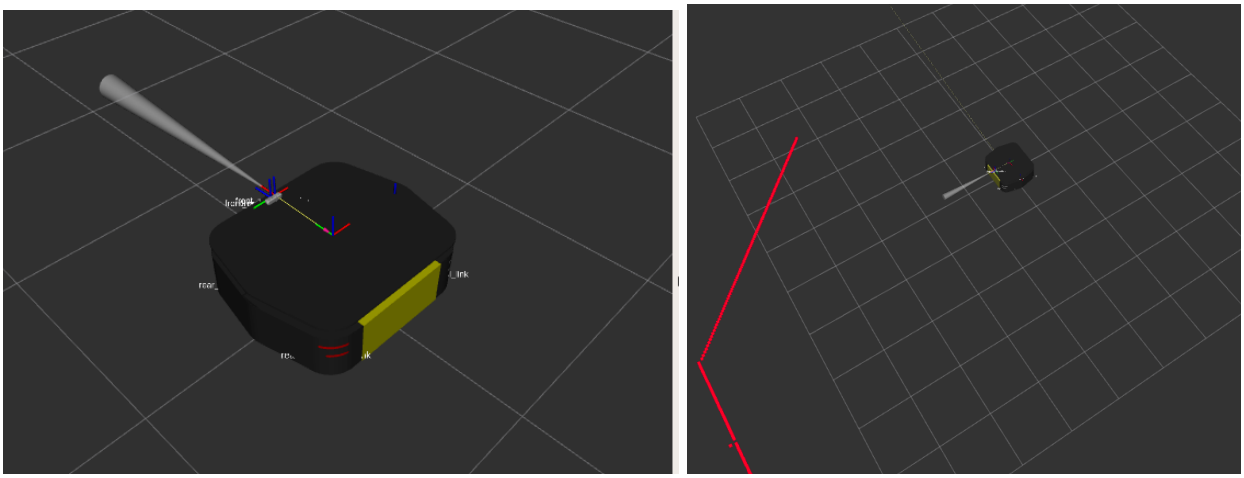


Figure 17: Rviz simulation: LiDAR and Ultrasonic Sensor Data Fusion (Zoomed In vs. Zoomed Out)

To achieve the calculation of the ${}^{W}_{U}T$, the Unified Robotic Description Format (URDF) is modified to add the transformation between the ultrasonic sensor frame and the robot base frame. The URDF is an XML(Extensible Markup Language) file format used in ROS to provide the robot description by describing all the elements of a robot. All packages are uploaded from the Clearpath website which enables simulating the robot and uploading the necessary attached objects on the robot [2].

A 3D model of a camera sensor was used since there was no available 3D model for the ultrasonic sensor. A model is necessary for the robot because it will need the frame of the robot and the model so the data collected by the sensor is accurate when recording the measured values. On the left image of Figure 17, the red, blue, and green lines represent the frames of each object. The frame farthest away is the frame of the ultrasonic sensor model. Furthermore, there is an obtrusive red line on the right image which represents the obstacle to be detected from the sensor input from simulation.

4 Experimental Results

4.1 LiDAR Sensor Results

This section aims to delve into the results of the data collected by the LiDAR scanner from a real-world test environment, by controlling the robot with a joystick. As seen in Figure 18, the Ridgeback operator is using a Playstation gaming system controller to traverse the Ridgeback around the hallway for data collection. During this process the robot goes no faster than 0.1m/s and the only person within a meters range of the robot is the operator themselves. There are two joysticks to control the robot - one being the pivot functionality of the robot addressing the omnidirectional capability of the robot and the other being the directional movement of the robot. The left image shows the pivot functionality in motion with the operator holding the controller, while the right image displays the direction functionality as the robot is moving forward.

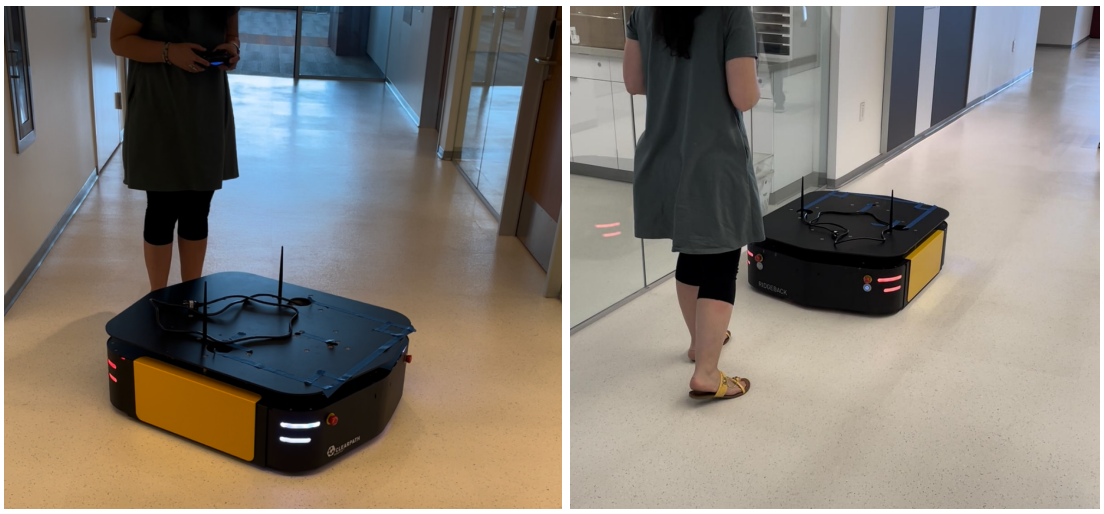


Figure 18: Rover Moving through Test Environment (4th Floor Hallway) while Controlled with a Joystick

As aforementioned, the Gmapping software takes in the data object from the LiDAR data acquisition and processing package. A visualization of this data object is logged, as shown in Figure 19. The communication data contains the angle increment, min angle, max angle, range min, range max, and time per scan. This produces two 1081x1 matrices: one for range data, and one for laser intensity.

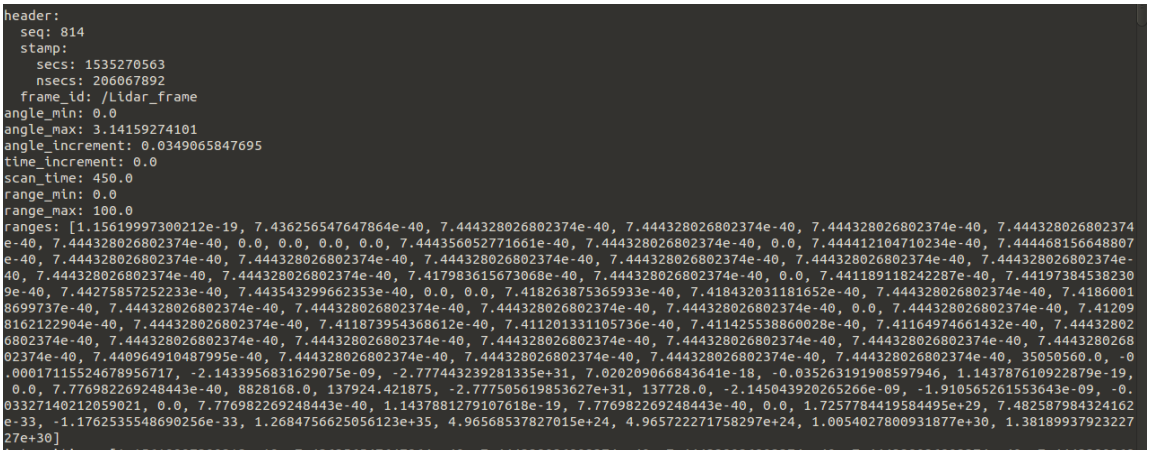


Figure 19: Laser Data Script Print Output

A direct screenshot of the parameters comments from the LiDAR scan script file which include definitions for each variable is shown in Figure 20. The most notable parameters are angle min/max, time increment, and ranges. The angle min and max are the ranges from which the sensor will receive the pulse ranges from with the min angle origin being (90° or π/2 rad) assuming that the robot is facing North. Next, the time measurements are the time taken between each of the measurements, which will correspond with the localization and odometry with respect to the measurements taken. Finally, the ranges are the raw data measurements from the LiDAR pulse inputs. The respective parameters are fed into the Gmapping package which will appropriately construct and modify the 2D map.

```
float32 angle_min        # start angle of the scan [rad]
float32 angle_max        # end angle of the scan [rad]
float32 angle_increment  # angular distance between measurements [rad]

float32 time_increment   # time between measurements [seconds] - if your scann
er
                         # is moving, this will be used in interpolating posit
ion
                         # of 3d points
float32 scan_time        # time between scans [seconds]

float32 range_min        # minimum range value [m]
float32 range_max        # maximum range value [m]

float32[] ranges         # range data [m] (Note: values < range_min or > range
_max should be discarded)
float32[] intensities    # intensity data [device-specific units].  If your
                         # device does not provide intensities, please leave
                         # the array empty.
```

Figure 20: Definition of the Laser Scan Parameters

The preconceived understanding of the LiDAR not being able to detect the glass wall was no different from the recordings. The Gmapping software displayed what was simply past the wall as if the glass wall was not there at all. Figure 20 and Figure 21 show the Ridgeback placed approximately at the same position within the test environment from which the readings are being taken from. When observing Figure 20, the visible glass obstacles/boundaries are the immediate close glass wall and the glass cubicle further away on the immediate corner on the right side of the hallway. When comparing the visual patterns from the real-world environment and the 2D map, the further cubicle is still registered as unmarked territory as that region is still grayish blue. While the panel closest to the picture is not detected as a boundary and therefore reading all the empty space in the room as unmarked territory. Furthermore, the door is equated as the unmarked territory in the shape of a triangle at the bottom left quadrant of the picture as the door. The reason that it is shaped as a triangle is because of the angle of viewable/markable territory from the perspective of the sensor from that position.

Figure 20: Picture of Ridgeback at ~Position(X,Y)

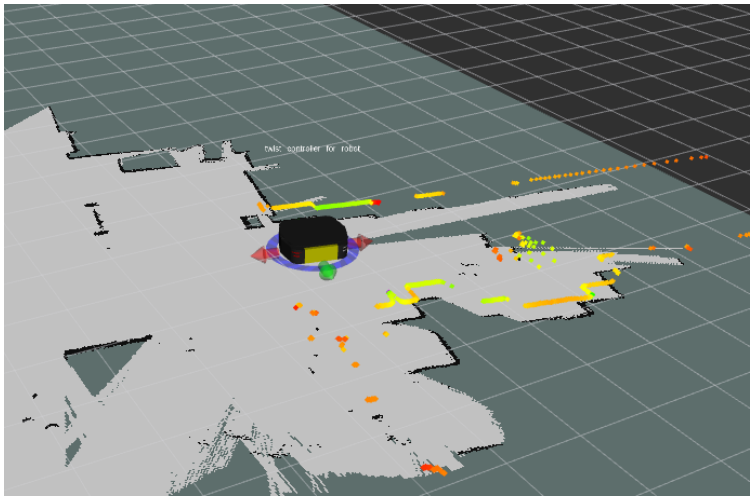


Figure 21: Actual Snapshot of LiDAR 2D Mapping at ~Position(X,Y)

Simply put, there is quantitative data from the experiment readings that render the use of LiDAR sensors useless when trying to register glass obstacles. This section aims to visualize the inadequacies of the LiDAR sensors and will go into more depth in the performance section of the report.

4.2 Ultrasonic Sensor Results

Testing the sensors in various environments is necessary because the sensor will possibly undergo changes through factor(s) not previously considered while testing the sensor in the engineering building, the main environment. The measured data collected from the sensors and compare the sensor performances in two environments in the engineering building: Room 4029 and the Balcony is discussed in this section.

Figure 22: SCDI Room 4029

As shown in Figure 22, the experimental setup in SCDI Room 4029 is as follows: A wheeled cabinet is used as a simulated robot and a measuring tape (pink color) is positioned on the ground to measure the distance between the simulated robot and the glass wall. The ultrasonic sensor is positioned securely on top of the simulated robot and points towards the glass wall. The simulated robot is moved to several distances (e.g. 25 cm, 50, 100, and so on) and the ultrasonic sensor data is recorded. The same procedure is followed for both ultrasonic sensors.

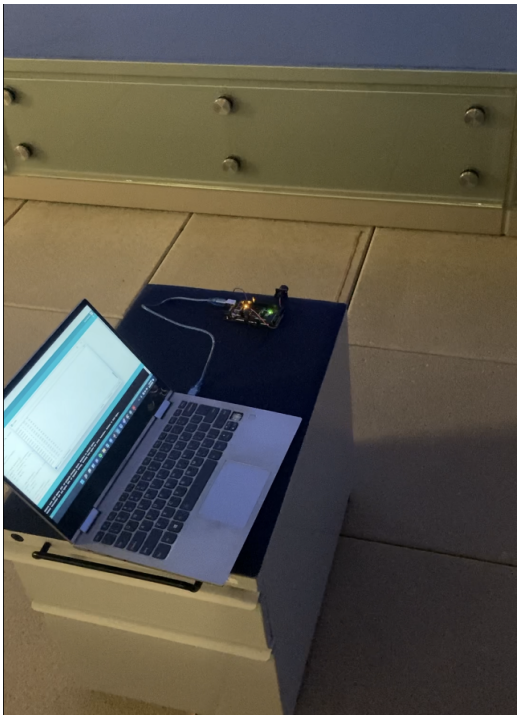


Figure 23: SCDI Balcony

As shown in Figure 23, a similar experimental setup was constructed as that of the cart, which serves as the robot simulator, and therefore followed the same experimental measurement procedure from the same experiment performed in the Room 4029. The only difference is that the balcony experiment is conducted at the balcony of the fourth floor.

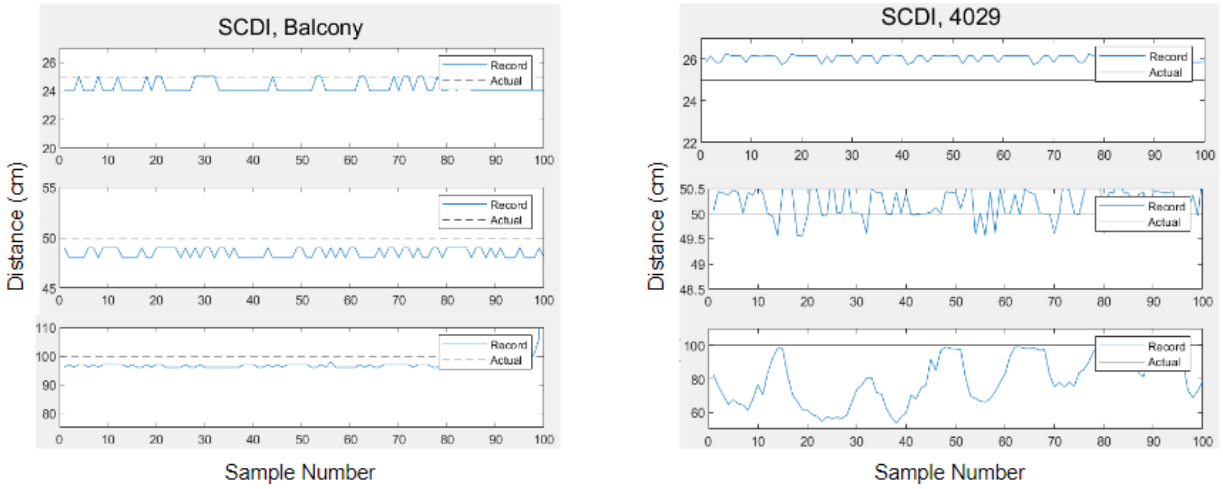


Figure 24: Distance vs Sample Number with Ping Sensor

Figure 24 shows the data collected by the Ping sensor in the Balcony and in Room 4029 of the engineering building, SCDI, at three different distances: 25 cm, 50 cm and 100 cm. The blue lines represent the recorded values while the black dashed, or black solid lines, represent the actual distance measured by the measuring tape. When the sensor is really close to the glass wall in both environments, the recorded values are consistent and are very close to the actual measured value. When the sensor is far away from the glass wall, the measurements are less accurate and it underestimates the expected value. In the Balcony, the sensor values are mostly consistent and they are very close to the actual value measured. On the contrary, the sensor values in Room 4029 underestimate the expected measurement and oscillate greatly.

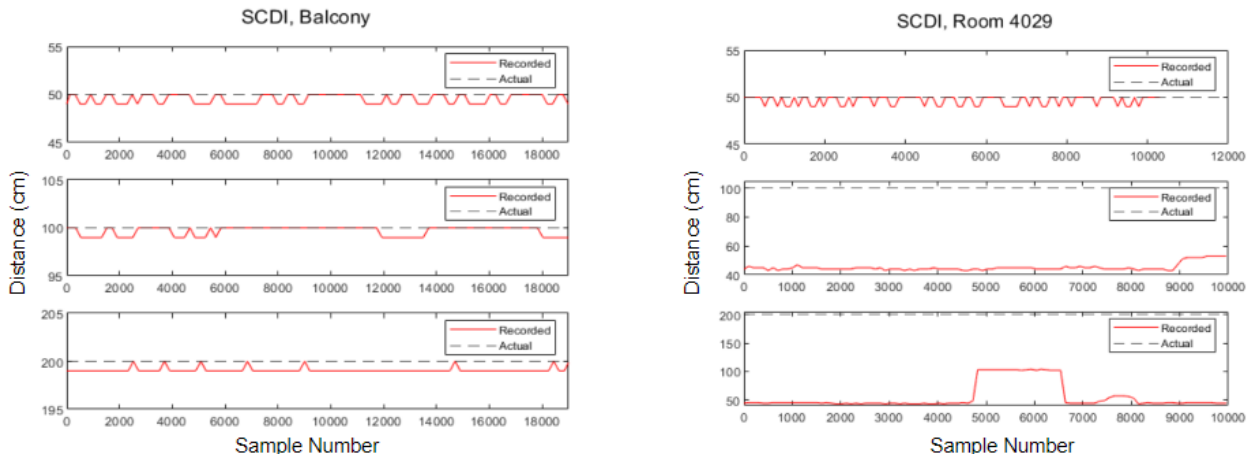


Figure 25: Distance vs Sample Number with Maxbotix Sensor

Figure 25 the data collected by the MaxBotix sensor in the Balcony and in Room 4029 of the engineering building, SCDI, at three different distances: 50 cm, 100 cm and 200 cm. The red lines represent the recorded values while the dashed lines represent the actual distance. When the sensor is really close to the glass wall in both environments, the recorded values are consistent and are very close to the actual measured value. When the sensor is far away from the glass wall, the measurements are less accurate and it underestimates the expected value. In the Balcony, the sensor values are mostly consistent and they are very close to the actual value measured. On the contrary, the sensor values in Room 4029 underestimate the expected measurement and the values are nowhere close to the expected values.

After comparing the data, an important observation from the findings is that the data collected from ultrasonic sensors are affected at long distances while inside the engineering building. This indicates that there must be a device within the engineering interfering with the ultrasonic sensors. The device interfering with the ultrasonic sensor is the motion detector because motion detectors emit ultrasonic waves [25]. With another sensor emitting ultrasonic waves, both the ping sensor and MaxBotix data collection will be interrupted, thus providing the wrong measurements. Figure 26 shows a picture of the motion detector.

Figure 26: Dual Technology Ceiling Mount Occupancy Sensor

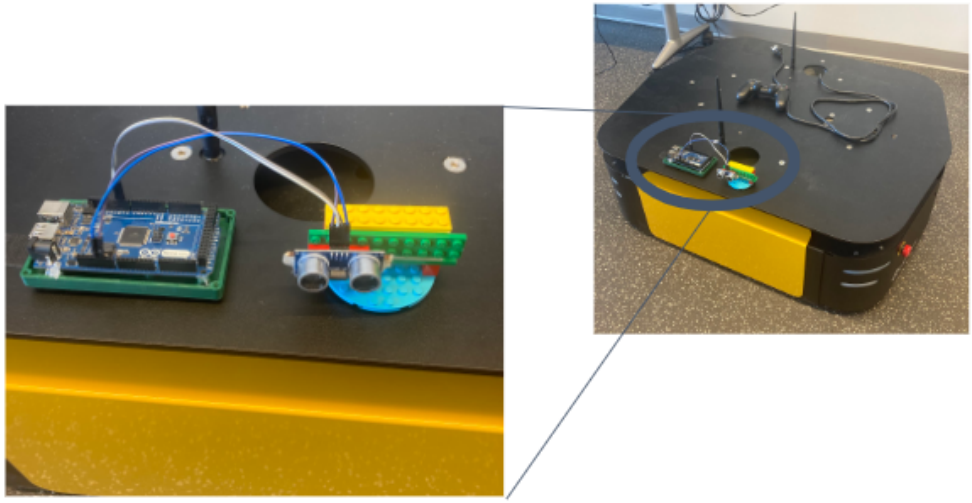


Figure 27: Ping Sensor Attached to Ridgeback

After comparing the data collected from the two environments, the sensor best fit for the project is the ping sensor. The ping sensor is the best option because it operates within a greater range of low distances compared to the MaxBotix sensor. Within the engineering building, the MaxBotix sensor operates properly between 2cm and 50cm, while the MaxBotix sensor functions between 30cm to 50cm. Figure 27 shows the lego mount so the ultrasonic sensor easily attaches and detaches to the robot.

4.3 Gmapping with LiDAR and Ultrasonic Sensor Results

For the Ultrasonic sensor system, the Arduino file used to acquire the raw data from the sensor was implemented into a ROS node which publishes the preprocessed sensor data into the ROS environment. As the rosserial ROS package uses Arudino's UART communication directly and asynchronously, the board is converted to a ROS node. Through this node, it sends a message called range_msg which is the distance from the sensor system to the obstacles detected. Due to the low computing power on board, the values were unable to be checked in the Arduino workspace, and were only visualized in the ROS workspace which made debugging unorganized. To visualize the data, we used a visual tool on Rviz called Range, which shows the range measurements. The visualization of the ultrasonic readings are depicted as the gray cone while the red line represents the data collected from the LiDAR sensor in the left image on Figure 28. The size and length of the cone changes in real-time depending on the readings from the ultrasonic sensor. Each range measurement is a cone starting at the origin of the frame given in the header of the topic [39]. Conversely, the right image is the Ridgeback robot simulation in Gazebo which tested the actual functionality of the sensors.

The last step to create a final product was effectively combining the respective systems to correspond with the Gmapping system in Gazebo. While analyzing the simulation, it was important to note that the data collected from the sensors are from two different environments. The data collected from the ultrasonic sensor is from the real world environment while the laser scanner collects the data from the gazebo, as shown in Figure 28.

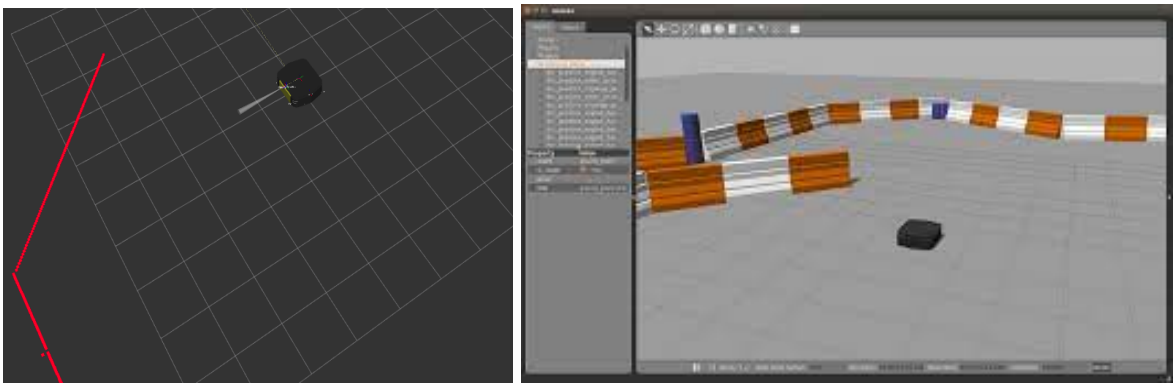


Figure 28: Laser Scanner combined with Sonar Range in Rviz (Left) and Gazebo (Right)

When reverse engineering the Gmapping software, an important discovery made was that gmapping only takes an object of laser_scan_msg which inherently excludes any other messages including sonar messages. Being the case, we decided to convert the readings from the ultrasonic sensor node to output an object of laser_scan_msg to the Gmapping software concurrently. So, the final method of procedure was to modify the actual content of the laser_scan_msg parameters to correspond with that of the sonar message when fed into the gmapping as shown in Figure 19 [4]. The appropriate modifications required understanding the functionality of both sensors and how the readings from the sensors provide useful data to create a map. The first modification was changing the angle min and max to a range much narrower than that of the LiDAR as the sonar message only takes in a single point. Next, the 90 points array in the parameter, ranges, are set as duplicates of the same reading. Then, there is no intensity feature regarding the ultrasonic sensor as it only detects an obstacle or free space and therefore correspondingly sets it as 0 or 1. Finally, the scan time between the readings for the ultrasonic sensor was raised by a magnitude of 10 as the bandwidth in terms of computing power and latency is much more limited in comparison to the LiDAR sensor. After making all the appropriate modifications, we were able to effectively send the ROS node in the object of a laser_scan_msg to the Gmapping software which was a step in the right direction, but soon found that the Gmapping software can only take in one message at a time. Being the case, we had two options: to integrate the two messages into one or create two separate maps and layer them on top of each other. We decided to go with the latter as it seemed to be the simplest solution.

5 Deliverable Analysis and Conclusion

5.1 Performance

The addition of ultrasonic sensors to a mobile robot proved to have its challenges. The fact that other devices in the robot's environment, like the Dual Technology Ceiling Mount Occupancy Sensor in Figure 25, use ultrasonic sound waves of similar frequencies leading to less accurate measurements limits the reliability of this solution. If the devices that operate with similar frequencies were not present in the robot's environment, the ultrasonic sensors will collect accurate data and it can utilize its operating range of distance values. Therefore, even if the integration worked as intended, the data from the ultrasonic sensor itself was unreliable at ranges beyond 50 cm inside SCDI due to the motion detectors interfering with the ultrasonic sensor measurements. This issue is similar to what was reported by Wei *et al.* [18]. The authors use ultrasonic sensor fusion and they discover that "*if the robot is far from the glass during the map construction, the glass detection algorithm will fail*" [18]. Furthermore, compared to the LiDAR sensor the sampling rate is much slower and beam angle is different than that of the ultrasonic sensor. Since the sampling rate is slower, and in one direction compared to LiDAR's 270 degree range, the robot will take longer to map the area using both sensors, because it has to follow along the walls in order to map it.

The way the sensor mount is effective and easily detachable due to the velcro and legos. However, the sensor is easily movable and exposed; this will lead to errors when measuring the distance between the sensor and glass wall. Another risk with the sensor being exposed and moveable, someone walking along could easily tamper with the sensor or the wiring if the robot was sitting out.

5.2 Sustainability Consideration

Power consumption of the ping sensor compared to the power consumed using other existing solutions was considered for sustainability. For example, using additional software with the LiDAR sensors to predict glass and mirror locations uses extra energy and storage. When considering power consumption, one alternative to using ultrasonic sensors is using Bluetooth ranging technology. Bluetooth ranging is useful for indoor navigation and it is an

asset for localization and tracking. Currently, Bluetooth ranging systems achieve at most a 2 meter accuracy [36]. Not only do these systems reduce the power, but the range data is fairly accurate compared to ultrasonic sensors. However, implementing bluetooth ranging for 2D map creation is difficult because Bluetooth requires a beacon, or a bluetooth radio transmitter to interact with another wireless beacon, and the measurement is from that beacon to another [37]. This means it is necessary to place multiple antennas/beacons on the glass walls around the glass walls when creating the map. Though power consumption will be reduced, implementing this for glass detection will be difficult because its localization depends on multiple beacons [38].

Regardless, having a precise map will enable the robot to select an optimal path, as well as automating the monotonous task that it is given such as package transportation. Optimal path implementation is viable in a fairly stable indoor environment like the SCDI building, when an algorithm pre-plans the trip. By selecting the optimal path, the robot will minimize the path length and the energy consumption as it traverses from one point to another. Based on previously detected idle obstacles, the algorithm will ensure that the path taken is efficient for power consumption because there is less time taken and path traveled. Conversely, if the robot was given a point of destination relative to its current point without a path planned, it needs to wander around potential obstacles and find potential routes to the destination which takes much more time and distance traveled.

5.3 Reliability Expectations

While comparing the data in various settings, it is clear that the reliability of data fusion depends on the ultrasonic sensor which is subject to the medium of the environment. When the sensors capture data outside of the engineering building and there are not any outside interferences, the ultrasound is reliable. On the contrary, when the sensors collect data in the inside of the engineering building, the ultrasound is not reliable. The main cause of the interference is the motion detector because it operates in frequency spectrum in ultrasound range, at least 20kHz [25]. Because the ultrasonic sensors operate in the same range, this affected the measurements at higher distances, thus making the ultrasound unreliable. The data fusion relies on the accuracy of the sensors that are used for the fusion. Therefore, if one of the sensors provides inaccurate measurements, the fusion will not provide an accurate estimation.

5.4 Future Work

This work has provided several routes of possible research and it is the hope that some of these areas will be explored by future researchers and engineers. For instance, the fusion between laser scanner data and ultrasonic data is a promising approach for detecting the glass accurately. Gmapping package will require to be updated in order to support ultrasonic data. Additional real-world tests will be required to ensure the accuracy of the data fusion algorithms.

The issue of low range and signal interference may be circumvented with an ultrasonic sensor with a signal with a larger amplitude and smaller beam angle. Moreover, the benefit of implementing an inhouse ultrasonic system versus a plug and play ultrasonic sensor was the customizability and option for manipulation of sensor message parameters to mesh with pre-existing software. For example, as seen in Figure 28, the LiDAR sensor data acquisition software was used for the ultrasonic sensor as well by taking into consideration the relative difference between the two sensors. When comparing the angle width and number of pulse inputs, the LiDAR has a much wider range up to 270 degrees with 90 pulses for one reading, while the angle of the ultrasonic sensors is much narrower with only one pulse. The issues with sampling speed have the potential to be modified with predictive algorithms. Regardless, these are interesting electrical and robotics engineering problems to solve that could advance robotic technology.

5.5 Professional Constraints

Throughout the project, there were numerous constraints to take into consideration. The first constraint was the cost of parts because one of our project goals states to design a system that can be affordable for typical indoor robot users. The ultrasonic sensors we purchased which allowed us to develop an affordable proposed system. Furthermore, it is much easier to implement an off-the-shelf sensor system rather than designing a custom sensor.
Next, the time constraints made it seemingly difficult to ever be ahead during our project process. Before working with the robot, the members of the team had to learn a sufficient amount of ROS i.e. run simulations with the robot. ROS was learned via online resources and courses at Santa Clara University. Throughout the project, we learned how to collect data from the sensors via ROS, how to create ROS packages, and properly simulate the data on Rviz with the robot model. This project used standard communication protocol within ROS. The lead time

for delivery of the physical robot only allowed for on-system experimentation and development during the last quarter of the academic year. Third, this project prioritized safety as it is a requirement when robots move within human environments. As mentioned before, cameras are typically implemented into robots to map environments. At the beginning of the project, the team investigated several sensor technologies. Taking into account that the robot requires to navigate into a human environment, privacy was a big concern. Therefore, the team decided to exclude cameras to ensure that the system does not violate human privacy. Moreover, the LiDAR works sufficiently for the purposes of this project. Furthermore, the proper pre-emptive safety measures were taken to ensure no hazards when deploying the robot in a real-life environment. The area was cleared of people before running real world experiments. Then, the robot was manually controlled and its speed was limited to a slow pace no faster than 0.1 m/s. Additionally, the robot is equipped with 4 emergency stops and the human operator could easily press in case of an emergency. By minimizing the speed of the robot and clearing the space, the risk of injury to bystanders or robot damage is minimized.

5.6 Summary and Conclusions

This thesis presents a methodology to combine both data from the LiDAR sensor and the ultrasonic sensor. Different sensors were explored by creating experiment protocols to test and evaluate the reliability of the sensors. Our experiments between the Maxbotix sensor and the ping sensor showed that smaller beam angles are more accurate and tend to be cheaper to produce. While comparing the recorded data from the sensors to the ground truth data in each scenario, there was a trend for the ultrasonic data inside the engineering rooms to be less accurate, especially at large distances. This was due to an interfering signal from a motion sensor on the ceiling of the room. Most likely the smaller beam angle produces a signal of higher concentration.

A ROS package which captures the data from the ultrasonic sensors and publishes the data to the topic was created. A URDF model of the sensor to the robot in Rviz that allows the data to be sensed from the robot itself was created. With the data from the topic, the robot was able to visualize the data produced from the sensor in Rviz as range data. This proposed system is effective in environments with interference as long as it is constrained at close distances around 50 cm, and a sensor with a stronger ultrasonic signal increases this range of

effectiveness in the presence of other ultrasonic sensors. Though the 2D Map from the LiDAR data and the ultrasonic data was not generated, a multisensory system that addresses the weaknesses of other sensors was developed that collects the data from the LiDAR sensor and the ultrasonic sensor, and has the data in the same ROS environment. In addition, the future work that can come from our multisensory system will benefit the future researchers and engineers in a variety of robotic applications.

Along the journey of this project, the team members gained a deep understanding of ROS about publisher nodes, subscriber nodes, and topics from no prior knowledge at all. Furthermore, applying research skills, programming in Python and Arduino, and designing experimental protocols were all crucial as we progressed in our concentrated areas in electrical engineering. The team also obtained a deeper knowledge of key concepts in robotics, such as transformations and robot motion. In addition, we attained a better understanding of project management, team coordination, time management, technical writing, and public speaking on technical topics.

References

[1] *A Model-based Approach to Acoustic Reflector Localization with a Robotic Platform.* (2020, October 25). Retrieved from IEEExplore: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9341437

[2] *Clearpath Robotics*. (2015). Retrieved January 7, 2022

[3] *Creating Map using Laser Scanner and Gmapping*. (2018). Retrieved from KiranPalla: https://kiranpalla.com/autonomous-navigation-ros-differential-drive-robot-simulation/creating-map-using-laser-scanner-and-gmapping/.

[4] *gmapping.* (n.d.). Retrieved from ROS wiki: http://wiki.ros.org/gmapping

[5] Hopley, P. (2019, April 23). *Rodney - A Long Time Coming Autonomous Robot (Part 7)*. Retrieved from CODE PROJECT:

https://www.codeproject.com/Articles/2465567/Rodney-A-Long-Time-Coming-Autonomous-Robot-Part-8

[6] *How Does an Ultrasonic Sensor Work?* (2022, May 27). Retrieved from Teach Engineering: https://www.teachengineering.org/lessons/view/umo_sensorswork_lesson06

[7] *How to Use an Ultrasonic Sensor with Arduino [With Code Examples]*. (2021). Retrieved from MaxBotix: https://www.maxbotix.com/Arduino-Ultrasonic-Sensors-085/

[8] *MaxBotix.* (2005). Retrieved from HRLV-MaxSonar - EZ Series: https://www.maxbotix.com/documents/HRLV-MaxSonar-EZ_Datasheet.pdf

[9] *MaxBotix*. (2021). Retrieved from MB1013 HRLV-MaxSonar-EZ1: https://www.maxbotix.com/ultrasonic_sensors/mb1013.htm

[10] *Multi-sensor Fusion Glass Detection for Robot Navigation and Mapping.* (2018, August 16). Retrieved from IEEExplore: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8584213&tag=1

[11] *Robot Operating System (ROS)*. (2022). Retrieved from Springer Link: https://link.springer.com/book/10.1007/978-3-319-26054-9?noAccess=true

[12] *ROS - Data display with Rviz*. (2021). Retrieved from Stereo Labs: https://www.stereolabs.com/docs/ros/rviz/

[13] *rviz*. (n.d.). Retrieved from ROS wiki: http://wiki.ros.org/rviz

[14] *Ultrasonic Arduino-to-Arduino Communication*. (n.d.). Retrieved from ARDUINO: https://blog.arduino.cc/2018/06/01/ultrasonic-arduino-to-arduino-communication/

[15] *Understanding How Ultrasonic Sensors Work*. (2021). Retrieved from MaxBotix: https://www.maxbotix.com/articles/how-ultrasonic-sensors-work.htm?utm_term=&utm_source=google&utm_medium=cpc&gclid=Cj0KCQiA09eQBhCxARIsAAYRiymgRAq1D3ydGfHOw6s-iyJNgdmmwRLoErbVtvY5ObOTbXECXzvpb5QaAuYuEALw_wcB.

[16] Liu, X. (2008). Airborne LiDAR for DEM generation: some critical issues. Progress in Physical Geography: Earth and Environment, 32(1), 31–49. https://doi.org/10.1177/0309133308089496

[17] *Gmapping*. (n.d.). Retrieved from ROS wiki: http://wiki.ros.org/gmapping

[18] H. Wei, X. Li, Y. Shi, B. You and Y. Xu, "Multi-sensor Fusion Glass Detection for Robot Navigation and Mapping," 2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA), 2018, pp. 184-188, doi: 10.1109/WRC-SARA.2018.8584213.

[19] *A framework for spatial map generation using acoustic echoes for robotic platforms*. (2022). Retrieved from ScienceDirect: https://www.sciencedirect.com/science/article/pii/S0921889021002633

[20] *Radar and LiDAR Comparison*. (n.d.). Retrieved from Resource Explorer: https://dev.ti.com/tirex/explore/node?node=AN87cVrGWlIylnt4nH.UWg__VLyFKFf__LATEST

[21] Deberunne, C., & Vivet, D. (2020, April 7th). *A Review of Visual-LiDAR Fusion based.* Retrieved from sensors: https://mdpi-res.com/d_attachment/sensors/sensors-20-02068/article_deploy/sensors-20-02068.pdf?version=1586254962

[22] OMRON. (2007). Retrieved from Ultrasonic Sensors: https://www.ia.omron.com/support/guide/50/introduction.html#:~:text=Unlike%20photoelectric%20sensors%2C%20Ultrasonic%20Sensors,detected%20with%20the%20same%20settings.

[23] *Speed of Sound*. (n.d.). Retrieved from sfu: https://www.sfu.ca/sonic-studio-webdav/handbook/Speed__Of_Sound.html#:~:text=In%20air%2C%20for%20instance%2C%20temperature,sec%20or%20344%20m%2Fsec

[24] *Time-of-flight: what you need to know about these new means of computer vision*. (n.d.). Retrieved from avsystem: https://www.avsystem.com/blog/time-of-flight/#:~:text=The%20time%2Dof%2Dflight%20principle,that%20object%20from%20the%20source.

[25] *Motion Detectors or Motion Sensors*. (n.d.). Retrieved from engineers garage: https://www.engineersgarage.com/motion-detectors-or-motion-sensors/

[26] *Time-of-Flight principle Example*. (n.d.). Retrieved from terabee: https://www.terabee.com/time-of-flight-principle/

[27] *Hokuyo UST-10LX Scanning Laser Rangefinder with Ethernet*. (n.d.). Retrieved from acroname: https://acroname.com/store/scanning-laser-rangefinder-ethernet-r359-ust-10lx

[28]M. Awais, "Improved laser-based navigation for mobile robots," 2009 International Conference on Advanced Robotics, 2009, pp. 1-6.

[29] Yousif, K., Bab-Hadiashar, A. & Hoseinnezhad, R. An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics. *Intell Ind Syst* 1, 289–311 (2015). https://doi.org/10.1007/s40903-015-0032-7

[30] automaticaddison.CategoriesRobotics. October 26, 2019 https://automaticaddison.com/create-a-hello-world-project-in-ros/

[31] Tibebu, Haileleol, et al. “LiDAR-Based Glass Detection for Improved Occupancy Grid Mapping.” Sensors, vol. 21, no. 7, Mar. 2021, p. 2263. Crossref, https://doi.org/10.3390/s21072263.

[32] U. Saqib and J. Rindom Jensen, "A Model-based Approach to Acoustic Reflector Localization with a Robotic Platform," 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 4499-4504, doi: 10.1109/IROS45743.2020.9341437.

[33] Shao-Wen Yang and Chieh-Chih Wang, "Dealing with laser scanner failure: Mirrors and windows," 2008 IEEE International Conference on Robotics and Automation, 2008, pp. 3009-3015, doi: 10.1109/ROBOT.2008.4543667.

[34] H. Tibebu, J. Roche, V. De Silva, and A. Kondoz, “LiDAR-Based Glass Detection for Improved Occupancy Grid Mapping,” Sensors, vol. 21, no. 7, p. 2263, Mar. 2021, doi: 10.3390/s21072263.

[35] J. R. Jensen and U. Saqib, “A framework for spatial map generation using acoustic echoes for robotic platforms,” https://www.sciencedirect.com/science/article/pii/S0921889021002633 , 10-Jan-2022

[36]imec, “Accurate and secure distance measurement with bluetooth” https://www.imec-int.com/drupal/sites/default/files/2018-11/Accurate%20and%20secure%20Distance%20Measurement%20with%20Bluetooth.pdf

[37] Beaconstac. (n.d.). *What is a bluetooth beacon? how do ble beacons work?* Beaconstac. Retrieved June 12, 2022, from https://www.beaconstac.com/what-is-a-bluetooth-beacon

[38] *ACCURATE & SECURE DISTANCE MEASUREMENT WITH BLUETOOTH*. imec. (n.d.). Retrieved from https://www.imec-int.com/drupal/sites/default/files/2018-11/Accurate%20and%20secure%20Distance%20Measurement%20with%20Bluetooth.pdf

[39] Wiki. ros.org. (n.d.). Retrieved June 13, 2022, from http://wiki.ros.org/rviz/DisplayTypes/Range