ROOMSCALE 3D LIDAR SOLUTION

By

Cheng Xu

Terence Lee

Xizheng Fang

Final Report for ECE 445, Senior Design, Spring 2021

TA: Andrew Chen

1 May 2021

Project No. 7

Abstract

Our 3D Light Detection and Ranging (LiDAR) scanner scans its immediate surroundings capturing a large number of distance measurements. The collected data will be sent to our python program and used to generate a point cloud file. This file serves as a digital visualization of the scanned area which has numerous applications including target recognition, artificial intelligence and geographical surveying.

Contents

1. Introduction

1.1 Problem & Solution Overview

Since the introduction of the autopilot feature by Tesla in 2016, there has been an explosive interest in autonomous driving. Many companies have since spent large amounts of resources and manpower in this profitable field, especially in the research of 3D Light Detection and Ranging (LiDAR) sensors, one of the more vital aspects of self-driving vehicles [1] (along with countless applications in surveying, movie special effects, sim racing, etc.) This device is utilized to collect the data of a vehicle's surroundings and output an accurate 3D point cloud map so the vehicle can avoid obstacles accordingly [2]. Because of the growth in popularity of autonomous vehicles, the interest in 3D LiDAR sensors has seen a similar increase among enthusiasts and hobbyists. However, unlike big corporations that could obtain industry-grade LiDAR technology for years, individuals typically cannot afford the high cost of entry for a 3D LiDAR sensor. To make it easier for hobbyists to explore the self-driving vehicle field, we are developing a system that will provide similar functionalities as the current 3D LiDAR industry solutions utilizing 1D time-of-flight (ToF) sensors while also remaining both affordable and easily obtainable.

The system we built is aimed towards the hobbyist market which has been largely neglected by the market up until now. We plan for the solution's friendly cost of entry to be our main selling point. Instead of using typical single-phase or solid-state 3D LiDAR sensors which cost upwards of thousands of dollars, we will build the system based on a ToF sensor (also called 1D LiDAR) that can be a hundredth of the cost. The LiDAR will scan the environment and perform distance measurements on a 2-axis platform with a stepper motor helping to rotate the LiDAR 360-degree on the platform. Meanwhile, another motor will pitch the sensor up and down so the sensor can scan its environments at various height levels. The recorded spherical coordinates of the measured area will then be converted into Cartesian 3 coordinates and eventually exported as a point cloud file that can be visualized to show the scanned room.

1.2 High-Level Requirements

- The system can scan the room and output successfully within a time frame of 70 seconds
- The point cloud output by the system will have an accuracy of ±0.15 m (meters) at a distance of 3 m and ±0.06 m at a distance of 0.2 meters.
- The output file will use the industry-standard LASer (LAS) file format specified by the American Society for Photogrammetry and Remote Sensing (ASPRS) and be readable by most external software.

2. Design

2.1 Block Diagram

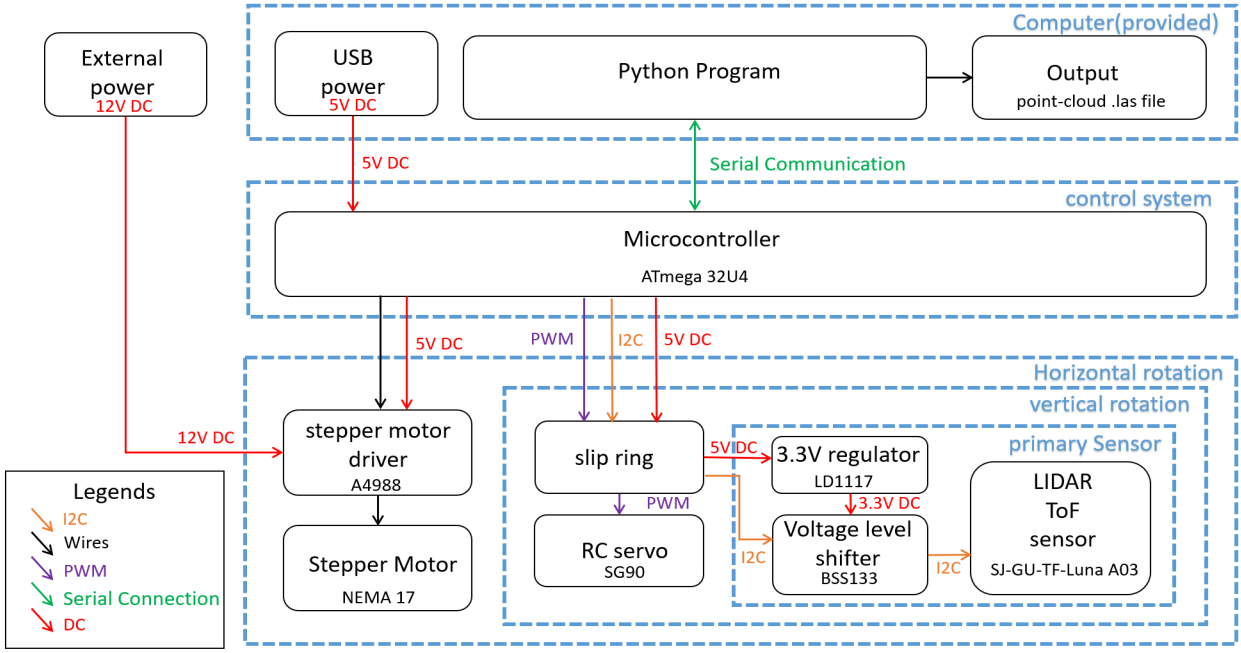

Figure 1: Block Diagram

Our 3D LiDAR system consists of five major subsystems: primary sensor, vertical rotation, horizontal rotation, control system, and computer. We also use a 12 V external power to power the stepper motor driver. The primary sensor subsystem is responsible for ensuring the ToF sensor readings are accurate. The vertical and horizontal rotation subsystems control the rotational movement of the ToF sensor in vertical and horizontal planes respectively. They work in tandem with the primary sensor subsystem to emulate a scanning effect similar to its industrial 3D LiDAR counterparts. Additionally, these three subsystems are managed by the control system allowing the sensor to scan an indoor room with an approximate dimension of 10x10x5 m in 70 seconds with an accuracy of ±0.1 m at a distance of 0.2 to 3 m. The sensor communicates with the microcontroller with the help of a voltage regulator and a bi-directional level shifter so the sensor can have its 3.3 V logic voltage to function normally. After the control system receives the measurement data from the sensor under the I^2C communication protocol, it will convert the 3D spherical coordinates to Cartesian coordinates then pass the processed data to the computer subsystem from which the python program will output a 3D point cloud file of the scanned environment.

2.2 Physical Design (See Appendix C)

Since our product is designed to be integrated into our clients' projects, we focused most of our effort on minimizing the physical footprint and simplifying everything down to the essentials. As shown in Figure 1 in Appendix C, our final design consists of only a base platform and a rotating sensor, with a total dimension of 113×102×64 mm. From the outside, the base platform has a wall power port for powering the device and communication between the device and the computer, a DC barrel jack for redundancy, and a 1/4-20 UNC thread at the bottom of the housing, on-axis to the rotation for easy mounting on standard tripods.

From the exploded view of our physical design (Figure 2 in Appendix C), we can see that we have conveniently divided the mechanical subassemblies in the same way as the block diagrams and kept every block separated and modular. Since our design revolved around a continuously rotating subassembly, a lot of elements needed to be fitted onto the axis of rotation, including a slip ring, the motor, the mechanical restraint, and the subassembly itself. We choose to use a GT2 timing belt along with two pulleys to offset the NEMA 11 stepper motor. The slip ring remains in the center of the rotation, with a larger bearing ensuring the accuracy and rigidity of the rotating assembly.

Because the vertical block is going to be rotating continuously, the footprint of this subassembly directly dictates the final dimension of our product. The LiDAR sensor pivots vertically while being driven by the micro-servo and we chose to "fold-up" the servo-sensor assembly using an o-ring as a short belt, cutting the diameter of the bounding cylinder in half while also minimizing the rotational inertia.

We have also determined that it is possible to fit both the microcontroller and the A4988 driver inside the base housing using a double-layered PCB with custom dimensions without increasing the footprint of our product.

2.3 Block Descriptions

Subsystem 1: Primary Sensor Block

The primary sensor subsystem consists of a ToF sensor. The subsystem references the environment through the sensor by taking 1D point-to-point distance measurements from the environment. It receives DC power and sends digital signals from and to the microcontroller subsystem through the slip ring. The supply voltage is kept between 3.7 to 5.2 V to ensure the sensor can function reliably, and we use a LD1117 voltage regulator and a BSS138 bi-directional level shifter to regulate the 5 V level voltage that comes from the microcontroller into 3.3 V for the sensor to make it function normally.

The sensor communicates with the microcontroller under the I^2C protocol and is responsible for sending the collected data to the control subsystem to be processed with the maximum transmission rate of 400 kbps. The connected wire starts from the sensor, passes through the slip ring, gets level shifted by the BSS138 shifter and finally goes to the microcontroller. With the help from both the vertical and horizontal rotation subsystems, the combined system can achieve a 3D scan of an indoor room (such as living rooms, classrooms, etc.)

Primary ToF Sensor (SJ-GU-TF-Luna A03) (See Figure 1 in Appendix F)

We use the SJ-GU-TF-Luna A03 LiDAR module as our ToF sensor. This specific sensor was chosen after researching the options available due to it having the smallest footprint and the lowest cost of entry while maintaining a maximum range of 20 meters and a maximum refresh rate of 250 Hertz [3]. The sensor has a 360-degree continuous rotation and takes distance measurements in every direction in a 2D plane with the help of the stepper motor. In addition, the RC servo pitches the ToF sensor up and down to ensure the sensor can reach various height levels allowing for a 3D environment scanning. The ToF sensor's frame rate is adjusted to within 5% of 200 Hz so it can fully scan the room in 70 seconds to leave enough time for the point cloud file generation. In addition, the sensor has a 2-degree field of view, it can be operated to take a 360-degree scan in a horizontal plane in 200 steps in 1 second. This step number is in accordance with the step number that the Nema 11 stepper motor can take to have a horizontal rotation in 1 second. Thus we can ensure that the sensor can work synchronously with the horizontal motor.

3.3 V Regulator (LD1117) & Bi-directional Level Shifter (BSS138)

Though the sensor needs a power supply from 3.7 to 5.2 V, it needs a logic voltage around 3.3 V to function properly. However, the microcontroller can only provide a 5 V power from the USB port. Therefore, we use a 3.3 V regulator, which is a LD1117 chip, and a bi-directional level shifter, which is a BSS138 chip, to achieve the smooth communication between the microcontroller and the sensor. The signals from the microcontroller are regulated and transmitted to the sensor while the signals from the sensor are also increased in voltage level and fed into the microcontroller.

Subsystem 2: Vertical Rotation Block

The vertical rotation block is in charge of pitching the sensor up and down so the sensor can scan the various horizontal planes at different heights. It consists of a slip ring and an RC servo. When the scan process begins, the RC servo pitches the LiDAR sensor up and down under the PWM control from the control block. Meanwhile, the control signal and the data collected by the sensor are transferred past the slip ring to the microcontroller in the control block. In addition, the servo receives a DC voltage ranging from 5 to 5.5 V from the microcontroller passing through the slip ring to keep it operating at the voltage within 10% of 4.8 V.

Slip Ring

The slip ring inside the block is connected to the ToF sensor and the RC servo in the primary sensor block on one side, and the microcontroller on the other. It makes sure the wire does not get twisted when the sensor is rotating so that the stepper motor in the horizontal rotation block can rotate infinitely to ensure the sensor's continuous scan. The slip ring acts to maintain the signal integrity of three components. One of the leads supplies 5 to 5.5 V DC power to the electronics within the vertical rotation and primary sensor block. The communication between the ToF sensor and the control system is run through the slip ring and is under the I^2C protocol. The PWM signal coming from the microcontroller into the RC servo also passes the slip ring.

RC Servo (See Figure 2 in Appendix F)

We use an SG90 9G Micro Servo as the RC servo in this subsystem. The servo, which is regulated by the control system through PWM, can pitch the sensor up to 180 degrees so that the ToF sensor can aim up and down. However, due to the limitation of the height of the sensor and the structure of the scanned environment, the servo usually only pitches a 60-degree range in the vertical plane to help the sensor to collect the required distance measurements. In addition, we use 20 steps in the vertical rotation, which means we have to angle across 3 degrees between each rotation for vertical rotations. Because the angular resolution of the LiDAR ToF sensor is 2 degrees, we keep the accuracy of the servo around 1 degree to guarantee accuracy and repeatability. The servo receives a 5 to 5.5 V DC power from the microcontroller and the power is within 10% of 4.8 V to ensure the servo functions normally.

Subsystem 3: Horizontal Rotation Block

The Horizontal Rotation block, which consists of a stepper motor driver and a stepper motor, rotates the sensor in the horizontal plane so the sensor can scan 360 degrees around its surroundings. It works with the vertical rotation block and the primary sensor block to make sure we can have a 360-degree 3D scan of an indoor room with an approximate size of 10x10x5 m in 70 seconds. The stepper motor driver operates at a voltage between 3.3 to 5 V. It receives DC power ranging from 5 to 5.5 V and the control signal from the microcontroller in the control block and also receives a 12 V power from a wall power. When the scanning process begins, it controls the stepper motor to rotate the system horizontally to ensure that the sensor can have 360-degree scanning of the horizontal plane.

Stepper Motor Driver & Stepper Motor (See Figure 3 in Appendix F)

We use a Nema 11 stepper motor as our stepper motor and an A4988 16 micro-stepping motor driver as our motor driver. As described by the A4988 schematic in Figure 4 in Appendix F, the motor driver receives commands into the STEP pin through wires from the microcontroller in the control system block. After it receives the commands, it operates the motor so the motor can rotate the sensor in the primary sensor block horizontally with a minimum step angle of 1.8 degrees. In addition, the motor is connected with the platform that holds an LiDAR sensor by timing belts and the sensor has a field of view of 2 degrees so it needs around 200 steps to have a scan with a reasonable resolution. Because the pulley for the platform has 60 teeth and the pulley for the stepper motor has only 20 teeth, in order to coordinate the platform RPM with the required steps for the sensor, the motor driver operates the motor at a step angle within 5% of 1.8 degrees and step 3 steps for each measurement. We choose to set our motor driver with a microstepping resolution of full step, thus we set three logic input pins MS1,MS2,MS3 to low. By implementing the motor driver as described above, the stepper motor can maintain enough accuracy to ensure the integrity of the data.

Subsystem 4: Control Block

The Microcontroller coordinates all the other subsystems so that the rotation of the vertical and horizontal axis is coherent, systematic, and repeatable enough for the ToF sensor to obtain meaningful measurements. It is also responsible for instructing the stepper motor driver to move horizontally to its next position (current position + 1.8°) after the ToF sensor receives a reading and pitching the sensor to the next vertical angle (current position + 6°) after a full 360° horizontal revolution. At the same time, the sensor is gathering data, the microcontroller sends collected data via a serial connection to the computer subsystem for coordinate conversion and LAS file generation.

Microcontroller (Hardware) (See Figure 5 in Appendix F)

We use an ATmega32U4 chip as our microcontroller. As shown in Figure 2 below, the microcontroller receives a 5 V power from an USB port. In the meantime, it powers the A4988 chip by this 5 V power source. In addition, it controls the pulses of the A4988 chip by sending signals to the STEP pin in the A4988 chip. After receiving the pulse limit from the microcontroller, the A4988 chip can control the Nema 11 stepper motor by the four output pins in the middle of the schematic in the figure below so we can ensure the motor can be operated at a step angle within 5% of 5.4 degrees. Therefore, the stepper motor can work synchronously with the sensor and the servo to ensure a scan cycle for at most 70 seconds. Also, since the sensor needs a 3.3 V logic voltage to function normally, we use a LD1117 voltage regulator, which is the component on the top left corner to regulate the 5 V signal from the microcontroller to the required 3.3 V. Additionally, in order to successfully communicate with the sensor, we use two signals from the microcontroller that run under I²C protocol; then we connect these signals and the regulated 3.3 V power to a BSS138 bi-directional logic shifter, which is the component on the left of the microcontroller in the schematic below; after that we connect the outputs of the level shifter to the ports that are connected with the sensor, which is the component on the down left of the schematic. Therefore the data coming from the sensor can be shifted up and sent to the microcontroller while the microcontroller can power and control the sensor properly. The PCB footprint in Figure 3 below also fits perfectly inside the base housing, saving physical space.

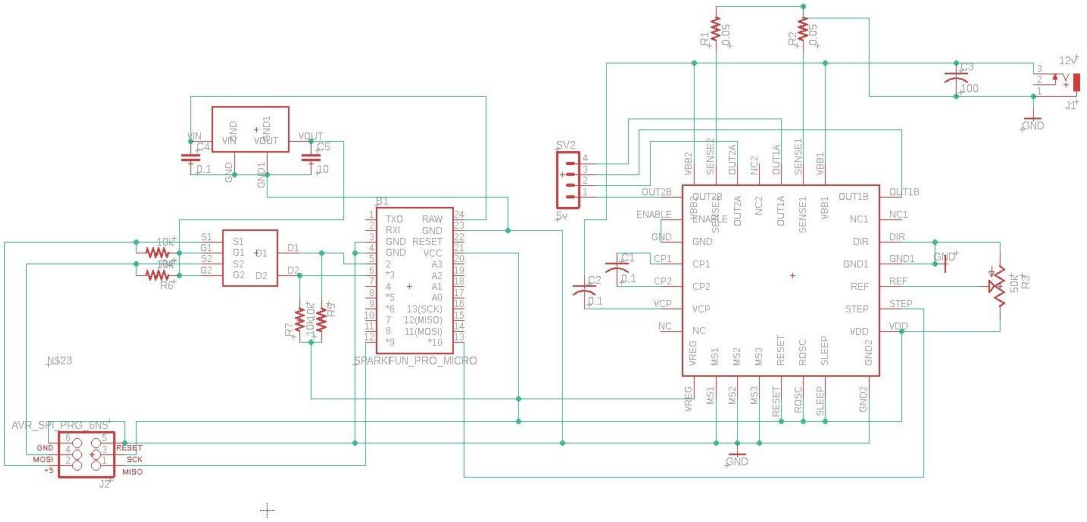

Figure 2: PCB Schematic

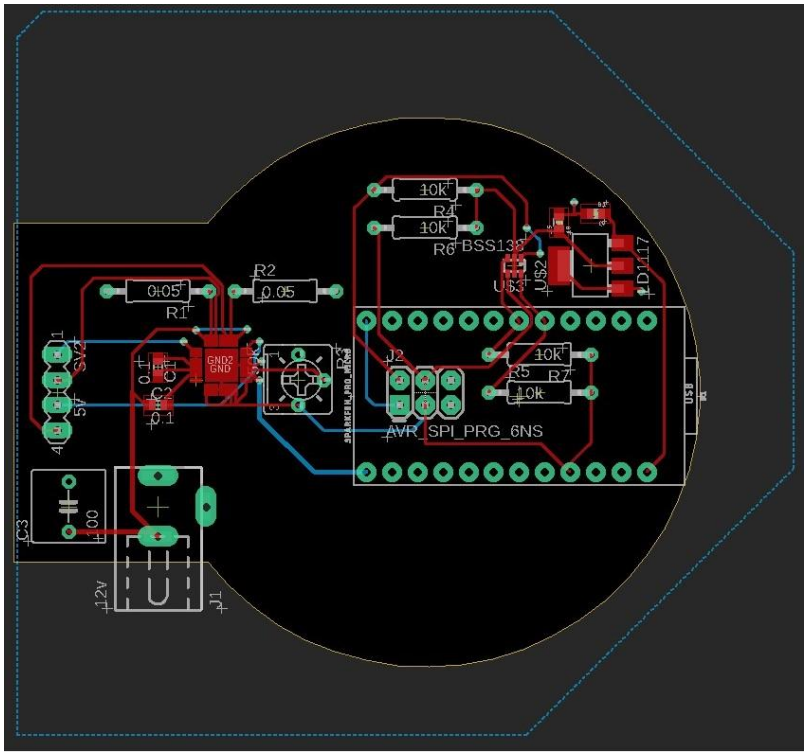

Figure 3: PCB Layout

Microcontroller (Software)

The microcontroller software is written in Arduino C (See Appendix D) and utilizes a number of different libraries related to rotating the TOF sensor and sending the collected data to the computer. In the setup phase, the motor angles are set to a default value and a serial connection is established to start sending data to the computer. Two separate functions are defined (get_TOF_meas, pulse_stepper) that send the appropriate pulses to the motors required for movement and get the measured point-to-point distance from the TOF sensor. In the main looping function, there is a continuous check for when the full scan has completed. Until that check has been met, the code grabs the TOF sensor data, sends movement pulses to the motors, and sends the current distance and two angles to the computer via a nested for loop. Due to the way data is sent via serial, the distance and angle values need to be parsed by the computer before they can be processed. Each successful loop increments the angle variables by a preset value and once the nested loop closes, a stop flag is printed to serial to signal the computer the scan has completed.

Subsystem 5 : Computer Block

The computer script is written in python3 (See Appendix E) and is responsible for parsing the data sent from the microcontroller and encoding it into the industry standard for LiDAR data, the las file format. After importing all the necessary python libraries, the script establishes a serial connection to the microcontroller and sets up a while loop that constantly reads new lines from serial and appends to an empty list. Once the program reads the stop signal sent from the microcontroller, it parses the data list

using numpy. Originally, we established a test for the spherical to cartesian coordinate conversion as part of the microcontroller subsystem. As part of the design process, however, we discovered that performing the conversion on the microcontroller then sending the converted values to the computer via serial created issues due to an implementation difference between arduino and python library functions that caused different conversion outputs. Ultimately, we decided that implementing the coordinate conversion in python would eliminate said issues as well as preserve a higher degree of data precision. The spherical coordinates are converted into cartesian coordinates using built-in numpy functions and then separated into three separate cartesian coordinate arrays representing x, y, and z. From there, the data is processed and ready to encode into a las file using the laspy library. A built in library class Header and the output file are instantiated before numpy is used to analyze the data arrays to gain important metadata required for the header class. From there, the calculated offsets and scales are assigned to the header and the three cartesian data arrays are equated to the output file before the las file is closed and ready for validation. Before viewing, the output file is validated using a laspy native terminal function that runs four tests on the output file that examine the bounding area, scaling factors, ranges, and other important values to ensure the encoded data makes sense. After all the tests pass, the LiDAR point cloud is now ready to be viewed by any of the numerous point cloud visualization software.

3. Design Verification

Subsystem 1: Primary Sensor Block

We want to make sure that our sensor functions properly because it is the most fundamental component in our project. First we want to check if the power that ranges from 3.7 to 5.2 V into the sensor is steady and reliable so there are no power issues. Therefore we connect our system to the computer through the 5 V USB power supply, then we start the operation of the system and use an oscilloscope to measure the working voltage for the sensor. The oscilloscope's results are seen in Figure 4 below. Thus we can see that the sensor has a quite steady working voltage from 4 to 4.86 V, which meets our requirements.

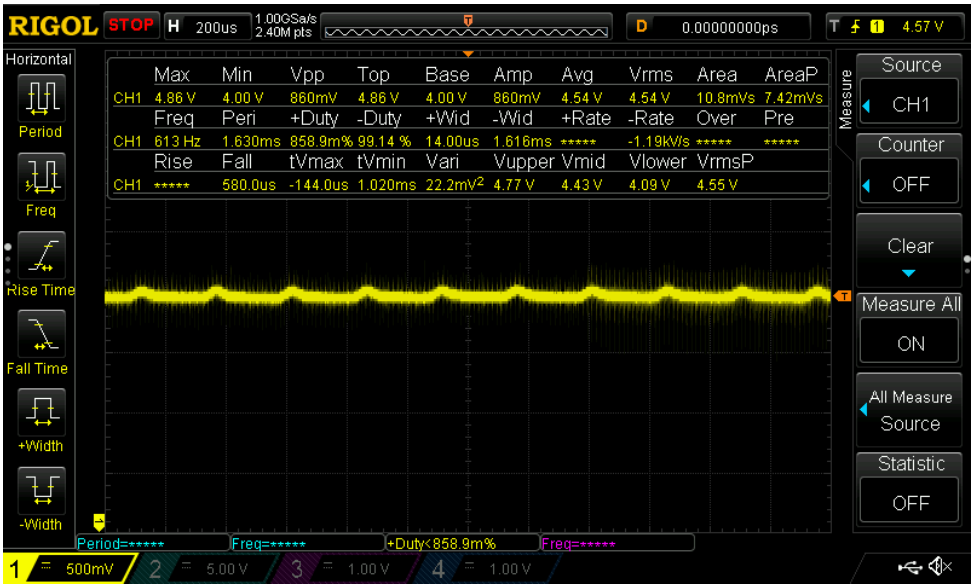

Figure 4: Sensor Working Voltage Test

Aside from the power supply, we also want to ensure that the sensor can maintain a high accuracy at an operating range around 6 meters. We set the operating range as 6 meters because this is suitable to scan a medium-sized room (10x10x5 m), which is the target environment and test environment for our system. As seen in Figure 5 below, when we do the test, we angle the sensor to a wall in a distance and only start the operation of the sensor. Then we move the sensor back and forth while checking the data from the computer to see if we have reached the targeted 6 meters. After we confirm we have reached the 6 meters and have a steady output from the computer, we set a marker for the position of the sensor and use a tape to check if the distance is actually 6 meters or not.

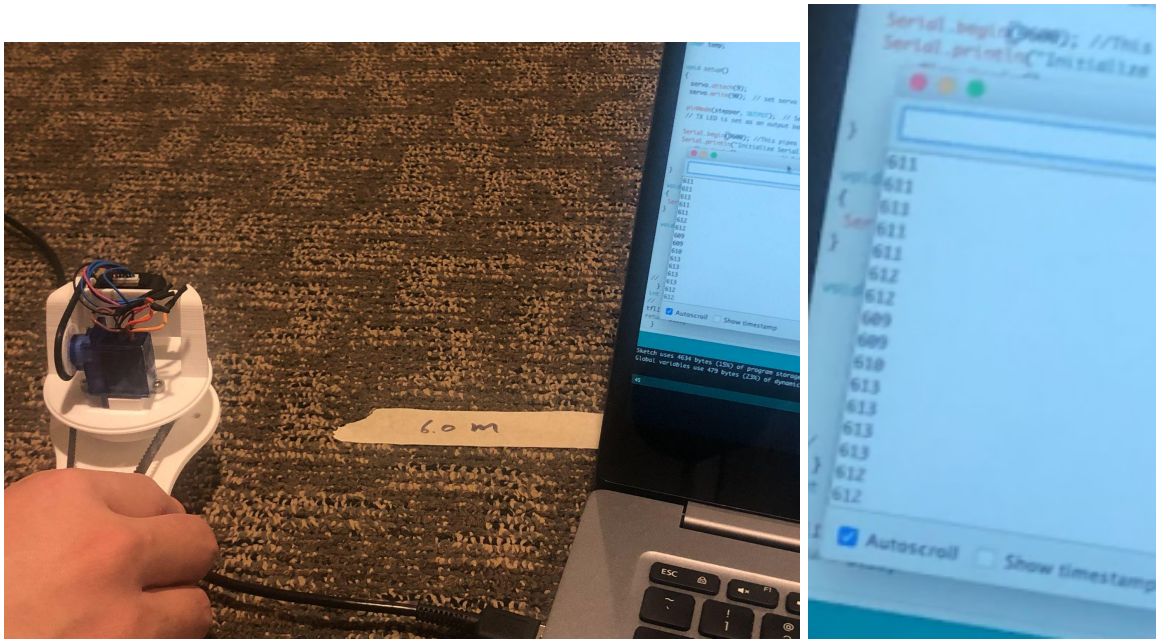

Figure 5: Sensor Operating Range Test (mm)

We also have to check the polling frequency of the sensor to ensure that it can collect enough data to generate the las output file in 70 seconds. According to the datasheet, the sensor can have a maximum frequency of 250 Hz. As seen in Figure 6 below, we start the operation of our system and write a program to confirm that the data reading in the computer is consistent and has a frequency around 250 Hz.

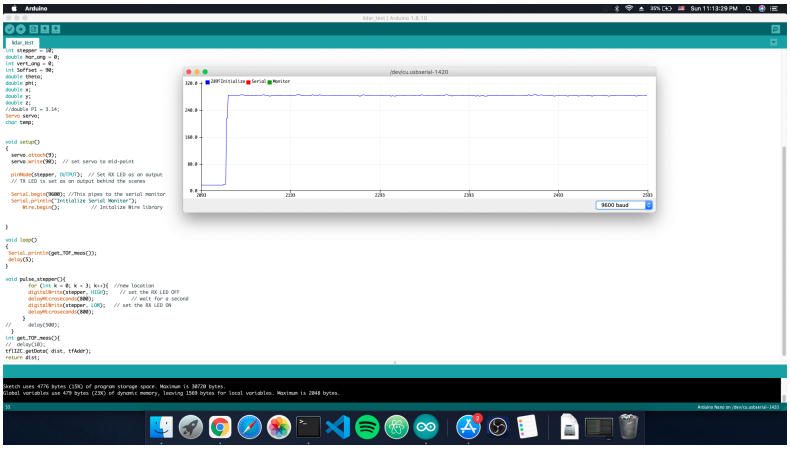

Figure 6: Sensor Frequency Test

Last thing we want to check with the sensor is that it can send the correct data to the computer block. To this end, we fix the position of the sensor and start its operation. The result can be seen in Figure 7 below. We can see that the data in the columns is consistent and regular, which confirms our requirement.

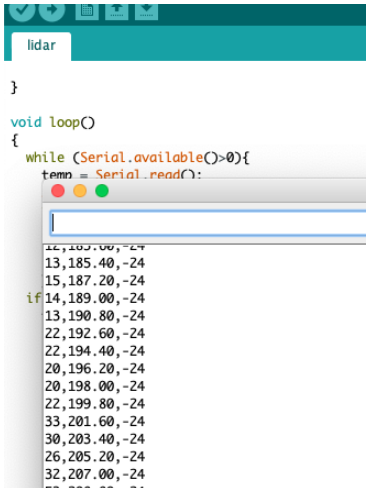

Figure 7: Sensor Data Correctness Test

Subsystem 2: Vertical Rotation Block

Because the vertical rotation block has a slip ring that acts as a transition point for signals to the sensor and the RC servo, first of all we want to test that the PWM signal and the I²C signal are not compromised by the noise from the slip ring. To this end we start the operation of the system. While the system is operating, we use an oscilloscope to collect the PWM signal and the I²C signal; the result can be seen in Figure 8 below. We can see that the yellow signal stands for the PWM signal sended to the RC servo and the blue signal stands for the I²C signal sended to the sensor. Thus these two signals are not compromised by the slip ring and can be differentiated easily.

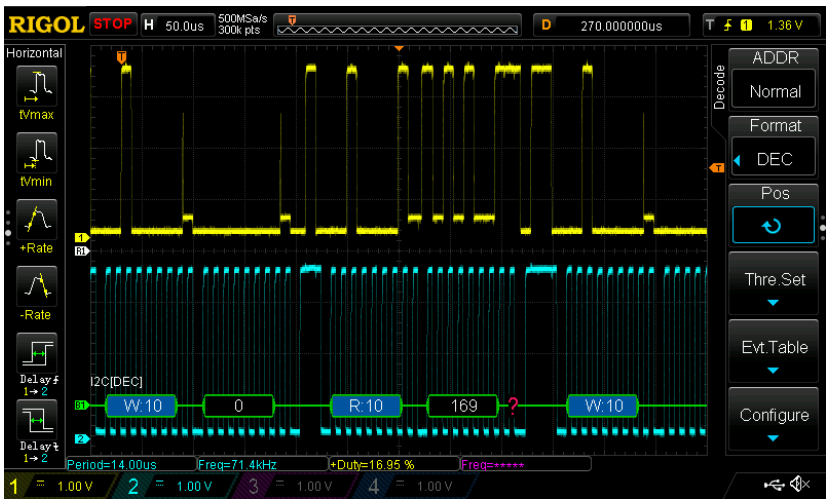

Figure 8: Slip Ring Noise Test

Subsystem 3: Horizontal Rotation Block

In this block, according to the datasheet, because the stepper motor driver requires both a 11.5 to 13.5 V power supply and a 3.7 to 5.2 V logic supply, we test if the motor driver actually has these study inputs during the operation. We start the operation of the system and use an oscilloscope to separately test the power at the power supply pin and the logic supply pin on the A4988 stepper motor driver chip. The result can be seen in Figure 9 below. It is clear that the power at the power supply pin, which corresponds to the top picture, has a voltage from 11.8 to 13.1 V and the power at the logic supply pin, which corresponds to the bottom picture, has a voltage from 4.02 to 4.8 V. Thus we can conclude that the requirements are met.

Figure 9: Stepper Motor Driver Power Test

Subsystem 4: Microcontroller Block

For the microcontroller block, because the microcontroller is the core of our system, first of all we want to make sure that it can operate with a steady power thus we can ensure the system can work continuously for at least 70 seconds. Therefore we test if the microcontroller has a power supply that is within 10% of 4.5 V while it is operating. The result can be seen in Figure 10 below. It is clear that the microcontroller has a steady power supply that ranges from 3.96 to 4.76 V, which meets our requirements.

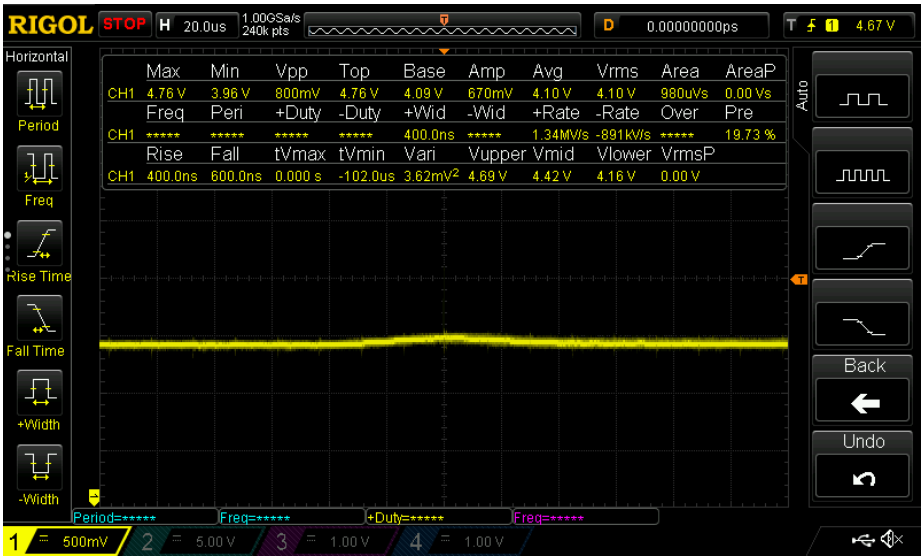

Figure 10: Microcontroller Power Test

Because the successful communication between the microcontroller and the sensor is the most fundamental element in our project, the BSS138 bi-directional level shifter is one of the most important components in the system since it ensures the reliability of the communication. Therefore we test if the level shifter works properly in the system. We start the operation of the system and collect the signals on the two sides of the level shifter by an oscilloscope. The result can be seen in Figure 11 below. The yellow signal, which stands for the signal from the sensor, has a voltage around 3.4 V and the blue signal, which stands for the signal from the microcontroller, has a voltage around 4.64 V. Thus we can confirm the level shifter functions as we expected.

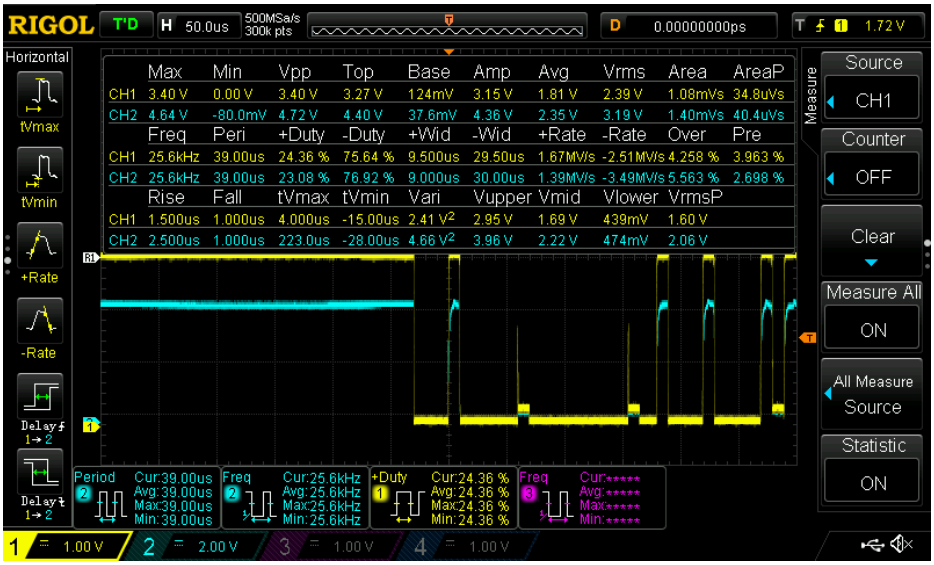

Figure 11: Bi-directional Level Shifter Test

Subsystem 5: Computer Block

Since the computer subsystem is responsible for the processing of the data and encoding and output of the final point cloud las file, it is important for us to verify its integrity while the system is in use. More specifically, since the Computer provides ~5 V/500 mA via the USB port to the system, we ran a test to ensure that power is continuously supplied to the microcontroller. As can be seen below in Figure 12, the oscilloscope reading on the voltage output for the USB port ranges from 3.96 to 4.76 V which falls within our acceptable range of 3.7 to 5.2 V.

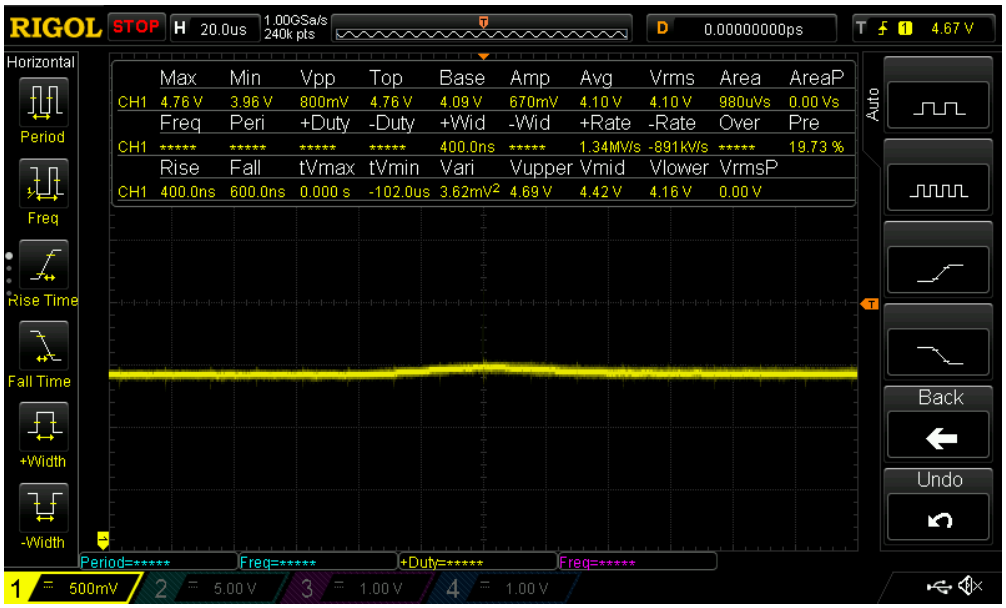

Figure 12 USB Port Power Test

As part of the python script, the data transfer process from the microcontroller to the computer via serial connection plays an important part as any data errors or an unreliable connection could have drastic effects on the final output point cloud. We tested the reliability and accuracy of the serial data transfer by comparing the 3 coordinate tuples sent from the arduino side to the serial lines of data received on the computer side. As observed below in Figure 13, the point cloud data sent from the arduino (shown in the notepad file) exactly matches the data that was received and stored in the python "strdata" list which demonstrates that the serial connection can transmit the 3D spherical coordinates in a reliable and accurate manner.

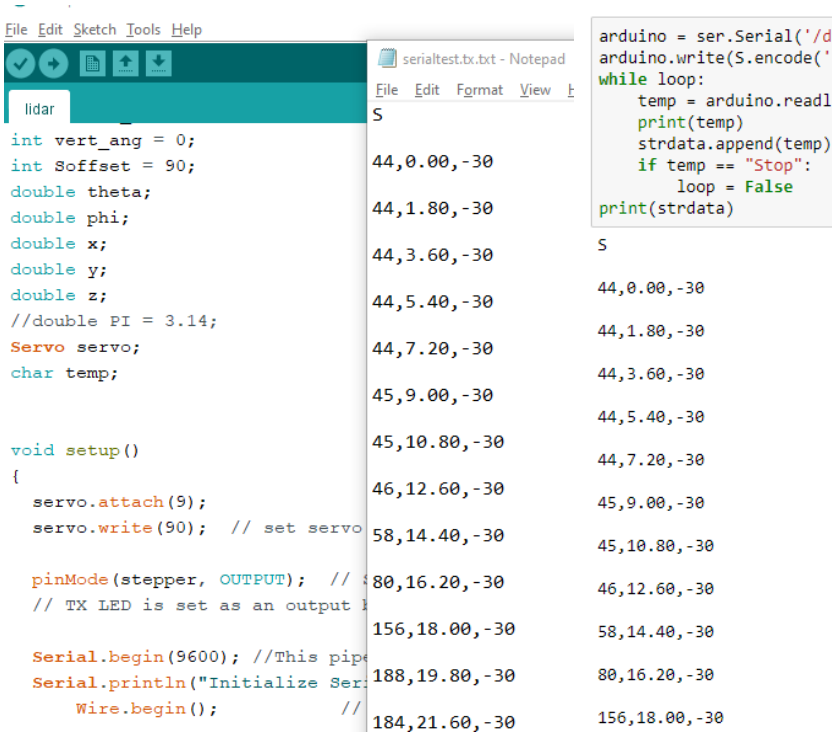

Figure 13 Serial Data Stream Reliability Test

Finally, after the point cloud data has been decoded and parsed, it is then encoded using the laspy library into a las file. However, there is still no guarantee that the resulting point cloud visualization will resemble the scanned environment or even load properly in the visualizer software for that matter. To address this, we utilize a laspy library function called lasvalidate. This function runs 4 tests on the output las file testing the file makeup, bounding box precision, point data positions, and dimension ranges all of which can indicate errors in the point cloud data. By properly passing all the tests run by lasvalidate, the confidence level of the output las file increases significantly. In Figure 14 below, the terminal output of a lasvalidate test can be seen where the las file has successfully cleared all the test ran as well as the visualization of the point cloud data which accurately resembles the environment scanned.

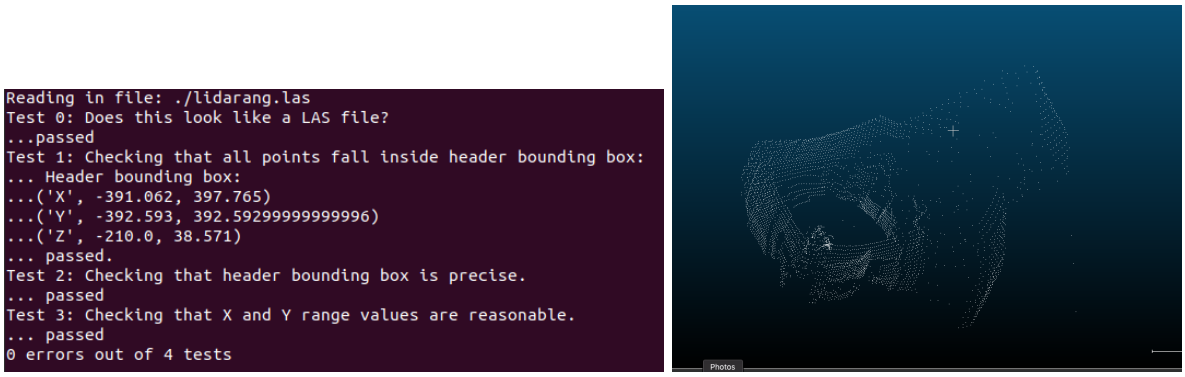

Figure 14 Point Cloud File Output Validate Test

4. Costs

4.1 Parts

Parts Costs

Part	Part Number	Manufacturer	Single Unit Cost ($)
6 wires x2A flanged slip ring	SRW-12-06A	Adafruit	$ 5.49
Micro RC Servo	SG90	Miuzei	$ 1.70
NEMA 11 stepper	11HS12-0674S	OSM Technology	$ 18.79
GT2 60T timing pulley	KINGPRINT-WCL000015	Walfront	$ 8.99
GT2 60T timing pulley	M-GT2-PULLEY-20T		
GT2 200mm belt	200-2GT-6		
Single-point LiDAR module	SJ-GU-TF-Luna A03	Benewake	$ 22.89
Microcontroller development board	Atmega32U4	Microchip Technology	$ 4.99
Stepper motor driver	A4988	Allegro	$ 1.59
Total			$ 64.88

4.2 Labor

Using the hourly labor rate of $50 as provided in the instructions with a 250% overhead multiplier, each senior design business labor hour translates to $125. On average, 8 hours per week is spent each for three people over a period of 8 weeks from the proposal deadline to the demo date, the total labor cost is calculated below:

$$\$50/hr \ * \ 2.5\ (overhead) \ * \ 8\ hr/week \ * \ 3\ people \ * \ 8\ weeks \ = \ \$28,800$$

4.3 Manufacturing

Type	Hours	Rate Per Hour	Material Cost	Overall Cost
3D printing	20	$2.00	$19.99	$59.99
machine shop	1	$50.00	$0.00	$50.00
PCB	N.A	N.A	$4.99	$4.99
Total				$114.98

4.4 Total

$$\$64.88\ (parts)\ +\ \$28,800\ (labor)\ +\ \$114.98\ (manufacturing)\ =\ \$28,978.86$$

The total development cost of this project is $28979.86 when labor is considered. It is worth mentioning that the material cost can be reduced when bulk ordering and the 3D printing cost can be reduced down to a few dollars per part when converting to injection molding. With all this in mind, we believe we can reduce the material and manufacturing cost down to close to $100, leaving us enough overhead to market to the enthusiast and hobbyist market.

5. Conclusion

5.1 Summary

Looking at our final system's performance, our team believes that the overall project was successful. The system was able to scan its environment and output a recognizable point cloud file with relatively decent speed and accuracy. Through taking a cheap one-dimensional distance scanner and incorporating it into a two dimensional rotational mechanism, we can emulate the functionality of a pricey flagship 3D LiDAR sensor. At the same time, by feeding the data into a computer and encoding it with an easily adaptable script into an industry standard point cloud format for visualization, we successfully circumvent any proprietary hurdles that may arise from using industry level LiDAR scanners. It is our hope that individual hobbyists interested in the LiDAR field can now incorporate our straightforward and affordable solution into their own projects and catalyze more interest and innovation in the LiDAR field and its many possibilities.

5.2 Ethicals/Safety

Our project aims to lower the cost of entry into the light-based scanning field. By minimizing the construction cost and streamlining the assembly, setup, and maintenance processes, we hope to make it easier for everyone to gain a better "understanding...of conventional and emerging technolog[y]"[4] that is 3D LiDAR.

As stated in the IEEE Policies Section 7.6, the "safety, health, and welfare of the public" [5] is to be held paramount. With the ToF sensor utilized in our design, there is an 850 nm laser that has the potential to cause damage to the human eye. With this wavelength of light, humans cannot distinguish its intensity and can easily be injured as a result [6]. To account for this, our team will utilize special safety laser goggles that protect against the laser and will include a safety warning to any consumers.

It is important to note that a cheaper cost of entry for 3D LiDAR systems could make it easier for an individual to conduct more precise attacks via an unmanned device. Already, higher complexity LiDAR systems are being utilized by the US military. "...military scientific research institutes in the United States have developed...LiDAR seekers with specific algorithms have the ability of automatic target recognition [7]." Though our project does not exist to promote such uses, there remains a definite possibility of a third party utilizing our system alongside others to threaten public safety which also contradicts Section 7.6 mentioned above. We acknowledge the slight possibility of the use of our system for unintended purposes that could breach the IEEE code, however in the efforts to maintain device functionality and given the low probability of misuse, the system will not be designed to counteract this specific concern.

5.3 Uncertainties & Future work

Looking at the future, there are several areas of our system that could be improved upon. Because currently our project is partly restricted by our sensor's refresh rate and precision when it is in operation, by utilizing a higher refresh rate and precision TOF sensor, the resulting point cloud resolution should increase, providing more sharp and noticeable features in addition to potentially increasing the system's

maximum scan dimensions. Because the rotation speed of the stepper motor is in accordance with the refresh rate of the sensor, we are also partly limited by this factor. Thus alternatively, by decreasing the cycle duration with a faster stepper motor, the overall scan time can be reduced allowing for smaller vertical angle increments which will also increase point cloud resolution.

Another uncertainty of our project is that currently, our system utilizes a 12 V DC barrel jack power supply for power, which is inconvenient. Therefore, we hope to replace the power supply such that the system can be powered solely from the 5 V provided by the computer USB port through voltage stepping up with a transistor. By doing this, any reliance on an external power supply is eliminated improving device versatility and portability.

Apart from all the uncertainties above, our A4988 stepper motor driver is potentially to be burned due to high current. Thus we can possibly add a polyfuse to protect the motor driver in the future.

On the software side of the system, currently we have to generate the las file each time we take a scan and drag the file to software to get a final point cloud. Thus the possible enhancements include updating the code so that multiple intermediary las files are generated which can illustrate the real time point cloud data generation slice-by-slice as each device cycle is completed for better visualization purposes. In tandem with this, implementing the point cloud file into a virtual reality space where individuals would be able to explore the surroundings and navigate using VR movement rather than through mouse movements would also vastly improve the user experience.

References

[1] R. Amadeo, “Google's Waymo invests in LIDAR technology, cuts costs by 90 percent,” Ars Technica, 10-Jan-2017. [Online]. Available: https://arstechnica.com/cars/2017/01/googles-waymo-invests-in-lidar-technology-cuts-costs-by-90-percent/. [Accessed: 18-Feb-2021].

[2] C. Domke and Q. Potts, “LiDARs for self-driving vehicles: a technological arms race,” Automotive World, 03-Aug-2020. [Online]. Available: https://www.automotiveworld.com/articles/lidars-for-self-driving-vehicles-a-technological-arms-race/. [Accessed: 18-Feb-2021].

[3] Benewake Co. Ltd, “TF-Luna LiDAR Module” SJ-GU-TF-Luna A03 datasheet, Available: https://github.com/budryerson/TFLuna-I2C/blob/master/documents/TF-luna-A03%20Datasheet.pdf [Accessed: 18-Feb-2021]

[4] Ieee.org. 2021. IEEE Code Of Ethics. [online] Available at: <https://www.ieee.org/about/corporate/governance/p7-8.html> [Accessed 17 February 2021].

[5] Ieee.org. 2021. IEEE Policies. [online] Available at: <https://www.ieee.org/content/dam/ieee-org/ieee/web/org/about/corporate/ieee-policies.pdf> [Accessed 17 February 2021].

[6] P. D. Alex Kilpatrick, IR Illumination and Eye Safety, 14-Sep-2017. [Online]. Available: https://medium.com/@alex.kilpatrick/ir-illumination-and-eye-safety-f0804673ca7. [Accessed: 17-Feb-2021].

[7] M.-le Li, Y.-hua Hu, N.-xiang Zhao, and Q.-shu Qian, “Modeling and analyzing point cloud generation in missile-borne LIDAR,” Defence Technology, 18-Oct-2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214914719308062#bib4. [Accessed: 17-Feb-2021].

Appendix A: Requirement and Verification Table

System Requirements and Verifications

Requirements, Verification, and Results per Project Subsystem		
Requirement	Verification	Result
Sensor Subsystem		
The LiDAR sensor can receive enough power ranging from 3.7 to 5.2 V to ensure the sensor can work in normal conditions for at least 70 seconds.	1. Connect the LiDAR sensor to a computer through the USB port. 2. Use an oscilloscope to measure the voltage across the power pins of the LiDAR sensor to make sure that the voltage is within the range from 3.7 V to 5.2 V while the platform is rotating at its typical speed of 60 RPM	Satisfied; Voltage was within specified range
The LiDAR sensor sends the data smoothly to the control subsystem through a wire under the I^2C protocol.	1. Fix LiDAR sensor position so it can take a distance measurement in a specific direction. 2. Connect the LiDAR sensor to the microcontroller through a wire under I^2C protocol. 3. Write a program that streams the reading from the sensor over serial. 4. Check if any of the readings are lost or irregular.	Satisfied; No readings seemed irregular
The LiDAR sensor polls at a frequency of 200 Hz ±5% to ensure adequate scanning efficiency outlined in the high-level requirements.	1. Connect the LiDAR sensor to the microcontroller through a wire that is under I^2C protocol. 2. Write a program that streams the reading from the sensor over serial at 200 Hz. 3. Check if the LiDAR sensor readings are consistent. 4. Push the frequency even higher to see the tolerance range.	Satisfied
The sensor has a max operating range of 6 m that enables it to scan a room with a dimension of 10x10x5 m.	1. Fix LiDAR sensor position so it can take a distance measurement in a specific direction. 2. Set object ~6 meters from the LiDAR sensor connected to the microcontroller under I^2C. 3. Start the sensor and check the controller for a clear distance output.	Satisfied; Max distance around 9 m where random noise increased error significantly

	4. Use a tape measure to measure the actual distance and ensure the error is within 0.10 m of the actual distance.	

Requirements, Verification, and Results per Project Subsystem		
Requirement	Verification	Result
Vertical Rotation Subsystem		
Slip ring ensures the PWM and I^2C signals are not compromised under constant rotation. Also, the noise induced by the slip ring must not interfere with the I^2C signal sent from the primary ToF sensor or the PWM signal going to the RC servo.	1. Attach TOF sensor on the vertical rotation block. 2. Connect the LiDAR sensor to the microcontroller under I^2C and the servo to the controller via a PWM wire passing through the slip ring. 3. Start the operation and check if the system works normally to make sure that the I^2C and PWM signals are transmitted correctly. 4. Check if the sensor sends clear outputs and the servo is rotating constantly and continuously ensuring that slip ring noise does not interfere with I^2C and PWM signals.	Satisfied
The 5 to 5.5 V DC power coming out of the slip ring must be smoothed out enough to fit the 3.7 to 5.2 V requirement of the ToF sensor.	1. Connect the LiDAR sensor to a 5 to 5.5 V DC power via computer USB port through a slip ring wire. 2. Use an oscilloscope to measure the voltage across the sensor while the slip ring is rotating to make sure that the voltage is still within the range from 3.7 to 5.2 V.	Satisfied

Requirements, Verification, and Results per Project Subsystem		
Requirement	Verification	Result
Horizontal Rotation Subsystem		
This subsystem must be repeatable enough to maintain spatial awareness over 30 rotations.	1. Connect the stepper motor to the stepper motor driver andthe motor driver to the microcontroller. 2. Write a program that rotates the stepper motor continuously at 60 RPM until given an angle from	Verified

	the computer which the stepper motor rotates to. 3. Wait for 30 revolutions and then input a random angle. 4. Observe whether the stepper moves to the correct angle using a protractor.	
The motor driver receives a voltage between 3.3 to 5.2 V for the logic supply from the microcontroller and 11.5 to 13.5 V for the power supply from the 12 V DC jack.	1. Connect the stepper motor driver to the microcontroller and motor driver. 2. Connect the DC power jack to the motor driver. 3. Start the operation of the motor driver and the motor; use an oscilloscope to measure the voltages across the motor driver at the logic supply pin and the power supply pin to ensure that the motor driver has a voltage between 3.3 to 5.2 V at the logic supply pin and has a voltage from 11.5 to 13.5 V at the power supply pin.	Verified
The stepper motor has a step angle within 5% of 1.8 degrees to make sure that the scanning process has a resolution and precision of ±0.10 m at a distance of 3 m and ± 0.02 m at a distance of 0.2 meters.	1. Connect the stepper motor to the stepper motor driver. 2. Connect the motor driver to the microcontroller. 3. Write a program that will step the stepper motor 1 full step 4. Measure with a protractor whether the rotation is 1.8 ± 5% degrees.	Verified

Requirements, Verification, and Results per Project Subsystem		
Requirement	Verification	Result
Microcontroller Subsystem		
Microcontroller receives 3.7 to 5.2 V DC power from a computer to operate at a voltage within 10% of 4.5 V.	1. Connect the microcontroller to a computer via USB. 2. Run the microcontroller on some arbitrary code and use an oscilloscope to measure the voltage across the controller to ensure that it operates at a voltage within 10% of 4.5 V.	Verified
Each PCB trace is uncompromised.	1. Use a multimeter in continuity mode to test each trace to ensure the PCB manufacturing doesn't have any faults	Verified

Components will work with the PCB incorporated.	1. Solder every component onto the PCB. 2. Write a program that will briefly unit test the functionality of each component.	Verified
The BSS138 bi-directional level shifter successfully shifts the signals from the sensor, which is a voltage within 10% of 3.3 V, and from the microcontroller, which has a voltage from 3.7 to 5.2 V.	1. Connect the sensor to the microcontroller via a BSS138 bi-directional level shifter 2. Start the operation of the system and collect the voltage at the two sides of the level shifter by an oscilloscope. 3. Check if the two signals match with each other in the time frame; also check if one of the signals has a voltage within 10% of 3.3 V, and the other has a voltage from 3.7 to 5.2 V..	Verified

Requirements, Verification, and Results per Project Subsystem		
Requirement	Verification	Result
Computer Subsystem		
Converts the spherical coordinates into Cartesian coordinates at a rate of 200 Hz ±5% (refresh rate of ToF sensor).	1. Connect the microcontroller to the computer; establish serial connection 2. Run python program and establish serial connection 3. Artificially generate 200 tuples of measurements for the distance, vertical and horizontal angle. 4. Time the spherical to cartesian conversion rate to confirm the entire process is at or under 1 second. 5. Collect and verify that the conversion between spherical and cartesian coordinates was successful. 6. Repeat steps 3 to 5 multiple times to ensure that we can have a successful conversion at a rate within 5% of 200 Hz.	Verified (Changed subsystem from microcontroller to computer)
The USB port can continuously output a voltage from 3.7 to 5.2 V and a maximum current draw of 500 mA ±5% to the rest of the system.	1. Connect the computer, microcontroller, and the sensor-servo subsystem that consists of the servo motor,	Verified

	LiDAR sensor, stepper motor, and micro-stepping motor driver. 2. Attach a multimeter in between the Computer and the cable providing power to the system. 3. Ensure that the voltage ranges from 3.7 to 5.2 V and current do not exceed 500 mA.	
The program successfully converts the raw data into a LAS file within the 20 seconds.	1. Feed-in pre-generated spherical coordinates into the python program and encode the LAS output file to be output. 2. Run a file hash checksum (lasvalidate) on the output LAS file to ensure the file has been encoded properly.	Verified

Appendix B: Sensor Rotation Module Visualization:

Appendix C: Physical Design:

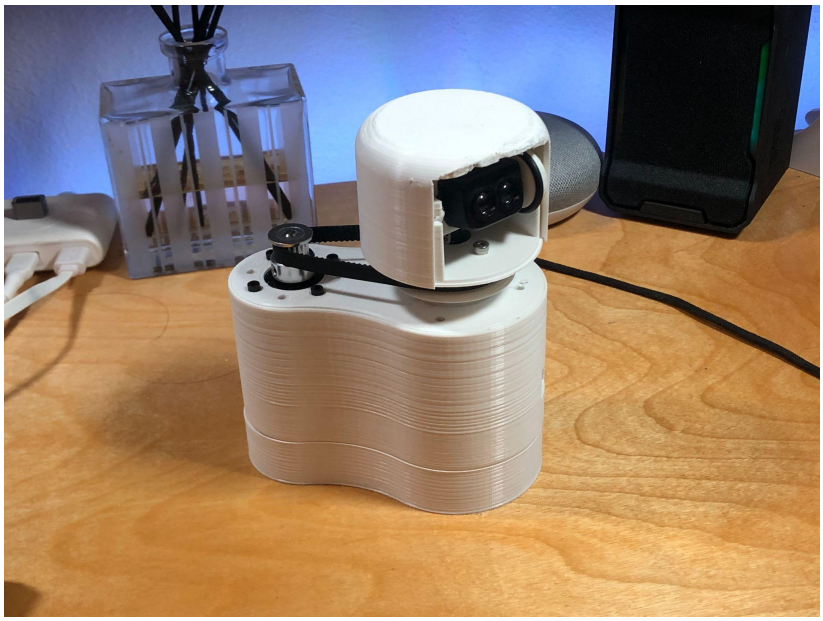

Figure 1: Final Design Prototype

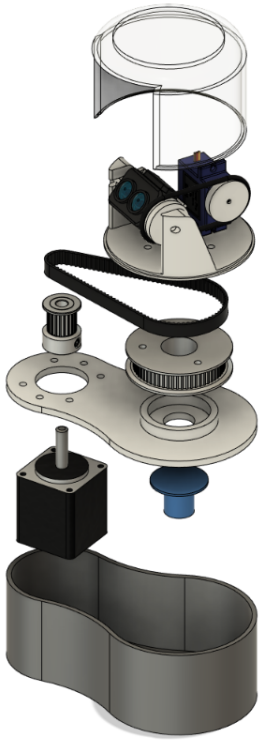

Figure 2: Exploded CAD Design View

Appendix D: Arduino Code (lidar.ino):

```
#include <Servo.h>
#include <Arduino.h>     // Every sketch needs this
#include <Wire.h>        // Instantiate the Wire library
#include <TFLI2C.h>      // TFLuna-I2C Library v.0.1.0
#include "math.h"
TFLI2C tfII2C;

int16_t  dist;    // distance in centimeters
int16_t  tfAddr = 0x10;  // Default I2C address
int stepper = 13;
double hor_ang = 0;
int vert_ang = 0;
int Soffset = 90;
double theta;
double phi;
double x;
double y;
double z;
Servo servo;
char temp;

void setup()
{
  servo.attach(9);
  servo.write(90);  // set servo to mid-point

  pinMode(stepper, OUTPUT);  // Set RX LED as an output
  // TX LED is set as an output behind the scenes

  Serial.begin(9600); //This pipes to the serial monitor
  Serial.println("Initialize Serial Monitor");
     Wire.begin();          // Initialize Wire library
}

void loop()
{
  while (Serial.available()>0){
    temp = Serial.read();
    Serial.println(temp);
    }
  if(temp == 'S'){
```

```
   temp = ' ';
   for (int i = 0; i < 20; i++){
    servo.write(vert_ang+Soffset-30);
    for (int j = 0; j < 200; j++){
     pulse_stepper();
     dist = get_TOF_meas();

     Serial.println(String(dist)+","+String(hor_ang)+","+String(vert_ang-30));
     hor_ang = fmod(hor_ang + 1.8,360.0);
    }
    vert_ang = fmod(vert_ang + 3,60.0);
   }
   Serial.print("Stop");
  }
}

void pulse_stepper(){
      for (int k = 0; k < 3; k++){  //new location
      digitalWrite(stepper, HIGH);    // set the RX LED OFF
      delayMicroseconds(800);            // wait for a second
      digitalWrite(stepper, LOW);   // set the RX LED ON
      delayMicroseconds(800);
     }
  }

int get_TOF_meas(){
     tfII2C.getData( dist, tfAddr);
     return dist;
  }
```

Appendix E: Python Code (lidar.py):

```
import serial as ser
import numpy as np
import laspy

strdata = []
curr = 0
counter = 0
x, y, z = 0, 0, 0
allX = []
allY = []
allZ = []
S = 'S'
loop = True

arduino = ser.Serial('/dev/ttyUSB1', baudrate = 9600, timeout=.1) #assuming serial connection is
(COM3)
arduino.write(S.encode('utf-8'))
while loop:
   temp = arduino.readline().decode('ascii')
   print(temp)
   strdata.append(temp)
   if temp == "Stop":
      loop = False
print(strdata)

strdata = [x.replace("\r\n","") for x in strdata]
strdata = [ x for x in strdata if "," in x ]
data = [ x.split(",") for x in strdata]
data = np.array(data).astype(float)

data1 = np.zeros(data.shape)
data1[:,0] = data[:,0]*np.cos(data[:,2]*np.pi/180)*np.cos(data[:,1]*np.pi/180)
data1[:,1] = data[:,0]*np.cos(data[:,2]*np.pi/180)*np.sin(data[:,1]*np.pi/180)
data1[:,2] = data[:,0]*np.sin(data[:,2]*np.pi/180)

hdr = laspy.header.Header()
lasOutput = laspy.file.File("lidarang.las", mode = 'w', header = hdr)
xMin = np.floor(np.min(data1[:,0])) #use numpy to get int min values of x y z
yMin = np.floor(np.min(data1[:,1]))
zMin = np.floor(np.min(data1[:,2]))
lasOutput.header.offset = [xMin, yMin, zMin] # set header offset
```

```
lasOutput.header.scale = [0.001, 0.001, 0.001] #3 decimal point precision, can change as
necessary
lasOutput.x = data1[:,0]
lasOutput.y = data1[:,1]
lasOutput.z = data1[:,2]
lasOutput.close()

#validate function to test output file
import os
print("Running validate function")
os.system("lasvalidate ./lidar.las") #replace lidar.las with whatever the output file name is
os.system("cat ./lasvalidate.log") #open log file
```

Appendix F: Physical Design for Each Unit:

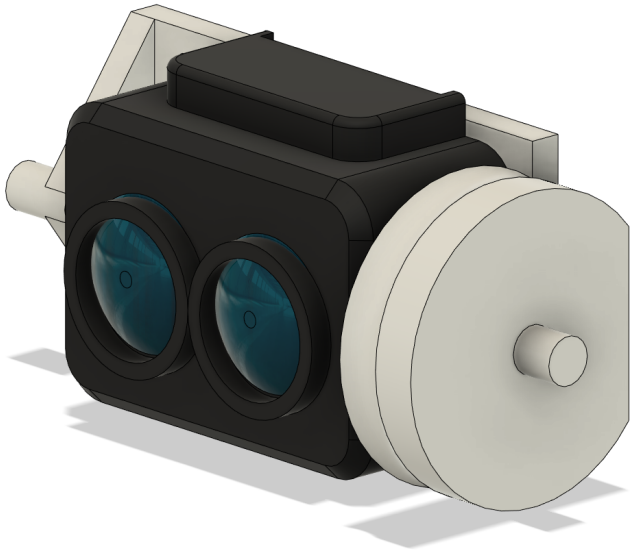

Figure 1: Lidar TOF Sensor

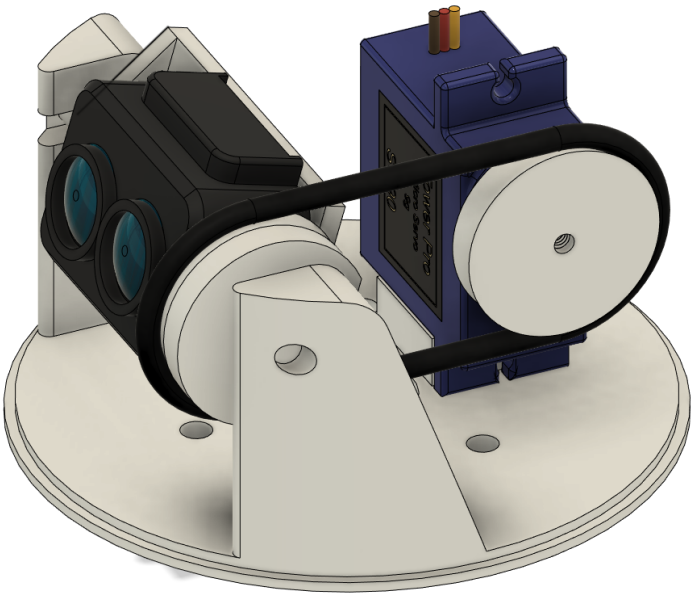

Figure 2: RC Servo

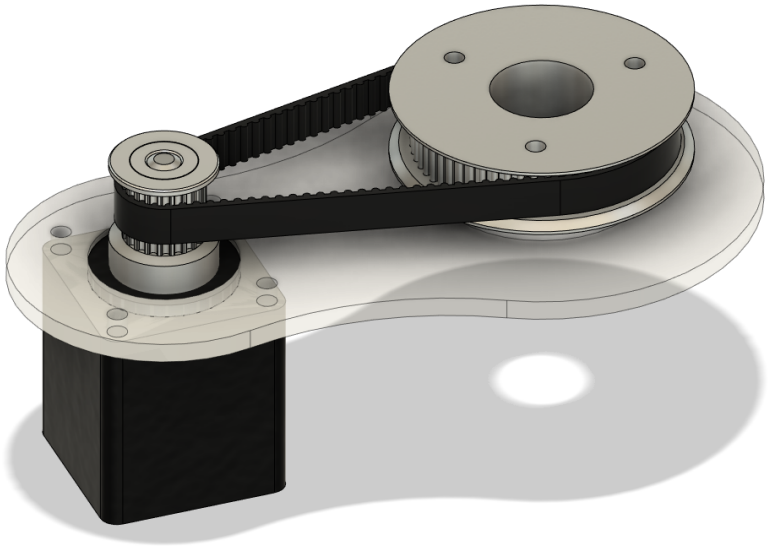

Figure 3: Stepper Unit

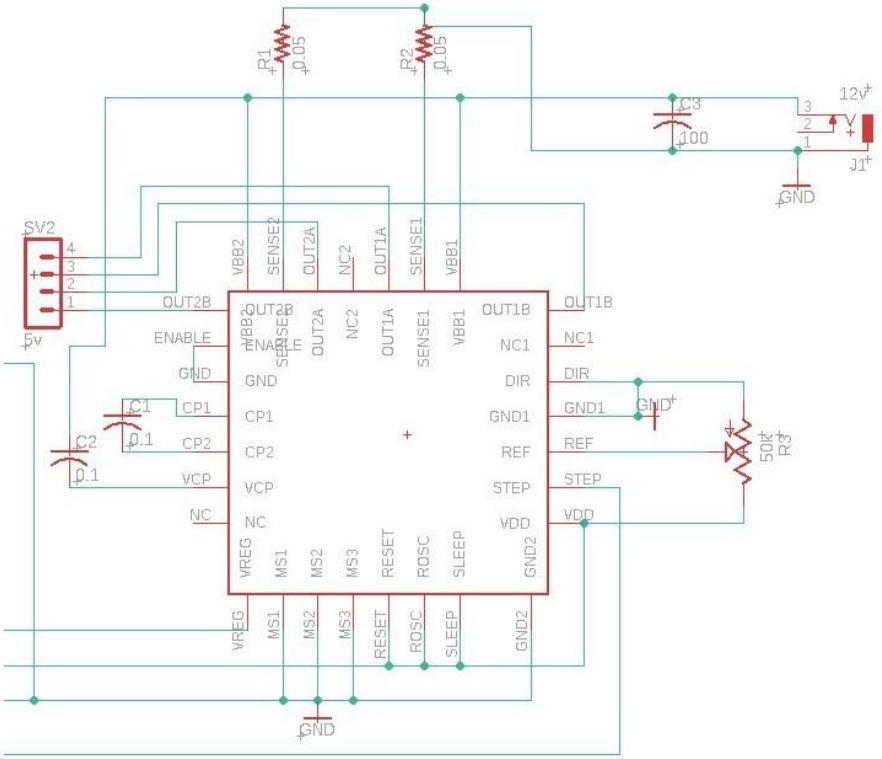

Figure 4: Schematic for A4988 Stepper Motor Driver

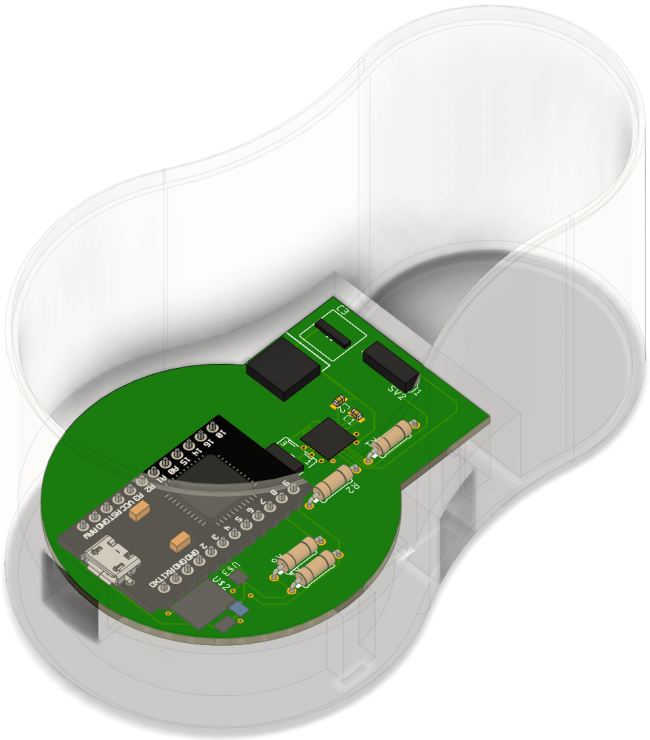

Figure 5: Microcontroller