Modeling Environments through Range Scanning

A Senior Project submitted to
The Division of Science, Mathematics, and Computing
of
Bard College

by
Erik Shagdar

Annandale-on-Hudson, New York
May, 2011

Abstract

This project explores creating three-dimensional models of environments using a series of two-dimensional range images. Firstly, available sensor options are surveyed and a three-dimensional laser range finder is designed and built using a tilting two-dimensional laser range finder. Secondly, data is collected from various environments under different scanning conditions. Finally, the environments are modeled as three-dimensional clouds of points

Contents

List of Figures

Dedication

To my grandfather – the man I respect the most.

Acknowledgments

I would like to acknowledge Keith O'Hara for his endless help on this project, without whom this would have been a rough year. I would also like to thank Sara and Meghan for being models in the scans.

1
Introduction

1.1 Overview

Range detection is used in applications varying from military applications, naval navigation, 3D model building, and even map building. Range finders use the concept of transmitting a signal, waiting until it hits some obstacle and returns to the source to be received, and then measuring the time of flight to cover the distance. As technology improved, so did detection – simple scanners became faster, more complex and accurate.

Laser range scanners transmit a pulse of light (typically an infrared laser) and measure the time difference between when it was sent and when it was received. Light is used because its speed is constant. The only variable is the distance to the object, and is measured by taking half of the product of the time it takes for the pulse to travel to the object and back. with the speed of light

Most scanners available provide one-dimensional (1D) or two-dimensional (2D) scans. This work explores three-dimensional (3D) model building by combining a 2D scanner

with the ability to rotate about the other axis. There are existing 3D scanners available, such as those produced by Velodyne, which cost about $75,000. The laser used in this project costs just under $1,200, but with all the servos the total cost becomes $2,100. This is still a small fraction of the cost used to accomplish the same concept. However, the laser developed in this project is slower

As Blais mentions in his paper on the review of range sensor development [6], range sensing has made many advancements in recent years. 2D range sensing is the most popular and efficient way of imaging environments. It is a logical step from single point detectors and offers a much faster, easier, and complete approach to one-dimensional scanning. These point scanners serve as a basis for establishing a way to optically recognize surfaces. A laser projection and an imaging lens are used to triangulate the distance of the object with the use of mirrors. This concept is then extended to a 2D detector to provide planar projection. With the use of a tilting mechanism, the planar scan can be extended into a third dimension.

A laser range finder capturing a scan looks at a 2D surface, and collects all the points within it's range of view where it senses the nearest obstacle. Each point, not being able to know anything about it's neighbors, stands alone. The nature of the scanner doesn't provide any information about what is behind that first obstacle point. This combined with the way 2D scanners are built and measure distances, offer limited information that decreases as the obstacles move farther away. With the help of a servo, the scanner tilts and scans a new plane. The problem becomes more interesting when going beyond this sparse information about points: would it be possible to recreate the environment the scanner is looking at? Further, given these limited points, is it possible for someone to infer whether the recreated model looks similar to the environment, or is the loss of some

information vital to the inability to create models?

The 3D scans will be assessed based on how more detail can be gathered and presented in a neater and more user friendly way. As shown in the paper, the slower the scans are, the denser and more detailed the visualizations will be. Given a collection of points with coordinates- a point cloud, how well can the environment be recreated? Of course the evaluation is left somewhat to the viewer and their ability to infer what the points depict. One goal is to have viewers agree that the image accurately conveys the actual environment.

1.2 Explorations of Moving the Laser

Figure 1.2.1. Arm

The project started with a concentration that differed from the final product. The initial idea was to create a manipulator that would learn about itself and complete tasks. Learning was done by using forward learning models where the robot would bend each

joint to see what the effects would be [8]. After this initial step, the manipulator would scan for obstacles within its range and move them based on what shape it thought they were.

Figure 1.2.2. Arm and Laser

The manipulator consisted of a series of joints and a grabbing mechanism. The joints are made up of two servos used to bend the arm. Two servos were needed because a single servo was too weak to lift the manipulator by itself. Bending of the joints was a trick of its own. The servos, having a serial connection, could communicate with the central module one at a time. To overcome this, the two servos alternated moving a small amount (much like people walk). One servo would lock itself and support the weight of the arm while the other would move slightly past than the first and lock itself. The first would then unlock, move past the first, and re-lock. This process was repeated multiple times to achieve bending at any angle. The servos could move slightly past one another because of the nature of the construction of the pieces. The combination of the nuts and bolts holding two pieces together allowed for slight bending between the two pieces. Although this doesn't seem like much of a benefit, having three or four pieces being held together in such a way provided plenty of room for the servos to function as they needed to in order to lift a heavy object. Although the arm was not used, a much simpler version is the foundation of the 3D scanning laser range finder.

2
2D Scanning Options

2.1 Overview

In this paper, two potential sensors for range finding are explored. One is the sensor module that comes with the Bioloid Kit, and the other is Hokuyo's URG-04LX-UG01 laser range finder. The laser performed better by getting an accurate and linear result while the module returned a logarithmic result. Additionally, the laser did not skew much as the surfaces varied. Ultimately, the laser range scanner was chosen for it's basic scanning technology. As evidenced below, it proved to be much more accurate than the IR scanner and provided more data

2.2 Sensor Module

The AX-S1 is a sensor module that is included with Bioloid's Comprehensive Kit. The module is used to interact with the environment via range, light, and noise detection. Three sides of the sensor module have a range detector that measures the distance to an object, using values between 0 and 255. Moving an object away from any one of the sensors results in a value that starts high and drops down to zero for that sensor. The reason for

this decrease has to do with the way the sensor works. The detector is supplied a voltage and returns another analog voltage from the receiver. As the distance between the sensor and obstacle increases, the distance travelled from the transmitter to the object and back to the receiver also increases. The greater the distance between the two, the bigger the voltage drop, and so the closer to zero volts received. This receiving voltage almost directly corresponds to the number outputted.

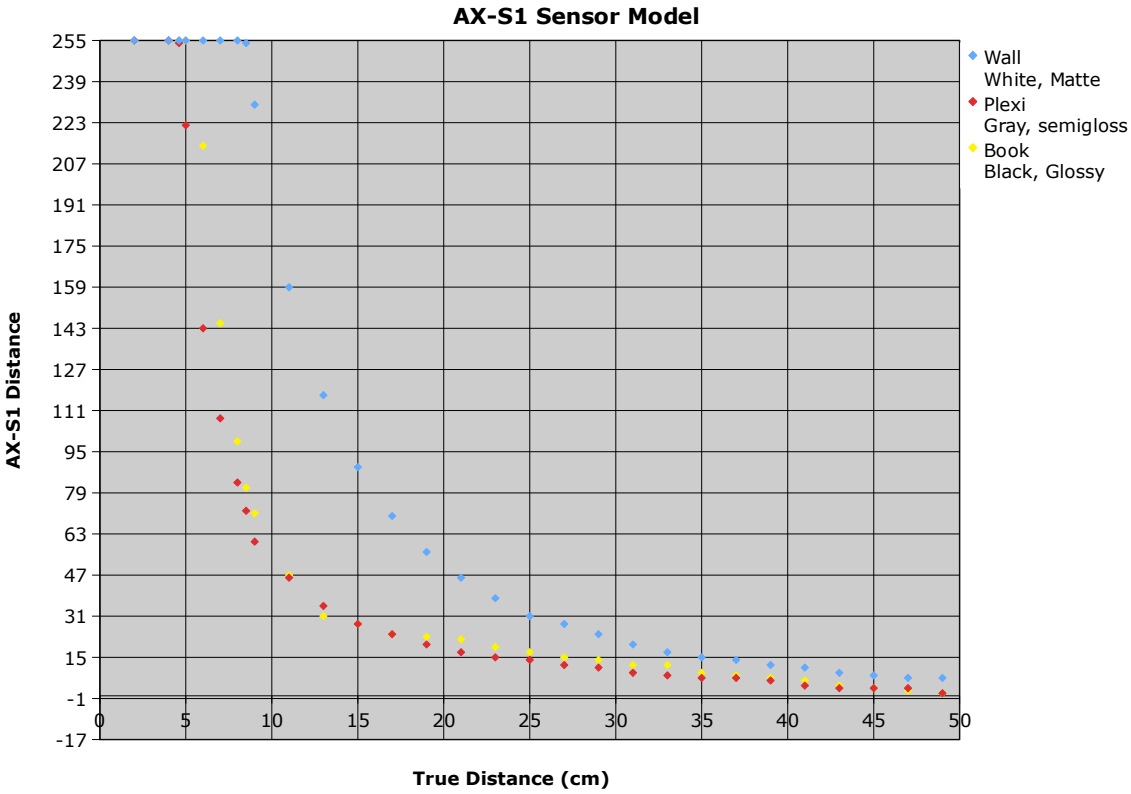

Figure 2.2.1. AX-S1 sensor model

Data was collected by varying the distance and materials of the surface. Each distance was measured ten times and each point was plotted, as shown in Figure 2.2.1. The type of surface affects the resulting distance measurement. Given the same distances, the values differ somewhat based on the color or reflection of the obstacle. It is clear from the figure

that as the obstacle gets glossier and darker in color, the value the sensor returns for the particular distance decreases. For example, at a distance of 25 cm away from the module, the values read are just under 15, just over 15 and 30, whereas at 11cm away, the values are 47,47, and 118. Comparison of these two distances shows just how much noise there is in the sensor.

2.3 Laser Range Scanner

Contrary to the sensor module, the laser range scanner was built to accomplish only the specific task of measuring distances, and so offers a much cleaner output. The laser used here is able to do a 240° scan of a plane in 100 milliseconds. Calibration for the measurement and conversion of the actual distance versus the distance read by the laser was based on the midpoint of the range it looks at - the point directly in front of the laser, effectively turing the 2D laser into a 1D sensor. Measurement and data collection for the calibration was done in the same way as the laser - each distance was measured ten times, recorded, and plotted. As shown in figure 2.3, even given the different surfaces of varying colors and gloss to skew the outcomes, the readings stayed fairly consistent. There was some inconsistency in the readings that increased as the object moved farther away from the source, but it stayed within a fairly small range.

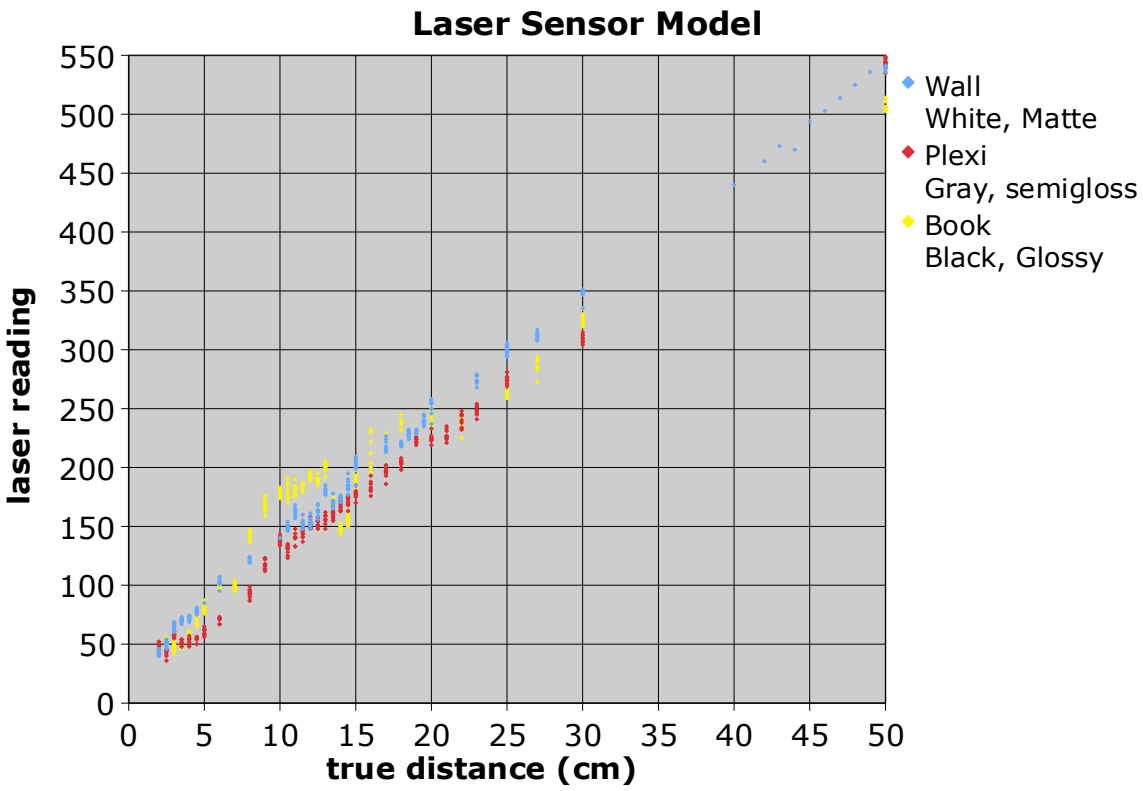

Figure 2.3.1. Laser sensor model

3
3D Scanner Platform

3.1 Hardware

3.1.1 Laser Rig

Figure 3.1.1. Laser Rig: Laser attached to control module via a servo

The Laser Rig consists of four parts. The first is the 2D laser range finder itself. It is what actually collects the data in a plane. The second is the servo which allows tilting

of the laser. The third is the connector piece between the two previous parts. The plate attaches the laser to the servo allowing it to tilt. Lastly, the CM-5 is a controller that allows for the control and communication of the servo with the computer.

3.1.2 Bioloid Overview

The Bioloid Kit is a robot kit designed to allow users to assemble robots to accomplish almost any task. The kit allows for alteration of its physical and behavioral patterns much in the same way as Lego or Erector kits do. It allows for pieces to be put together with only nuts and bolts to produce a wide variety of robots. The Bioloid kit uses a series of serially connected servos and sensor modules all connected to a central controller that sends and receives information.

3.1.3 Bioloid Controller

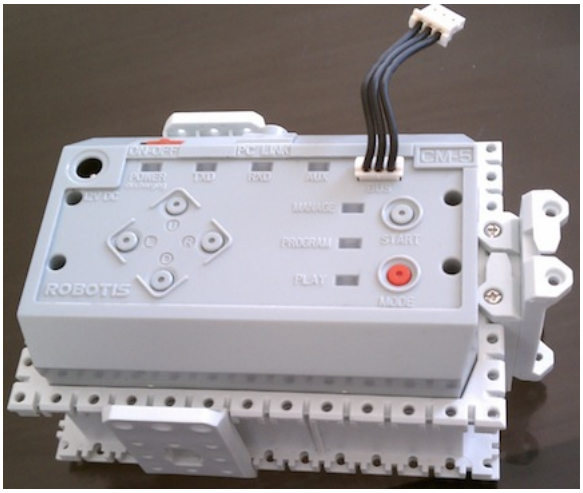

Figure 3.1.2. CM-5

The CM-5 is the controller module and the brain of the whole kit. The CM-5 comes with different modes, buttons, and LEDs for user interaction. Communication between the CM-5 and a computer is accomplished with a serial cable, but can also be achieved with a USB adapter. Once the program is loaded, the CM-5 talks to the servos and sensor modules

with serial cables that can be daisy-linked together using anything from one to three buses.

The module comes with three different modes, each serving a unique function. The play mode is used to run the residing program once it has been downloaded. The program mode allows for the CM-5 to communicate with the computer as a motion editor. Lastly, the manage mode is used to verify and run programs on the fly. It allows for a connection with a computer through which code can be directly executed and sent to the CM-5 without downloading.

There are also four buttons of the module. Each can be programmed to perform a certain task, as desired by the user. Lastly, there are three state indicating LEDs. Each is responsible for showing the user what the module is doing, similar to the status LED on a desktop.

3.1.4 Bioloid Servos

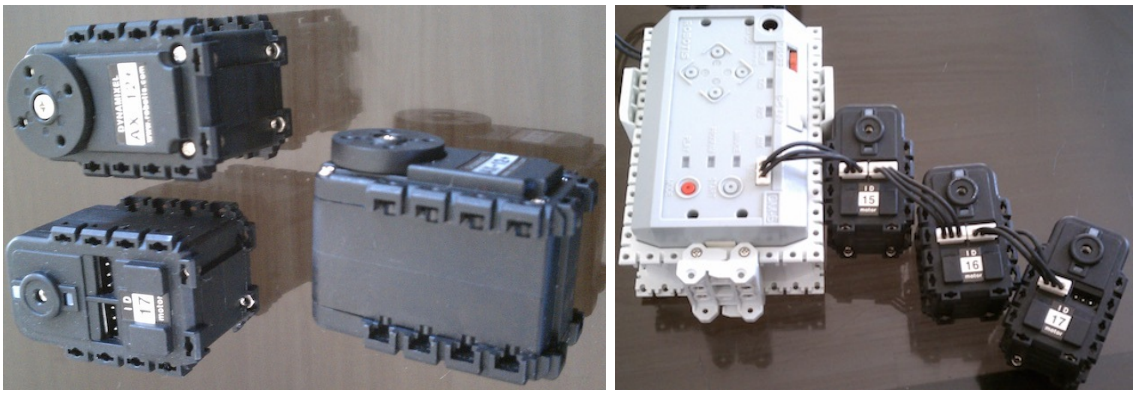

Figure 3.1.3. Left: Different sides of the servo, Right: Servos serially linked to the CM-5

Motion is provided by the AX-12+ servos that are serially controlled. Whenever there is an action sent to a particular servo, feedback is provided about the relevant information. Some of these characteristics include continuous rotation, goal position, moving speed,

temperature, voltage, and LED. The LED is used mostly for the purposes of debugging – it can be turned on and off, but has no actual functionality. Rotation is set either as an endless turn or it is set to a desired goal location. With the latter, the servo has 300° of rotation and corresponds to a value range of 0 to 1023. The center resides at position value 512 and is considered to be at 150°. If it is the case that the angle limit for both directions is set to 0, then it is possible to implement an endless turn mode. This is only possible if the speed is set without a limit. This allows for the use of a continuous rotating wheel. Temperature and voltage are both used as safety methods, and if either exceeds the preset limits, the servo shuts off.

Write Address	Writing Item	Length (bytes)	Min	Max
3(0X03)	ID	1	0	253(0xfd)
4(0X04)	Baud Rate	1	0	254(0xfe)
5(0X05)	Return Delay Time	1	0	254(0xfe)
6(0X06)	CW Angle Limit	2	0	1023(0x3ff)
8(0X08)	CCW Angle Limit	2	0	1023(0x3ff)
11(0X0B)	the Highest Limit Temperature	1	0	150(0x96)
12(0X0C)	the Lowest Limit Voltage	1	50(0x32)	250(0xfa)
13(0X0D)	the Highest Limit Voltage	1	50(0x32)	250(0xfa)
14(0X0E)	Max Torque	2	0	1023(0x3ff)
16(0X10)	Status Return Level	1	0	2
17(0X11)	Alarm LED	1	0	127(0x7f)
18(0X12)	Alarm Shutdown	1	0	127(0x7f)
19(0X13)	(Reserved)	1	0	1
24(0X18)	Torque Enable	1	0	1
25(0X19)	LED	1	0	1
26(0X1A)	CW Compliance Margin	1	0	254(0xfe)
27(0X1B)	CCW Compliance Margin	1	0	254(0xfe)
28(0X1C)	CW Compliance Slope	1	1	254(0xfe)
29(0X1D)	CCW Compliance Slope	1	1	254(0xfe)
30(0X1E)	Goal Position	2	0	1023(0x3ff)
32(0X20)	Moving Speed	2	0	1023(0x3ff)
34(0X22)	Torque Limit	2	0	1023(0x3ff)
44(0X2C)	Registered Instruction	1	0	1
47(0X2F)	Lock	1	1	1
48(0X30)	Punch	2	0	1023(0x3ff)

Figure 3.1.4. Control table of data ranges [2]

3.1.5 Bioloid Sensor

The AX-S1 module provides of a variety of sensors. It has functions for sensing distance, brightness, remote control infrared, heat, and sound. The module can also generate any combination and sequence of 52 pitches of sound.

Both distance and brightness can be measured from three of the sides of the sensor, whereas sound and remote control reception are done from only one side. Both brightness and distance are measured as a value from 0 to 255. The distance sensor can provide close

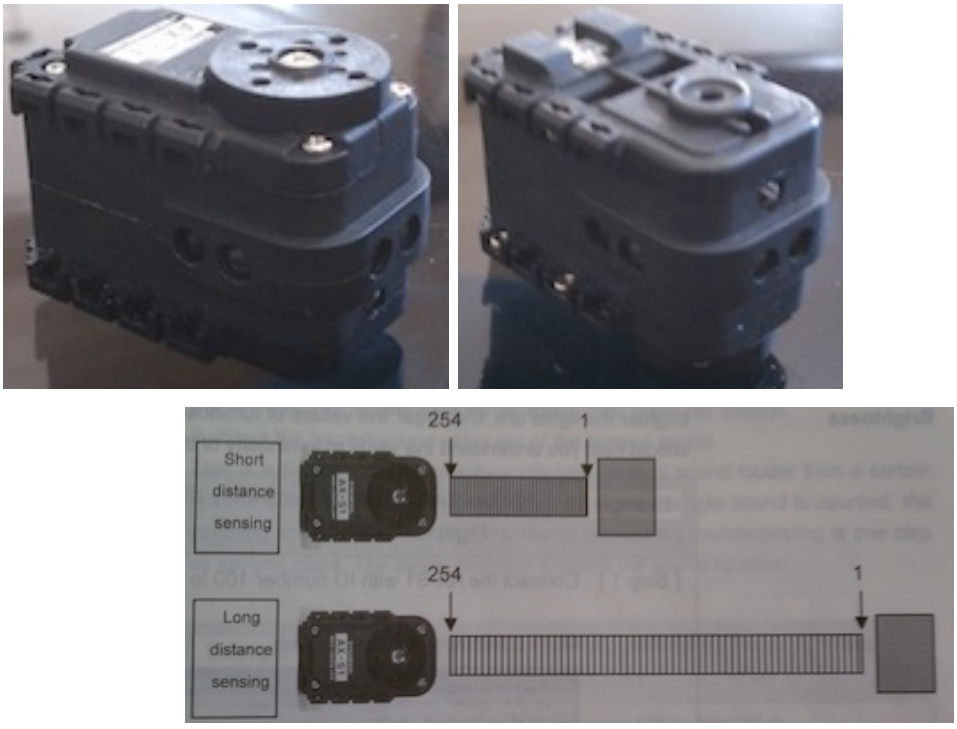

Figure 3.1.5. Top: Sensor module, Bottom: Close and long range sensing [1]

and long range sensing. The short range offers a more precise value at close proximity, while the long range allows for sensing farther away.

3.1.6 Hokuyo Laser

Laser range finders are extensively used in object recognition and modeling. The laser used in this project was a $1,200 Hokuyo laser range finder. It uses an infrared laser and calculates the distance based on the phase shift. It uses a rotating mirror to do the scanning in a plane. It is able to scan 240°, and has a range of about 20 centimeters to upwards of 5 meters, but is guaranteed to give an accurate measurement for ranges up to 4 meters.

After a planar scan, the laser returns an array of values. As shown in Figure 3.1.6, there are over seven-hundred steps, with the middle value being step 384. The laser obtains the

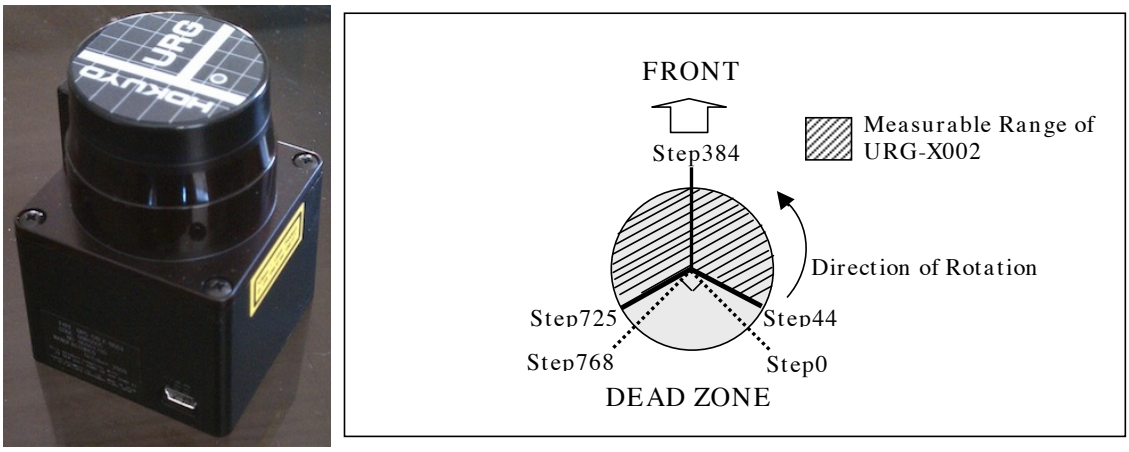

Figure 3.1.6. Left: Laser range scanner, Right: Measurable range of the scanner

distance between itself and the first obstacle it finds, places that value in the step it is in and moved to the next step to repeat the process.

3.1.7 Connector

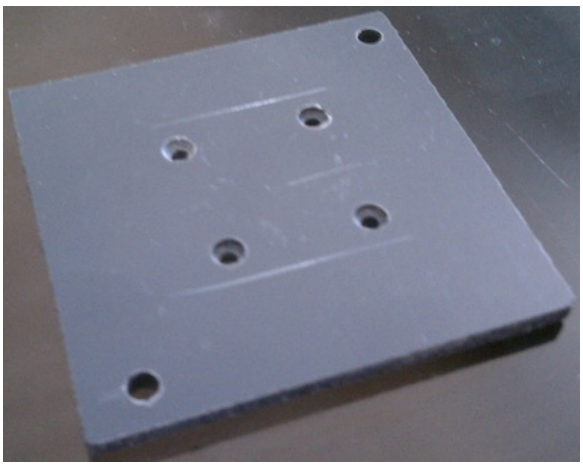

Figure 3.1.7. Plate connector

Both the Bioloid Kit and the laser have screw mounts. The kit uses a series of tiny nuts and bolts of ranging lengths to hold two pieces together. The laser uses two screws on the opposite sides of its base in order to be attached onto a plate of some sort. To connect

the two pieces, a joining connector plate needed to be manufactured. A piece of custom machine shopped plexiglass was used. Six holes were precision drilled, two for the laser and four to attach to the kit. Because the laser and kit were on opposite sides of each other, the four holes needed countersinks to keep the plate flush with the connector.

3.2 Software

3.2.1 Bioloid Standard Applications

The Bioloid kit comes with three standard computer programs used to create programs for robots. Each one is unique to it's main function and purpose: The Behavior Control Programmer (BCP), the Motion Editor (ME), and the Robot Terminal (RT). The BCP is a graphical user interface to create programs. It uses icons that can be placed, one below the other, to create a list of commands to be executed. Similar to any programming language, there are loops, branches, and conditions. The ME allows for low-level communication. It controls individual servos to create actions that are very precise. The actions are presented in a 3D interface and can be accessed by the BCP. The RT is a lower-level communication application that talks directly to the servo through the CM-5. The Robot Terminal is similar to the terminal application in a Mac and the Command Prompt on Windows, but communicates only with the Bioloid.

3.2.2 Python Libraries

Both the Bioloid kit and the laser scanner come with their own libraries. The kit comes with three software applications and is programmed in C. With the help of Doug Blank at Bryn Mawr College, communication with the CM-5 in Python was established. Blank provided a skeleton that gave basic functionality to the servos.

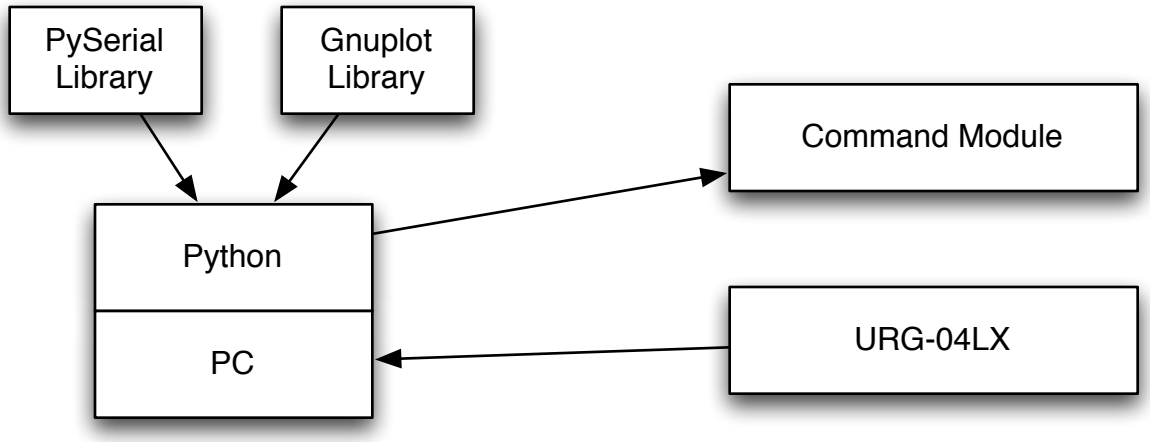

Figure 3.2.1. System-Level Block Diagram

A Python library that communicates with the laser was found on github [1]. Mainly, this library is used to read the array of distances measured in a scan. Other interesting functions include the ability to turn the laser on and off, access various information such as version and parameters.

4
3D Model Building

4.1 Algorithm

Stitching together a series of 2D scans by far the most interesting and difficult part of the project. The raw information was provided by two sources that don't mix particularly well. The problem was that the data was captured in different coordinate frames, and needed to be integrated into a common frame. Looking at the laser alone, it provided an array of distances. The azimuth, the angle between two points at which the distance is measured, was known, and therefore a 2D polar plot could be made in that plane. With the addition of a servo, a third dimension was given. Given the three dimensions, the points exist in spherical coordinates. Conversion to Cartesian coordinates proved to be a big challenge. After a series of tries of different algorithms, a new approach was taken.

The servo tilts in the X-Z plane. If the laser is flat it scans in the X-Y plane, and has no variation among any of the points in the Z-plane. If, however, it is facing up, then it scans in the Y-Z plane, and has no variation in the X-plane. Given this, X and Z-planes are opposites of each other. With the rotation of the servo, the angle was ignored and labeled

the "B-axis." This B-axis is thought as a plane that rotated anywhere about the Y-axis If the laser captures a scan in this B-Y plane, it will simply create a polar plot. Conversion from this "B-Y plot" into an X-Y-Z plot, was accomplished using the following algorithm:

```
while (servo moving):
     tilt = angle of B-axis above the X-axis
     for i in length of all the distances in a planar scan:
          x = i * [cos(i) * (azimuth angle)] * [sin (tilt)]
          y = i * [sin(i) * (azimuth angle)]
          z = i * [cos(i) * (azimuth angle)] * [cos (tilt)]
```

Once the conversion is complete, it is then visualized in two different programs. Gnuplot, which is integrated into Python, reads the converted file and plots all the points. The points are colored based on their distance in the X-axis. The user can rotate the plot in any direction they wish. The second visualization is done in Processing, a Java-like programming language that plots all the data and rotates the captured environment around the Z-axis.

4.2 Early Results and Problems

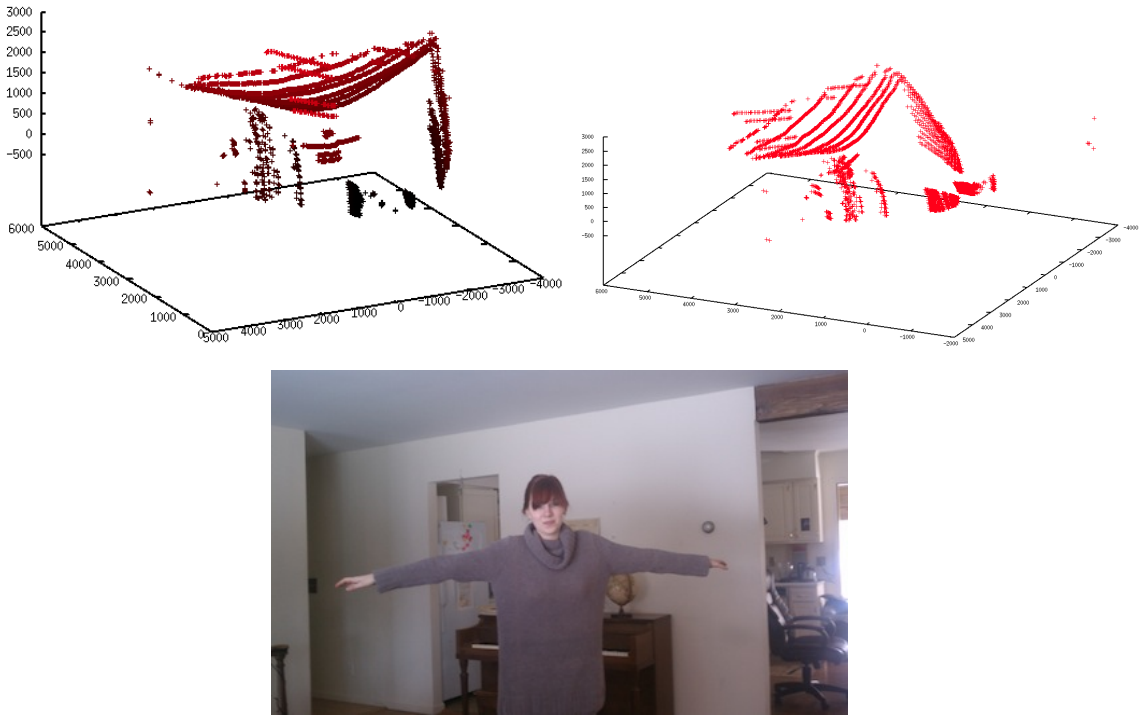

Figure 4.2.1. Model standing in the middle of the room

The scans above (Figure 4.2.1) show the range scanner in the corner of a room pointing at a person standing in the middle of the room with their arms out, perpendicular to their body. This scan was done using an early algorithm. This scan confirms that the algorithm was roughly correct as it presents a few similarities to reality. Having the individual put their arms out shows that the resemblance is fairly accurate. The slant in the ceiling is not the poor construction of the house but rather an early stage in the algorithm. The early stage of the algorithm was missing components that would correctly convert the frames into the X-Y-Z axes.

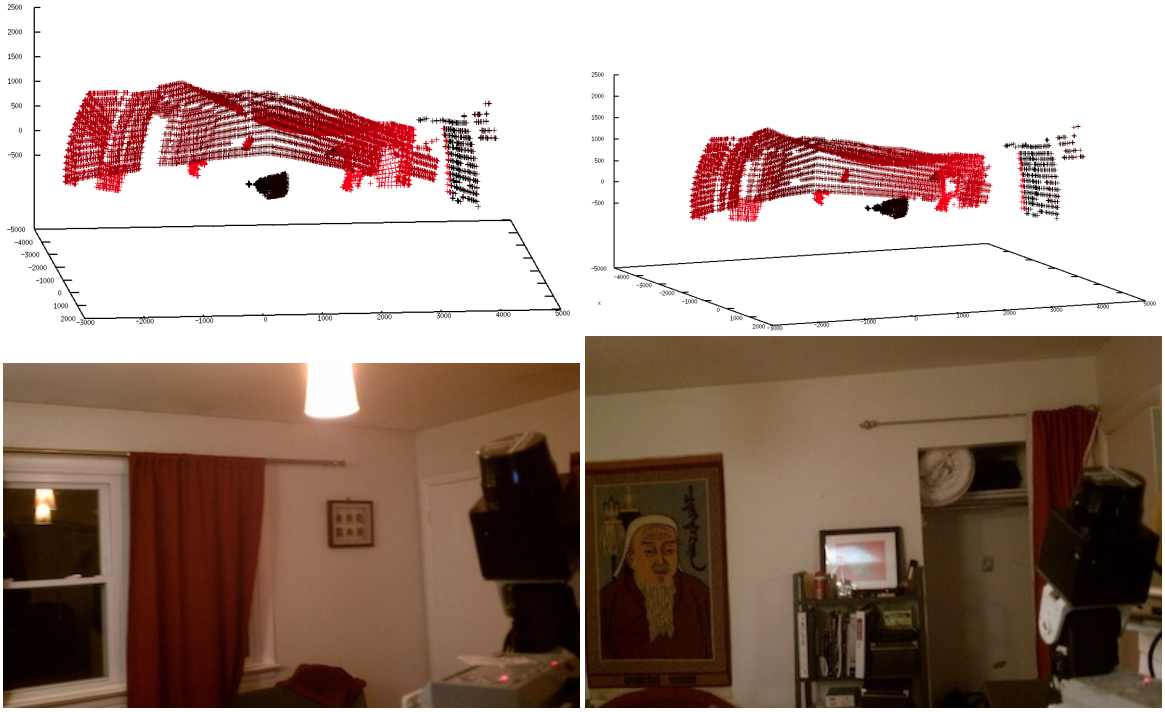

Figure 4.2.2. Far wall and corner of a room

With some basic orientation established, improvement came next. Figure 4.2.2 shows an attempt to improve on the straightness of flat surfaces. Some of the extremeness of the sagged ceiling is gone. The geometry of the window, which appears as a hole in the scan on the left side of the room, seems to be near rectangular but almost perfect with the shape of the room. The break between the room and the scan on the right is created by a closet whose back wall is outside of the range of the scanner.

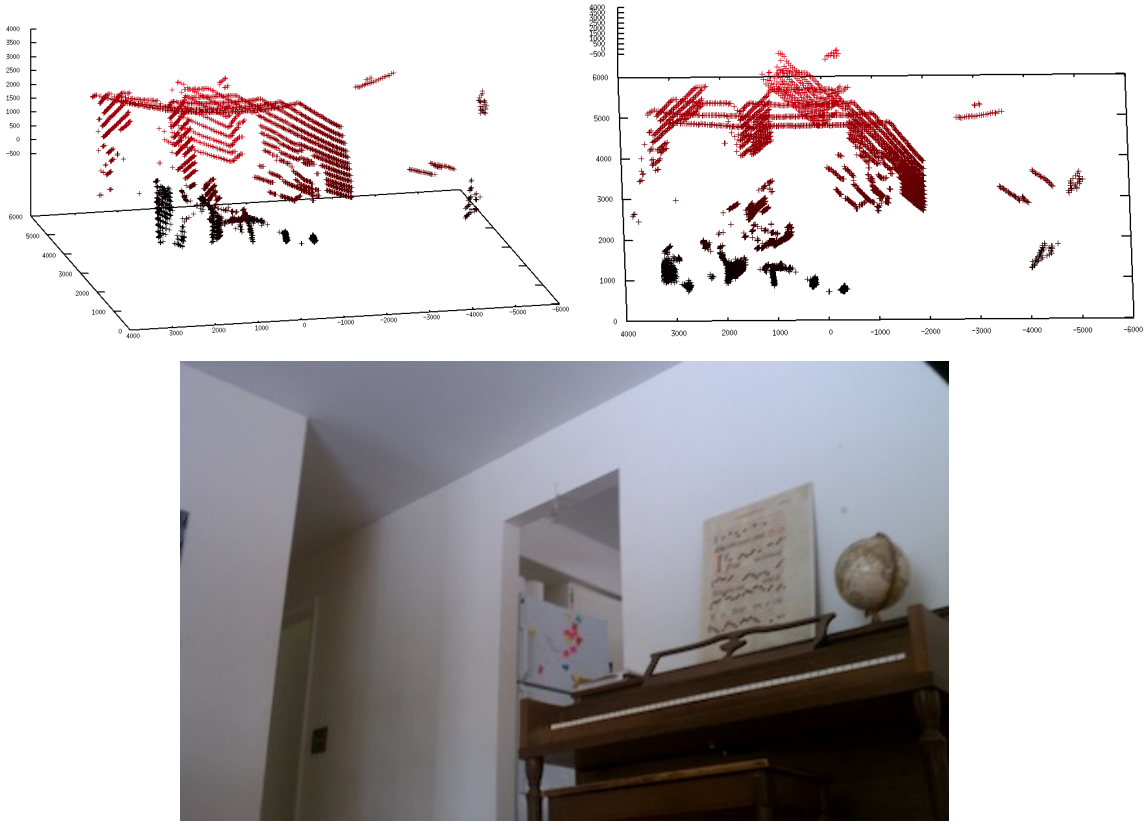

Figure 4.2.3. Corner of a room leading to two directions

The recreation of the environment shown in Figure 4.2.3 presents a curious and somewhat complicated scan. The early algorithm used to create this model shows the piano near the corner of a room. That corner also leads to another room via an archway on one side and into the hallway on the other. The points protruding from the right side of the wall represent a piano. The scan was taken from the ground with a fairly big rise between each scan, and so the piano does not have many points to accurately portray it.

Figure 4.2.4. Corner of the sun room

In Figure 4.2.4, the walls became more of a challenge and hence more engaging. There is a corner that protrudes in the middle of the scan. To the right is a wall and a window. Even further to the right, a part of a nook with a table starts appearing. Looking closely, it is possible to notice that the part where the edge of the table appears (see later for the picture), the scans shows to be somewhat protruded from the wall.

4.3 Finalized Algorithm

The following scans use the finalized algorithm. The environments are each scanned at four different speeds. Each scan is visualized and two different views are shown to better

represent the scan. The speeds determine how fast the servo moves, which in turn determines how much data is collected. Each subsequent speed is roughly twice as fast as the previous and so collects almost half as much data. At the slowest speed, the amount of data collected is quite large. The average file size is about 4.5MB. Looking at the number of points, the most of the environments hover at about 100,000 points with the larger files being just over 130,000 points.

Figure 4.3.1. Living Room Panorama

The scans, as shown in Figures 4.4.1 - 4.4.4 capture the environment pictured above (Figure 4.3.1). Each pair shows the same environment as the servo speed increases. Comparing each clearly shows how much information is lost by doubling the speed. At speed 10 it is near impossible to even tell where the windows are. The scan shows an armchair in the middle of the room and a sofa to it's left. There is a window below the couch as well as another to the right of the armchair. Opposite the couch is an archway. The scan offers a variety of interesting objects of various difficulty. The windows, walls, and ceiling

confirm the geometry and correctness of the coordinate conversion. The armchair is by far the most fascinating and complex part of the environment as it has curved edges and uneven surfaces.

Figure 4.3.2. Model standing in the middle of the room

The laser range finder captures a person in the middle of the room. The model is standing in the same place and has her hands out just as an earlier capture had. This scan is taken to show the improvement over the early algorithm. Figure 4.3.6 visualizes the capture in Gnuplot so that the two can be compared. This capture is done using a servo speed of two. As figures 4.4.5 - 4.4.8 show, the ceilings and walls are all flat, and the model is set away from all the walls.

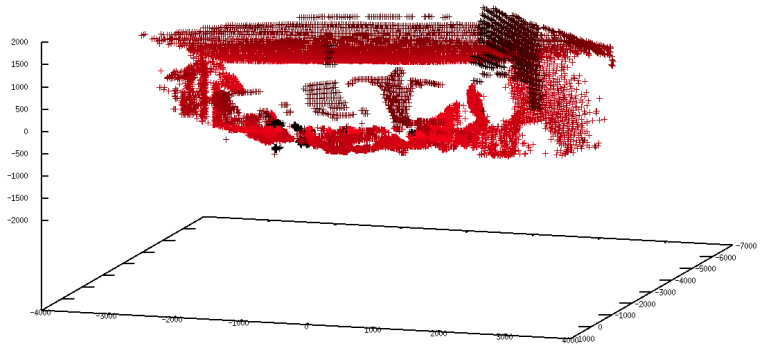

Figure 4.3.3. Scan of the model standing in the middle of the room using the finalized algorithm

Figure 4.3.4. Weis Cinema

This environment captures the walkway and chairs in Weis Cinema. As the visualizations show in figures 4.4.9 - 4.4.12, the chairs line up perfectly. Similar to other environments, the density of the scans reduces and detail is lost as the speed at which the servo moves increases.

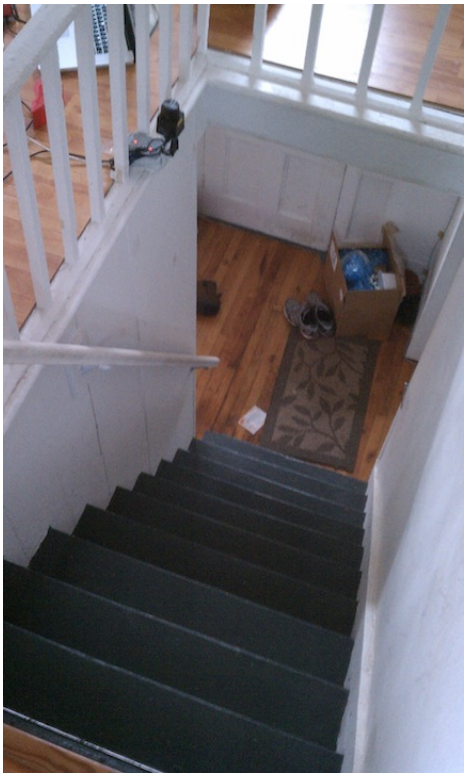

Figure 4.3.5. Stairs

The stairs data set, as pictured in Figure 4.3.5, is the most impressive scan in the project. As shown in Figures 4.4.13 through 4.4.16, the environment captures the stairs going down, and a room at the landing. Like previous scans, this shows four speeds of the servo and two views of each speed. Similarly, it also shows that as the speed increases, the number of points decreases. A noteworthy observation about this environment is that parts of it arguably look better at a speed of 2. At the slowest speed, the top few stairs look busy and the start to blend with each other. Overall though, this environment does not need all the points - the risers between each step can be blank as long as the step itself is visible.

4.4 Scans

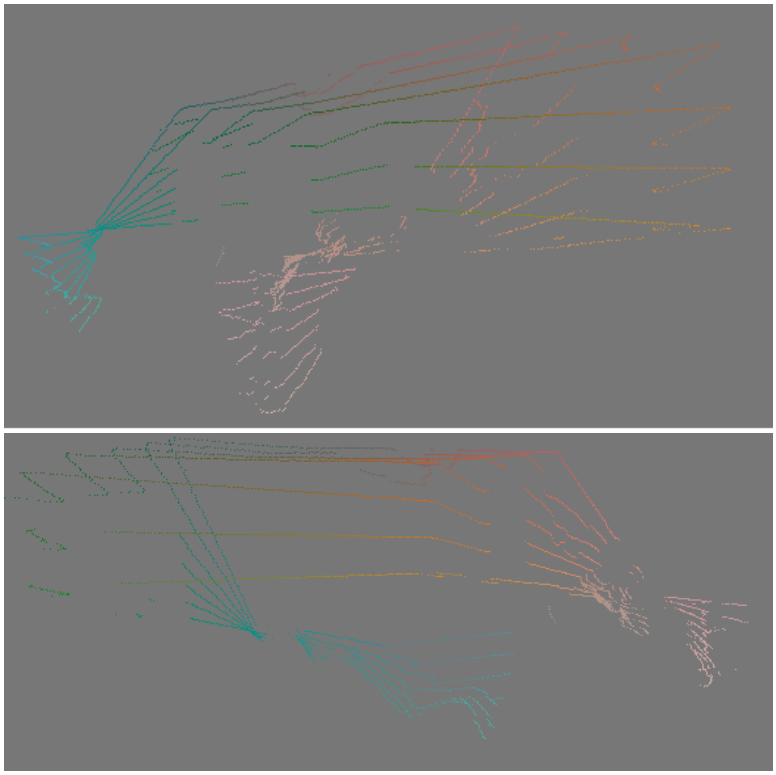

Figure 4.4.1. Armchair 10

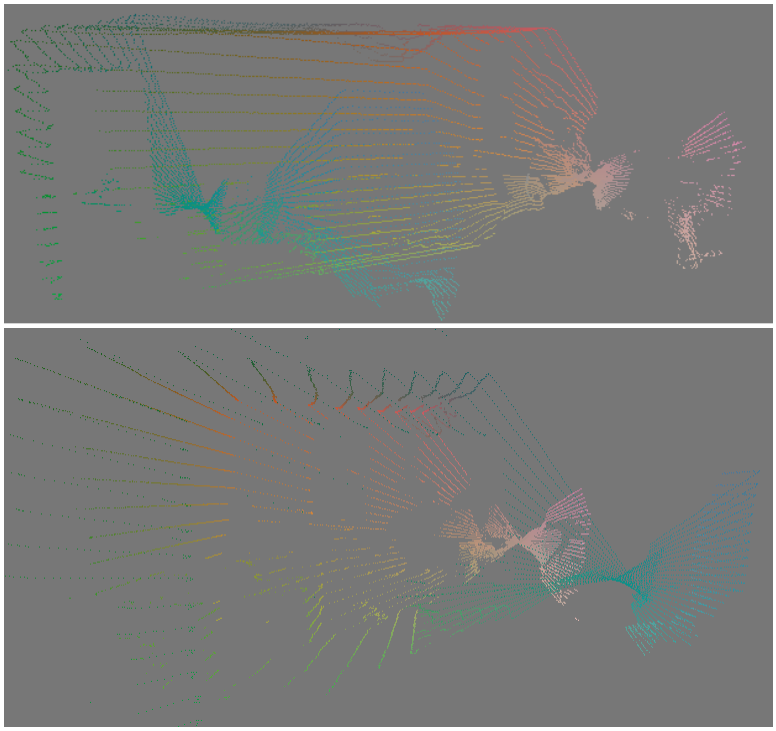

Figure 4.4.2. Armchair 4

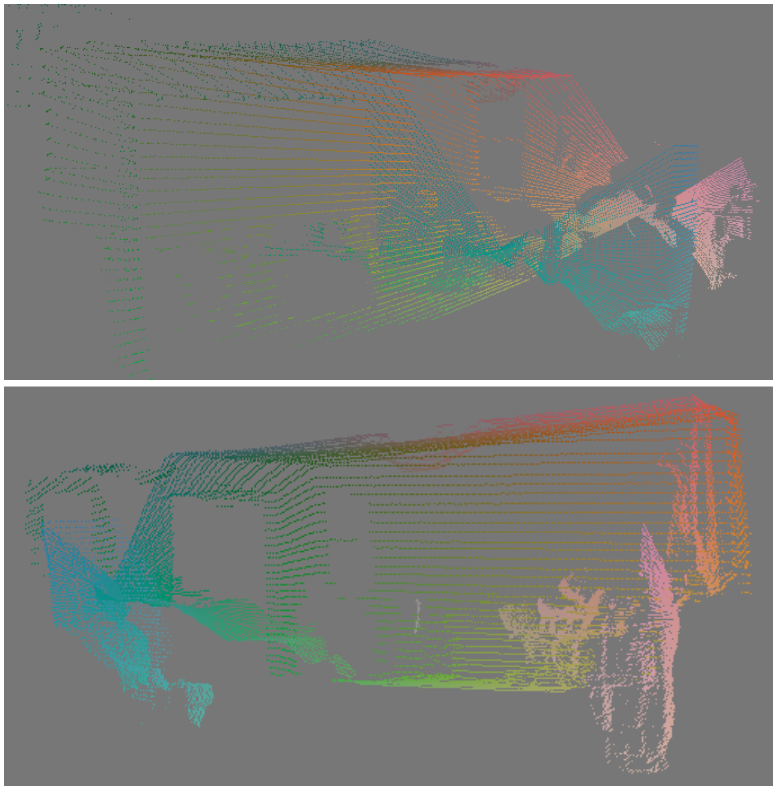

Figure 4.4.3. Armchair speed 2

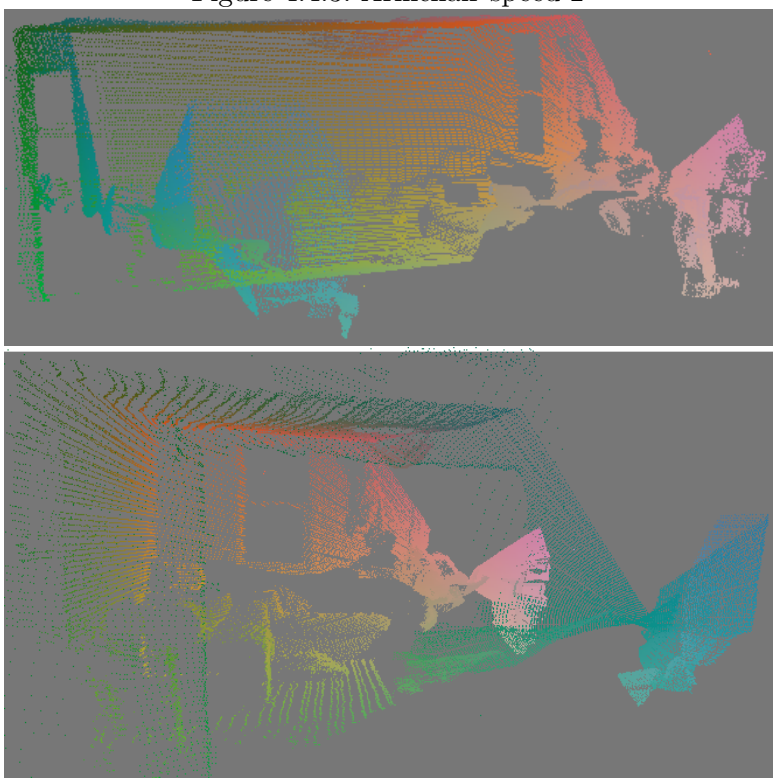

Figure 4.4.4. Armchair speed 1

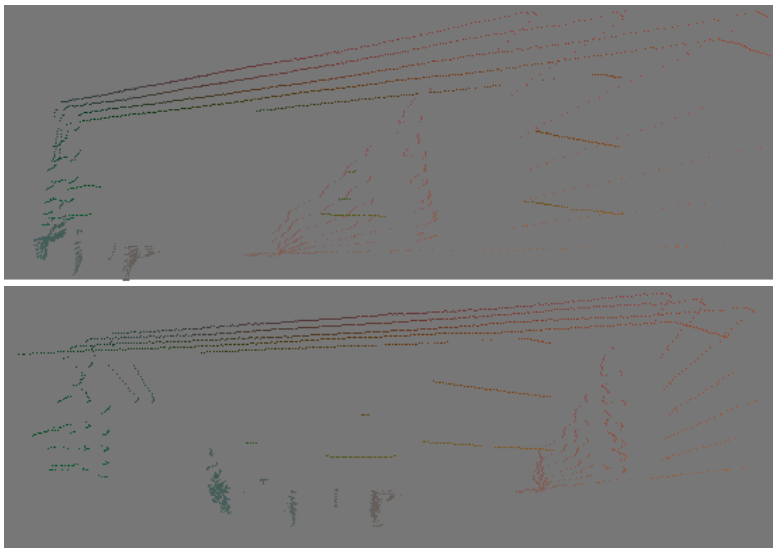

Figure 4.4.5. Model speed 10

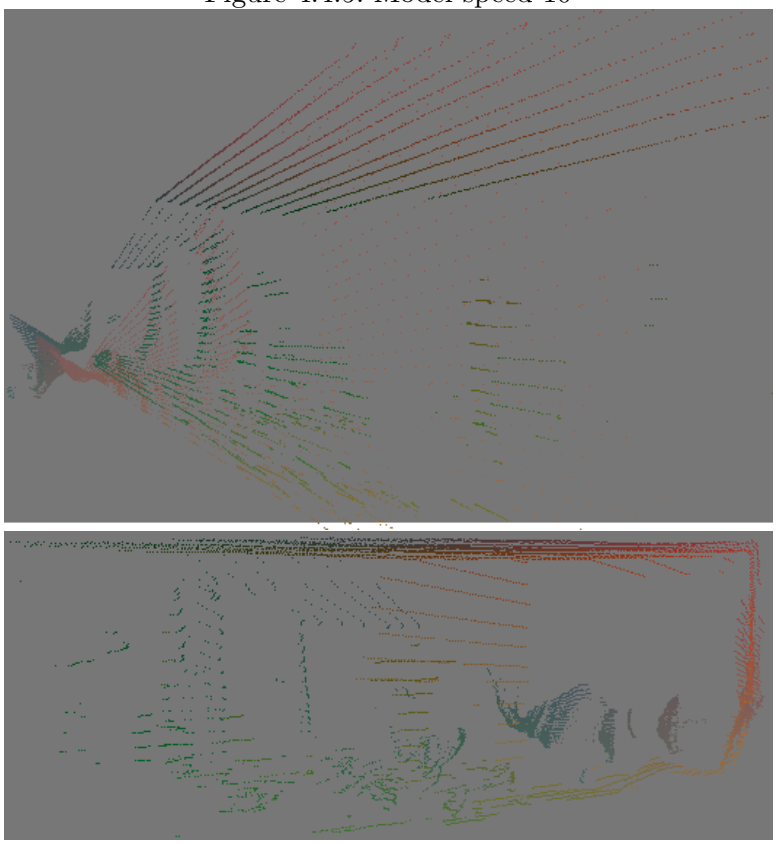

Figure 4.4.6. Model speed 4

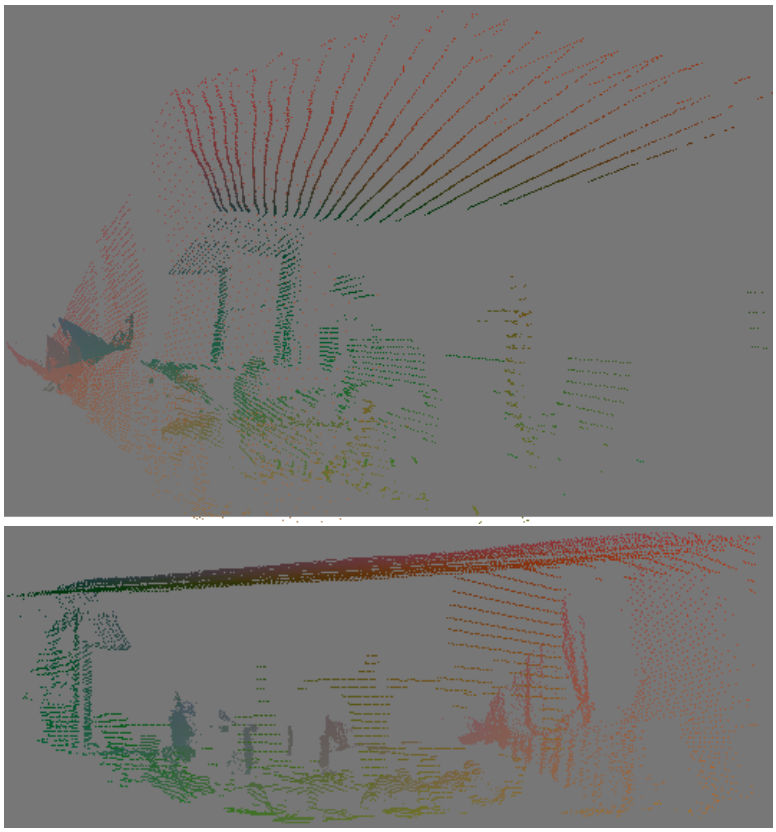

Figure 4.4.7. Model speed 2

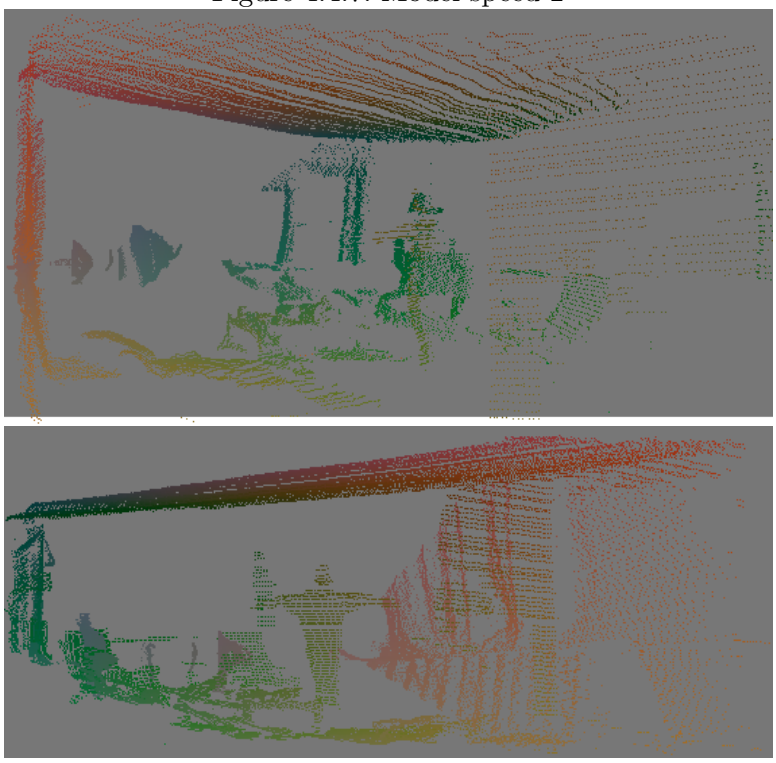

Figure 4.4.8. Model speed 1

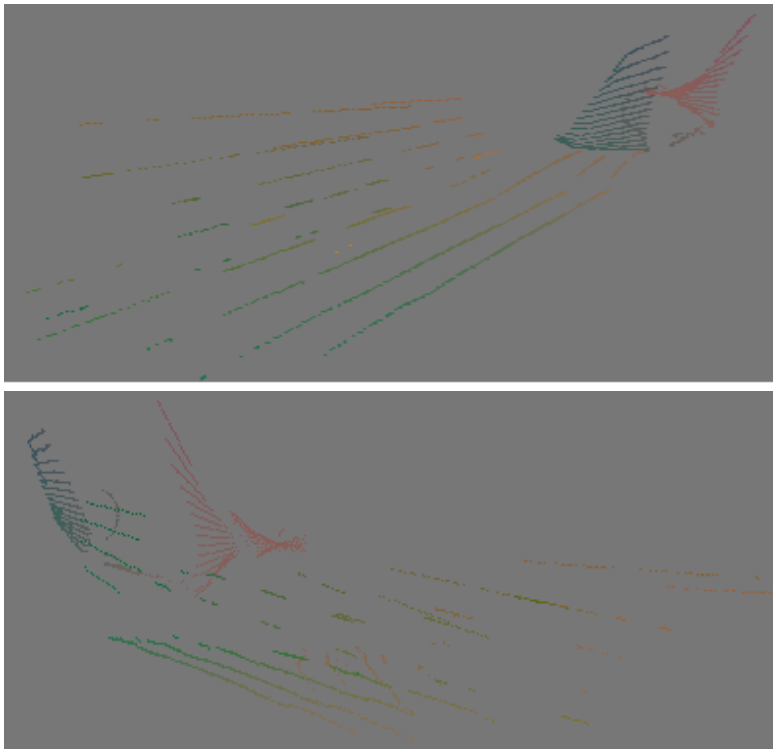

Figure 4.4.9. Weis speed 10

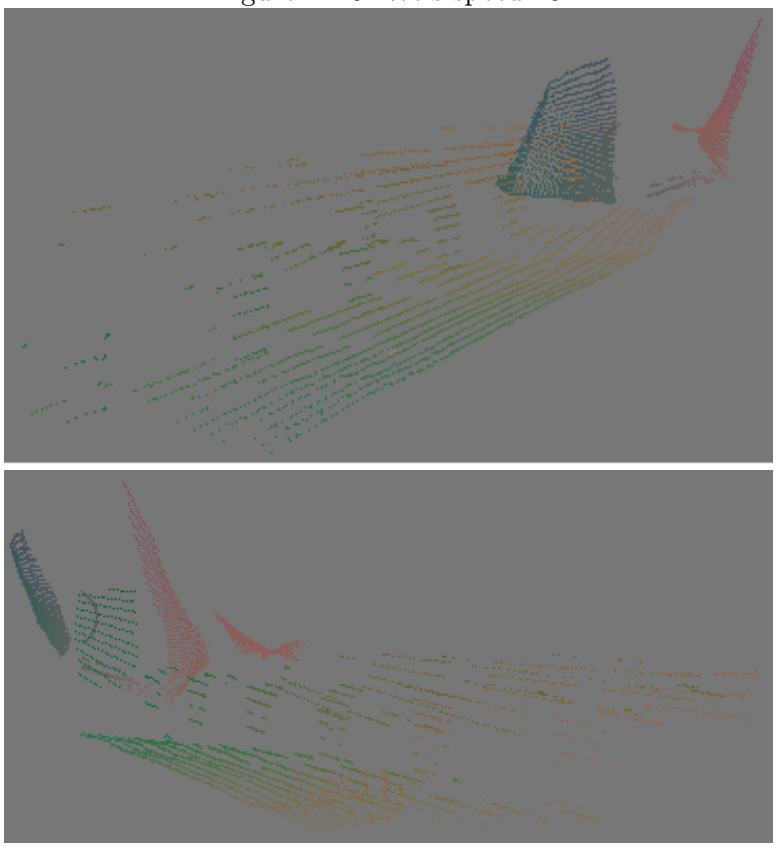

Figure 4.4.10. Weis speed 4

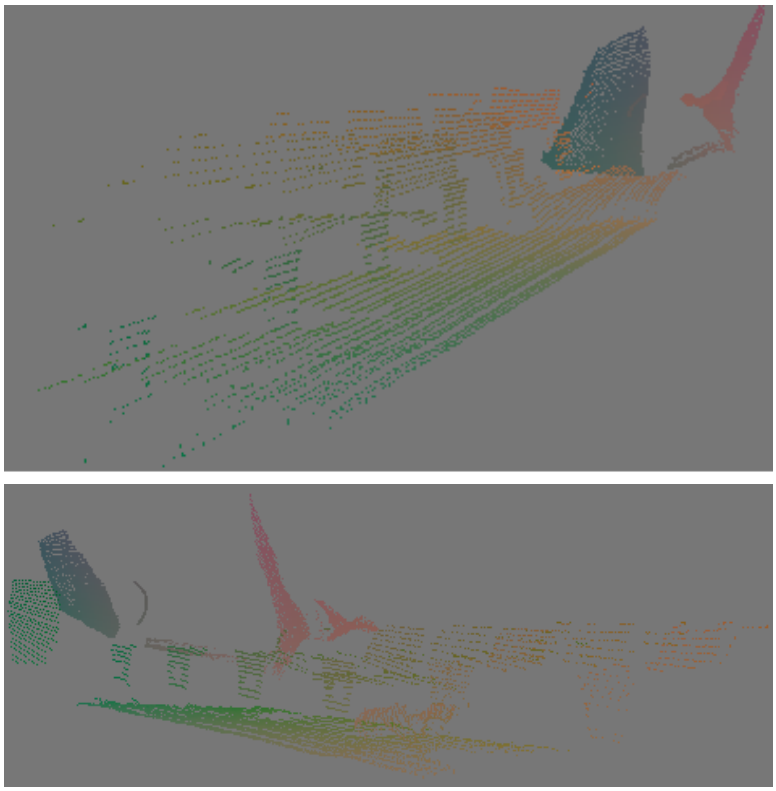

Figure 4.4.11. Weis speed 2

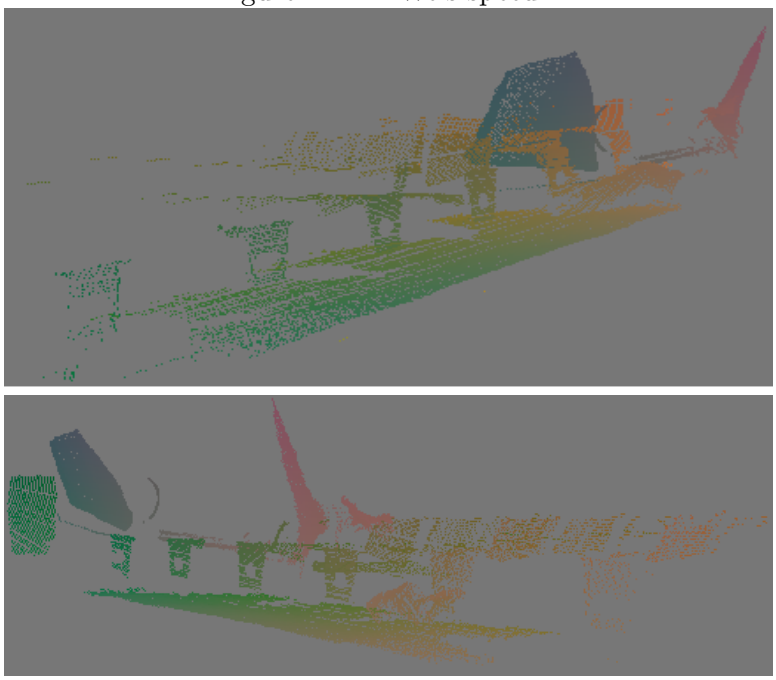

Figure 4.4.12. Weis speed 1

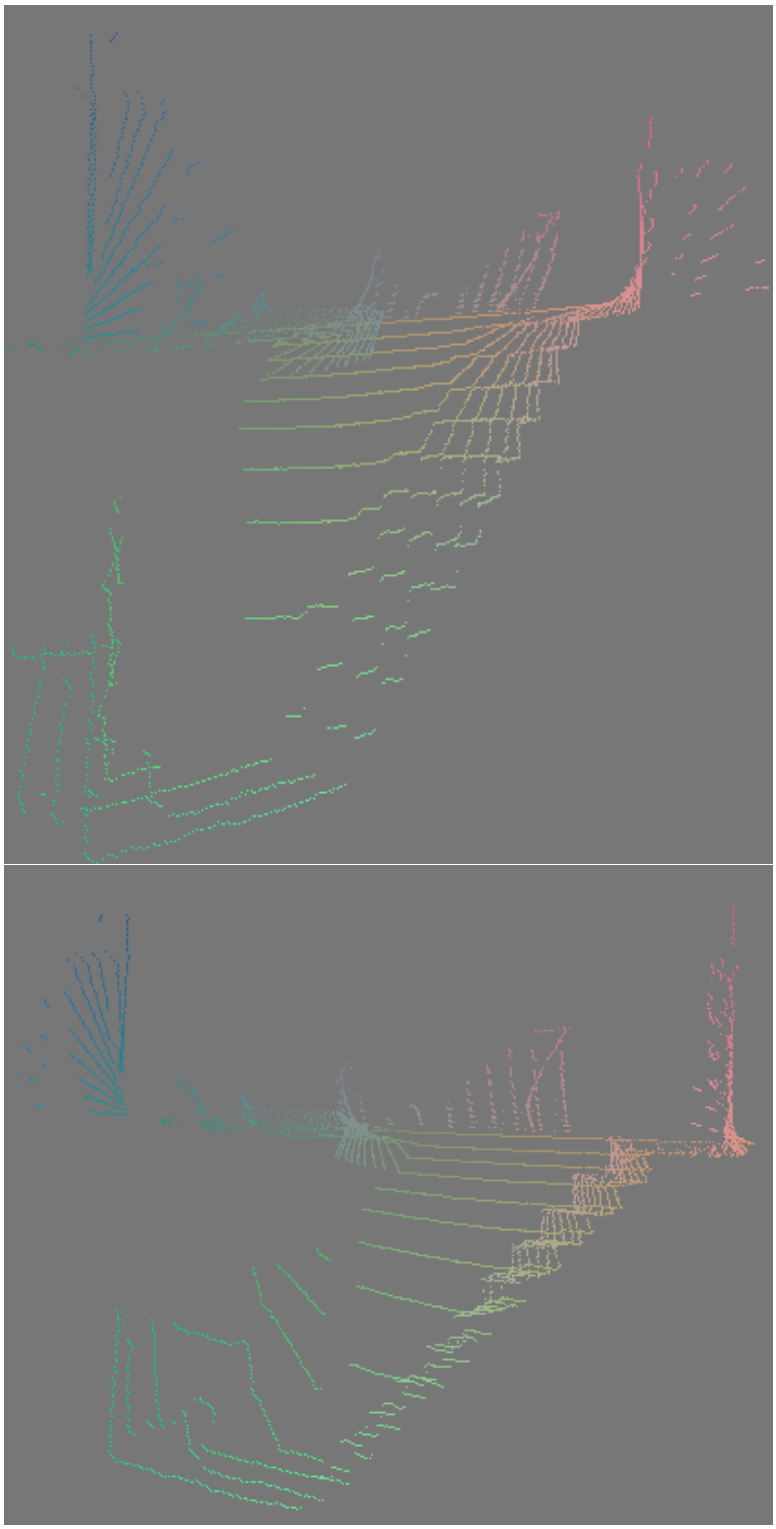

Figure 4.4.13. stairs Speed 10

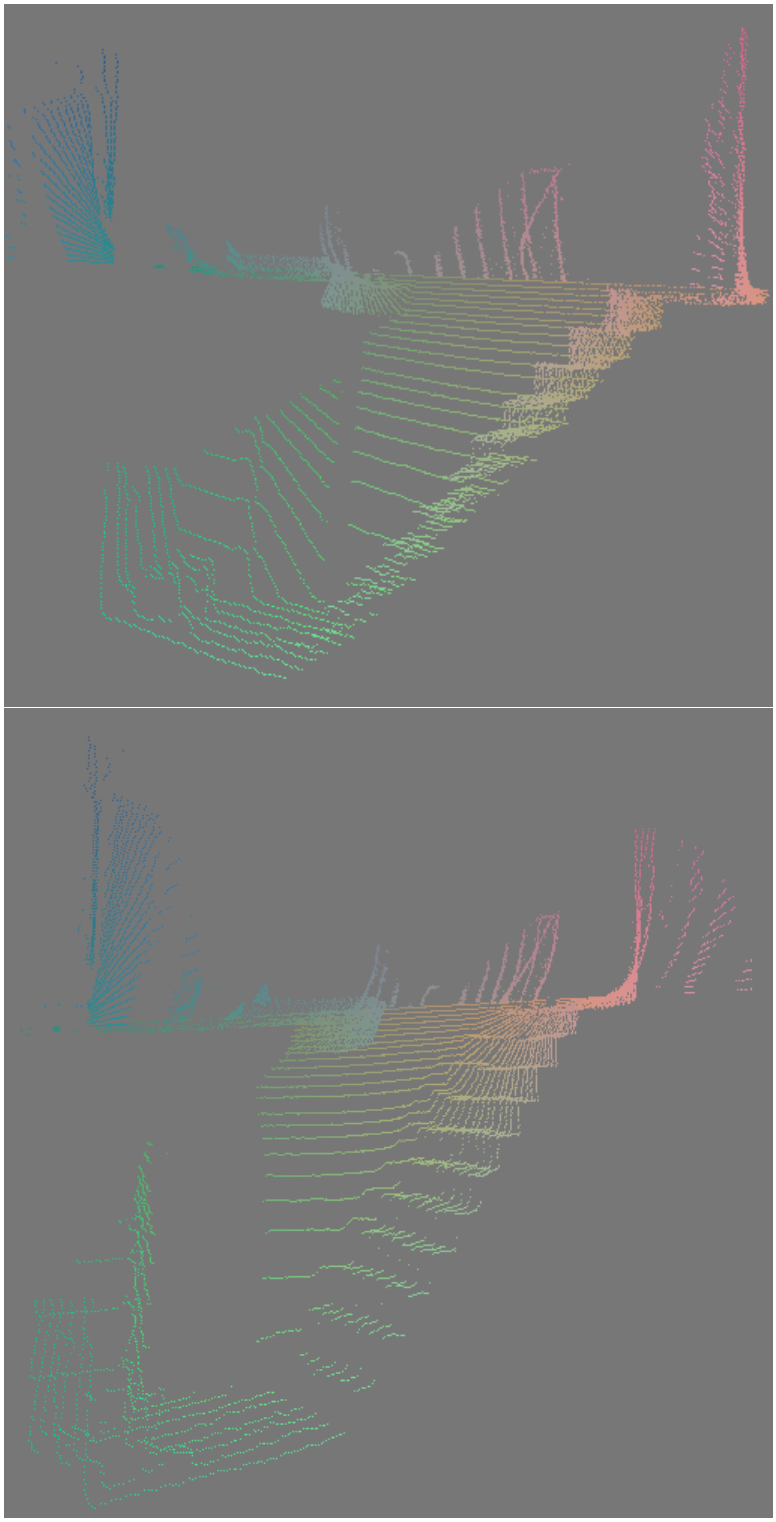

Figure 4.4.14. stairs Speed 2

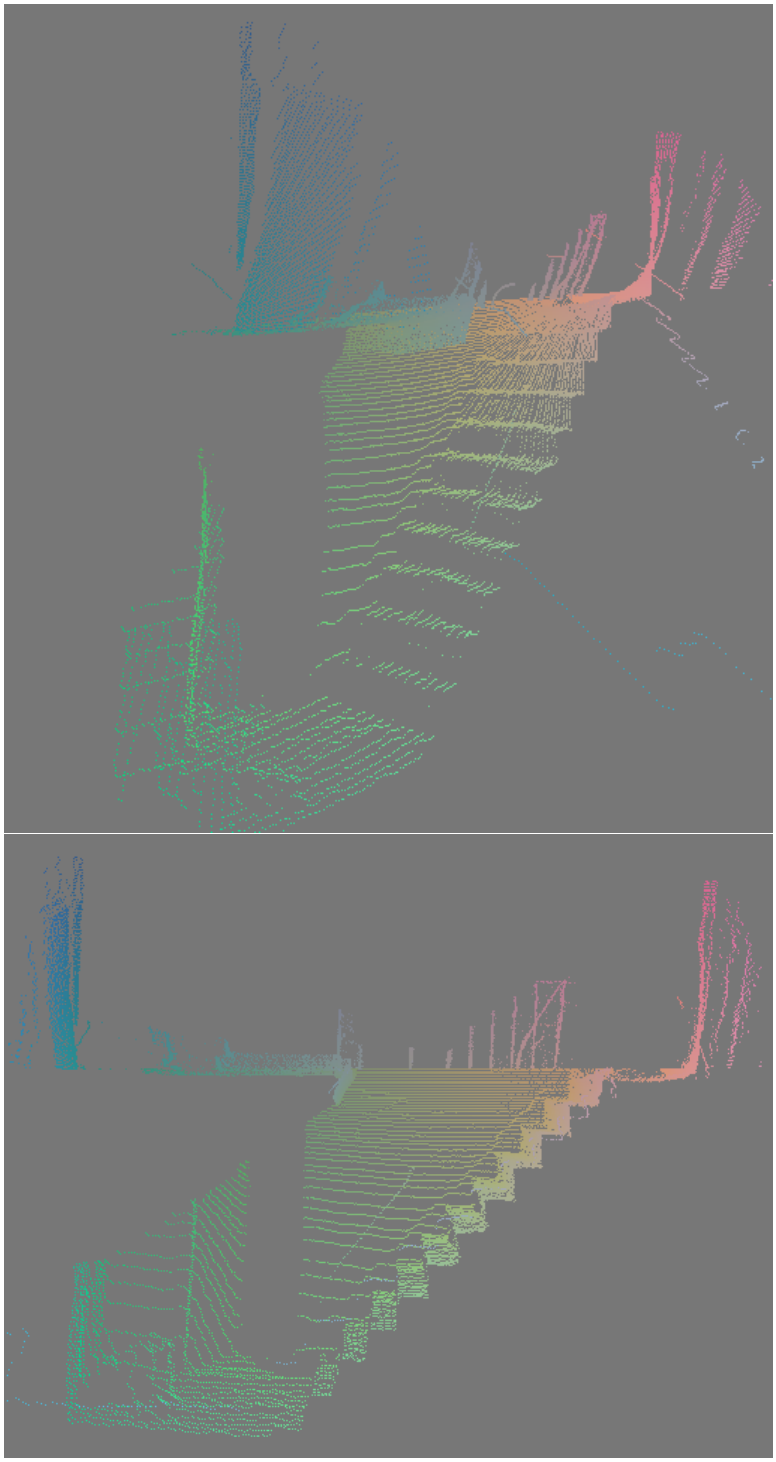

Figure 4.4.15. stairs Speed 2

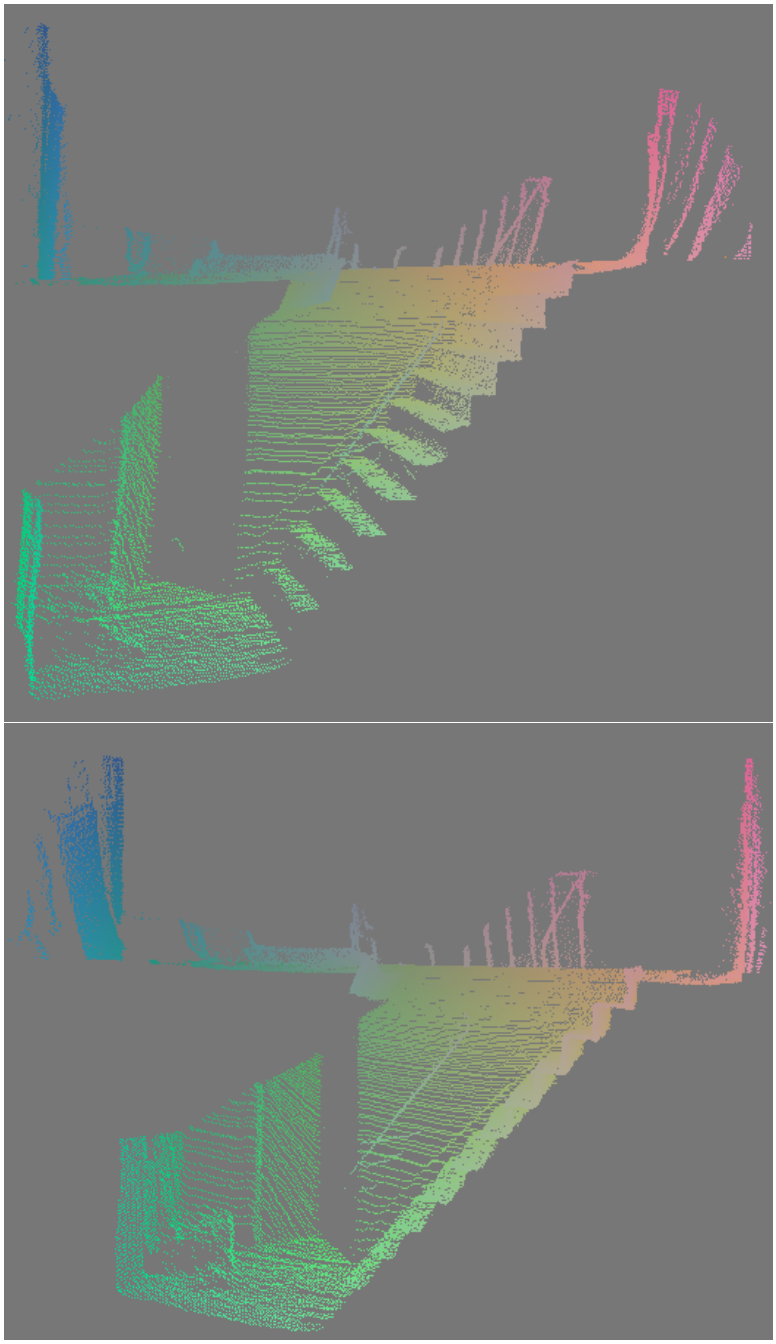

Figure 4.4.16. stairs Speed 1

5
Discussion and future work

5.1 Attachment

The Laser Rig can be attached to a variety of robots, for example an exploration rover. It can move around and scan various environments as it explores. As the rover explores, it will create 3D environments that are much bigger than a single scan. At the end of the exploration, the environment will be visualized to create a complete map. Another application is a flying robot. The laser rig can function in the same way, but return a more complete visualization from different elevations.

5.2 Algorithm Improvement

A further improvement of the algorithm is to create surfaces out of the point clouds. Multiple points can be grouped together to create a flat surface that better represent the environment. Multiple surfaces can be visualized together to better depict an environment.

A further implementation is to sync two captures into a single one via an Iterative Closest Point (ICP) algorithm.This would correct the inaccuracies in the measurements

of the tilt angles. The captures are taken of the same space from two angles. This gives a more detailed environment as it provides depth to objects. The ICP algorithm takes point clouds and compares individual points in relation to one another to be able to sync captures. An example is to scan a box from the front and again from the corner. The ICP algorithm uses the two captures to create a box that represents the front of the box and it's side. Each consecutive capture is added to the final scan to create a complete box. This same idea can be used on more complex objects to better depict an environment.

Bibliography

[1] *Hokuyo-URG Laser Range Sensor in Python*, Access Date January 2011, `https://github.com/nus/pyURG`.

[2] *Bioloid User's Manual: Programming for Bioloid*, Robotis Co., LTD, Bucheon, 2007.

[3] *AX-12 User's Manual*, Access Date January 2011, `http://www.robosavvy.com/RoboSavvyPages/Support/.../AX-12(english).pdf`.

[4] *AX-S1 User's Manual*, Access Date January 2011, `http://www.robosavvy.com/RoboSavvyPages/Support/.../AX-S1(english).pdf`.

[5] *URG-04LX-UG01 Specifications*, Access Date January 2011, `http://www.hokuyo-aut.jp/02sensor/07scanner/download/data/URG-04LX_UG01_spec.pdf`.

[6] *URG Series Communication Protocol Specification*, Access Date January 2011, `http://http://www.hokuyo-aut.jp/02sensor/07scanner/download/data/URG_SCIP11.pdf`.

[7] Francois Blais, *Review of 20 Years of Range Sensor Development*, Journal of Electronic Imaging **January** (2004), 231-240.

[8] Yang Chen and Gerand Medioni, *Object Modeling by Registration of Multiple Range Images*, Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on **Vol. 3** (1991), 2724 - 2729.

[9] Anthony Dearden and Yinnis Demiris, *Learning Forward Models for Robots*, in Proceedings of IJCAI (2005), 1440–1445.

[10] Dirk Haehnel Alensandr V. Segal and Sebastian Thrun, *Generalized-ICP*, In Proceedings of Robotics: Science and Systems **June Issue** (2009).

[11] Pletro Morasso and Vittorio Sanguineti, *Self-Organizing Body Schema for Motor Planning*, Journal of Motor Behavior **Vol. 27** (1995), 52-66.

[12] Jurgen Sturm Christian Plagemann and Wolfram Burgard, *Adaptive Body Scheme Models for Robust Robotic Manipulation*, Robotics: Science and Systems (RSS) **June** (2008).