PoE Lab 2 Report

Isabel Blancett
Sabrina Tamames

December 21, 2017

Abstract

The objective of this lab is to build and program a rudimentary 3D scanner using a pan/tilt mechanism and softwares such as Solid-Works and Arduino. This DIY 3D scanner will be able to scan an object of known, well-defined geometry, and visualize the output of the scanner onto a computer. The purpose of this lab is to introduce sensors and actuators and to become comfortable calibrating them.

1 Materials

- Arduino
- USB cable
- 1X Sharp GP2Y0A02YK0F IR distance sensor
- 1X 3-pin Molex connector with red, black, and white wire tails
- 2X Vigor VS-2MB servo motors
- 8 4-40 screws and nuts
- Software: Solid-Works, Python, Arduino

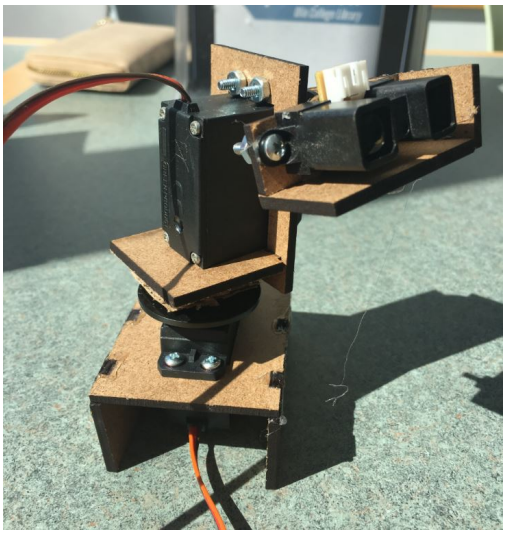
2 Process

2.1 Design and Assembly of Pan/Tilt

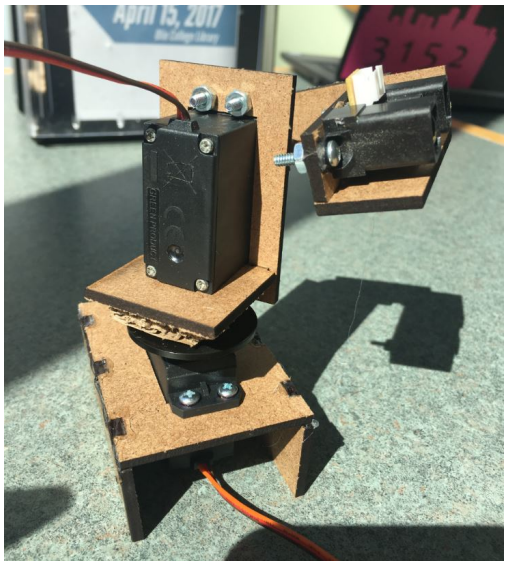
To design and manufacture the servo mounts for the servos and the IR distance sensor, we used Solid-Works and a Helix Laser Cutter. The main servo mount houses the "pan" Servo. It is a "U" Structure made of three separate laser cut parts. The parts are attached and held together using lap joints and hot glue. The servo is attached to the top surface using 1/2 inch 4-40 screws and nuts.

The next structure is the mount for the "tilt" servo. This mount is an "L" structure that is pieced together using mortise and tenon joints and secured with hot glue. This servo was also attached using the same screws and nuts. Using the servo wheel attachment, we hot glued the "tilt" mount to the "pan" mount.

The last structure we designed was the mount for the IR distance sensor. It is an "L" structure with a third wall, all attached using lap joints and hot glue. This structure attaches to the tilt motor using an "arm" and hot glue as well. The arm attachment was added so that the IR sensor



(a) Front View of Scanner



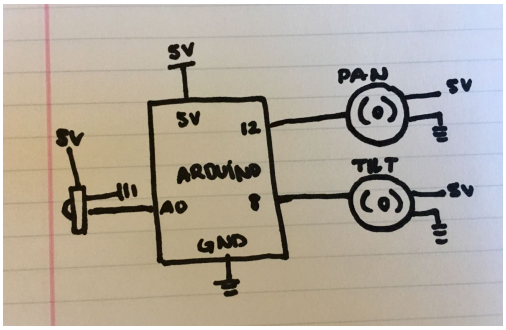
(b) Side View of Scanner

Figure 1: Pan/Tilt Mechanical Structure

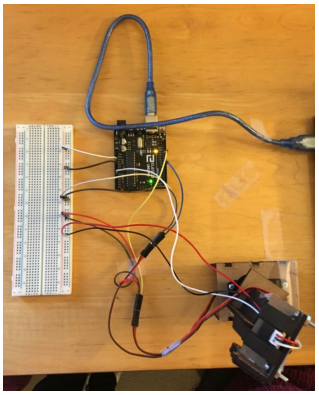
would be perpendicular to the "tilt" motor and the sensor would not move, but rather rotate in place. Overall, the way the mechanism works is by the "pan" motor panning the "tilt" motor. Since the "tilt" motor is directly attached to the IR sensor, the sensor has the ability to both pan and tilt. Images of the completed mechanism is found in Figure 1a and 1b. Lastly, we created was a 8 inch tall cardboard letter "I" along with a base that the IR sensor would analyze.

2.2 Hardware

The electrical hardware for this lab was straight forward, only using direct connection from the Arduino to the motors and sensors. To verify that the pin setup was right and the motors were working, we used the "AnalogInput" example from the Arduino development environment. To make sure the servos were working, we used the "sweep" example under the Arduino IDE. Images of the schematic and physical setup are found in Figures 2a and 2b below.



(a) Full Schematic



(b) Hardware and Breadboard Setup

Figure 2: Schematic and Physical Setup

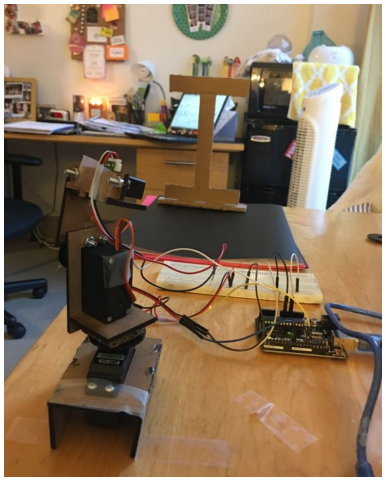


Figure 3: Full Assembly of Pan/Tilt with Letter

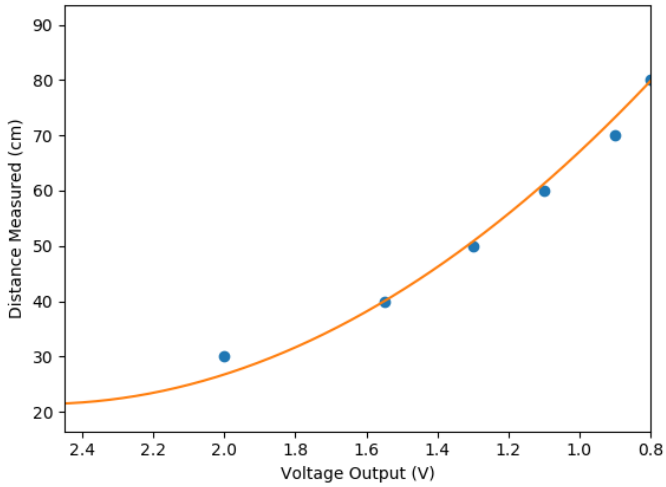


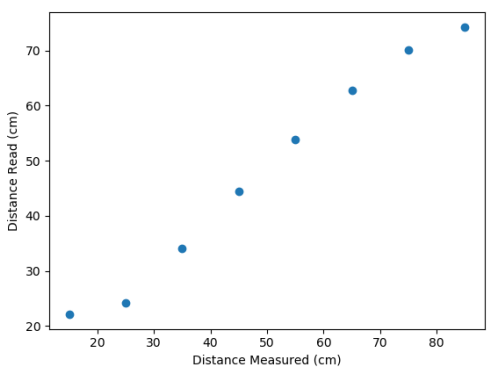
Figure 4: Calibration Graph Created with Python

2.3 Software

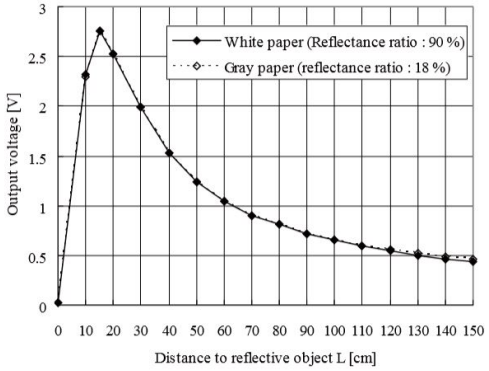
Before beginning assembly and programming of our scanner, we calibrated our distance sensor. To do this, we recorded our sensor's output voltage at hand-measured distances in increments of ten. This set of data then allowed us to configure a calibration curve (shown in figure 4) with numpy's 'polyfit' function which gave us the following relationship between voltage and distance:

$$x = 19.73 * V^2 - 99.46 * V + 146.7$$

We then tested if our calibration was accurate by taking voltages and converting them to distances at hand-measured not used in our original calibration (shown in Figure 5a). We discovered that it was fairly accurate between the 25 and 55 cm, which was deemed logical when compared to the datasheet graph showing a prominent drop off in slope in the range 50-70 cm (figure 5b). This range was kept in mind when testing our 3D scanner with our object.



(a) Calibration Error Plot



(b) Sensor Datasheet Calibration Graph

Figure 5: Calibration Error Analysis

ans

1x20 double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1	147	77	80	151	77	83	93	107	73	72	87	105	84	88	89	108	117	100	105	102	

Figure 6: Sensor Values from a 20 Degree Rotation of the Tilting Servo

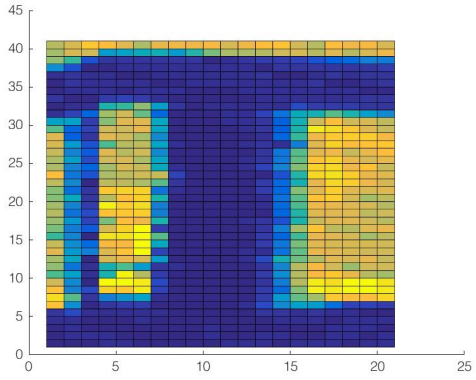
Next, we created a program that tilts the upper sensor top down to make sure we were confident in our ability to send data from Arduino to MATLAB. This code consisted of a for loop that sent our tilt servo 40 degrees counterclockwise and a Serial.println statement that sent out a concatenated string of the servo's position and the sensor's reading. MATLAB recieved this code and sensor's value and sorted it into a matrix using the 'sscanf' function (shown in Figure 6). The panning ability was simply added by nesting another sweeping for loop within the pre-existing one. Following this, the voltage was calculated by multiplying our voltage (5V) over the sensor's range (1023) and the distance was computed by plugging in the voltage values into our calibration curve function. Finally, actual distance was calculated by multiplying the percieved distance by the cosine of each of the measured angles. We then plotted the resulting values on a MATLAB Surface Plot (Figure 7).

2.4 What Were the Problems?

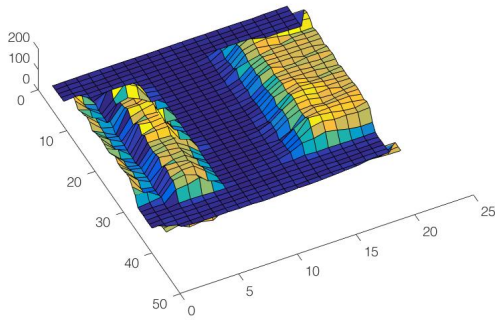
Mechanically, one problem we ran into during assembly was how to set up the sensor so it wouldn't sweep around the panning servo, but rather rotate above it. We fixed this problem by adding an L-arm to extend the sensor outwards. This solution was actually advantageous because it made our scanner more compact.

For software, the first problem we encountered was not being able to send multiple pieces of information, e.g. servo position and sensor value, at the same time. The simple fix was, as mentioned above, to concatenate this information as a string. Another problem that occurred while communicating between Arduino and MATLAB was the fact that MATLAB begins indexing at 1, not 0. Therefore, we had to add and subtract 1 to all our servo positions/indices accordingly.

One problem that was never quite resolved, however, was a mysterious vertical line that occurred to the left side of our letter which can be seen in our final plot. To understand exactly what was happening, we ran several different tests, experimenting with various lighting and objects in the

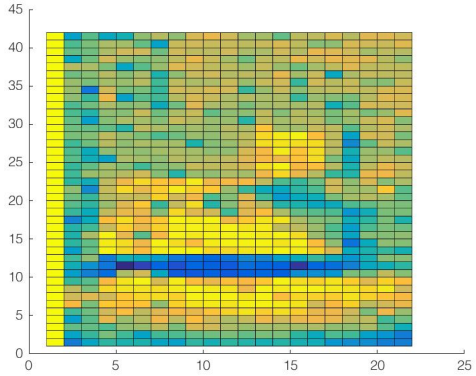


(a) 2D View of Letter Scan

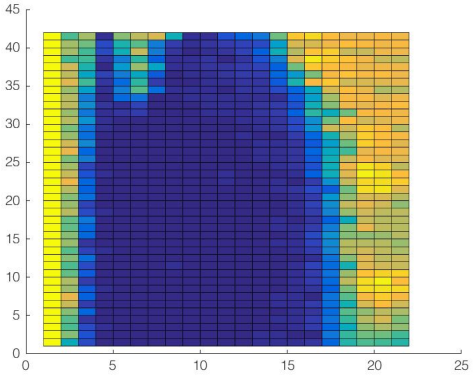


(b) 3D View of Letter Scan

Figure 7: Final Letter Scan



(a) No Object with Intensified Lighting



(b) Water Bottle with Intensified Lighting

Figure 8: Vertical Line Investigation

same environment. The results show that the line is present with different objects, but not when there is no object, indicating to us that the cause is lighting and the resulting shadows. If we had more time for this project, factoring in lighting would be an interesting and challenging next step.

3 Conclusion

In this lab, we were successfully able to create a pan/tilt 3D scanner out of laser-cut hardboard, a sensor, and two servos controlled by an Arduino. We did this by transmitting the servo angle and distance information from the IR sensor to a MATLAB script that filtered this information with calibration data to produce a visualization of our letter, 'I'. Overall, this lab was a great refresher for our Solid-Works skills and challenging introduction to sensors, actuators, and communication between Arduino and other programming languages. This lab forced us to partake in good coding practices, such as commenting and readable variable naming, to make the frustrating debugging process, well, a little less frustrating. Most importantly, it left us with more questions that we're ready to tackle in Lab 3.