



University of Central Florida

Department of Electrical & Computer Engineering

EEL4914 Senior Design I



AUTOMATED
PHOTOGRAMMETRY
STATION

Instructor: Dr. Lei Wei

Group 8

Alhusain Ali Al Badi	—	Electrical Engineering
Ryan Dimmig	—	Electrical Engineering
Isaac Liljeros	—	Computer Engineering
Nelson Vargas	—	Electrical Engineering

Review Committee

Suboh Suboh	—	Computer Engineering
Mike Borowczak	—	Electrical Engineering
John Aedo	—	Computer Science

Table Of Contents

1	Executive Summary	1
2	Project Description	2
2.1	Background & Problem Statement	2
2.2	Project Proposal	2
2.3	Specifications and Requirements	4
2.4	House of Quality	5
2.5	Project's Early-Stage Diagrams	6
2.5.1	Hardware diagram.....	6
2.5.2	Software diagram	7
3	Research and Existing Technologies	8
3.1	Related Technologies and Topics	8
3.1.1	3D Reconstruction Methods	8
3.1.2	SfM Photogrammetry Optimization	10
3.1.3	Linear Rail Actuators	12
3.1.4	Auto-Homing: Resetting Position.....	14
3.2	Existing Projects and Products.....	14
3.2.1	3D Scanner by Tobychui	15
3.2.2	Motorized Horizontal Camera Slider By Superb Tech.....	16
3.2.3	“Medusa” Turntable by Audiomatica	17
4	Components and Parts Selection	18
4.1	Mechanical Structure: 3 Degrees of Freedom	18
4.1.1	Vertical Motion & Tilting Motion	18
4.1.2	Turntable Motion	18
4.2	Camera	19
4.2.1	Phone Camera – iPhone 12 Camera.....	19
4.2.2	Raspberry Pi Camera Module 3.....	19
4.2.3	Raspberry Pi HQ Camera.....	20
4.2.4	Arducam 64MP Autofocus Camera.....	21
4.2.5	DSLR – Canon T7i	21
4.2.6	Comparing the Options	21
4.2.7	Final Decision	22
4.3	Microcontroller Unit (MCU)	22
4.3.1	Power Usage	22
4.3.2	Motor Control	23
4.3.3	Sensor Control	23
4.3.4	Communication.....	23
4.3.5	The Operating System Requirements	24
4.3.6	A Realistic Design: Separation of Concerns.....	24
4.3.7	Choosing the ARM-Based Linux Minicomputer.....	25

4.3.8	Choosing the Accompanying Microcontroller.....	27
4.3.9	The Operating System Revisited	28
4.3.10	The Microcontroller Development Board.....	29
4.3.11	Interfacing the ATMEGA2560 with USB	29
4.4	Motors.....	30
4.4.1	AC Brushless Motors.....	30
4.4.2	Stepper Motors.....	30
4.4.3	Part Selection: Stepper Motor.....	33
4.5	Motor Drivers.....	34
4.6	Auto-Homing Sensor	35
4.6.1	Limit Switch Sensor.....	35
4.6.2	Hall Effect Sensor	36
4.7	Display Module.....	37
4.7.1	LCD Screen with Buttons	37
4.7.2	Touch Screen	37
4.7.3	Computer Interface	38
4.7.4	Phone Application Interface	38
4.7.5	Final Comparison and Decision.....	38
4.8	Power System.....	39
4.8.1	System's Power Analysis.....	39
4.8.2	Linear Regulators.....	41
4.8.3	Switching Regulators.....	42
4.8.4	Regulators Comparison.....	42
4.8.5	Part Selection: 110-AC to 12-DC *More research*	43
4.8.6	Part Selection: 12-DC to 5-DC Regulation.....	44
4.9	3D Reconstruction Software	45
4.9.1	Alicevision Meshroom.....	45
4.9.2	RealityCapture	46
4.9.3	3DF Zephyr.....	46
4.9.4	Software Comparison & Selection.....	47
4.10	Parts Selection Overview	48
5	Related Standards and Realistic Design Constraints	49
5.1	Related Standards.....	49
5.1.1	IEEE 802.11n.....	49
5.1.2	USB 2.0.....	50
5.1.3	ISO/IEC 9899:2018	50
5.1.4	NEMA 5-15-P.....	51
5.1.5	IEC 60130-10.....	51
5.1.6	NEMA-17	52
5.2	Realistic Design Constraints	52
5.2.1	Economic Constraints	52
5.2.2	Time Constraints	53

5.2.3	Environmental Constraints.....	54
5.2.4	Social Constraints	54
5.2.5	Ethical Constraints	54
5.2.6	Political Constraints	55
5.2.7	Legal Constraints	55
5.2.8	Health & Safety Constraints	57
5.2.9	Manufacturability Constraints	58
5.2.10	Sustainability Constraints	58
5.2.11	Constraint Conclusion & Summary	59
6	Overview of Project Design.....	61
6.1	Project Electrical Architecture	61
6.2	Project Software Architecture	62
7	Design Details: Mechanical & Electrical Hardware.....	63
7.1	Mechanical Hardware	63
7.1.1	Vertical Motion Hardware Design	63
7.1.2	Camera Tilt Hardware Design	65
7.1.3	Turntable Hardware Design	66
7.1.4	Mechanical Overview Design: Mechanisms Integration	67
7.2	Electrical Hardware	68
7.2.1	DSLR Camera Setup	68
7.2.2	Stepper Motors Setup	68
7.2.3	Auto-Homing Sensor Setup	69
7.2.4	Touch Screen	70
7.2.5	Power Delivery	71
8	Design Details: Software.....	72
8.1	Raspberry Pi Software Design	72
8.1.1	Requirements	72
8.1.2	Implementation Details	72
8.1.3	Development Environment	74
8.2	ATmega2560 Software Design	75
8.2.1	Requirements	75
8.2.2	User Interface Software	75
8.2.3	Calibration Software	77
8.2.4	Motor Control Software	78
8.3	Linux Workstation Software Design	81
8.4	Inter-Device Communication Protocols	81
8.4.1	Raspberry Pi & Arduino Communication	81
8.4.2	Raspberry Pi and Workstation Connection	82
8.5	Development Environment Introduction	82
8.5.1	ATmega2560 Development Environment	82
8.5.2	Raspberry Pi Development Environment	83

9	Prototype Testing & Integration	84
9.1	Mechanical & Electronics Testing.....	84
9.1.1	Vertical Motion	84
9.1.2	Camera Tilt Motion.....	84
9.1.3	Turntable Motion	84
9.1.4	Auto-Homing Sensor Testing	85
9.1.5	Power Supply	85
9.1.6	Touch Screen Testing	85
9.2	Software Testing	86
9.2.1	Motor Operation Control	87
9.2.2	Touch Screen UI Design and I/O Handling	88
9.2.3	Raspberry Pi 3 & Atmega2560 Communication	89
9.2.4	Camera Remote Capture	89
9.2.5	Meshroom 3D Reconstruction Testing	89
9.3	Project Operation	96
10	Overall Design & Construction	98
10.1	System's Integrated Schematic	98
10.2	PCB Vendor & Assembly	98
10.3	Final Coding Plan	98
11	Administrative Content	99
11.1	Team Overview & Responsibilities	99
11.2	Project Milestones.....	100
11.3	Project Budget & Materials	101
12	Project Summary & Conclusion.....	102
13	Appendices.....	i
13.1	Copyrights.....	i
13.2	References.....	ii

List of Figures

Figure 1: Typical Photogrammetry process	2
Figure 2: ARAGO Automated System Designed by Rigsters	3
Figure 3: House of Quality	5
Figure 4: Early-Stage Hardware Diagram	6
Figure 5: Early-Stage Software Diagram.....	7
Figure 6: Structure from Motion (SfM) Triangulation.	8
Figure 7: Photogrammetry Image Capturing Process	9
Figure 8: Ball-Screw driven Linear Actuator.	13
Figure 9: DIY 3D scanner by Tobychui	16
Figure 10: "Medusa" Turntable Design by Audiomatica.	17
Figure 11: Pi Cam Example by Adafruit Industries under CC 2.0	20
Figure 12: Raspberry Pi 3 Model B by Onepiece84 under CC 4.0.....	26
Figure 13: ATmega2560 Board by Adafruit Industries under CC 2.0	28
Figure 14: Stepper Motor (Left) vs Servo Motor (Right)	31
Figure 15: The Different Coil Setups for Unipolar and Bipolar Stepper Motors	32
Figure 16: Power supply for a Bipolar Motor.....	32
Figure 17: Bipolar Motor Drivers (A4988 vs TB6600).....	34
Figure 18: Limit Switch vs Hall Effect Sensor with LED	36
Figure 19: Project's Power Conversion Diagram	40
Figure 20: A Typical Linear Regulator Circuit.....	41
Figure 21: A Typical Switching Regulator Circuit.....	42
Figure 22: AC to DC Power Supply Choices	44
Figure 23: Buck Converter Module	44
Figure 24: Hardware Diagram	61
Figure 25: Comprehensive Software Flowchart	62
Figure 26: Vertical Motion Design 3D Visualization.....	64
Figure 27: Camera Tilt Mechanism Design 3D Visualization.....	65
Figure 28: Turntable Mechanism 3D Visualization.....	66
Figure 29: Photogrammetry Rig 3D Design Visualization.....	67
Figure 30: User Interface Software Diagram.....	76
Figure 31: UI Option Selection Layout Mockup	77
Figure 32: Calibration Software Diagram.....	78
Figure 33: Styracaceous Model (Testing Dataset).....	90

List of Tables

Table 1: Requirement Specifications	4
Table 2: Photogrammetry vs LiDAR technology comparison	10
Table 3: Image Quality (RAW vs JPG)	11
Table 4: Recommended Camera Parameters for Photogrammetry	12
Table 5: Linear Actuator (Ball-screw vs. Timing belts)	13
Table 6: Vertical Motion Slider Components	18
Table 7: Comparing the Camera Options	22
Table 8: Comparison of Communication Standards	24
Table 9: ARM-Based Minicomputer Comparison.....	25
Table 10: Microcontroller Comparison	27
Table 11: Operating System Comparison	29
Table 12: USB Interface Chips	29
Table 13: Stepper Motor Comparison.....	33
Table 14: Motor Drivers Comparison.....	35
Table 15: Auto-Homing Sensors Options.....	36
Table 16: Display Module Options.....	39
Table 17: Project's Voltage and Current Demands	40
Table 18: Linear vs. Switch Regulators Comparison	43
Table 19: Vertical Motion Slider Components.....	47
Table 20: Part Selection Overview	48

1 Executive Summary

This senior design paper presents the development and implementation of an innovative, cost-effective, and user-friendly automated photogrammetry station, aimed at simplifying the process of capturing high-quality images for photogrammetry applications. The primary objective of this project is to provide a consumer-level product that democratizes access to photogrammetry technology, enabling a broader audience to benefit from its numerous applications, such as 3D modeling, digital archiving, and virtual reality experiences.

Photogrammetry is a technique used to extract geometric and spatial information from two-dimensional photographs to create accurate and detailed three-dimensional models of objects or environments. This technology has numerous applications across various industries, such as archaeology, architecture, geospatial analysis, and entertainment. The intended applications of the proposed automated photogrammetry station include cultural preservation, historical restoration, medical prosthetics fitting, filmmaking, and game development.

The automated photogrammetry station is designed with three degrees of freedom to ensure optimal image acquisition. These degrees of freedom include the ability to move a camera vertically, tilt it at various angles, and rotate the object being captured. This flexibility allows users to easily obtain comprehensive image sets required for accurate and high-resolution 3D reconstructions, without the need for complex manual adjustments or expensive equipment.

A key component of the proposed solution is the integration of Structure-from-Motion (SfM) algorithm-based software, which processes the captured images to generate a detailed 3D mesh. This software not only reconstructs the geometric information of the object, but also extracts the color and texture data from the images, allowing for a seamless and visually appealing final output.

The automated photogrammetry station has been developed with sustainability and scalability in mind. Its modular design ensures easy expandability and adaptability to future enhancements in photogrammetry techniques or hardware requirements. The use of cost-effective and readily available components reduces the overall investment, making the product accessible to a wider range of consumers, from hobbyists to small businesses.

Furthermore, the user-friendly nature of the automated photogrammetry station is a significant advantage in promoting its adoption. The streamlined process of image capture and 3D reconstruction reduces the learning curve for novice users, enabling them to generate professional-quality results with minimal training. This ease of use, combined with the cost-effective nature of the product, positions it as an attractive solution for a diverse range of potential users.

In conclusion, the automated photogrammetry station presented in this senior design paper represents a significant advancement in the accessibility and affordability of photogrammetry technology. By simplifying the image capture process and automating the 3D reconstruction workflow, this product has the potential to revolutionize the way in which various industries and individuals utilize and benefit from photogrammetry. The sustainable and scalable design ensures that the automated photogrammetry station will remain relevant and adaptable to future advancements in the field, positioning it as a long-term solution for a growing market.

2 Project Description

2.1 Background & Problem Statement

For many years, “Photogrammetry” has been an ever-growing part of a wide range of industries and practices. Photogrammetry, a word comprised of “Photo” and “Geometry,” is the process of reconstructing a 3D mesh (model) from a collection of overlapping images via an algorithm that extracts the geometry data from the images. Photogrammetry has been and will continue to be a critical aspect of many industries such as engineering, paleontology, medical prosthetics fitting, and cultural heritage preservation. However, a few of the prominent applications are video game development, film production, and 3D printing. This in turn resulted in photogrammetry making its way into the average consumer’s life. This project aims to tap into that market and create a product which further enhances the relevance of photogrammetry to the general consumer.

The current implementation of the Photogrammetry process involves two steps, and is outlined in Figure 1:

- 1) Images of the object are captured from various angles, and then they are pre-processed to ensure consistency and overlap.
- 2) The Images are fed into a *Structure-from-Motion (SfM)* algorithm. The output of which is a 3D mesh in the form of a *wavefront (.obj)* file and texture maps in the form of a *material library (.mtl)* file.

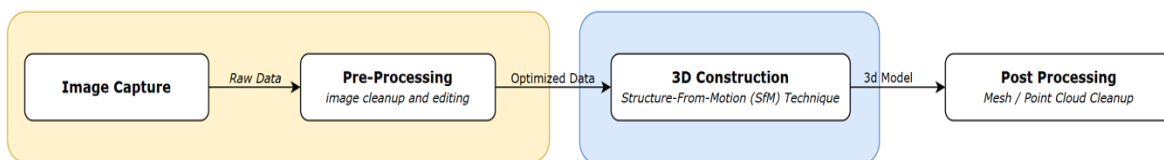


Figure 1: Typical Photogrammetry process

However, the manual process of capturing pictures can be time-consuming as it takes from as little as 10 images to as big as 200+ images depending on the object. Moreover, a key requirement for the images is at least 50% overlap which makes the process prone to errors when a human is behind the gears leading to inaccurate results and reducing the efficiency of the process. Even worse, the lighting conditions for the capture matter. Sufficient lighting conditions are essential to get the most out of the *Structure-from-Motion (SfM)* technique. Keeping all these conditions in mind while capturing pictures generally results in a tedious and ineffective result.

2.2 Project Proposal

Our project aims to minimize the stress and inconsistencies of the manual capturing and processing stages by providing an automated procedure that is specifically designed to deal with manual process issues. To address this challenge, the primary aspect of the project is aimed at designing an automated photogrammetry station that can efficiently capture high-quality images of the object in a controllable and customizable approach as well as ensuring accurate and consistent results that favors the SfM technique. The secondary aspect of the project is aimed at integrating 3D

reconstruction software into the process in case it is desired or required by the end user. By automating the photogrammetry process, this project aims to revolutionize the field and bring significant benefits to professionals and enthusiasts alike.

Ideally, the process would go as follows: First, you would place an object on the marked spot on the rig. Then, with the press of a button, the rig will take a series of pictures from different angles, format them, and send them either to the user, or the converting software outright. Finally, the converting software will take the photos as input and provide the needed 3D modeling file as output. To accomplish this smooth, automated process, there are some specific deliverables that are required. We need to have a way to take pictures from different angles. We address this deliverable by having a rotating platform on which the object sits, and a motor-controlled camera mount which can adjust the camera angle and height position. To account for shadows and other things that could throw off the camera, we have a constant backdrop setup, as well as several lights around the rig. We also are contemplating what our display will look like and what data it will provide to the user (i.e., progress bar) and we are undecided on whether that will be a simple LCD screen on the rig, a companion application, or a mix of both for optimal user experience.

This project is not something that someone would ideally use on-the-go, so we have no reservations about making the rig battery powered or portable. This rig would make its home in a workstation or workshop and would be plugged into a wall outlet. It would utilize stepper motors and a microcontroller, as well as a camera. Our rig would utilize external software to reconstruct the images into the 3D model, and some sort of external user interface to facilitate the photo taking process in case there are any issues.

This is a project that has been done before, we are simply aiming to put our own spin on it. The project was inspired by “Arago”, shown in Figure 2, which is a high-end product for autonomous photogrammetry demonstrated in the figure below. We aim to replicate our own version of this product that is more cost effective and affordable for smaller businesses and average consumers.



Figure 2: ARAGO Automated System Designed by Rigsters

Once the system has been validated, we will demonstrate its potential for use in various industries and applications. One of the applications we will be testing is 3D printing and 3D rendering.

2.3 Specifications and Requirements

To accomplish the general goals of this project, the following requirements were laid down so that there would be a clear objective to achieve. The goal was to make planning an easier and more informed task. As Table 1 makes clear, there are four main aspects of the project that need to be fleshed out.

Table 1: Requirement Specifications			
Specification		Demo/Stretch	Constraints
Object Plate	Object Capacity Limit	<u>Demo</u>	Maximum weight should be at least 5 kg
	Object Dimensions	<u>Demo</u>	Minimum plate radius of 10cm
	Table Rotational Rate	<u>Demo</u>	Minimum rotational speed of 1 rev per minute
	Table Stability	-	Table should tilt less than 1° with maximum weight
Cam.	Camera Stand Size	<u>Demo</u>	Should be at least 70cm.
	Camera Adjustability	-	Should point 45° up and down.
	Camera Quality	-	Should at least be 12 Megapixels
	Capture Speed	-	Should have fast shutter speed of at least 1/60 sec.
MCU	MCU Feature Set	<u>Demo</u>	Should coordinate 3 motors, 1 Camera, 1 LCD touch screen and 1 sensor
	Transmission Rate	-	Should Transmit at a rate of 100 MB/s
UI	Desktop App Scope	Stretch	Initiate Photogrammetry & Transfer to Algorithm
	UI Polish	Stretch	-
	LCD Size	-	Display should be 9cm across
Power	Power Output	-	Should output at least 12V for motors.

As the above table demonstrates, there are four main categories: Object Plate, Camera, MCU, and User Interface. Each of these areas represents a measurable yet achievable amount of work that can be segmented off to one or more members of the group. This will help to increase productivity by leveraging parallel work.

For example, the object plate (and its related components such as the motors and rails) will be controlled by the microcontroller unit. But one does not require the other to be in a fully operational state to work on the other. There will need to be some form of communication between the various aspects of the project, which are not represented in this table, but they are not design goals but rather part of the initial design.

Importantly, all the specifications have been marked as demo-able, as stretch goals, or as neither. The demo-able ones are intended to be demonstrated at the end of the project to show that it has been complete as originally planned. The stretch goals are goals that would be excellent to include in the result, but if that is not possible then it is not a huge issue. Finally, there are those which are not marked. These goals are either trivial, essential, or unimportant and consequently the team is not specifically chasing them.

2.4 House of Quality

Figure 3 is the house of quality. It summarizes many of the initial constraints of the project and how they relate to one another.

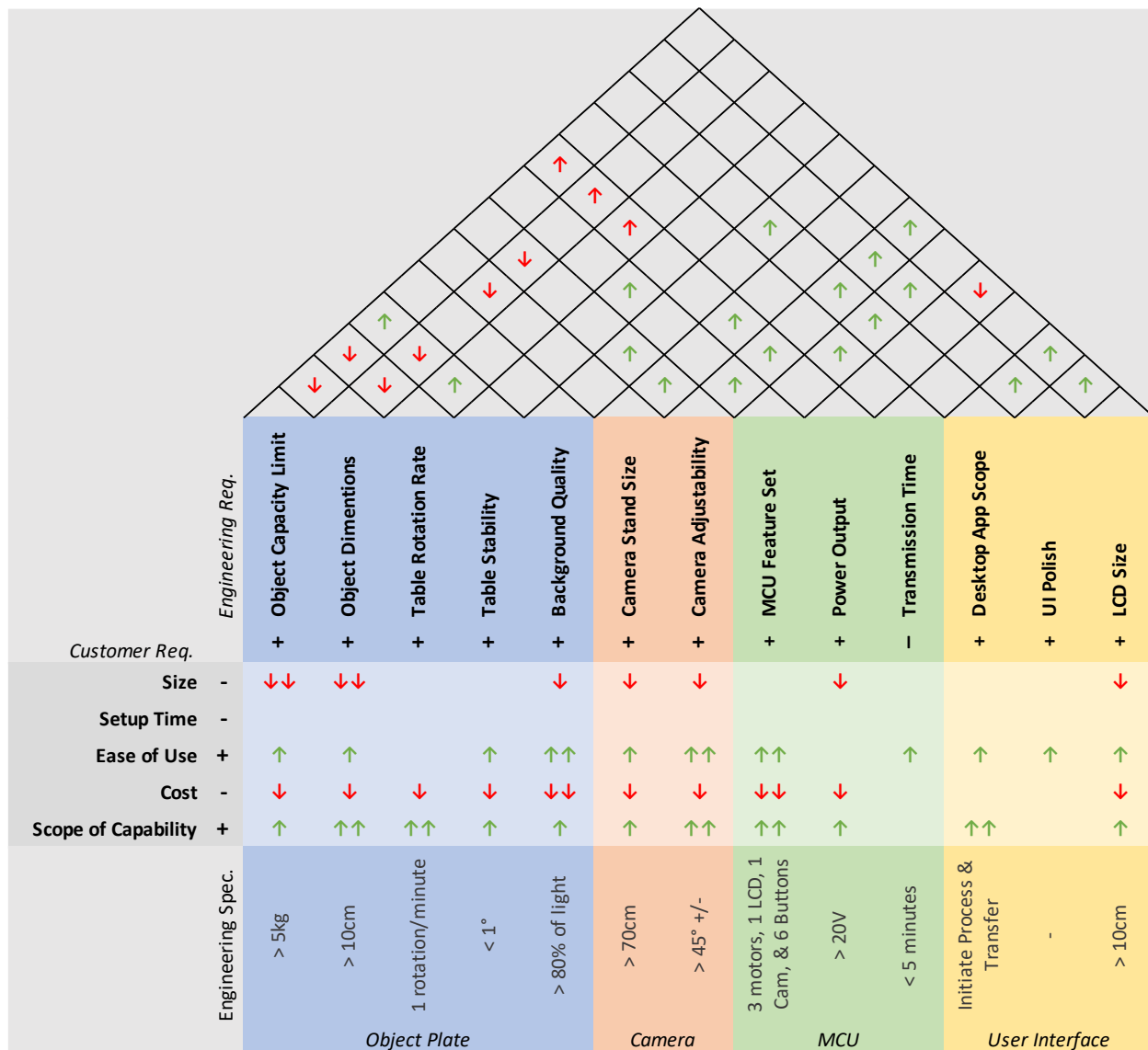


Figure 3: House of Quality

In the house of quality, there are a multitude of relationships and specifications being represented at the same time:

1. The relationship between engineering requirements and the consideration of how they compete with one another.
2. The relationship between engineering requirements and customer requirements
3. The minimum specs that constrain the engineering requirements

It should be noted that some of these requirements do not have a clear correlation. However, the ones that do have been colored to represent whether this correlation is **beneficial** or **detrimental**. Notice that power output is positively correlated with rotation rate but marked as detrimental; this is intentional because it shows that even though increasing one increases the other, we do not want to have to increase the power output.

2.5 Project's Early-Stage Diagrams

With all the requirements set, the next major discussion is the operational and constructional nature of the project. The purpose of this section is to establish an early-stage development diagrams that will help guide the research and construction of the project. These diagrams will be revisited in later section once additional research, investigation and part selection has been established. The section is divided into two parts: Hardware and Software diagrams.

2.5.1 Hardware diagram

The purpose of hardware diagram, showcased in Figure 4, is to establish a general overview of the structure of the project. While it lacks the details and specifics of the project, it serves as a good starting point for development.

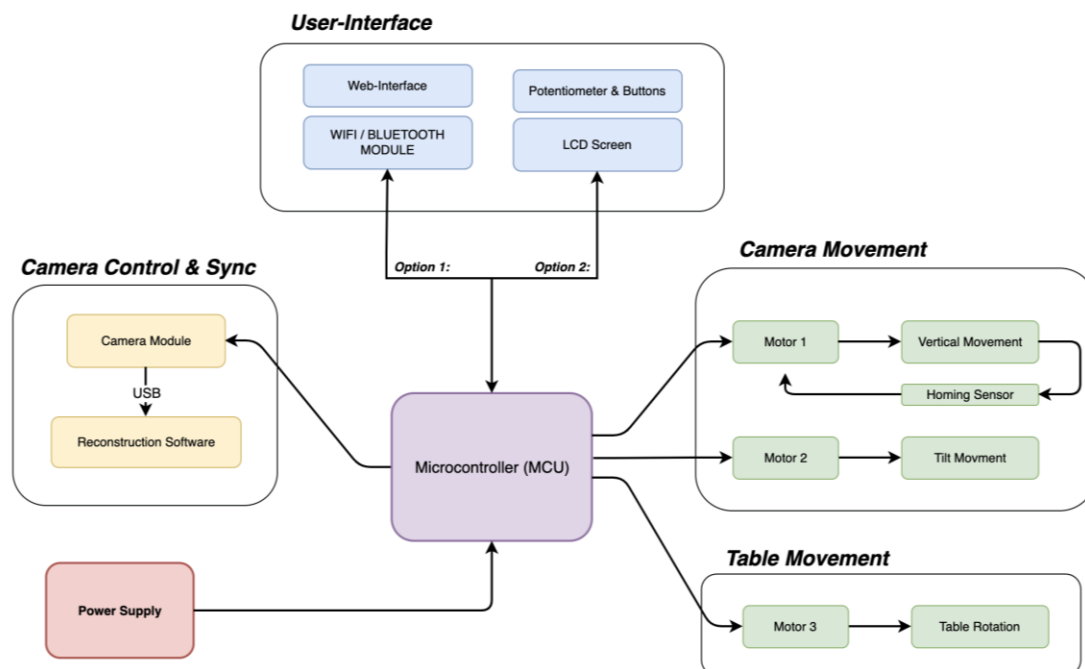


Figure 4: Early-Stage Hardware Diagram

2.5.2 Software diagram

The purpose of hardware diagram, showcased in Figure 5, is to set a general working pipeline for software side of the project. It consists of general, but very important, algorithms that need to be researched and developed to ensure correct operation of the project.

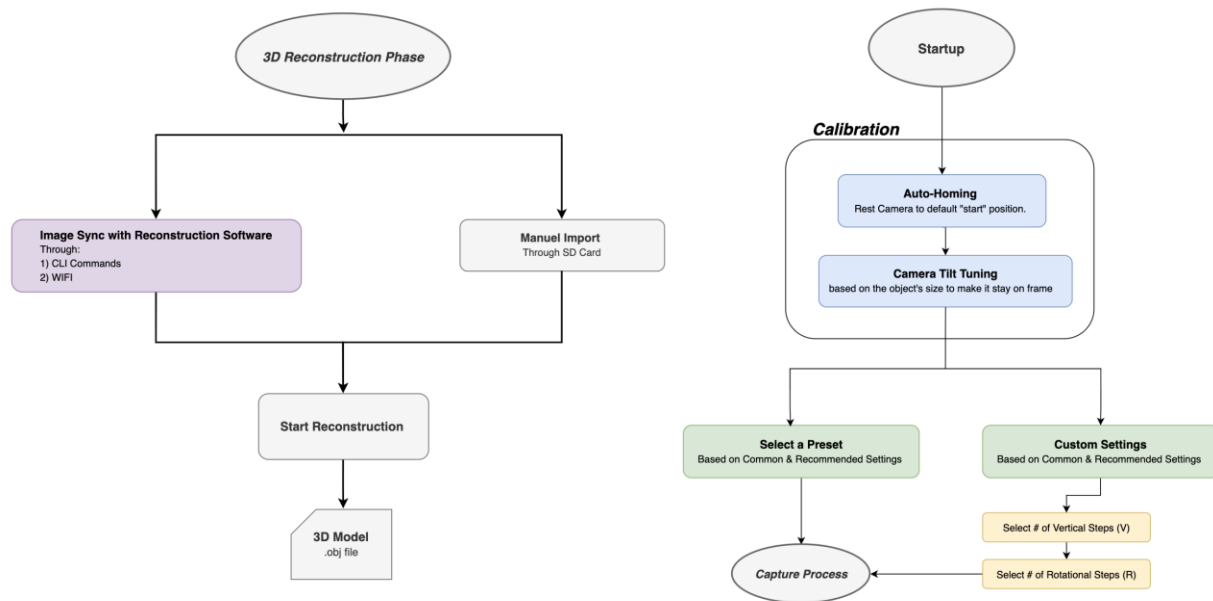


Figure 5: Early-Stage Software Diagram

3 Research and Existing Technologies

This chapter covers the project's structure by exploring relevant topics, technologies, and related projects. As the last step, the appropriate parts and components are selected to build the system. These parts are selected from extensive research into relevant alternatives and through detailed comparison charts.

3.1 Related Technologies and Topics

Following is a brief discussion of some of the technologies that will be used throughout the project and how they work.

3.1.1 3D Reconstruction Methods

In the field of digital reconstruction, there are two major technologies that enable the process of extracting physical features of an object, converting them to digital data in the form of a 3D Mesh, and can be rendered in an appropriate software. These two technologies are: Photogrammetry and LiDAR. Note that Photogrammetry is associated with the algorithm Structure-from-Motion (SfM) and hence the terminology between the two is interchanged, at least this is the case for this paper. Additionally, it should be noted that while the title of the project indicates that Photogrammetry was the technology the team has chosen to implement, it is still important to mention LiDAR as a comparable technology since the team have considered it at the initial stages of the project.

3.1.1.1 Structure from Motion (SfM)

As the name implies, Structure from Motion is an algorithm that is capable of generating a 3D structure (i.e., 3D Mesh) from a series of overlapping, offset images. SfM relies on tracking matching points and/or features between two overlapping images. In other words, SfM requires no physical contact with the object or environment since the data are simply 2D images. This results in a 3D dimensional space relationship between the two images, where the cameras' relative position is mapped [1]. This method is called “triangulation” and is illustrated in Figure 6: Structure from Motion (SfM) Triangulation. Expanding this method to many images, the algorithm is capable of reconstructing the scene in 3D space with all the relative cameras positions.

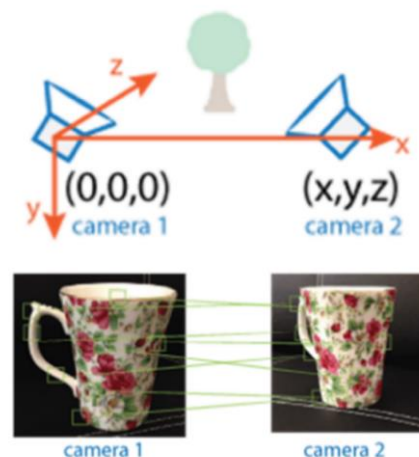


Figure 6: Structure from Motion (SfM) Triangulation.

To ensure detailed and accurate results, the image capturing process involves rotating 360° around the object, both vertically and horizontally. Generally, the higher number of images the better the results, however, the minimum number of images is dependent on the complexity of the object. This process is outlined in Figure 7: Photogrammetry Image Capturing Process. The next step of photogrammetry is then to convert the relative cameras and images position into a point cloud in space. A point cloud is a discrete set of points representing data in 3D space. They represent the surface of the object or environment. The density of the point cloud (i.e., the number of points in a given area) determines the level of details of the output mesh. The point cloud can be processed and converted into a 3D Mesh. A disadvantage of Photogrammetry is that the output 3d mesh can be unnecessarily dense requiring either some additional cleanup or filtering. The filtering process is sometimes called “decimation.” Lastly, an optional step of Photogrammetry that is not offered with LiDAR is the ability to “colorize” the mesh. Because Photogrammetry deals with 2D images, it is possible to also extract the color information of the object and generate what is called “UV Texture Map”. [2]

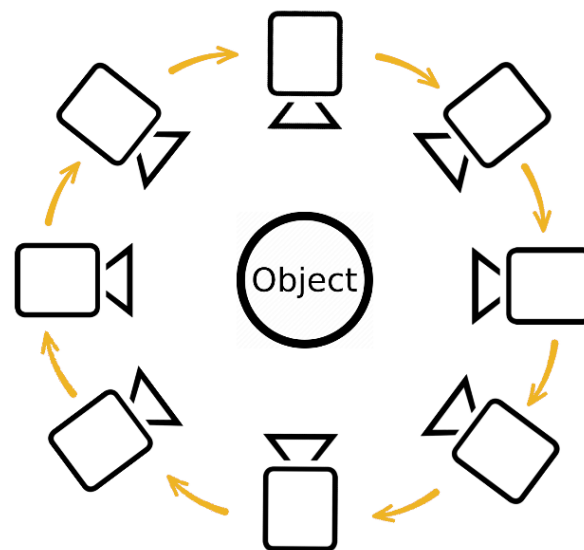


Figure 7: Photogrammetry Image Capturing Process

In summary, the outputs of this method are an object file (.obj) that holds the 3D mesh data, and a material library file (.mtl) that captures the unwrapped textures of the object.

3.1.1.2 Light Detection and Ranging (LiDAR)

Light Detection and Ranging (LiDAR) is a remote sensing method technology that utilizes laser pulses to measure distances and create a high-resolution 3D meshes by generation a point cloud. It functions by emitting laser pulses at a target and measures the travel time for the pulse to hit the target and return to the sensor. In contrast to SfM, this method therefore requires direct physical contact with the environment. This process of sending a pulse and receiving it is repeated many times per second which enables this method to generate point clouds that can be converted to 3D meshes. [3]

LiDAR is very powerful and widely pertinent to variety of applications, but the most important application in the scope of this project is 3D reconstruction. LiDAR offers very accurate results compared to SfM and is often used for high-precision mapping, as it can capture details as small as a few millimeters because it directly deals with physical quantities rather than data extracted from 2D images. However, compared to SfM it does not offer the option to capture the textures and materials of the environment. [3]

3.1.1.3 Technology Comparison

There are three categories of interest when comparing the two technologies: price, accuracy, and working conditions. Regarding price, photogrammetry is significantly cheaper with equipment requirements being very minimal. LiDAR on the other hand is expensive, requiring high-accuracy laser and sensor electronics. In terms of accuracy, LiDAR is more accurate and offers precise mapping of the environment. However, recent comparisons of both methods suggest that SfM Photogrammetry can be just as accurate as LiDAR with the right equipment and calibration. Lastly, working conditions refer to the environmental conditions required for the method to work as expected. It is evident that since photogrammetry works images, the environmental conditions such as lighting are an important part of the process. In contrast, LiDAR requires less environmental monitoring because it generates laser pulses that can travel through most environmental conditions [3]. Table 2: Photogrammetry vs LiDAR technology comparison summarizes these comparisons. For the project, the team has decided to go with SfM as the technology of choice given that it offers comparable accuracy to LiDAR and generates a colorized output that can be 3d rendered and demonstrated as an application. More importantly, it is affordable and fits the budget of the project.

Table 2: Photogrammetry vs LiDAR technology comparison		
	SfM Photogrammetry	LiDAR
Price	Substantially Cheaper	Very Expensive
Accuracy	1-2 Centimeters	Millimeters
Work Conditions	Restrictive	Simple
Outputs	3D Mesh & Texture Map	3D Mesh only

3.1.2 SfM Photogrammetry Optimization

Now that SfM is our selected technology of choice, it is important to look more at details and specifics of SfM. The goal of this section is to investigate the parameters of the algorithm and their influence on the results in hopes of finding the optimal operational conditions of the process, as well as setting up a guide for selection the appropriate parts and component for the project. [4]

3.1.2.1 Image Acquisition

One of the first key steps of photogrammetry is image alignment. As explained in the previous section, the algorithm needs to find the relative 3d dimensional position between the cameras (i.e.,

the images). The result of this requirement is the need for overlap between images. It is recommended that a portion of the previous image be visible in the following image. Typically, the overlap between the images should be between 50% and 80% depending on the specific context and requirements. A trade-off of a higher overlap is an increase in the number of images that must be taken. Therefore, if the process of image capturing is to be automated, the supporting software should be able to smartly adjust the overlapping percentage based on the desired quality specified by the user. [4]

3.1.2.2 Image Quality

Image quality directly affects the accuracy and precision of the resulting 3D model. The quality of the images affects the ability of the SfM algorithm to find common points and features between the images, which in return affect the final mesh. Therefore, the images must be of high quality. Most software recommends cameras that are capable of shooting at least 12 Megapixels images. However, megapixels are not the only factor describing the quality of the image. To elaborate, images are recommended to have sharp focus to distinguish the object from the background, accurate colors, and minimal distortion. Any blurring, noise, or other artifacts in the images can result in errors in the point cloud, which can propagate and affect the accuracy of the final output including both the mesh and texture map. [5]

Additionally, a very important decision to make is the choice between shooting RAW or JPG. RAW is a lossless, uncompressed file format that offers a higher dynamic range allowing post processing capability. JPG is compressed file format where it trades dynamic range with file size. Table 3 offers the advantages and disadvantages of both file formats. For this project, the team decided to go with JPG as it offers file sizes and better texture results. The limitation of JPG will be compensated for with careful selection of camera parameters discussed shortly. [5]

Table 3: Image Quality (RAW vs JPG)		
Feature	RAW	JPG
File Size	Large File (6x JPG Size)	Small File Size
Post-Processing & Color Correction	Allows Color and Light Adjustments	Limited adjustments
SfM Processing Speed	(10% - 20%) Faster than JPG	Slower than RAW
SfM Texture Quality	Good	Better

3.1.2.3 Camera Parameters

There are few important camera parameters that needs to be addressed. Let us begin the discussion with the camera's focal length: an appropriate camera focal length is between 50mm to 70mm for medium sized objects. The large the object, the larger the focal length should be. This is because smaller focal lengths introduce lens distortion that can trouble the reconstruction algorithm. Another parameter is ISO (sensor's light sensitivity) which is associate with exposure. The recommend ISO is variable depending on environment. However, as a rule of thumb, the ISO should be as low as possible as not to have the subject underexposed. Moreover, shutter speed is also important, and it depends on several factors, including the lighting conditions and the speed

of the moving objects or camera. In general, it is recommended to use a fast shutter speed to minimize motion blur to ensure sharp images. However, using too fast of a shutter speed can result in underexposed images, hence it is important to find a balance between speed and exposure. In this project, since we are dealing with mounted cameras, we have the ability to make the shutter speed fast, but we will opt to comprise some shutter speed for a better exposure on the object. Another very important factor is f/stop, which is associated with depth of field (i.e., background clarity). It is desired to set the f/stop so that the entire frame is in focus which enables the algorithm to find common points in the 3d scene. Summary of all these factors is outlined in Table 4: Recommended Camera Parameters for Photogrammetry, and illustrates the decision the team have made to build the project around. [4] [6]

Table 4: Recommended Camera Parameters for Photogrammetry	
	Recommendation
Resolution	≥ 12 Megapixel [7]
Focal Length	50mm – 70mm
ISO	100 – 800
Shutter Speed	$\geq 1/200$
F/Stop	F/6-13

3.1.2.4 Object and Environment Setup

Not all objects can be converted into 3D models, at least not by default. Shiny or transparent objects pose serious problems during reconstruction that cannot be resolved. Such objects reflect and refract the environment around them, and this is dependent on the viewing angle. As a result, the algorithm cannot find common points and features between images. Fortunately, there are simple solutions to overcome this problem, such as spray painting the object with a matte finish. This process, called object setup, is the responsibility of the end user and not a problem for our project. Environmental setup largely involves lighting conditions, which are a key factor in the photogrammetry process. The lighting should eliminate any hard shadows or dark areas to prevent them from appearing in the texture map. A 2-point lighting setup placed $\pm 45^\circ$ from the camera should be sufficient to meet this requirement. [4]

3.1.3 Linear Rail Actuators

Linear rail actuators, such as in Figure 8: Ball-Screw driven Linear Actuator., are a type of motion sliders commonly used as motion components to move loads from one end point to another.

3.1.3.1 Functionality

The system consists of a platform or carriage that moves along a rail known as the track. The track system is normally made from aluminum or steel and is designed to provide smooth and precise linear motion. The system functions by transforming circular motion generated by a motor, ranging from 12V to 48V DC, into linear motion along a desired axis. This system is the perfect method of motion translation for our project as it can move the image capturing device to adjust height and angle for image acquisition. Linear rail actuators come in a variety of designs and configurations, such as ball screw and belt-driven systems. Ball screw systems use a threaded rod and ball bearings

to provide linear motion, while belt-driven systems use a belt and pulley system to drive the load. Both types of systems can provide high precision, however ball screw systems, like in Figure 8, are generally considered to be more precise. Additionally, linear actuators differ in the way they are built, some use single rails while others use double parallel rails. Double rails offer additional support for the load while also allowing for another axis of motion for appropriate applications. [8]



Figure 8: Ball-Screw driven Linear Actuator.

3.1.3.2 Technology Comparison

Ball-screw linear actuators use a threaded shaft (screw) and a ball bearing nut to generate linear motion. They are characterized by accuracy and efficiency compared to timing-belt linear actuators. They are ideal for applications requiring precise positioning and repeatability. Ball-screws are also durable and have a longer lifespan than timing-belts. However, they tend to be more expensive and require more maintenance. Timing-belt linear actuators use a toothed belt to provide linear motion. They are generally less expensive than ball-screw actuators and can achieve high speeds and provide longer strokes (5-4 meters). They are also easier to maintain and can handle greater loads than ball-screw actuators of similar size. However, timing-belt linear actuators have lower accuracy than ball-screw actuators. Table 5 summarizes these comparisons. [9]

Table 5: Linear Actuator (Ball-screw vs. Timing belts)		
Feature	Ball-Screw	Timing Belts
Cost	Expensive (More Components)	Less Expensive
Speed	≤ 1 m/s	up to 4 m/s
Stroke Length	-	Longer Stroke
Precision	More Accurate	Less Accurate
Ease of Design & Installation	-	Simpler

Regarding our project, timing belts are more appealing form of linear actuators as they offer an inexpensive design choice and provide higher speed. In addition, photogrammetry does not require very precise camera placement, and therefore, ball-screw linear actuators would be an overkill approach to the design. As a result, the team has decided to go with timing-belt linear actuators as the technology of choice.

3.1.4 Auto-Homing: Resetting Position

Auto-homing is a feature commonly found in 3D printing and CNC machines, which automatically positions the machine's toolhead or print bed to a known starting position at the beginning of the 3D printer or CNC machine operation. This starting position is typically referred to as the "home position" or "origin" and is used as a reference point for all subsequent movements and processes. It is an essential feature because it allows the machine to automatically position itself for a common reference point that all software code and processes are based on. This is typically done with some sort of sensor that detects the position of the toolhead. Common sensor types for this application are limit switches, optical sensors, and hall-effect sensors. Limit switches are mechanical switches that activate upon pressure. Optical sensors are sensors that detect a change in light to calculate the position. Hall effect sensors detect changes in magnetic field to indicate a reach of position. Regarding our project, this feature may prove to be important since the project requires a common starting position for the capture process. It is a feature that simplifies the setup process and allows for consistent accuracy and makes software implementation easier.

3.2 Existing Projects and Products

The purpose of this section is to examine the current market and open-source implementations by companies or enthusiasts of any major component of the project. This will provide insightful information and a deeper understanding of the current state of research and development in the field. The projects and products discussed below have been selected based on their relevance and potential to influence design and part selection, as well as encouraging further research.

Our project by nature contains several moving parts. The requirements of a rotating table, camera angle adjustment, and capture height variance all require different mechanisms by which to utilize the motors. There are a variety of ways that one can utilize a motor to move another object. Some common mechanisms used to achieve movement are pulley systems, mechanical arms, and couplers. We could utilize any or all of these in our design.

Couplers will be used to make connections to the shaft of the motor. We could connect a plate directly to the coupler of one motor to rotate the spinning table, or alternatively we could use gears. The use of gears could reduce the stress on the motor, which would be good for our power consumption. The use of gears also has the potential to improve the aesthetics of the overall design by reducing the vertical form factor, opting for a more petite setup. A downside to gears is the additional cost of materials, and the risk that one of those components could fail as well.

Another possibility is the use of pulleys to facilitate the movement of our design. Pulleys also are useful for reducing torque requirements and will help reduce power consumption. They could be implemented on the turntable, but they are more practical for the vertical movement of the camera.

While we could use a lead screw setup like some 3D printers use, pulleys seem to be a cheaper and easier implementation, also providing a way to get by without an encoder on that z-axis motor. Additionally, a mechanical arm setup would be a huge deviation from similar previous projects and would not provide any advantage. In fact, it would limit the design and performance of the rig, because it would set the size range of scannable objects in stone. While the pulley system does not necessarily provide a direct solution, it does enable expansion of the size range, and would not require a drastic change to the software implemented as the mechanical arm setup would.

As for the camera angle variance motor, we would like to explore if a pulley system where the motor can remain stationary (as opposed to moving with the DSLR camera). This would also reduce power consumption, as the vertical motor would no longer be mobilizing the angular motor with the camera. This setup would require a creative setup for the pulley connecting to the angular motor, though, and we will have to consider those mechanics before assembly.

Ultimately, we will use a combination of these mechanisms to achieve the movement requirements of this project. Our primary goal will be power efficiency, followed closely by economic and supply chain constraints.

3.2.1 3D Scanner by [Tobychui](#)

In this project (shown in Figure 9), Tobychui implements the same structural idea that we are planning to implement in our photogrammetry rig. The main difference between this project and ours is that Tobychui's project is utilizing LiDAR technology rather than the photogrammetry aspects of our project. It consists of a vertical axis movement that exploits a motor as a generator of action, and a turntable that rotates 360 degrees. The biggest take away from this project is the proof of concept of the mechanical structure design and integrity: it demonstrates that the general design which we also are looking into will work and is relatively easily achievable. However, there are some design requirements that this project does not implement, one of them being the tilt rotation of the sensor which is a crucial feature to our project. Moreover, reading community feedback on the system suggests that the maximum weight the turntable can handle is not ideal. Our project design will implement a similar but improved and superior structural design to Tobychui design, with the biggest difference in camera and tilting motion integration, as well as increased weight support.



Figure 9: DIY 3D scanner by Tobychui

To achieve these improvements, we will carefully consider aluminum rails as the material of construction with minimal 3D printed parts for housing electronics and components we use in our design. We plan to incorporate a tilting mechanism for the sensor that will allow for greater flexibility in capturing images from different angles. In terms of weight support, we will ensure that the turntable can handle the weight of our camera setup, as well as any additional equipment we may need to add in the future. This will involve selecting a motor with enough torque to handle the weight and designing the turntable to distribute the weight evenly. The following two projects address the issues of this design including the limited carrying capacity of the turntable and the complexity of the camera slider.

3.2.2 Motorized Horizontal Camera Slider By [Superb Tech](#)

The objective of this project is to function as a camera slider specifically designed for timelapse photography. This slider uses a 20x40mm linear rail actuator, timing belts, and a bipolar NEMA17 motor to enable horizontal linear motion, capable of supporting significant weight. One of the most significant advantages of this design is its modularity, allowing for easy expansion, which aligns with our project goals. Although the initial design is 500mm in length, it can be easily extended to any required length by the user. In addition, this design incorporates another NEMA17 motor and a ball-head camera mount to generate the tilting motion, providing the capability to adjust the camera angle. The system is controlled by an Arduino Nano mounted at the base of the system, alongside an OLED display and potentiometer to launch the program and adjust primary controls.

For our project's purposes, we plan to repurpose this design for vertical motion. This requires a redesign of the base holding the aluminum rails, and extending the design to at least 750mm length track, as well as selecting a higher torque motor capable of lifting and holding the load at a specified vertical location. A drawback of this design is that both motors are mounted as a load adding non ideal weight to the system. In our repurposed design, the vertical driving motor will be mounted at the base.

3.2.3 “Medusa” Turntable by [Audiomatica](#)

As stated previously, one of the biggest concerns of our photogrammetry project is the maximum weight support by the turntable. This concern has been validated in community feedback in Tobychui’s scanner design in section 2.2.1. The obvious solution is to select a higher torque motor, but this is costly and goes against one of our project’s goals, that is cost effectiveness. This is where this “Medusa” turntable project comes in play, shown in Figure 10. Its underlying functionality is to increase the torque of the motor by using gears with a large gear’s ratio m_G . As a result, the turntable weight capacity can be increased without the need to select a motor with higher torque. The tradeoff of this design is a lower angular speed because it is directly proportional to the inverse gear ratio $1/m_G$. To enable rotation, the design incorporates a “Lazy Susan” bearing, a type of hardware that allows smooth rotation of a turntable. It consists of two circular plates with ball bearings layered between them. The larger gear is mounted on this bearing. A timing-belt attached to both gears where the motor is then capable of generating the required rotational motion.

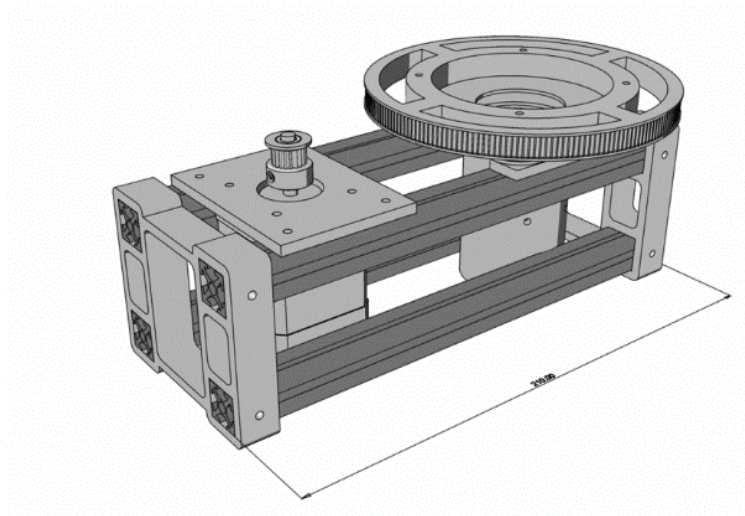


Figure 10: "Medusa" Turntable Design by Audiomatica.

The reported number of teeth for the driven gear is 120, making the gears ratio 9 (for 20 teeth pulley). In other words, the output torque of the design allows for 9x the input torque. Fortunately for us, this design is under creative commons 4.0 commercial license, granting us the ability to remix, transform, and use the design in any way we deem fit. Like the camera slider in the previous section, this design is also modular and allows for future expansion. It is built with 3d printed material and some aluminum rails, and hence, the design can be modified specifically to fit the end user’s requirements. With slight modifications in component selection, we will integrate this design into our project.

4 Components and Parts Selection

Based on the research from the previous sections, parts must now be selected. The following are a multitude of comparisons between existing products as well as decisions on which one will be used for the project.

4.1 Mechanical Structure: 3 Degrees of Freedom

The mechanical structure, in terms of design, has already been addressed in section 3.2. In summary, the design of the mechanical structure allows for 3 degrees of freedom: vertical motion, tilting motion, and turntable motion.

4.1.1 Vertical Motion & Tilting Motion

As stated previously, the team has decided to go for a timing-belt driven linear actuator for the vertical motion, for its simplicity, cost, and ease of expansion (refer to section 3.1.3 for more). The mechanism consists of a 20x40 aluminum rail profile, a load carriage platform transporting another motor and camera mount to generate the tilt mechanism, and a base holding the structure. Table 6 summarizes all the components selected. Other components, such as the base holding the structure as well as the base holding the motor and the camera mount on the carriage platform will be 3D printed using an available Prusa i3 MK3S 3D printer to one of the team members.

4.1.2 Turntable Motion

The turntable mechanical structure is based on the “Medusa” turntable design discussed in section 2.2.3. The mechanism consists of two 20x20 aluminum rail profiles, and lazy Susan bearing, and 3D printed base. In contrast to the original “Medusa” design, our design reduces the number of aluminum rails to two to minimize the cost of unnecessary components.

Table 6: Vertical Motion Slider Components		
	Supplier	Cost
Vertical & Tilt Mechanism		
2040 Aluminum Profile (100cm)	Open Builds	\$15.29
V-Gantry (Carriage Platform)	Walfront (Amazon)	\$14.14
G2 Timing-Belts & Pulleys	Zeelo on (Amazon)	\$15.99
Ball-Head Camera Mount	CAVIX (Amazon)	\$10.18
(2x) Mount Base	3D Printed	
Turntable Mechanism		
(2x) 2020 Aluminum Profile (20cm)	Open Builds	\$6.58

Lazy Susan Bearing	DMSTECH (Amazon)	\$4.35
Mount Base	3D Printed	-

All 3D printed parts are printed using PLA (Polylactic Acid) with a 0.4mm nozzle. To ensure rigidity and structural reliability, 1.5mm is set for the layer height and 50-70% is set for the infill with a gyroid pattern as it offers the best strength/weight ratio.

4.2 Camera

The next component needed for this project is the Camera. Given that the project is based on the process of photogrammetry, it makes sense that one of the most important components is the camera. In the [Image Quality](#) and [Camera Parameters](#) sections of the document, the requirements for the camera have been clearly laid out, so those sections will not be repeated here. But keeping those requirements in mind, there are a few viable options that could be adopted for this project: a phone camera (the following discussion will consider the iPhone 12 camera; however, as it turns out, most modern smartphone cameras are capable of meeting the minimum requirements for this project), a [Raspberry Pi Cam 3](#), the [Raspberry Pi HQ Camera](#), a [64MP Autofocus Camera](#), and a DSLR (the [Canon T7i](#) was investigated for this project). The following sections break down the pros and cons of the different viable options, and then the final decisions are made.

4.2.1 Phone Camera – iPhone 12 Camera

One of the most accessible cameras available to the average user is a phone camera. And many phone cameras reach the requirements set forth in the previous sections. Though some are better than others, a decent sample of a phone camera (and one that was available during the research of this project) is the camera on the iPhone 12. The main shooter on this phone is a 12MP sensor with a $f/1.6$ aperture. Additionally, the iPhone includes a powerful image processor which turns the otherwise ok raw image into a clean final image [10].

All this sounds great. However, there is one major drawback to using the iPhone's camera which will keep this project from using it: there are generally no stock (built-in) ways to interact directly with the phone's camera using another device. Instead, an entire iOS specific app would likely need to be developed to enable this operation. Though this will become more evident in a few sections, there is already a great deal of software development and orchestration which will need to take place in this project, and adding a custom phone application to that list of tasks would be unwieldy.

4.2.2 Raspberry Pi Camera Module 3

Another fantastic option considered for use in this project was the Pi Camera Module 3, known briefly as the Pi Cam 3. This product is a considerable improvement over the previous iteration of the Pi Camera Module: the Pi Cam 2. The Pi Cam 2 was equipped with an 8MP camera, whereas the Pi Cam 3 has a 12MP Camera. The Pi Cam 3 also upgrades the manual focus of the previous camera to an all new autofocus system [11].

One of the main benefits of using the Pi Camera Modules is that they are easily compatible with the Raspberry Pi boards. This is true for two reasons. First, they use the standard CSI camera

connector to interface with the Raspberry Pi boards. Since this is an open standard and the designs for these cameras are completely public, many people have been able to write software which interfaces with these cameras. Secondly, since these cameras were created by the same people that created the Raspberry Pi boards that they interface, the experience of using them is truly plug in play. This saves time and translates directly to a higher rate of success [11].

The main downside to this camera is found in its lack of post processing power. The proprietary post processing that is contained within products such as the Google Pixel and Apple iPhone have been developed over many years to turn the images coming from these tiny lenses into decent looking phones. Unlike these tech giants, the Raspberry Pi Foundation does not have the time or resources to create super high quality image processors, and so the end result suffers. Though the image samples from this camera are not terrible, it is much less versatile than a phone camera or a nice DSLR [11].

Figure 11 shows the Raspberry Pi Camera Module 2. This camera module looks visually identical to the third version. As the image clearly shows, the cable that connects it is short. This would need to be lengthened for this to be implemented in the project. Fortunately, this is easy to accomplish.

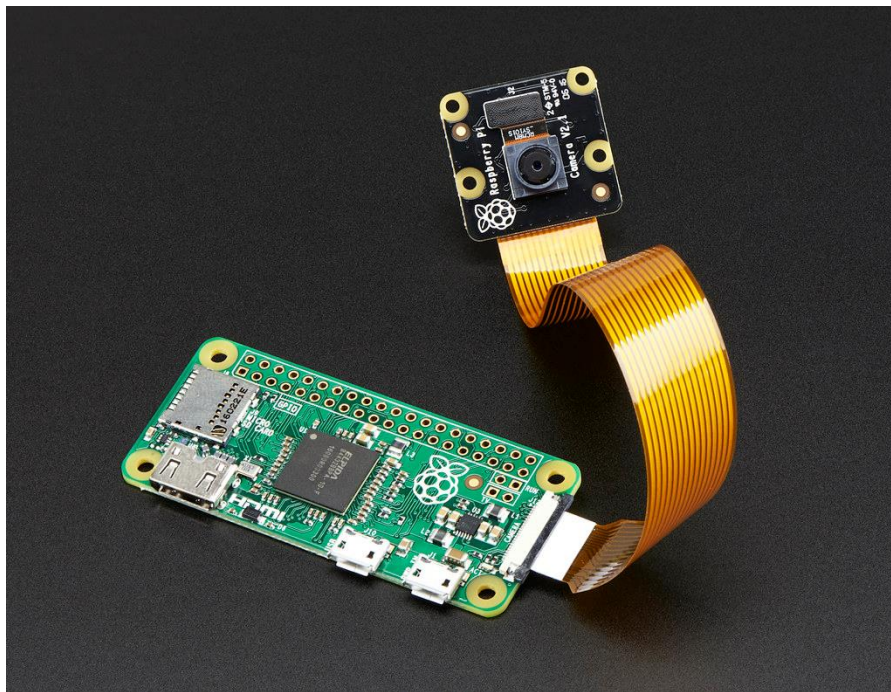


Figure 11: Pi Cam Example by Adafruit Industries under CC 2.0

4.2.3 Raspberry Pi HQ Camera

Most of the information about this camera is identical to the Pi Cam 3. Where this module differs is in its sensor and in its ability to have additional lenses added to it. Unfortunately, the larger and improved sensor can only go so far. Since the sensor within the HQ Camera is still relatively small and the price is often more than twice that of the Pi Cam 3, it is hard to justify the HQ Camera over the regular Pi Cam 3 without a very specific use case. Additionally, the focus on this camera must be adjusted manually by the operator, which would add either an additional level of

interaction that the user needs to think about, or an additional (and potentially difficult) engineering step to make the focus automatic. Nevertheless, there are some notable improvements, thus, this camera will still be considered [12].

4.2.4 Arducam 64MP Autofocus Camera

Like the Raspberry Pi HQ, this camera attempts to make up for the limitations of the Pi Cam 3. But unlike the HQ Camera which increases the size of the sensor and the diversity of the lenses that can be used, Arducam's camera drastically increases the megapixel count. While increasing the pixel count more than four-fold sounds great on paper, the problems of the Pi Cam 3 remain: the sensor size is still relatively small, and the image processing is still relatively limited. Therefore, the individual pixels are much smaller than that of the regular Pi Cam 3. As such, the resulting photos are not greatly improved over the regular Pi Cam 3 [13].

4.2.5 DSLR – Canon T7i

Stepping away from the idea of using one of the Pi Camera Modules or one of the related cameras, another option up for consideration is a DSLR camera. Though this option is considerably more expensive, DSLRs are also much more capable than the small cameras mentioned previously. Since the result of the product (a 3D representation of the object) is directly affected by the image quality of the source material, if there is a way to increase this quality then it should be pursued by the team.

With that in mind, the Canon T7i was researched and tested. The reason for testing this specific camera was simple: this camera was easily available at the local library. The Canon T7i boasts a twenty-four-megapixel camera and, of course, an adjustable aperture. Though it is hard to convey on paper, the image quality is drastically higher with the DSLR. Not only is the sensor much larger, the zoom and aperture are manually adjustable. This means that the system can be more flexible since the camera has more adjustability.

One potential drawback of using a DSLR is that these cameras are generally not purpose-built for use with minicomputers like the Raspberry Pi. Canon does not have any easily accessible software which allows for direct access to the cameras from a device such as the pi. However, there are several open-source software options which have been developed to solve this problem. One notable software solution is gphoto2. This application allows for extensive control of the DSLR cameras, and it is especially good at interfacing with Canon cameras.

Another major drawback of this option is the cost. The camera tested for this project retails for around nine-hundred dollars on Amazon. That price extends well beyond the scope of this project. Fortunately, the used market has many options available on it. To combat the excessive price of the Canon T7i, it would likely be bought second hand from an online seller.

4.2.6 Comparing the Options

Table 7 does its best to summarize the findings that have been set forth above. Where specific values were possible, they have been added.

Table 7: Comparing the Camera Options					
	iPhone 12	Pi Cam 3	HQ Camera	Arducam	DSLR
Aperture	$f/1.6$	$f/1.8$	Variable	$f/1.8$	Variable
Mega Pixels	12MP	12MP	12MP	64MP	20MP
Image Quality	Good	Fine	Fine	Fine	Great
Price	\$0 - \$1000	\$25	\$50	\$50	\$200 - \$900
Usability	Low...needs an app	Excellent	OK	Excellent	Excellent

The most important thing to note is that factors such as image processing and cost are difficult to quantify. With the hardware behind image processing in items such as phones and DSLRs being proprietary, there is not a generally accepted or realistic way to compare these with numbers. Additionally, while most people do not have a Raspberry Pi Cam sitting around, it is much more likely that the general consumer has a DSLR they could temporarily hook up to this project. Furthermore, it is almost guaranteed that someone owns a phone they could use for this. Thus, putting generalizations in the chart is practically unavoidable.

4.2.7 Final Decision

With all this in mind, the team decided to push forward with the DSLR. Though its price is well above that of the other options, the image quality is also much better. With the potential for saving money by utilizing the used market, this option seemed to have the best potential. Additionally, with the ease of use provided by gphoto2, the Pi Cam 3 and similar options lost their competitive edge in the usability category.

4.3 Microcontroller Unit (MCU)

The MCU will be the device that controls the entire project as it will run the code that sends commands to all the other parts such as the motors, the user interface, and the camera. There are multiple important functions that the microcontroller unit must be able to handle. These include controlling three stepper motors, communicating with the camera to capture pictures, sending the pictures to a computer for processing, and taking in user input to control the entire picture taking process. The ability of the microcontroller to control all these elements is important, as the device needs to be able to complete the picture capturing process by itself without relying on an outside computer, as that is only used for running the model generation algorithm. If a tether to an outside computer system were required, the portability of the project would be compromised. The following section contains a detailed analysis of various aspects of the project that the microcontroller needs to be able to handle. This section is rather long due to the importance of the microcontroller in this project.

4.3.1 Power Usage

Our project will be designed to operate in a home, studio, or office, which means that we do not need to minimize power consumptions with the goal of powering the device via a battery. We will

be able to connect the device to a wall outlet, and so a power supply will be needed to convert the AC power to usable DC power to feed to microcontroller, and the connected motors and sensors. This means we will need to design a power supply, which is elaborated on in the [Power System](#) section, that will power the board from a wall outlet. In other words, we will not have to worry about having a microcontroller that can be powered by a battery and therefore will not be limited to lower voltage devices. However, that is not to say that power should be used excessively. The system should not use more power than is needed to both minimize the size of the device and the environmental impact it has.

4.3.2 Motor Control

The microcontroller must be able to give commands to three motors—the details of these commands are mentioned in the [Motors](#) section. One of the motors will control the turntable, one will control the height of the lift arm which changes the camera height, and the last will control the angle of the camera on the arm. The microcontroller will most likely not be able to power the motor drivers by itself. However, this is not necessary as a separate power supply can be used for that purpose. The microcontroller does need to have enough pins to connect to the motor drivers though. The number of required pins varies depending on the type of motor, so the motor selection could impact our selection of the microcontroller. Though, most microcontrollers have plenty of pins, so this will likely not be an issue.

4.3.3 Sensor Control

Another thing that the microcontroller must be able to accomplish is reading signals from the auto homing sensors mentioned in the section [Auto-Homing Sensor](#). The microcontroller needs to connect to the sensor and take input that it can use within its software to calibrate the motor positions. The microcontroller needs to have enough pins to connect to the sensor. The sensor may be either digital or analog output, and thus the microcontroller needs to have a digital or analog general-purpose pin that matches the sensor so it can be configured to take input from the sensor. Most likely a simple digital pin will be enough.

Additionally, the microcontroller will need to be able to interface with a camera. As discussed in the previous section, the camera will most likely be a DSLR. This means that the micro controller should ideally have a USB interface. Additionally, the microcontroller should also be able to run the gphoto2 library (or some close equivalent) to communicate with the microcontroller. It is possible that a simpler camera, like the Pi Cam 3, will be selected. If that happens, the requirement for USB will be relaxed; however, some alternative, such as CSI, will be required instead.

4.3.4 Communication

The microcontroller will need to communicate with a separate computer to send the pictures that the project takes to be processed by the modelling software. There are a few different ways that this could be accomplished. There could either be a wired or wireless connection. The amount of data being sent will not be unreasonable, so there is no need for exotic cables or connection protocols.

One possible approach to a communication system would be to use one of the many common wireless protocols. Wi-Fi is a common standard that has been adopted by almost every device. With the general adoption of Wi-Fi 4, 5 and 6, speeds are fast enough that no bottleneck would be

in consideration. Another option is Bluetooth. While Bluetooth is nowhere near as performant as Wi-Fi, it is intended for close-range performance, and it would not result in much of a slowdown. The main benefit of both options is that they remove the tether which would be induced by a wire (or wires).

However, one more option, which has the potential to be simpler and more reliable, is USB. Universal Standard Bus, or USB for short, is generally easier to work with, and since there are no wireless connection protocols to deal with, it can be more reliable than wireless connections. However, this will create wires that take up space and limit the flexibility of the setup. Table 8 summarizes these comparisons.

Table 8: Comparison of Communication Standards			
	Wi-Fi 802.11n	Bluetooth LE 5.0	USB 2.0
Reliability	Good	Low	High
Throughput	Up to 600Mbps [14]	1.4Mbps [15]	480Mbps [16]

For this project, the most enticing option is to go wireless with Wi-Fi. Due to the limitations and lower reliability of Bluetooth, the Wi-Fi standard is more desirable. Though USB could be used, the addition of a wire that must connect to a computer makes this less desirable. In the situation where the computer and photogrammetry system cannot be next to each other, the USB connection would be a great hinderance.

4.3.5 The Operating System Requirements

Another aspect of the microcontroller that needs to be considered is the operating system that runs on it. One of the most important engineering design constraints for a modern operating system is its ability to provide an abstraction layer between the bare metal hardware and the software (or programs). The abstraction provides a cushion that keeps software developers from having to worry about all the complexities of the low-level hardware using drivers and system calls. Thus, the operating system needs to provide abstractions that allow for access to Wi-Fi hardware, USB hardware and the basic components of the microcontroller like the CPU, RAM, storage, and similar components. In the case of this project, the operating system of choice must specifically support the gphoto2 library. As mentioned previously, this library enhances the depth of access to more complex DSLR cameras.

4.3.6 A Realistic Design: Separation of Concerns

It is often the case in engineering that assigning too many responsibilities to a single component can lead to less than desirable results. For instance, requiring that a single component be capable of interfacing over high level communication protocols such as USB and Wi-Fi whilst also being able to communicate with simple sensors could be too much for a single microcontroller. From the standpoint of there not being enough computational power, modern technology does not have this limitation. However, from a developmental standpoint, if the high-level computation and low-level sensor interfaces could be separated into two devices, this would simplify development. It would enable the work to be more easily split between members of the team.

As such, the project will utilize both an ARM-based Linux minicomputer and a dedicated microcontroller. The Raspberry Pi 3 is an excellent example of the former, and the Arduino Uno is an excellent example of the latter. In combination, these two components will be able to handle all aspects of the project. In reference to the original diagram of the project introduced near the beginning of the paper, the role of the two components is quite straightforward. The more complex ARM computer will be running Linux. This will provide access to libraries such as gphoto2, access to protocols that make the use of Wi-Fi communication easier, and a great place to run the main control program which ultimately sends all the commands that make things happen. The microcontroller, then, will be able to focus on simply controlling the sensors, motors, and the LCD display. Finally, between the two devices there will be a layer of communication in which the ARM computer will take the role of master, and the microcontroller will take the role of slave. The ARM computer will send commands to the microcontroller and listen for changes that would result in communications or related events occurring.

By separating requirements between two aspects of the project, these parts can be more easily developed. Someone can work on the parts that are related to the ARM computer while another user works on the parts related to sensor control. There is, however, a layer of added complexity with this approach: there now needs to be a layer of communication between the two components. But this tradeoff is justified by the resulting boost in developmental speed and the simplicity that is added by using parts for what they are specifically built for. A Raspberry Pi is excellent at running Linux and complicated programs like those that interface with cameras, but it is a bit burdensome to try and directly control motors from it. The opposite is true of a microcontroller: it is excellent for controlling simple motors and sensors but falls short when trying to interface with complex devices such as a Wi-Fi card or DSLR camera.

4.3.7 Choosing the ARM-Based Linux Minicomputer

In this section, multiple options for the Linux based ARM minicomputer are compared to offer a clear comparison between the various options that are generally available. The main ones that were extensively researched were the Raspberry Pi 3, the Raspberry Pi 4, and the Orange Pi 3. All these boards implement various ARM processors and are available to buy immediately. The problem, however, is that demand for these types of systems has skyrocketed at the time of writing this paper. The means that boards like the Raspberry Pi 4, which could have been purchased for thirty-five dollars, cost over one hundred thirty-five dollars. Table 9 summarizes the options between these modules.

Table 9: ARM-Based Minicomputer Comparison			
	Raspberry Pi 3 [17]	Raspberry Pi 4 [18]	Orange Pi 3 LTS [19]
Core Count	4-Core	4-Core	4-Core
ARM CPU	BCM2837	BCM2711	Allwinner H6
Architecture	Cortex-A53	Cortex-A72	Cortex-A53
RAM	1GB LPDDR3	1-8GB LPDDR4	2GB LPDDR3
Wi-Fi	802.11n	802.11ac	802.11ac

Bluetooth Version	5.0	5.0	5.0
40 pin GPIO	Yes	Yes	26 Pin
Ports	<ul style="list-style-type: none"> • 4x USB 2.0 • HDMI 1.4b • 100Mbps Ethernet • 2-lane MIPI DSI display port • 2-lane MIPI CSI camera port • Micro USB (for power) 	<ul style="list-style-type: none"> • 2x USB 3.0 • 2x USB 2.0 • 2x micro-HDMI • Gigabit Ethernet • 2-lane MIPI DSI display port • 2-lane MIPI CSI camera port • USB-C (for power) 	<ul style="list-style-type: none"> • 1x USB 3.0 • 2x USB 2.0 • HDMI 2.0a • Gigabit Ethernet • USB-C (for power)
Dimensions (mm)	85 by 56	85 by 56	85 by 56
Price	~\$60	~\$135	\$48

Something else that should be noted here is that it is difficult to quantify the real-world performance of these chips. Though one may seem to be powerful enough to handle the project's workload, this may not be the case in practice. Fortunately, all these boards are fairly similar to each other, so switching from a less powerful one to a more powerful one down the road is very much an option.

Given that the processing power required by the project is not that extensive, the Raspberry Pi 3 (shown in Figure 12) should be more than sufficient for this project. Since the price of a Raspberry Pi 4 is so inflated at the time of writing this paper, it seems reasonable to stick with the less powerful chip. Again, since the operating systems which run on these boards abstract the differences between them, it is not difficult to swap the boards out in the case that the performance is not adequate.

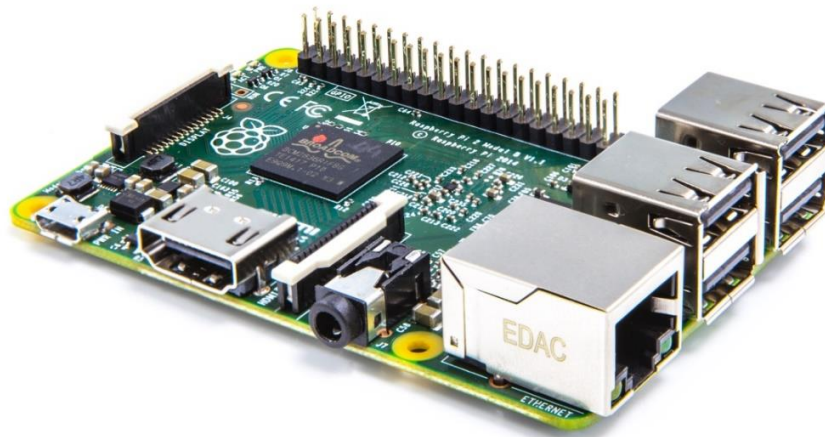


Figure 12: Raspberry Pi 3 Model B by Onepiece84 under CC 4.0

The Orange Pi 3 LTS seems incredibly tempting in the above comparison. It has both higher specs than the Raspberry Pi 3 and a lower cost. But there are a few reasons why it is not going to be used which a spec sheet simply cannot convey. Firstly, the Orange Pi 3 does not have the same size userbase that the Raspberry Pi boards do. This means that if the team runs into a problem, there would be less resources to help with this. Secondly, the Orange Pi has less connectivity. It lacks the full 40 pin GPIO pin layout present on the other two boards, and it does not include the interfaces for the display or camera modules. Though the team is not planning on using any of these features, they all present easy backup solutions if the current design needs tweaking. Finally, the team has access to a Raspberry Pi 3, and since it meets all the requirements, it makes sense to use what is available.

4.3.8 Choosing the Accompanying Microcontroller

The chosen microcontroller determines the eventual workflow for how the project's software is written and tested. It also governs how the other hardware is designed and wired-up. The processor has an architecture that determines how the code is written and runs. It also has pins that determine how hardware can interface with it. It is important to select a microcontroller that can interface with the motors and sensors and run the necessary code.

One microcontroller being considered is the ATmega2560. This processor is implemented on the Arduino® Mega 2560 Rev3, a development board which could be used for testing purposes. Another microcontroller option is the MSP430FR6989. Again, this chip is implemented on the MSP430 Launchpad Board by Texas Instruments, which would allow for testing the before implementing it into a printed circuit board. Table 10 summarizes our options.

Table 10: Microcontroller Comparison		
	ATmega2560	MSP430FR6989
GPIO Pins	54	83
Operating Voltage	5V	1.8 - 3.6V
Architecture	8-bit AVR-RISC	16-bit RISC
Communication Protocols	UART, SPI, I2C	UART, SPI, I2C
Program Memory	256kB	128kB
Ease of Programming	Simple	More Technical

The board that will be used in this project is going to be the ATmega2560 for its ease of use and relative availability. Figure 13 shows this item.

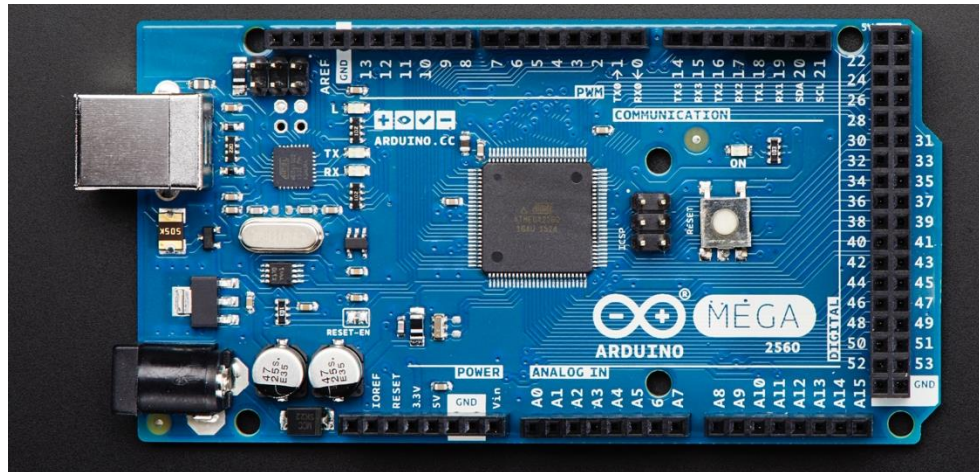


Figure 13: ATmega2560 Board by Adafruit Industries under CC 2.0

Though the MSP430 has an impressive list of features, everyone on this team has had experience with both boards, and the user experience of the ATmega2560 board is greatly preferred.

4.3.9 The Operating System Revisited

The Raspberry Pi 3 has several operating systems that it is compatible with. The most common operating system to be run on Raspberry Pi boards is Raspberry OS. This operating system is developed by the Raspberry Foundation and is based on Debian Linux. The Debian base provides a simple yet stable base resulting in a robust operating system. Additionally, since it is maintained by the same organization developing the computers it runs on, it has fantastic compatibility with the Raspberry Pi computers [20].

One notable alternative to Raspberry OS is Ubuntu. Most people have heard of Ubuntu. It is a very common Linux distribution used on a wide variety of hardware for a broad spectrum of purposes. Consequently, there is a large userbase and helpful community that has spawned around Canonical's project (Canonical is the company that develops Ubuntu). Like Raspberry OS, Ubuntu is based on Debian giving it a stable base for developers. In the last few years, an ARM version of Ubuntu has been released and is available as an alternative option for installation on the Raspberry Pi boards [20].

Canonical also releases server versions of Ubuntu. The main difference between the server release and the regular release is its lack of a desktop environment. The desktop environment utilizes a large amount of memory, so it can be great for efficiency to not have to render this on these tiny boards. However, the lack of a desktop environment makes it harder to things done of the board while in the development phase [20].

Finally, there is Manjaro. Manjaro is a roughly Arch-based Linux Operating System. Because it is Linux Arch based, it has access to many bleeding edge technologies that are not immediately available on slowly developed systems like Debian [20]. Table 11 outlines the comparison between these operating systems.

Table 11: Operating System Comparison

	Raspberry OS	Ubuntu	Ubuntu Server	Manjaro
Desktop Env.	Custom	Gnome	None	KDE
Compatibility	Excellent	Good	Good	Good
Ease of Use	Excellent	Poor (very slow)	Fine	Good (slow)
Version	22-02-21	22.04	22.04	Semi-Rolling
Runs gphoto2	Yes	Yes	Yes	Yes
Runs on Pi 3	Yes	Technically	Yes	Ok

Because the team had access to a Raspberry Pi 3 while choosing the operating system, all the above options were able to be tested. Raspberry OS is not the nicest option available; nor is it the most polished. But what it lacks in shine, it makes up for in compatibility and ease of use. Raspberry OS was purpose built for the Raspberry Pi, and it shows. It ran the fastest and presented the least number of challenges compared to the other options, so it was chosen for the project.

4.3.10 The Microcontroller Development Board

The final project will include a custom PCB with a microcontroller and other components. However, a development board would make the software design process much more straightforward, even if we do not implement the board itself into the final design. Using a microcontroller with a robust development board will allow us to more easily develop software to control the motors and camera.

4.3.11 Interfacing the ATMEGA2560 with USB

The Arduino development board includes an interface for connecting to USB devices. This luxury does not exist when the ATMEGA2560 is integrated directly into a PCB. One solution to this problem is adding an additional device which lets the microcontroller interface with USB. Since the designs for Arduinos are open sourced, it is easy to see that they add the USB interface through the addition of the ATMEGA8U2. Another chip which accomplishes the same thing is the FT232HPQ. Both parts are readily available on Digi-Key, and Table 12 demonstrates the differences between these two parts.

Table 12: USB Interface Chips

	ATMEGA8U2	FT232HPQ
Ease of Use	Simple	Needs Custom Design
Connection	SPI, UART/USART	I ² C, FIFO, JTAG, Serial, SPI, UART
USB Version	2.0	2.0
Price	\$3.54	\$4.77

As the table shows, the ATMEGA8U2 is the more desirable of the two options. It is simpler to integrate due to Arduino designs being open sourced. It also costs less and avoids all the extra pins associated with the additional incoming connections. Therefore, this will end up being the converter that is chosen for adding a USB interface to the PCB.

4.4 Motors

For the project, we required the use of motors to satisfy the design requirements. Three motors are required altogether for the project. We require a spinning platform, and a camera that moves up and down on the rig, with the added capability of adjusting the camera angle with respect to the aforementioned table. One motor will be used in the turntable, one to lift the arm, and one to change the angle of the camera. We may be able to utilize some different kinds of motors to mitigate costs and optimize the function of our project.

4.4.1 AC Brushless Motors

One possible implementation of motion is with the use of AC brushless motors. The operating principle behind these types of motors is the use of alternating current (AC) to power the coils of the motors and produce the necessary magnetic field to control them. They are “brushless” because, unlike brushless motors, they rely on external control unit to power and control the coils. Their main advantage is offering smooth and continuous rotation, making them ideal for applications where high speed and smooth operation is atop priority. AC brushless motors, however, offer less precise motion compared to other types of motors, but they compensate for that by offering high torque and high speeds. Moreover, they are also energy efficient; they draw current only where they need to, allowing them to be the perfect implementation for low-energy systems.

On the disadvantages side, brushless motors are more complex, and hence, pricier. In addition, they require more work to be integrated in Arduino project, which is the coding environment this project will be built in. They require an Electronic Speed Controller (ESC), a device designed to serve as the integration layer between MCUs and the motors. However, the level of control and precision offered is generally lower to other motors alternatives. For these reasons, brushless motors are dropped as considerable choice for the project. An investigation of the next alternatives (stepper motors) is better suited for the project’s needs.

4.4.2 Stepper Motors

There are two main types of stepper motors: Steppers and Servos. Servos in general are low power, high-speed motors, and may work for rotating a platform at a constant rate for the duration of the project. However, for the camera movement we will need something a bit more precise. An image of each type of motor is shown below in Figure 14.

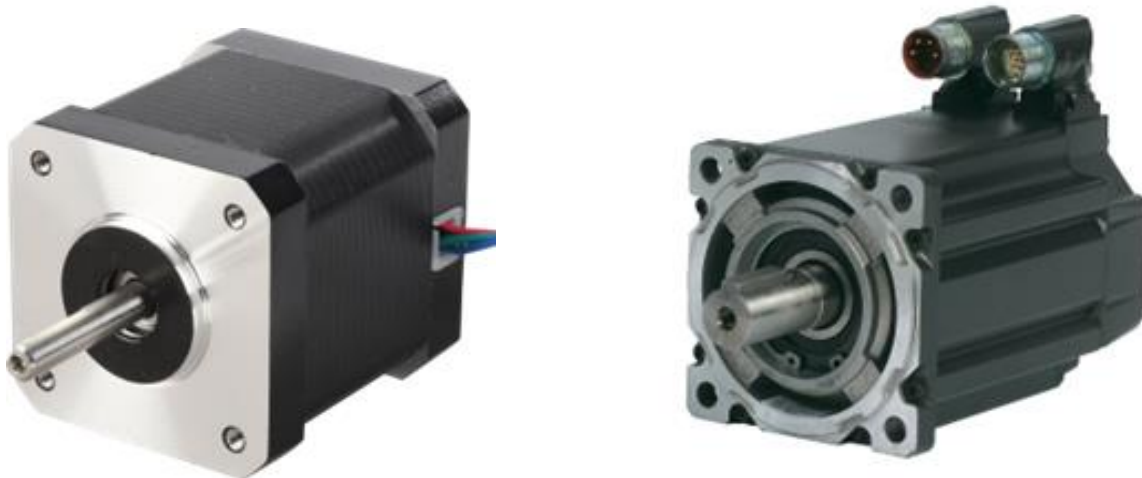


Figure 14: Stepper Motor (Left) vs Servo Motor (Right)

Stepper motors work on the principle of electromagnetism, and they offer precise position control and torque output at low speeds. This is perfect for mobilizing the camera because we would like to be able to fine tune the positions the camera gets into, which will be necessary to implement our design with consistency, and for getting a good quality image. Stepper motors are called such because they rotate in “steps” and you can manipulate these steps to determine exactly how much you want the motor shaft to rotate, thus allowing the user the additional precision. The steps are created by putting a charge through a coil to create a magnetic field, and this magnetic field draws a magnet within the motor to certain positions. Alternating the direction of the current through the coil and which coils have the current running through them changes the magnetic field within the motor and rotates the magnet.

How these coil windings are set up determines the type of stepper motor we are dealing with. There are two types of stepper motors: Unipolar and Bipolar stepper motors. The windings in the Bipolar are connected on each end of the coil and nowhere else. The Unipolar contains an additional connection, shared by multiple coils, which is attached in the middle of each coil. This setup is reflected in the number of wires that are used to control each motor. The Bipolar contains four wires that need to be serviced, and the Unipolar contains five or six. Bipolar stepper motors generally offer more torque. The different coil setups are shown below in Figure 15.

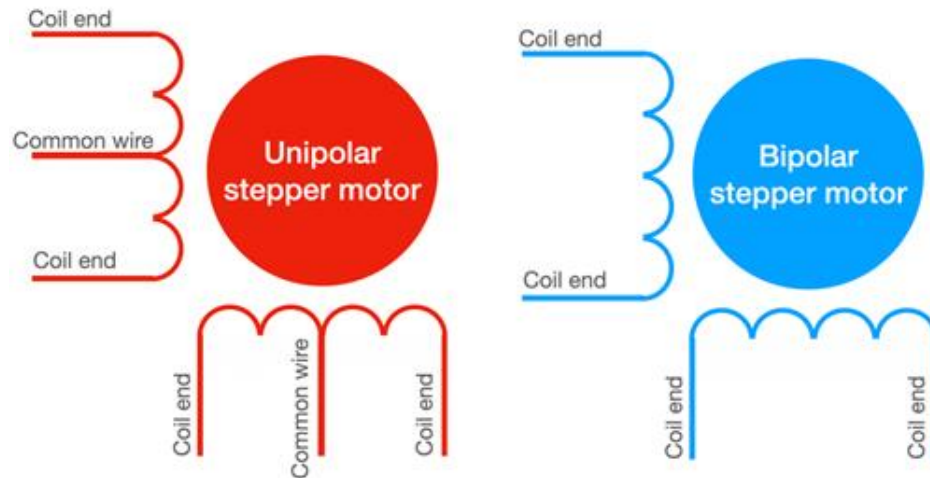


Figure 15: The Different Coil Setups for Unipolar and Bipolar Stepper Motors

Naturally, to keep a smaller footprint on our printed circuit board (PCB), we chose to utilize bipolar stepper motors. Another factor being the weight of the camera, bipolar stepper motors will provide the necessary holding torque that will allow us to move the rig with precision and provide repeatable, consistent camera angles and positions.

Usually, it is not a good idea to set up your motor(s) in a way that they are drawing power from the PCB. This can create issues because the power and amperage required by the motor to run at certain settings does not match what the PCB can usually output. For this reason, a sperate power supply is required for the motors. Figure 16 is a typical power supply.



Figure 16: Power supply for a Bipolar Motor

One thing we would like to research more is the implementation of a motor driver on our PCB board itself. This would reduce the overall footprint of our project but increase our PCB footprint. We are also unsure of how much this would affect the cost of our PCB.

4.4.3 Part Selection: Stepper Motor

When selecting a motor, we must decide primarily what kind of motor we want, stepper or servo, then, if stepper, like ours, bipolar or unipolar. These decisions have been reasoned and explored above. Since we now know what general type of motor we want, we now get to the specific capabilities, power specifications, shaft length & shape, the form factor, and price. A quick search on amazon offered up several options that seem more than powerful enough, so the decision was made to purchase a cost-effective option primarily.

When looking at two of the common motor types, we were faced with the NEMA-17 and NEMA-23 standards. There are a variety of motors within each of these categories, and both could do the job for our project. However, generally, NEMA-23 motors are more powerful, larger, more precise, and more expensive than what we are looking for. Those motors are more powerful and pack way more torque than we would need for the application of moving a camera and motor, which collectively weigh about 1.5kg. NEMA-23 motors do offer more precision and accuracy; however, this will not impact on our application. NEMA-23 motors have a higher precision and accuracy because they are intended to run continuously and for longer amounts of time. This makes the extra robustness in accuracy and precision necessary, because over a long period of time, a small lack of precision can result in a large offset. NEMA-17 motors are a better option for a few practical reasons. They are smaller, and so their form factor matches the goals envisioned by the project better than their larger cousins. While less accurate, they are still accurate enough for precise positioning of the camera and angle for the desired photographs to be taken. And finally, while the holding torque is lower for the NEMA-17 motors, they are still powerful enough to handle the loads that our project requires of them. Table 13 compares the NEMA-17 and NEMA-23 motors.

Table 13: Stepper Motor Comparison		
	NEMA-17	NEMA-23
Supplier	StepperOnline	StepperOnline
Cost	\$10.19	\$25.99
Weight	7 - 12oz.	1.3 - 2.6lbs.
Size (Mounting Face)	1.7 in.	2.3 in.
Holding Torque	0.2 - 0.6 Nm	0.5 - 3 Nm
Max Speed Range	1000 - 5000 rpm	1000 – 3000 rpm
Gear box	No	Yes
Ease of Programming	Simple	Simple

These factors are somewhat tied together with other aspects of the project as well. For example, had the larger NEMA-23 motors been selected, the power requirements would be higher, making for a larger, more expensive power supply after spending more funds on the motors. This would

also produce more heat, and possibly introduce other issues, since NEMA-23 motors generally have different applications. NEMA-17 motors are a better fit for this project all in all.

This led us to select a NEMA-17 (17HE15-1504S) motor. This common motor was found on Amazon at a reasonable price and will provide for our requirements quite nicely. The current drawn by this motor is quite reasonable as well and considering that we will implement three separate motors in our design, this enables us to use a relatively cheap power supply.

4.5 Motor Drivers

Another topic in relation to the motors that must be brought up is motor drivers. Motors are not well suited to be hooked up directly to the MCUs, both for power reasons, and for signal reasons. The power dilemma has been brought up already, but the signal issue is one that still needs to be addressed. What we require is a motor driver. Motor drivers are electronic circuits that are responsible for controlling motors, and hence, eliminating the need for MCUs to dedicate resources to control them. The driver's main functionality is to provide the necessary current and voltage to the motor's terminals. The role of the MCU is then to send the control signals to the driver which in turn converts those signals into readable signals for the motor. These signals are what "drive" the motor, hence the name. These motor drivers also are correlated to the steps the motor takes and amperage that flows through the coil windings of the motor. This is called "Micro-stepping" that allows for finer control of the motion of the motor. It functions by dividing a full step of the motor into smaller steps, which allows for precise and smooth motion. The number of smaller steps is different from one driver module to another. Moreover, this has a direct impact on the holding torque of the motor and can also be a factor to consider with respect to how hot the motor gets. Therefore, a design that accommodates heat dissipation is necessary. Most common drivers come with heat sinks installed, but further airflow design consideration is also important. Example of these drivers are shown in Figure 17.

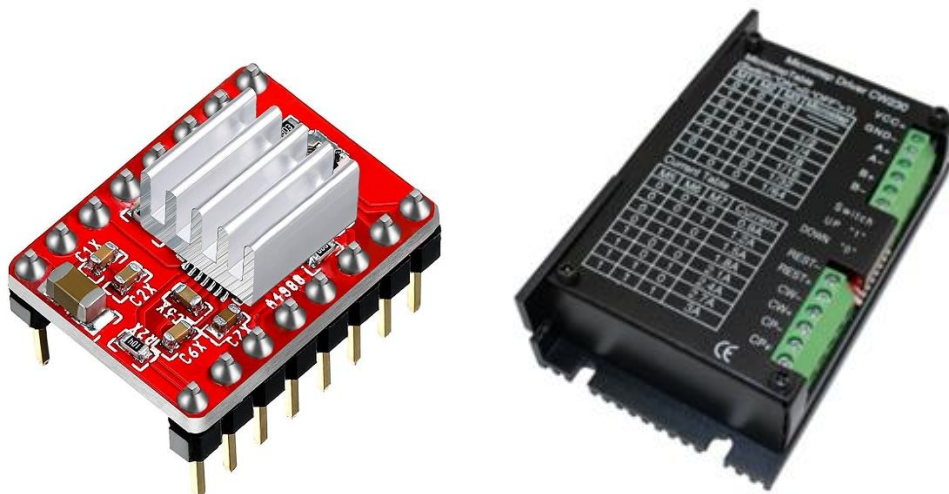


Figure 17: Bipolar Motor Drivers (A4988 vs TB6600)

In addition, safety features are essential for integrating the motors into an existing system. Fortunately, most common drivers come with safety features, such as overcurrent and overheating protection. As for selecting a motor driver, the parameters to select a driver that need to be considered are the amperage that it is capable of outputting, footprint, and whether or not it is compatible with the motor we have selected. One of the most popular drivers for Nema-17 Stepper Motor is the A4988 Driver, a small form factor driver compatible with most MCUs, it comes with a heat sink, and has current limit adjustment. Another popular driver is the DRV8825 driver of similar form factor as the A4988 driver, offering up to 1/32 stepping mode. On a different form factor is the TB6600 driver, a slightly larger but versatile driver where most settings, like micro-stepping, can be changed without accessing the code. Table 14 compares motor drivers' most relevant features.

Table 14: Motor Drivers Comparison			
	A4988	DRV8825	TB66000
Supplier	WWZMBDiB	HiLetgo	Jusnboir
Cost	\$2.66	\$3.00	\$9.98
Supported Micro-Stepping	Up to 1/16	Up to 1/32	Up to 1/32
Size	15.24 x 20.32 mm	15 x 20 mm	96 x 56 x 35mm

Due to similar budget considerations that we mentioned in the motor selection, we decided to go with the A4988 Driver, as it offers all the requirement needed for it to be integrated in the project while also being the least costly option between the drivers. The TB6600 driver is a good second option, however, its size requires further consideration and accommodation in the design that is not preferable at the moment. At the current projected implementation of the project, the A4988 driver will be sufficient for the project and provide us with an early-stage prototype of development. The footprint is also considerably smaller allowing it to be easily integrable in the PCB design, since it will leave more room for our other circuitry.

4.6 Auto-Homing Sensor

The purpose of the auto-homing sensor is to add the position resetting feature for the camera movement. This allows for a common starting point that the program will always start from. Details on how the sensor works are described in the technology investigation section 3.1.4. There are several options that can be implemented in the project. We explore three options here, starting with a limit switch sensor.

4.6.1 Limit Switch Sensor

A limit switch sensor core functionality is to indicate when a stop point has been reached. When the limit switch is pressed, an electrical signal is generated, either HIGH (closed switch) or LOW (open switch). The switch has three terminals: common pin, normally open (NO), and normally closed (NC) pins shown in Figure 18: Limit Switch. The circuit is formed with either common/NO pins or common/NC pins. The former configuration raises a HIGH signal when contact is initiated,

while later raises a LOW signal when contact is initiated. The advantages of a limit switch are it requires little to no calibration. A good module on the market is the Omron Snap Action Switch SS-5GLT. Its implementation in our project is installed at the lowest point on the linear actuator. This does mean that we might need to 3D print some parts to accommodate it into the design.

4.6.2 Hall Effect Sensor

Just like the limit switch sensor, the hall effect sensor can indicate a reach of position. Compared to the mechanical limit switch, the hall effect sensor is an electrical switch that requires no contact because it realizes on detecting the presence of a magnetic field. When the sensor is placed within a magnetic field it sends a signal through its output pin indicating the presence of magnetic field. Its range is very limited however, requiring the magnetic to be very close to the sensor. The code is more advanced compared to the limit switch sensor. A popular module in the market is the *HALL EFFECT ANALOG TO92-3*. Its implementation in the project is to place it at a specific location in the vertical rig with a magnetic attached close to the camera. It is relatively easy to setup and configure. Moreover, there is a different module of the sensor where it includes an LED light to indicate when the sensor is activated. This might prove to be useful for debugging and problem finding. Its cost is within \$2 dollars but finding it in single units is difficult. Figure 18 shows an image of both type of sensors, while Table 15 compares these sensors.

For our project, we decided to go with the hall effect sensor for its small size and the convenient LED light that will prove to be useful in the prototyping stage. However, this module might require additional coding, but its advantages overcome this issue with one of them being its precise detection.

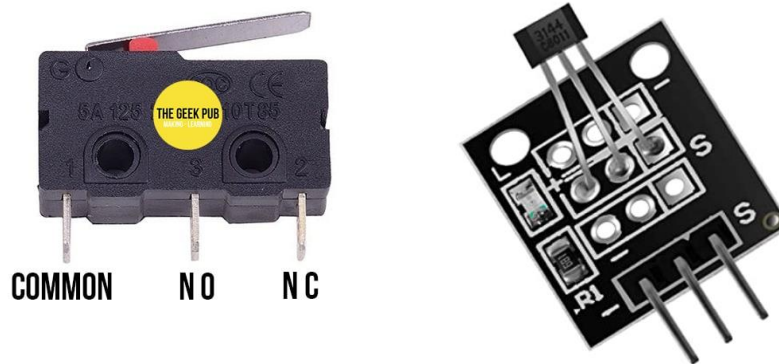


Figure 18: Limit Switch vs Hall Effect Sensor with LED

Table 15: Auto-Homing Sensors Options			
	Limit Switch	Hall-Effect	Hall-Effect Sensor Module
Module Name	Omron Snap Action Switch SS-5GLT	DRV5053PAQLPG Texas Instruments	3144E Hall Effect Sensor KY-003 (20pcs)
Cost	\$2.72	\$1.80	\$6

Ease of Integration	Medium Size 3D Printed Parts Simple Code	Small Size 3D Printed Parts Relatively Advance Code
Detection Sensitivity	Big movements	Small movements (precise)
Detection Range	Detection on pressure	Magnet must be close to the sensor

4.7 Display Module

The device will need to be able to take input from the user for certain options to be selected in the photogrammetry process, such as the number of pictures to be taken, and to start the picture-taking process. A display module must be present to allow the user to interact with the device and select settings. Some things that must be considered when choosing a display module are how difficult it is to implement, whether we have enough pins to connect the display to the microcontroller, and how intuitive the interface will be for the user to operate. A few of our options for displays are discussed below.

4.7.1 LCD Screen with Buttons

One option for our display is to use an LCD screen to display the user interface. This would make it simple to display the options to the user as they can be shown directly on the stand. It would be easier to implement than other options as we would only need a basic graphics library to display text on the screen. The screen only needs to be large enough so that it is easy to read the text, and it must have a high enough resolution to be able to display the text options clearly. To allow the user to select options and give the device input, we would have buttons next to the display. The display would list options in a menu that could be navigated with directional buttons and selected with a confirm button. This would require more pins to connect the buttons to the microcontroller. This approach uses a simpler display unit but requires more pins and more separate interacting parts to function.

A specific hardware option for an LCD screen would be the 2.2" 18-bit color TFT LCD display with microSD card breakout - EYESPI Connector. This LCD screen module is a color display with 320x240 resolution. The display uses 4 pin SPI communication but would require additional pins for the buttons to connect to the microcontroller and has a graphics library the supports the SPI communication.

4.7.2 Touch Screen

Another option for our display is a touch screen. This would reduce the number of separate components that need to work together, as buttons would not be needed. The touch screen would have a driver that would handle the display output for the user and the input via touch. This may be more difficult than a screen with separate buttons, as the software would need to display buttons and calibrate the touch screen for user input. The screen would also need to be larger, as it would have to display text with a larger area for the user to press. The display would have options displayed as digital buttons to be pressed, which would provide a more intuitive user experience. This approach is overall simpler in terms of hardware but may be more difficult to design the software.

A specific hardware option for a touch screen would be the 3.5" TFT 320x480 + Touchscreen Breakout Board w/MicroSD Socket – HXD8357D. This touchscreen module is a color display with 320x480 resolution with a resistive touchscreen. The display has a built-in controller and has two communication modes: 8-bit and SPI. The 8-bit connection requires 12-13 pins to read and write to the display, while the SPI connection requires 8-9 pins to communicate but is slower than 8-bit. The display does not need to update very quickly or respond to input at a very fast rate, so either connection model would work for our purposes. The display has a graphics library written for both the 8-bit and SPI interface, as well as a library for the touch screen.

4.7.3 Computer Interface

Another option for our display is to not have the user interface on the station itself, but rather on the separate computer that runs the 3D reconstruction software. This would reduce the need for any additional screens or input methods connected to the project, as the computer already has a user interface. This would cut some costs for hardware, as it makes use of hardware that is already a part of the project and would thus be a more software heavy approach. This would, however, require more complicated software to be developed for the project as a computer application that communicates wirelessly with the station's microcontroller may be more challenging to develop than software that runs on the microcontroller itself and interfaces with the user interface through wired connections. This approach reduces the extra hardware required but introduces more complicated software development

4.7.4 Phone Application Interface

Another option for our display, without being an interface on the station, is to have the interface as a mobile phone application. Like the computer interface, this would reduce the amount of hardware that needs to be designed and included in the station. The use of a phone application would shift the focus from a hardware to software approach for our development. However, much like the computer interface, it requires more complicated software to be developed than that which would run on a microcontroller. App development may take a lot of time, as it requires another type of coding that works on mobile devices and may need to meet certain standards for being published and accessible on standard mobile devices. This approach reduces the extra hardware required but introduces complicated software development that may greatly extend development time.

4.7.5 Final Comparison and Decision

We examined our options for the display module and compared what research and design each would necessitate and how each one would affect the user experience. We decided that having an extra piece of hardware attached to the stand would be preferable to having a separate interface on the computer or a mobile device. This narrows our decision down to an LCD screen with separate buttons or a touch screen.

The screen itself needs to be able to connect to the microcontroller to allow direct communication between the processor and the interface. If we choose to use an LCD screen without touch capabilities, that will require another method of collecting user input, likely buttons. The buttons will need to connect to the processor and so the microcontroller will need enough pins to support

the screen and buttons. Button input would be managed through software in the microcontroller, which would then be reflected in the display screen. Table 16 lists and compares our options.

Table 16: Display Module Options		
	LCD Screen	Touch Screen
Module Name	<u>2.2" 18-bit color TFT LCD Display</u>	<u>3.5" TFT 320x480 Touchscreen HXD8357D</u>
Screen Dimensions	55.23mm x 40mm	56mm x 85mm
Resolution	320x240	320x480
Pins for Communication	SPI: 4 Pins More Required for Buttons	8-bit: 12-13 Pins SPI: 8-9 Pins
Libraries	Graphics Library	Graphics Libraries Touch Screen Library

After analyzing our options for the display module, we decided that a touch screen would be the best choice. This would reduce the complexity added by connecting a purely display screen and separate buttons that need to interact and would improve the user experience by creating a more intuitive interface. The user would be able to press dynamically drawn digital buttons on the touch screen instead of moving a cursor with physical buttons. We will use the HXD8357D touch screen as the display, which includes multiple options for communicating and has prepared libraries.

4.8 Power System

A project without a sufficient and reliable power system is a failure even if every other subsystem in the projects has been designed and tested successfully. The purpose of a power system in a project is to provide a reliable and stable power source for the electrical components used in the project. With a variety of components, it is natural that there would be a range in the voltages and currents that are demanded by the components, and hence, the power system must be able to provide said voltages and currents for components to operate correctly and efficiently.

4.8.1 System's Power Analysis

As mentioned in the project description, the project is intended for indoor use, which simplifies the decision to opt for an Alternating Current (AC) source from an electrical outlet rather than a Direct Current (DC) source from a battery. Furthermore, powering the system from a wall-outlet allows for higher power output compared to a battery, as well as providing continuous power delivery which in turn eliminates power fluctuation in the system. To develop the power system, the operational voltage and current of each component must be first identified to select the maximum operational voltage and current which then can be stepped down using regulators to accommodate every other low voltage component. Table 17 outlines the voltage and current demands of the project and Figure 19: Project's Power Conversion Diagram illustrates power conversion in the power system.

Table 17: Project's Voltage and Current Demands		
Component	Voltage	Current
Arduino	5V	200mA
Raspberry Pi 3	5V	Idle: 400mA Max: 2A
Motor (NEMA17)	12V	1.5A – 1.8A
Motor Drivers	5V	(Very small)
Hall-Effect Sensor	5V	(Very small)
Touch Screen	5V	(Very small)

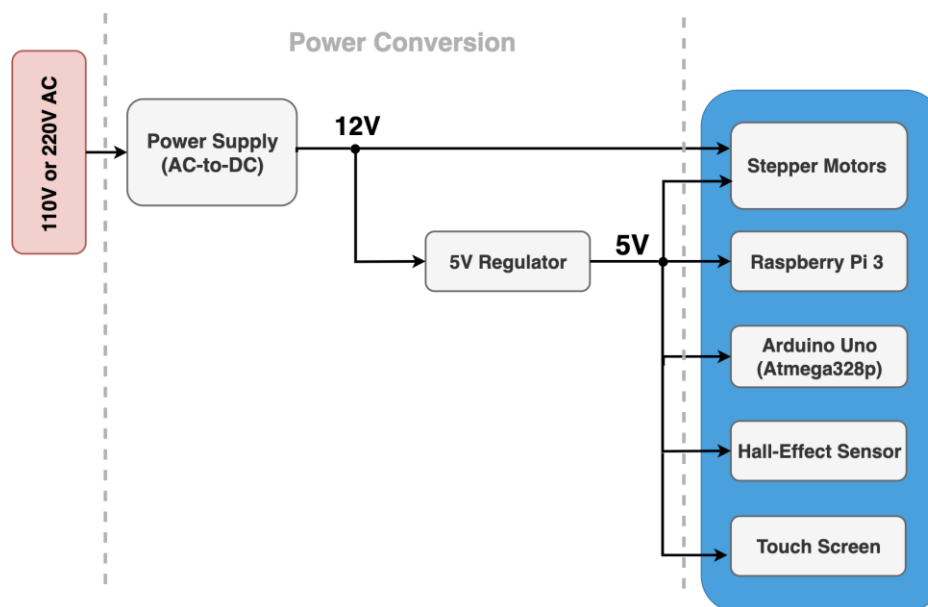


Figure 19: Project's Power Conversion Diagram

Observing the system demands in the table, we can establish 12v as the voltage rating of the system. Current rating, on the other hand, is a bit more complicated. Summing all current running in the system, we come to at least 8A of current. However, this is assuming that all motors, and microcontrollers are in full operating all the time. In reality, all three motors are running at separate times from each other with little to no overlap. Meaning that, for all three motors 2A of current is technically sufficient. In addition, the Raspberry Pi is also not running at full operating all the time, but it will have some overlap with some of the motors, therefore an additional 2A of current is a

good idea. All other components draw very small amounts of current, and they are also not running all the time. In total, 5A is good theoretical current rating for the system. Now to select the power supply and regulators, we have two technological choices: linear vs switching power conversion. Voltage regulators are essential to the overall performance of the system as they provide a stable and constant voltage output independent, to some degree, from the load input voltage, load current fluctuations, and temperature. The next few subsections discuss the choice between these two technologies, describing their underlying operation, performance, and suitable applications, as well as selecting the components to implement the power conversion.

4.8.2 Linear Regulators

Linear regulators maintain a steady output voltage by varying their internal resistance. In other words, it varies the resistance between the input and output voltages, dropping excess voltage and dissipating excess power in the form of heat. The variable resistance device (Control Element) is adjusted by a negative-feedback loop (Control Circuit) according to an error signal between the input and output voltages. Figure 20: A Typical Linear Regulator Circuit shows a typical linear regulator circuit.

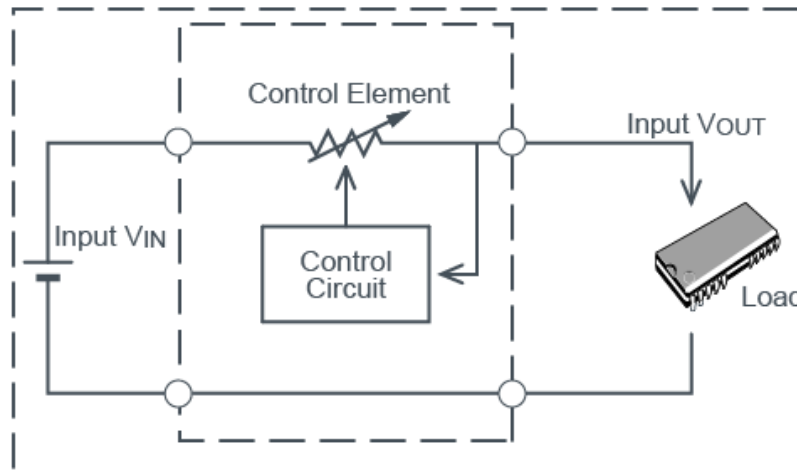


Figure 20: A Typical Linear Regulator Circuit

A typical linear voltage regulator consists of three pins: an input pin, output pin, and ground pin. The internal construction of linear regulators typically consists of an error amplifier as the control circuit, and series pass transistor as the control element. The error amplifier computes the difference between the reference voltage with the output voltage and feeding back to the input of the error amplifier, producing an output signal proportional to the difference between the output voltage and the feedback voltage. The amplifier's output signals control the resistance of the series-pass transistor. In the case of a bipolar transistor, the control signal is the base current. In the case of MOSFET, the control signal is the gate voltage. When the output voltage changes, the error amplifier increases or decreases the base current or gate voltage, changing the transistor's resistance and allowing a corresponding and inversely proportional current change to output line to maintain the desired steady output voltage.

The nature of linear regulator of dissipating power as heat to regulate the voltage make them inefficient. Their efficiency is given by ($\eta = \frac{V_{out}}{V_{in}}$). In addition, there is a minimum dropout between the input and output voltages associated with linear regulators for them to operate currently. Another advantage of linear regulators is their low level of noise output allowing them to be perfect for noise-sensitive applications, as well as having a low output voltage ripple. They are also simple to implement and offer low-cost solutions for regulation.

4.8.3 Switching Regulators

Switching regulators, also known as switching-mode- power supplies (SMPS), are electronic circuits that convert a DC voltage level to another DC voltage level by rapidly switching the circuit on and off. Like linear regulators, the switching operation is performed by a series-pass transistor (Switch Element) controlled by a negative-feedback Control Circuit. The purpose of switching is to temporarily store energy in an energy storage device and then release that energy at a different voltage level. Figure 21 shows a typical switching regulator circuit.

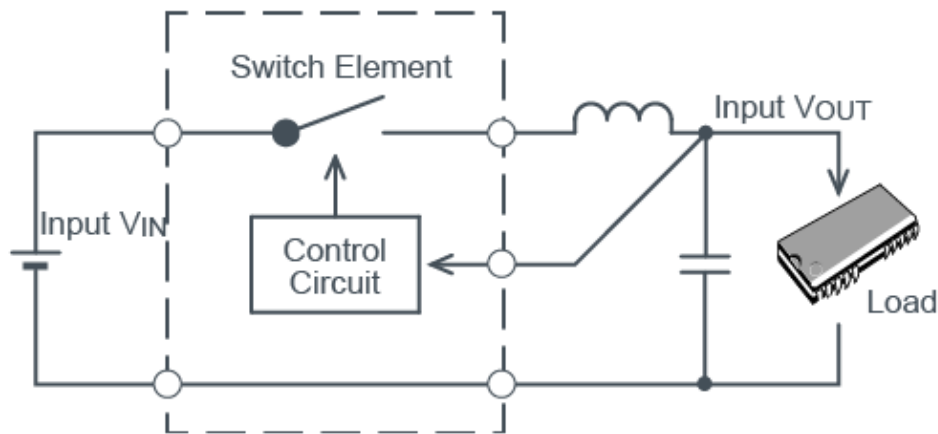


Figure 21: A Typical Switching Regulator Circuit

Switching regulators typically consist of a series-pass transistor as the control element, an error operational amplifier as part of the control circuit, an inductor or a capacitor as an energy storage device, and a pulse-width-modulations controller (PWM) to control the switching operation. Like linear regulators, the error operational amplifiers compare the difference between the output voltage and input voltage to switch the transistor on and off. The PWM controller uses the operational amplifier output to control the duty cycle of the switching waveform. In typical constructions of switching regulators, additional inductors and capacitors are placed at the output to filter and smooth the output voltage. There are many types of switching regulators, but the one relevant in our discussion is the Buck Converters. Buck Converters are step-down switching regulators that convert a higher input voltage to a lower-level voltage. The major advantage of switching regulators is their exceptional efficiency in the range of 70%-99%.

4.8.4 Regulators Comparison

The main differences between the two types of regulators are their efficiency and noise output. Switch regulators implement higher efficiency compared to linear regulators. However, that is only the case when there is a substantial difference between the input and output voltages. A disadvantage of switch regulators is their nature to generate output ripple and noise that might influence any surrounding sensitive components. Moreover, linear regulators require some additional heat dissipating in the form of heat sinks, and thus, any design that includes linear regulators must consider this as a design choice. Linear regulators are also simpler to implement and require lower component count and cost. Table 18 outlines these differences.

Table 18: Linear vs. Switch Regulators Comparison		
	Linear Regulators	Switch Regulators
Efficiency	Low the higher the difference between input and output voltage	High
Cost & Complexity	Cheap and Simple	More Costly and Complex
Noise Output	Small	Noticeable by sensitive components
Working Conditions	Minimum dropout voltage	No minimum dropout voltage

To select the technology of choice, we must first consider the key requirement in the power system, that is the ability for the power system to deliver 5V from a 12V line. The regulator of choice should be able to accommodate this requirement with a cost-effective solution. In addition, the regulator should be able to handle a conversion of a large amount of current to accommodate the stepper motor requirements. With all of this in mind, a switch regulator, specifically a buck converter, is the right choice for the project. The voltage drop-off from 12v to 5v makes switch regulators more favorable to avoid low efficiencies. The fact that buck converters generate noise is not a significant concern since the project does not deal with any noise-sensitive electronics.

4.8.5 Part Selection: 110-AC to 12-DC

The requirement to selecting an AC to DC power adapter are straightforward; the output voltage should be 12V. The current rating, however, is variable. The adapter should be able to provide at least 5A of current (refer to section 4.8.1 for more). Furthermore, it should also be FCC certified to ensure that it meets the necessary safety standards.

A possible product of choice is [ALITOVE Power Supply Adapter \(12V 10A\)](#), it matches all the required specs and is very easy to integrate to a Printed-Circuit-Board (PCB) design with an appropriate barrel jack adapter. On the current implementation of the project, the team plans to use this power adapter as the product of choice. However, if the teams run into any power related issues due great power draw from the motors and/or problems in PCB integration due to high amount of current running through the circuit, the team has a second backup option with a multi-output power supply unit by [GESD](#) with the same 12V voltage and 10A current ratings. Both supply choices are shown in Figure 22.



Figure 22: AC to DC Power Supply Choices

4.8.6 Part Selection: 12-DC to 5-DC Regulation

Now that the power supply is selected, a step-down voltage regulator is to be selected next to accommodate for the lower voltage components. To select the right buck converter, the team considered products and components that are easy to prototype with and easy duplicate to a PCB design. An additional consideration is the inclusion of two low current rated buck converts, instead of one 8A rated component. This choice was made to limit the complexity of the buck converts, as well as to separate motors from other electronics to ensure ease of prototyping, ensure the overall performance of the system under heavy loads, and accommodate for any future expansion of the project. For testing, the team has opted for [Buck Converter Module 5V 5A by UTRONICS](#), as shown in Figure 23. It is well suited for testing as it comes with a barrel jack connection that works with the AC-to-DC power supply the team has chosen. It is relatively simple and easy to integrate into future PCB design.

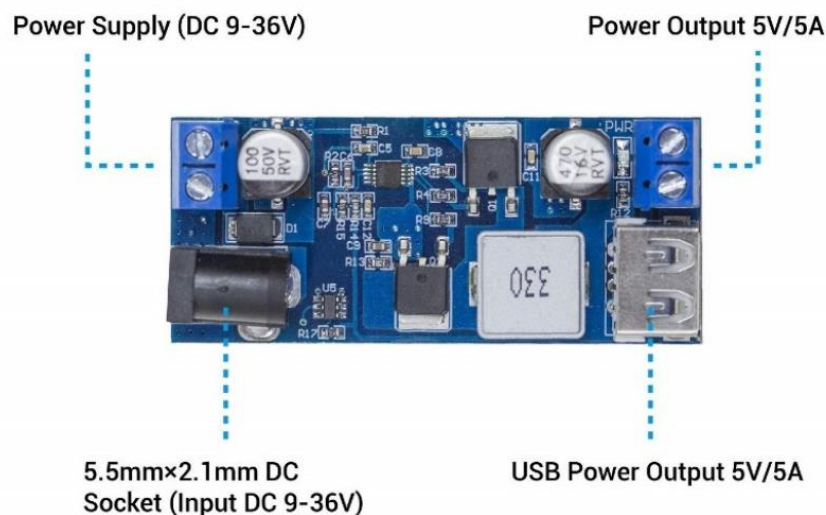


Figure 23: Buck Converter Module

4.9 3D Reconstruction Software

Selecting the right photogrammetry software for a project requires careful consideration and investigation. With numerous SfM software options available, it is essential to identify the factors that will narrow down the selection process for the project. Performance, hardware acceleration, features, cost, and community support are crucial factors to consider. Based on these factors, three potential software options for the project are Meshroom, RealityCapture, and 3DF Zephyr. By considering each software's strengths and weaknesses, the optimal choice can be made to achieve the best results for the project.

4.9.1 Alicevison Meshroom

Meshroom is an open-source photogrammetry software developed by ALICEVISION, a photogrammetric computer vision framework. It is completely free and available on both Windows x64 and Linux operating systems. Meshroom is node-based software, meaning that the photogrammetry pipeline is structured as a series of nodes, each representing a different process, such as feature extraction, image matching, and texturing. All setting related to a process is confined within its node and each node can run separately provided that a set of required inputs have been previously computed. While node-based software usually come with a learning curve, this approach makes it flexible to rearrange and connect nodes in different ways, allowing for customized workflow by the end-user. In addition, nodes permit for the possibility of automation integration into the pipeline. Meshroom offers GPU acceleration, since it is designed to work with NVIDIA GPUs that support CUDA technology with a minimum of 8GB of RAM (Random Access Memory) and support for CUDA 10.0 or later.

Getting into details, Meshroom offer friendly graphical user-interface; all nodes and process are documented on their purpose and input parameters. And although the graphical interface can be improved, it is a very well documented interface for an open-source software. By default, Meshroom offers a default pipeline that works very well for all applications, freeing the end-user from the need to understand the detailed operation of photogrammetry. If required, additional nodes can be integrated into the pipeline to produce a desired output. Meshroom presents the ability for mesh cleanup and filtering, such as decimate or filter nodes, to reduce the number of faces and points in the final mesh. The use of these nodes is recommended because the default pipeline produces an obscene number of faces compared to other software discussed later. The speed of the pipeline is acceptable if GPU acceleration is enabled, and there are some optimizations that can be integrated to increase the operation speed. The software is also compatible with a variety of input formats, including JPEG, PNG, TIFF, and RAW files. A tremendously important feature of Meshroom is its support of CLI (Command-Line-Interface) commands, allowing users to run the software from the command line interface without having the need for a graphical interface, which is a key feature for automation. With CLI, users can set the input and output paths, select parameters, and execute nodes. However, compared to other software, Meshroom CLI commands are not as sophisticated and comprehensive. Overall, Meshroom provides a complete set of photogrammetry features and produces high-quality meshes, but its utmost notable feature is that it is entirely free, making it a community favorite as a powerful photogrammetry software without the financial investment typically required for commercial applications.

4.9.2 RealityCapture

RealityCapture is a sophisticated and comprehensive commercial photogrammetry software that is developed by CaptureReality. Its commercial nature allows it to offer advanced features and powerful tools for photogrammetry through continuous development, support, and updates. The software offers traditional workflow process where processes are categorized to tabs, each containing a related set of tools. RealityCapture requires NVIDIA GPU that support CUDA for full operation. Without an NVIDIA graphics card, the software cannot create the final 3D mesh. Additionally, the software is currently only available for windows-based machines with at least 8GB of RAM.

The major advantages of RealityCapture are its advanced algorithm and processing optimization, allowing it to create highly detailed and complex 3D mesh with minimum processing power. From its incredibly friendly user-interface, real-time graphical interface, and advanced set of features, RealityCapture is a community favorite for those who can tolerate spending the hefty price of \$3,750 USD to obtain a license agreement. Fortunately, a recent patented credit-based license has made its way to the software, making it a viable option for small-business and enthusiasts who want to take advantage of the software. RealityCapture calls this license PPI (Pay-Per-Input) that allows the user to access the full range of features offered by the software, with the exception that export features are locked until the user pays for PPI credits. The credits are based on the pixel size, meaning that for a single 12.8 Megapixel image the average cost is \$0.01 USD, making it a cost-effective option for most consumers. Moreover, the recorded processing speed is top of line when compared to other commercial software. Most importantly, it offers an incredible list of CLI commands that allows automation and integration in any desire workflow. There is a good amount of online documentation of these commands, as well as basic scripts that are designed for automation. All in all, RealityCapture is a solid choice for any photogrammetry project, with its only downside being price even if it is reasonably valued.

4.9.3 3DF Zephyr

3DF Zephyr is another very popular commercial photogrammetry software developed by 3DFLOW. Like RealityCapture, it offers an advanced set of features and tools that are continuously updated and supported given its commercial nature. It supports all common input formats, including those coming from a DSLR, a smart phone, or a simple sensor, as well as supporting all kind of 3D output formats, such as OBJ, STL, and FBX. What makes 3DF Zephyr shine is its ease of use, and friendly user-interface, allowing it to be the perfect starting point for those who are little inexperienced with photogrammetry. It features a modern graphical interface, with all tools and sections separated into tabs, making it very intuitive to use and learn. In terms of performance, Zephyr is known for its speed and quality as it supported advanced and optimized algorithms designed to run on modern hardware. It supports NVIDIA GPU acceleration running CUDA, however, it is not required for full software operation. Moreover, it does require 18GB of RAM as a minimum spec and it is currently only available on Windows x64 machines.

Most importantly, 3DF Zephyr supports Command-Line-Interface (CLI) allowing it to be integrated in an automation and batch processing, adding flexibility to the workflow. The downside of the software is its \$250.0 dollar price tag per month. However, the software does offer a Lite version with a lower price tag for a reduced set of features, like export options and basic editing tools. The Lite version is priced at \$200.0 dollars lifetime, making it more reasonable than its full

version. Furthermore, the software also offers a free version with even more reduced feature sets compared to the Lite version. Its biggest disadvantage is a 50-image limit input, making it non-ideal for complex objects. All in all, 3DF Zephyr is an excellent software of choice if price is taken out of the equation.

4.9.4 Software Comparison & Selection

Comparison between software requires us to prioritize a list of the most relevant features to our project. Namely we prioritize output quality as it is a critical component to the success of the project; an automated process needs to be at least on par with the manual process. Moreover, because this is a cost-effective implementation of commercially expensive photogrammetry rigs, we prioritize cost after quality. Finally, the automated nature of the project requires that the software provide CLI support. Table 19 outlines the most relevant features of the software ranked from a scale of 1-to-5 (credit goes to formslab.com for the research and ranking in the table).

Table 19: Vertical Motion Slider Components			
	RealityCapture	Meshroom	3DF Zephyr
Quality	★★★★★	★★★★☆	★★★★☆
Speed	★★★★★	★★★★☆	★★★★☆
Features	★★★★☆	★★★★☆	★★★★☆
User-friendliness	★★★★★	★★★★☆	★★★★☆
Cost	PPI License (Credit-Based)	Free	Free (max. 50 imgs) Lite-Version (\$199)
CLI support	★★★★★	★★★★☆	★★★★☆

If we take cost out of the equation, RealityCapture is the clear and obvious winner without any further consideration as it ranks 5 out of 5 in almost every category. 3DF Zephyr is also a good candidate, with its ranking on the same level as RealityCapture. However, cost is important, and for that reason 3DF Zephyr is a terrible option for its expensive license and very limited free version. On the other hand, cost makes Meshroom a viable candidate, and despite its lower ranking in other categories, it makes up for that in the cost category. In addition, since the project is automated, the user is less likely to spend time navigating the user-interface and exploring additional features. Speed is also irrelevant since the photogrammetry process is known to be slow by nature, and therefore, it is almost always dismissed in marketing campaigns.

With all this in mind, the team has the choice between RealityCapture and Meshroom. The team has decided to implement the project with Meshroom as the software of choice for its free nature and excellent set of tools and quality. RealityCapture will be a backup option in case the team discovers any problems with Meshroom during prototyping and integration testing phases.

4.10 Parts Selection Overview

Table 20 summarizes the parts that we've selected for our project and gives an overview of the cost and supplier for each component. Some components are Team-Supplied, meaning that a team member already has the part and will provide it for the group without costing anything else.

Table 20: Part Selection Overview			
Description	Component	Supplier	Cost
Mechanical Hardware			
Linear Actuator	2040 Aluminum Profile (100cm)	Open Builds	\$15.29
Carriage Platform	V-Gantry (Carriage Platform)	Walfront (Amazon)	\$14.14
Belts	G2 Timing-Belts & Pulleys	Zeelo on (Amazon)	\$15.99
Camera Mount	Ball-Head Camera Mount	CAVIX (Amazon)	\$10.18
Turntable Structure	2020 Aluminum Profile (20cm)	Open Builds	\$6.58
Spinning Mechanism	Lazy Susan Bearing	DMSTECH (Amazon)	\$4.35
Electronics Hardware			
MCU (Driving)	Raspberry Pi 3 Model B	<i>Team-Supplied</i>	\$0.00
MCU (Driven)	Arduino Mega (ATmega2560)	<i>Team-Supplied</i>	\$0.00
USB Interface Chip	ATmega8U2	Digi-Key	\$3.54
DSLR Camera	Canon T7i	Facebook Marketplace	\$200
3 Stepper Motors	Nema17 (45Ncm)	StepperOnline	\$12.99
3 Motor Driver Modules	A4988	HiLetgo	\$10.19
Homing Sensor	KY-003 - A3144	MUZHI	\$1.16
Touch Screen	HXD8357D	Adafruit	\$39.95
Reconstruction Software	Meshroom	AliceVision	\$0.00
Power Supply	Power Brick (12V 10A)	ALITOVE	\$20.00
Voltage Regulator	Buck Converter (5V 5A)	UTRONICS	\$14.99

5 Related Standards and Realistic Design Constraints

Every engineering project necessarily has related constraints which limit the scope of the project. However, these constraints are not necessarily a negative. Oftentimes standards help to reduce redundant work and increase the compatibility between independently developed machines. Thus, the following sections seek to discover the outside constraints and standards which affect this project. Furthermore, the section explores how these findings will both limit and assist the project and its goals.

5.1 Related Standards

The following sections include public standards which directly impact or influence the design of this project. One of the benefits of leveraging standards is that it reduces the amount of time spent working on things that have already been done. Additionally, they increase the level of capability of a project by ensuring that others know how to use it. Some of the more important standards that this project utilizes are IEEE 802.11n (Wi-Fi standard), USB 2.0, and ISO/IEC 9899:2018 (C language standard).

5.1.1 IEEE 802.11n

In 2009, IEEE released the new standard for Wi-Fi connections: this standard, called 802.11n, defined the new version of Wi-Fi. It runs on both 2.4Ghz and 5Ghz connections and, by using multiple antennas, it can reach maximum connection speeds of up to 600Mbps. Another common name for the standard is Wi-Fi 4. This name helps to quickly differentiate 802.11n from 802.11ac and 802.11ax which are known as Wi-Fi 5 and Wi-Fi 6 respectively [21].

A key improvement of this standard over its predecessors is its ability to maintain multiple connections at the same time. This allows for increased throughput by leveraging the power of parallelism. It also enhances the stability of wireless connections, an incredibly important aspect of this project [21].

The Raspberry Pi 3 Model B uses the BCM43438 wireless LAN and Bluetooth chip [17], which implements 802.11n. This means that the project is constrained by the limitations of this standard. More specifically, one of the main operations which will be involved in this project is the transfer of images from the photogrammetry station to the machine responsible for running Meshroom. In this operation, the time it takes to transfer the images from the Pi to the workstation will be capped by the maximum throughput of Wi-Fi 4. Fortunately, this is not really a problem as Wi-Fi 4 is relatively fast despite its age.

In a hypothetical situation, assume that a photogrammetry operation is occurring. The camera might take 100 total files with each file having an average size of 10MB (in reality, the files are smaller). This results in 1GB of data. Now assume that Wi-Fi 4 runs at a more reasonable speed of 300Mbps. That translates to 37.5MBps. Consequently, it would take just over 27 seconds to transfer all the photos to the photogrammetry machine. This sounds like a long time but compared to the 20 to 30 minutes it takes to create the 3D image, this is not that much time.

IEEE 802.11n is only responsible for describing how layer 1 and layer 2 of the network stack operate. It is not responsible for any of the higher layers that run on top. This includes the transport (i.e., TCP or UDP) layer and the application layer (the actual data being sent across). However,

since Wi-Fi 4 is limited in range, reliability, and speed, it still plays an indirect role in these upper layers.

5.1.2 USB 2.0

USB is short for Universal Standard Bus. It is a standard port used on many different devices for a variety of purposes. The second version of this standard was defined in 2000 by a large conglomeration of companies including Microsoft, Intel, and HP to name a few. It was proposed as an evolution of USB 1.0 and 1.1. Consequently, it is fully backwards compatible with these earlier standards, albeit at their slower speeds. While the original version of USB was only able to transfer data at a measly rate of 12Mbps, USB 2.0 can handle speeds upwards of 480Mbps. This is a huge increase. USB 2.0 was commonly referred to Hi-Speed USB to differentiate its devices from their slower counterparts [22].

USB uses a tiered star topology to enable the connection of multiple devices to a single USB hub. In the setup, there is a single host device which is responsible for hosting the connection. Every other device that is connected communicates with this host device. Connected to the host device are hubs. These hubs are assigning specific addresses to each of the devices that are connected. It is possible to place additional hubs (up to 5 layers) on the main hub to increase the number of devices that can be connected. Of course, this will also eat into the bandwidth available, but that is not usually a problem. The resulting structure is a tree where the root is the host device, the branches are formed by the hubs, and the leaves are the connected devices [22].

One of the main advantages to USB is its ability to hot swap devices. This means that peripherals can be added and removed without damaging themselves, disrupting their neighbors, or disrupting the host. This is a huge advantage over alternative connections like PCI and SCI. It reduces the number of points of failure that the system can experience [22].

This project plans to use a USB 2.0 port to communicate with the camera. In this case, the Raspberry Pi will be the host machine, and the camera will be one of its peripherals. Since there will only be a single peripheral being added to the host machine, the limit of five levels will not affect the project. However, the maximum transfer rate will slightly impact the speed at which images can be captured from the camera since the amount of time to transfer an image will be limited by this speed.

5.1.3 ISO/IEC 9899:2018

In 2018, the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) published ISO/IEC 9899:2018, the newest standard for the C programming language. This standard offered technical corrections and improvements but no major changes from the old ISO/IEC 9899:2011 standard. It is commonly known as C17 or C18 to differentiate it from previous versions. C17 defines the syntax of the C language, including the important terms and symbols, as well as the function of libraries and environments that translate and execute programs. [23]

The C standard is incredibly important, as it rigorously defines the rules for the language and how it is to be written and interpreted. This allows the language to be incredibly common and portable, which enables us to use the language for the bulk of the code in our project across various devices.

Both of our MCUs have development environments that allow us to easily code in C, which will streamline the software writing process.

The C language is a very commonly used language that is supported by many development environments, as C compilers exist for most operating systems and architectures. It offers low-level memory access and minimizes runtime operations, which means the code is lightweight and well suited for microcontrollers with limited program memory while allowing control over memory values through pointers and memory allocation by the user.

This standard affects our design by defining what we can achieve with the C language, both by limiting the data structures and operations we can perform and allowing us control at the low-level to reduce overhead and program space. The C language being a long-defined standard means that it is very common, and thus all team members are familiar with the language and our hardware can run programs written in C and compiled for the specific architecture. It will allow us to effectively manage space in memory as well as prototype code on development boards that may not even be used in the final project.

5.1.4 NEMA 5-15-P

The National Electrical Manufacturers Association (NEMA) defines standards for electrical plugs and outlets, with NEMA 5-15-P defining the standard for the North American 3-pronged grounded electrical plug that plugs into the standard wall outlet. This standard defines the dimensions of IEC Type B electrical plugs as well as the voltage and current the plug is designed to handle. A United States wall outlet provides 120V of AC power and is designed to connect to the standard NEMA 5-15 plugs. This standard is very important because many appliances and consumer electronics utilize electrical power from wall outlets, and if each device had a different plug, it would require converters or not guarantee safe connections to the wall outlets. [24]

The NEMA 5-15 plug relates to our project as our device will plug into a standard US wall outlet for power, and thus the plug will need to comply with the NEMA standard in order to safely connect. Our power supply will connect to the wall outlet, so we will purchase a power supply with a NEMA 5-15 plug that will convert the higher wall voltage to the lower voltage that is needed to power the station. This standard guides our design process by making it easier to utilize wall power, as a standardized connection is already defined and mass-produced that we can use.

5.1.5 IEC 60130-10

In 1971, IEC published standard IEC 60130-10, which specifies a variety of information about coaxial connectors designed for coupling an external power supply to lower voltage equipment. This standard describes the various types of connectors, including their dimensions, as well as their connections, and voltage and current limits. These connectors come in a variety of sizes, and they require standardization as they are a common plug for connecting power supplies, and they have an inner and outer diameter, both of which can vary. [25]

The IEC 60130-10 connector relates to our project as we will use the plug and socket to connect our power supply to our PCB. Using a standardized plug once again makes the process easier for connecting all the various components of our design and will aid in finding parts that can be found on development boards for testing and attached to our printed board for the final product. The connector we plan to use is a Type A connector, with a 5.5 mm outer diameter and 2.1 mm inner

diameter. This standard guides our design process by making it easier to connect a pre-purchased power supply to our custom board without worrying about connecting the circuits internally.

5.1.6 NEMA-17

The National Electrical Manufacturers Association (NEMA) specifies the mounting hole pattern, shaft diameter, and overall dimensions of the motor. In particular, it specifies that the motor has a faceplate of 1.7 inches (43.18 mm) square, with four screw holes spaced 1.2 inches (30.48 mm) apart. [26]

With these dimensional standards defined, we can purchase mounting hardware with confidence that it will fit our motor selections contained in our design. The shaft diameter is also important because without an appropriate coupler, we will not be able to use the shaft to apply any torque. This standard gives us confines with which to search for a coupler.

5.2 Realistic Design Constraints

The following constraints portray the ways in which external circumstances influence this project. There are economic constraints, time constraints, environmental constraints, social constraints, ethical constraints, political constraints, legal constraints, health and safety constraints, manufacturability constraints and sustainability constraints. These all have direct and indirect effects on the decisions that have been made in the previous sections and the designs that are composed in the following sections.

5.2.1 Economic Constraints

The economic constraints of completing our project mainly lie in purchasing the materials needed to construct the device. The 3D reconstruction software Meshroom we plan to use is open-source, and thus does not require us to purchase a license of any kind. The other software that will be run on our microcontrollers will be open-source or written by us, and thus does not introduce any cost constraints. Our project is not sponsored, and thus does not have outside funding from a company or the university. All funds will be provided by the members of the team, and the cost will be split fairly among the group. As the team consists of four college students, we do not have access to a large amount of economic resources, and so cost minimization will be an important part of the process. However, this project is incredibly important, and funding must be sufficient to ensure a successful project, so we will invest what is necessary yet reasonable into the project.

The hardware that is included in the final device is not the only hardware that we need to purchase. We will need to perform research into what technologies work best for our project, which means that we may need to purchase components that will not end up in the final design. We may not realize that a motor or railing does not work with our project until we test it within a prototype design. The prototyping process may also require designing multiple iterations of our PCB and having them made before we achieve the final design that will be used, which costs money to have printed and shipped.

The elements of our project that require money are the hardware components that make up the photogrammetry stand. The most expensive piece of hardware is likely the camera. Our project involves the use of a DSLR camera, which will be necessary for the demonstration of the project and will need to be purchased to be implemented into the design. Another piece of expensive

hardware is the two processors that we chose to use, as we will utilize both an ATmega2560 and Raspberry Pi 3. We will need to purchase the development boards for both microcontrollers as well as a custom PCB that will likely include the ATmega2560 processor embedded in it. As we do not plan on removing the CPU from the Arduino® Mega 2560 Rev3 development board, we will need to purchase the processor individually to be added to our custom board.

Our project is designed to be viable as a commercial product, and thus there are not only monetary constraints in the research and development of the project, but the device must be able to be constructed for a reasonable price for consumers to be able to purchase it. Our project is designed as an affordable option for streamlining the photogrammetry process for people who are already working with the technique. The hypothetical retail version of our project would not include the DSLR camera, as it is expected to be provided by the customer who would already own a camera and be using it manually. This means that one of the most expensive elements of our project would not inflate the price of a retail version, and we only need to purchase the camera for our demonstration.

5.2.2 Time Constraints

The time constraints of our project are determined by the deadlines as set by the instructors. There are various milestones that we must complete for the class, which is required to graduate and impacts our members' GPAs. Parts of our report have deadlines to keep us on track for the project, which will need to be designed by the end of Senior Design 1 in April and finished and demonstrated at the end of Senior Design 2 in July. These are strict deadlines, which means we need to effectively budget our time to have the project completed to demonstrate it and leave time to deal with issues that arise in the design and construction process.

One time constraint that we face is that we need to wait for parts that we order to arrive. The parts that we order for the stand and the interface will take time to be shipped and delivered. This is a major constraint because we do not control the amount of time this takes, so we must schedule the ordering of parts so that they arrive with enough time to test and assemble. We also may receive parts and only after we attempt to implement them realize that they are insufficient or not functional. If motors arrive that are broken, we need to have time to return or repair them, and if parts arrive that do not meet our requirements, we need to research other parts that can replace them in the design.

Another time constraint is our custom PCB, which will need to not only be designed by us but also printed and delivered. The design process of the board will take time that we need to account for, as it may take a few iterations to achieve a design with which we are happy. We may need to prototype boards that we have printed and shipped, which means that we will have to wait in between designs being completed and receiving the physical boards. There may be shortages of parts or delays in the shipping, so it is important that we begin the process early enough that we have enough time to test the boards, improve their design, and implement them in the project.

Aside from the design and construction of the project, there is also the time constraint for demonstrating the project and operation of the photogrammetry stand by the user. The stand will take pictures of an object from a variety of angles using a single DSLR camera, which means that motors must move the camera to different angles and rotate the object. The motors will need to operate at a moderate speed, as moving too quickly could cause unnecessary strain on the

mechanical parts due to the weight of the camera or disturb the position of the object that is being captured which could ruin the images. However, the motors cannot operate too slowly, or the process will take too long, which would defeat the purpose of the device as it would be quicker to take the pictures manually. In addition, our project only involves taking pictures of objects in an automated way and feeding them into an algorithm to generate the 3D model that was designed by another party. Therefore, we need to consider the time that the software takes to generate the model when we demonstrate the project, which may take some time to run. We will solve this by pre-preparing a 3D model of an object using our device to take the pictures and the SfM software to have a final model to show. We will then demonstrate the process of image capturing and show what is produced by the stand itself and compare our images during the demonstration to the images we used to generate the model to show that if we were to allow the newly taken pictures to finish being processed, we would get the desired model.

5.2.3 Environmental Constraints

The environmental constraints of our project are mainly related to power consumption, as we want to minimize electricity use as much as possible. Wasting a lot of power would not only drive up the cost of using the device, but also contributes to harming the environment through using more electricity that is likely gained through fossil fuel consumption than necessary. We aim to use only the necessary amount of electricity to power the device.

Another relevant constraint is electronic waste, which refers to old discarded electronic devices and components, and is a growing problem for consumer electronics. For our project, we want to minimize the number of unnecessary components that will end up as waste. This includes components within the finished product as well as components that we may use in the prototyping process that we get rid of at the end of the project. It is important to focus on repairability of the device as opposed to building it with replacement of the entire device in mind. Replacing parts instead of replacing an entire device is a way to reduce waste. Another reason to reduce the number of parts that we use or get rid of is that the silicon and metals in the devices are mined from the earth, which damages the environment. Not wasting parts allows the parts already produced to be fully used.

5.2.4 Social Constraints

The social constraints of our project mainly guide the design of the user interface, as we will consider the way people typically interact with consumer electronic devices. The user experience is a very important part of the project, as we want to ensure a positive experience with the photogrammetry station. One way we can achieve this is to automate as much of the process as possible, since minimizing unnecessary manual input is convenient for the user. We can also improve user experience with an intuitive interface, which is why we plan to use a touch screen for selecting options for the picture taking process. Touch screen interfaces are incredibly common on consumer electronics, so we will utilize that familiarity to improve user experience.

5.2.5 Ethical Constraints

Our project does not pose any major ethical issues, and thus there are minimal ethical constraints for our project. One common concern in consumer electronics is privacy and security of personal data. Our device will not require an account to operate and will not ask for any personal

information. The 3D reconstruction software Meshroom does not require an account either, so the entire process will not gather any personal data from the user. Our project poses no ethical concerns regarding user data as none is collected.

Another aspect related to ethics is ensuring our device is safe to use, which is elaborated on in Health & Safety Constraints. It is the team's responsibility to ensure that our device is safe to use, which means providing instructions to the user on how to use the station properly, as well as concealing any dangerous elements such as wires.

5.2.6 Political Constraints

Our project does not pose any political issues as the project is a basic consumer electronic device that does not gather user data. It streamlines a process that people are already performing without much issue, and no major policies specifically limit photogrammetry. If policies which limited either the use of photogrammetry techniques or open-source software were enacted, then there could be potential problems. However, as previously stated, there are no such policies.

5.2.7 Legal Constraints

There does not appear to be any legal constraints placed on photogrammetry itself as there might be for rocketry or weaponry. However, there are still some legal areas that impact the development of this project. One of the major legal constraints that affects this project is licensing. This project plans to make extensive use of opensource software, the use of which is governed by copyright laws and reuse licenses. There are two main licenses which are represented by the major software components that will be deployed in the final product: the GNU General Public License and the Mozilla Public License version 2.

5.2.7.1 GNU General Public License – gPhoto2 & Raspberry OS

The first of the licenses that is utilized by two of the prominent programs being deployed, namely Raspberry OS and gPhoto2, are published under the GNU GPL license. This license is a copy-left license which ensures that the user of the software maintains the freedom to do what they like with that software. In other words, this license ensures the freedom of software which is distributed under this license.

The term freedom does not refer to the monetary cost of the software. Distributors and developers are allowed to charge for their free and opensource software under GPL. In fact, this is almost encouraged: the license is trying to block anyone from placing restrictions on the software. In other words, GPL ensures that software is distributed as (or with its) source code such that the end-user (the consumer) can see, modify, and redistribute the source code as they see fit. The main caveat to this license stems from it ensuring that future products which implement GPL software are also published under the GPL license. This essentially prevents the use of GPL licensed software in proprietary solutions since these software packages would have to be published with the corresponding proprietary software under GPL [27]. This would consequently make these solutions opensource.

As mentioned before, there are two main libraries in this project which are published under the GNU GPL license. As such, there is a legal obligation to publish this software under the same

license and make it publicly available. If this is not done, then the project will not be in compliance with the copyright regulations of the software packages it relies on.

For some GPL licenses are viewed as a major downfall when considering the use of opensource software. For the purposes of this project, however, the software being published under the GPL is a huge positive. The amount of time that would be lost generating internal hardware and software capable of directly interfacing with the Raspberry Pi, DSLR, motors, etc. would result in enormous overhead; this project would not be doable in the two-semester timeframe. However, since these software applications are freely available, this project can be put together quickly. So rather than opposing this license, the project will embrace it to ensure that everything goes as planned.

Additionally, since there is no restriction on how the software is used, this project is not constrained external parties and their interests. It is often the case that companies that provide software services have limits on the software can be used for. For instance, it might not be allowed to use the software for commercial purposes. Or it might cost more to use the software for commercial purposes. With opensource software under GPL, there is no time need be spent on worrying whether or not the software is in compliance with the regulations and interests of external parties. The only thing which needs to be considered is whether or not the source code is being made freely available.

As such, the team will release the software developed for this project under the GNU General Public License. This will ensure that anyone who looks at or uses this project will also be free to access all the software that was developed for it. Using this new software in conjunction with the other software published under GPL, any user should be able to recreate this project and implement their own changes freely.

5.2.7.2 MPLv2 – Meshroom

As mentioned previously, there is another license that is used in this project: the Mozilla Public License version 2. It is very similar to the GPL license in the previous section; however, there is one major difference between the two. MPLv2 does not require that new software which utilizes MPLv2 licensed opensource software also be released under MPLv2. Rather, only the MPLv2 software and the modifications made to it need to be released to the end user [28].

For example, imagine a situation in which Microsoft Word integrates a modified version of software which had been distributed under the MPLv2 license. When a user purchased a copy of Microsoft Word, they would also need to receive the source code for the MPLv2 software which had been integrated as well as the modifications made to it. These modifications could be received as the modified source code, or as a detailed list of all the exact changes made.

This is distinct from what would happen in the same situation had the integrated software been released under the GPL license. In this case, the entirety of Microsoft Word would need to become an opensource GNU GPL product. So, MPLv2 seeks to enable the free use of opensource software even in proprietary systems. It maintains the proprietary nature of the proprietary software whilst also preventing the unchecked use of the opensource software.

Since the GPL license is more restrictive (of course this depends on the perspective), this project will simply release all the software under the GPL license. This will ensure that the project is not held back by regulations which would otherwise prevent it from being released. In the case of this

project, it does not enhance the result by using MPLv2 rather than GPL because the most important part of the product is the hardware and integration, not the software.

5.2.7.3 A Quick Disclaimer About Licenses

Since this product is not actually going to hit the market and since there is not an actual need to publish this in a public setting, these licenses do not hold much weight. In the real world, they would keep this project from being released as a closed source product, but since this is not being released in the real world, this is not so much a concern. However, since this whole project is meant to simulate a real-world design product, there is no reason not to follow these regulations and release the software accordingly.

5.2.7.4 Areas Where This Project Avoids Legal Constraints

There are some areas where a 3D reconstruction system could run into some problems. Firstly, if drones were being used to capture landscapes for 3D reconstruction, this would result in a significant number of legal concerns. This is partially because there are now many regulations on drones. But it is also due to the fact that remote flying devices like drones could photograph sensitive areas where a human would not be able to reach. Our project is self-contained, and therefore it does not run into these issues.

Additionally, there could be concerns about what is scanned by the photogrammetry system. If proprietary, secret, or copyrighted objects are scanned, then this could infringe on some laws. However, cameras also suffer from this issue. Every camera is capable of capturing and rendering illegal images. But this is not a concern for the manufacturers of the cameras because disclaimers are included which state how the product should and should not be used. Thus, the manufacturer is not held responsible for misuse.

Consequently, this project should include instructions which clearly demonstrate how the product should and should not be used. These instructions could be written, but ideally, they will include pictures demonstrating proper use of the product. There should also be a disclaimer which states that if the product is not used for its intended purpose, then the manufacturer will not be held responsible.

5.2.8 Health & Safety Constraints

The health and safety constraints of our project relate to the safety of the team while we manufacture the photogrammetry stand as well as the safety of the user when operating it. We have a mechanical stand to construct using aluminum rails and 3D printed parts that connect them. Our stand will not require welding, machining, or other dangerous processes to construct, which will make building the stand safer for us. Welding introduces dangerous levels of heat and light, and so needs to be done by trained individuals, and machining introduced metal shavings and particles that can be hazardous to one's health if inhaled. Avoiding these types of construction protects the health and safety of the team. It is possible, however, that we will need to solder components in the prototyping process or for the final design. Team members are more experienced with soldering which makes the process less dangerous, but we still need to be careful and use proper safety protocol when soldering circuit elements onto boards. Much of the prototyping is likely to be done using development boards and breadboards to reduce the need to solder, but it still may come up.

Another constraint that is important is that we are working with high levels of voltage and current. We have motors, microcontrollers, and sensors that require certain voltage and current levels, which means that we will have to be cautious when making electrical connections. The motors will altogether need relatively high levels of current, which means we need to be mindful when handling the electrical components. Our device is also powered using a wall outlet, which is high voltage and can be dangerous if not careful. We will need to carefully select components to design a power system that will regulate the power to the appropriate levels to power our device.

In relation to the safety of the user, we need to design the device so that any high voltage or current wires are shielded and cannot accidentally shock the user or start a fire. It is also important to prevent fingers getting caught or crushed in the moving rails, which will be addressed by recommending only adults operate the machine and reminding the user to be mindful when the motors are active.

5.2.9 Manufacturability Constraints

The manufacturability constraints of our project are mainly related to the availability of materials and parts that are necessary to construct the device. Due to the issues in the supply chain that have been occurring for the past few years, it may be difficult to always obtain our first choice of parts. Some parts are not being produced in a high enough quantity for us to acquire the small amount needed, and some parts are on back-order, which would take too long to arrive. The availability of parts was an issue in Junior Design and severely limited our ability to complete projects. Some parts that we need have alternatives, such as motors or common circuit elements, but other part selections are more restrictive such as microcontrollers. We rely on other manufacturers to produce the parts that we need for our design.

Another constraint is the assembly of the parts together, which will need to be done by the team. This involves the electrical connections that need to be made to wire together the microcontrollers, power supply, sensors, camera, and motors. It is possible we will need to solder components ourselves which requires the knowledge to solder safely and effectively. We will also need to be able to assemble the mechanical parts of the design, as we will utilize motors, gears, belts, and linear actuators to allow the stand to move the camera and object. The design needs to be achievable to construct by ourselves with the basic tools to which we have access. In a retail version of our design, the device could be manufacturable by machine, but for our project that needs to be demonstrated for Senior Design we will need to produce only a single functional unit.

5.2.10 Sustainability Constraints

When it comes to sustainability, there are a lot of aspects that come into play. Many companies tout their products as green when in reality this is not the case. Take for instance the latest iPhones. Apple states that the reason for not including chargers in the box is due to environmental (sustainability) concerns. They allegedly do not want to continue to create additional chargers for new products, but there is an issue with this: they simultaneously switched to new cables which do not work with all of the old charger blocks they had been previously providing.

The point of this discussion is not to trash other companies for hypocritical practices. Rather, the point of the previous illustration is to point out how easy it is to claim that something is sustainable

when, in reality, it is not. As such this project needs to be clear and realistic in how it views the topic of sustainability.

5.2.10.1 Reuse

One of the most sustainable strategies for products is reusing already existing materials. In the case of this project, many of the components are common items which will likely not need to be purchased. In addition, the camera, Raspberry Pi, and Arduino, and motors are all components that could be used in other projects or for other hobbies. Consequently, these products will likely not be wasted once the user no longer has a need for this project.

Furthermore, this project is intended to be set up in an existing environment (i.e., on a table or in a room). Because it will have its own homing sensor and manual control, there will be very few areas where this product will not work. Therefore, there will be no need to have dedicated space for this, which saves money and makes this more realistic to implement in an already constrained business setting.

5.2.10.2 Repairability

Another positive aspect of this project that boosts its long-term sustainability is its ability to be easily repaired. All the individual components will be easily removable from the project. And since all the components are broadly available online, there is no reason that a normal user would not be able to repair this product themselves.

Furthermore, because the hardware and software of this product will be released under an opensource license, users will be able to modify the hardware to better suit their needs. This will hopefully reduce the number of users that stop using this product because it cannot serve their specific needs. Additionally, in the case that there is some bug which cannot be fixed remotely by this team, having the software be opensource means that technical users apply patches to the software themselves.

5.2.11 Constraint Conclusion & Summary

In conclusion, there are a handful of constraints that are guiding our project in how we can design and construct it. The main constraints are time and economic constraints, which will inform us how long we have for the design and building processes and which parts we can use. We also considered things such as health & safety and legal constraints by only relying on building processes that we were able to achieve with our current knowledge and did not risk our safety, as well as using opensource software with licenses that allow us to work without many restrictions and reuse excellent preexisting software.

Our time constraint is the strictest one, as the project has various due dates around which the entire project is planned. The main due dates involve the submission of the 60-page document including part selection and standards, the 120-page document that includes the design and testing plan, and the final demonstration of the completed project. Time must be carefully considered as we need to wait for parts to arrive and circuit boards to print before we can fully build the station.

The next strictest constraint is economic, as we do not have funding from a sponsor or the school, and our budget is supplied by the members of the team. This means that we need to consider the price of the parts used in the final design and during prototyping, as we have a limited budget. The

camera, microcontrollers, and PCB are the most expensive items that we need to budget around, so using the free and opensource software is a huge help.

The general concern for an environmentally friendly and sustainable project plays a significant role in the ultimate design. There are many long-term benefits to having a product that is user serviceable. There are less support calls, less full replacements being built, and less wasted machines. Since this team is small and lacks the funding to handle large quantities of requests for replacements, having a product that is sustainable is very important.

Finally, the last significant constraint that is also a huge benefit are the licenses that this project deals with. Between the GPL and the MPLv2, there are a lot of regulations that this project needs to keep in mind. However, there are also many benefits to not having to rewrite large chunks of code. It will be well worth it to make use of these software solutions.

The constraints listed in the previous sections will inform all the design decisions that we make and limit the time and monetary investments we can make in the project. Identifying these constraints is an important part of the design process and understanding them is necessary to succeed in creating our project.

6 Overview of Project Design

This section is brief; its main intent is to introduce an overview of the hardware and software design of this project. There is a significant level of details in the following section, so this section takes a moment to step back and explain how the whole system will function.

6.1 Project Electrical Architecture

Figure 24 illustrates the major aspects of the product's hardware design. It also includes some of the software that will run on (and control) the hardware. Additionally, the chart has been colorized to express which team member will handle each aspect of the project. Keep in mind that the size of each section in the diagram has no tangible meaning: it neither represents the scope of the work, nor the physical size of the resulting components.

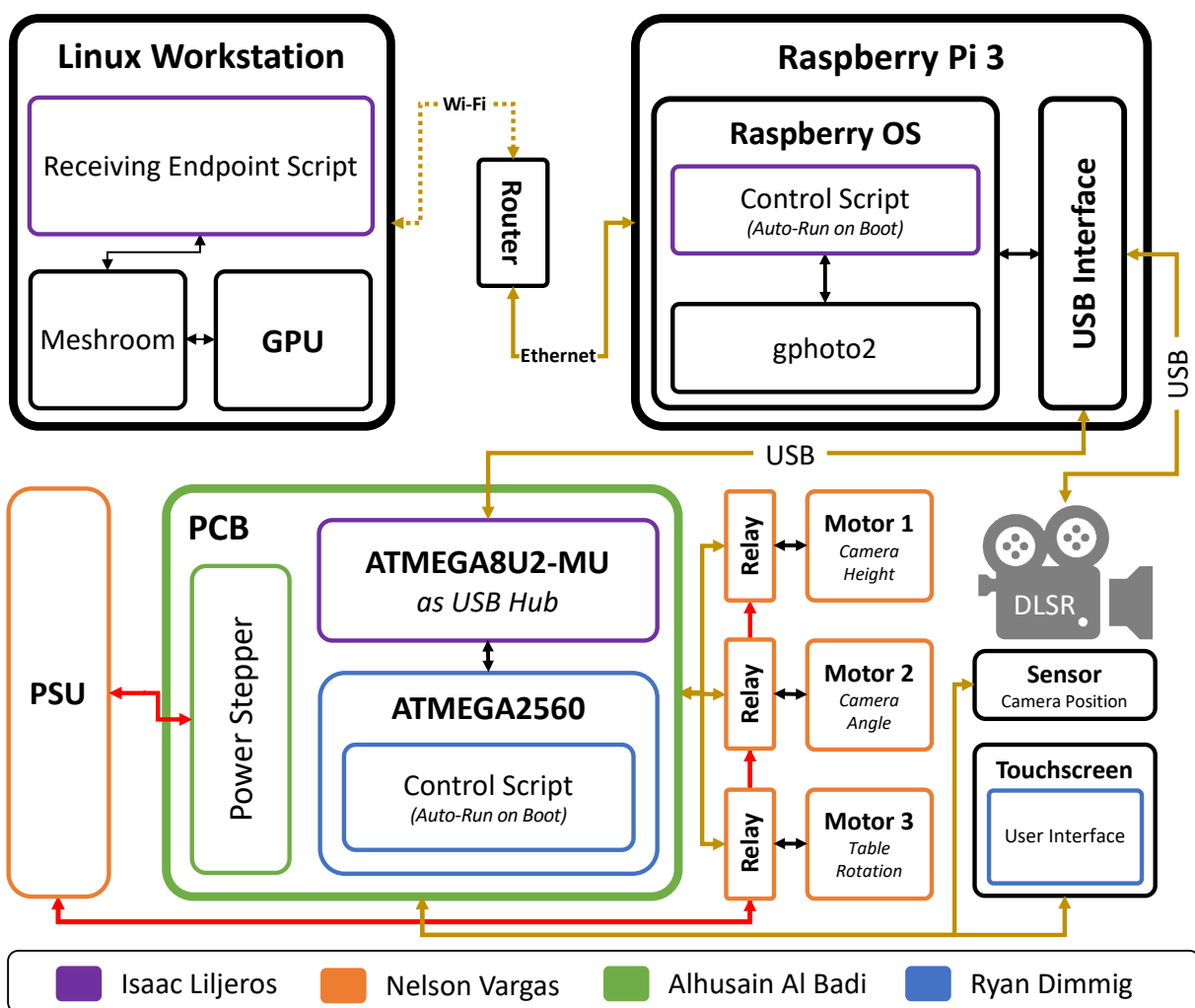


Figure 24: Hardware Diagram

Note that there are three main areas which must communicate with each other: the Linux Workstation (top-left), the Raspberry Pi (top-right), and the ATMEGA2560 Microcontroller. Each of these major components is responsible for attached sensors.

In this diagram, it should also be noted that there are some other details which are not present. For instance, the Arduino and power supply will be placed directly onto a printed circuit board. Some additional electronic components are needed to connect the Arduino to the Touch Screen. Finally, there are mechanical structures in this project which are not represented (e.g., there is a mechanical arm to which the camera will be mounted).

6.2 Project Software Architecture

Moving on to the software, Figure 25 details the comprehensive flow of the project's software. Note that this does not include any details of the various ways that the software will be developed. However, it is a useful tool when trying to understand the overarching goal that the software design team is going to try and accomplish. Again, like the hardware, the individual boxes are of similar size but they that is not intended to indicate that the software design the represent is of similar complexity.

The different shapes have different meanings in the following graph. The pill shape indicates a starting point. The rectangles represent actions that the software is doing. The rotated squares represent decisions. And finally, the parallelograms represent portions where the user needs to input something for the process to continue. The lines connected the different processes together to show the flow of the software. Not that after confirming, the flow loops back into the box “Select Options”; this indicates that the process loops here. In other words, the user is able to run the process back-to-back.

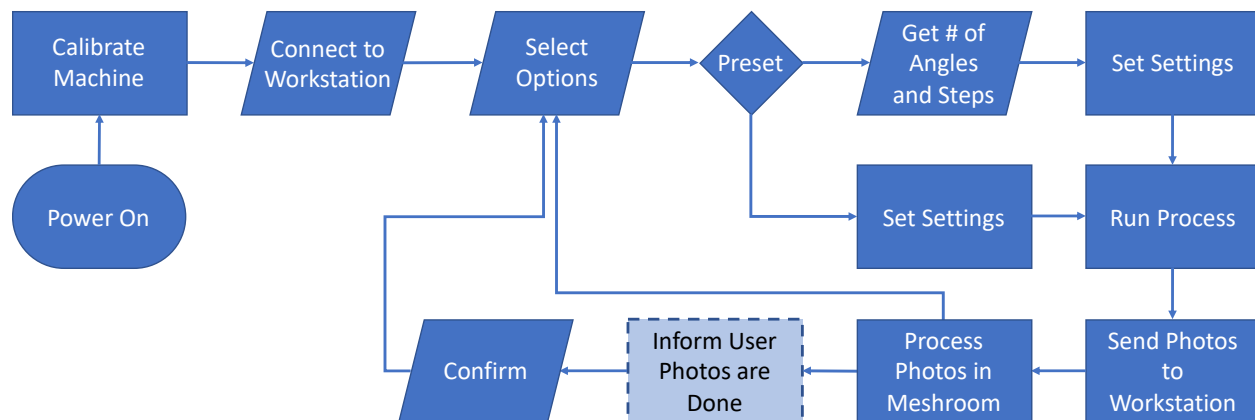


Figure 25: Comprehensive Software Flowchart

When the scanning process is done, the photos will be transferred to a remote workstation hosting Meshroom. While the flowchart does not show this, the transfer is a time-consuming process. If the user chooses to wait for the process to conclude, there will be a notification on the scanner letting them know the process was completed. Otherwise, they can immediately return to the main screen to start a new scan.

7 Design Details: Mechanical & Electrical Hardware

In this section, a presentation of the mechanical and electrical design details of the project is discussed. It includes a discussion of mechanical design and construction as well as electrical component operation and discussion. On the next chapter, a thorough testing of these components is investigated and validated. The current discourse of this chapter aims to provide an in-depth understanding of the underlying designs and principles of the individual parts of the system. Moreover, a discussion of the Printed-Circuit-Board (PCB) design schematic is shown where appropriate.

7.1 Mechanical Hardware

This section provides a comprehensive overview of the mechanical hardware design utilized to accomplish a successful automated image capturing process. It provides an overview of the essential components and technologies that allow for high quality and accurate image capture. The general and inspirational design has been already discussed in section 3.2. This section outlines the specific and modified mechanical designs that are tailored to the needs of the project. A discussion of the mechanical functionality is therefore included here, as well as a discussion of construction and material selection.

7.1.1 Vertical Motion Hardware Design

Inspired by the horizontal camera slider discussed in section 3.2.2, the team has gone through the process of adopting that design to fit the needs of our photogrammetry project. While the inspiration for the design has been collected from previous projects, a complete design and redevelopment has been made from scratch to specifically tailor to achieve the requirement set by the project. This redevelopment has allowed us to overcome the limitations and drawbacks of previous projects. One of the essential components that has been redesigned is the linear actuator that is responsible for carrying out the camera setup.

7.1.1.1 Operation Requirements

To integrate this design to the project, the design must comply with the following requirements:

- **Minimum rail length of roughly 1000mm.** To allow for a 700mm camera movement length plus other components integration such as base and belt tensioners.
- **A Supporting motor that is capable of handling about 2.5kg of weight.** This was addressed before by the selection of a high torque motor. Furthermore, the placement position for the motor is at the bottom/base.
- **A Supporting base capable of holding the structure upright.**
- **A Carriage platform capable of holding the Camera and its tilting mechanism.** This was addressed before the selection of a V-Gantry. Further details about the camera tilting mechanism are addressed in the next section.
- **Belt tensioning mechanism.** This will make the process of building and configuring the setup a lot more manageable and easier.

7.1.1.2 Design Overview

With requirements set, the team has gone through the process of designing the mechanical structure of the vertical camera slider. As mentioned before, the structure is supported by a 20x40mm aluminum rail. The carriage platform is also supported by a V-Gantry that is connected to the aluminum rail by a fitting wheel that slides through the grooves of the rail. To allow motion, the gantry is connected with timing-belts on both ends. The motor, placed at the base, is connected with the timing belts using a pulley coupler. This concludes all the purchased components of the structure. All other components were either sourced from the web or designed by one team member using Fusion360. Figure 26 illustrates a 3d visualization of the structure.

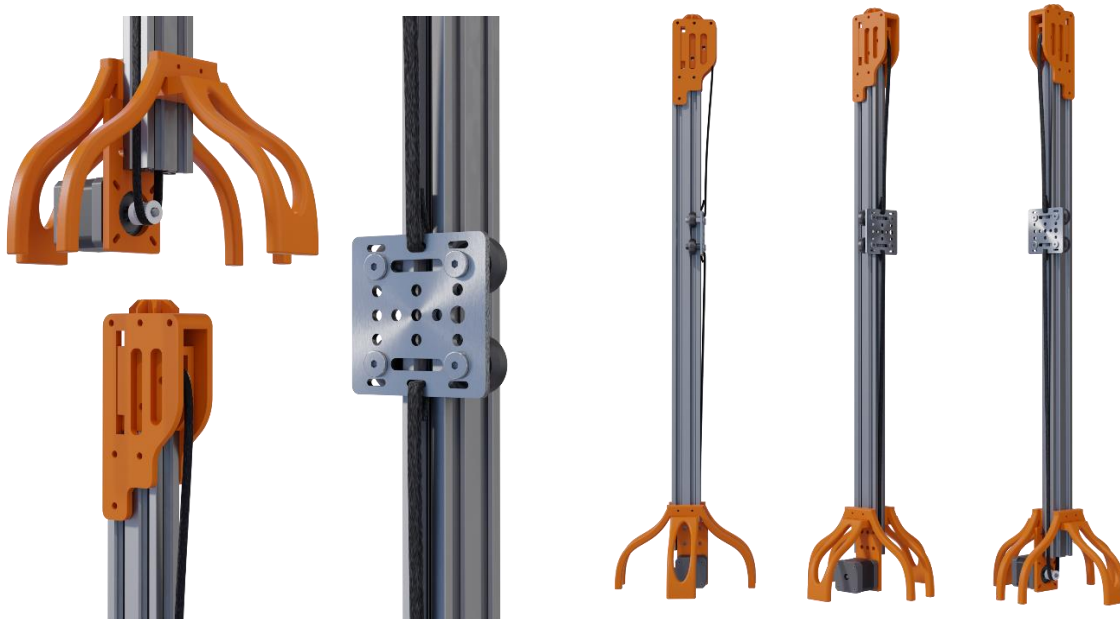


Figure 26: Vertical Motion Design 3D Visualization

To hold the motor in place, a placement bracket was designed to mount on one of the sides of the aluminum rail. The base, responsible for holding the structure upright, consists of a four-legged design capable of supporting the weight of the design. Lastly, on the opposite end to the motor is a belt-tensioner mechanism responsible for adjusting the belt tension to allow for smooth and predictable motion of the carriage platform. The belt tension is adjusted using a screw at the end that adjust the distance the belt wraps around, and hence, controlling the tension.

7.1.1.3 Construction Specifications

All designed components are 3D printed using additive manufacturing. The printing material of choice is Polylactic Acid (PLA). To ensure rigidity, the printing layer is set to 0.15mm with a 50% infill. Moreover, all the components are mounted to the aluminum rail using M5 and M4 screws that are easily accessible from a traditional hardware store.

7.1.2 Camera Tilt Hardware Design

The next essential mechanism to the success of the automated operation is the camera tilt mechanism. The tilt mechanism makes it possible to adjust the camera angle as it travels through the linear actuator (i.e., the vertical motion mechanism). As stated before, this is important to guarantee the object stays within the capture frame of the camera. The mechanism is mounted on the carriage platform alongside some other components, such as the homing sensors.

7.1.2.1 Operation Requirements

To integrate this design to the project, the design must comply with the following requirements:

- **Must hold a motor perpendicular to the carriage platform;** to ensure that the camera axis of rotation is perpendicular to the motor axis of rotation.
- **Connection mechanism to couple the motor shaft with the camera mount.** The mechanism must accommodate the purchased ball-head camera mount described in section 4.1.
- **Includes an integration design for the two sensors;** a requirement set by the auto-homing capability of the project.

7.1.2.2 Design Overview

With the requirement in mind, the team has gone through the process of designing the mechanism. The design of the mechanism is rather straightforward; it comprises of a box holding the motor in place, with additional room for components and wire housing. Figure 27 illustrates a 3D visualization of the mechanism.

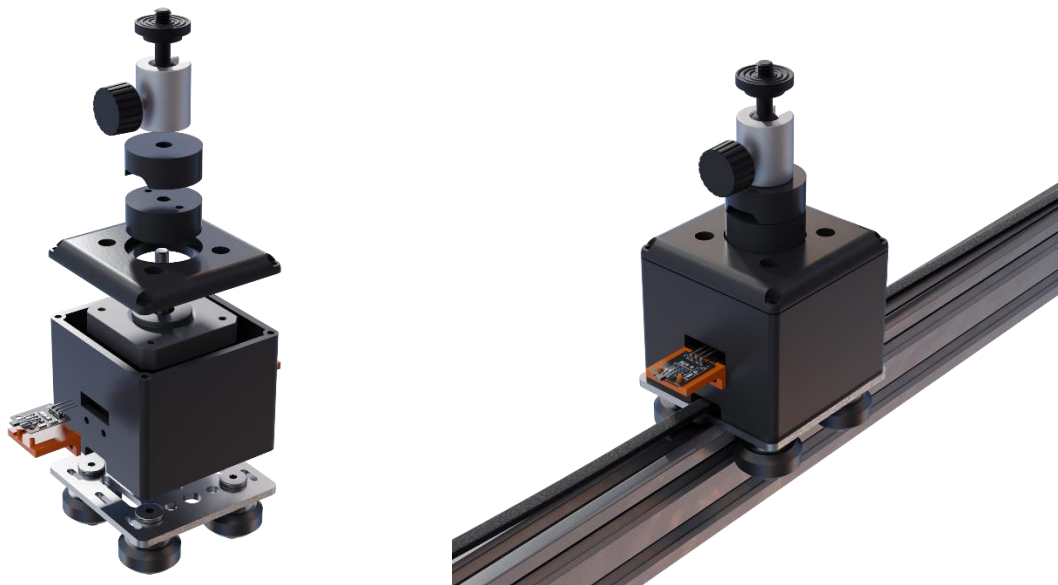


Figure 27: Camera Tilt Mechanism Design 3D Visualization

To make the integration of the ball head mount possible, two couplers have been designed. The motor shaft is connected with a motor coupler that rotates with the shaft rotation. The ball-head mount is then mounted on another coupler that in turn connects to the motor coupler. Moreover,

the design accommodates the two hall-effect sensors that are both mounted on the sides of the housing box.

7.1.2.3 Construction Specification

Similar to the vertical motion mechanism, all parts are 3D printed with the same material and setting specified previously. The components are mounted to each other using M3 and ¼” screws. The wires of both the motor and sensors are fed through the box through an opening to an opposite end of the camera.

7.1.3 Turntable Hardware Design

The last key component to the operation of the automated process is, of course, the turntable mechanism. To summarize its role, the mechanism is responsible for turning the object 360 degrees slowly to capture every angle of the object. Fortunately for us, the turning mechanism has been adopted from an open source, free to use design by Audiomatica and discussed in section 3.2.3. However, due to some component supply and cost restrictions, there is still a need to redesign some components to fit the specific components purchased for this operation.

7.1.3.1 Operation Requirements

To integrate this design to the project, the design must be repurposed to fit the following changes:

- **Accommodate 20x20mm aluminum rail instead of 15x15mm.** Due to supply constraints, acquiring a 20x20mm rail is more manageable.
- **Accommodate a 5” Lazy Suzan Bearing,** for similar reason to the pervious constraint.
- **Eliminate some unnecessary/overkill components to reduce cost.** The major change is the reduction from the use of 4 aluminum rails to 2, as well as some 3d printed parts that are not essential for the operation of the mechanism.

7.1.3.2 Design Overview

With the changes in mind, a redesign of the original mechanism has been developed. While it does not look different from the original design, it is more manageable to build and costs a lot less. Figure 28 illustrate a 3D visualization of the design.

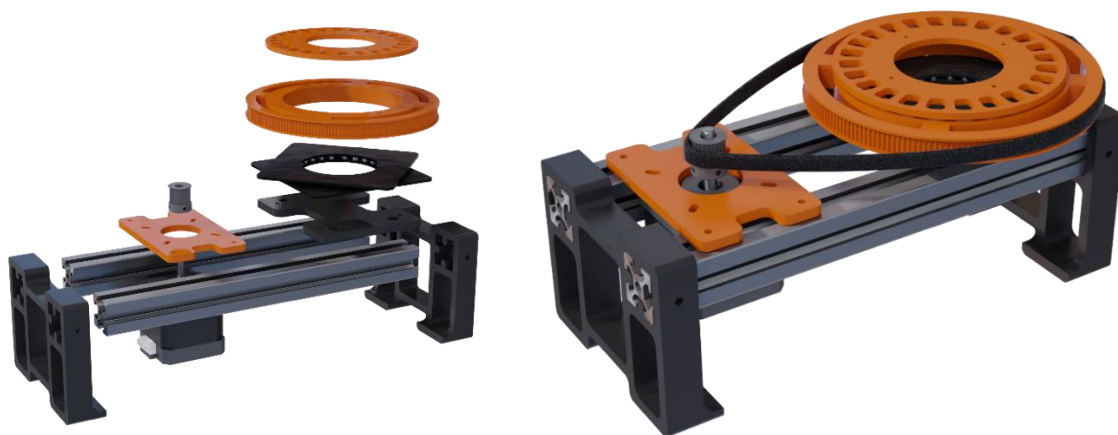


Figure 28: Turntable Mechanism 3D Visualization

The specific detail of the design is described in the original design by Audiomatica. The motor connected with a 30 teeth pulley is mounted on the aluminum rail. Another large 120 teeth pulley is connected to the Lazy Suzan bearing which in turn is connected to the aluminum rail. A timing-belt is wrapped around both gears and its tension are adjusted by adjusting the motor bracket distance from the larger gear. Additional plates can be mounted on the larger gear to increase the surface area or accommodate any weirdly shaped objects. The plates can be designed, and 3D printed, to the needs of the user.

7.1.3.3 Construction Specifications

Like previous mechanisms, all parts are 3D printed with the same material and setting specified previously. The components are mounted to each other using M3 screws.

7.1.4 Mechanical Overview Design: Mechanisms Integration

Now that all the individual mechanisms have been designed, they must be integrated together to ensure a successful operation. Fortunately for us, all the mechanisms have been designed to be self-sustained and easy to integrate to any design changes that might occur in future revisions of the project. In other words, any future updates, including future stretch goals, are easily manageable due to the modularity of all components. Figure 29 showcase all mechanisms coming together to construct the overall design of the photogrammetry station/rig.



Figure 29: Photogrammetry Rig 3D Design Visualization

7.2 Electrical Hardware

In this section of the paper, we go in-depth into the electrical hardware design. While in early sections of the paper, a brief discussion of the part's technology was made, this section aims to delve into the details of operation for these components and their specific role in the design. In addition, we will attempt to set up and configure the components to be ready to integrate them as part of the project.

7.2.1 DSLR Camera Setup

- Camera Mode
- F/stop, aperture setup ... etc.
- USB Connection.

7.2.2 Stepper Motors Setup

Stepper motors are the bone of the project operation. They are responsible for generating the motion required to automate the process of photogrammetry. As stated in part selection, the project will utilize 3 NEMA17 stepper motors, each with varying torque power. While the motors themselves do not require any sophisticated setup, their drivers do. Each driver will have a unique setup depending on its functionality (i.e., the mechanism it drives).

7.2.2.1 A4988 Driver Setup

In our setup, we are using three A4988 drivers; one for each motor. The setup for all drivers is the exact same apart from one feature discussed in the next sub-section. Now, to set up the motors we have selected, we need to first identify the voltage and current recommendation for the motors. The motor datasheet suggests a voltage supply of 12V and a max current input of 2A. Ideally, we will be running the motors at 1A each to reduce heat dissipation and ensure equipment longevity.

Now, the necessary connection pins for the module are outlined in the table below. The driver has a built-in chip translator, allowing it to only need two control pins: one for stepping and another for direction selection. Both pins are connected to the digital pins of our ATmega2560. There are few features of the drivers that we will not be needing, such as the reset and sleep feature. Thusly, both pins are connected high to disable them. In addition to all of this, we need to supply the driver with two power sources: 5V for in-board logic operation, and 12V for motor operation.

Table 21: A4988 Driver Pins Logic		
Pin	Logic	Description
EN	GND	Default Active Low – Enable Motor Operation
RST	HIGH	Default Active Low – Disable Reset Feature
SLP	HIGH	Default Active Low – Disable Sleep Feature
STEP	MCU Digital Pin	Send High Pulse – Step the Motor
DIR	MCU Digital Pin	High = Clockwise rotation

The driver's datasheet recommends a decoupling capacitor of 100uF to be placed near the 12V line to ensure smooth and undisturbed driver's operation. An extensive overview of the power delivery is discussed in the power delivery section. Another metric to ensure reliable operation is to set a current draw limit for the motor, which is fortunately a feature of the A4988 driver. The drivers come equipped with a potentiometer that allows to adjust the current draw limit. The datasheet recommendation to calculate the current limit is given by:

$$I_{MAX} = \frac{V_{pot}}{8 \times R_C}$$

Where V_{pot} is the voltage measured across the potentiometer, and R_C is the resistor value in series with pin. In our case, the resistor value is 100Ω.

7.2.2.2 *Micro-stepping*

An important feature relevant to the project is micro-stepping. By default, the stepper motor we have selected has a 200 step per full rotation. However, by controlling the control signal pulse length, the motor can increase the number of steps per rotation, and thus allowing for more precise and smooth operation. Fortunately, the A4988 driver eliminate the need to directly control the control signal and provides alternative way of selecting the stepping resolution; that is, by controlling the MS1, MS2, and MS3 pins. By default, the motor is in full step mode with all MS pins set LOW. The logical (High and Low) combination of these pins switches the stepping resolution. In our project, the vertical motion motor as well as the turntable motor are set to full step mode as to simplify the circuit schematic. The camera tilt motor, on the other hand, is set to half-step mode to provide smoother tilting and precise control.

7.2.2.3 *Integrated Schematic*

XXXX

7.2.3 **Auto-Homing Sensor Setup**

An important feature of the photogrammetry station is its ability to reset to a home position automatically. This feature is enabled by two hall-effect sensors and two magnetics. The underlying principle is magnetic interference; the sensor detects the presence of magnetic field, changing the voltage measure across it which can be detected by the MCU. More specifically, whenever a magnetic field is applied perpendicular to the flow of current, a voltage is induced. In our case, we have opted for the module KY-003-A3144, a non-latching hall-effect sensor. Meaning that the sensor gives output high (5V) whenever the north pole of the magnetic is close by and switches low as soon as the magnet is removed. The connection for the pin is straightforward: two pins for power (5V and GND) and digital pin reading the sensor state. The digital reading pin is hooked as an interrupt pin to the Atmega2560 to allow for ease of programming and software integration. Note that sensor is activated depending on which pole of the magnetic is applied to which side of the sensor. In otherwards, both magnet placement and polarity matter.

7.2.3.1 *Installation*

As shown in previous hardware diagrams, both sensors are installed as part of the camera tilt mechanism. One sensor operates as the lower point limit and the other as the high point limit. One design consideration that was not shown before is the installation of magnets. Both magnetic are

installed as part of the rail, each placed at the desired upper and lower limits. As the camera carrying platform moves up and down, the sensor will come within distance contact with the magnetic, activating the stoppage protocol for the vertical motion motor.

7.2.4 Touch Screen

The touch screen is a very important element of the photogrammetry station, as it makes up the user interface for communication between the user and the station. The user interface is wired to our PCB and is controlled by an ATmega2560. The touch screen hardware needs to be modified slightly to function as intended, and it needs to be connected to various pins on the ATmega2560.

7.2.4.1 Hardware Modifications

The touch screen came equipped with two modes of communication, SPI and 8-bit protocols. To implement one of them, we needed to slightly modify the breakout board to set our protocol. We decided to use the SPI communication protocol, as it required fewer connections, which is both simpler and uses fewer pins on the ATmega2560. To use the SPI protocol, we needed to solder the IM2 Jumper on the back of the breakout board to the 3.3V connection, as shown in Figure 30. This told the screen to communicate using the SPI pins on one side of the board. We also needed to solder pins into place for the connectors on the board, as this was necessary to establish a reliable connection to all the SPI pins so that we could prototype the screen on the breadboard.



Figure 30: Touch Screen Breakout Board IM2 Jumper Connection for SPI Protocol

7.2.4.2 Pin Connections

Connecting the touchscreen to the ATmega2560 required us to first connect the screen to power and ground, and then connect the 5 pins for SPI communication. We connected the screen's CLK Pin to the ATmega2560's Digital Pin 52, the MISO Pin to Digital Pin 50, and the MOSI Pin to Digital Pin 51. We also connected the Chip Select Pin to Digital Pin 10, and the Data/Command

Pin to Digital Pin 9, but this was a mostly arbitrary choice for testing and may be changed in the final design.

We then connected the additional 4 pins used for the screen's touch capabilities. We connected Y+ to Analog Pin 2, X+ to Digital Pin 8, Y- to Digital Pin 7, and X- to Analog Pin 3. These choices were also somewhat arbitrary and may change for the final design, as two analog and two digital pins are all that is needed for the touch screen.

- PCB Schematic.

7.2.5 Power Delivery

- Intro (conversions).
- Components selected.
- PCB Schematic.

8 Design Details: Software

The following section explains the software of the project. We will go into detail about individual pieces of software, their role in the project, and how they interact with other aspects of the project. Our project involves a variety of software used to automate the photogrammetry process. We are using a Raspberry Pi 3 minicomputer to handle interactions with the camera and 3D reconstruction software, an ATmega2560 microcontroller to control the user interface, calibration sensor, and motors, and a separate computer that runs the reconstruction software and takes commands from the Pi. The software design consists of multiple aspects that interact with each other to achieve a fully automated process between multiple computers.

8.1 Raspberry Pi Software Design

The first section of the software design that will be considered is the Raspberry Pi's control program. See Figure 24 for more details on where this program fits into the bigger picture. The most important detail is that the Pi's control program sits at the center of the project. Though it is not necessarily the most important part, it does have multiple avenues of communication which it must keep track of.

8.1.1 Requirements

Once the computer has booted and established a connection with the ATmega2560, the actual main program can commence. At a high level, these are the goals of the Raspberry Pi:

- **It must make it easy for the ATmega2560 to utilize the camera.** The Raspberry Pi adds the ability to easily interface with a camera. The ATmega2560 cannot easily do this on its own.
- **It must enable the scanner to communicate with the remote Linux workstation.** The Raspberry Pi has external connectivity capabilities (i.e., Wi-Fi and Ethernet) that the ATmega2560 does not.
- **It must save previous user preferences.** The Raspberry Pi has ample non-volatile memory that can be used to save user preferences.

These services must all be implemented in a reliable and transparent way. The user should never have to worry about how this control script works or whether the Pi has booted. These processes should be entirely automated.

8.1.2 Implementation Details

On Raspberry Pi's, there are two main languages that programs are typically written in: C and Python. Of these two, Python will likely be easiest to use. This is primarily because of the abstractions provided for USB connections. However, there is a lot more involved in getting this program to work than simply writing the control script.

8.1.2.1 Initial Boot Sequence

When the Raspberry Pi boots up, it should immediately launch the python control script that controls the device. This will need to connect to the ATmega2560 as soon as possible in order to introduce as small of a delay as possible. The following image helps to show exactly what will go

on in this section. Note that there are some details referring to inter-device communications. These have been defined in more detail in a later section.

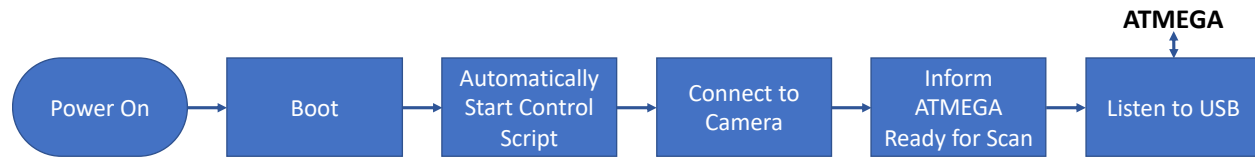


Figure 31: Boot Sequence

First, the device will boot. This usually takes around fifteen to twenty seconds on the Raspberry Pi. Following this, the control program will be run from a shell script that has been configured to run on boot. This can be done easily using the `rc.local` file on the Pi [29]. By adding a command to run the python code (using python3), the script can be automatically run on boot. This means that there will be no intervention necessary from the user to get things working. And if something ever goes wrong, restarting the machine will fix the issue.

Once the python control program is running, it will need to establish connections with the other devices which are directly attached to it: the camera and ATMEGA. Both devices will be physically connected to the Pi. To connect to the camera, the control script will need to use `gphoto2`. The summary feature in `gphoto2` can be used to confirm that the camera is attached. Connecting to the ATMEGA2560 will be very simple: a single line in python will attempt to establish a connection with a USB device [30].

Finally, the Pi will let the ATMEGA that it is finished booting. This will allow the ATMEGA to stop showing the loading screen. Then the Pi will begin listening to the Arduino over the USB bus to know what to do next. A few sections later, the inter-device communications that facilitate this communication will be discussed.

8.1.2.2 Main Program Control Flow

Now the main portion of the program will be defined. There are multiple avenues of control flow that could be followed. Ultimately, it depends on what the user has selected on the screen, which is connected to the ATMEGA. Rather than having the Pi or the ATMEGA be the master in this design, each will have control at different points in the process. As seen in Figure 31, the control initially resides in the hands of the Pi, but it is quickly handed over to the Arduino.

Figure 32 shows how the Pi's code will follow the commands coming from the ATMEGA. At all times the python script will be listening to the USB. And based on what is placed in the USB buffer, there are four main paths that the Pi will follow:

- If the microcontroller requests to connect to the workstation, the Pi will need to be the one that attempts the connection. This is because the Pi is the device that can connect to the internet. It will attempt to connect to the workstations API running at the specified IP address that the microcontroller sent. It will then let the microcontroller know if it was successful so that the microcontroller can inform the user of what happened.

- When the user is setting up the scan, one of the options provided will be to use a previous command. This will need to be pulled from the Pi as the ATMEGA has not persistent storage.
- On that note, the ATMEGA will also need to send setups that have been used to the Pi so that they can be saved for future use. This will simply append the configuration to the end of a text file.
- Another control branch will be to take a photo on the camera. The ATMEGA cannot easily connect to the camera, so instead, it will tell the Pi that it needs a photo taken. The Pi will then take a photo using gphoto2. Once the photo has been taken and stored, it will return control back to the ATMEGA.
- Finally, once the process of taking the photos has been completed, the photos will need to be sent to the workstation hosting Meshroom. This section will utilize the IP that was sent earlier in the process to access the workstation. It will then let the ATMEGA know when the files have been successfully uploaded. As Figure 25 shows, the user can optionally go straight into another scan while Meshroom runs. However, if they wish to wait till Meshroom has been completed, the Pi will also wait for a completed response from the Linux workstation and return this to the ATMEGA.

Note that every one of these branches could be easily implemented with a function that is called from a main loop which continually checks the value in the USB buffer. And at the end of each branch, a value will be sent back to the ATMEGA by writing something to the USB buffer.

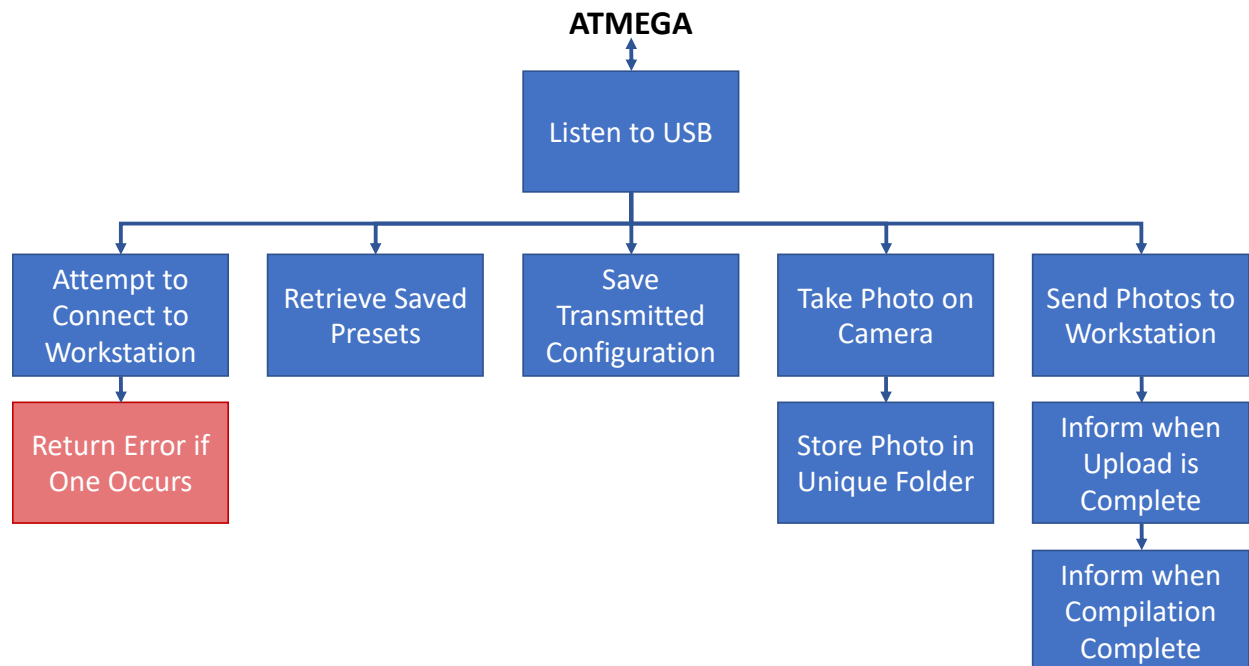


Figure 32: Control Program Flow

8.1.3 Development Environment

To easily implement the code for the raspberry pi, there will need to be a useful development environment. In the case of this project, the development environment will likely use Visual Studio

Code on a separate machine though an SSH connection. This means that the raspberry Pi will need to be on the same network connection as the development box.

To test the inter-device connections, the Pi will be connected to an Arduino and/or a ATMEGA2560 to validate the parts of the product. This will enable portions of the project to be tested before the entire project is completed. This is vital to being able to complete the project in a reasonable amount of time.

8.2 ATmega2560 Software Design

At a high level, the ATmega2560 software will involve the coordination of the user interface, sensor, and motors, as well as communicating with the Raspberry Pi. The ATmega2560 will utilize the USB communication protocol to communicate with the Raspberry Pi and trade off control, as well as send user settings from the interface. It will also use I/O pins to communicate with motors to move them, as well as the sensors to detect the position of the camera for calibration of its position.

8.2.1 Requirements

Once the Raspberry Pi has established a connection, the ATmega2560 will begin the process that is visible to the user. The goals of the ATmega2560 are as follows:

- **It must display options on the touchscreen and collect data about user settings.** This information will then be used to coordinate the motors and sent to the Raspberry Pi to coordinate the picture taking process.
- **It must calibrate the position of the camera.** This will involve using the homing sensors to find the position of the camera on the stand, which needs to be done before the Raspberry Pi can take control of the picture taking process.
- **It must control 3 stepper motors to position the camera for taking pictures.** It will use one motor to control the camera vertically, one motor to control the angle of the camera, and one motor to operate the turntable to spin the object.

The ATmega2560 should reliably be able to coordinate all these devices, which means that some of the most complicated software in the project will be implemented in the ATmega2560. The organization of the software will be very important, and so it will be broken down, in the following sections, into the interaction with each element.

8.2.2 User Interface Software

The software that controls the user interface involves the communication between the ATmega2560 and the display to write an image to the display and read the user's touch. We must also consider the layout of the user interface, which will be designed in the software using a graphics library.

8.2.2.1 Screen Interaction Logic

The software design for the user interface needs to involve two-way communication between the ATmega2560 and the touchscreen. This will involve the SPI communication protocol built into

the screen controller and ATmega2560. The touchscreen model we chose to use, the Adafruit 3.5" 320x480 Color TFT Touchscreen, comes with two software libraries to make the process much easier. One library is used to draw to the screen and utilizes various basic shape rendering code, as well as pictures and texts in a variety of fonts. This will help with drawing the options to the screen for the user to select, since it will be important to display text to ask the user to select settings. Shapes such as arrows and boxes are also useful, as they can be used to denote where the user is supposed to press on the screen. The other library is used to take input from the touchscreen and allow the software to read user input. This is done with a coordinate system that reads the x and y coordinate of the pressure on the screen, as well as a z coordinate that denotes the amount of pressure. This will help with easily reading the user's contact with the screen and translating it into usable data in the software. User interaction with the touchscreen is important, and these libraries allow for the process to be made easier.

Planning out the logical flow of the program that will control the touchscreen and process user input is a helpful step in the software design. It will begin with the ATmega2560 communicating with the screen to display the instructions upon starting, which involves helping the user connect to the network. Then the screen displays the option selection menus where a preset can be selected, or options such as the height of the object and number of pictures to take can be individually set. Then the screen will take input from the user via touch capabilities and transmit the pressure information to the ATmega2560. The code will then interpret the touch coordinates based on which menu the touchscreen is displaying, and the software will keep track of the options, which will affect what the motors will do. After the camera position is calibrated, the screen will display to the user instructions for focusing the camera on the object. The screen will then wait for the user to press start and the automated photogrammetry process will begin. The general logic of the software is shown in Figure 33.

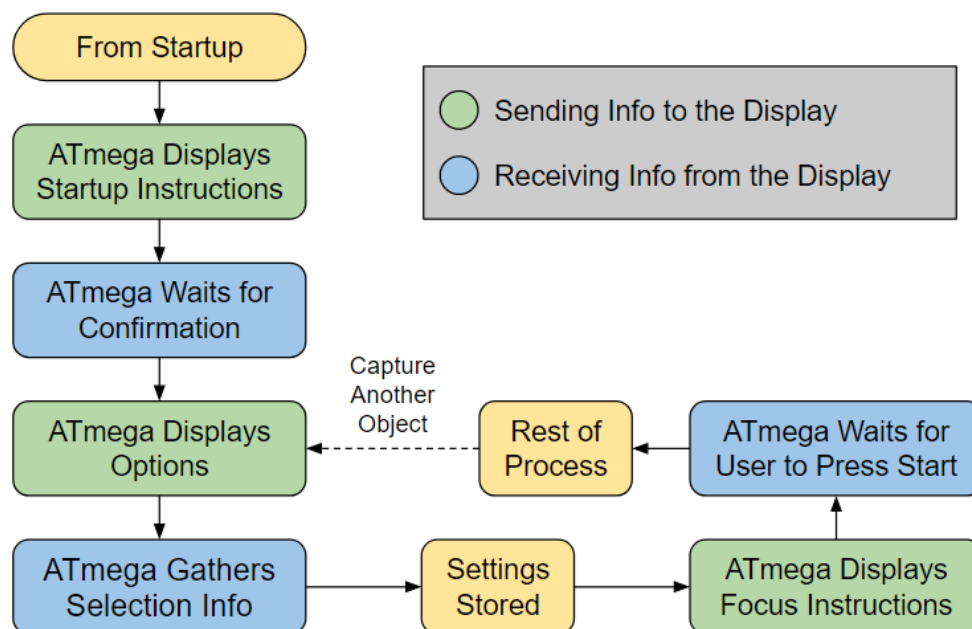
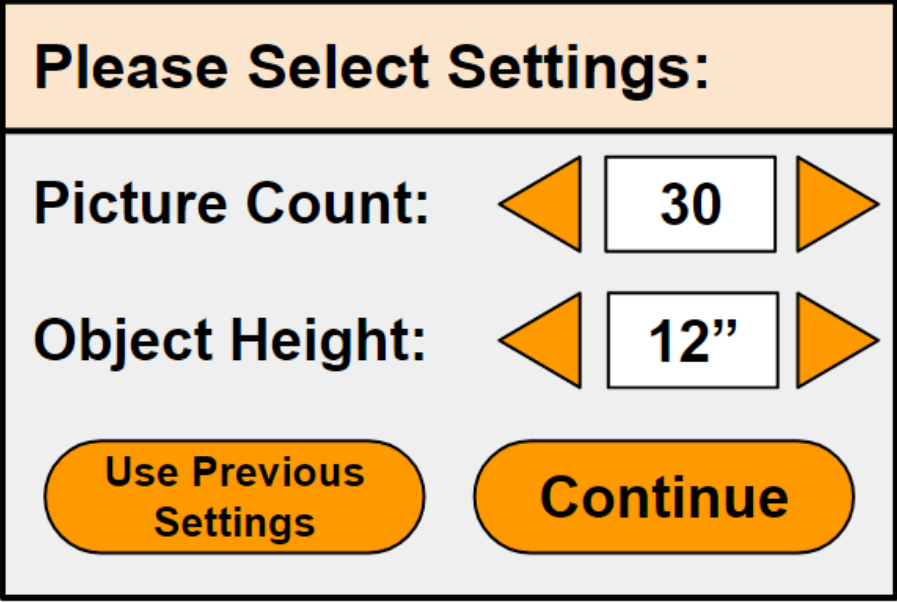


Figure 33: User Interface Software Diagram

8.2.2.2 UI Design

The layout of the user interface is also important, as it must be simple and clear for the user to input their settings and begin the photogrammetry process. There will be two main types of screen layout to display, one that is mostly instructions for the user to follow and a confirmation button to continue, and one that displays a variety of options with a way to select a response. The first screen layout type will simply consist of text that explains to the user what they need to do, whether it be connecting to a network or focusing the camera on the object, and a button to confirm the instruction is completed and the user is ready to move on. The second screen layout type will consist of a list of options, arranged vertically and left aligned, with arrow or a slider on the right to allow for the selection of settings. A mockup of what the second type of screen layout could look like is shown in Figure 34.



The mockup shows a rectangular interface with a light orange header bar at the top containing the text "Please Select Settings:". Below this, the interface has a light gray background. There are two settings displayed: "Picture Count:" and "Object Height:". Each setting has a left-pointing orange triangle, a white input box with a black border, and a right-pointing orange triangle. The "Picture Count" box contains the number "30", and the "Object Height" box contains "12"". At the bottom of the interface, there are two orange rounded rectangular buttons. The left button contains the text "Use Previous Settings" and the right button contains the text "Continue".

Figure 34: UI Option Selection Layout Mockup

Some things to keep in mind for UI design are clarity with where to press on the touchscreen, as well as making buttons big enough for people with larger fingers to easily press. Text and buttons need to be of a considerable size, so it is easy to read and make contact with the area of detection for the button on the 3.5" screen.

8.2.3 Calibration Software

The software that calibrates the vertical position of the camera involves using the homing sensors, which are two hall effect sensors placed on the top and bottom of the rig. At the beginning of the process, before pictures are taken, the stand will need to be calibrated so that the software knows how much to turn the motors to move the camera. The rig uses stepper motors and belts that may not have a constant translation between real-world movement and number of steps for the motor to move. Therefore, it will be important to calibrate the camera position by moving the camera to

the top and bottom of the stand. Since the stand has a constant known height that the motor control software will be able to access, the calibration software will move the camera to the bottom of the stand, which can be detected by the lower hall effect sensor. Once the software knows that the camera is at the bottom of the stand, it will move the camera toward the top of the stand, which can be detected by the higher hall effect sensor. When the camera is detected at the top, the software will read how many steps the vertical motor took and be able to calculate the number of steps required to reach various points on the stand. Once the software knows a translation between steps that the motors have taken and vertical distance that the camera has travelled, it will be able to move the camera to the optimal position for taking pictures of the object regardless of its height. The software will be able to find the height of the camera to get a clear, straight-on image, and it will also be able to calculate a position to get a lower and higher angle view of the object. The general logic of the calibration software is shown in Figure 35.

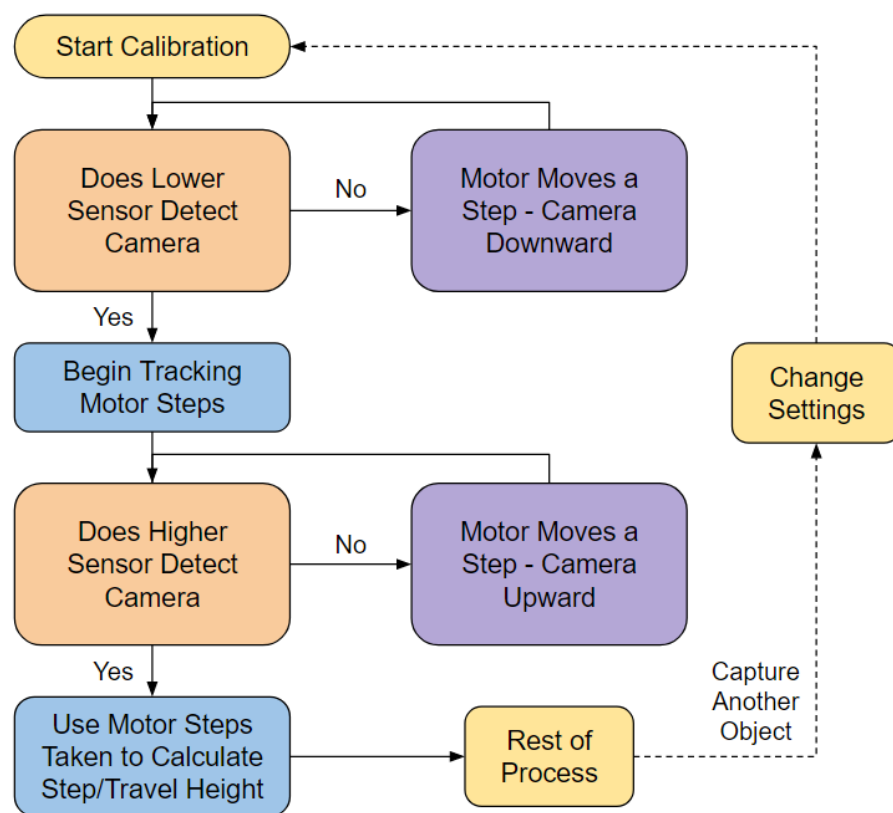


Figure 35: Calibration Software Diagram

Calibration will be performed before every capturing process to ensure accuracy of the camera position, but also because the height of the object may change for subsequent capture, and the code must recalculate the positions for taking angled pictures.

8.2.4 Motor Control Software

The development of the Arduino controlled motor code was kept in a modularized format for a few reasons. Firstly, this promotes conciseness and clarity for code readability. Modularized code is generally easier to comprehend since it is in sections and not as convoluted as it could be.

Secondly, modularized code makes for code that can easily be interrupted should the need arise. For example, if certain images are blurry or unfit for use in the Meshroom algorithm, the modularized setup means that a user can navigate to the portion of the code that handles that particular image and modify the code if needed. It also means that the development of user options on the software side is more straightforward. We can implement options on the user interface that allow for the user to select what settings to implement for the rig. Such as the size of the object being scanned or what image to retake. Another benefit of modularizing the code is that the developer does not have to write multiple scripts for different functions. Rather, the developer can have different scripts that simply reuse the code modules in different ways for the different functions. This way, if a certain error occurs multiple times, the development team can simply edit the part of the code that pertains to the certain functionality that keeps misbehaving. Additional functionalities that are preceded by the modularization of code include starting and stopping the automated process, and being able to run the process from a point which the user can choose. This is useful if say, an object the user is capturing topples over during the picture-taking process, and the user would like to restart the process from 2/3 of the way through the images. Figure 36 is a flowchart depicting this functionality:

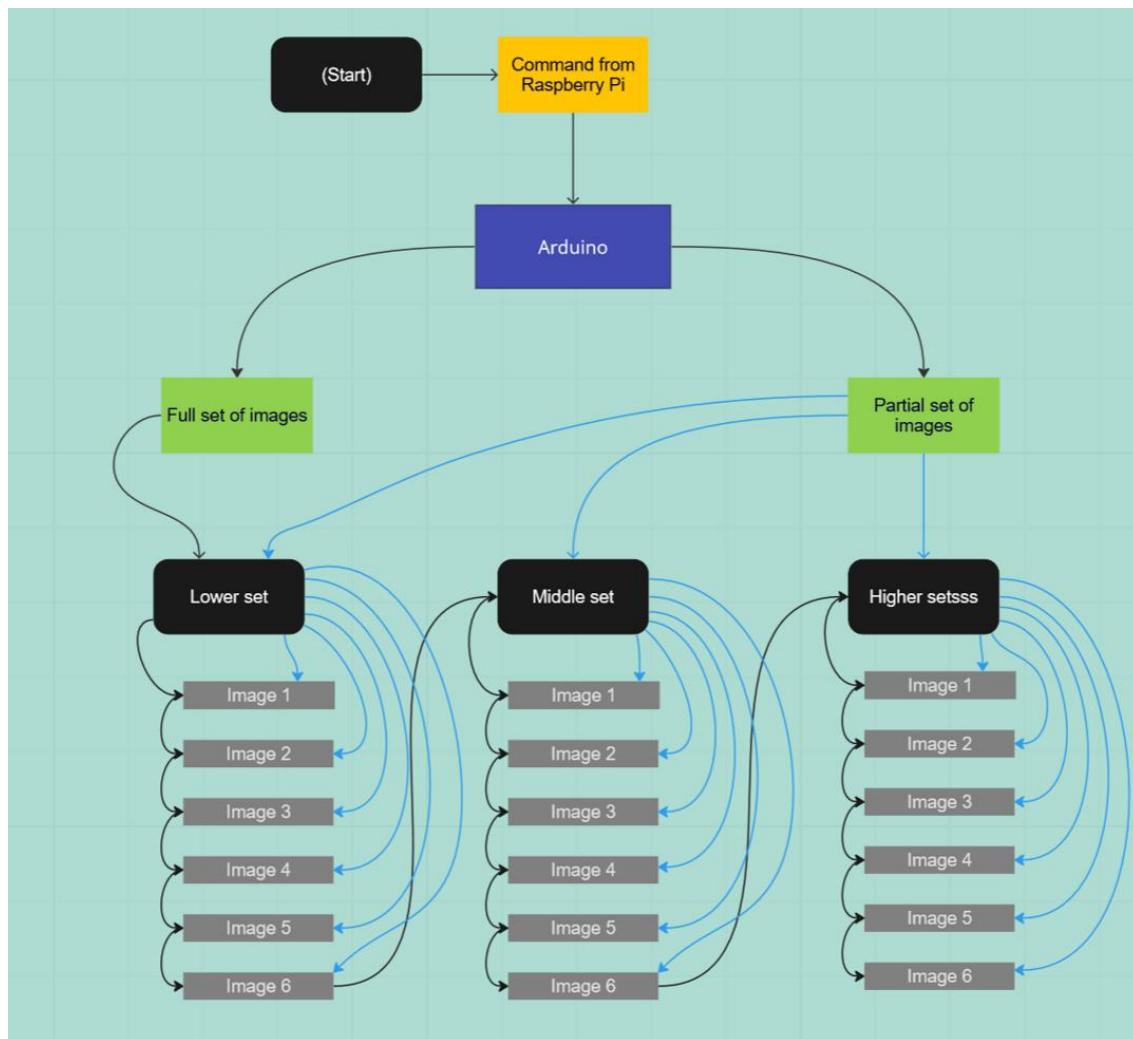


Figure 36: Flowchart Depicting Modularized Code for Motor Operation.

This relates to the motor movement methodology in this way: each image will have its own motor process. So, for images 1-6 for example, motor A will move the camera to the correct height and motor B will adjust the camera angle, both to remain fixed for this set of photos. Then motor C will rotate the turntable 60 degrees in between each image taken, and all the motors will reset to zero. This motor process is added to the previous figure to create the flowchart seen in Figure 37. Figure 37: Updated Flowchart Depicting Modularized Code and Motor Implementations.

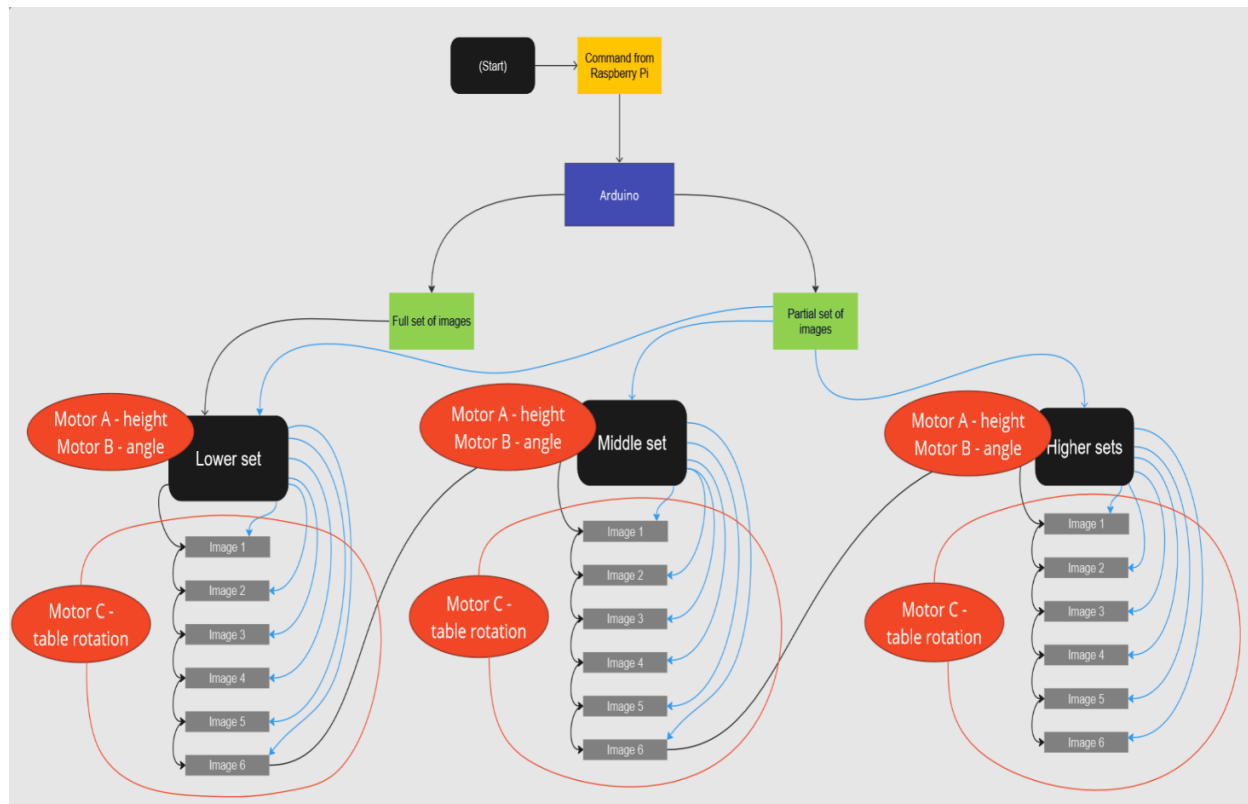


Figure 37: Updated Flowchart Depicting Modularized Code and Motor Implementations.

Each motor movement will be comprised of a section of code that rotates the motor a set number of steps, thus determining the angle of rotation or the position of the camera. Each motor driver (corresponding to one motor per driver) contains pins dedicated to the pulse (movement), direction of the movement, and an enable pin. By changing the direction of the movement and backtracking by the same number of steps used to move forward, we can reset the position of the motors. This could also be attempted by utilizing gravity, and just disabling the motors. The motors have some resistance to them even when completely powered off, so allowing them to “free wheel” down to the starting position would not damage the rig or the camera, since it will be at a tame, albeit uncontrolled, speed. Figure 38 is a model depicting the ways in which the motors facilitate movement within our design:

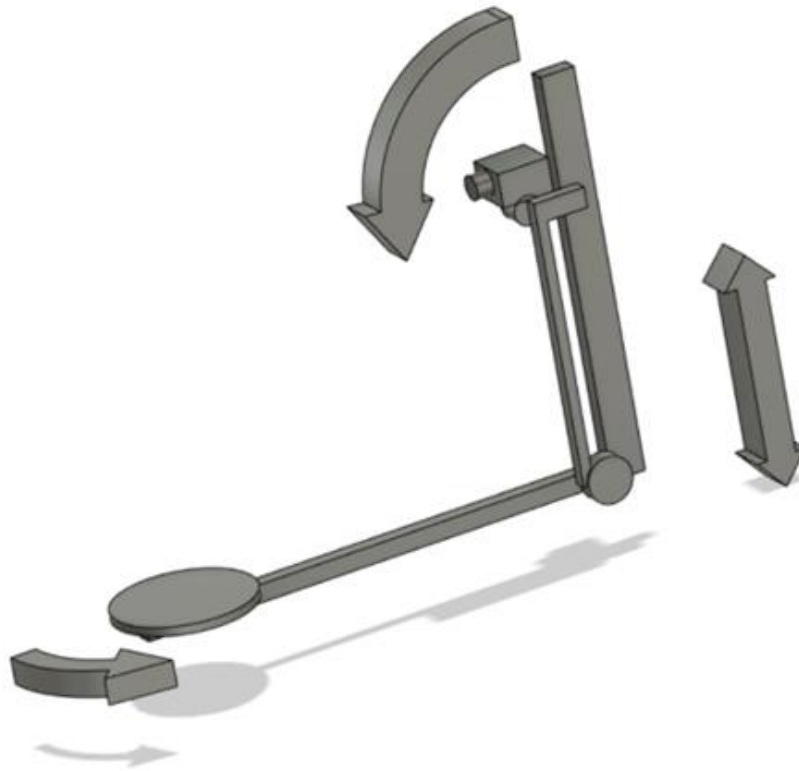


Figure 38: Motor Mobilization

8.3 Linux Workstation Software Design

XXX

8.4 Inter-Device Communication Protocols

Though it is important the individual parts of the project are well designed, if they cannot communicate effectively with one another, then it does not matter how well they are designed. As such, this section seeks to define the exact communication protocols that will be used between the individual devices. This includes the connection between the Raspberry Pi and the ATMEGA and the connection between the Raspberry Pi and the Workstation. Since the more complicated of the two is the connection between the Pi and microcontroller, that is where this technical discussion will begin.

8.4.1 Raspberry Pi & Arduino Communication

Since the project is essentially handing off control between the Pi and the ATMEGA, both will be listening to the value in the USB buffer waiting for the other device to tell it to do something. Table 22 details what the different communications will be between the two devices.

Table 22: Raspberry Pi and ATMEGA Communication

Command	BUS Value	Description
Send IP		
Take Photo		
Take Photo Done		
Ready to Transmit		
Process Complete		
Save Settings		

8.4.2 Raspberry Pi and Workstation Connection

XXX

8.5 Development Environment Introduction

An important aspect of designing software is being able to have a development environment that facilitates trial and error. The overarching goal in setting up the various development environments was giving the developers an area to fully test the software before integrating with the rest of the project. This will hopefully mitigate the number of bugs and technical difficulties which must be solved at the final stages of the project.

8.5.1 ATmega2560 Development Environment

The software for the ATmega2560 will be written using the dedicated Arduino IDE. This development environment will make it easier to structure and test the code that the ATmega2560 must run. Along with the IDE, the team will use an Arduino development board to prototype the system. The board we're using is a variation of the Arduino Mega because it implements the ATmega2560 microprocessor. Though this board will eventually be substituted by an ATmega2560 mounted directly to a PCB, it will still provide an excellent environment to test initial software. Because of the portability of the high-level C language used, it will be simple to transfer the code written for the development board to the PCB-mounted microcontroller.

The Arduino development IDE will simply be run on a personal computer. Most computers have ample USB ports, so they will be able to communicate with the Arduino to write code to it. After development is complete, the code will be housed on the PCB with the ATmega2560, and a USB connection will be established between the ATmega2560 and the Raspberry Pi for communication.

The Arduino IDE provides a user-friendly development environment that makes it easy to view and edit code, compile, and send it to the development board, and view the Serial Monitor to test communication and debug the code. Various libraries will be used to control the stepper motors and the touch screen, which are easy to download via the IDE and import into the project.

8.5.2 Raspberry Pi Development Environment

XXXXXXX

9 Prototype Testing & Integration

The testing process is a continual part of the project that helps us evaluate our progress and whether we will meet our goals. In this section, we will explain our testing procedures for various hardware and software aspects of the project. The testing involves making sure our components function as intended within our allowed tolerances to achieve the goals laid out earlier in the report, as well as making sure our software functions as intended to properly manage the coordination of all components. The methods and results of our tests are described in the following sections. Our testing includes the demonstration of individual components to make sure they work in isolation, such as the ATmega2560 communicating with just motors or the touch screen, as well as testing that many aspects of the project work when integrated. We will detail our testing of individual parts, as well as our plans for testing the project in a more integrated state once a full prototype has been developed.

9.1 Mechanical & Electronics Testing

XXX

9.1.1 Vertical Motion

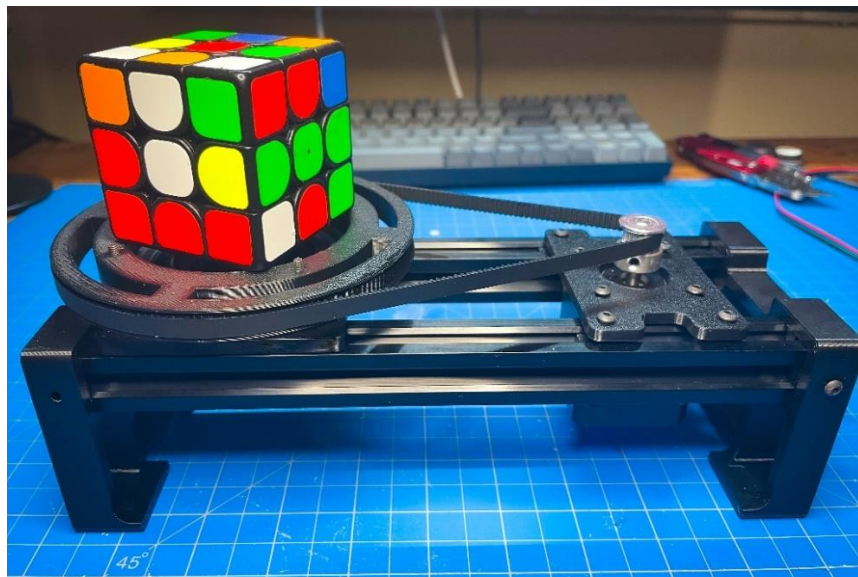
Includes: Design, Motors Operation, Belt Tensioner, and Weight limit

9.1.2 Camera Tilt Motion

Includes: Design, Motor Operation & Camera Weight verification.

9.1.3 Turntable Motion

The next mechanism to test is the turntable mechanism.



9.1.4 Auto-Homing Sensor Testing

The last mechanism to check is the auto-homing feature. Again, this feature is enabled by two hall-effect sensors. The sensors are placed at each side of the camera tilt mechanism. The magnets themselves are mounted on the rail, one at each end of the aluminum rail. Figure 39 showcases both mechanisms (the sensor and the magnet) installed.

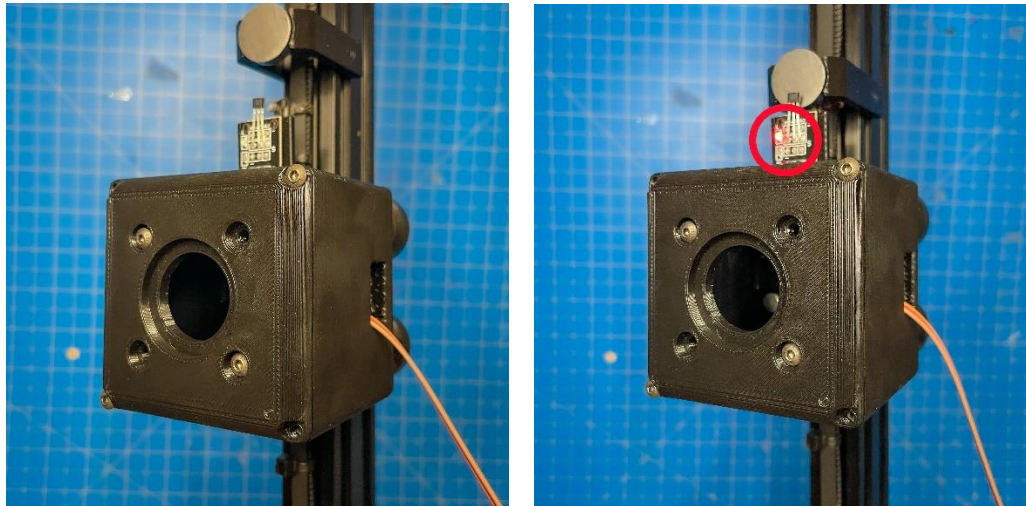


Figure 39: Auto-Homing Sensor Installed

During the test, we established that the mechanism is functional according to our requirements. The sensor module we have selected has an onboard equipped LED light when magnetic field is detected. Figure 39 showcases the mechanism working. It is worth noting that during the initial testing of the mechanism, we used a regular art and craft magnet that was very hard to detect by the sensor, due to its weak magnetic nature. The detection range was about 5mm, and anything beyond that was hard to detect. Later, we purchased neodymium magnet that had strong magnetic nature, and the detection range has increased, making the mechanism more predictable and consistent. Additionally, the magnet pole facing the hall-effect sensor had to be the right polarity. In our case, the north pole was facing upward toward the sensor. If the magnetic polarity was reversed, the sensor would not detect the magnetic field. All in all, the mechanism works well and can be integrated into the project without any future modifications.

9.1.5 Power Supply

Oscilloscope testing explanation to fill space lol. The example paper did it

9.1.6 Touch Screen Testing

Our touch screen is a very important piece of electrical hardware that is needed for the user to interact with the photogrammetry station. The hardware aspects of the user interface, being the

physical screen itself, its wiring connections, and its ability to display and read user input, must be tested to ensure this screen will be suitable for our final iteration.

As explained in section 7.2.4, the touch screen breakout board requires hardware modifications such as soldering pins into the side of the board to establish solid connections and soldering the IM2 Jumper on the back to set the board to SPI communication mode. With these modifications made, we needed to test the screen to make sure the connections were made correctly. In addition, there were a total of 11 Pin connections that needed to be made to the Arduino development board to test the screen. The board needed connections to power and ground, as well as 5 digital pin connections to communicate with the display via SPI, and 2 analog and 2 digital pin connections to use the touch screen functionality. To test the display and touch capabilities of the screen to make sure they function correctly, we used the example code that was included with the screen. Figure 40 shows the touch screen board's connections to the development board and shows that the screen functions as expected and runs the example code designed to demonstrate the screen's display functionality. We also ran another test using the example code to demonstrate the screen's touch functionality. This test also worked as expected, and the board could correctly read user input anywhere on the screen.

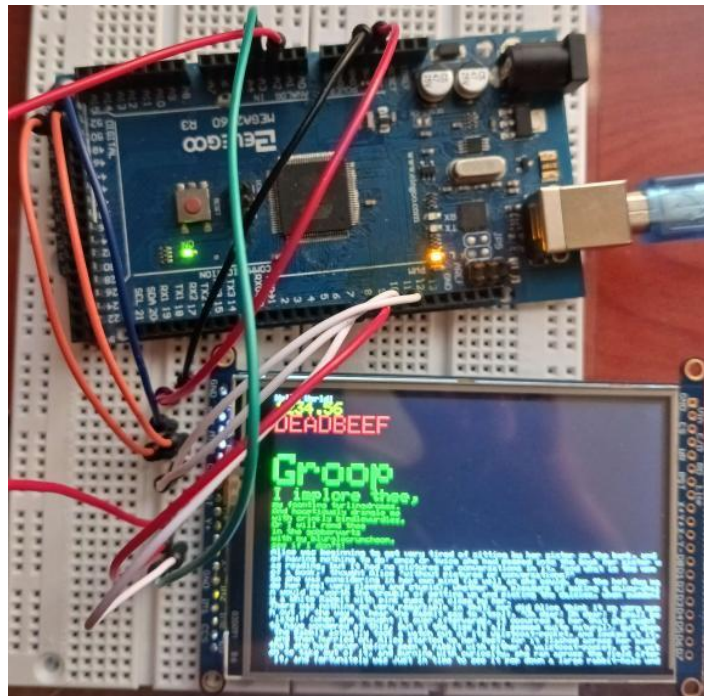


Figure 40: Touch Screen Hardware Test Using Example Code

9.2 Software Testing

In previous sections, we have described abstractly how specific portions of the project are supposed to function. In this section, we will discuss how the specific portions are implemented from a software standpoint. This will include more robust code functionality and some specifics on code commands used within our selected IDE's (Integrated Development Environment). Some small code examples will also be provided. This is a deeper dive into how and why our code works

with our selected parts. For our more general ideology on why we wrote our code in this particular way, please reference Section 8.

9.2.1 Motor Operation Control

Previously, the methodology for how the motor code functions with respect to the basic Arduino code structure was expounded upon. In this section, the code that drives and controls the motor movement will be explained.

The bipolar stepper motors we are using measure their movement in steps. The number of steps that a motor or motor driver specifies is how many steps are in one full rotation of 360 degrees. So, if a motor consists of 200 steps, each step it takes will rotate the motor shaft 1.8 degrees. This information helps us gauge how far we want to rotate the motor shaft, and depending on the pulley or gear we use in tandem with the motor, how far a component moves or rotates.

In our code, we implement different counter variables to keep track of how many steps are in each desired motion, for each particular motor. The basic framework for movement will remain consistent no matter how the motor is commanded to move. A for loop will iterate as many times as steps are desired, and within the loop, a pulse pin will be set HIGH, then LOW. This will move the motor shaft one step at a time. In between the pulses, there will be a millisecond delay to determine how fast the motor moves. It is important to facilitate the speed for the motors because if a motor is too slow, the project is impractical, and if any individual motor is too fast, it could cause issues during the automated process. If the camera angle is adjusted too quickly, it could result in a blurry picture. If the vertical motor is too fast, the rig could shake or even topple over. And if the plate rotator motor is too fast, it could blur the picture, shake the rig, or throw the object from the plate. Additionally, any of these motors moving too fast could result in a broken component, which is something that must be avoided.

In addition to controlling the speed of the motor rotation, a state variable will be used to determine which direction a particular motor should go. This is important for resetting the rig and/or going back to retake pictures. It is also important because this functionality makes the process faster. If our motors could only rotate one direction, we would have to reset by setting the motors all in a cyclical motion and run through that motion multiple times. That would also be prone to calibration issues, as over time the rig would likely get out of alignment. A snippet of code showing how the motor moves is shown in Figure 41.

```
//-----  
  
    digitalWrite(EN, HIGH);  
    digitalWrite(DIR, LOW);  
  
    for(count = 0; count<steps; count++){  
        digitalWrite(PUL, LOW);  
        delayMicroseconds(pd);  
        digitalWrite(PUL, HIGH);  
        delayMicroseconds(pd);  
    }// end for
```

Figure 41: Motor Movement Code Snippet

So, from the software standpoint, the code directly controls how much each motor rotates as well as which direction it rotates.

9.2.2 Touch Screen UI Design and I/O Handling

We previously tested the electrical connections of the touch screen, and in this section, we detail the tests we performed to make sure the libraries allowed us all the functionality we require for our final user interface. To do this, we wrote code to display a mockup of the user interface to test the ability to display text to the user, allow screen transitions, and update numbers on the screen without having to clear the whole screen and cause a flickering effect. Our demo code was designed to have two screens, one for selecting settings that allowed for changing two numerical options and continuing to the next screen, and a screen for telling the user to focus the camera and press Start when they are ready, which also included a Back button to allow changing between both screens. This test code shows that we will be able to display our intended UI design on the screen and easily take user input. Figure 42 shows the screens that we designed, complete with buttons to update numbers on the current screen and transition to another screen. The red arrows in the figure show the buttons that transition between screens.

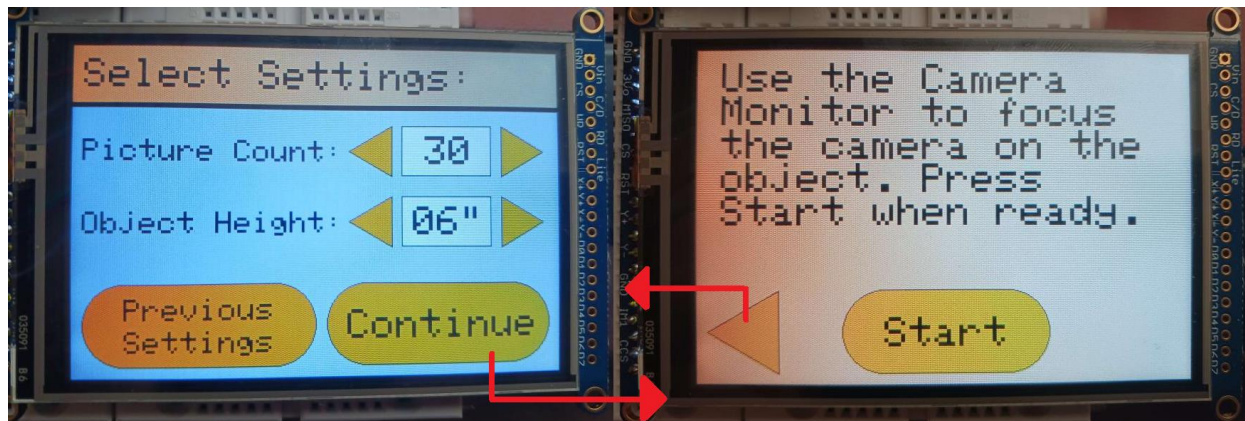


Figure 42: Touch Screen Software Test of our Intended User Interface

9.2.3 Raspberry Pi 3 & Atmega2560 Communication

Communication demo.

9.2.4 Camera Remote Capture

Image capture demo with Raspberry Pi.

9.2.5 Meshroom 3D Reconstruction Testing

Up to this point, all discussion related to the project's design and testing only involved the photogrammetry station itself, both its hardware and software. However, one of the major components of the project is the 3D construction software that will convert the images captured by the photogrammetry station to a 3D model. Therefore, it is important to also test the operational and working condition of the Meshroom, the 3D reconstruction software of choice.

9.2.5.1 Hardware Specification & Testing Setup

Before we start testing the software, it is important to note some key specifications that indirectly relate to the overall operation of the project. As stated before, Meshroom utilizes both CPU and GPU for fast operation. Therefore, it is important to note the hardware setup of the test. This specification will likely be the same specification we will be using for the final project demo to ensure correct and fast operation. The specifications are:

- **Operating System (OS):** Windows 10 64-bit
- **CPU:** AMD Ryzen 7 5800X (8 Cores, 16 Threads)
- **RAM:** 32GB (DDR4)
- **GPU:** NVIDIA 3080 (10GB RAM)

In addition, Meshroom initial setup is also worth noting. Through investigation of the operation of the program, the team has found the Meshroom creates a directory (the directory path is set by the user) that contains the cached data, as well as the settings for each step of the working pipeline. This will prove important later when the program is controlled via CLI commands. For now, we go through the manual process of testing the software.

9.2.5.2 Testing Dataset

Of course, to test the software we require a dataset of images to feed to the program. There are many datasets that exist on the web we can test on, but for the purpose of the project we require a specific kind of dataset. The requirements of the dataset must comply with the following:

- **Images are taken with a DSLR camera** to ensure that the same results are replicated by our own camera setup.
- **The captured object is of size comparable to what would be captured by our photogrammetry station.** Large dataset of large objects or scanned rooms are not within the operational conditions of the project.
- **The dataset must include deliberate errors.** This allows us to test the limit of the program, as well as ensuring it can handle errors if the process is to be made autonomous.

Fortunately, we have found a dataset that fits within these specs. A dataset provided with courtesy of Dr. Peter L. Falkingham, a Paleontologist professor that provided this [Styracaceous dataset](#) [31]. The dataset consists of 53 images and includes deliberate errors in lighting and background clarity meant to test the limits of the SfM reconstruction software. Figure 43 showcase an example image from the dataset. The background consists of a glass table that might present problems to the software due to reflections.



Figure 43: Styracaceous Model (Testing Dataset)

9.2.5.3 Operation Pipeline

With everything set up, we can shift our attentions to details of the software operation. Some basic and important details about Meshroom were discussed in section 4.9.1. The goal here to dive into the details of the program's operation in order to find the best working pipeline to function as the base line of operation. Of course, some details can be changed, but for a fully automated

experience, base line parameters and operation are in order. The graphical interface of the software is showcased in Figure 44 where it also shows the dataset selected and imported.

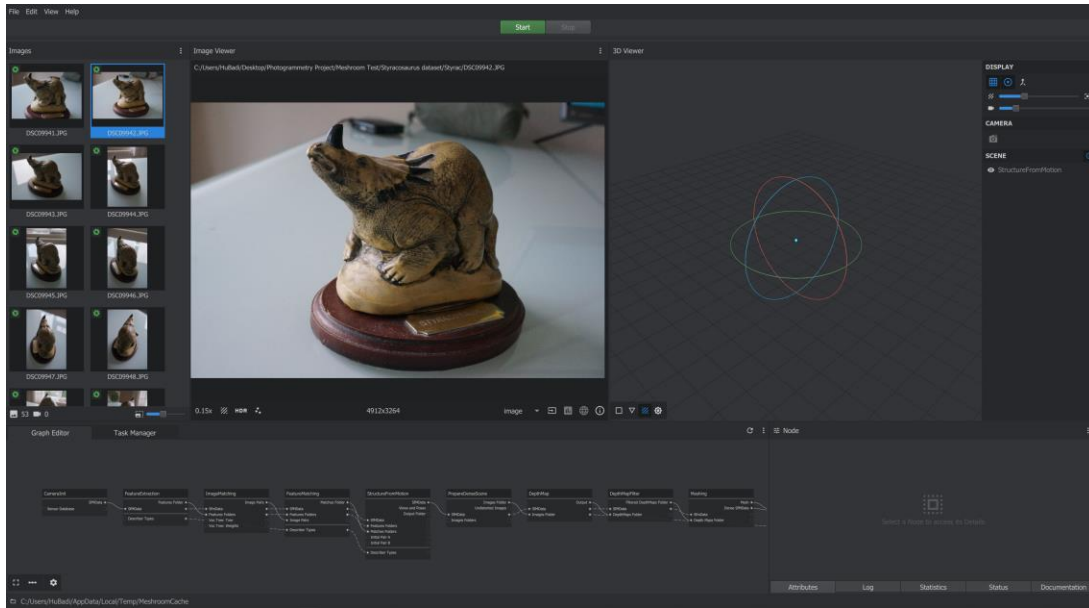


Figure 44: Meshroom Graphical Interface (image import)

As explained previously, Meshroom asks for a directory folder to store the cached data as well as any other relevant data such as the output data. In addition, Meshroom operates on node-based workflow where each node corresponds to a specific operation, and by default, Meshroom provides a basic startup workflow that works very well. However, this workflow is basic and lacks additional features that are required within our workflow; more on this later. An important discussion to have at the moment is related to the specific operation of each node in the workflow. Table 23 outlines the workflow nodes in order and provide a simple description of their operation. Nodes marked with the symbol * are the most essential nodes of operation. They are marked because within all nodes of Meshroom, there are many parameters to change and adjust within each single node, making the workflow complex and extensive. However, in this paper we will discuss the most important parameters to change that greatly affect the result and speed of operation. Those parameters are therefore included within the nodes marked with a star *.

Table 23: Meshroom Nodes Descriptions (in order)

Node	Description
CameraInit	Extracts metadata from images and match them
FeatureExtraction*	Extracts distinctive groups of pixels that are invariant
ImageMatching	Pair matching images
FeatureMatching	Performs feature matching on candidate image pairs.
StructureFromMotion*	Analyze matched feature to extracts geometric relationships

PrepareDenseScene	Exports undistorted image (useful to Texturing Node later)
DepthMap*	Estimates the depth value of each pixel from the camera.
DepthMapFiltering	Filters incoherent depth map values that do not match
Meshing*	Creates a dense geometric surface representation of the scene
Texturing*	Computes the texturing on the mesh and apply it as UV map

Through rigorous investigation in Meshroom’s online community, we found the most important parameters to look for are outlined in Table 24. Some parameters are obvious in their nature, such as Max Iteration affecting speed, File Type affecting Quality, or Downscale affecting both speed and quality. Some other parameters, like Force CPU Extraction are not obvious. Leaving this parameter unchecked allows Meshroom to perform its tasks with the help of the GPU. Another parameter is the Custom Bounding Box in the Meshing node. By default, the output mesh will contain the entire construction of the 3D scene, including the background. However, with the help of this node, we can “crop” to any mesh that resides outside of the bounding box. The last important parameter is the Unwrap Method. The default “basic” unwrap method, exports a multiple set of images each containing a region of the mesh. However, most traditional 3D rendering software, like [Blender](#) or [Cinema4D](#), prefer textures that are confined within one image. Fortunately, Meshroom offers this option by selecting LSCM as the Unwrap method.

Table 24: Key Parameters Affecting the Output Mesh

Node	Parameters	Value
FeatureExtraction	Force CPU Extraction	<input type="checkbox"/>
	Describer Density	Normal
StructureFromMotion	Localizer Max Ransac Iteration	4096
DepthMap	Downscale	1
Meshing	Custom Bounding Box	<input checked="" type="checkbox"/>
Texturing	Texture Side	8192
	Texture Downscale	2
	Texture File Type	png
	Unwrap Method	LSCM

With all this workflow setup in mind, it is time to hit the start-compute button and see the raw output result.

9.2.5.4 Raw Results

Running the current configuration of nodes has result of an overall processing time of 82 minutes. The final output textured 3D mesh, with over 4 million face count, is shown in Figure 45.



Figure 45: Raw Textured 3D Mesh (Meshroom Test)

In addition, Figure 46 outlines the processing time of each node in the workflow. We can see that the most expensive computations were related to the DepthMap, Meshing, and Texturing nodes, and while the 3D mesh quality is considered excellent, the face count being over 4 million faces is a significant overkill resulting in an unnecessarily large file size. The final file size is over 300MB. For such a simple and small sized object, the computation time and the output results are undesirable. There is, therefore, the need to optimize the workflow to reduce computation and processing power. An additional issue with the current iteration of the workflow is the fact that there is no direct or automated method of exporting the mesh and its material. We will have to look more into this.

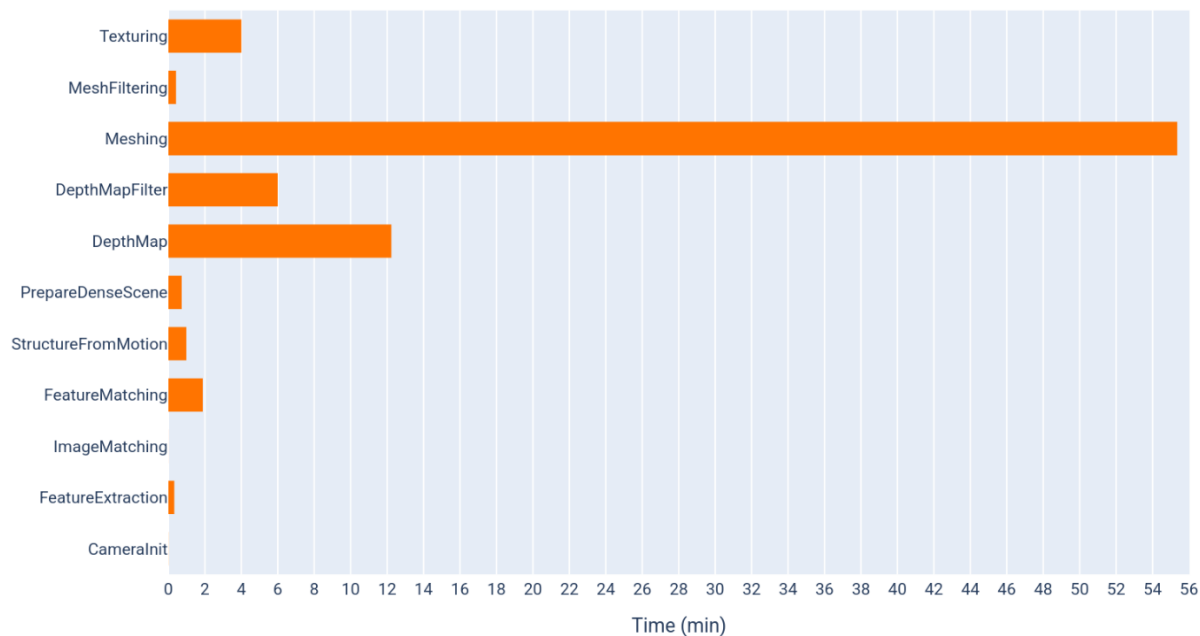


Figure 46: Nodes Processing Time (Unoptimized Pipeline)

9.2.5.5 Workflow & Result Optimization

There are a few changes we can make to the workflow pipeline that will reduce computation time and speed up the process without sacrificing quality. The new changes are mostly related to downscaling the processing mesh and textures. These changes exist within the DepthMap and Texturing nodes since they are high on the list of the most computationally expensive nodes. The Meshing node is more directly affected by the DepthMap parameters compared to its own parameters. In addition, to reduce output file size (i.e., 3D mesh face count), the “MeshDecimate” node is added right after the “MeshFiltering” node. The most important parameter of this node is “decimate factor”. A 0.5 decimate factor will reduce the face count by 50% percent. Lastly, to make it possible export the output 3D mesh and its material, the “Publish” node is inserted after the “Texturing” node, where the user can select the save destination folder. Table 25 summarizes the changes to the workflow.

Table 25: Changes Made to the Meshroom Workflow		
Change	Node	Description
Adjustment	DepthMap	downscaling 1 changed to 2
Adjustment	Texturing	Texture Downscaling 1 changed to 2
Node Addition	MeshDecimate	Reduces the mesh face count by a specified percentage (0.5).
Node Addition	Publish	Exports Mesh + Material to specified folder.

With all these changes in mind, a rerun of the same test has been done. The optimized output result is showcased in Figure 47. Despite the meshing being decimated to a final face count of 700K faces, the quality has not been affected. In fact, it seems that the quality has gone up, perhaps because the “MeshDecimate” node contains an algorithm to enhance quality as well as reducing face count. More impressive is the fact that the file size has now been reduced to 50MB (almost 80% size reduction). These new results are to be expected if we take a look at the new nodes processing time shown in Figure 48, with the orange and blue colors representing the unoptimized and optimized results, respectively.



Figure 47: Optimized Textured 3D Mesh (Meshroom Test)

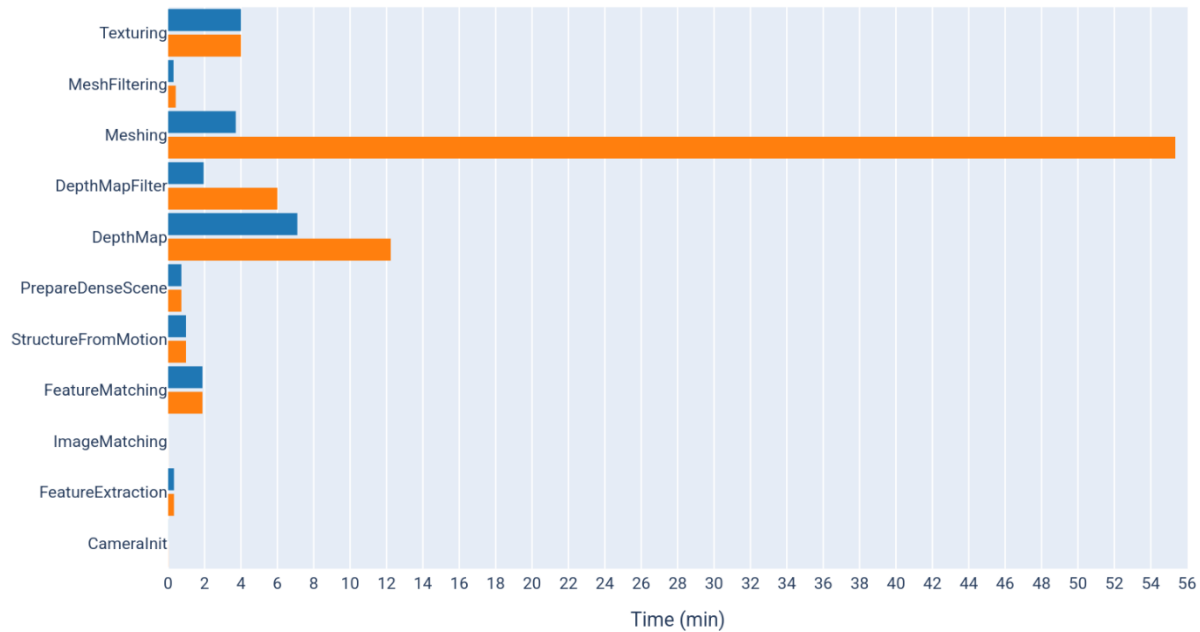


Figure 48: Nodes Processing Time (unoptimized vs optimized Pipeline)

All in all, with all this testing done we have arrived at the final working pipeline. We have validated that Meshroom will work excellent with the project and provide us with the tools necessary for 3D reconstruction. Of course, in the final implementation of the project the process would be automated. Converting what we have tested to an automated process is easy with sufficient knowledge with CLI commands. For now, it was enough to test Meshroom manually.

9.3 Overall Testing Plan

We have tested individual pieces of hardware and software, such as the functionality of the motors and movement of the mechanical parts, as well as the user interface graphics libraries and 3D reconstruction software. However, testing these aspects individually is not enough to ensure a functional project. These tests were to make sure we were on the right track with our decisions for parts and design methodology, and we have concluded that our hardware and software are working as we expected and will be ready to be integrated together into a final project.

When we design and build a full prototype, we will need to run tests to make sure that it functions as intended, and if any issues arise, figure out their origin. Our plan is **XXX**

9.4 Project Operation

The automated photogrammetry station is designed to automate the process of taking various pictures of a physical object from many angles and running them through a 3D reconstruction algorithm that will generate a 3D model of the object. This process may be automated, but it is still necessary for the user to understand what is happening to fully utilize the station.

9.4.1 The Photogrammetry Station Process

The process begins by placing the object you want to create a model of on the turntable. Upon plugging in the station to a wall outlet, the user interface will display a message to wait for setup to complete, which will allow the microcontrollers and peripherals to establish a connection. Then the screen will prompt the user to enter information regarding the Wi-Fi network they wish to connect to for transmitting the pictures to their personal computer to run the 3D reconstruction software. After a connection to the network has been made, the user will be prompted to select settings regarding the images that the camera will take, and the characteristics of the object that will impact the process, such as the height. After entering this information, or simply selecting a preset from a previous use, the station will calibrate the position of the camera by moving it up and down the stand and coming to rest in a neutral position pointed at the center of the object. The user will then be prompted to make sure the object is placed in the center of the turntable and the camera is correctly focused on it, before confirming the settings and beginning the automated process. From there, the station will move the camera and rotate the object to capture all the necessary pictures, and the pictures will be sent to the computer to be processed by the 3D reconstruction algorithm and produce a 3D model.

9.4.2 Recommended Conditions and Settings

XXX

- Intro
- Project Recommended Setting
 - Camera Setting
 - Background Environment
 - Lighting Conditions
- Project Operation Process (in dummy terms)
- Project Troubleshooting Manual

10 Overall Design & Construction

XXX

10.1 System's Integrated Schematic

- Overall schematic of everything
- Is it surface mount? through hole?

10.2 PCB Vendor & Assembly

- Where we are getting PCB made
- Where are we getting components from?
- How is it going to be assembled? In-house or by the manufacturer?
- Time Limitation (why we didn't test PCB)

10.3 Final Coding Plan

- Summary of everything that has been coded already.
- Summary of everything that has not been coded yet.
- Additional notes? Current Problems?

11 Administrative Content

This chapter discusses the management and accounting aspects of the project that relate more to the team and project overall than to technical specifics. Managing time and money is a very important part of the project that all team members should be aware of, as the team needs to work together to supply the budget to acquire all necessary parts, as well as ensure that all assignments are completed by their due dates. The responsibilities of each team member were based on their knowledge and experience, and the milestones for research and design were set by the team during weekly meetings.

11.1 Team Overview & Responsibilities

Our team consisted of three electrical engineering majors and one computer engineering major. Our electrical engineering majors also had experience with software, which helped to balance out the workload of software and hardware design. The greater number of electrical engineers with hardware experience helped us achieve our goals, because we took Junior Design at a time when the supply chain severely limited our access to parts, meaning we gained less experience with various aspects of hardware design and construction during that time. The mix of engineering types allowed us to have team members with knowledge of different areas from hardware to software, which provided the team with experience in different areas. All team members contributed a lot to the project with their varying knowledge and experience, which was essential for designing a project that involved software, hardware, and some mechanical design.

Task delegation is an essential part of any group project, as it not only ensures all of the necessary tasks are completed, but the organization of these tasks allows work to be done in parallel to stay on track. In our weekly team meetings, we discussed the work that needed to be done in the following week and decided who would be responsible for each task. Sometimes the team split into smaller groups comprised of members in charge of various aspects of the same project goal, such a group for designing the communication between the multiple computers, or a group in charge of the software that controlled a specific piece of hardware.

Alhusain Ali Al Badi is an electrical engineering major with the most experience in photogrammetry.

Ryan Dimmig is an electrical engineering major with experience microcontroller programming and user interface design. His responsibilities include researching communication between Arduino and Raspberry Pi, researching the touch screen and associated software, and doing software design related to the ATmega2560 and its peripherals. He has experience programming in Python, C, and GML.

Isaac Liljeros is a computer engineering major XXX

Nelson Vargas is an electrical engineering major XXX,

11.2 Project Milestones

Project milestones are incredibly important for keeping the team on track to complete the project in a timely manner. There are a few major due dates that are defined by the instructors and are therefore very rigid. Other minor milestones have been set by the group to keep everyone on track and make sure we can achieve what is necessary by the important dates. Table 26 shows a breakdown of every major milestone and many smaller ones to illustrate the progress that the team has made and intends to make. Due dates in [blue](#) were strict dates set by the instructors, others were more negotiable depending on how they affected other aspects of the project.

Table 26: Project Milestones					
#	Task	Start Date	End Date	Status	Responsible
Senior Design I					
Foothold Phase					
1	Brainstorm & Idea Selection	1/10/2023	1/24/2023	Complete	Group 8
2	Divide & Conquer Document (10 Pages)	1/24/2023	2/3/2023	Complete	Group 8
Research & Development					
3	3D Reconstruction Software Research & Demo	2/6/2023	2/17/2023	Complete	Alhusain
4	Pi & Arduino Communication Research	2/6/2023	2/17/2023	Complete	Ryan
5	Pi & Camera Interaction Research	2/6/2023	2/17/2023	Complete	Isaac
6	Arduino & Motor Connection Research	2/6/2023	2/17/2023	Complete	Nelson
7	Create Team Website	2/6/2023	2/22/2023	Complete	Alhusain
8	Mechanical Stand Research & Design	2/20/2023	3/3/2023	Complete	Alhusain
9	User Interface and Display Research	2/20/2023	3/3/2023	Complete	Ryan
10	Meshroom Server & DSLR Research	2/20/2023	3/3/2023	Complete	Isaac
11	Motor & Driver Research	2/20/2023	3/3/2023	Complete	Nelson
12	Miscellaneous Parts & Standards Research	3/6/2023	3/17/2023	Complete	Group 8
13	Draft Document (60 Pages)	2/3/2023	3/24/2023	Complete	Group 8
Design & Prototyping					
14	Update Team Website	3/27/2023	4/8/2023	Complete	Isaac

15	Major Software Design	3/27/2023	4/14/2023	Complete	Ryan & Isaac
16	Major Hardware Design	3/27/2023	4/14/2023	Complete	Alhusain & Nelson
17	Demo Video & Update Team Website	4/8/2023	4/17/2023	Complete	Group 8
18	Component & Project Testing Plan	4/17/2023	4/21/2023	Complete	Group 8
19	Final Document (120 Pages)	3/24/2023	4/25/2023	Complete	Group 8
Senior Design II					
Project Construction & Testing					
20	Build Project Components and Test All Aspects	3/27/2023	TBD	In Progress	Group 8
21	Build First Prototype	TBD	TBD	Preparation	Group 8
22	Test First Prototype	TBD	TBD		Group 8
23	Build Revised Prototype	TBD	TBD		Group 8
24	Test Revised Prototype	TBD	TBD		Group 8
25	Build Final Version	TBD	TBD		Group 8
26	Test Final Version	TBD	TBD		Group 8
Project Demonstration					
27	Team Practice Demo	TBD	TBD		Group 8
28	Final Project Demo	TBD	TBD		Group 8

11.3 Project Budget & Materials

Our project is funded by the members of the team, as it is not a sponsored project. This means that budgeting is incredibly important as we need to supply the parts with what we can afford. All team members have contributed various parts for development and the final product, which helped make the required budget easier to accommodate. Table 27 shows a bill of materials for all major components.

Table 27: Bill of Materials			
Component	Cost	Quantity	Total Cost

12 Project Summary & Conclusion

XX

13 Appendices

13.1 Copyrights

Still in progress ...

The screenshot shows the MISUMI website's contact form. The header includes the MISUMI logo, the tagline "Your Time, Our Priority", a search bar, and navigation links for Catalog, Account, Cart, and Help. The main navigation bar lists Products, Brands, Industries Served, Solution Center, and About Us, along with buttons for Request Quote and Place an Order. The contact form is a white modal with a close button (X) in the top right corner. It contains the following fields: First Name (Alhusain), Last Name (Al Badi), Company Name (Student at University of Central Florida), Company Email (hubadi@knights.ucf.edu), Address Line 1, Address Line 2, Phone Number, Country (United States), State (Florida), City, and Postal Code (32817). A message field contains the text: "Hello I am writing to request permission to use an images from your website, in a school research project. The images in question are linked at: https://us.misumi-ec.com/blog/strengths-limitations-belt-drive-vs-ball-screw-actuators/ I believe that this image would be a valuable addition to my project report. and I would be grateful for your permission to use it." Below the message field is a disclaimer: "The information you provide is for internal use only and will not be disclosed to any third party. If you have any questions or concerns, please refer to our Privacy Policy." A yellow SUBMIT button is at the bottom of the form. At the bottom of the page, there is a cookie consent banner with an "I Accept" button, and footer sections for Online Resources (Sitemap, How To Use, Bearing Finder) and About MISUMI (Career Opportunities, Terms and Conditions, Customer Privacy Policy).

MISUMI | Your Time, Our Priority

Enter Keyword or Part Number

Catalog Account Cart Help

Products Brands Industries Served Solution Center About Us Request Quote Place an Order

First Name* Alhusain

Last Name* Al Badi

Company Name* Student at University of Central Florida

Company Email* hubadi@knights.ucf.edu

Address Line 1

Address Line 2

Phone Number

Country* United States

State* Florida

City

Postal Code* 32817

Message

Hello

I am writing to request permission to use an images from your website, in a school research project. The images in question are linked at: <https://us.misumi-ec.com/blog/strengths-limitations-belt-drive-vs-ball-screw-actuators/>

I believe that this image would be a valuable addition to my project report. and I would be grateful for your permission to use it.

The information you provide is for internal use only and will not be disclosed to any third party. If you have any questions or concerns, please refer to our [Privacy Policy](#).

SUBMIT

MISUMI uses cookies to provide you services and improve this website. For more information please see our [Privacy Policy](#). By continuing to use this website or clicking "I Accept", you consent to our use of cookies. Cookie settings can be updated via your browser.

I Accept

Online Resources

Sitemap

How To Use

Bearing Finder

About MISUMI

Career Opportunities

Terms and Conditions

Customer Privacy Policy

13.2References

- [1] "Structure from Motion Overview," Mathworks, [Online]. Available: <https://www.mathworks.com/help/vision/ug/structure-from-motion.html>.
- [2] "Photogrammetry: Step-by-Step Guide and Software Comparison," formslab, [Online]. Available: <https://formlabs.com/blog/photogrammetry-guide-and-software-comparison/>.
- [3] "Drone Surveying Misconceptions: lidar vs. Photogrammetry," Propeller, 21 06 2021. [Online]. Available: <https://www.propelleraero.com/blog/drone-surveying-misconceptions-lidar-vs-photogrammetry/>.
- [4] "A Guide to Photogrammetry Photography," Online News Association , [Online]. Available: [https://journalists.org/resources/a-guide-to-photogrammetry-photography/#:~:text=When%20using%20a%20camera%20with,equivalent\)%20works%20well%20for%20objects..](https://journalists.org/resources/a-guide-to-photogrammetry-photography/#:~:text=When%20using%20a%20camera%20with,equivalent)%20works%20well%20for%20objects..)
- [5] D. P. L. Falkingham, "Photogrammetry: Does shooting RAW or JPG make a difference?," [Online]. Available: <https://peterfalkingham.com/2020/05/22/photogrammetry-does-shooting-raw-or-jpg-make-a-difference/>.
- [6] M. Jonathan Cohrs, "Capturing Images for Photogrammetry," R&D, [Online]. Available: <https://rd.nytimes.com/projects/capturing-images-for-photogrammetry/#:~:text=An%20ISO%20setting%20of%20100,9%2D13%20will%20work%20best..>
- [7] P. Team, "Megapixels and Image Quality – What Really Matters?," [Online]. Available: <https://www.pix-pro.com/blog/post/megapixels/#:~:text=For%20photogrammetry%20and%20many%20other,and%20the%20technique%20is%20there..>
- [8] "https://www.iqsdirectory.com/articles/linear-actuator.html," [Online]. Available: <https://www.iqsdirectory.com/articles/linear-actuator.html>.
- [9] "Ball screw or timing belt, when should you use what?," ROLLCO, 30 08 2022. [Online]. Available: <https://blog.rollco.eu/ball-screw-or-timing-belt-when-should-you-use-what/#:~:text=deviation%20per%20meter.-,Speed,in%20return%20slightly%20less%20accuracy..>
- [10] "iPhone 12 Technical Specifications," Apple Inc., [Online]. Available: <https://www.apple.com/iphone-12/specs/>. [Accessed 01 March 2023].
- [11] "Raspberry Pi Camera Module 3," The Raspberry Foundation, [Online]. Available: <https://www.raspberrypi.com/products/camera-module-3/>. [Accessed 1 March 2023].

- [12] "Raspberry Pi High Quality Camera," The Raspberry Foundation, [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-high-quality-camera/>. [Accessed 1 March 2023].
- [13] "64 MP Flagship Camera, Now for Flagship Pis," ArduCam, [Online]. Available: <https://www.arducam.com/64mp-ultra-high-res-camera-raspberry-pi/>. [Accessed 1 March 2023].
- [14] H. Aronoff, "WiFi standards explained: WiFi 4 vs WiFi 5 vs WiFi 6," Minim Inc., 29 July 2020. [Online]. Available: <https://www.minim.com/blog/wifi-4-vs-wifi-5-vs-wifi-6>. [Accessed 21 March 2023].
- [15] K. Ren, "Higher Speed How Fast Can It Be?," Bluetooth, 20 February 2017. [Online]. Available: <https://www.bluetooth.com/blog/exploring-bluetooth-5-how-fast-can-it-be/>. [Accessed 21 March 2023].
- [16] P. Whitener, "USB 2.0 vs. 3.0: Which Is Right for You?," USB Memory Direct, 10 June 2021. [Online]. Available: <https://www.usbmemorydirect.com/blog/usb-2-0-vs-3-0/>. [Accessed 21 March 2023].
- [17] "Raspberry Pi 3 Model B," Raspberry Foundation, [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b/>. [Accessed 21 March 2023].
- [18] "Raspberry Pi 4," Raspberry Foundation, [Online]. Available: <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>. [Accessed 1 March 2023].
- [19] "Orange Pi 3 LTS," Orange Pi, [Online]. Available: <http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/details/orange-pi-3-LTS.html>. [Accessed 21 March 2023].
- [20] P. Fromaget, "15 Best Operating Systems for Raspberry Pi (with pictures)," RASPBERRYTIPS, [Online]. Available: <https://raspberrytips.com/best-os-for-raspberry-pi/>. [Accessed 21 March 2023].
- [21] "802.11n-2009 - IEEE Standard for Information technology-- Local and metropolitan area networks-- Specific requirements-- Part 11: Wireless LAN Medium Access Control (MAC)and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughp," *IEEE Std 802.11n-2009 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, and IEEE Std 802.11w-2009)*, pp. 1-565, 2009.
- [22] Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips, Universal Serial Bus Specification, 2000.

- [23] "ISO/IEC 9899:2018(en) Information technology — Programming languages — C," Online Browsing Platform, 2018. [Online]. Available: <https://www.iso.org/obp/ui/#iso:std:iso-iec:9899:ed-4:v1:en>.
- [24] "International Standards Reference Chart," StayOnline , [Online]. Available: <https://www.stayonline.com/product-resources/reference-international-plugs.asp>.
- [25] "IEC 60130-10:1971 International Standard," International Electrotechnical Commission, [Online]. Available: <https://webstore.iec.ch/publication/823>.
- [26] "Nema 17 - 42 x 42mm," StepperOnline, [Online]. Available: <https://www.omc-stepperonline.com/nema-17-stepper-motor#:~:text=A%20Nema%2017%20stepper%20motor,%C2%B0%20version%20of%20the%20motor..>
- [27] Free Software Foundation, Inc, "GNU GENERAL PUBLIC LICENSE," 29 June 2007. [Online]. Available: <https://www.gnu.org/licenses/gpl-3.0.html>. [Accessed 20 March 2023].
- [28] Mozilla, "Mozilla Public License Version 2.0," [Online]. Available: <https://www.mozilla.org/en-US/MPL/2.0/>. [Accessed 21 March 2023].
- [29] "Run script on start-up with your Raspberry Pi," [Online]. Available: <https://raspberrypi-guide.github.io/programming/run-script-on-boot>. [Accessed 10 April 2023].
- [30] J. Woolsey, "Controlling An Arduino From A Raspberry Pi," 5 February 2020. [Online]. Available: <https://www.woolseyworkshop.com/2020/02/05/controlling-an-arduino-from-a-raspberry-pi/>. [Accessed 10 April 2023].
- [31] D. P. L. Falkingham, "Trying all the free Photogrammetry!," [Online]. Available: <https://peterfalkingham.com/2016/09/14/trying-all-the-free-photogrammetry/>.