The University of Central Florida
Department of Electrical Engineering and Computer Science
Dr. Samuel Richie
Senior Design I
Portable 3D Scanner Feasibility Study
Group 17
Members and Sponsor:

1. Jean Cestin (Computer Engineering)
2. Sergio Arciniegas (Computer Engineering)
3. Rayan Hamada (Computer Engineering)
4. John Paszynski (Electrical Engineering)

Table of Contents

1. Executive Summary

From the creation of the daguerreotype camera to the advanced camera's implemented into devices we keep in our pockets; cameras have constantly evolved allowing us to capture moments in time to forever remember. With the introduction of 3D camera's ability to capture objects in three dimensions for multiple points of views, technology will eventually advance and perfect it to the point 3D video's will be commonplace. Most 3D camera's, however, cost extremely large amounts of money, which is not ideal for commercial use. 3D cameras have many uses today. From fun photos of the family to 3D modeling, 3D cameras have many practical uses. From creating models for customers to view online, to models or references for video games, whether it be for work or education, 3D cameras will continue to be useful for a great number of applications. If 3D cameras become cheaper and cheaper, more people will purchase them, therefore increasing the demand to advance technology and improve the capabilities 3D cameras are lacking in.

Currently the market for scanners that can create a 3D model can cost easily up to $3000. Our group is motivated to find a way to build a reliable and efficient scanner that can also model 3D, but with a way lower cost. We feel that this is important as it can allow a lower barrier of entry for people that are interested in this area of study, which would be positive for the overall advancement of modeling. This would allow for advancements in space exploration, automated vehicles and even in certain medical practices with nanotechnology.

The creation of our scanner will therefore focus on ensuring a cost efficient device, with the aim of creating a portable and relatively accurate scanner. We will be utilizing ultrasonic technology. The sensors will be scanning for depth of the object in the field of view. This will allow for us to collect data at every coordinate in our sensors predtermined coordinate system. This will let the scanner continuously scan at every point, and we will use an algorithm to then process the data and create a STL file for the scan that took place. This image file will then be able to be accessed by the user via their computer. The scanner will have a battery in order for the scanner to work without a direct power source connected to the device, to further progress our portability goal.

2. Project Description

This section serves to provide the background of a few applications of the 3D scanner as well as objectives, goals, and motivations of the project. The goals and objectives within the feasibility project will be introduced and followed within the project requirements specifications as well.

2.1 Project Background

The portable 3D scanner can be used in a variety of different ways in both commercial and personal use. With current 3D scanners in the market costing thousands of dollars, the portable 3D scanner provides a low-cost easy to use device that can provide that user what they need at a reasonable cost.
Some of the applications of the portable 3D scanner are:

1. Hobbies: The portable 3D scanner is cheap enough that the average user may take 3D scans of objects for personal use as nearly all current devices on the market are not catered for the average user.
2. Manufacturing: The portable 3D scanner would be extremely useful for small businesses and/or startup companies to render images for commercial applications (e.g scanning a car for a 3D render at a body shop.)
3. Maintenance: The portability of the 3D scanner makes it an excellent device in the field of maintenance given its uses in finding leaks, dents, or other unwanted defects that could impair the functionality of the object(s).
4. Education: Given the absurd price point of market 3D scanners, the affordability of the portable 3D scanner gives school districts/universities the ability to provide multiple 3D scanners for educational purposes in bulk for the cost of today's devices.
5. Medical: The 3D scanner can be used in the medical field as a device to map a variety of different objects, such as dentures and crowns. By using the 3D scanner, medical offices can accurately map out important dental features for an accurate fit.

2.2 Objectives

In this project, our goal is to assess the practicality of creating a low cost, 3D camera scanner with sensors that is portable and accurate enough to be similar to those found on the market. We will attempt to approach the feasibility study by emphasizing the following objectives.

1. Have a camera that can 3d scan an object using ultrasonic sensors for depth.
2. Upload images to a website/interface.

3. Have an interface that a user can save and edit images, along with exporting them.

The first objective is important as it is the crux of the project, as we intend to create a camera that can 3d scan. The way we will approach is by using sensors. For now, there are a variety of ways to use sensors, as we could take an approach of using radio waves, or light such as lidar or x-rays using CT scanning. However, using multiple ultrasonic sensors, we can accurately measure the depth of an image captured using cameras.

The second objective is so that the user can make use of the images that have been scanned. We intend to create a webhost that the camera will be connected to via Wi-Fi. This webhost will then allow the user to be able to access the cameras memory and be able to let the user view their images.

The third objective is for editing the images that the user has taken, as well as interacting with them. The user can delete images, and if they want to export them as an image file to be used outside of the interface, they will have the option to.

Our ultimate objective is to be able to study the viability of a 3D scanner that is affordable relative to what already exists on the market but is up-to par in performance and reliability. Creating an interface will also be important as that is something that most 3D scanners that are in the market lack for their customers but is imperative for the user-experience.

2.2.1 Motivation

The motivation for this project is to be able to demonstrate and apply the knowledge that we have attained throughout our collegiate experience at the University of Central Florida. This project also gives us the opportunity to work in a group, delve into complex applications that will serve us for our careers and future. Ideally this will spark passion and creativity in our respective fields to make us better engineers. Additionally, 3d scanners are playing a much more important role in society and will be an essential part of the future. With artificial intelligence and the path towards automation, and bridging reality with technology, the role of having data to be recorded from the environment surrounding us would require sensors to capture and render images of it.

According to Tesla's autopilot, it currently uses eight cameras and 12 ultrasonic sensors. The cars are also equipped with forward radar which is used to read lane lines and be able to detect cars that are nearby. This is merely one example of the applications that exist with the emergence and importance of 3d scanning technology. But besides feeding into human laziness to want to avoid manually driving, the role in 3d scanning also plays a major role in the healthcare industry.
3D scanning gives a doctor the ability to create an exact image of their patient which will allow them to create a much more personalized form of treatment. 3D scanning allows the doctor to view and visualize the patient's body, in a way that would otherwise be impossible. And unlike MRI or Xray, 3d scanning is considered harmless for patients as the scanner is using photos, lights or a laser beam to create the image unlike the harsh radiation that comes from MRI’s and X Rays.

3D scanning does not just benefit the patient, but also the doctors as being able to accurately map the body will allow doctors to be better equipped and trained when in surgery. Being able to view the size and shape of a tumor before operating would provide the doctor with the ability to practice and learn how to best approach when they are in an actual surgery.
One of the most practical applications of 3d scanning in the health industry is for cosmetic dentistry. Scanning a patient's mouth will give the dentist or orthodontist an exact image of the mouth so that they can effectively operate and have a better approach for their treatment. Additionally, when it comes to improving smiles via veneers, digital scans for patients will be able to show them how the final product will look like. One of such scanners is called the intraoral scanner which can take thousands of pictures per second when scanning someone's teeth which is able to create a digital model of the patient's mouth.

Therefore, we can see that there are numerous uses of 3d scanners, and that similar technologies are essential for the advancement of civilization. LiDAR which is a unique way to measure and scan an area, is relevant to 3d scanning as it is one way to be able to 3d scan an area. There are over 75 LiDAR companies, 6 of which have gone public with a market valuation of approximately 14 billion dollars. These companies mostly focus in the autonomous vehicle niche currently. Amidst

heavy regulations that exist currently in the autonomous vehicle niche for these companies given the public space that they exist in, and massive scrutiny for human safety makes it a tough hurdle for these companies to continue to improve and grow with limited implementation of their technologies in the real world which would hinder revenue and overall growth. Therefore, it is important for these companies to continuously and expeditiously improve their technologies so that we can eventually reach a mass scale implementation of these systems which we know would be a massive improvement in our society and another benchmark for us humans.

So for now, these companies are also delved into AoT's which represent autonomy of "things", which can represent robots, machines etc. These companies are focusing on these areas as they are in either private or semi-private environments in which the public safety is not of much concern which would allow for a faster use and implementation of technologies, not having to wait on regulations to ease or for slow approval from the government. Problems that these companies are focusing on are mostly for addressing problems such as worker safety, productivity, costs, precision and quality. One example of these applications has been by the company caterpillar which has implemented autonomous equipment for their mining operations ranging from drilling equipment to haul trucks. Given the shortage of workers during the pandemic, having these automated systems/equipment allowed for the company to relatively keep up to demand with limited workforce which shows another positive example that autonomous technology can provide, which is that even though a pandemic occurred, and many people could not show up to work, the job could still get done. We view the advancement of autonomous systems as imperative, which is why we chose to undergo studying the feasibility of a portable 3d scanner as it provides a means to further improve the space of autonomous technology.

2.2.2 Design A

To design a portable, handheld camera scanner. The product will be utilizing one scope to capture and render the images and provide a 3d model of such image. The core motivation for this specific build is so that the product can be moved easily and is not set in a fixed location. This sort of building is very user friendly and allows for more versatility in applications.

2.2.3 Design B

This design is going to make use of studying the feasibility of utilizing 3 separate cameras, 2 of which are being used to create a 2-d image, while the 3rd camera is measuring the depth of the image by utilizing ultrasonic sensors. The cameras will be connected to a jetson nano while ultrasound sensors will be connected to an adafruit feather. This will restrict the camera's portability but will allow for a more robust camera that can do a certain extent accurately capture and render a 3d image.

2.2.4 Design C

This design is going to make use of 1 camera that will be able to provide imaging for the scan. The way that the system would be structured would be similar to an MRI scan, having the camera focus on an object that exists in a box. The camera would be controlled by a shutter remote which connects to an MCU, the camera is mounted to a structure that can hold it at an elevated and stable place aiming directly towards the box. The front end of the box will illuminate the object using high powered LED's and the object is rotated by utilizing a motor and an MCU. The collection of the images would then be reconstructed to create a 3d scan of the object.

The limitation of this design is that although the design would accomplish our goal of 3d scanning. The 3d scan would be pertained to the size of the box we would make; therefore we would be limited to scanning a small object and not of a large open area which we find more useful for practical purposes. Additionally, the box that we would need to situate the object, and basically have the camera attached to would bring the portability of the scanner down a lot, as it would not really be feasible to bring a large structure with a box around to scan. However, the novelty of using a design that mirrors that of how an MRI scan works did seem interesting and would be something we would keep in mind for our actual design when it comes to taking a creative approach to our system.

2.3 Requirements Specifications

- The device should operate as a portable, but not handheld, device and transmit data wirelessly to a PC over Wi-Fi 802.11ac/n.
- The device should be no longer than 10 inches in length.
- The device should not weigh more than 5 pounds.
- The device should not cost more than $200.
- The power supply must supply more than 10W of power given the power consumption of the components.
- The device must have an interactive button to capture images within 5 seconds as well as an I/O switch for power.
- The device will capture at a resolution greater than 640x480.
- The capture range should be within 1.0m – 5.0m.
- The device power will be powered via USB 2.0/USB 3.1.
- The sensor aspect ratio shall be at least 16:9.
- The power supply must last for a minimum of 2 hours before recharging.
- The device should be powered by a NiMH 7.2V battery.
- A buck converter is needed to step-down 5V to 3.3V for the microcontroller.
- Overall operating temperature needs to be below 70 °C.

- DDR4 RAM must be used over DDR3, since it has performance gains.
- ≥2GB GB RAM must be available on the SoC ARM board.
- Microcontroller must have an area of ≤ 1500 mm.
- 3 ultrasonic sensors must be utilized for triangulation and depth sensing.
- Ultrasonic sensors should operate within 4 meters.

2.4 Market Analysis

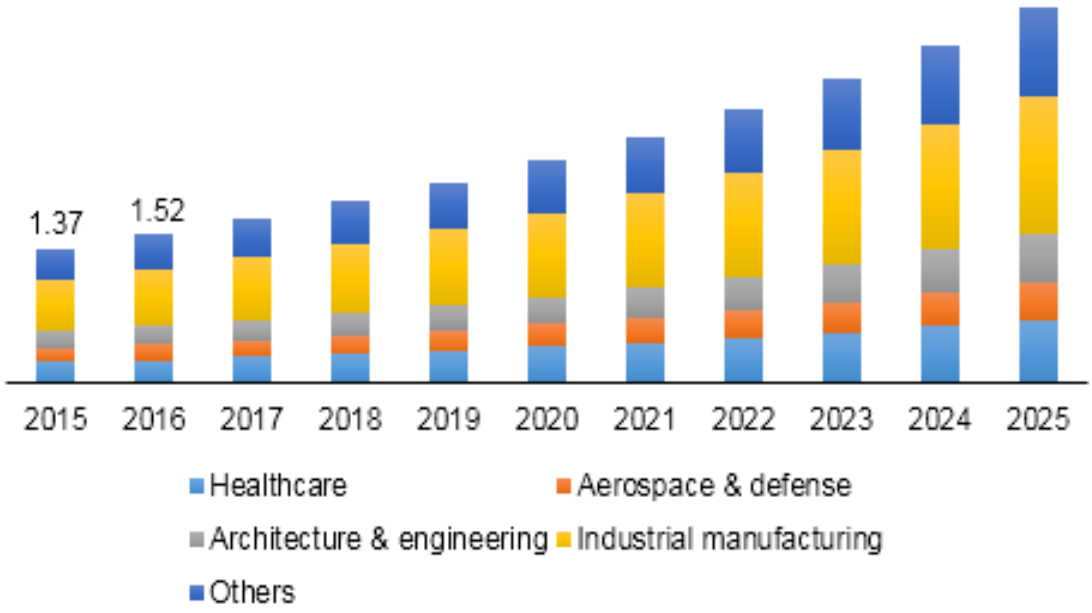

Figure 1: Global 3D scanner market size, by application 2015-2025 (USD Billion)

Currently the 3d scanner market is valued at about 920 million US dollars in 2021, the market is projected to reach 1.2 billion by 2026. This large growth is speculated to be driven by a focus and increase in research and development spending on 3d metrology, a larger focus on quality controls and higher demands for high productivity electronics manufacturing companies. The compound annual growth for the 3d scanner market is estimated to be 7.5% from 2021 to 2026 and North America is going to be the 2nd highest market during this forecast period. The overall 3d scanner market is segmented into 3 main markets, consisting of hardware, software and solutions, and services. In terms of an actual product, the market has segmented into classifying tripod mounted, fixed CMM based, portable

CMM based, and desktop as individual products that exist in the market relating to 3d scanning. These are considered distinct product models in the market, one of which that our group is attempting to take on after is the portable based. We find this one to be an interesting product as it would cater to more individuals that would care for having a 3d scanner on the go ready, rather than having a set based, or even an uncomfortable tripod mounted one. In terms of technology, the market has segmented into laser triangulation, laser pulse based, and laser phase-shift based.

The lasers that are playing a role all have different ways of interacting with the scan given its build and application therefore having a variety of laser application exists in the market. There are scanners in the market that are short ranged, medium ranged, and long ranged. Each will have its drawbacks when it comes to precision given its range, with that in mind our group will focus on a short to medium range product as it is one with the highest demand for usage both in the commercial and private applications compared to that of long ranged. For application, the market views quality control and inspection, virtual simulation, reverse engineering, autonomous driving and many others as applications for the 3d scanners. We view that there exists almost an infinite number of applications that can exist using 3d scanner as it is one way to interpret the world in a way that is as precise as possible. For the end use, the market is segmented into aerospace and defense, automotive, architecture and construction, medical, electronics, energy and power, artifacts and heritage preservation, mining and many others as end-use industries for the 3d scanner market. The more relevant and popular markets is automotive for now, as Teslas race to achieve full autonomous driving is the one that has the public attention for now, but the application of 3d scanners in basically every industry has a good and, with a fair amount of certainty a necessary need in them.

2.5 Marketing and Engineering Requirements

The House of Quality is a figure that defines the planning stage and how different requirements, such as customer requirements, relate to the contrasting methods on how engineers and industries can achieve those requirements presented. This can assist us on developing a relationship between engineering and marketing requirements. By using the house of quality, we can easily identify what is required for the project and how they correlate by using the varying degrees of correlation, which are positive, strong positive, negative, and strong negative. Using these correlations, we can as a group clear the way for decision making related to what is most ideal to focus on.

The benefits of a house of quality enables a customer or client to establish their needs within the process, identifying those needs, and ensuring that they are satisfied. Cost is one of the major constraints when analyzing engineering requirements since it represents what the client is willing to pay for the project or

design and has an upper limit that must not exceed. Since we are capped at a maximum price, the other requirements must be molded and selected to be kept within the budget and can ultimately limit the scope of the design. In essence, the lower the cost of the product the higher the demand will be.

Below is a figure of the house of quality for the 3D scanning device. The upward facing arrows show positive correlation while the downward facing arrows show negative correlation.

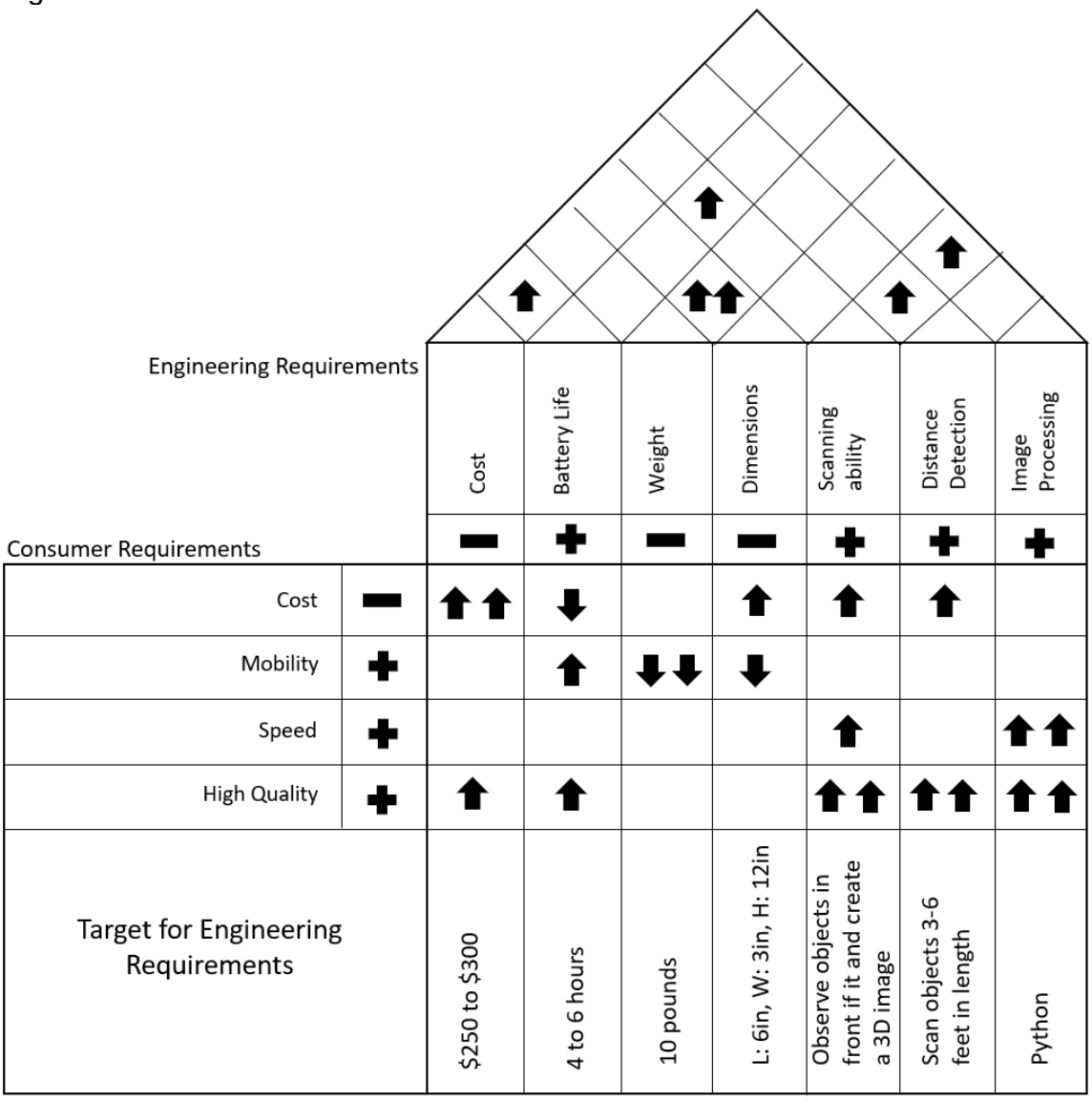

Figure 2: House of Quality

↑ = Positive correlation
↑↑ = Strong positive correlation
↓ = Negative correlation
↓↓ = Strong negative correlation
\+ = Increases the requirements
\- = Decreases the requirement

3. Research Related to Project

3.1 Existing Projects and Products

For related scanners that exist, a unique approach was using a CT scanner that works similarly to the way MRI scans are done. The CT scanner instead of using X-Rays, was using visible light incorporating a method called optical CT.

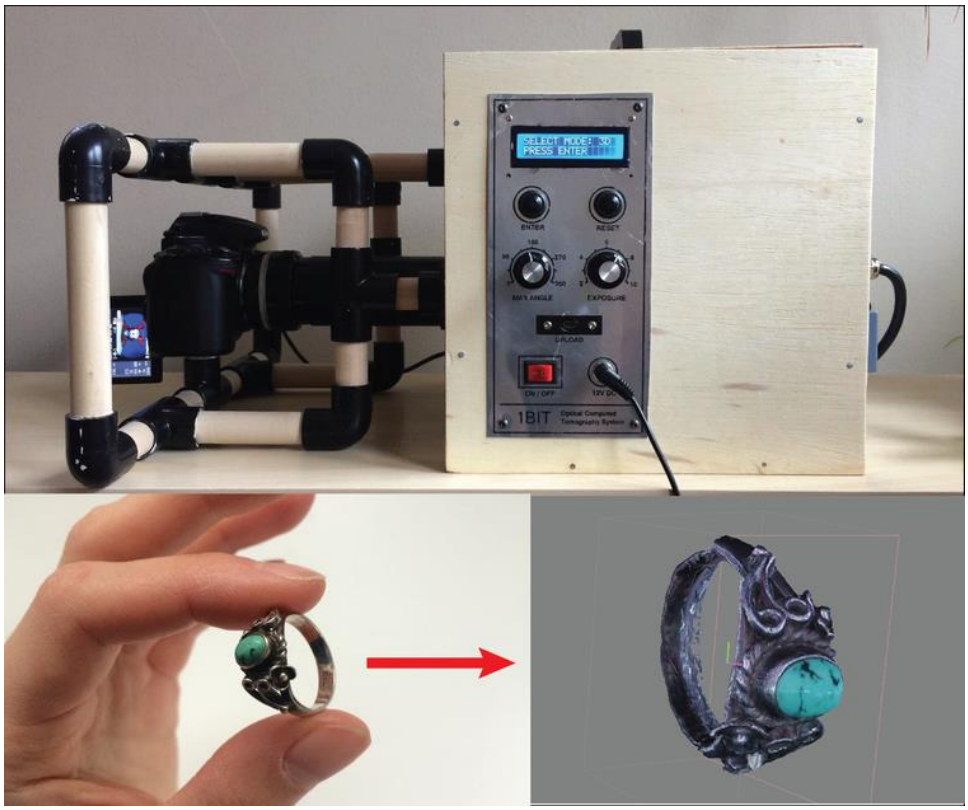

Figure 3: 3D CT Scanner

Figure 1 shows the way that the scanner is set up. An object is placed within a box, which is attached to a camera that is “scanning” the object. The camera was controlled using a shutter remote which has been connected to an Arduino nano. The object is illuminated using high-power LED arrays, therefore the light that has been collected by the camera would be dependent on how much has been absorbed by the object. The limitation of this design is that the scanner would only work for objects that can be constrained within the box. This is a huge setback for consumers that are interested in being able to scan objects that are larger than a 1 ft x 1ft box.

3.1.1 Existing Methods for Scanning

EINSCAN HX 3D SCANNER

The Einscan HX scanner is similar to what we are trying to accomplish, except for the fact that the Einscan HX uses Blue Lasers and LED Lights. With it's LED lights, it is able to scan 3d objects with an accuracy of 0.04 mm. Our project is different, we plan on testing whether or not it is feasible to create a 3d scanner that uses ultrasonic to output 3d images similar (but not exactly) to what the Einscan HX can produce.

The issue with the Einscan HX is it's price. The price of the scanner is at minimum $10000, which is way over the budget for many projects. It comes with software that in real time displays the object that is being scanned, with the scanner connected to it. For us it will be different, we plan on making it so that the device can scan the object, show what was scanned on the scanner, then be able to take it to the computer of the user's choice and use software we create to create the final 3D object. There is the possibility of changing this however depending on limitations of how powerful the microcontroller is. The processing of 3D images takes up a lot of RAM, so we might have to change our way of thinking.

Below is a figure showing the Einscan HX 3D scanner.

Figure 4: Einscan HX 3D scanner

The scanning accuracy of any 3D scanner is important. If the 3D image looks blocky and shows many inaccuracies, everyone would want to refund their scanner and the company selling them would go out of business. Accuracy is the most important thing any scanner should have. With the Einscan HX 3D scanner, it can scan objects with an accuracy of up to 0.04 mm. For normal 3D scanners, there are probably two things that influence accuracy. One would be the lasers, two would be the real time programming that takes in the data. For our project, we don't

have to worry about quick time scanning, so we mainly have to worry about the accuracy of the ultrasonic sensor we use. Accuracy is also important because the higher the accuracy the higher the detail of the 3D image.

The next important thing for a 3D scanner is it's Volumetric Accuracy. This meaning is the 3D scanner able to properly re-create the volume of the object being scanned. This is important because if the volumetric accuracy of the scanner is bad, it could show an image being more bendy or blocky than it actually is. The volumetric accuracy of the Einscan HX 3D scanner 0.04 mm + 0.06 mm/m. The +0.06mm/m meaning for every meter farther away from the object, the more inaccurate it will be. This will apply to our ultrasonic project as well, the farther away the object is from the ultrasonic sensor, the more inaccurate it will become.

Another important aspect of a 3D scanner that not many people realize until they actually hear it is it's scan speed. How many points of an object is a scanner able to scan in one second. The Einscan HX 3D scanner can scan an object at about 480,000 points per second, and is able to collect images at 55 frames per second. This will be extremely different in comparison to our project. The reason why is because our design will only use one to three ultrasonic sensors, we wouldn't be scanning in a grid using lasers like the Einscan HX scanner does. Therefore, we are unable to capture 3D images in a sense of capturing points on a grid so many times per second, if we did want to capture a grid like that, we would have to slowly go at each point one by one in order to measure the distances of the object.

A very important factor when scanning an object is how far away you are when scanning the object. The farther away you are, the less accurate the image will be, and if you're too far away, you wouldn't be able to scan the object at all. The Einscan HX 3D scanner is able to scan objects 470 millimeters away, which is about 18.5 inches. If you think back to the volumetric accuracy of the scanner, you'll see that it becomes less accurate with every meter, but we see that the Einscan HX 3D scanner wouldn't even work from not even a half meter away. The maximum inaccuracy that could be added to the volumetric accuracy of the scanner would be less than 3 millimeters. In our design, the only thing limiting our working distance is how far our ultrasonic sensor can scan objects. The ultrasonic sensor we will be using can detect objects ranging from 20 mm to 4500 mm, which is a much larger range than the Einscan HX 3D scanner.

The depth of field of an image is how well you can tell the difference between the subject of the image and the background of the image. For 3D scanners, it's the scanners ability to determine what is the image it's scanning and the background. The depth of field of the Einscan HX 3D scanner is 350 millimeters to 610 millimeters. This is important to us in our project because we need to make sure when an object is being scanned, it doesn't scan something far in the background.

A feature we will have to work on is the Field of View of the scanner. How wide of an angle is the scanner able to detect objects is important because the wider the angle the less the person has to scan. On the Einscan HX 3D scanner, there is a maximum field of view of 380 millimeters to 400 millimeters. The ultrasonic sensor is unable to view objects at an angle, so it has no field of view. We can remedy this by either adding more ultrasonic sensors, or add the ability to move the ultrasonic sensor.

Finally, the important part of an ultrasonic sensor is how it collects the data that turns into the 3D image. For the Einscan HX 3D scanner, the way it scans the object into the image is using 7 blue laser crosses. The way we will be scanning is by using an ultrasonic sensor, if it wasn't known before.

Scan Accuracy	0.04 mm
Volumetric Accuracy	0.04 mm+0.06 mm/m
Scan Speed	480,000 points/s 55 FPS
Working Distance	470 mm (~18.5 inches)
Depth of Field	350mm – 610mm (~13.78 inches - ~24.01 inches)
Max FOV	380mm*400mm (~14.96 inches * ~ 15.75 inches)
Light Source	7 Blue Laser Crosses

Table 1: Einscan HX 3D scanner features

METRASCAN 3D

This is another 3D scanner which uses a grid of blue lasers to scan the object to create a 3D image. The accuracy of this model is 0.025 mm and requires a computer to be connected to the device. The price of this scanner is around $20000, which is another high price for a 3D scanner. Our project will not be as portable as this scanner, we might have to place our scanner on a stand to make sure it stays leveled. This brings up the possibility of adding some functionality to our scanner.

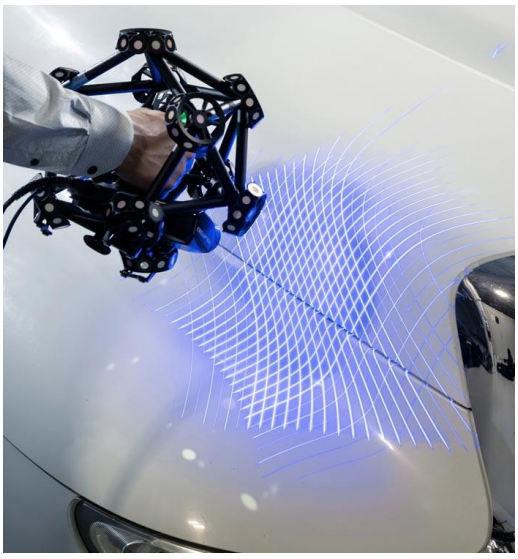

Figure 5: MetraScan 3D scanner

The MetraSCAN 3D sensor is vastly different from the Einscan scanner. In terms of Scan Accuracy, the MetraSCAN 3D sensor is able to scan with an accuracy of 0.025 millimeters. This is much better than the Einscan's 0.04 millimeter accuracy. This could have something to do with the number of lasers the MetraSCAN 3D sensor uses, as the MetraSCAN uses 15 blue laser crosses and the Einscan uses only 7.

The Volumetric Accuracy of the MetraSCAN 3D sensor is actually surprising. The volumetric accuracy of the MetraSCAN 3D sensor is 0.44 millimeters, which is worse than the Einscan, but the accuracy changes by only 0.015 millimeters per meter, which is way better than the Einscan. The initial accuracy might be worse because the quality of the lasers might be worse, but the change in accuracy might be better because there are multiple lasers.

The scanning speed of the MetraSCAN 3D sensor is amazing. The MetraSCAN 3D sensor can scan on average 1,800,000 points of an object per second. This is incredibly better than what the Einscan can do, and is something we will definitely be unable to do with our ultrasonic sensor. The great increase in scanning speed is again most likely due to the larger amount of lasers used in a grid.

The drawbacks with the MetraSCAN 3D sensor is the working distance. The distance at which the MetraSCAN 3D sensor is able to scan things is 300 millimeters. This may or may not matter depending on the user's personal preference. Again, our ultrasonic sensor would go way over that amount. We would probably not want to use the full range of the ultrasonic sensor, because saying that our scanner could scan things from 4500mm away could cause images to show inconsistency.

The Depth of Field of the MetraSCAN 3D sensor is also disappointing. The Depth of Field of the MetraSCAN 3D sensor is only 250 mm, which compared to the Einscan is not that great of a thing. This is possibly due to the many lasers it has, or the fact that the lasers might have a lower quality.

The Field of View of the MetraSCAN 3D sensor is understandably lower than the Einscan's FOV. The FOV of the MetraSCAN 3D sensor is 310 millimeters by 350 millimeters. If you think about it, the reason for this is most likely because the developers focused on a more condensed grid with many points so that the level of detail from the scanned object is much clearer.

The number of lasers the MetraSCAN 3D sensor has is more than what the Einscan has, which is 15 blue laser crosses instead of 7. This is most likely the reason why the MetraSCAN is much more expensive than the Einscan. With more lasers, you can have more detail in your objects, which is shown by the volumetric accuracy gain per meter.

One possible way to make sure our ultrasonic sensor captures an entire area instead of what is in front of it, is if we make it so that our camera and ultrasonic sensor move so that the distance and color of the object is scanned in a grid. The problem with this however would be that it might cause inaccuracies. One thing that this scanner does not do is that it does not scan the colors of the object. This will be a stretch goal in our project if we are able discover whether it is feasible to create a 3D scanner with an ultrasonic sensor.

Scan Accuracy	0.025 mm
Volumetric Accuracy	0.044+0.015 mm/m
Scan Speed	1,800,000 points/s
Working Distance	300 mm (~11.8 inches)
Depth of Field	250 mm (~9.8 inches)
Max FOV	310mm*350mm (~12.2 inches * ~ 13.8 inches)
Light Source	15 Blue Laser Crosses

Table 2: Metrascan 3D scanner features

XBOX KINECT SENSOR

In early November of 2010, Microsoft released a new device that revolutionized a different way to play video games. They released a device called the Kinect, which was an accessory to the Xbox 360, which uses an advanced camera system in order to track the users movements, which is then used in games that involve movement. For example, Star Wars games where you move your hands around to use a lightsaber, or dancing games which require your hands to be in certain

places to score points. The Kinect was then redesigned for the Xbox One, with an upgraded model.

The system can track the user's distance from the sensor in real time, and is able to track movement. This led to the idea that the Kinect is able to create 3D images. With the 360 version of the Kinect, a third party software was created to snap a 3D photo of an area. This revelation led to Microsoft developing their own app. This app only works with the Xbox One version of the Kinect, and with it, you can in a sense take the Kinect, connect it to your PC, and circle around an object to be turned into a 3D image. This method is only successful on non-reflective objects (ie, it would not work on clear glass).

You can connect the Kinect to your PC or laptop, you can download software developed by Microsoft can use the Kinect to scan objects into a 3D image which can be viewed in another one of Microsoft's app's, called 3D Viewer. This is similar to what our original project was going to look like. Using an Xbox 360 Kinect, we planned on creating a portable device that could scan the object and store it in the device to be processed and formatted so that the computer could view the 3D image, or that the scanner would take the pictures and the computer would format the 3D image. These were projects already completed by others in senior design, and the fact that most of the project would be done for us with the Kinect made us decide we should change our direction in the project.

There are no specifications for the Kinect in scanner mode, mainly because the device is not mainly used for entertainment purposes, but it's fairly safe to assume that the Kinect isn't as accurate as the previously mentioned 3D scanners.

3.2 Relevant Technologies

3.2.1 Microcontrollers

So far, only three microcontrollers have caught our eye. The Adafruit Feather M0, the Seeeduino Cortex M0+, and the Arduino Zero. The Adafruit Feather M0 is aptly named for its extremely small size. This will be useful when it comes to fitting it into our design. The Seeeduino Cortex M0+ can handle many complex calculations with high speeds for data transfer. The high transfer speed for data is nice, but we do not need a microcontroller that handles complex calculations. The Arduino Zero is a very powerful microcontroller, useful for many projects involving robotics and automation. This might be way over what we need in terms of specs for the microcontroller. Below are the pictures of the three Microcontrollers that catch our eyes for the project.

Microcontrollers are computer chips mainly used for embedded systems. Their functions are limited, much weaker than a proper computer CPU, but for the purpose of smaller jobs it is much more cost effective to use a microcontroller rather than an actual CPU. The microcontroller we use is the STM32F103T6, which operates at 3.3V and at a temperature of up to 85 degrees Celsius. It has a good amount of I2C, SPI, UART, and USB capabilities, which we can integrate into our PCB.

3.2.2 Image Processing Software

Depending on how we collect the data from the ultrasonic sensors and camera, we can create a program that creates a *.stl or *.obj file using the distances observed at multiple points to create vertices in the file so that when it is opened in a 3D image viewer, we get a somewhat accurate shape of the original image. And if we are feasibly able to complete that, we can then move on to adding color from the picture to the 3D image.

One way we might be able to handle the image processing is through AI. The NVIDIA Jetson Nano Developer Kit (which is the development kit we will most likely be using) specializes in AI programming, so it is not a bad idea to use AI to process the image. Some people in the group already know a little bit of python and its usage in AI, so it is not a foreign concept.

Another way we might be able to handle the image processing is through a lot of Java or C programming. The format of *.stl files or *.obj files are not a secret.

With this knowledge and the data from the ultrasonic sensors, we could create a program that creates and hardcodes these values into a file of our choice, which would in the end be the finalized file that the user would use.

3.2.3 Camera Technologies

The camera that we use will mainly be a part of our stretch goal, our stretch goal being if we are able to feasibly create a 3D image using ultrasonic sensors, we will then try to color the 3D image to further increase the productivity of our project.

There are many points we must consider when choosing the correct camera for our board. There is the angle that the camera will be able to view when taking the picture. It must match the same angle of what the ultrasonic sensors detect. There is the output of the camera, which must be able to output the image immediately to the development board. The resolution of the camera has to be good as well, we wouldn’t want a low-resolution image making the final 3D image fuzzy. The size of the camera is also a very important factor, because the more weight/bulk there is to the camera, the worse it will be for portability.

3.2.4 Battery Technologies

The types of batteries that are commonly used for such similar electronic projects would be the nickel-metal hydride batteries, NiMH. These are popular because they are rechargeable and come in standard sizes similar to alkali batteries. At 1.25V per cell, they provide less voltage than alkaline batteries, though slightly more than nickel cadmium alternatives. These batteries are a good alternative to alkaline batteries in most situations, since they have higher power density with similar sizes. A downside of nickel metal hydride batteries is that they have a high self-discharge rate and do not last as long as other technologies. Nickel-metal hydride batteries are like nickel-cadmium in that they both are chemically composed of nickel oxide hydroxide, apart from the former having higher energy densities due to a different implementation of the chemicals.

Comparable to the Nickel-Metal Hydride battery, Nickel-Cadmium is an older rechargeable technology that still sees a few applications. Even though it has a lower power density than NiMH, they are cheaper and can discharge slower when not in use. These batteries are used in implementations where performance is second thought to price. An advantage of this battery is that it can be quickly charged without harming the chemical nature of it. A disadvantage to it being rechargeable is that it suffers from a memory effect. A memory effect is when the battery is constantly charged without discharging completely to zero. Fully charging and discharging is needed to reduce the impact of memory effect, which can cause crystals to develop on the battery plates.

Another technology that is the most widely used is the lead acid battery. Due to the incredible ease of manufacture, these batteries are readily available at low cost. Robotics and other electronics which draw lots of power and don’t consider weight as a constraint require these types of batteries. The low power density of these batteries tends to cause them to be manufactured as large, heavy, and bulky. This is a driving limitation to consider when selecting the right battery for our project. Due to being highly toxic to humans, lead acid batteries are the most recycled battery to prevent lead emissions from entering the environment. Furthermore, the applications of these batteries tend to power larger pieces of hardware such as electric vehicles, robotics, and emergency backup power.

Alkaline batteries are highly available with a wide range of different types to match the needs of someone seeking a battery source. The longer shelf life, safer, and higher power density compared to NiMH and NiCd gives them an advantage. Although they provide this, they constantly need to be replaced due to being a one-time use battery. This will ultimately accumulate cost for those who employ it to their design. If an alkaline battery does not get replaced, it will slowly self-discharge and eventually the battery will start leaking potassium hydroxide over time. Potassium hydroxide is a compound that will cause irritation to humans.

The Lithium-ion battery is a much newer technology available in the market compared to the previous options. They are lightweight and fantastic for consumer electronic devices that require very high power densities. These batteries are not negatively affected by high discharge rates, memory effect, nor high cycle count. Although these batteries provide the right necessities for consumer electronics, they are highly volatile chemically and must require circuitry to prevent them from exploding or causing a fire, if it is charged too quickly. Another disadvantage to Lithium-ion is that the most common voltages available are around 3.6V or 7.2V, which electronics need to be designed around rather than having the battery be designed for the electronics.

The power capacity of a battery is measured by how much energy is being stored within the battery itself. Commonly, the power capacity is denoted in Watt-hours or Wh.

	Lead Acid	Alkaline	Lithium-ion	Nickel-Metal Hydride (NiMH)	Nickel-Cadmium (NiCd)
Advantage	Inexpensive and easy to manufacture. Reliable and slow discharge.	High energy density. Can be recycled easier. Inexpensive.	Lightweight and high energy. Most efficient.	No toxic metals. Higher density than Nickel-Cadmium.	Inexpensive and easily stored.
Disadvantage	Low energy density. Environmentally not suitable.	Heavier. Leak over time. High internal resistance.	More expensive. Cannot be recycled.	Low charge cycle and high production cost.	Low energy density and toxic metals.

Table 3: Battery Type Advantages and Disadvantages

Above is a table that summarizes the advantages and disadvantages of each battery technology type.

3.2.5 Battery Regulation

Voltage regulators are important for circuitry in that they regulate the voltage output and throughput of a battery source. These are the most common pieces of electrical components since most electronics require them in some shape or form. Voltage regulators are divided into two classes, step-down and step-up. Step-up voltage regulators take the output voltage of a battery and translates it into a higher voltage at the output of the regulator. A step-down regulator does the contrary as it shifts the voltage to a lower one that may be required for the device.

For a design, there are two classes of voltage regulators that are on the market: linear and switching regulators. Linear regulators are essentially the cheap, non-complex, and noise free option. These types of regulators are only capable of stepping down a voltage to a desired one. However, stepping down a voltage with a linear regulator is not very efficient since they have low power efficiency. For example, if you have a 10V to 4V linear voltage regulator, then it will take the input at 10V and output it at 4V. This means that 6V will be wasted in the form of heat, which can become an issue if the project requires maintaining thermal stability. On the contrary, the other type of regulator is the switching regulator. The switching regulator is a more complex and expensive option that can not only step-down but also step-up its voltage. Linear regulators are the most conscious choice due to being cheaper and much simpler than its counterpart. However, if it's required to step up a voltage or if the power dissipation is draining the battery source quicker within the set of electronics, then a switching regulator is the right choice.

Another technology in the voltage regulation field is the buck converter. A buck converter is a DC-to-DC converter that converts high voltage to low voltage, like a linear voltage regulator and makes for a great alternative to the voltage regulator. Unlike voltage regulators, buck convertors do not need heat sinks and therefore in portable instances, it would limit the amount of excess heat generated within your circuitry. Some buck convertors offer a way to change the output voltage on the circuit using a potentiometer.

3.2.6 Wireless Transmission

Wireless transmission is when two or more devices transfer information through a medium such as the air. There are quite a few wireless transmission technologies; from radio waves to Bluetooth and to Wi-Fi. Radio communication is the most widely used technology that allows communication from a few feet to millions of miles away. Radio waves are on the electromagnetic radiation spectrum (EM) that encompasses microwaves, infrared, visible light, ultraviolet, radio waves, x-rays, and gamma rays. Each designation is divided into order of wavelength and energy. Radio waves are the longest wavelengths within this spectrum, but also has the lowest frequency. These waves are further classified in bands, which are appointed by the frequency of the wave. The extremely low frequency of radio waves below a few kilohertz can maximize the distance in communication as these waves travel hundreds of miles. On the contrary, extremely high frequency band in the gigahertz can only be in the order of a few millimeters, which can limit the application of it in certain instances. However, although they have short wavelengths, they allow high-bandwidth transmission between two fixed locations within a very small range. The higher frequency bands are typically used with frequency modulation, which is widely used in mobile telephone communications, and can deliver better quality signal as these waves are not affected by

environmental effects when travelling. This is due to the fact that FM waves are sent with a constant amplitude that won't be altered.

Another form of wireless transmission is Wi-Fi. Wi-Fi is considered to be a wireless LAN connection between computers or mobile devices. LAN is a local area network, which is a limited computer network within a home, school, or office and does not interconnect to the internet. Wi-Fi utilizes the IEEE 802.11 standard for communicating and use radio waves to transmit the necessary bandwidth between a client and host. Wi-Fi initially was used in the 2.4 GHz frequency range, however in recent years this has changed to support 5GHz and eventually 6GHz, which transports higher throughput than the former frequency ranges. The larger the spectrum, the more channels it can support, therefore less overlap in communications in crowded, high traffic areas. The key benefits of Wi-Fi 6GHz include higher data rates, increased capacity, and improved power efficiency [2]. Wi-Fi is widely used compared to a decade and a half ago and is only improving day by day, and for that reason it is a highly supported technology with thousands of products to implement this technology within a project.

An additional short range wireless communication technology is Bluetooth. Bluetooth is similar to Wi-Fi in that they are both radio waves that produce high-speed transfer of data between devices. While Wi-Fi connects to the internet, Bluetooth does not and is limited to either one or a couple of devices connected simultaneously depending on the version of Bluetooth module the device carries. A shortcoming of Bluetooth is that the range and speed is not to the capabilities of Wi-Fi, however it is much simpler to operate and connect to than the latter as it only requires an adapter on each end. A benefit of this technology is that it can transfer small amounts of data while limiting the usage of energy from the battery source. Therefore, compared to Wi-Fi it is much more efficient to operate in a system where battery life is an important aspect of the design.

With the NVIDIA Jetson, there are many USB ports in which either a Bluetooth adapter or a Wi-Fi adapter can be added. This might give us the ability to transfer the 3D image without the need to connect the device to your computer with a wire or must transfer the 3D image via SD card. This may or may not be a very difficult task, so this would be another stretch goal.

Adding Wi-Fi to the development board is possible with the Edimax 2-in-1 Wi-Fi and Bluetooth 4.0 adapter. It is an extremely small device, so it would not make the device too bulky, and would not block any of the other USB ports.

3.2.7 Ultrasonic Sensors

Ultrasonic sensing is one of many techniques used to sense proximity and range. It is one of the most reliable forms of sensing as it is insensitive to environmental influences such as dust, light, smoke, and can even detect transparent objects for applications such as water level management and detecting glass objects. Light or photoelectric sensors struggle with detecting transparent surfaces or highly reflective metallic surfaces and therefore ultrasonic sensors have an immediate advantage. A disadvantage of using ultrasonic sensors is that they don't operate well in environments which have flucuating variables. A shift in temperature or wind change can affect the measurement as these variables change the medium in which the sound waves travel though. These shifts in variables can affect the speed of the sound waves as higher temperature environments will have different medium densities. Ultrasonic sensors work by generating sound waves by vibrating a transducer when current is applied, like a speaker, which are reflected back, if there is an object within the operating range of the sensor. These sound waves are used to calculate and measure the distance using the time the signal took to travel. By using the following formula:

$$\text{Distance} = \text{Speed} * \text{Time}$$

And knowing the speed of sound through a medium, which would be air at sea level as the most common application at 343 m/s, we can code an accurate calculation for the distance of an object. The ultrasonic sensor can use this to detect accurately the distance as long as impediments such as snow, condensation from humidity, and dust. Another type of object that is not detectable by ultrasound are objects that have sound absorbing materials that absorb the energy of the traveling wave, such as fabric. Larger, more flat objects are easier to detect at maximum operating range, while smaller objects will not reflect a strong echo signal back to the sensor at longer ranges. On the contrary, objects closer to the minimum operating range of the sensor must be far enough away to not be within the deadband zone. The deadband is a similar concept to blind spots when driving vehicles. This zone is where objects cannot be accurately detected by the sensor and the size of this zone varies between models of sensors. When objects are placed too close to the sensor, the sensor can miss detection of the first echo sound wave while still registering the sequential waves that are being bounced back.

An ultrasonic sensor is a device that uses ultrasonic sound waves to determine the distance to an item. A transducer is used in an ultrasonic sensor to emit and receive ultrasonic pulses that communicate information about the proximity of an item. High-frequency sound waves reverberate off of surfaces, creating different echo patterns.

Ultrasonic sensors function by emitting a sound wave that is above the human hearing range. The sensor's transducer functions as a microphone, receiving and

transmitting ultrasonic sound. To deliver a pulse and receive the echo, our ultrasonic sensors, like many others, employ a single transducer. The sensor measures the time between delivering and receiving an ultrasonic pulse to estimate the distance to a target.

This module's operation is straightforward. It emits a 40kHz ultrasonic pulse that travels through the air and bounces back to the sensor if it encounters an obstruction or item. The distance may be estimated by multiplying the transit time by the sound speed. Ultrasonic sensors are an excellent choice for detecting clear objects. Because of the intended translucence, applications that employ infrared sensors, for example, suffer with this use case for liquid level sensing.
Ultrasonic sensors identify things regardless of colour, surface, or substance for presence detection (unless the material is very soft like wool, as it would absorb sound.)

Ultrasonic sensors are a trustworthy solution for detecting translucent and other things where optical technology may fail. For many years, ultrasonic sensors have been a common tool for measuring distance and detecting objects. The ultrasonic sensor's main functioning concept is detecting the time it takes for its produced ultrasonic pulse of sound to bounce off an object and return, hence the name. Due to their capabilities and adaptability in a range of designs, ultrasonic sensors are finding new applications in the market for autonomous robots, automobiles, and other related devices.

Before choosing an ultrasonic sensor for a specific application, it's crucial to understand how they work and how to choose the right transmitter, receiver, or transceiver based on critical performance parameters. Ultrasonic sensors have proved to be a go-to proximity sensor option owing to their numerous benefits, but they do have certain limits, just like any sensor technology, that may not make them the ideal choice in every situation.
"The Basics of Ultrasonic Sensors," a CUI InsightsTM blog article, goes through these critical subjects in further depth, including how an ultrasonic sensor works, key parameters, and benefits and drawbacks.

3.3 Strategic Components and Part Selections

3.3.1 Development Board

The role of the development board has two major functions. The first function is to receive the data from the microcontroller to measure the distance of three points on an object using an equal number of ultrasound sensors and to capture an image using multiple cameras. The processor should be powerful enough to tie each image capture to the sensor data and operate multiple accessories. The microcontroller will then upload the data locally to an application created to view the scanned object and will use a programming language that is familiar to the

group. An FPGA board will be the best solution for the application requirements, since using the serial output of a camera module, it provides frame-accurate synchronization. However, the time and resources required to develop FPGA firmware and the cost of hardware offsets the benefits. Given the high cost and high development time of an FPGA board, some frame synchronization error will have to be tolerated, thus an embedded ARM or x86 embedded computer is the most logical choice for this project.

NVIDIA JETSON NANO DEVELOPER KIT B01

The NVIDIA Jetson Nano provides a solid out of box development kit that is popular for machine learning or, in our case, image processing. Unlike more standard microcontrollers such as the TI MSP430 variants, the Jetson Nano has multiple ports. The nano supports 2-lane CSI interface flex connectors that can support cameras that support the power from the developer kit. The multiple ports include a 40-pin expansion header needed to connect multiple ultrasound sensors and USB ports capable of plugging in generic USB cameras, if we decide to go that route instead of using an embedded MIPI CSI-2 camera using the respectful connector provided on the device. The Jetson Nano also includes an 802.11ac wireless USB adapter, which is necessary for uploading the data remotely. If a remote upload fails due to internet connectivity, then the MicroSD card slot can provide local backup for data transfer retries or viewing locally on the device. Not only does the board have a processor, but it also has a dedicated graphics processing unit that is able to take advantage of NVIDIA's accelerated image processing. Thus, it lets us process all of our imaging right on the board locally and seamlessly. Since the Jetson Nano is built with the Nvidia Maxwell (128x CUDA cores), it has the most powerful GPU chip out of the bunch. Given the price ($99.00) to performance ratio, the Nvidia Jetson Nano Developer Kit is an excellent candidate.

NVIDIA JETSON NANO 2GB DEVELOPER KIT

Due to supply issues preventing from purchasing the NVIDIA Jetson Nano Developer Kit B01 in time for the testing portion of the project, we must compare it to the NVIDIA Jetson 2GB kit. The Jetson Nano is a perfect A.I focused microcomputer; similar to the Raspberry Pi and can be used in other applications that require more power than is provided by microcontrollers on the market. While the Jetson Nano B01 has 4GB of LPDDR4 ram, the Jetson Nano 2GB model features just half that. While it can't perform more intensive tasks compared to the B01, it has enough power to provide just what we are looking for in our design and will not set us back in terms of computational power. The Jetson Nano 2GB is powered by a quad core ARM Cortex-A57 processor and a 128 core Maxwell architecture processor for graphical computations. The Jetson 2GB also features

a WiFi adapter within the box that can be plugged in via USB 3.0, which is a nice addition. Another difference between these two models is that the 2GB Nano cannot be powered by a proprietary barrel connector, unlike the B01 that provides one that feeds 5V/4A of power to the board. This is a major drawback as it can complicate power supply design as it is only powered via USB C. Not only was the barrel jack removed from this board, but the power supply jumper was removed as well. The USB-C port requires a USB-C charger than can deliver at least 5V/3A of power to the device without risk of powering down. While the B01 has 2 MIPI-CSI camera adapters, this version only features one, which is not ideal for stereo cameras which require two ports as each camera lens requires its own dedicated channel. Some accessory headers on this board have been moved and unpopulated under the microSD slot, therefore its required to solder on pin headers, if these ports are required to be used. To reduce cost at manufacturing, NVIDIA removed three USB 3.0 ports and transitioned two of the removed ports to USB 2.0. In addition, the DisplayPort connector was axed as a feature, although it does still come with an HDMI port for video displays. At $59.99 USD, the Jetson Nano 2GB is 40 dollars less expensive than its counterpart.

NANOPI M4V2

Another embedded ARM board is the NanoPi M4V2, which is based on the Rockchip RK3399 and has a similar footprint to the Raspberry Pi 3 Model B+. This SoC ARM board has plenty of features that make it a good platform for quick deployment for numerous applications. The onboard 2.4G and 5G wireless modules are excellent for the purpose of transferring the data, since they're provided on chip. Unlike the Nvidia Jetson and MaaXBoard Mini, the NanoPi provides dual MIPI-CSI channels for simultaneous camera input that makes it ideal for 3D scanning. While the MaaXBoard mini has 4 USB 2.0 ports, the NanoPi provides the same quantity of ports, however they are higher in bandwidth and speed as these ports are USB 3.0. USB 3.0 provides 625MB/s transfer speed while its predecessor only transfers at 60MB/s, which is a significant performance increase. The added vision processing unit, VPU, is able to decode 4K H265/H264 video at 60fps.

As well as a 1Gbps Ethernet port, this board supplies an HDMI 2.0 Type A connection, a 3.5mm jack, and a single USB Type-C port that can be used for data transfer. Not only that, but the NanoPi has a PCIe x2 interface so more USB 2.0 ports, SATA ports, or an NVMe SSD is possible through added modules. The power consumption of this embedded computer is 5V and 3A through a power adapter. The NanoPi supports Ubuntu, Lubuntu, Android 8.1/10, therefore offering a rich software development environment. A negative to the NanoPi M4V2 is the power consumption. With all CPU cores active, the board consumes 10W of power with peripheral devices connected and can easily exceed the 2A limit. In addition,

the RK3399 chipset is not a super fast chip compared to the likes of the Tegra X1 found on the Jetson Nano.

MAAXBOARD MINI

An alternative development board that can be considered is the MaaXBoard mini. The MaaXBoard is an embedded computer that is mostly in audio and video projects, however it is powerful enough for multi-use. This board features 2GB of DDR4 memory and a Quad Arm Cortex-A43, which is equivalent to a very low-end desktop computer, but powerful single board computer for embedded applications. Also, there is an expansion connector that allows us to install Raspberry Pi HAT modules, if so desired. UART, I2C, 40 GPIO pins for external devices, and SPI are all included on the base board. The MaaXBoard Mini uses a USB Type C interface to input +5V. While this port powers the system, it does not support data communication for applications. The MaaxBoard rather provides 4 USB 2.0 connectors for high-speed data transfer. Similar to the Jetson Nano, this board features a Wi-Fi module for 2.4gHz and 5gHz bands as well as Bluetooth version 4.2. A micro-SD card slot is available to initiate code as well as provide external memory as expanded storage. Unlike the Jetson Nano, the MaaXBoard boasts a 30 pin FPC connector that is available for displaying high-definition images or video through MIPI-DSI. Although not as speedy as the Jetson Nano, the MaaXBoard provides similar performance with some added features that can be utilized. Unlike the former, MaaXBoard provides an onboard MCU with the Arm Cortex-M4, Bluetooth 4.2, and support for Android 9.0. Although the MaaXBoard Mini does not have a MIPI CSI-2 camera interface, it still provides a MIPI-CSI port as well as MIPI-DSI. A couple of negatives to the MaaXBoard is that it does not have a robust developer environment that Nvidia provides to the Jetson Nano and is slightly more expensive.

Below is table 4 comparing the specifications of each development board.

Operating Voltage: Given that most microcontrollers/boards operate on 5 volts or less, we must choose a board that has the correct voltage rating in comparison to the added modules to the project. If a module's voltage does not match the operating voltage, then additional level convertors would need to be added.

Operating Temperature: The range of ambient temperatures that the board can operate in. Assuming that the board and device would be encapsulated within a portable package, keeping the board under the maximum operating temperature with added modules and power supply is necessary.

Max Clock Frequency: The highest stable frequency a clock input can be accomplished. Considering many development boards have robust computational

power, a higher max clock frequency would only mean higher power consumption. Thus, we shall choose one with lower max frequency.

Memory: The amount and type of RAM available on the chipset. Since we are processing high resolution images along with multiple depth sensors, having more RAM would be a benefit. Compared to DDR4, LPDDR4 has similar performance at the reduction of power consumption, which is another thing to consider.

Bit Count: Since single computer boards are fast and secure, 32-bit boards are harder to come by and affect the performance as such. Therefore, a 64-bit board is what we will consider.

Power Consumption: Choosing a board which provides lower power consumption is ideal, however higher consumption is usually a result of more performance. A lower power consumption rating should be taken into consideration, since it increases the thermal properties of the device and can also shorten the life expectancy of the device.

Feature	NVIDIA Jetson Nano Developer Kit 2GB	NVIDIA Jetson Nano Developer Kit B01	MaaXBoard Mini	NanoPi M4V2
Operating Voltage	≥4.75V	≥4.75V	5V	5V
Operating Temperature	-25°C – 80°C	-25°C – 80°C	0 - 70°C	-25°C – 70°C
Max Clock Frequency	1.43GHz	1.43GHz	1.8GHz	2.0GHz
GPU	Nvidia Maxwell	Nvidia Maxwell	GC NanoUltra	Mali-T864
CPU	Arm Cortex-A57	Arm Cortex-A57	Arm Cortex-A53	2x Arm Cortex-A72
Memory	2GB LPDDR4	4GB LPDDR4	2GB DDR4 SDRAM	4GB LPDDR4
Power Consumption	5W-10W	5W-10W	5W	10W
Price	$59.00	$99.00	$73.00	$70.00

Table 4: SoC ARM Board Comparison

After evaluating this table, the needs for the feasibility study into portable 3D scanning can be achieved efficiently with the Nvidia Jetson Nano B01. While not

having as many features as the NanoPi, such as dual Arm Cortex-A72 CPUs and on-board wireless communication via WiFi, the Jetson Nano provides the right amount of computational power with its Tegra X1 and Maxwell chipsets required for the task of this project. However due to supplier issues, the Jetson Nano B01 is unable to be obtained on time for the testing portion of Senior Design 1. We have considered the alternative, which is a fairly identical board in the Jetson Nano 2GB, however since the camera obtained for this study has 2x MIPI connectors, the 2GB version is incompatible with the chosen camera. This oversight would have been mitigated if the other components were selected with the 2GB version in mind rather than the 4GB version. The MaaXBoard Mini and the NanoPi are also alternatives, however like the Jetson Nano 2GB, it would require redesigning the battery system, which would set us back. The logical solution is to continue with the NVIDIA Jetson Nano B01 and have it acquired before Senior Design 2 start.

3.3.2 Camera

When selecting a camera for this study, we had to analyze what type of camera would be most ideal. Choosing the ideal camera had to consider four categories: resolution, output interface, field of view, and feature set. With resolution, we are targeting something greater than 640x480 pixels. Any lower resolution would cause a drop in fidelity that we are looking for. For the output interface, its essential to decide whether we would like to connect via MIPI CSI-2 or USB 2.0/3.0. MIPI CSI-2 is faster than USB 3.0, higher net image bandwidth, and uses fewer CPU resources [1]. USB 3.0, while not as fast as CSI-2, is widely used and an abundant number of cameras are available for it on the market. A drawback of USB 3.0 is that, specifically for this project, USB 3.0 connectors are not small and flexible which may be an issue when designing a portable 3D scanner.

Since we have decided to select the Nvidia Jetson Nano to interlace with the camera system, we must select camera modules that would be supported by the hardware. The Jetson Nano provides 12 CSI lanes that can be assembled in these configurations:

Three x4 lane cameras
Two x4 lane cameras plus Two x2 lane cameras
One x4 lane cameras plus Three x2 lane cameras
Four x2 lane cameras

The x2 lane interfaces can support a single x1 camera and the Nano features two additional pins that can support an additional 2 camera master clocks at GPIO pins 1 and 11. A buffer for the clock is necessary if more than 4 cameras are integrated into the system, which is not an issue as we are using a maximum of two cameras.

SVPRO-1MP2CAM001

The SVPRO is a dual lens camera that connects via USB and effectively produces 1280 x 720 MJPEG images with a 90-degree wide angle view. Being a dual camera, it gives us the added benefit of supplemental depth information alongside the high accuracy provided by the ultrasound sensors. Some features of the dual OV9712 CMOS sensors are adjustable parameters such as sharpness, contrast, white balance, as well as an electronic rolling shutter. The SVPRO is a USB 2.0 powered device therefore you can expect 5V of DC as the input. The operating current of this device is listed to be roughly 130mA to 180mA, which means this is a low drawing current device and will not draw more power from the entire system. Although it is a plug and enable device, the USB cable provided is clunky and might get in the way of other electronics within the device. The USB cable provided to power and communicate with the board is listed as 1 meter long, which is longer than needed for the project. A solution would be to use a shorter cable, but then maneuverability and flexibility becomes an issue. While it is not a true stereo camera, it gives synchronization using two cameras with two different channels on one board. The 90-degree field of view provides an easier way to scan larger objects, however we will lose more detail than going for a lower field of view camera. The operating temperature of this module is between 20-85C; therefore, it is possible for this camera to become hot if it is constantly operational and in use as well as if the device is completely enclosed with the other components in the device.

DUAL OV9281

Like the SVPRO-1MP2CAM001, the Dual OV9281 provides a fixed focus stereo camera module with 1280x800 resolution and 120-degree field of view. These cameras operate as multiple cameras in a series array and acts like a single camera connection that is connected via ribbon cable to either the Raspberry Pi, Jetson Nano, or other similar embedded computers. A key difference between this module and the previous one, the SVPRO, is that it connects via MIPI CSI-2. This gives lower latency and higher throughput. Not only that, but the clunky USB cable disadvantage that the previous camera had is eliminated. The framerate on this module can be quickly changed on the fly from 5fps to 80fps, which at maximum value captures more frames per second than most traditional cameras that refresh at 60 Hz. Other features of this module are the ability to change frame rate and utilize Video4Linux. Image captures are able to be displayed via VLC media player on external computers if they are exported as RAW8 image format. The drawback of this module is that it provides a monochrome image at 1 megapixel with its two synchronized cameras.

IMX219-83 Stereo

The IMX219-83 Stereo 8MP camera module is similar to the Dual OV9281 in that it's a board mounted dual camera system which passes through MIPI CSI-2. However, the IMX219-83 uses two Sony IMX219 sensors that provide 3280 x 2464 still images per camera with an increase to 8 megapixels from the 1 megapixel provided by the other sensors. The Dual OV9281 provides effectively 1,024,000 pixels per image per camera. On the contrary, the IMX219 offers 8,081,920 pixels per camera at essentially 8 times the resolution. The IMX219-83 offers a higher resolution solution that can only be done through still images, which is not an issue for this study as we are not interested in taking full video capture. The integrated ICM20948 chip on the camera provides essentially multiple motion tracking sensors such as an accelerometer, magnetometer, and gyroscope. While each sensor is an added feature, they draw negligible current within the system as they are in the range of microamperes. Applications with this sensor include stereo vision as well as depth vision in higher fidelity due to the high resolution. Out of the box, the IMX219-83 is Jetson Nano compatible, thus we are certain that it works for the project. A drawback of this sensor is that it requires 2 camera ribbon cables to connect with the Jetson Nano, which is why we are limited to one photo sensor.

IMX219-77

The IMX219-77 is essentially a single camera module version of the IMX219-83 Stereo. Using the same 8 megapixel Sony IMX219 image sensor, it's able to capture images at 3280 x 2464. This sensor is essentially the same camera used in smartphones and tablets on a board that connects via MIPI ribbon cables. However, since it is a single sensor camera, the field of view is only 77 degrees compared to 83 degrees on the stereo model. At only $18.90, the IMX219-77 is the cheapest solution. While it is the smallest package of the cameras in comparison, it has 4 screw holes that makes this able to be mounted wherever required for the application. The biggest downside to choosing this camera sensor is that, since it is a single camera, it does not provide the features of depth vision that is required to supplement the project's data.

Active Array Size: The active array size of the sensor, or commonly called the resolution. Given we are assessing the practicality of a portable 3D scanner, having a high-resolution image is imperative.

Output Interface: Choosing a camera with the correct output interface that matches with the available ports on a development board is vital. Most development boards offer USB I/O ports, while some offer MIPI CSI and even less offer MIPI CSI-2. The connection type depends on the amount of data throughput needed to transfer fast and efficiently. Although USB is commonly used and offers the "plug and play" aspect with minimum tinkering, USB is not ideal for portable aspects.

Field-of-View: The field of view of the camera or viewing angle. Lower field of view offers more detail in a smaller frame which is more important in our project than having a higher FOV. By using a narrower angle lens, we can specifically target an object to capture.

Transfer Rate: The data transfer rate of the cameras is proportional to the output interface. Given that we do not intend to make continuous image captures or video, the internal buffer of the camera will not be filled, therefore each capture will be relatively instantaneous.

Board Size: A smaller board size is always considered when developing a portable device. Nonetheless, given the already small form factor, a bigger camera footprint is not detrimental.

Feature	SVPRO-1MP2CAM001	Dual OV9281	IMX219-83 Stereo	IMX219-77
Active Array Size	1280 x 720	1280 x 800	3280 x 2464	3280 x 2464
Output Interface	USB 2.0	MIPI CSI-2	2x MIPI CSI-2	MIPI CSI-2
Field-of-View	90°	120°	83°	77°
Transfer Rate	30fps@1280x720	60fps@1280x800	30fps@1920x1080	30fps@1920x1080
Board Size	80 x 16.5mm	105 x 24 x 22.5mm	85 x 24mm	25 x 24mm
Price	$71.99	$99.99	$48.95	$18.90

Table 5: Camera Module Comparison

After evaluating each camera sensor, we decided the most optimal module would be the IMX219-83 Stereo. The IMX219-83 gives great fidelity in images with high resolution and narrow field of view needed to capture the details. By using two cameras, we can supplement our data using depth imaging alongside other sensors. At $48.95 per module, the IMX219-83 Stereo gives the best performance to cost ratio out of the selected parts.

3.3.3 Microcontroller

Using multiple ultrasonic sensors with the Nvidia Jetson Nano B01 is not practical as the Jetson Nano is incapable of doing real time processing. With using a small microcontroller interfaced with the Nvidia Jetson Nano, the ultrasonic sensors can be controlled in real-time without having to worry about scheduling issues.

I2C bus is a feature of the microcontroller that is needed as it will be used to interface with the Jetson nano. I2C, or inter-integrated circuit, is a transfer bus for data that allows master microcontrollers to control and communicate with multiple slave microcontrollers. This is important for our project as it allows one microcontroller, which would be the Jetson Nano, to record data. In essence, the microcontroller would communicate with the Jetson over I2C to transfer ultrasonic measurements.

ADAFRUIT FEATHER M0 ADALOGGER

Specification	Value
Main Clock Speed	48 MHz
Primary Storage	256Kb
Secondary Storage	32Kb
Communication Module	Serial, I2C, SPI
Clock System	Crystal oscillator

Table 6: Adafruit Feather M0 Adalogger Specifications

This is one of Adafruit's newest development boards, it is notable for its thin, light and overall fast performance. It has a built in USB and also has battery charging. The board has an ATSAMD21G18 ARM Cortex M0 processor, clocked at 48 MHz and at 3.3V logic, the same one used in the new Arduino Zero. This board has 256K of flash and 32 giga bytes of RAM. As the board has USB, it does have USB-to-Serial program and has debugging capability which does not need an FTDI chip. This chip is also very useful for portable devices, such as for applications that we are trying to make since it does have a connector for 3.7V Lithium polymer batteries so that it could be recharged from a micro usb connector. The board can detect when a USB port is being used to power the device and switch back to battery if needed.

The board weighs approximately 4.6 grams. Its chip runs at 48MHz with 3.3V and has a 3.3V regulator with 500mA peak current output which serves for an ample amount of throughput power. There are 20 GPIO pins, and has hardware serial, hardware I2C and hardware SPI support. There are 8 PWM pins and about 10 analog inputs. The lithium polymer batteries have a 100 mA charger with an LED

light that indicates the charging status. The board also has 4 mounting holes and a standard reset button. For the power supply there is a BAT pin that is tied to the lipoly JST connector and with a 500 mA peak regulator, the board would be able to get the full 500 mA but not continuously from a 5V power as the regulator will overheat.

Given the projects need for a lightweight design, this chip is definitely very useful for us. Additionally, the rechargeable features of the batteries is something that we would also find essential. Overall, the development board has a very good layout arming us with what we deem to be essential. The board also has very good documentation and easy to find datasheets pertaining to everything about the board. The Adafruit website also provides step by step tutorials ranging from specifications to mounting and soldering the board. The site also provides sample code. Some of the sample code is useful especially for the power supplies as they provide code to measure the battery, find the average power draw and much more. Also, when it comes to the features of this board, the clock system is extremely useful as it can take in multiple sources for an input, this would allow for versatility in our design as we would be including many sensors, and the same can be said for the communication modes as it has SPI, I2C and Serial hardware.

ADAFRUIT FEATHER M0

The Adafruit Feather M0 is a lightweight board that is thin and roughly the size of two quarters, it is a suitable candidate for implementing it into the portable scanner as it already has a development board. The Adafruit Feather M0 boasts 32KB of on-board RAM and a connector specifically for 3.7V Lithium batteries, although we can still use the board powered straight through USB 5V as this microcontroller regulates the USB voltage to 3.3V. This microcontroller has many powers supply options apart from the ones previously mentioned. Two other options are the BAT pin and USB pin, which are sourced by the lithium JST connector and +5V USB source when connected, respectfully. The primary ways the Adafruit Feather M0 is powered is either the 3.7 LiPo battery using the dedicated JST connection or a USB cable. A negative about this board is that you cannot use alkaline or NiMH batteries as it would destroy the LiPoly charger, and it is not designed for external power supplies as this is a very small board. Connecting an external 3.3V power supply to the 3V pin while grounded will cause the board to behave unexpectedly and the EN pin will cease to function. This board damaging will also happen if a 5V power supply to the USB port as it can damage anything connected to it as the port will be back powering. The Feather also can operate via USB and 3.7V LiPo battery as backup. When the battery is not in use, it switches to a state of accepting 100mA charge through the USB port. If the USB port is disconnected, the Feather will automatically switch to battery powered mode so that the system does not cease operation. If we need to measure the voltage of the battery during operation, the added 100K resistor bridge divider to the JST port that lets us measure the active current via pin D9.

SEEEDUINO CORTEX-M0+

The Seeeduino Cortex-M0+ is a board based on the ARM Cortex-M0+ processor, which is the same processor featured on the Adafruit Feather M0. Unlike the previous microprocessor, this board features a USB type C interface in comparison to the micro-USB featured on the Adafruit. Instead of a Li-ion JST connector, it features a 7~15V DC jack to supply power if desired. A benefit of this board is that it is powered on a 5V logic, which matches the battery output. Using the Seeeduino will eliminate the need for a voltage regulator. Internally, the Seeeduino Cortex-M0+ is extremely comparable performance wise, however it boasts a 2.8 times larger area footprint than the Adafruit Feather. While it is half the price, the size of this board is too large to incorporate with the already existing NVIDIA Jetson Nano into a portable package.

Item	Value
Microcontroller	SAM D21
Power Input	USB Type C
Operating Voltage	USB: 5V
Digital I/O Pins	14
PWM Channels	10
Analog Input Channels	6
DC Current per I/O Pin	40 mA
IO Input Voltage	3.3V
SRAM	32KB
Flash Memory	256KB
Maximum CPU frequency	48 MHZ

Above is a table that contends the specifics items and values about the SEEDUNO CORTEX-MO+

ARDUINO ZERO

Similar to the previous microcontrollers, the Arduino Zero features an identical processor, the ATSAMD21G18. The Arduino architecture is an established platform with many documentations available, therefore it is more user friendly. A difference in the Arduino Zero is that it features two USB ports: Native port and programming port. While both ports can be used in programming the board, its recommended to use the latter port as it uses embedded debugging tools. While double the price of the Adafruit Feather M0 and roughly the same size as the Seeeduino, the Arduino Zero might have portability issues. This board features an embedded debugging tool from Atmel, that allows a full debug interface without installing additional modules, which is a quality-of-life improvement for developers. While most Arduino boards run on 5V logic, this board runs on 3.3V, similar to the

Adafruit Feather M0, and is susceptible to damage if voltages greater than 3.3V is applied to any GPIO pin. Unlike the feather, the Arduino Zero can be powered by either USB connection or an external power supply through the provided VIN and GND pins of the power connector. When connected in such manner, the board supports power supplies in the range of 6 to 20 volts, while the recommended specs are 7 to 12 volts. Each of the 20 GPIO pins are capable for either digital input or output using the proper Arduino IDE functions. While a big board, the board provides 3 screw holes for mounting purposes and is compatible with most shields in the Arduino family.

Feature	Adafruit Feather M0	Seeeduino Cortex-M0+	Arduino Zero
Operating Voltage	3.3V	5V	3.3V
GPIO Pins	20	20	26
Weight	4.6g	17g	12g
Dimensions	51 x 23 mm	65 x 52mm	68 x 53mm
Price	$23.15	$9.90	$39.50

Table 7: Microcontroller Comparison

Operating Voltage: The operating voltage ideally must match the development board and the battery as it would not need the services of a voltage regulator to step-up or step-down a voltage. The Seeeduino Cortex-M0+ runs on a 5V logic, which essentially is the required operating voltage of the microprocessor we would like.

GPIO Pins: The general-purpose input/output pins typically come with a 40-pin header on fuller size microcontroller boards. However, each candidate featured provides roughly 20. We are using two ultrasonic sensors; therefore we only need four pins for TRIG and ECHO to connect to. This is not accounting for the GND and VCC pins which would be connected directly to the Jetson Nano.

Operating Frequency/CPU: The ARM Cortex M0 processor chip presented on the board features 48 MHz clock frequency, which is 3x the clock speed compared to the TI MSP430FR5969, which operates on a 16-bit RISC architecture at 16 MHz. This is plenty of processing power to run the two ultrasonic sensors in realtime.

Weight/Dimensions: The weight and dimensions of the microcontroller is the most important constraint in selecting the right board for the purpose. Since all boards have identical processing power, the smaller the microprocessor the better as we already have a board with a large footprint.

Price: The price of the microcontroller is second to the weight and dimensions in priority when settling on the correct board. The price of the Adafruit Feather M0 to

the Seeduino Cortex-M0+ and the Arduino Zero to the Adafruit Feather M0 is double, respectfully to each other. Therefore, balancing between size and price is more beneficial.

Any microcontroller we use for this project will be used for making sure the entire project runs smoothly and orderly. Unfortunately, the development board we will likely be unable to process the image in real time. The number of ultrasonic sensors and cameras would overwhelm the development board, which is the reason why the microcontroller is required for our project.

So far, only three microcontrollers catch our eye. The Adafruit Feather M0, the Seeeduino Cortex M0+, and the Arduino Zero. The Adafruit Feather M0 is aptly named for its extremely small size. This will be useful when it comes to fitting it into our design. The Seeeduino Cortex M0+ can handle many complex calculations with high speeds for data transfer. The high transfer speed for data is nice, but we do not need a microcontroller that handle's complex calculations. The Arduino Zero is a very powerful microcontroller, useful for many projects involving robotics and automation. This might be way over what we need in terms of specs for the microcontroller.

3.3.4 Battery Power and Other Solutions

The battery is the source of the power of the system. A battery is important to the system since it completely powers all the electronics within the system and the wrong battery system can fry circuitry. Keeping the battery running for the duration of the time that the system must run is paramount, since reliability is one of the most important constraints of this project. Some things to consider is how many watts the system requires, the size of the batteries, weight and power density, voltage, price, and reusability. In this study, since our device is portable, we will have to utilize smaller batteries, therefore we will not consider lead acid batteries for the project. Portability is an important aspect and constraint, as a result choosing a smaller, lighter, higher density battery is the most logical approach to power the device. Higher density is proportional to the price of the battery and must be balanced accordingly.

Another thing to consider is whether our battery should be rechargeable which is important considering the specifications state the battery must last 2 hours before being recharged. Since our development board will draw 5V input and the microcontroller will draw below that (3.3V), 6 NiMH cells at 7.2V can provide the 4.75V minimum necessary to power the NVIDIA Jetson Nano Developer Kit. A 5V to 3.3V level converter can then be utilized to power the necessary ultrasonic devices within the microcontroller or a printed circuit board featuring the necessary voltage divider bridges.

The NVIDIA Jetson Nano can reach a maximum power consumption of 20+ Watts, therefore we can narrow the options down to a couple of solid candidates to power our system. Given that our project is not overly power hungry and does not draw multiple amps of current, we do not need to utilize a lead acid battery. Although they are inexpensive, they are bulky and cannot be utilized in a portable manner. NiMH batteries are a possible choice for the project since they offer standard alkaline standards with higher power density. The high self-discharge rate in NiMH might be a downside to develop a finished product, but since this is a feasibility study, service life and/or storing the product is not a constraint.

Another alternative to battery power is AC power from the wall outlets. This source of power feeds 120VAC which requires an AC to DC power supply to convert it to a stable 5V DC power. This requires a much more robust voltage regulator to rectify the voltage down drastically from the source. However, since the Jetson Nano natively has a barrel jack female connector present on the board, designing an AC to DC convertor can become much simpler as that jack is commonly used for such power devices. On the contrary, while it does require an AC to DC convertor, the parts are much more lightweight to construct and implement than having a battery pack within the system. A negative of this solution that it would limit the portability aspect of this design as it will be tethered to the wall outlet.

3.3.5 Ultrasonic Sensor

Utilizing Ultrasonic waves is extremely useful for not only the detection of objects, but also for the evaluation of them in any kind of medium such as gas, liquid and solid. Ultrasonic waves are powerful and have been used particularly in the health care industry for diagnostics, as electromagnetic waves attenuate rapidly throughout the human body as well as metal objects. These waves therefore are relatively safe. Compared to those of x-rays, ultrasonic is much safer and offer excellent imaging at a much lower cost.

Another important feature of ultrasonic waves is their low propagation speed. This helps its measurement and imaging, as the time of flight can be used to provide an estimation for the distance that exists between an object and ultrasonic transducers which would be used in this project as it is able to create three-dimensional pictures.

As we are looking for multiple ultrasonic sensors to be installed within the device, the issue of each sensor affecting the reading of one another will be a cause for concern when designing and selecting parts. Sound propagation from the transmitter transducer travels within a cone while the receiver transducer acquires an echo signal within that cone. Each pulse is sent through the air in front of the transducer, spreading and increasing the diameter of the cone relative to the distance. When multiple ultrasonic devices are situated side by side, then each sensor’s propagation cone will likely interfere with another leading to unwanted

cross noise that would prevent the system from accurately measuring the distances at each point. By partially enclosing each HC-SR04 sensor from the sides, we can bottleneck each propagation cone and prevent and/or limit cross interference.

Another reliable solution is to sequentially read each ultrasonic sensor. By connecting the Echo pin of the first sensor with the TRIG pin of the following sensor, then it is possible to chain each sensor to sequentially range after the previous one has finished. This will prevent interference by limiting the total number of current active sensors at any given time to just one. A downside of this method is that the update rate of each sensor will be the single refresh rate times the total number of sensors. Additionally, if, at the same time, multiple sensors are transmitting and receiving unsynchronized then drifts in frequency will interfere with the operation of them.

HY-SR05

This ultrasonic sensor has a working voltage of 5 VDC. Static current is less than 2 milliamps and has a sensor angle of less than 15 degrees. The sensor can detect objects that are between 2 cm and 450 cm with an approximal precision of 2 millimeters. It has an input trigger signal of 10 us TTL impulse and has a 5 pin layout consisting of VCC, trig (T), echo (R), OUT and a ground. This sensor works by having a pulse signal detected to I/O TRIG, which will have the module start detecting. The module will send eight 40khz square waves and will wait for a reflect signal. When the reflect signal is sent back, the ECHO I/O will have a high level for output, the duration of the high-level signal is going to be the time from the ultrasonic launch to return. Therefore, the measured distance =

(T(Time of high level output) * (340M / S))/2

HC-SR04

The HC-SR04 is one of the most selected ultrasonic components on the market. The HC-SR04 features two transducers, a transmitter and receiver. The transmitter takes electrical current and sends it as 40 kHz ultrasonic pulses, while the receiver listens to the feedback and delivers and output pulse that is used to calculate the distance at up to 4 meters away. The HC-SR04 requires a 10-microsecond pulse to initiate the rage at 40 kHz which will return an echo pulse. This ultrasound sensor, unlike the HY-SRF05 has 4 pins rather than 5 pins. Pin 1 is VCC, which the HC-SR04 operates on 5 volts and can be used to connected directly to 5V logic microcontrollers. Trig, or Trigger Pin, is used to prompt ultrasonic pulses, while the Echo pin produces a pulse that measures the distance in proportion to the length of time it took. The last pin is the GND, or Ground pin, which should be connected to the ground rail of the project. Furthermore, this device has four mounting points that allows it to be installed stably in the front of the system.

One downside of the HC-SR04 sensor is if the object has a reflective surface and the angle of the device in relation to the surface is lower than 45 degrees, then the sound would not be reflected to the device as the transmitted sound wave will be directed away at an equal angle from the incident wave. Since the angle of reflection is equal to the angle of incidence within sound waves on flat reflective surfaces, this limits the operation of the device to objects that tend to diffuse sound in different directions rather than reflect within a point so that detection is made possible. Another limitation is that some objects that have soft, porous surfaces would absorb rather than reflect sound and limit detection.

URM37 V5.0

Another ultrasonic sensor candidate is the URM37 V5.0 sensor from DFRobot. While similar in design and shape to the HC-SR04, the URM37 offers different specifications at over 3 times the cost. Since this ultrasonic sensor operates at 3.3V to 5.5V, it can be used with the Adafruit Feather M0 without use of a level converter due to being able to run at 3.3V stable. While the URM37 has identical deadband within 0 to 2cm as the other candidates, it's operational range far exceeds any at 800cm. With 9 pins, this ultrasonic sensor uses either RS232, which is highly reliable, or TTL- level output. Choosing between TTL or RS232 can be done by pressing a single button on the device for 1 second. However, since the ultrasonic sensor will be connected via MCU, then TTL level output should be selected.

The URM37 features temperature compensation within the module itself in case of environmental change in temperature that can help in the case of accurately measuring distance.

Feature	HY-SRF05	HC-SR04	URM37 V5.0
Operating Voltage	4.5V ~ 5.5V	5V	3.3V ~ 5.5V
Resolution	0.3cm	0.3cm	1cm
Shape	Sqaure	Rectangle	
Current Draw	10 to 40mA	15mA	20mA
Pins	5	4	9
Operational Range	2cm - 450cm	2cm - 400cm	2cm - 800cm
Precision	~ 3mm	~ 2mm	
Dimensions	45 x 15 x 27 mm	45 x 20 x 15 mm	51 x 22 x 13 mm
Price	$2.49	$3.95	$13.90

Table 8: Ultrasonic Sensor Comparison

The table above shows a comparison between the HY-SRF05 and HC-SR04.

3.3.6 Voltage Regulator

Since the NVIDIA Jetson Nano requires a 5V input ideally and is the highest power drawing component in the project, it is wiser to feed 5V to the system and step down accordingly to each electronic component. Since the HY-SRF05 requires a 5V input and the microcontroller (Adafruit Feather M0) runs on 3.3V, then the sensors would need to be powered via the Jetson Nano, even though it's connected to the microcontroller. The ECHO feedback signal is 5V as well on the ultrasonic sensor, so a PCB design of the appropriate resistors to convert the response from the sensor to a safe level is needed.

Since the two boards are at operating voltage mismatch, a simple step-down regulator is needed. Since linear regulators are cheaper and simpler than a switching regulator, they are the considered type of regulator to be used for this project. The lower power efficiency of linear regulators is not an issue for this project, because we are only utilizing one. With the input of 5V to 3.3V, the circuitry only experiences a loss of 2.7V in the form of heat.

Like voltage regulators, voltage dividers are used in electronics with resistive elements. These are not proper voltage regulators, however they can be used in place for when the voltage is regulated and the draw of current through the divider is low. A voltage divider does not regulate the voltage as a change in voltage from 10V to 5V would output 6V, if the input voltage changes to 12V and the voltage divider limits the voltage by half.

LD1117-3.3

The LD1117 is one of the most popular and used linear voltage regulators on the market. It can bring a minimum of 4V up to 15V from a battery down to a stable 3.3V. While this regulator is reliable, it is a hefty component at 29mm lengthwise, which brings it to 1/5 the size of the Adafruit Feather microcontroller. The LD1117 is also a low drop voltage regulator that able to provide up to 800 mA of output current. Furthermore, the LD1117 is supplied in: SOT-223, DPAK, SO-8, TO-220 AND TO-220FM. The SOT-223 and DPAK surface mount package optimize the thermal characteristics even offering a relevant space saving effect. High efficiency is assured by NPN pass transistor. In fact, in this case, unlike than PNP one, the quiescent current flows mostly into the load. Only a very common 10*u*F minimum capacitor is needed for stability. On chip trimming allows the regulator to reach a very tight output voltage tolerance, within plus or minus 1 % at 25^0 C. So, the adjustable LD1117 is pin to pin compatible with the other standard. Adjustable

voltage regulators maintaining the better performances in terms of drop and tolerance.

AMS1117-3.3

The AMS1117 voltage regulator is a smaller linear regulator than the LD1117. It can step-down a 4.75-12V battery to 3.3V. A difference between this and the previous component, is that it can step-down a maximum of 12V rather than 15V max in a smaller package and is a buck converter rather than a linear regulator. The AMS1117-3.3 is a 1A low dropout voltage regulator. Therefore, it is designed to provide 1A output current and to operate down to 1V input-to-output differential. The dropout voltage of the AMS1117-3.3 is guaranteed maximum 1.3V at maximum output current, decreasing at lower load currents. The applications of the AMS1117-3.3 are high efficiency linear regulators, post regulators for switching supplies, 5V to 3.3V linear regulator, battery chargers I, active SCSI terminators, and power for notebook, battery powered instruction. The AMS1117 is also a mini voltage regulator that fits perfectly with the smaller size design requirements.

Parametrics of AMS117-3.3, AMS1117-3.3 absolute maximum ratings; power dissipation is internally limited, Input voltage is 15V, operating junction temperature for control section is 0^0C to 125^0C, for power transistor is 0^0C to 150^0C, for storage temperature is $–65^0$C to $+150^0$C and soldering information leads temperature for 25 sec at 256^0C.

Below shows the table of some features of each voltage regulator. We have decided to stick with the AMS1117-3.3 since the LD1117-3.3 is a much chunkier design and will still provide the 3.3V / 500mA minimum required to operate and power the Adafruit Feather M0.

Feature	LD1117-3.3	AMS1117-3.3
Output Voltage	3.3V	3.3V
Regulation	1%	0.4%
Output Current	800mA	800mA
Minimum Input Voltage	4V	4.75V
Maximum Input Voltage	15V	12V
Pins	3	3
Dimensions	29 x 10 x 4mm	8.6 x 12.33mm
Price	$1.25	$7.99 / $0.3995 per unit

Table 9: Voltage Regulator Comparison

In senior design 2, we have decided to pursue on-board chip voltage regulators in the form of buck converters, therefore we did not utilize these cumbersome voltage regulators.

3.3.7 P-Channel MOSFETS

P-Channel MOSFETS are important to the printed circuit board aspect of this project and the power management. If during testing, we accidentally plug in the battery terminals in the reverse order, then the entire circuit may end up frying. This is where the p-channel MOSFET comes in; through reverse polarity protection. The transistor will only allow current to flow into the circuit, if the battery is connected in the right way. If not, then the transistor turns off and closes off any flow of current.

3.3.8 Micro Servo Motors

While looking into what servos to use, the one most important aspect we looked into was if whether or not the servos had a pan/tilt kit that comes with it. The most lightweight servos that came with a pan/tilt kit was the SG90 micro servos. The SG90 micro servo has an operating range of 5V, and an operating speed of moving 60 degrees per 0.1 seconds. The operating range of the servos is 0 – 55 degrees celcius.

Figure #: Servo Pan/Tilt Kit

3.4 Possible Designs and Related Diagrams

The most likely design for this 3D scanner would be in some way where the scanner fits on top of a stand or tripod when scanning. It must be portable so that it can easily be taken anywhere, and it must be stable when scanning or else the 3D image will be negatively affected. Usual portable 3D scanners would use Blue Lasers in a grid to make sure the 3D image is correct even when shaky, but for us we are using an ultrasonic sensor, which will need a second to calculate the distance of the object.

3.5 Parts Selection Overview

Description	Model Number	Quantity	Estimated Cost	Total Cost
Development Board	NVIDIA Jetson Nano Developer Kit B01	1	$99.00	$99.00
Microcontroller	STM32F103C8T6	1	$7.00	$7.00
Ultrasonic Sensor	HC-SR04	3	$2.49	$7.47
Camera Sensor	IMX219-83 Stereo	1	$48.95	$48.95
Voltage Regulator	AMS1117-3.3	1	$7.99 / $0.3995 per unit	$7.99
			Grand Total:	$186.56

Table 10: Budget estimates for project

After selecting all of our parts for this project, we can see that the total cost of the portable 3D scanner is around $186.56. This is indicative that we have kept the total cost down and below $200 stated in the engineering requirements specification. For the development board, the most expensive option was chosen in the Nvidia Jetson Nano B01. The 2GB version of the Jetson Nano is similar in performance as the internals are the same and it is $40 cheaper. However, this was not selected as the feature set of this board, as it is a more barebones solution, does not have enough camera slots for this design. The cheapest part of the total design are the ultrasonic sensors at $2.49 per sensor. Due to the high resolution and quality of the camera sensor, IMX219-83 Stereo, it costs roughly 50 percent of the total cost of the Nvidia Jetson Nano B01 board. While it is an expensive option compared to a single sensor alternative, it provides the intended depth imaging that is essential for the project.

4. Related Standards and Realistic Design Constraints

This section will pertain to standards that going to be applicable to the study of the feasibility of our 3d scanner design and constraints of said design. Protocols that must be followed to ensure that the designing and application of the product will adhere to safety standards to remove any potential liability issues. Product quality is one of the most important factors that we need to consider, so that the product is able to perform its functionality properly, but also be able to withstand anticipated frequent use. In order to ensure that our product quality is high standards pertaining to IEEE and other related standardized methods will be followed in order to have not only a working product, but one that is also safe and reliable.

4.1 Related Standards

Standards are imperative to take note of, they provide an outline and base ground for which to build designs and its applications based off. These standards pertaining to the feasibility of the 3d scanner will be attained from IEEE Standards Association. Additionally, the ISO, or the International Organization for Standardization is an international standard setting body built on representatives from various national standard organizations. Given our products goal to study the feasibility of a 3d scanner, approaching our product that adheres to global standards will provide us a much more complete scope to develop our product. Our product is very complex, it incorporates the use of optics which will require standards for the accuracy but also consistency. The power units utilize electricity, therefore following standards pertaining to safety is going to be important.

4.1.1 Quality management systems

The “ISO 9000:2015 Quality management systems” will be followed as we will use its guidelines as the team dynamic will require success of an implementation of a quality management system, this is not only in the delivery of a product, but also delegation of tasks that will be needed to be carried out. Additionally, our product is created with the intention of serving clients/customers, therefore customers will seek confidence in our organizations ability to provide a product that adheres to the general requirements, and a product that is consistent and easy to use. Additionally, the ISO 9000:2015 provides an outline for an organization to ensure that the development of a product or service is adhering to not only these standards, but of others that will be applicable to it. Moreover, the development and improvement of communication between members utilizing common/standardized vocabulary to ensure proper development but also application is stressed, considering the complexity of the product, using the same sort of language, whether it be variables or layouts is important to provide clarity to the development of our product and its report.

4.1.2 Product Quality

Given that product quality is something that we will strive for. Given the nature of being a high reliability electronic, there are standards such as that of the Government Electronics & Information Technology association (GEIA). The IEC 61709 Ed, 3.0 b:2017 is a document that focuses on serving as a reference for conditions relating to failure rates and stress models for conversion. The guidance for failure rate data in prediction of electronic components in the system is also stressed. Since the scanner will be largely based on being able to accurately depict an object, ensuring that our group is being guided on how to use our data to minimize any pejorative outputs will be important. Guidance on utilizing a database of component failure data and how it can be constructed to provide data to be included in stress models, given that the scanner will be put in a variety of environments and with potential heavy use, following guidelines that aim to provide a clear and complete understanding on how to analyze stress tests in order to continue to improve the function and performance of the scanner will be done. The IEC 60721 – 3-7 has a section that delves into the classification of groups of environmental parameters and their severities. The scanner could be in a cold environment and should be able to operate and have similar output to that of the scanner in a warmer environment. Ensuring consistency with the product is something that we must consider for the success of our study in its feasibility of being a portable scanner. The IEC 60605-6 has guidelines on equipment reliability testing, testing for the validity and estimation of failure rates and the intensity is also something to take note of. In order to create the best stress models that we can, having a thorough tester will be important.

4.1.3 Battery Standard

Utilizing batteries, our group will follow the IEC 600500 international Electrotechnical standards. This will ensure we are complicit in our communication to ensure validity and consistency throughout our documentation and implementation of said products between all groups that will be required in the development of this product, between us, professors, advisors and even manufacturers when we purchase the parts that will make up our product.

Moreover, the IEC 60086 focuses on the standardization of batteries in regard to dimensions, terminal configurations, markings, test methods, performance and safety/environmental aspects. These standards for the most part are to benefit the battery users, which would include the customers but also the device designers. These guidelines additionally provide test methods for the examination of primary cells and batteries to ensure proper testing can be conducted.

4.1.4 Design Impact of Battery Standard

It is very important to demonstrate that a battery complies with the IEC standards. Complying with the standards enhances a consumers trust in the product and brand. Demonstrates consideration and implementation of safety in the products. Additionally, batteries that do not have the IEC 60086 mark could be prohibited in certain markets and areas. Additionally, there are specific standards to take note of given the nature of a portable battery as those requirements are different than a typical battery which would be considered either an industrial or automotive battery. Understanding the variety kind of primary batteries is also important, and is listed as the following:

Electrochemical composition of batteries	End Use
Alkaline Manganese	Multi-purpose batteries
Zinc Carbon	Low/moderate drain applications
Lithium	Used for cameras and small electronic applications
Zinc Air	Used in hearing aids
Silver Oxide	Miniature batteries (used for watches)

Table 11: Types of Batteries

Given that we will be dealing with a camera and small electronic hardware, lithium batteries standards will be emphasized.

In the United States, the American National Standards Institute (ANSI) developed the nomenclature for these portable batteries using a letter system which indicate the size in height and width of the battery. Such as “AA” which represents “50.5 mm x 14.5 mm”. Ensuring that we are using the best kind of batteries based off the standards, but also referring to them in a way that is utilized in the commercial industry will be stressed.

Considering that the battery is going to power the camera, it is going to need to be consistently powering the unit and ensure that it does not overload the microcontrollers and the other components in the system. Therefore, it will be extremely important that standards are followed to ensure safety, but also reliability of the batteries and its functionality. The system must be compatible with the kind of batteries that will be utilized. Taking note that lithium is the probable battery that will be implemented, ensuring the electrochemical composition of the system is compatible with one another. Each component in the system has tolerance that differs with one another when it comes to temperature, signals and therefore taking into consideration the compatibility will be needed. This level of attention to electrochemical composition is emphasized in the IEC 60086.

Additionally, another interesting design proposition is that of a rechargeable battery. Given consumers would rather avoid purchasing batteries, and instead

just charge their devices a rechargeable battery would have to be considered. A table of these so-called secondary batteries can be shown

Electrochemical composition	Type of end use
Nickel Cadmium	Rechargeable multi-purpose batteries
Nickel-metal Hydride	Rechargeable multi-purpose batteries
Lithium Ion	Used for mobile phone
Lithium Polymer	Used for mobile phone, laptops, portable audio
Lead-acid	Hobby applications

Table 12: Continuation of Types of Batteries

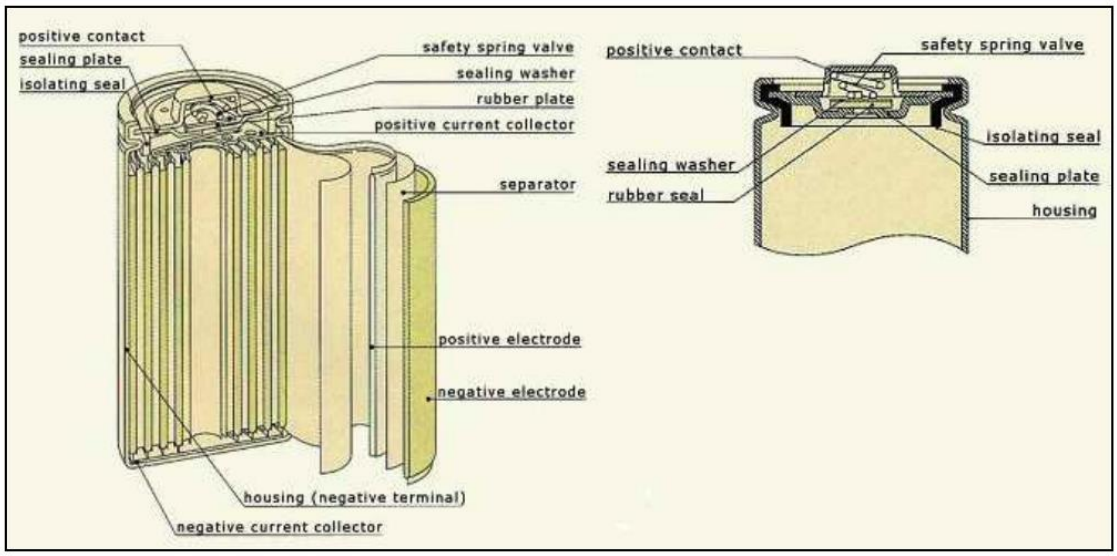

Figure 6: Design of a Rechargeable Battery

4.1.5 Programming Languages – C Standard

When programming, it is not just enough to write a program that executes what is needed, but that it is written in a standardized way in order to provide ease in debugging, understanding and modifying. This is not just limited to the language that is being utilized for specific applications, but also in the styling of the code and the way that variables are conjured and utilized throughout the program.

When programming, comments play an important role in being able to communicate with the reader, as well as providing better organization for the coder. These nuances are touched on in the ISO/IEC 9899:2018 Programming

languages – C. Notation, similarly to how the electronic vocabulary must be standardized across documentation will also be needed. One way in which the programming will be standardized as well is by utilizing the Nasa C style guide. Nasa states that in its purpose the Software Engineering Laboratory (SEL) recommends code that is in “good style” where it is:

Organized
Easy to read
Easy to understand
Maintainable
Efficient

Nasa also emphasizes the use of white space and blank lines in order to enhance the readability of the code. This is more or less used to create a logical sequence of lines being more apparent in the program. However, Nasa also notes that the overuse could defeat the purpose of grouping which would then reduce readability and overall organization of the program. So it is important to focus on just using one single blank line to separate parts of the code within the program

The use of comments in code is also important, however placing too many can decrease the organization and readability of the code. Nasa emphasizes that there should be a README file in order to provide a general description of the program and is able to explain its organization. Additionally, a prolog that can explain the purpose of the files and provides more information. Nasa states that comments should be used to **add information** for the reader or to **highlight sections** of code.

Given that the MCU will be programmed in C/C++, it will be important to follow the standards of the manuals in regards to C. Following the manual will ensure that advanced code can be written, considering the product will be a complex system. Having advanced code requires consistency throughout which will be achieved following the standardization process of programming in C. The product should not be limited by the inability to conjure a complex program. Following the standards will ensure that the expansion of the code can introduce complex systems within the system and its design given the powerful nature of the C language.

4.1.6 C Testing

For testing purposes, the ISO/IEC/IEEE 29119-2 is a section of the standard where test processes used to govern and manage for the implementation of software testing. Given the complexity of the system, ensuring that proper debugging is

done for the multitude of electronics ranging from sensor data to calibration is done. The test process has three groups, consisting of organizational test process, test management processes and dynamic test processes.

The former, is made up of procedures for the creation, maintenance, and the review of test specifications. Test management process is based on test planning, monitoring and control, and completion. Once a test plan is established, test monitoring will allow for anything that is unplanned in the system to be able to be addressed and rectified immediately. Once these are met the group should move over to test completion.

Within the test completion, there exists the dynamic test process that includes test processes that are related to the test design and implementation, its environment set-up and maintenance, along with execution and incident reporting.

4.1.7 IPC PCB Standards

For IPC/PCB standards, it is important to use software that is tailored to following guidelines that follow a standardized organization. The Altium designer, is one such where they are designed to ensure reliability and manufacturability. Using this software allows a user to minimize disparities within their documentation, and therefore provides a way to achieve consistency and correctness across their documentation in regards to PCB layout
The IPC 2221 focused on establishing a generic design and performance requirement in PCB's and other forms of component mounting or interconnecting structures. This will be important as the scanner will have many components that we will have to build and order so that it can work. Given that product quality is one of the more important components of our design, IPC 4761 provides a benchmark for guidelines regarding reliability, manufacturability and quality, and the IPC-a-600 takes it a step further to define acceptance metrics for printed boards and standard classes for printed circuit board assemblies.

For printed boards there are a variety of internal and external observations that are to be required. The purpose of which is to portray specific criteria of current IPC specifications, the printed board should comply with design requirements of the applicable IPC-2220 series document and the performance requires of applicable IPC-6010 series document. This serves to ensure that the supplier gets a correct understanding of the type of product we will need, to receive the best service and highest quality product. This is going to be important as the group needs to ensure that the parts that get ordered, are what we intend for them to be. Given the nature of electrical components being so small, having the correctly build and desired components/parts ordered is going to be imperative given the fact that the project and its build up is a time sensitive matter.

4.1.8 Ultrasonic Testing

ISO 5577:2017 provides a guideline for usage in ultrasonic non-destructive testing that forms a basis for standards and general use. Additionally, this standard provides the users with common vocabulary to use in regard to what would come with ultrasonic sensors. Given the usage of ultrasonic sensors in our project, understanding the functionality of the sensors, but also how to refer to different features and how to have a standard description using its components is going to be important. Ultrasonic sensors will be used in this project in order to measure the depth.

In regard to testing the group will have to use probes that are able to transmit, receive, process and display ultrasonic signals. This will serve not only in aiding the camera to accomplish its goals, but it will also be used to measure the accuracy and troubleshoot the sensors when needed in the design.

The ISO 5577:2017 also provides a mean of understanding calibration block by which the sensors can be assessed and calibrated. Additionally, it shows how there can be reference blocks, test blocks and reference echo. These are all different features when it comes to calibration and testing for the sensors.

Additionally, the documentation provides are a variety of techniques that can be used for scanning. The tandem technique involves the use of two or more angle-beam probes, which while facing the same direction their ultrasonic beam axes in the same plane perpendicular to test surface where one probe is used for transmission and the other for reception. This technique is useful for mainly detecting discontinuities perpendicular to test surface.

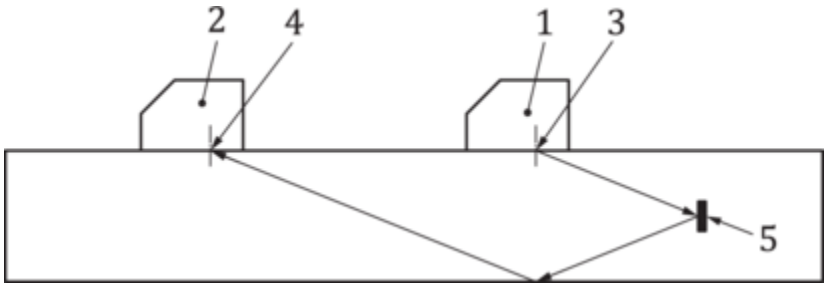

FIGURE SHOWING TANDEM TECHNIQUE

Figure 7: Tandem Technique in Ultrasonic Sensors

The next technique is called the through transmission technique, it is one which the quality of a material is assessed by transmitting ultrasonic waves through the entire material using a transmitter probe on one side of the object and a receiver probe on the opposite side. The gap technique however is one in which the probe is not in direct contact with the surface of the test object, however it is coupled to it utilizing a layer of liquid which is not more than a few wavelengths and example diagram of the gap technique is as follows:

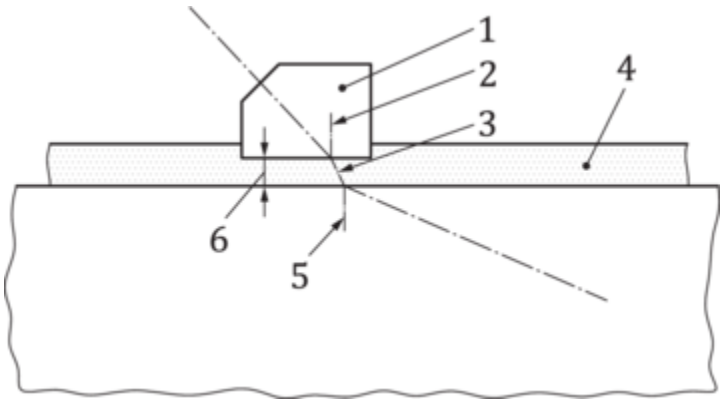

FIGURE GAP TECHNIQUE

Figure 8: Gap Technique in Ultrasonic Sensors

From the aforementioned figure, 1 is the angle beam probe which is set at the top. 2 is the prove index point, along with the couplant path (3) and the couplant (4). 5 is the point of incidence and lastly 6 represents the distance or the “gap”. All in all, this documentation provides us with a variety of techniques that we can use in certain situations to actively access and modify the sensors in order for it to be calibrated to the best it can be.

4.1.9 Design Impact of Ultrasonic Testing

The ultrasonic incorporation into the project will be a challenge as this would be the first time many in the group get to deal with sensors in such a way, therefore it will be imperative to really understand the physics that is happening, and to follow the guidelines as a way to correctly utilize the sensors, and make sure we get the most out of them through the variety of techniques to calibrate. We will actively be researching and learning to continue to get a better understanding of them, and make sure that the sensors are doing what they are supposed to do.

The guidelines provided in the ISO 5577:2017 actually do have a very straightforward approach in how to properly assess the sensors and gives us vocabulary that lets us know as much information that the sensors is responsible for such as a variety of signals and indications that the ultrasonic sensors can be

responsible for. In terms of echoes, there are back-wall, surface, side wall, echo width, echo height etc. Taking a look at the documentation and reading it thoroughly will teach us a lot about what the ultrasonic sensors has the potential to measure/do and therefore it will be imperative to follow the guidelines.

4.2 Realistic Design Constraints

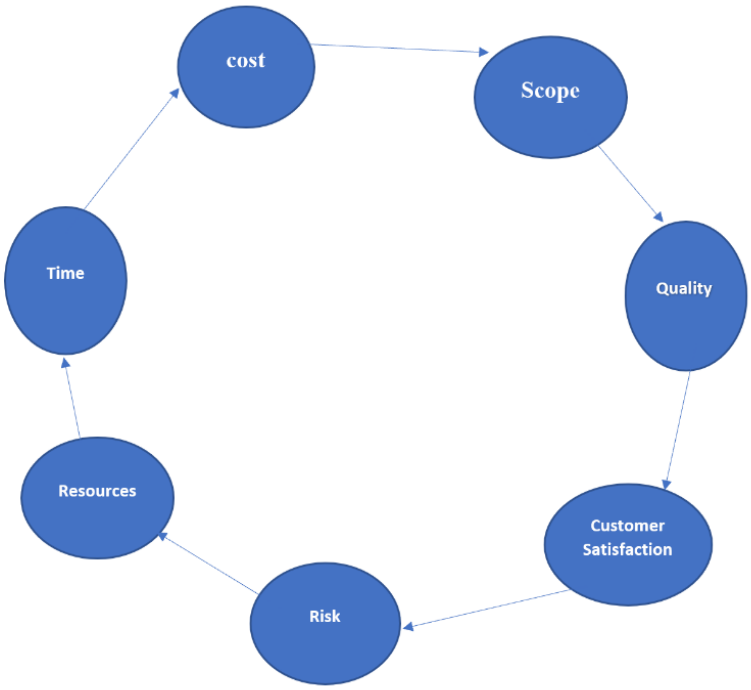

Figure 9: Project Constraints

As shows in the block diagram above we can say the constraints are the live cycle of our project. Therefore, if there is a change in any of the constraint, we have that will affect the others. So, our project which are modeled after figure 2, will have 7 constraints to follow.

In every system design, constraints are developed. Constraints are limitations that makes it impossible to do everything you want within the project and can affect the design. The constraints considered for the design are: economic, time, safety, health, environmental, ethical, social, political, manufacturability, and sustainability. When constraints are identified and developed, then the bounds of the project are set, and it allows us to carry out what is realistic within the time limit

imposed. By trouble-shooting our programs and testing the device extensively to have a clear understanding of its limits, we can see whether those will be appropriate enough for the final design.

4.2.1 Economic and Time Constraints

The economic constraint is our main constraint to the project, since the entire system funding to this feasibility study is solely from the team without third party monetary funding. Many Senior Design projects are funded by organizations or companies, however as we are not, the scope of the project is limited due to the limitation of available funds. That is why, as sponsors of the project, we have a budget maximum which is $200. We will try to find a way to incorporate as much functionality, in the most optimal microcontrollers and sensors we will find. Additionally, one of the motives for this project is to combat the already expensive existing 3d scanners that exist in the market which run well into the thousands of dollars range.

Time constraint is another factor that we need to consider. As this is a feasibility study, it is possible that we could accurately map out a project plan that would feature elements within the design that would exist solely if time was not a constraint. However, since we have Summer, which is the shortest semester, and Fall to complete this study, we have to minimize our stretch goals. There were several possible additions to the project that could have enhanced functionality, such as a real time display viewer within the device itself, which was not added due to the time factor.

Another time constraint that was needed to be reviewed was the time it took to have an image available to be viewed after capture. The duration of the capture must be balanced with image fidelity, so that no quality compromises will be taken over the factor of time. Since the sensor will capture depth measurements sequentially, the time it takes to generate a scan is the refresh rate of a single ultrasonic sensor times the total number of sensors. Therefore, the time it takes to complete a scan will be dependent on the number of sensors needed to accurately measure the depth of an object, while factoring in that each sensor will multiply the time it takes to capture an image. This relationship must be optimized to meet the requirement specifications regarding the scan duration. We must compute the project, test the device and get it ready for the presentation. Therefore, this device should be ready for presentation by the end of the fall semester which is mid-December 2021.

4.2.2 Safety, Health, and Environmental Constraint

We will also be focusing on safety, as the device will have electrical components and we will ensure that our focus will prioritize a user-friendly and safe device. We

must make sure all the components in the design are compatible with each other and minimize the risk of components and battery from catching on fire, exploding, etc. Accurately matching the input voltages and currents in tandem with the battery is a precaution needed to take to prevent the aforementioned safety risks. Safety is an important constraint that we must consider and not ignore as it is not a secondary concern. When selecting the individual parts of the 3D scanner, the economic constraint must not influence favoring affordability over some necessary safeguards.

Along with safety, health is another constraint imposed on the design of the project. Creating a project with materials and/or techniques that deem a health risk in the production of the device must be handled carefully. If the device causes harm during operation, then it is a health concern to either the operator or individuals in the vicinity of the device and must not be implemented.

The 3D scanner will be used indoors to capture scanned images and will not be affected by the weather as an environmental constraint. Therefore, common ultrasonic sensor obstructions such as snow, dirt, and condensation from high moisture environments such as Central Florida will not limit the functionality of the device. The device must be situated in a way during capture those external vibrations will not affect the reading as it will lead to a defective reconstructed image. Any small but meaningful movements can limit the accuracy of the ultrasonic sensors. The battery of the device may be connected to a large load, so the thermal properties of the system must be considered as the operating temperature of the components may exceed the maximum values. If the unit operates with high temperatures, then the issue of safety comes into play as the device might injure the operator as it might be too hot to handle.

4.2.3 Ethical, Social, and Political Constraint

A common ethical constraint when it comes to capturing and/or scanning images is privacy and how it can affect the user. Since the images being captured are wirelessly being uploaded to an external program, the sensitivity and privacy of those images need to be considered. The program that runs with the scanner in tandem will not use any outside network connection to limit the concern of information being stored externally. If the data was stored anywhere that was not locally, then the user should be informed what is being stored and where it is being store as it is the issue of consent. To ensure no breach of privacy is taken, the software will be coded to ensure proper security when it is being utilized in the local network via Wi-Fi communication.

The idea that a 3D scanner could have any political constraints might sound preposterous. However, if you think about it, the future of 3D scanning could lead to political constraints. If the future creates better 3D scanning, we would be able to record video in 3D. If we are able to record 3D images of rooms, some people

might be able to record entire rooms, which if left in the hands of people who want to spy on politicians, might be able to record state secrets on documents or words spoken. Now this may sound ridiculous, but the advancement of technology is unpredictable, and if the market wants more in 3D recording, then it could become a possibility. The constraints 3D imaging would have politically would be the same as normal imaging or recording. There can't be any cameras in classified areas, and there would have to be specialists to make sure meeting rooms don't have any cameras to record or photograph the area.

4.2.4 Manufacturability and Sustainability Constraints

Due to our project not being a final product designed for consumer use, manufacturing a product for a target audience is not a design constraint. Therefore, the flexibility of selecting parts is much greater – and thus we can be more financially conscience. When choosing materials for the product, the general availability of the material and/or component can impact how quickly we can develop and test the design. For PCB manufacturing, we cannot ensure that the quality of the board will be manufactured up to standard for what is required. Therefore, ordering multiple boards will enable us to select a board that is up to standard and may give us multiple unused PCB boards in the event of error during design.

Constraints regarding sustainability involve constraints that limit the maintainability of the device. The life cycle of a device is related to sustainability as a prolonged period of device maintenance can extended the average life cycle. The life cycle is a 5-stage cycle consisting of purchasing the device, initial installation, maintenance, device removal from service, and then the final disposal. By having each party individually replaceable and modular, we can modify, replace, or exchange each part to extend the duration of operation. Selecting parts that require minimum maintenance can reduce future costs as they can become more expensive to replace. Parts that are more obscure to acquire can severely limit the life cycle, as the notion that eventually these obscure parts can become obsolete due to them not being profitable enough for the manufacture to continue producing.

Our 3D scanning device will be designed with sustainability into consideration. Since all constraints intertwine with one another, having cheap, common components that can be simply replaced will satisfy our economic and sustainability constraints. The device will be expected to be very user friendly, light weight meaning less than 5 pounds, and cost no more than $200. The device will be expected to be durable, as we understand the fact that the sensors and MCU could be prone to damage due to the extensive use that the device will have.

5. Project Hardware and Software Design Details

Because our project is going to consume power and it is also a Software project, we will need hardware to run it. But first, what is hardware itself? Hardware is any physical device used in or with your machine. So, we can say by the definition of software, it is a collection of codes installed onto your computer's hard drive, and all software uses at least one hardware device to run. Furthermore, Hardware refers specifically to electrical devices. There is also some similarities and differences between hardware and software. In another word software design is the process of envisioning and defining software solution to the sets of the problems we can also say solving the stakeholders.

Therefore, our software process will be a sequence of steps that enable the design to describe all aspects or all stakeholders of the software for building. So, there are some concepts to follow for software design because those concepts will provide us as the designers with a foundation from which more sophisticated methods can be applied. Those concepts involved Abstraction, Refinement, Modularity, Software Architecture, Data Structure and Information Hiding to name a few of them.

Abstraction
This concept is the process or result of generalization by the information content of an observable phenomenon. This will help us to retain only information which is relevant for a particular purpose in our project.

Refinement
Refinement is the process of elaboration. Therefore, a hierarchy is developed by decomposing a macroscopic statement of function in a stepwise fashion until programming language statement are reached. So, in our programming language either we choose to with C or java, each step, one or several instructions will decompose into more detailed instructions.

Modularity
For Modularity it will just a part where our software Architecture will divide into component called modules.

Software Architecture
Software Architecture refers to the overall structure of the software and the ways in which that structure provides conceptual integrity for a system. So, we can say that by making the software Architecture good that will yield a good return on investment with the desired outcome of our project. This goes with performance, quality also cost as this is the main reason why we are doing this project.

Data Structure
When it comes to data structure, it is just a representation of the logical relationship among individual element of data that will have in our project.

Information Hiding
For the information hiding, it will be a part in our program where each module will be specified and designed so that information contained within a module in inaccessible to other modules that will have no need for such that information.

Now for the Hardware design, as we already mentioned previously, to add more about we can say it is the description of the hardware on which the software resides and how it is to be connected to the system or plant equipment of our project.

5.1 Initial Design Architectures and Related Diagrams

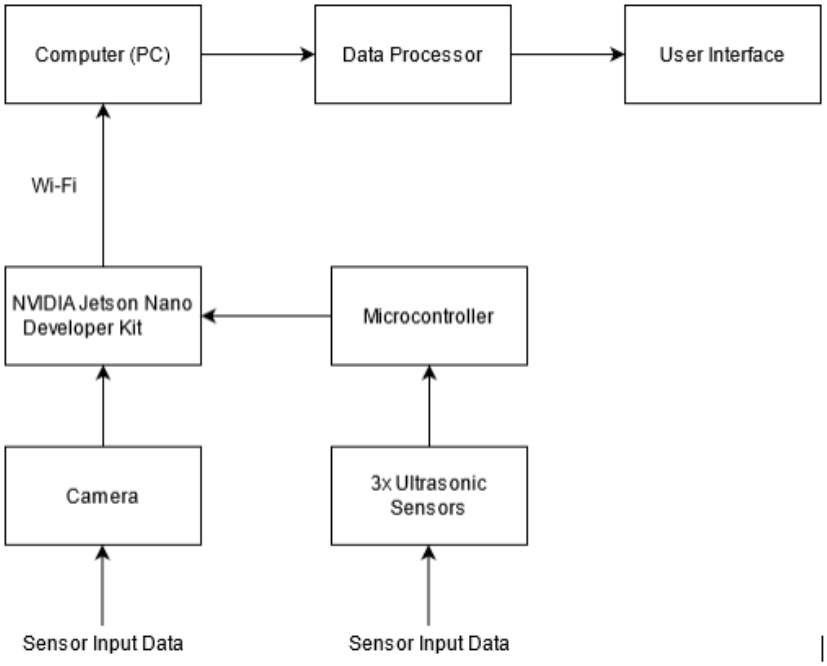

Figure 10: Block diagram for 3D scanning device

RYAN HAMADA - Nvidia Jetson Nano, User Interface, Image Processing Algorithm

JOHN PASZYNSKI – Ultrasonic Sensors, Microcontroller, Power System

SERGIO ARCINIEGAS – Nvidia Jetson Nano, Data Processor, Image Processing Algorithm

JEAN CESTIN – Raw data, Image Processing Algorithm, Image Display

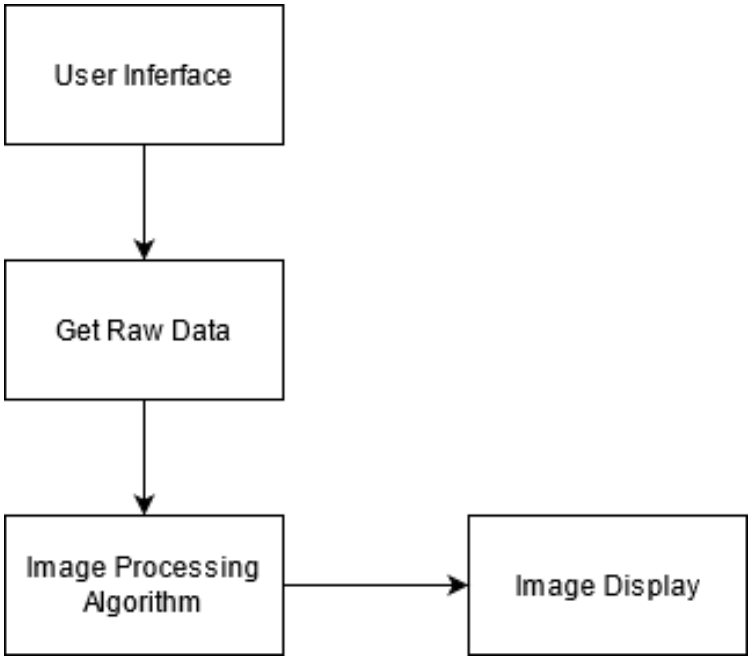

Figure 11: Software block diagram for 3D scanning device

NVIDIA Jetson Nano Developer Kit B01*:*
The Jetson Nano controls 2 parts of the system. First and foremost, it interacts with the user to allow them to start/end the scanning process using an interactive button. Once scanning is initialized and completed, the data is sent back from the camera and microcontroller to the Jetson Nano where it is sent to a remote PC via Wi-Fi.

Power Supply*:*
The power supply unit is a simple DC 5V power supply that provides power to the camera, microcontroller, and any hardware components so that it may operate independently as a handheld device.

Camera System*:*
The camera system uses a stereo camera that can provide supplemental depth image as well as high resolution images to use in tandem with the ultrasonic sensors.

Data Processor*:*
The use of the data processor is to collect and manipulate the raw data provided by the device/hardware so that it can be interpreted by the software. But first what is raw data? Raw data is sometime called source data or primary data, it is data that has not been processed for use. A distinction is sometimes made between data and information to the effect that information is the end product of data

processing. The data processing unit separates images, removes background and noise, and then reconstructs the image into a 3D model.

***User Interface**:*
The user interface is a software package that permits the user to manipulate depth sliders within the captured image so that it may be cropped from in front or behind the object to get an accurate scan. The interface will have an interactive 3D viewer as well as a file manager for unwanted captures.

5.1.1 Network Design

The purpose of this portable 3d scanner is the ability to view 3d images of pictures taken by the cameras. In order to view these images, we will be taking values from the sensors and camera and placing them into a text file within the memory of the board via SD card. This SD card will be able to upload to a computer so that the user can extract the files and use software to render the images and create the 3d view of the images. The type of file that we will be using is an STL file format. The STL file format is one that is native to stereolithography CAD software. This software was created by a company called 3D systems. STL stands for Standard Triangle Language, as the file is collecting 3 points for an image. STL only represents the surface geometry of a three-dimensional object without representation of its color, texture etc.

An SD card is a non-volatile memory card for use in portable devices. SD cards are able to support various bus types and transfer modes. The one that is mandatory is the SPI bus mode and having one-bit SD bus mode. For SPI bus mode the bus is able to support only a 3.3 volt interface. For this type of host no host license is required. For initial communication between host devices and the SD card a synchronous one-bit interface is used for communication where the host device is providing a clock signal that strobes single bits into and out of the SD card. The host device, in our case which will be a laptop, will be sending a 48-bit commands and will receive responses by the SD card. The host device can actually gather from the SD card the type of card, the memory capacity, and the capabilities of the card. Additionally, the host device can issue a different voltage, different clock speed and create an advanced electrical interface. This could be useful in situations where a design can be made where the SD card is connected to the laptop where live rendering of a 3d scan can be examined for ultimate precision, so in that case the SD card would have to be communicating at a much higher rate to the host device. The SD card has a block-accessible storage device where the host can read or write fixed-size blocks by specifying.

5.2 First Subsystem, Breadboard Test, and Schematics

The first subsystem of this design is the ultrasonic sensors and microcontroller. The Adafruit Feather M0 was chosen for the project primarily for the light weight, memory capacity, and small dimensions. The HC-SR04 sensors each run on 5V logic while the Adafruit Feather M0 requires 3.3V and nothing more. Since we are using one battery supply to feed 5V of power to the entire system, we need to match each component's voltage. The MicroUSB port on the microcontroller will regulate 5V USB input voltage down to a stable 3.3V. Each sensor will be connected on the 5V/ground rails of the NVIDIA Jetson Nano. When the ultrasonic sensors are initiated, each will operate sequentially to prevent crosstalk. We will use the Adafruit Feather M0 to send an input Trig signal at 3.3V. The ECHO will be 0V when it is triggered until it finds a return pulse that will initiate it to 5V for the time the sensor outputs a pulse to the return pulse being registered. While the Trig signal can be either 5V or 3V, the Echo signal on the return is 5V. Therefore, by using two resistors at 10K each, we can convert the 5V logic into 2.5V that the Adafruit Feather M0 can handle as the GPIO pins are rated for 3.3V input.

Figure 11 below represents a visual breadboard diagram of the ultrasonic subsystem. Currently the TRIG and ECHO pins are connected to 6 GPIO pins, where the microcontroller will carefully select each sensor to operate one at a time. As the cone of propagation of each ultrasonic sensor can interact with accuracy past the deadband, we must limit each sensor to operate independently and consecutively.

Another design solution would be to connect each ultrasonic sensor to its own de-mux/control combination. The control switch will initiate which de-mux will operate, therefore queueing each operation. This is a solution that would transition a software method of eliminating noise to a hardware solution

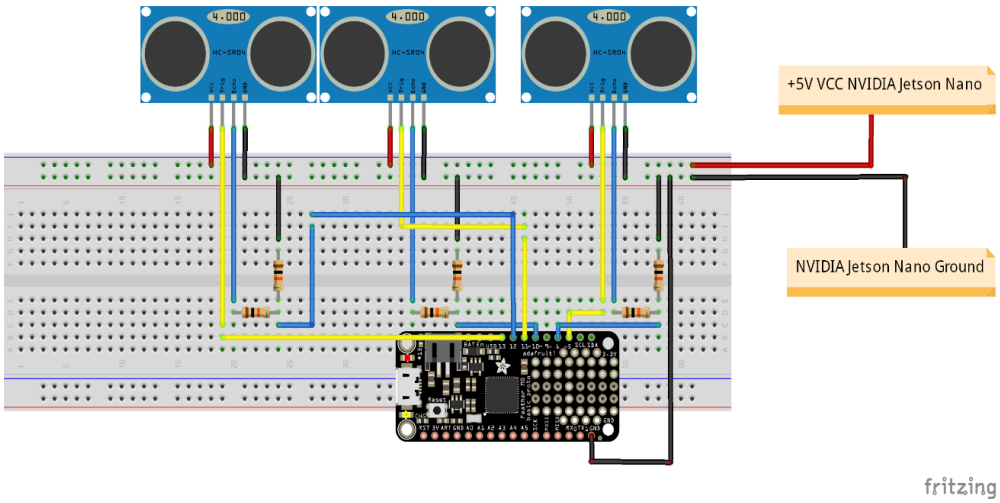

Figure 12: Breadboard Diagram of Microprocessor Subsystem

The HC-SR04 sensor has four pins: VCC (5V power supply), Trigger input (TRIG), Echo Output (ECHO), and a ground pin (GND). The VCC source and GND, which are connected to each ultrasonic sensor's VCC and GND pin are connected to the Jetson Nano's 5V logic. Each sensor's Trig pin is directly connected to pins 5, 11, and 13 to the Adafruit Feather M0 as they can receive the microcontroller's 3.3V input. Two resistors of 10K values each, are connected in series to the ECHO pins of each ultrasonic sensor. By using a divider bridge, we can limit the 5V signal to 2.5V and prevent damage our unprotected microcontroller 3.3V input ports. The microcontroller will receive input data from the sensors and relay that data to the Jetson Nano. The Nano will interface the ultrasonic sensors and camera and transmit the data through the Wi-Fi module.

Pictured below is the schematic for the ultrasonic subsystem. Note that in the PCB design the HC-SR04 sensors will not be included on the board. The sensors will be replaced by three separate 4 pin headers which will allow us to connect the sensors to the PCB without limiting the range of motion that soldering will impose to the design.

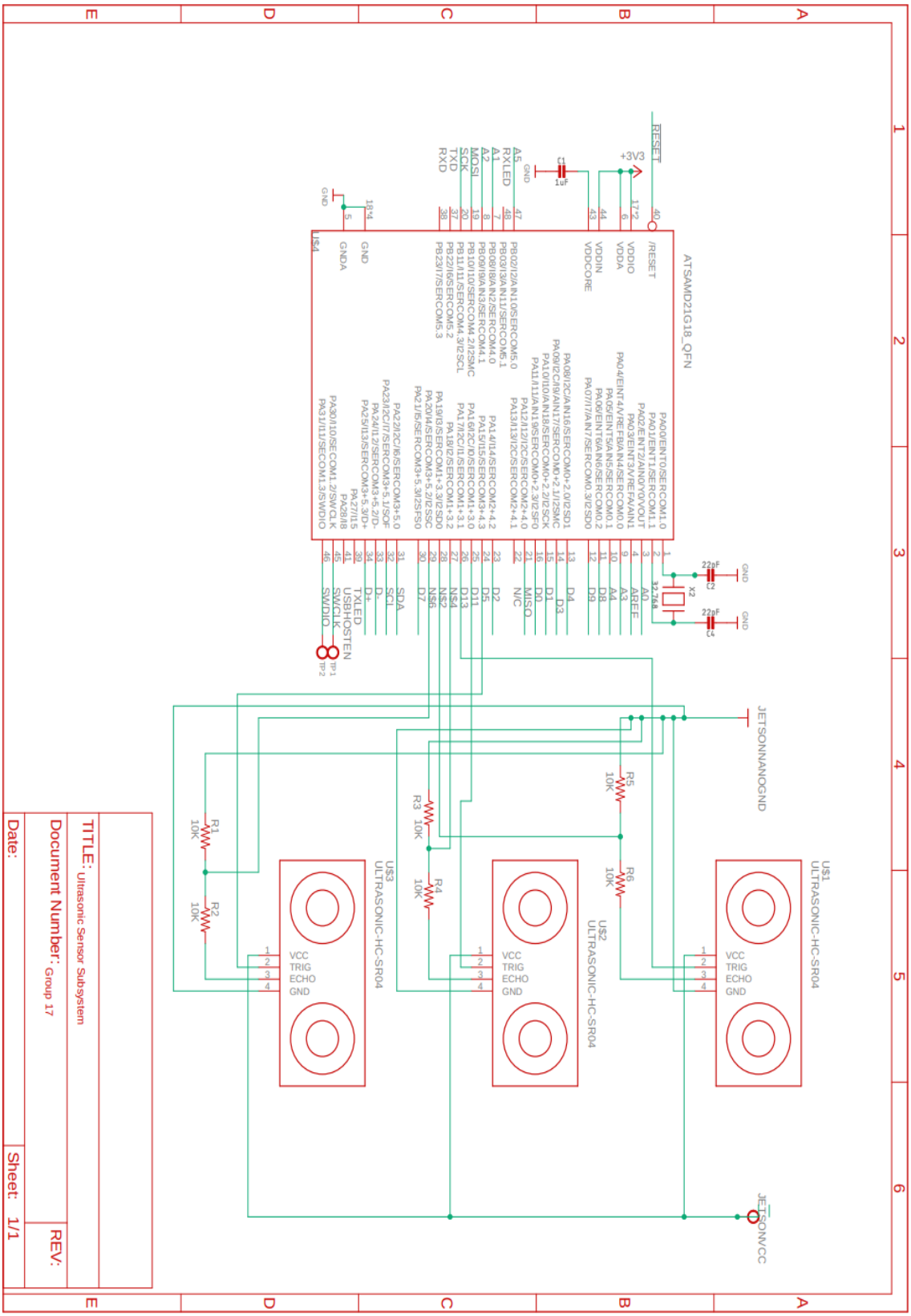

Figure 13: Schematic of Microprocessor Subsystem

In Senior Design 2, our design implementation and vision changed. Rather than having 3 ultrasonic sensors, we have opted into using one sensor while having three sensors as a stretch goal. Three sensors would allow us to narrow the field of view of the ranging sensors, thus creating a more high-definition image. With multiple sensors, the issue of crosstalk is an element that can't be ignored and must be considered. Given the duration of capture of a single ultrasonic scan, multiple ultrasonic sensors would extend the duration by a multiple of the existing update rate of the HC-SR04. The drawback to three sensors rather than one, is the increased development time, thus it must be our stretch goal.

Rather than utilizing the ATSAM microprocessor, pictured below is a total redesign of the project that was issued with an STM32 based PCB design with the STM32F103C8T6 in the forefront.

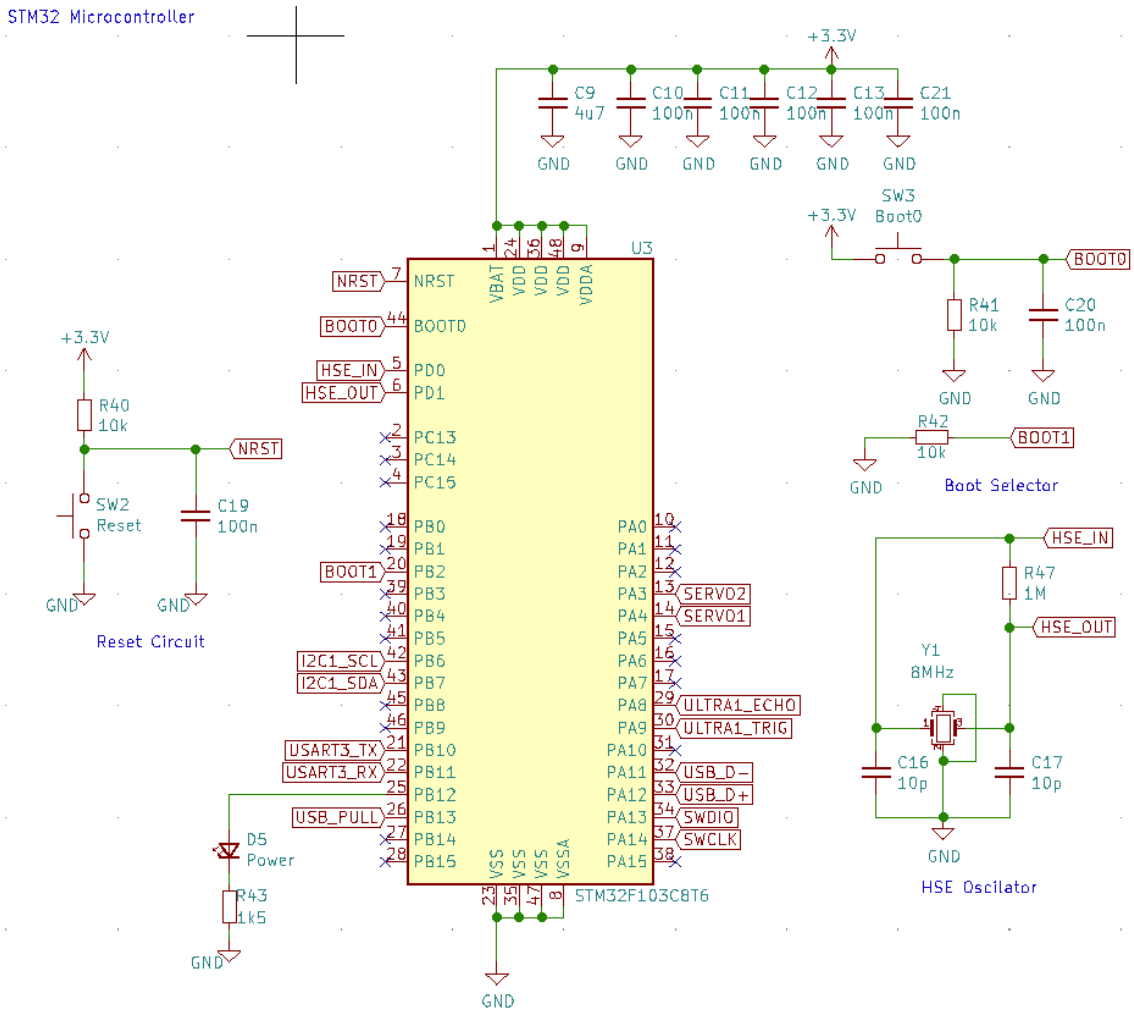

Figure 14: Updated Schematic of Microprocessor Subsystem

The STM32 power pins need to be powered via the 3.3V rail. VBAT is utilized if we require a backup battery power source, however for our application, we are not using the VBAT pin, therefore it is pulled up to 3.3 volts. The MCU requires not only the power pins to be pulled to 3.3V, but also requires decoupling capacitors very close to the chip. A 4u7F capacitor would serve as the bulk decoupling capacitor following 100nF capacitors per each VDD/VDDA pin. A switch is connected to the pin 7 of the MCU for a hard reset during debugging, when held low. According to the STM32 datasheet, there are three boot modes available to select. Boot from user flash, system memory, or embedded SRAM are the boot mode options. By pulling the BOOT0 pin to ground or high, we can select the appropriate modes. When BOOT0 is selected low, the chip will initiate a run state where it will initiate the programming code. On the contrary, when the pin is pulled to 3.3 volts, we are able to program the chip via USART, USB, etc. By using a push down switch, we can essentially have the BOOT0 pin constantly low unless it is held down to 3.3V. If we want to program the board with USART, then the BOOT0 pin would be required to be switchable.

Another piece of required circuitry needed for the microcontroller to operate is the HSE oscillator. 16MHz crystals were considered, but 8MHz crystals are found to be a more common component, therefore an 8MHz crystal with a GND24 package was selected. GND24 fundamentally has pins 2 and 4 grounded while the actual crystal pins are 1 and 3. Two load capacitors are needed to run the crystal as well as a feed resistor. The load capacitors depend on the perimeters of the crystal datasheet as well as the stray capacitance. The load capacitance is determined by the following equation:

$$Cload = 2 * (CL – Cstray)$$

The load capacitance of the selected crystal is 10pF, while the stray capacitance can be between 2-10pF, roughly somewhere in the middle with 5-6pF will be sufficient to operate the microcontroller. By utilizing the equation, we can place two 10pF capacitors connected to pins 1 and 3 of the crystal. A 1M ohm parallel resistor is placed across the crystal pins to bias it as many microcontrollers have this internally, but the STM32 lacks it. On PB12 pin, an LED with a pulldown resistor is added to signify that the MCU is powered.

5.3 Second Subsystem

Power management in any system is imperative. Without the correct power distribution, the components can become compromised to damage, such as in a mismatch of voltage, and must be balanced to keep your electronics running smoothly. There are two main boards required to be powered. The first one being the NVIDIA Jetson Nano B01, which can be considered the main distributor of the system. The Nano has a dedicated barrel jack connector which is an extremely common component in power design. The barrel jack is a low voltage DC power

connector that has manufacturer specified voltage and current ratings for various applications.

From the datasheet of the Nvidia Jetson Nano, the Nano takes a 5V input at 2-4A depending on the power mode the processor is in, which can be low or max mode. The microcontroller will be interfaced via USB 3.0 with the Jetson Nano. According to specifications, USB 3.0 delivers 0.9A per port, which is enough to power the Adafruit Feather M0 microcontroller and requires us to consider an extra 1A when designing the power system.

By using Webench Power Designer on Texas Instruments website, we can create a power supply circuit based on the requirements of the project. A DC/DC converter would be utilized that would input from the battery and output to the entire system. A consideration to make when designing the power supply circuit is that the voltage output of a battery is not a constant voltage. The displayed battery voltage is not the actual voltage at any point in time, but the resting voltage. The actual voltage may be high or lower depending on the state. To combat this, we will design a power system where the input voltage is +- 2 volts. When all variables are inputted and Webench provides choices for a design, selecting a low BOM count will allow minimized costs when sending the design to the PCB vendor. Not only would BOM count be important, but as well as keeping the BOM cost low as each extra board ordered will be more expensive to order if this cost is high. Power efficiency is the third important aspect in choosing the correct design to be utilized. With every circuit there will be power loss within the circuit itself, thus selecting 90+% efficiency is vital so that excessive heat and power loss is minimized.

The hardware components in the table below provide the total power draw of each component within the project. While the total power that is being used in the system is dependent on the application and its resource utilization within the NVIDIA Jetson nano, a liberal estimate is used to ensure that each component has enough power that it can draw. Note that the HC-SR04 has 3 components, while the data is for a single sensor, the total power of the system is calculated with 3 HC-SR04 components.

Component	Operating Voltage	Operating Current	Total Power
NVIDIA Jetson Nano	5V	2.5A	12.5W
IMX219-83 Stereo	1.8V	1.23mA	0.0022W
Adafruit Feather M0	3.3V	12mA	0.0396W
HC-SR04	5V	15mA	0.0075W
		Total:	12.5643W

Table 13: Power Management

The power supply circuit design that best fulfils the requirements of this project is the Texas Instruments TPS56637 shown below in Figure 15.

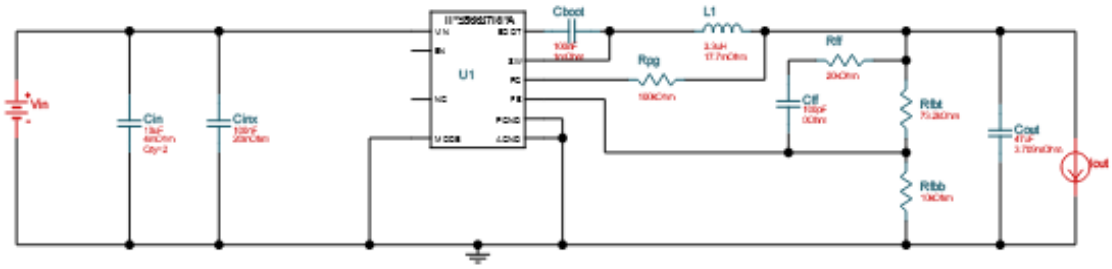

Figure 15: Texas Instruments TPS56637 buck convertor (Courtesy of Texas Instruments)

The Texas Instruments TPS56637 is a synchronous buck convertor that can input 4.5V to 28V and output at 0.6V to 13V. The maximum output current of this buck convertor is 6A which is plenty if the Jetson Nano runs at max mode at 5V/4A with the other power drawing components. At 94.6% power efficiency, this makes it a suitable design for a power system while maintaining a low BOM count of 12 and BOM cost of $2.49. The total footprint of this layout and design is 173 mm^2. In Figure 16 below, we can see that the steady state voltage at the output of this buck converter is approximately around 5V, which is what is needed to power the Jetson Nano.

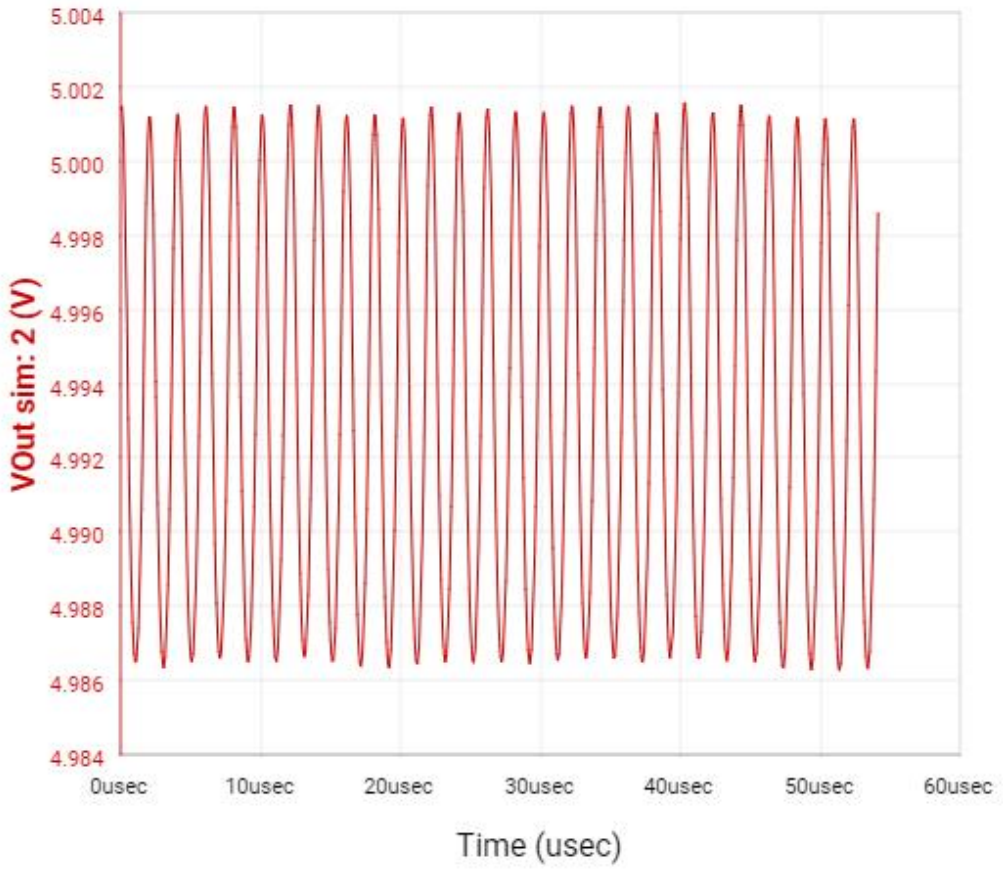

Figure 16: Texas Instruments TPS56637 steady state output (Courtesy of Texas Instruments)

For the PCB design of the power supply system, we will use Autodesk EAGLE which features a free version available to students to utilize. The free version allows us to use 80 cm^2 PCB designs with 2 layers at no cost. The TI TPS56637 will be mounted on a printed circuit board with soldering pads to solder a barrel jack cable on the output. The barrel jack cable will then run to the Jetson Nano's female barrel jack connector to power the system. The Adafruit Feather M0 will then be powered via USB 3.0 from the Jetson Nano as the Feather M0 can regulate USB 3.0 to 3.3V. Due to USB 3.0 having a limit of 0.9A, we must power the HC-SR04 ultrasonic sensors from the power distribution circuit directly. Since the echo signal of the sensors will be 5V, the voltage must be regulated once again so that the microprocessor does not burn out due to the higher voltage.

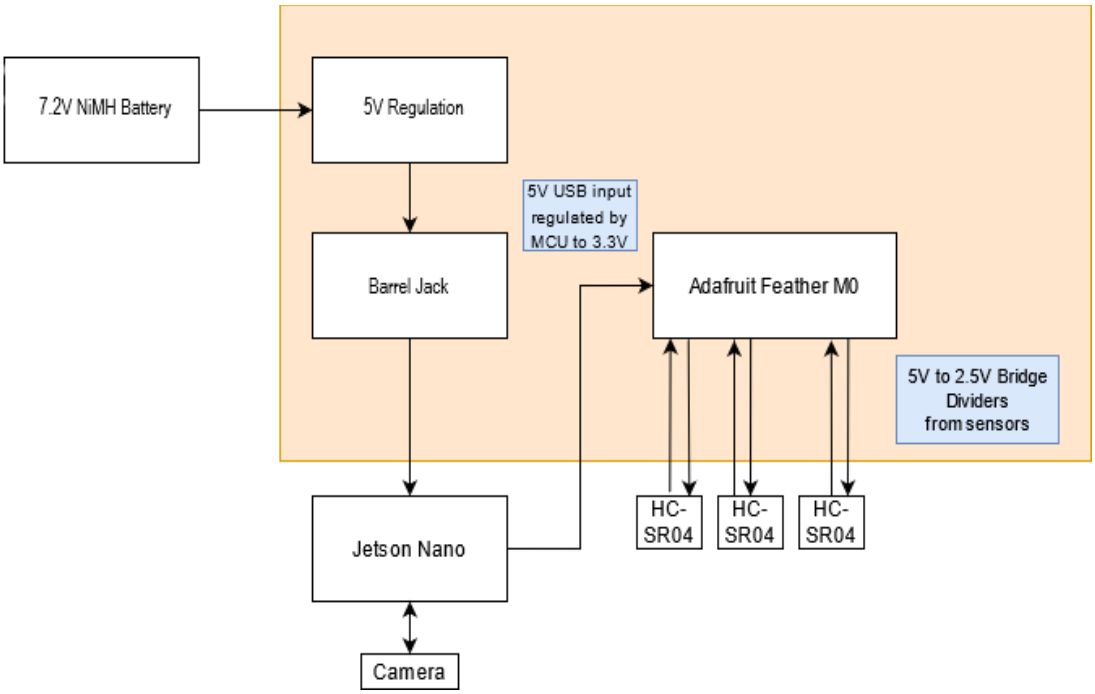

Figure 17: Power management system on PCB

In Senior Design 2, since we changed the overall design, new power options need to be considered to consider the new needs for our system. Rather than a 7.2V NiMH battery, we instead shifted to a 12V battery as there were more buck converter options available for purchase. As well as a buck converter change, another buck converter was added to regulate from 5V to 3.3V for the on board MCU. The following changes can be seen in the figure below.

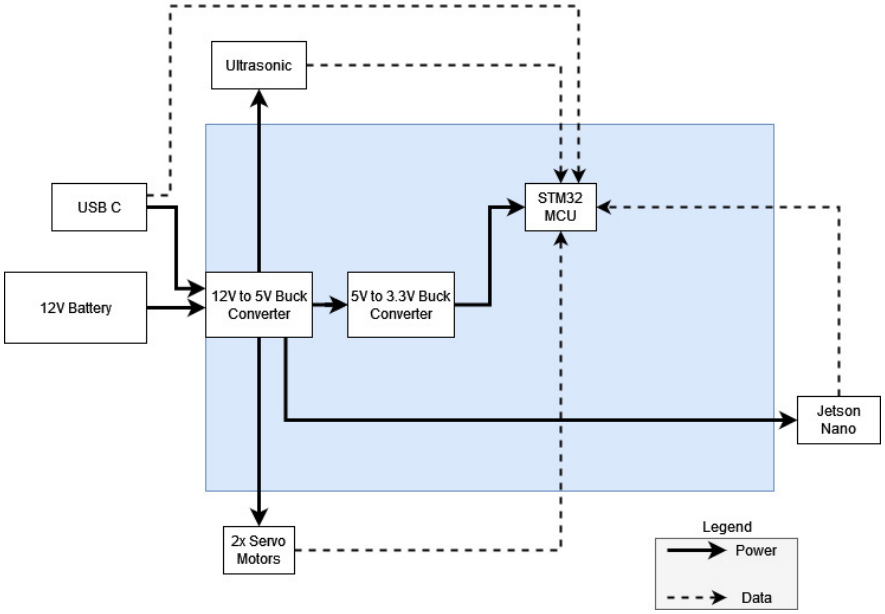

Figure 18: Improved power management system on PCB

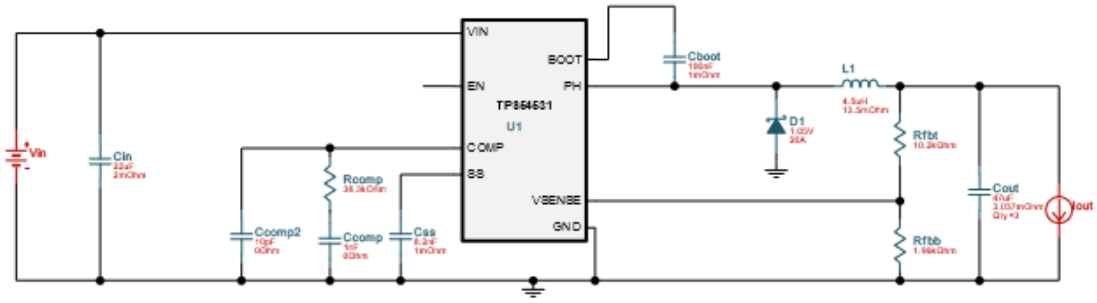

Figure 19: Texas Instruments TPS54531 (Courtesy of Texas Instruments)

The Texas Instruments TPS54531, shown above in figure 19, was chosen as the main buck converter of the system. The input voltage range of this chip is 3.5 to 28V with a minimum output voltage of 0.8V. This buck converter was chosen due to it delivering 5A of current as well as being designed to produce a maximum of 5V voltage. Unlike other common buck convertor options on the market, the TPS54531 features a ground PowerPAD under the chipset that needs to be fixed to the ground plane of the PCB when designing the PCB layout. In addition to the buck converter, a safety feature needed to be added to prevent the entire PCB from being fried due to connecting the battery the wrong way. A P-channel MOSFET was added as reverse polarity protection to the input of the design. The drain of the MOSFET is connected to the screw terminal of the battery, while the gate was pulled down to ground with a 10K resistor and the source was connected to the VIN pin of the TPS54531 converter. When selecting the P-channel MOSFET, the gate-source voltage needs to be selected carefully. According to the datasheet of the Si4435DDY MOSFET, the Vgs is +-20V, which is plenty as we are only connecting a 12V battery. If the gate-source voltage is less than the input, then the MOSFET will burn out.

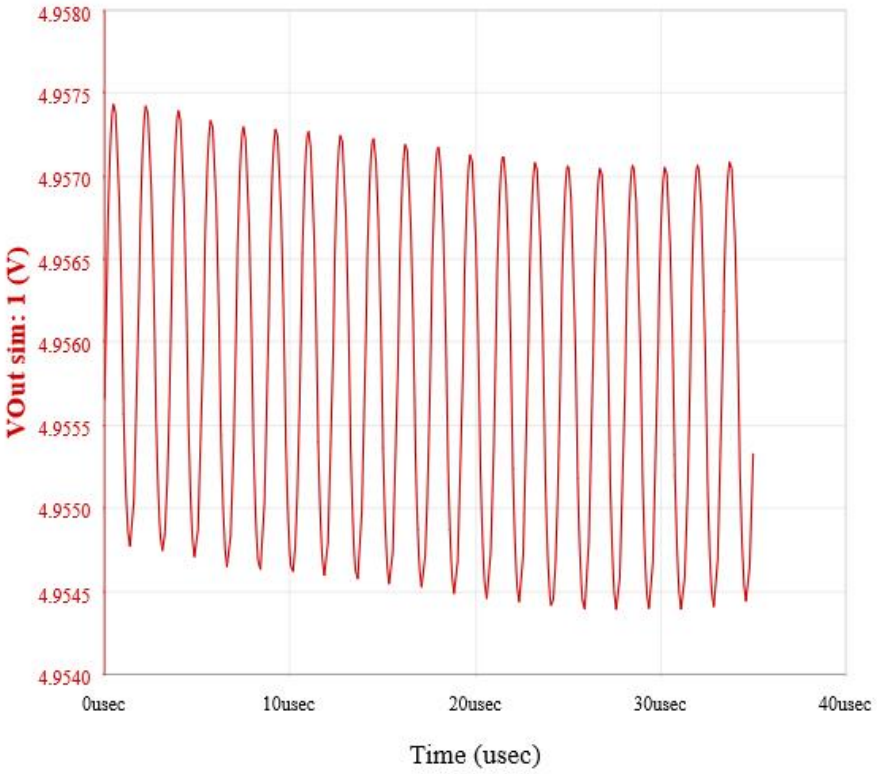

Figure 20: Texas Instruments TPS54531 steady state output (Courtesy of Texas Instruments)

The Texas Instruments TPS54531 is a non-synchronous buck converter that can input 3.5V to 28V and output at 0.8V to 30V. The maximum output current of this buck convertor is 5A which is plenty if the Jetson Nano runs at max mode at 5V/4A with the other power drawing components. At 83.8% power efficiency, this makes it a suitable design for a power system while maintaining a low BOM count of 14 and BOM cost of $3.06. The total footprint of this layout and design is 514 mm^2, which is almost 5x the original design. In Figure 20 above, we can see that the steady state voltage at the output of this buck convertor is approximately around 5V, which is what is needed to power the Jetson Nano.

The other buck converter necessary for our design is the TLV62569 from Texas Instruments, pictured in Figure 21. This synchronous step-down buck converter inputs 2.5 to 5.5V and outputs from 0.6V to Vin. This design is an extremely compact design at only 87 mm^2, which is necessary given the large footprint of the TPS54531 already on the board. At $0.69 BOM, this circuit is extremely cheap as well and provides 91.8% efficiency. The 3.3V output of this buck converter is necessary as the MCU on the board needs to be powered via this voltage. Due to the PCB being 4 layers, one layer would be reserved for the 3.3V output of this buck and another is the ground plane of the system.

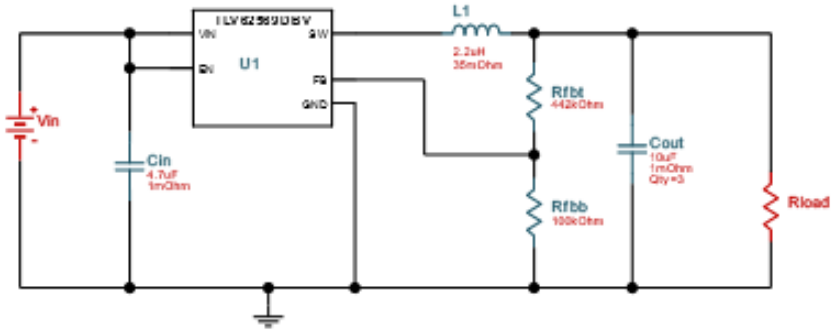

Figure 21: Texas Instruments TLV62569 (Courtesy of Texas Instruments)

In Figure 22 below, we can see the stead state output of the Texas Instruments TLV62569. The output voltage average is roughly 3.25V, which in practicality, is enough to power the 3.3V STM32 microcontroller within the PCB.

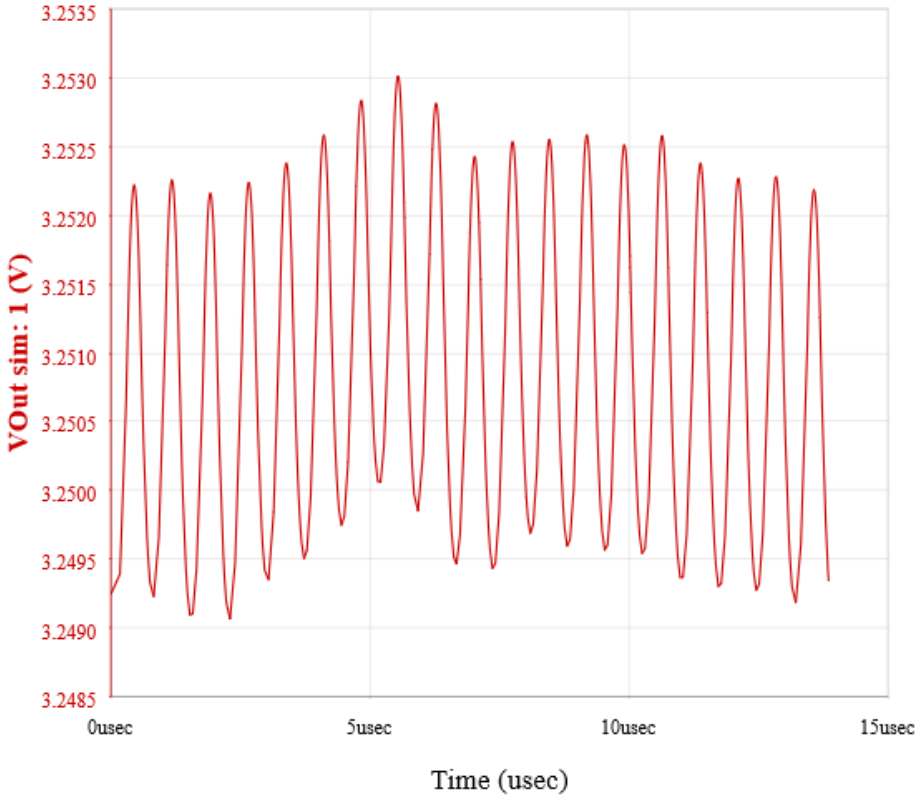

Figure 22: Texas Instruments TLV62569 steady state output (Courtesy of Texas Instruments)

5.4 Third Subsystem

The third subsystem of the hardware design is the USB-C circuit. According to the USB hardware and PCB guidelines for the STM32 MCU, the STM32F103 line of microcontrollers only supports USB A. However, other USB interfaces can be integrated with the STM32, if the appropriate circuit is designed to accommodate it. The USB D+ pin needs to be pulled up to 3.3V with a 1.5K resistor to be compliant with the electrical specification of the USB. Rather than pulling the resistor to the voltage rail, connecting it to a spare GPIO pin on the microcontroller allows the MCU to decide when it is ready to speak to the USB or force a disconnect/reconnect event. Two 5k1 resistors are needed for the CC pins, as well as completely being separate from each other. If we hook the resistors otherwise, then the microcontroller will not read the resistance of the other pin and form an equivalent resistance of 836 ohms and fail to provide power to VBUS from the source.

Since a USB port is constantly disconnected and connected, an ESD protection circuit is necessary to prevent the build-up of static charge that will form from the use of the port. When adding a USB port to a PCB, the ESD protection circuit is a necessity rather than an option.

In case the battery goes below 5V, the USB fuse will pop due to MOSFETs conducting current in both directions when on. To prevent the fuse from blowing, we connected the FET gate to VBUS instead of ground and made sure there was a pulldown on VBUS. By sensing VBUS and driving the MOSFET gate with its output, we can also prevent the MOSFET from burning when the battery is around 6V and USB is connected. This is due to when Vbat – Vusb is within the MOSFET sub-saturation range.

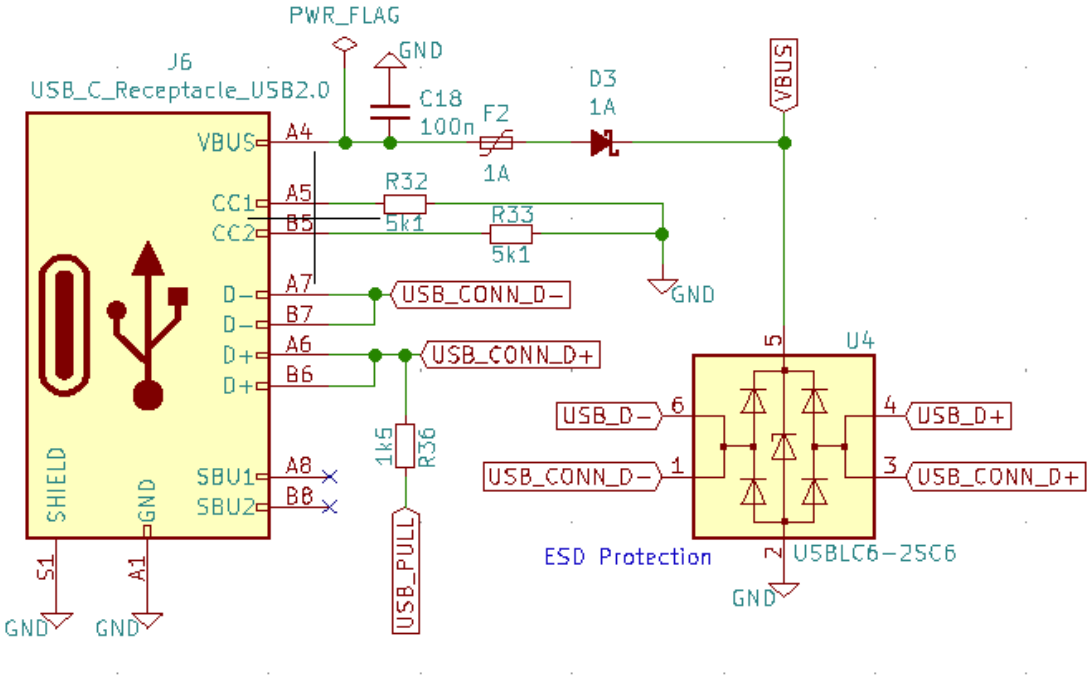

Figure 23: USB-C Subsystem

5.5 Software Design

There are three general ideas on how we can feasibly create a 3D scanner using an ultrasonic sensor. One way is to have three (or more) ultrasonic sensors scanning multiple points at a time. The second plan is to have one ultrasonic sensor scan multiple points by angling itself to scan in a grid fashion (as in it would scan the distance at point (0,0) then (0,1), etc, to then be used in making the 3D image). The third plan is to turn the portable 3D scanner idea into a small turntable style 3D scanner. These three plans could feasibly create a 3D scanner using an ultrasonic sensor.

The first plan, the multi ultrasonic sensor plan, is only feasible in one way. If we were able to program the sensors so that they record the measurements in real time. Having multiple sensors would be able to scan multiple points that could be seen through the camera next to the ultrasonic sensors would show where the ultrasonic sensors are pointing to. The ultrasonic sensors would have to do multiple scans at multiple points. In order for that to happen, the ultrasonic sensor must move. The Nvidia Nano would have to, in real time, read the values of the ultrasonic sensor, track where the sensor is recording on the object being scanned, and keep track of the camera and read which direction it moves in order to keep up with where the ultrasonic sensor is pointing to.

This plan is very similar to what the Xbox Kinect does, using a camera and measuring a distance between points to create a 3D image while tracking which direction the Kinect moves to. There is one issue that greatly hinders us when it comes to what hardware we have. The Kinect is able to scan the distance of any point at any place in the Kinect's field of view, while no matter how many ultrasonic sensors we add onto the scanner, we would only be able to scan a few select points. This would mean the user would have to move the ultrasonic sensor up and down multiple times to get the 3D image desired. This would not make our device a favorable alternative over other more expensive scanners which would be much easier to use.

The second plan was to have the scanner sit on a stand (or tripod) and move the ultrasonic sensor so that it can scan the points in a grid and record the distances of each point in the grid so that the grid could be turned into a 3D image. This (in a sense) would be doing the movement for the scanner that the user would have been doing in the first plan explained in the paragraph above. Only this time, the scanner would accurately be moving up and down, left and right, in order to accurately get each point for the 3D image. This would be (somewhat) like what the portable scanners like the Einscan or the MetraSCAN do, because they use blue lasers in a grid like fashion in order to record points of an object to then turn into a 3D image. The only difference would be they can scan 500,000 to 1,500,000 points per second, and we would only be able to scan 1 or 2 points per second. This however would be a major improvement to the user experience, making it so that the user doesn't have to do a ridiculous dance every time they want to create a 3D image.

There are two issues with this plan however. One, if our scanner tilts the ultrasonic sensor in any direction it can (up, down, left, or right), the sensor would be reading the distance of the object at an angle. This problem is extremely simple to solve (although it may be a little tedious). The way we would solve this problem is by using what is known in the engineering business as high school trigonometry. If the angle is known (I'll get to that part in a bit), we could use sin (theta) = (x)/(hypotenuse) or cos (theta) = (x)/(hypotenuse) to find the missing x value. The ultrasonic sensor would be scanning the hypotenuse of the "triangle" and with a predetermined angle, we can easily get the accurate distance in relation to the center point of the object. The equation would probably have to be run twice in order to get the accurate distance. Getting the predetermined angles will be the tedious part. In order to get those angles, we would have to set the ultrasonic sensor in front of a flat surface and record the distance it measures. Then, we tell it to move maybe 5 ticks in whatever direction. We would then measure the readings on the sensor. With the two measurements, we can find the third side of the triangle, and with that, we can determine the angles of the triangle. Divide the angle we need by 5 (because we moved 5 ticks earlier) and we get what angle per tick. This value would have to be used in the program we write as a constant variable that would be used a lot with every calculation the Jetson Nano makes.

The second issue is that in order for the image to be more accurate, the grid it scans has to be very large. The larger the grid is (as in the smaller the movements the ultrasonic tilts per tick), the more points will be scanned, which creates a more accurate 3D image. What would be amazing is if we could get around 10,000 points (a 101 by 101 grid, moving 50 ticks in any direction up, down, left, or right). This would then create a 3D image that is fairly accurate, but would require a lot of processing power that the Jetson Nano has. This would all depend on what is used to move the ultrasonic sensor (aka two servo's). Overall, the moving ultrasonic sensor plan is the middle ground when it comes to how easy or difficult a plan is.

The third plan was to scrap the idea of being able to scan farther away or larger objects, and use the ultrasonic sensor to scan small objects that rotate on a turntable. The ultrasonic sensor would have to either pan or tilt up and down in order to capture the full object in the center of the turntable. We would then have the scanner process the data from the ultrasonic sensor while the turntable is spinning so that it may create a 3D image. This would once again have to involve real time scanning like with plan 1, but unlike plan 1, the movement will be stable and consistent, which would make it easier for the Jetson Nano to read and process.

There are some issues with this plan though. This plan would require us to abandon our original goal of being able to scan large objects using a scanner more cost effectively. Being able to create a 3D scanner that can create 3D images to combat the high prices of portable 3D scanners on the market was why this project was created. Another technical issue is that we would have to then make sure the ultrasonic sensor is able to scan objects in a real time manor, which will either require an AI algorithm to handle it (which the Jetson Nano was made for), or a program that measures the distance the ultrasonic sensor detects every tick. This is doable if we are willing to completely change our original goal. In terms of plans, this one might be the easiest, for the programmers at least.

The first thing that we have to consider when designing the software is how the ultrasonic sensor will move. If we go with plan 1 (the multi-ultrasonic sensor plan), we won't have to consider how we will move the sensor because the user will do that for us. If we go with plan 2 (the plan where we scan points in a grid), the ultrasonic sensor would have to move in four directions. This would probably be done by using two servos to tilt the ultrasonic sensor in whatever direction needed. Starting from the center, we would program the servos to go to a corner of its field of view in order to have a starting point when recording the measurements of the grid. There is one problem however, if for some reason the scanner shuts down midway of scanning, the ultrasonic sensor would have a new starting point that would mess up the entire scanning process. Depending on how advanced the servos are, we could detect their position and make sure it's in the proper state when starting up. If that isn't possible, we would have to program the servos during startup so that they move as far as they can in one direction, and then head back to the midpoint so that it may be properly aligned. If we go with plan 3, we would

have to program two different types of movement. The first would be either to have the ultrasonic sensor pan up and down (which would be the best option), or to tilt the ultrasonic sensor up and down. The next form of movement would be to rotate the turntable at a constant speed for the ultrasonic sensor to record the object at all angles. The ultrasonic sensor would stay at one level, staying in place while we program the turntable to rotate 360 degrees. After the 360 rotation is complete, we program the ultrasonic sensor to move one point up or down (depending on where the sensor starts). We then tell the turntable to rotate again, continuing the pattern until the ultrasonic sensor reaches the end of its ability to move. The most important and most influential part of how this project will run is how the ultrasonic sensor will move. This also leads to how we create the 3D image of the ultrasonic sensor.

The next part of the programming design plan is how we're going to transfer data from the sensors to the Jetson Nano. Connecting the ultrasonic sensor to the microcontroller, and then the microcontroller to the Jetson Nano, we can use I2C communication to grab data given to the microcontroller to the Nano, and then convert the data into an understandable value. With that data, we can store it in different ways depending on what plan we have. If we go along with plan one, we would have a seriously difficult time storing the data. The easiest thing to do would be using the AI capabilities of the Jetson Nano to keep track of where the points are when moving in a direction. This would be extremely difficult to program, so it is unlikely we will be executing this plan without serious thought into how we can actually execute it. In the sense of plan two, we would create an array for the distances in startup. Then when we scan the distances in a grid like fashion, we can convert the values and place them one by one. They would then be used for converting points into vertices. The third plan would also be recorded in grid fashion, each point would be recorded at probably every degree of rotation or more. Each point would then also become a vertex on the 3D image to be created after any plan is done scanning their points of the object. The 3D image creation is the final and most important step in the feasibility study we are going to perform.

The 3D image creation process is going to be a very complicated process, and it's one that would apply to any plan we commit to in the near future. First off is how we're going to program the Nano to actually do anything. Because the Jetson Nano was created for AI usage, it most likely works well using python. In the testing of the Jetson Nano, we have to determine if we can actually program in C or Java. Only one member of the group has any experience in python, so in the long run it would be preferable if we could use a commonly used programming language so that everyone with programming experience can work on the project. If not, some people are going to have to take a crash course in python.

Going back to actually creating the 3D image, we have an array with many points of an objection, which means that the image will consist of vertices that connect up to create many faces on the object. In a standard .stl file, we have facets, which is a triangle made up of three points which are on an x y z plane. In order to make a square, you would use two triangles side by side, which will be important for our

project. When using a grid from either plan two or plan three, we can take four points in a square fashion(ie (0,0), (0,1), (1,0), and (1,1)) to use as reference points when creating two facets to make a square that connects to the four points taken previously. We do this multiple times to make sure the entire array is accounted for, and we then have a 3D image. Things will be different between plan two and plan three, however. Plan two is in a sense a flat image turned into a 3D image, while plan three is a full encirclement of the object in question. While plan two's grid is easy and flat so that you can just translate the points to a flat surface, plan three's grid is for a circular object, so we would have to take into account on the z axis that point A is a millimeter ahead or behind point B because the measurements weren't taken in a straight line. This can be solved by possibly creating a system that once again uses the angle difference to form a triangle where we can get an accurate location of the two points in terms of the z axis. If we measured a point every one degree turn like previously stated, we would already know the angle is one degree and we wouldn't have to do so much math to find all the correct angles and then hard code them into the system. We can find out easily where the points are on the x,y plane, the most difficult part will be the z plane, which depending on what plan, can be solved differently. The idea for plan three was already stated above, having to calculate the triangle of the two points and the center of the turntable to find out how far back the program would have to go on the z plane, but also change on the x plane as well. The idea for plan two is simple, we have the distances on each point, so then we know what the image looks like inside out. We would have to in a sense invert the values on the z plane and then create the 3D image from there. The reason why we would want to do that is because we want the farthest point of the object we're scanning to be the zero point of the 3D image. We also have to take into account the very far away values, or values where the ultrasonic sensor is just hitting air and not detecting anything. This would create a very unrealistic image. Therefore, we have to make sure when converting the distances calculated into the 3D image, we remove points on the graph that are way too far away. If the maximum distance the ultrasonic sensor can detect is 430 cm, we want to probably take in the values that are at max 400 cm away.

The difference between the 3D scanner we will be making, and the advanced (and expensive) 3D scanners on the market is that they use lasers to form a grid, and we use an ultrasonic sensor(s). This makes a huge difference when determining how to program the 3D scanner. The lasers on the advanced 3D scanner form a grid. In real time, the points of the object are measured, and because the lasers form a grid of points and lines, it would be able to calculate if the scanner moved in any direction on the x/y/z axis. This would be impossible even with multiple ultrasonic sensors on our scanner. The reason being because the lasers used in the expensive 3D scanners form lines which can measure the shape of anything on that line. The ultrasonic sensors can only measure the distance of whatever it's staring at, so we have to keep that in mind when not only programming, but also building the ultrasonic 3D scanner, in order for it to work, it can't act like a normal portable 3D scanner. The hardware and software have to adapt to the capabilities

of what the ultrasonic sensor(s) can do. Unfortunately, since we have yet to determine how the ultrasonic sensor will obtain the values required to make a 3D image, we just have to assume what to do when we have that array of data.

The way we create the 3D image file will be similar to ways AI programmers create programs that filter images. This being because we will be processing datasets for an image ourselves. However, the image we will be processing is in 3D, so we can't use any Convolution Neural Networks to process the data, we will be manually processing the data, which shall be easy in python. Using python's ability to read and write a file, we create an stl file, after writing the initial parts of the file, we can create two facets of the object at the same time. Each point is used multiple times in order to create multiple facets in which the image will be formed. If we want to create a more detailed image, we need to have more points. The less the ultrasonic sensor is moved for every point on an array of the grid, the more detailed the image can be. The issue however is that the more points to be scanned means the more time it takes to scan those points, process those points, and write them to the stl file. This all depends on how little the ultrasonic sensor moves every tick.

Not all points measured should be used however. Depending on how the data is measured, we might have to exclude data that is extremely off when creating the 3D image. If we include the points measured when the ultrasonic sensor is pointed into the air, we would have crazy looking edges to our image, decreasing the total quality of the image. Of course we would have to normalize the data too. Once the data is obtained, we cannot (for example) have an image from an object 10 cm away be 10 cm off the center of the file. The farther away the file is, the more off center it would be. The way we can solve this issue by normalization. Normalization involves taking the points measured in the array and either averaging the data between the numbers 0 and 1, or by making the smallest value 0 by subtracting all numbers by the smallest value. In order to make sure the relationship between the x, y, and z axis stays normal, we will be using the second method. In the instance of the facets being created using multiple unusable points, we would need to be able to ignore those points and not make unusable facets that could break the stl file. In the instance that one point being used is considered invalid, we must only create one facet only. In the case that two, three, or four points are considered invalid, we just ignore making any facet, because there wouldn't be enough points to make a triangle.

6. Project Prototype Construction and Coding

6.1 PCB Vendor and Assembly

For this project, a licensed PCB fabrication company is important to be utilized as precision is key in fabricating a circuit board from a design. Fabricating a printed circuit board as a group is not feasible by our own as embedded components need to be painstakingly put together and accurate to the millimeter. The main factors when selecting a PCB vender to produce the required boards are the cost and expected time of delivery. When it comes to cost, the Bill of Materials or BOM is a main factor that the manufacturer considers when selecting the price for the service. A higher BOM count would require a much higher cost to purchase as each component adds onto the total price. Another cost factor to consider is whether to select multilayer PCBs for the project. A multilayer PCB allows for more components to be placed on the same surface size and does not potentially lead to a short circuit or overheating issues. This is due to as you add more layers you can have more room for additional necessary components. Time is another important factor to consider as the complexity of the board governs the turnaround rate in which manufacturers can produce the product. Typically, boards will be delivered within 14 days on the higher end of complexity. Another element to take into account is the duplicate boards needed to be purchased. Extra boards are needed in case of instances where the board goes bad either from the prototyping stage or from other unknown circumstances. If multiple boards are purchased, then we would not have to wait around for the PCB manufacturer turnaround time once again. While this is the optimal way and a necessity, it will multiply the entire cost of this service depending on the amount of extra boards are needed in spare.

6.1.1 4PCB

4PCB is a PCB fabrication service provided by the company, Advanced Circuits. This PCB service is located in the United States, thus does not require to be shipped overseas leading to a potential issue in receiving the boards on time. Advanced Circuits also provides a student discount for ordering printed boards. This discount waves the minimum quantity requirement for the 2-layer PCB board down to $33 USB per board. While this discount is fantastic, it requires purchasing multiple boards separately and does not give multiple duplicate PCBs like some manufactures provide. Another downside to this service is that the discount requires shipments to be made to the University directly. This can lead to issues properly acquiring the boards in contrast to having them directly shipped to the group. The maximum board size with the student discount is 60 square inches, which is plenty for the portability project. This service from 4PCB also provides a green mask and lead-free solder with a 3-day shipping time.

6.1.2 Osh Park

Osh Park is another PCB manufacturer in the United States that ships worldwide. This company provides boards that are manufactured with a purple solder mask on bare copper with an immersion gold finish. They can deliver $5 per square inch of board within 12 days at order, however most boards ship below 12 days. A difference between Osh Park and 4PCB is that Osh Park provides 3 copies per design free of charge. If more copies are required, then more can be ordered in multiples of three. This is an incredible benefit to our group as multiple extra PCBs will allow us to continue the prototyping stage with extra boards. As well as the added benefit of not having to purchase each extra board separately, which will increase our total cost substantially. If at any point there are PCB issues, Osh Park provides a “Super Swift” service to rush 2-layer boards with a turn rate of 5 days rather than 12 days. However, this faster service charges $10 per square inch of board instead of the $5 per square inch in the normal turn rate. Osh Park orders are fulfilled by importing KiCAD, Eagle, or Gerber file types, therefore may limit which programs are used in the design aspect of this project.

6.1.3 ExpressPCB

ExpressPCB is an alternative option for selecting a PCB vendor. Like the other companies, ExpressPCB is located in the United States. Unlike the previous manufactures, ExpressPCB has its own PCB layout and design software. This software is free to use and has two versions, ExpressPCB Plus and ExpressPCB Classic. The difference between the plus and classic version is that the plus version gives more features such as additional copper layers. While ExpressPCB offers a free CAD software for designing boards, it would require redesigning the project for the included software, which would lead to increased time consumption. At $65, we can purchase a 2-layer PCB on a 1-day lead time. This is the fastest turn time out of all the vendors compared and can be valuable, if it's necessary to order more boards in a short time frame. Like Osh Park, ExpressPCB comes with 3 copies of a board and can lead to backup boards shall something happened to the board.

6.1.4 JLCPCB

JLCPCB is located in Shenzhen, China with a PCB prototype turnaround time of 24 hours. What makes JLCPCB different from other PCB vendors is that they offer full PCB assembly with over 230,000 parts available. When selecting the PCB assembly, JLCPCB offers a free DFM file check. The downside of having the PCB assembled is that they only offer this service for 1 oz copper layers, but not for 2 oz copper layers. If the PCB trace lines need to transmit higher current values, then the designer is forced to utilize thicker traces instead of opting into a 2 oz copper layer. The available layers are 1-6 layers with 5 pieces per order, which provides two more pieces than the other vendors, in case it being needed.

	4PCB	Osh Park	ExpressPCB	JLCPCB
Pricing	$33 per board	$5 per square inch	$65	$8 w/o assembly
Turn Time	3 days	Within 12 days	1 day	1 day
Number of Boards	1	3	3	5
Surface Finish	Lead-Free HAL	ENIG (Gold)	Tin-Lead	ENIG-RoHS
Max Board Size	60 square inches	352 square inches	≤ 10 square inches	310 square inches
Board Thickness	1.6mm	1.6mm	1.5mm	1.6mm

Table 14: PCB Manufacturer Comparison

Upon analysis, we have decided that Osh Park will provide the best service required for this project. While they are the longest in terms of production, they provide three copies for each order and are still a reasonable wait-time. The surface finish on the Osh Park boards is far superior to the other offerings since it provides ENIG as standard. The benefit of ENIG is that soldering is much easier compared to tin-lead or HAL provided by the other vendors. Osh Park is also eagle friendly in that they allow submitting .brd files without the need to generate gerber files. When submitting a design, the site allows you to preview different layers and view error warnings before submitting for fabrication, which can be very helpful to not spend extra money on fixing a design issue.

In Senior Design 2, we have changed the PCB vendor selection to JLCPCB. This is due to the need for the STM32F103 microcontroller. STM32 microcontrollers are hard to find on the market due to a shortage in supply, however JLCPCB has a large stock of parts and roughly had under a thousand STM32F103 MCUs in stock to purchase, if assembled through the vendor. Given the low-cost nature of the project and study, funds were available to purchase the service and still stay within the budget of the project. Not only was JLCPCB necessary for the completion of the hardware, but they provided 5 copies of the board (3 blank boards and 2 assembled). The extra assembled PCB was pivotal in successfully completing the project as the first board had a faulty TLV62569 buck convertor that failed to regulate the 5V to 3.3V for the MCU.

6.1.1 Prototype Expectations

In terms of evaluating the success of our prototype we will take into consideration a variety of factors in order to determine if it has met our standard. This is a very important aspect of the development and deployment of our product as we need to ensure that the product is not only working, but it has a high quality and is safe to use. Failure in our determination of the success of our prototype would mean that we have to revise the design and implementation in order to make the product feasible and practical.

Taking into considerations any sort of liabilities that our product can have on us, we would like to ensure that it meets not only the standards set by international engineering standards, but also the standards by any particular group or person interested in our product. We do not want to be on the receiving end of a lawsuit because of failures we overlooked during testing and design. We will place an emphasis on product safety, as we do not want to pose a risk to anyone, or anything. With that said we will be putting the product through a variety of tests in an environment of volatile conditions to ensure that the product is able to fare up and not cause any problems.

Additionally, we will have to expedite the process given the tight scheduling constraint that exists with the senior design project deadline. So ultimately this will be a tough task to accomplish, but we would like to at least mention that we would try to ensure that our product is completely safe and that it works by the time we are in delivering phase.

6.1.2 Potential Hardware Issues

Given the tight schedule that exists with this project, it is incredibly arduous to even be able to assemble one design, so part testing will be imperative. It is true that we will need to keep an eye on having a variety of back up plans as having 1 part that is not functioning might draw us back by weeks considering the pandemic situation we are in where a lot of items are on back order. So, testing and implementation will already be a tough task to hurdle therefore we will need to ensure we are the most prepared we can possibly be when handling with parts immediately when we receive them. There are some ways to address this as well such as that no feature in our product is created by only 1 part, therefore we always have our core features being accomplished by a system of parts, that won't be hindered by one not being the way we wanted it to.

Given that our product is extremely sensitive to accuracy and calibration, this is something we need to keep in mind as our sort of design would be most prone to being held up by limitations assessed by one of our parts in our system. Additionally, we might already recognize alternative parts we can use, and already order them so that way in case our first choice is not as we expected, we can

quickly use another part. Although this is pretty expensive, this is definitely something that we would need to do to ensure that our product can be delivered promptly, and not be held up by backorders or shipping/handling time.

Given the electrical nature of the parts that we are utilizing, making sure that the parts we utilize are handled correctly and are not damaged either by soldering, corrosion or simply damaging in general when handling them will be important. Damaging a component and having to buy a new one is definitely something we need to keep in mind. Therefore, we will have to review all the correct ways of handling electrical components, and make sure not only we are following the right methods, but that all of us are using the correct safety equipment especially when handling things that are being charged or have a current/voltage. Besides improper implementation, one can just simply order a part that is simply nonoperational. Ultimately the way to rectify this is to buy more parts, so that we would be able to have a sort of insurance. Even the cost of acquiring more than necessary parts still outweighs the possibility of having a late delivery for senior design II.

Additionally, we will need to ensure that when we are designing the board, that we are being adamantly careful in following standards and having the most clear of communications with manufacturers, regarding the board design. Utilizing the correct programs will be imperative. Taking note that we will be using Eagle AutoDesk as one of the software's to design and create the PCB board, it will be important to follow the correct standards to minimize the time manufacturers might have trouble understanding what we are intending to do since it will inevitably come to that due to the limits in our experiences. Since the board has to have a perfect trace, plane and pad are adequately made so that the components will be able to be correctly integrated, given the manufacturers will listen to what we will conjure, it is important to know exactly how to make it.

Despite having a proper design, a product might be prone to failure therefore it is important to note how exactly we will try to mitigate any errors which we will expand on. There is absolutely no room for error when handling the electrical components in the product, not only for it to be what we actually want, but to ensure that when everything gets mounted that nothing gets damaged to the point where something will not work. This will cost our team a lot of time, as well as money and therefore such tangible shortcomings need to be avoided at all costs.

Experience is something that as a group we are lacking, but that will not deter us from learning and following the proper guidelines to ensure a safe and proper implementation of the product and design.

A common error is when there are incorrect rankings of nested planes, such as having both a ground and VCC which are nested within one another, there needs to be extremely careful and meticulous care addressing their rank and the consolidation thereof. Rank being that of plane priority when the board is being manufactured. Therefore, it will be important to have the correct board layout sent

to the manufacturers so that the board that they produce is one that we would actually want and need for the success of our product. If a nested plane has a smaller rank than a plane that is surrounding it or on top, it could be overridden by the other plane and therefore have the whole product be ruined.

Additionally, understanding the nuances behind the software such as in Eagle, there are some shortcuts where the program will identify the layout and make sure to not short the circuit in an unintentional manner, however it will be important to recognize these shortcuts and understand what exactly is going on just in case there is something we did not intend to be represented in the layout of the board.

Additionally, given Eagles extensive library it can be easy to select the incorrect parts which would be catastrophic for the project as we want to ensure that no part is incorrectly labeled or made in the design. The plane issue which was mentioned in the previous section is actually not recognized in Eagle therefore close attention needs to be had. The chip is so sensitive that even mislabeling a ground pin such as the VCC pin, can easily destroy the chip and make it useless. This would be absolutely catastrophic for the project since a lot of time is focused solely on having to not only design the board, but wait for the board to be manufactured and then shipped to us. Therefore, if any part were to be destroyed by some misstep it will lead to a cascade of trouble for our group given the time sensitivity of senior design deliverables.

Emphasizing on how to correctly solder will be something that we can not undermine and overlook. As engineers, understanding how to solder properly is imperative as it is an application of the knowledge we should already know. Shorting pins by accident is definitely something that can happen easily, therefore we will have to review the correct guidelines and form to properly address soldering of the parts. Besides implementation, testing is a part of our project where failures can arise and can be difficult to detect. For example, the use of static electricity is one which is a major issue especially when dealing with active semiconductor devices. Even taking note of the temperature is something that we will need to pay attention to since humidity can have an influence on the functionality of our parts.

6.1.3 Potential Software Issues

Given that any software can be prone to errors based on not including error handling, misuse of variables, memory limitations etc. It is the job of our group to try our best to ensure that our coding is up to par with standards, which in our case will be based on NASA’s guidelines, and also actively monitor the command prompt to see any errors that might arise and cause problems for our product.

One of the major factors that we need to focus on is that of memory. Since our product will be taking in a lot of data, and the compilation thereof that data needs to be done, ensuring that our program is able to effectively handle the memory that is needed, and ensure that when memory needs to be cleared it can be done so

without having catastrophic causes on the product. This is commonly referred to as buffer overflow as well where the program just cannot keep expanding, as there is a limitation in additional space for any of the buffer safeguards, whether it be a variable such as a counter variable hitting its maximum limit which leads to overflow into another address.

Additionally conditions have to be planned for when there are concurrent accesses to the device, such as when a read/write cycle is occurring, if a sensor is being utilized at the same time and feeding info of another sensor to the same address, we need to ensure that memory leaks do not happen. Or if one is doing a read and one is doing a write, if the write is depending on the read first, then it is important to ensure that our program has the correct handling cases for instances such as those so that the program can actually handle the concurrent operations and accessibility of the chip is as open and smooth as possible so that the device is relatively fast.

6.1.4 Prototype Constraints

One of the major drawbacks in our prototype is time. As we will need to test variety of parts and ensure that the device not only works, but that it is accurate enough. Therefore, it will require a lot of time and that is something that unfortunately our group does not have much of due to the nature of the time sensitivity surrounding the senior design project.

Besides time, money is a constraint given that prototypes will involve testing a variety of parts which would not necessarily make the final build, so having to test a variety of parts which all costs money is inherent with our design and is something that will need to happen.

Additionally other constraints can be simply the accuracy of the scanner as we can obviously attempt to get more accurate scans, however we would need to buy more expensive sensors which have more accuracy. Obviously, this comes with a higher price, so we will need to find a middle ground to not only create the prototype and test, but also the final build because at the end of the day, we still want a fairly accurate scanner. Having a cooperative and driven team is also another constraint given the limited time that we might have in order to meet the deadline for this project. Ultimately, we need to address a variety of constraints ranging from time, money and willingness to work as part of the project in order to ensure that we can attain some level of success when determining the feasibility of a portable 3d scanner.

6.2 Final Coding plan

The final coding plan so far is not completely decided. This is mainly due to the fact that the way we scan the 3D object hasn't been completely decided yet. However, there is a clear way we will be creating the 3D image after the data has been collected.

The data will most likely be collected in a grid fashion. The ultrasonic sensor will be collecting multiple points of an object, so there needs to be an organized placement of those points measured. Therefore, an array would be best suited for containing these points. The specifics on how this array is filled is still in the works, but assuming we already have these points, we can construct a 3D image using this array. Python is well suited for data processing, and can handle what we need for the creation of a 3D image file.

The .stl file for the 3D images can be created in python by hardcoding the values into the file. The values of x and y for every point for the 3D image can be already predetermined before the 3D image file is being written, the only thing we need from the ultrasonic sensor is what is on the z axis. However, how these points will be made will be the complicated part. The .stl file uses triangles called facets. These facets have three points, these points being on an x/y/z plane. The program we write would have to work with one square at a time (one square being two triangles). The x and y will already be predetermined, but the way we will calculate the z is through normalizing the ultrasonic sensors data. All values from the ultrasonic sensor will be subtracted by the lowest value (aka the closest point). This will make it so that the object is not far off on the z axis when trying to view the 3D file. After the data has been normalized, we then create two facets at a time by measuring four points at the same time. If the four points (i,j), (i+1,j), (i,j+1), and (i+1,j+1) are used, we would use (i,j), (i+1,j), and (i,j+1) for one side of the square, and then (i+1,j), (i,j+1), and (i+1,j+1) for the second side of the square.

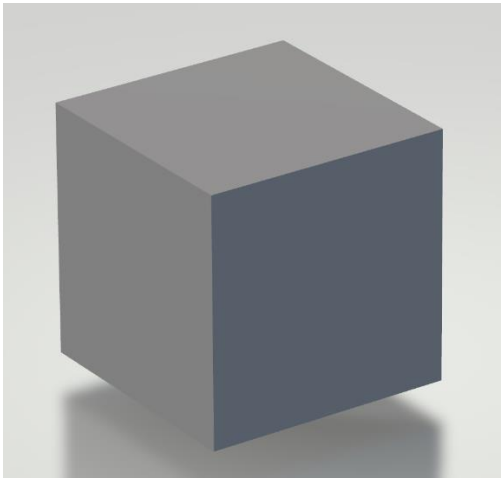

Figure 24: Cube from an STL file

20mm_cube.stl

```
solid model
facet normal 0.0 0.0 -1.0
outer loop
vertex 20.0 0.0 0.0
vertex 0.0 -20.0 0.0
vertex 0.0 0.0 0.0
endloop
endfacet
facet normal 0.0 0.0 -1.0
outer loop
vertex 0.0 -20.0 0.0
vertex 20.0 0.0 0.0
vertex 20.0 -20.0 0.0
endloop
endfacet
facet normal -0.0 -1.0 -0.0
outer loop
vertex 20.0 -20.0 20.0
vertex 0.0 -20.0 0.0
vertex 20.0 -20.0 0.0
endloop
endfacet
facet normal -0.0 -1.0 -0.0
outer loop
vertex 0.0 -20.0 0.0
vertex 20.0 -20.0 20.0
vertex 0.0 -20.0 20.0
endloop
endfacet
facet normal 1.0 0.0 0.0
outer loop
vertex 20.0 0.0 0.0
vertex 20.0 -20.0 20.0
vertex 20.0 -20.0 0.0
endloop
endfacet
```

Figure 25: STL file Example

After creating one square (by writing it into the file), we increment i, and after one row of i's, we increment j and reset the i counter. Going through the entire grid, we form a 3D image in the shape of what we scanned. There is the problem, however, of what about when the scanner scans a point extremely far away, or hits air. There would have to be a limit to what points are considered real or false. There would have to be cases when certain points from the four points selected are invalid. There are 13 individual cases we would have to account for, including if one, two, three, or all four points being used are invalid. If one point is invalid, the only thing the program must do is create one facet. The case for two, three, or four missing points is even simpler, we just don't create any facet. The reason why is because those points have probably already been used to create other facets.

When planning on how to program the Jetson Nano to do these things, we can follow along with how the JetBot gets programmed. While attaching a WiFi USB dongle to the Jetson Nano, we are able to program the Nano from our computers

without using wires, which we will have to do. Jupyter Notebook runs from a local host to where the files are located, so including WiFi would enable us to program the Nano in the first place.

The programming language will be python, this might be a setback due to the limited number of programmers in the group who have already used python, the lack of experience will cause the programmers to slow down production of working programs. However, it is definitely possible to complete the program before the final deadline. The programs we will be writing will not be so complex that it will confuse the entire team. First we will be programming the ultrasonic sensor(s) to move (how is still yet to be determined). Then we constantly fill an array with points found from the ultrasonic sensor(s). Afterwards we convert the data into facets for the stl file that we write using python's ability to write to files.

Now how will the user get the finalized file? The final stl file will be written to an output folder on the Jetson Nano. This will be on the microSD card that Jetson Nano's programming will be run on, so if the user accidentally deletes a file from the Jetson Nano's programming that makes it run, they would break it. Fortunately, as previously stated, the Jetson Nano will include a WiFi dongle, so the User will be able to transfer the file from the Nano to their PC. How will the Nano run on the user's command? A button would have to be implemented, so that the Nano isn't something that runs on startup only.

The next part in our feasibility study is our stretch goal, adding color to the 3D image. We can do this one of two ways. The first would be to have the camera record the color of every point of the object that the ultrasonic sensor records, this would make it easier for the programmers, because every point would have an allotted color, but harder for the hardware people, because the camera would have to accurately point to where the ultrasonic sensor points. The second would be to take a picture when the camera is at the center of the object, this would be harder for the programmers because then from the center of the ultrasonic sensors FOV, they would have to program the Nano to take color from certain points of the graph, and it would be easy for the hardware designers because the picture wouldn't have to be centered with the ultrasonic sensor, it would just have to be able to capture what would be in the ultrasonic sensors FOV.

The next part would be putting the color onto the 3D image. At this point we should know the colors found on each point of the grid used for the uncolored 3D image. It would be impossible for the point to be given color, because the points will be used to create facets. The facets would be the things to be colored in this situation. What if one point were to be red and another point to be orange, and another to be yellow, what color would the facet be. There is a concept that is well used in AI programming, known as pooling (specifically what we will be using is average pooling). Average pooling involves taking the three colors for the three points of the facet, taking the values of their colors (every color has a value that can be shown in the sense of the color spectrum), and averaging the values. The average of red, orange, and yellow would be orange. This is commonly seen in AI

languages so decrease the resolution of images in a sense pixelating them. This will somewhat be the case for our 3D image. Just like with the detail of the object, the more points there are, the more accurate the color will be. Accuracy of color and detail of an object solely relies on the number of points recorded during the scanning process.

6.2.1 Final Coding Plan – Jetson Nano

Our original plan was to turn the Jetson Nano into a JetBot, the reason why was because JetBot had the ability to run Jupyter Lab, and with that we would be able to access the Jetson Nano using WiFi. However, our plans fell short when trying to actually implement it. We first used a microSD card to flash the Jetson Nano operating system, Jetpack. Jetpack is the Jetson Nano's own Linux distribution, specifically Ubuntu 18.04. After placing the microSD card into the Jetson Nano and starting it up, we began the process of starting up JetBot.

First, we created a new swap file, so that the JetBot would be able to use more memory and process data faster. Then we moved onto installing the JetBot software. Because JetPack was its own Linux distribution, and we had a WiFi adapter plugged into it, we could go to the GitHub page and download the software. The next step was to run the JetBots Docker Container. This was the part where we had the most trouble. First off was to configure the Jetson Nano's power mode, along with other parameters. The next step was to configure some of the Jetson Nano's environmental variables, this step is where we had the most trouble. When running the configure script file, we learned that the latest version of JetBot didn't support the latest version of JetPack. After re-flashing the OS and going through the steps again, we discovered that even in a supported version of JetPack caused trouble. The issue we learned in the end was that the JetPack operating system did not allow us to install anything at all. This led us to thinking of alternate solutions.

We realized that in the end we only wanted JetBot for the programming IDE, Jupyter Lab. So once we realized that we set off to installing Jupyter Lab itself without the troubles caused by trying to run JetBot. While trying to install anything, all we had were errors. The JetPack operating system would not allow any updates for programs we needed, this caused errors while trying to install the latest software due to incompatibility. We found the solution to this, which was very strange, but still worked nonetheless. We found a solution that required us to just completely clear the folder that has the already installed programs, and then completely reinstall the programs, rather than updating them. This solution worked and allowed us to install everything we needed. It was then time to set up Jupyter Lab.

The first step was to install an SDK named DeepStream5, after trying to install it, we learned that the latest version of DeepStream was not supported on the version of JetPack we were running, so we had to roll back the version of JetPack once again, and then re-doing all the fixes we did previously.

Now that we could finally install things, we started setting up DeepStream. DeepStream is used for visuals and cameras. Our original goal for this project was to have a camera scan the area as well along with the ultrasonic sensor, creating a colored 3D image. However, we determined that starting off with just working on the 3D aspect of the image was more important than color in an image. We kept DeepStream, and installed it anyways, for the reason that the camera color scanning became a stretch goal of ours.

The next step was installing JetCam, another piece of software for the Jetson Nano. Again this part of the setup was only done so that on the possibility that we reach the point where we are able to work on our stretch goals. This part was easy, install the github repo on the Nano and run the python setup script provided. From there we know that in a python IDE to access the camera we would have to import CSICamera from JetCam, and then we can specify the camera width, height, and FPS. We use CSICamera rather than USBCamera because the camera we selected was a 2-lane CSI camera.

Our next step is to download any dependencies JupyterLab has for Ubuntu 18.04. The dependencies were Node JS, libffi-dev, libssl1.0-dev, numpy, scipy, matplotlib, and python3-pip. All of these were required, but one out of all of them was the most important, python3-pip. Python3-pip was important because the version pf python that JetPack uses and has automatically was Python 2, and we decided it was best to install the newer version of python, reason being we didn’t want to run into problems where even more things were outdated. And with all that it was finally time to configure and set up JupyterLab.

First, we had to add our bun directory to our PATH environment variables, which allows us to not only install JupyterLab, but also shows JupyterLab where to run, store code, etc. After that, we setup JupyterLab by generating a config file. With that we can then create a password for JupyterLab if we so wish. Now in order to access JupyterLab, we have to install more things. In order to access JupyterLab, we have to connect to it from a hosted server that it runs. In order to run a server that you can access through the web, we need to install Extension Manager, iPywidgets, and iPywebrtc. After this we can then run JupyterLab and go to our Linux web browser and go to localhost, where then we can see our JupyterLab and run code. After this we set up JupyterLab to be able to connect to it from our computers, so that we don’t have to always plug in the Jetson Nano to a monitor, keyboard, and mouse to use. First we find the IP address of the Jetson Nano, then we use the terminal window on our computers and run an ssh command to connect

to our Jetson Nano remotely. Then from there we run JupyterLab, and connect to localhost through a web browser.

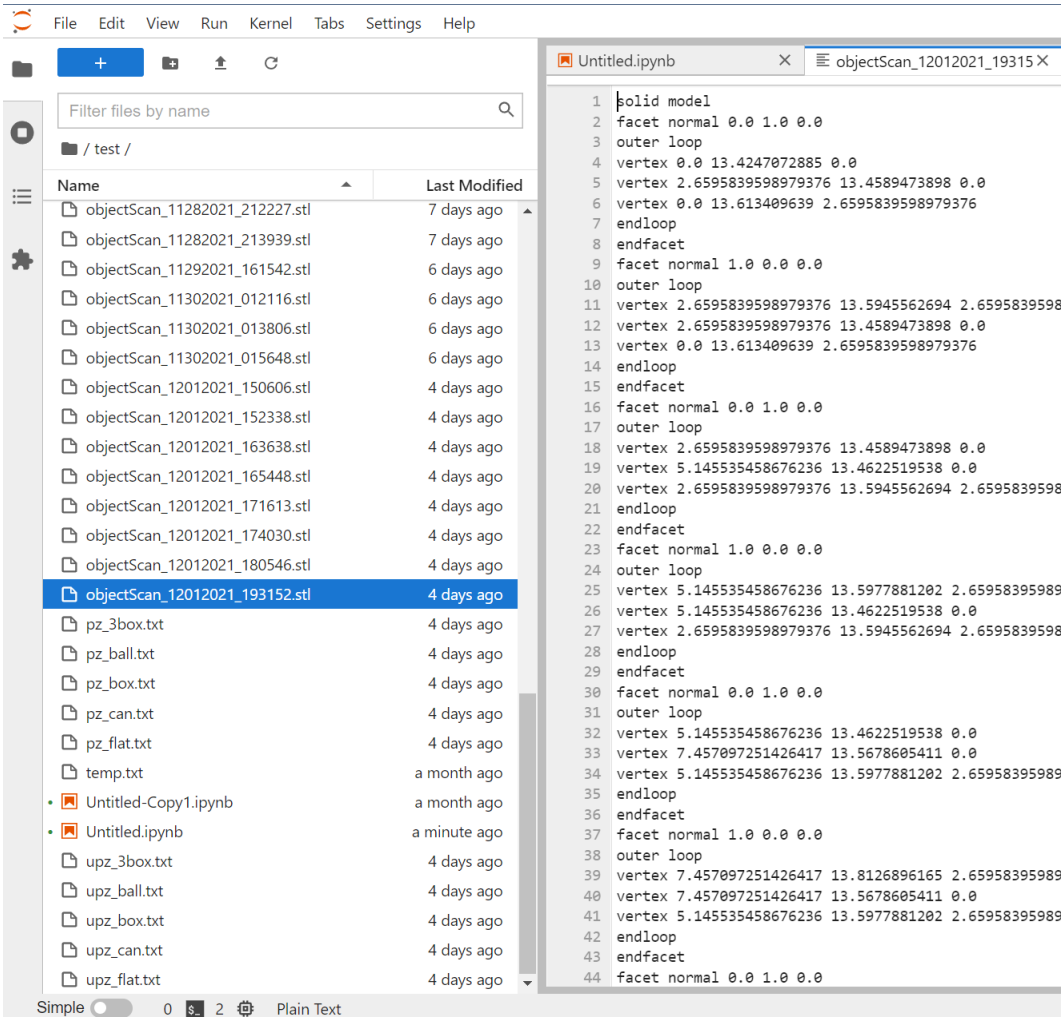

Figure 26: JupyterLab IDE

6.2.2 Programming Specifics

When developing our program, we decided to split up the work into multiple different phases. We first had the Data Collection phase, where the microcontroller and the Jetson Nano work together to collect data and store it for later use. The second phase is the Data Processing phase, this phase is to turn the Utlrasonic Sensor data into a set of data we can actually use (more on that later). Then, last but not least, is the Image Creation phase, where we finally write the 3D image file using the data we collected and processed.

6.2.3 Data Collection Phase

There are essentially two parts to the data collection phase. The first part is what the microcontroller does, and the second is what the Jetson Nano does. First is what the microcontroller does. The microcontroller was programmed through Arduino IDE, we were able to do this by first flashing firmware into the STM32F1 (our microcontroller) that allowed programming through our USB-C port. Programming the microcontroller, we know that the two external libraries we need to use are Wire.h and Servo.h. Wire.h is used for I2C, so that we can communicate between the microcontroller and the Nano. Servo.h is a special library for the servos we use in order to move them. In the setup of the microcontroller, we setup the wire connection to the Jetson Nano, along with setting up the request and receive events. Then we initialize the two servos to certain angles, we had to initially find the center x and y of the servos, the reason being is when the servos cover their 180 degree angles, the center of the pan/tilt kit wouldn't be 90 degrees by 90 degrees. The center point of the scanner we discovered was 92 degrees for the x axis and 86 degrees in the y axis.

The Ultrasonic Sensor's scanning process is part of a function we made called findDistance, where in which the ping pin is set to high for 5 microseconds, where the ultrasonic sensor will send out the sound waves it will detect, and then the distance pin is set to high, to detect how long it takes for the sound waves to travel and come back to the sensor. This function is only ever called when the microcontroller gets a request event. Request events are called whenever the Jetson Nano wants to obtain data from the ultrasonic sensor. When a request event is called, the microcontroller will call the function findDistance 10 times.

The data is then averaged out, and returned to the Jetson Nano. With the microcontroller, we also have a receive event. The receive event is meant for moving the servos in the angle they are needed. The Nano sends the angles based off of what point on the 2D array of data it needs. Since we're scanning with an FOV of 31x31, the center point of the array is [15][15]. Therefore, knowing the center of the scanner, and that we move 15 degrees up down left or right, we can just easily send how far up down left or right to pan or tilt the ultrasonic sensor. We have the Jetson Nano set up so that if we ever wished to scan an area multiple times over, we can collect the data in the array multiple times and find the average of the data we collect.

6.2.4 Data Processing

The next phase in the scanning process is the Data Processing phase. When scanning the values using the ultrasonic sensor, we scan at an angle. The scanner has a center point and will go up, down, left, or right at a max angle of 15 degrees.

First we will determine the x and y multipliers of the image. What we use is trigonometry (specifically sin(abs(a-15)), a being the x or y position on the 2D array) to find out how large the distances should be between each point on an x y plane. The next part is to process the data. The data needs to be run through a trigonometric function (cos(abs(a-15))) twice, the reason being is that the ultrasonic sensor is angled at two different directions.

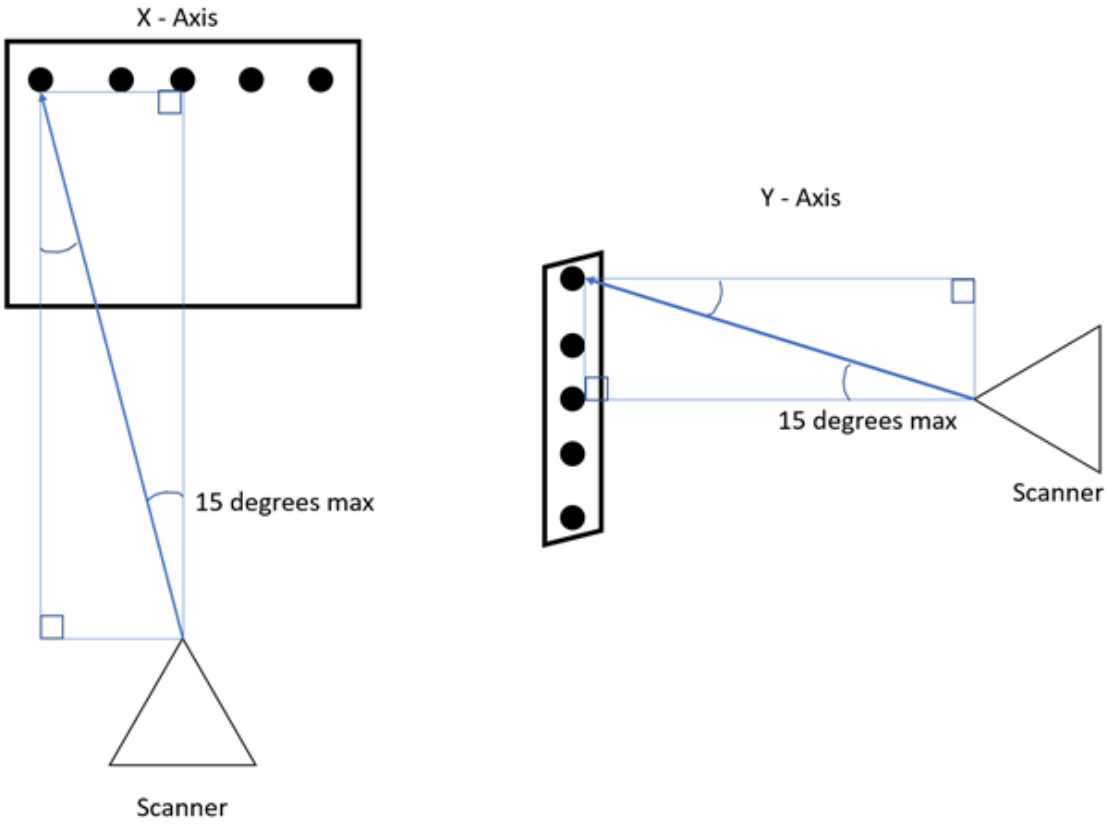

Figure 27: Trigonometric view of a flat surface

As you can see in the figure above, the ultrasonic sensor can be angled to 15 degrees in any direction. This causes the distance measured to be angled twice. Using basic trigonometric assumptions, we can deduce that the measurement scanned is the hypotenuse of a triangle. With that information and the angle in which we scanned it in, we can figure out the “adjacent” of our imaginary triangles, forming the correct distance between the object and the scanner.

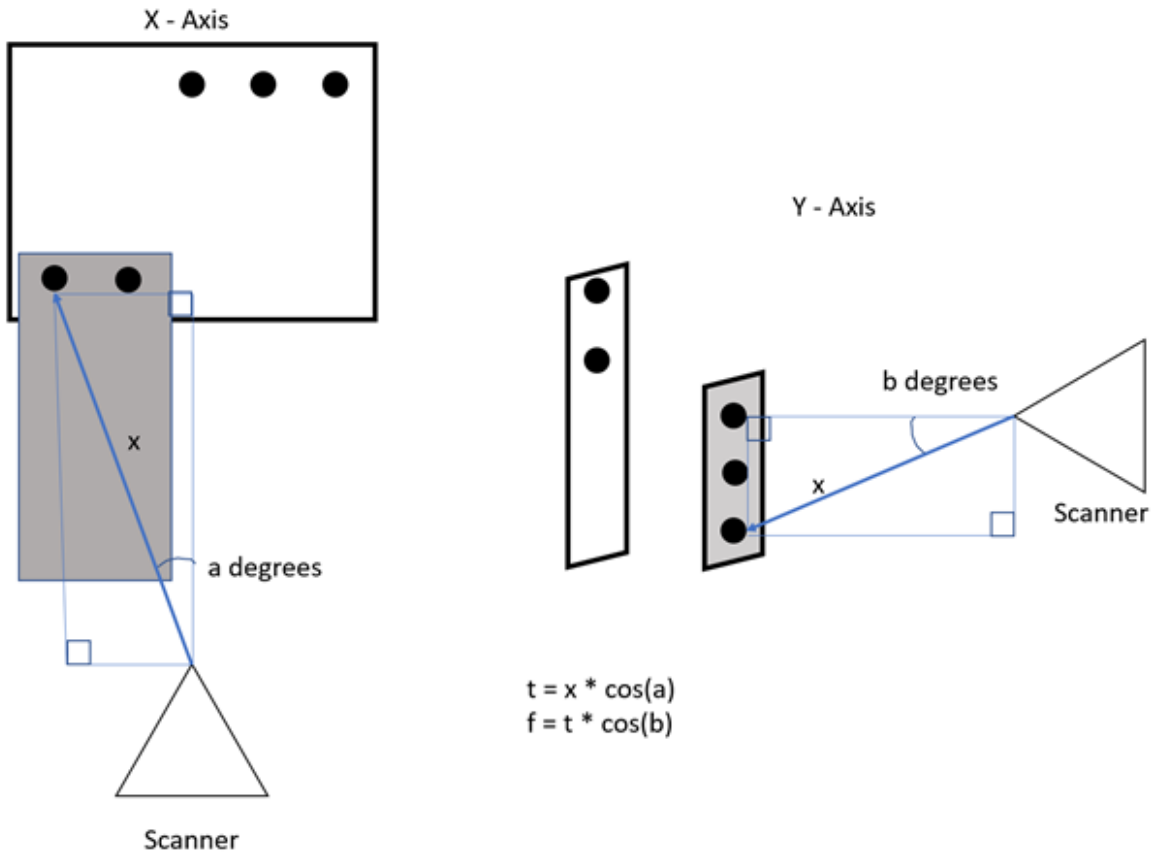

Figure 28: Trigonometric view of an object

6.2.5 Image Creation

The final and most important phase is the Image Creation phase. This phase is the most important phase of the scanning process. We first create an stl file. The name of the file we use is objectScan plus whatever the date and time are. Then, for the image processing, we base our technique on AI algorithms such as Convolution and Pooling, specifically how a convolution or pooling filters data. Convolution is used for image processing, it's mainly used for detecting patterns in images to tell if they are similar. For example, a Convolution Neural Network (or CNN), when trained, can look at images of birds and locate where the beak is. The way it happens is when looking at an image, what the program actually is looking through a 2D array of data, each part of the array is a pixel of color. With a filter, we look at a small portion of this data, and then process it to check if, for example, there's a beak in the image.

The methodology we use from this is the way the filter searches through the array of data. Our 2D array is 31x31, and the filter we use is 2x2. With a stride of 1, the 2x2 filter will travel through the 31x31 array incrementing by one each time. When we have the look at what's in the 2x2 filter, we take this data and turn it into the points of the 3D image. The four points on the array making a square are turned

into two triangles. The X and the Y of every point has already been predetermined as stated in the Data Processing phase, and during the Data Processing phase, we also process the ultrasonic sensor data into what will be our Z for every point.

```
solid model
facet normal 0.0 1.0 0.0
outer loop
vertex 0.0 8.023909236273086 0.0
vertex 6.3235837518209355 8.060209829871884 0.0
vertex 0.0 8.435103310331042 6.3235837518209355
endloop
endfacet
facet normal 1.0 0.0 0.0
outer loop
vertex 6.3235837518209355 8.473264167865171 6.3235837518209355
vertex 6.3235837518209355 8.060209829871884 0.0
vertex 0.0 8.435103310331042 6.3235837518209355
endloop
endfacet
facet normal 0.0 1.0 0.0
outer loop
vertex 6.3235837518209355 8.060209829871884 0.0
vertex 12.234328718899402 8.094055205475497 0.0
vertex 6.3235837518209355 8.473264167865171 6.3235837518209355
endloop
endfacet
facet normal 1.0 0.0 0.0
outer loop
vertex 12.234328718899402 8.225215854149702 6.3235837518209355
vertex 12.234328718899402 8.094055205475497 0.0
vertex 6.3235837518209355 8.473264167865171 6.3235837518209355
endloop
endfacet
facet normal 0.0 1.0 0.0
outer loop
vertex 12.234328718899402 8.094055205475497 0.0
vertex 17.730434431059756 8.030953250507503 0.0
vertex 12.234328718899402 8.225215854149702 6.3235837518209355
endloop
endfacet
```

Figure 29: example of an STL file we make

6.3 Integrated schematics

In the following page is a schematic on how the initial PCB design would look like. The design includes a power distribution system to step down the voltage to 5V. The design will feature solder pads so that a barrel jack cable can be connected to the Jetson Nano. Each ultrasonic sensor has been replaced by pin headers where they can be connected to the actual sensors via wire. This eliminates the need for soldering down these sensors and limiting their positions as the cone of propagation is an issue with multiple sensors. The sensors will be connected to the Jetson Nano 5V logic where it is directly connected to the output of the TI TPS56637 buck converter. The low BOM count of the Texas Instruments buck converter at 10 BOM, will allow us to design a smaller PCB board and limit the total

footprint for all the necessary components. By using resistors as a voltage bridge divider for the ECHO pins of the ultrasonic sensors, we can limit the need of using any additional buck convertors or step-down convertors that can increase the footprint and cost of the PCB when fabricating. The voltage bridge should decrease the 5V logic of the ECHO pin down to a safe 2.5V needed for the microcontroller to not be damaged in any shape or form.

However, shown in figure 31, in Senior Design we have changed the overall schematic and design of the system to match the desired needs.

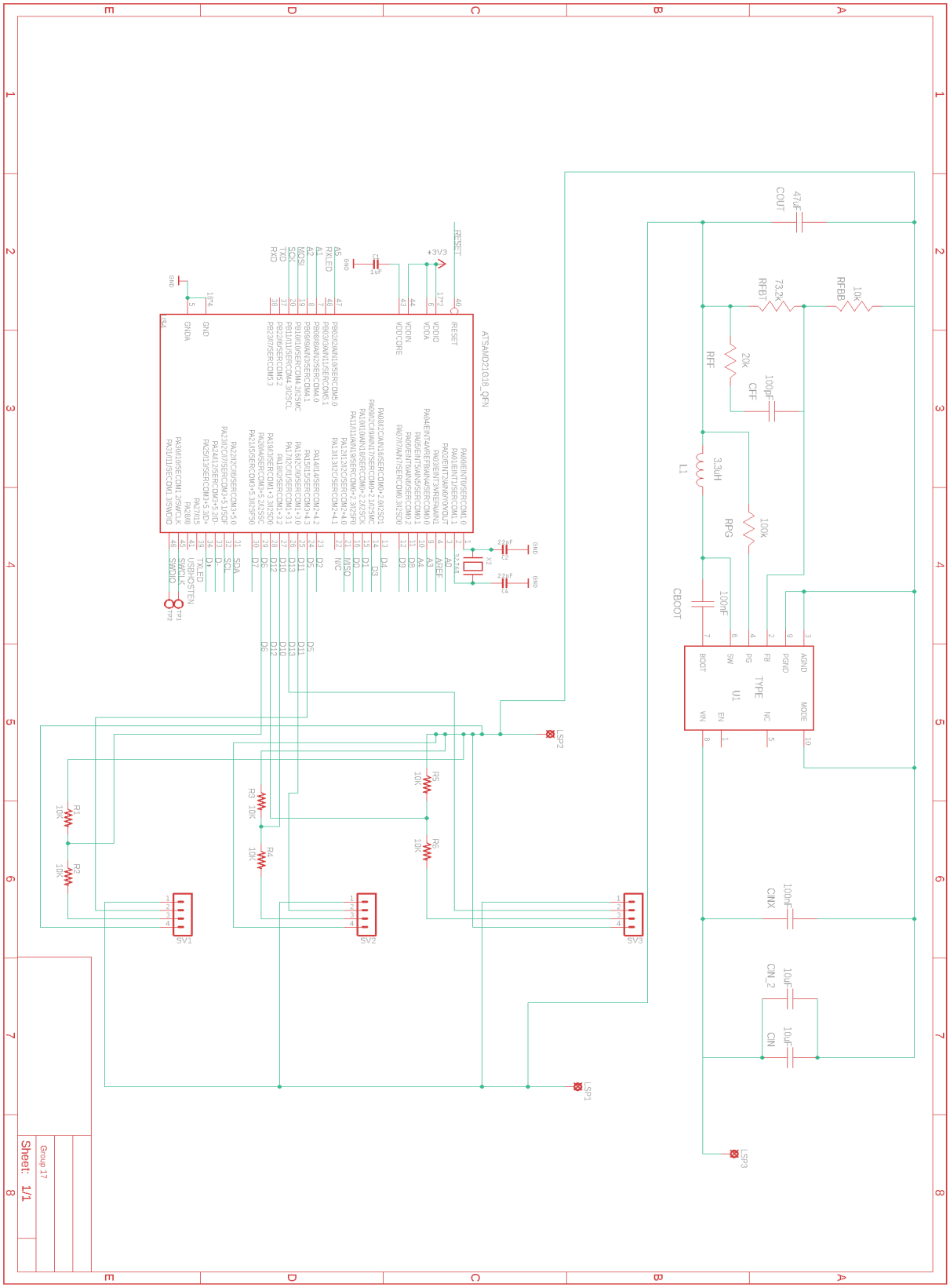

Figure 30: Original PCB Schematic

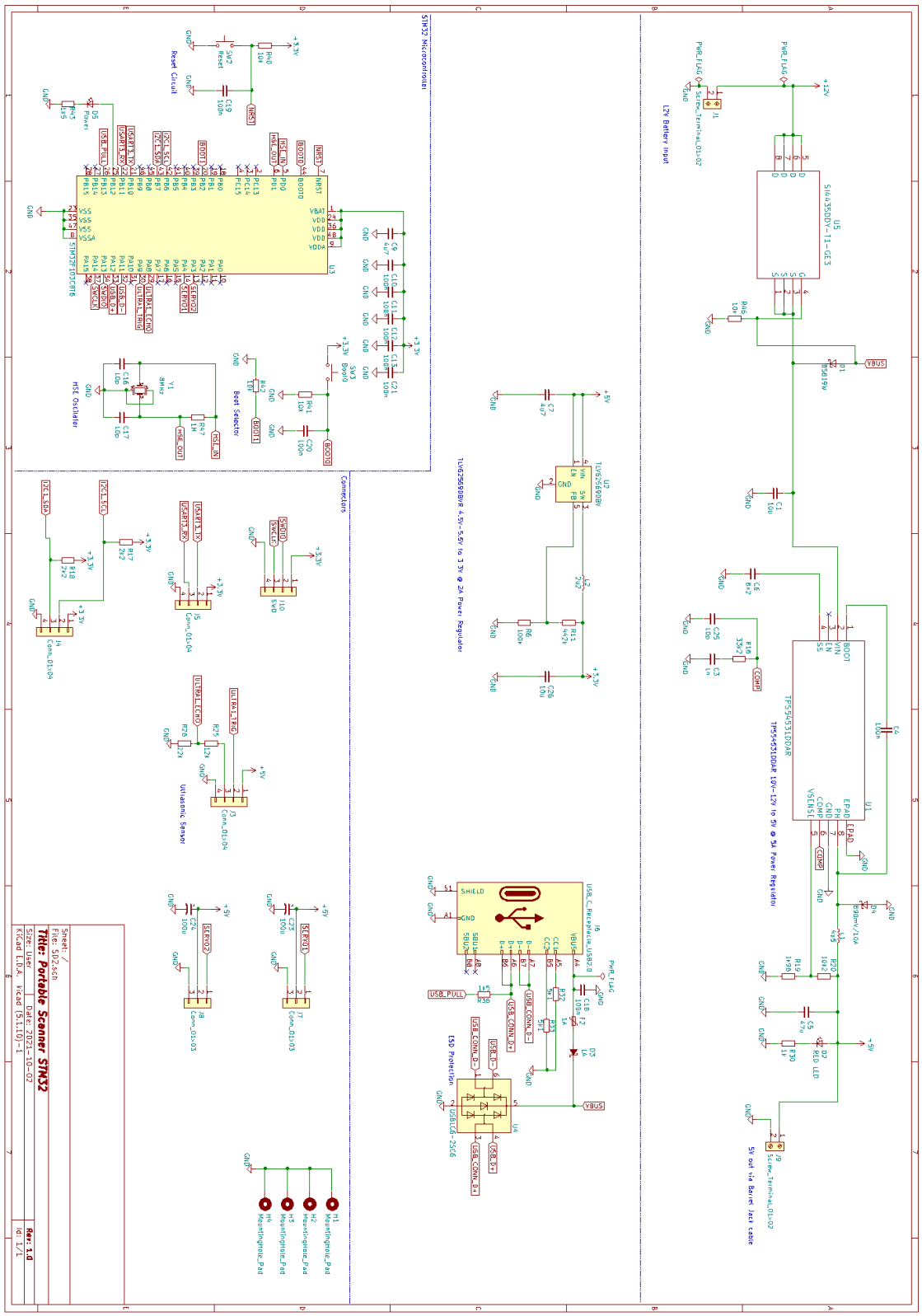

Figure 31: Updated PCB Schematic

7. Project Prototype Testing Plan

7.1 Hardware Test Environment

We will have two different environments when testing the hardware. The first will be in the ENG1 room 456 lab. This is where many others will be testing their hardware and software. We will be conducting experiments and constructing the device here. In the lab there will be tools and an area for the group to properly work together. There will also be other groups working there too, so any advice needed that our group needs at the time could be answered or improved upon by peers in the same field. The next environment we will be working in will be at our own homes. Working from our own homes will benefit only the one who has the hardware. This would mean the one who has the hardware won't have to wait for the other members of the group in order to test the hardware. This will make use of the Development Kits provided by UCF. The downsides are that the other members of the group would not be able to participate in any software or hardware testing.

7.2 Hardware Specific Testing

The first thing we would have to test is the Jetson Nano. The Jetson Nano is the most important part of the project, as it will be receiving all the data to create a 3D image. The Jetson Nano will have button headers, camera connectors, power selector jumpers, an HDMI port, USB ports, a power jack, an ethernet jack, a microUSB port, a fan header, a SODIMM connector, an expansion header, an M.2 slot, and a POE header. All pieces must be properly implemented onto the board. We must check every port/connector to make sure they work. The Feather M0, for it's tiny size, has up to 52 programmable I/O pins. Not all of the pins will be used for this project, so we should only test the number of pins that we need. The ultrasonic sensors and the camera both have to be tested after the Jetson Nano and Feather M0 have been tested, as we will be using both to use the camera/ultrasonic sensor.

For the PCB, the first thing we would have to test is the battery input, if it's able to obtain power. Then the next most important test would be to make sure the 5V to 3.3V buck converter was functional, this provides 3.3V to the microcontroller, 5V would damage the STM32F1. The next step would be to burn firmware into the microcontroller, so that programming the microcontroller could begin.

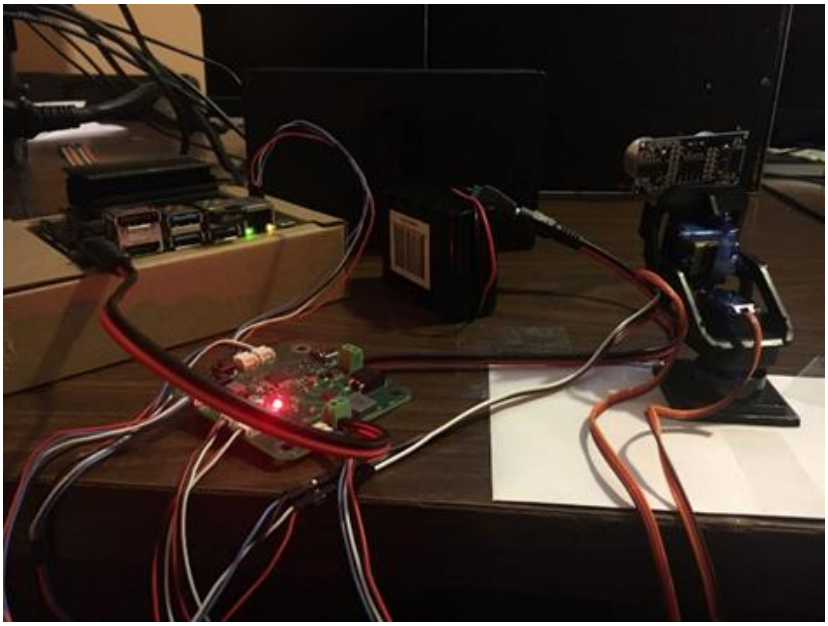

Figure 32: Final assembly of the components

The Portable 3D scanner is complete in terms of components. We have the Jetson Nano with a WiFi adapter for communication between the computer and the Nano. The Nano is connected to PCB through I2C, providing communication between the Nano and the Microcontroller.

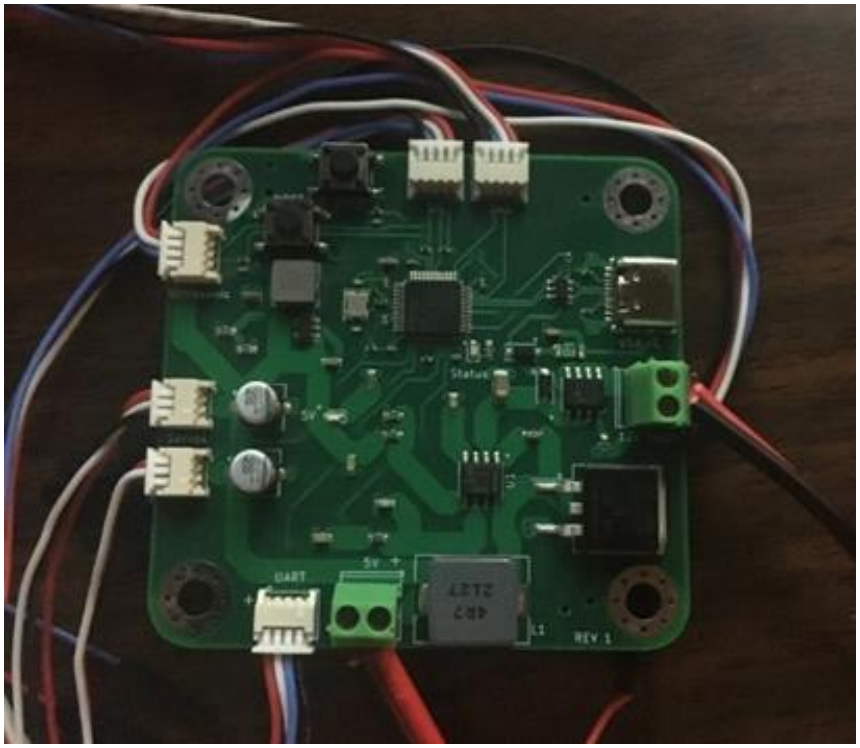

Figure 33: Close up of the Printed Circuit Board

The PCB also supplies power to the Nano. The PCB, powered by a 12V battery, regulates the voltage twice right on the board to power all 5V and 3.3V logic components. There are I/O ports on the PCB used to also connect to the Ultrasonic sensor and the two servos that move the sensor. We have no casing for the scanner's components, so unfortunately the scanner looks a bit loose and messy.

One of our stretch goals was to have multiple ultrasonic sensors for improving scanning accuracy. The ultrasonic sensors would be lined up in what is known as a beam array. The beam array would be used to better keep track of the 15 degree beam that the ultrasonic sensor uses.

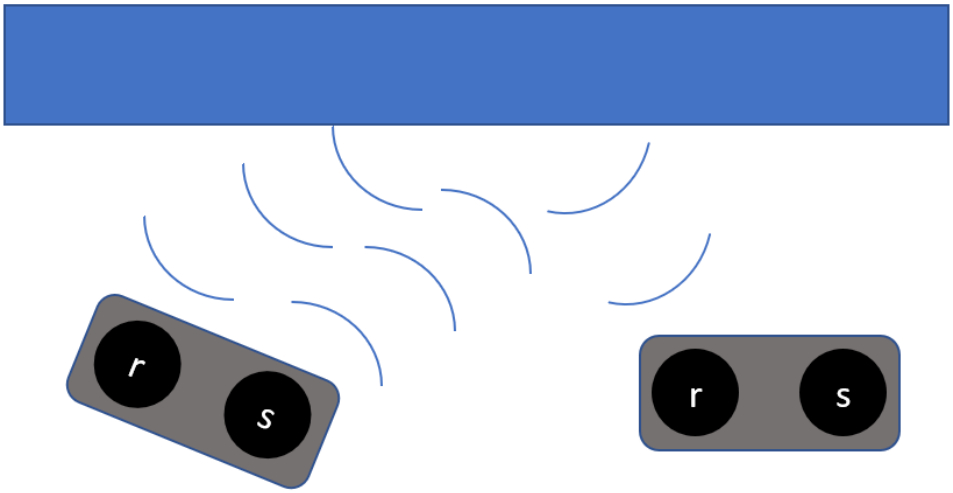

Figure 34: Beam Array example

7.3 Software Testing Environment

As shown here there are two key words to focus on: testing and environment. When we say testing environment in computer software, we basically mean the environment in which users run application software. When talking about testing we clearly mean to investigate the stakeholders with information about the quality of the software product or service under test. So, in our project we will use a software testing environment to conduct the quality of our device.

Following on the fact that the programming language we will be using is python (because the Jetson Nano is an AI based development kit), we will probably use an IDE named JupyterLab. Since we must deal with Arduino parts, we might have to use the Arduino IDE. JupyterLab is web based, so you can use it on a web browser, or install it onto your computer. This can allow portability of files since we can transfer the files we work on a USB, which allows us to use our code wherever the group meets up on campus if there is a computer nearby. This allows our software testing environment to change in accordance with where we'll be testing the hardware.

If we can use C or Java on the Jetson Nano, the programming team will have an easier time in general. It would be a shame on engineering if the computer engineers of the group didn't have any C or Java IDE's on their personal computers. It would also be a shame if the computers in the engineering labs didn't have C or Java IDE's installed as well. So as long as our hardware testing environment is in the engineering labs with computers when we meet up as a group, we can also easily move files from one computer to another with a USB. So, for the programmers of the group, they can be as flexible as they need to be in order to work with the hardware.

For most of the project, programmers will have their software making environment at home, but since there is only one development kit we can use, the only software testing environment we have is the one where the hardware will be tested as well. This can cause our software development to slow down if we don't meet up often.

One important tool we will be using for our project is GitHub. Github is an online repository used to store code and to be shared between programmers. This will help when being transferred between software testing environments. It can also keep track of who made what changes to the code, so it is easy to keep track of who does their fair share of the work.

7.4 Software Specific Testing

In our project we will use testing to make sure our software meets the functional expectation. Therefore, we will a software testing standard. ISO/IE/29119 is a perfect one. The standard ISO/IE/IEEE 29119 the reason we choose that because it is defined an internationally agreed set of standards for software testing that can used by any organization when performing any form of software testing. The standard ISO/IE/IEEE 29119 has different section, or we can call them different part: Test processes part, Test documentation part, and Test techniques part.

Test process:
This part of the standard ISO/IE/29119 is defined the software testing processes at the organization level, test management level and dynamic test levels. This standard also supports dynamic testing, functional and non-functional testing, manual and automated testing, also supports scripted testing and unscripted testing. Furthermore, the process defined in this standard ISO/IE/29119 which is the international standards can be used in conjunction with any software development lifecycle model. It can also use to govern, manage and implement software testing for any organization. Here, we show an image of test process.

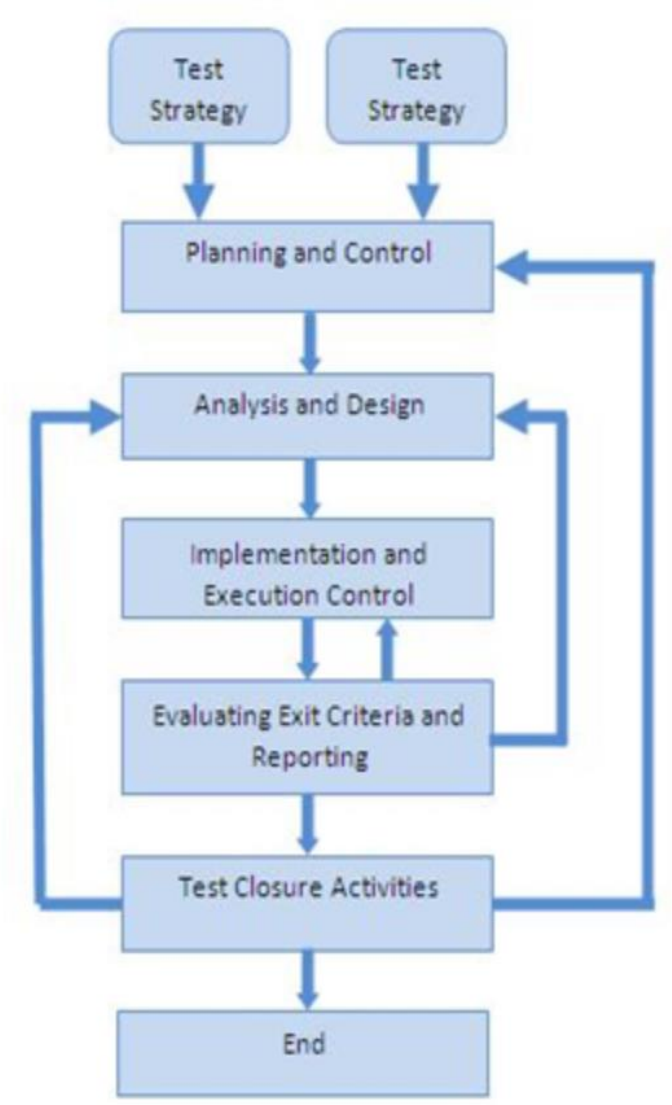

Figure 35: Test Process

Even though there are some other tastings available, but the main test we will use in our project are: Unit testing, integration test and System testing.
Unit testing is referring to tests a specific section of code or function. This test will write by us as the developers of our device. So, each function or specific section of code will test separately. We know that one function might have multiple tests, we also know unit testing alone cannot verify the whole functionality of the software in our device but, unit testing will let us know for sure the building blocks of the software work independently from each other. Then we will have integration testing which is an any type of software testing that seeks to verify the interfaces between components against a software design. We will use integration testing in our project to expose defects in the interfaces and interaction between integrated components. And we will have system testing that will test a completely integrated

system so we can make sure that the system meets all the requirements and perform as it supposes to work. The image bellow shows how the software will be testing and where it will be testing.

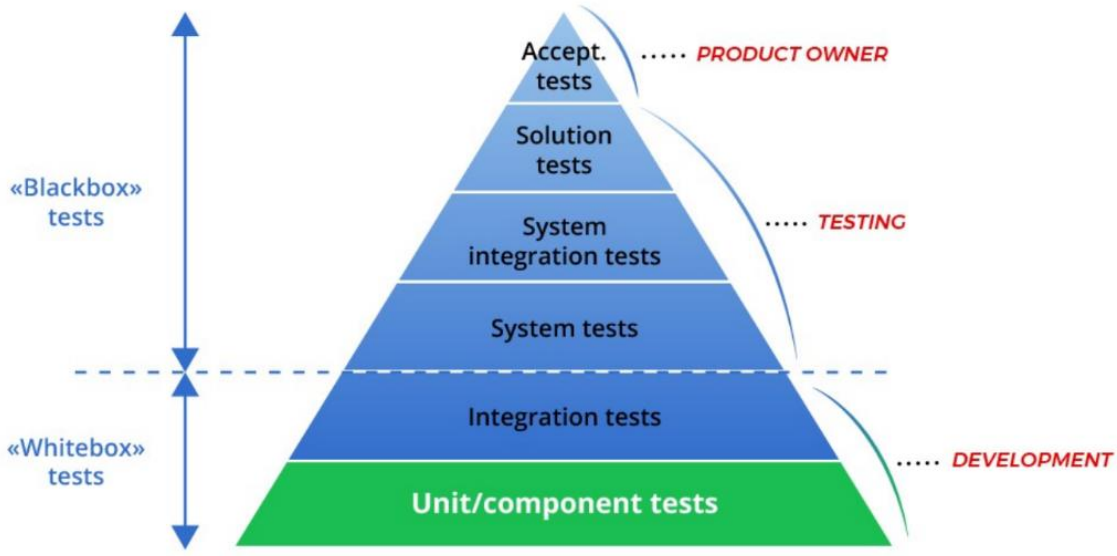

Figure 36: Software Testing

We have to test if the ultrasonic sensor outputs the correct values, so we have to put the ultrasonic sensor 10cm away from a flat surface, and then read the values off of it to make sure it accurately shows us the correct distance. This in turn would have us test the pins of the microcontroller and development board. If we are unable to get an accurate reading, then it's either the ultrasonic sensor, or the fact that something went wrong with the data transferring between the ultrasonic sensor and the microcontroller, or between the microcontroller and the development kit.

The next thing we must do is test out the Development Kit (the Jetson Nano). There are many things we must do in order to get the Jetson Nano up and running. First off is we have to make sure we have the Jetson Nano's image files on an sd card so then it can actually operate. After flashing the image into the microSD card, we can put the microSD card into the Jetson Nano and see if it operates smoothly. Then after setup is the most important part, testing how well we are able to get the Jetson Nano to run C and Java programs. There is almost no doubt that it can run C or Java, but most features in the Jetson Nano are attuned to python due to it being mainly an AI development kit. We have to check if things that are run using python can be run using the more commonly known programming language most group members know. We also have to be able to run JetBot on the Jetson Nano. JetBot is basically what turns the Jetson Nano into a quick responding robot that, for example, can be used to turn into a self-moving vehicle (also using an ultrasonic

sensor to determine how close an object is to it). We must test our development kit's ability to use the ultrasonic sensor in a (somewhat) similar fashion. JetBot, however, uses python as its programming language, so we have to test if we can recreate something similar to JetBot in C or Java, or find alternatives to JetBot.

Our prototype right now takes less than 15 minutes to scan the environment it's looking at. This is while the scanner had an FOV of 31 by 31, which brings the total number of points to 961. In the future after more testing and fine tuning is complete, we will be increasing the FOV to 45 by 45 so that our scanner can view a larger area. However, this would increase the number of points to 2025, which is more than double the points originally scanned. This would lead to the scanner taking more than 30 minutes to complete one scan. We are currently testing whether or not making the scanner run faster will negatively affect the scan. If the scanner running faster does not hurt the scan we will increase the speed, because if we want to be able to run multiple scans at the same time to average out the scan, we need to be able to run it faster so we don't end up spending 5 hours on one scan (with multiple runs), 10 if we increase the FOV.

Figure 37: Flat Edge Testing Object

Previously, when trying to scan a flat surface, we were unable to scan flat edges. The ultrasonic sensor would detect bumps or spikes when scanning just a flat edge.

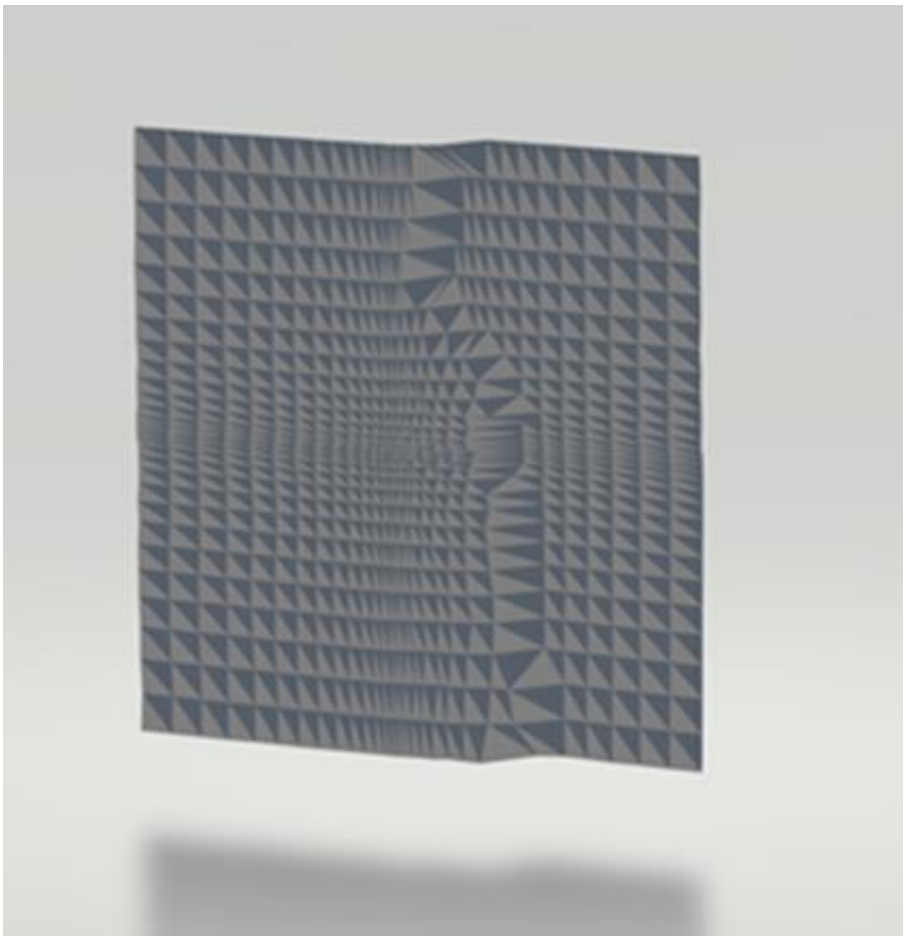

Figure 38: Example 1 of a bad flat edge scan

The scan above is one of our very first scans with the scanner. As you can see in comparison to figure 37, the depth difference between the box and the flat surface is much less than it actually is. Another obvious problem is the flat surface appears curved rather than straight. The image below was one of our more recent scans (before we eventually solved the problem). The 3D scan appears mostly flat, while at one point there is a tremendous spike, causing the scan to be inaccurate. To be later described, we learned that this was all due to the calibration of the scanner.

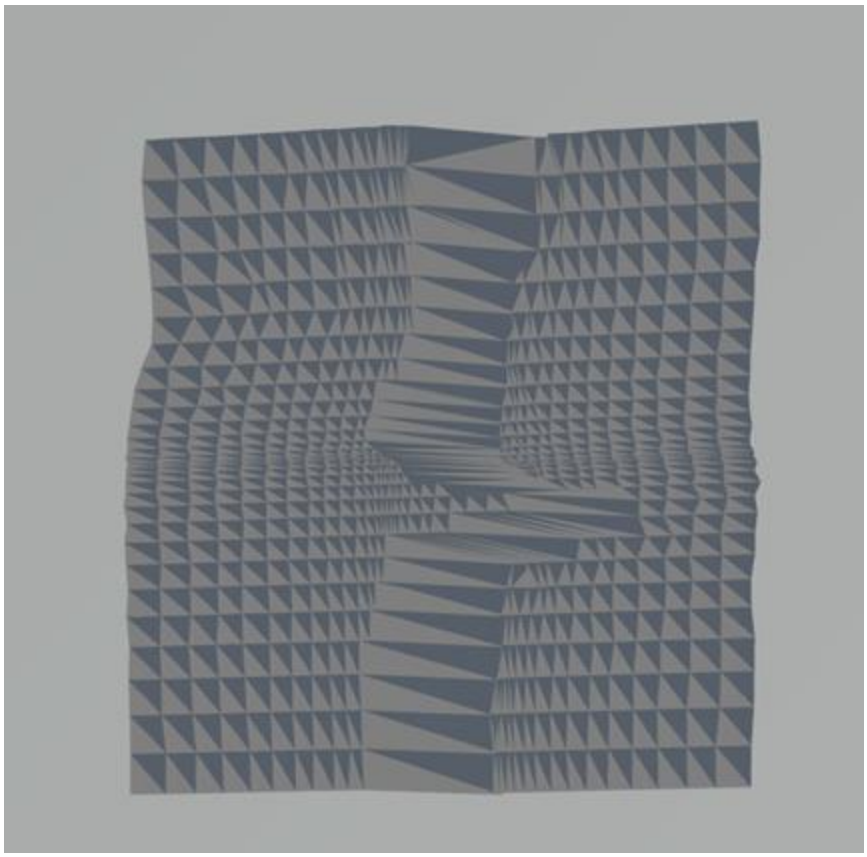

Figure 39: Example 2 of a bad flat edge scan

As you can see in figures 38 and 39, the flat edge between the two surfaces is not straight. There were some issues that could have caused this problem. One is that the ultrasonic sensor we use is not accurate enough. This is simply a matter of changing the ultrasonic sensors to a different one that boasts more accuracy, or by changing the way that we collect our data, either infrared or lidar. The other reason might be that the servo's jittering causes inaccuracies, this is because the servo goes in a set path collecting data at every point, sometimes the servo might shake too hard when going in its set path that it begins to slowly deviate the actual point at which it moves to. We were given recommendations as to how to solve this problem as well. One, if the scan was not consistent in the deviations on the flat edge, we could have the scanner scan multiple times and average the data out. The other solution was if the scan deviations are consistent, then we should calibrate the machine. At first trying to average out the data was our plan, but unfortunately the final product would appear almost flat, decreasing the quality of the scan.

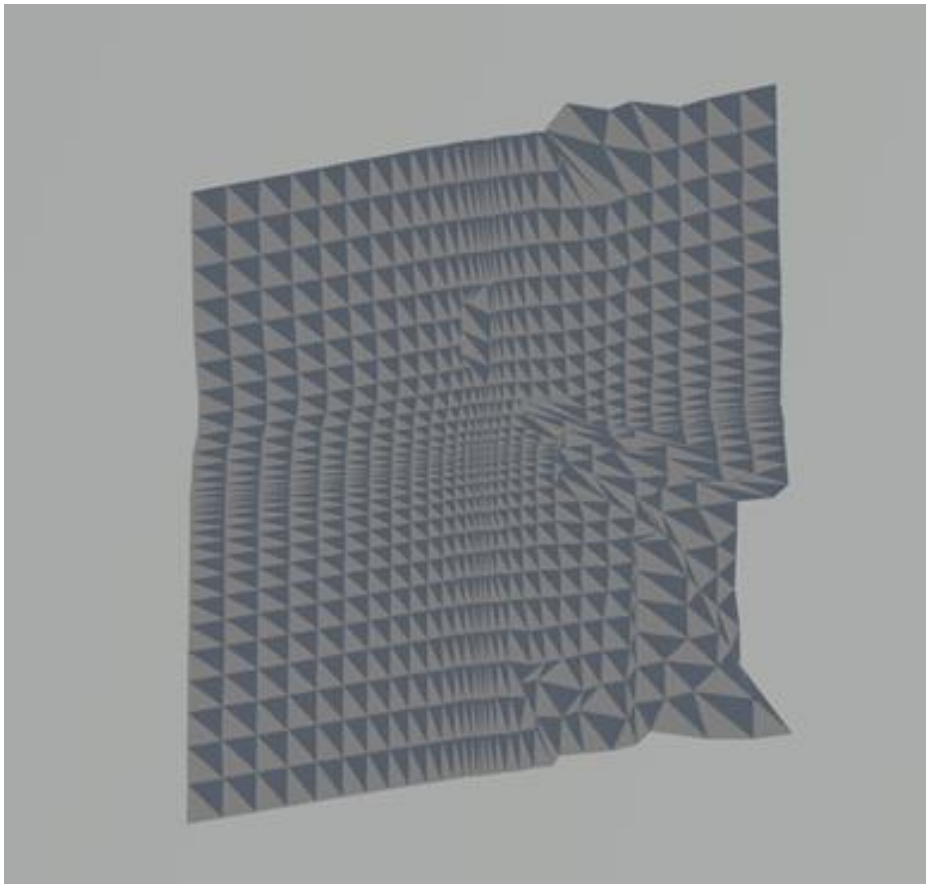

Figure 40: 3D scan implementing our multiple scan solution

As this method seemed unviable, we determined that calibrating the scanner would be our best option. Upon further testing we discovered that the lower our scanner scanned, the more distorted the image would be. We raised the FOV by a few angles and the scanner was able to detect a flat edge.

This shows that the scanner was able to scan flat surfaces with flat edges. This led to further testing with different edges and surfaces. When scanning a curved or round surface, the ultrasonic sensor will either determine the object is flat or nonexistent. This is most likely due to the nature of an ultrasonic sensor, using the bounce of sound waves to determine distance. The only way to fix this would be to change the type of sensor used. The other test we did was testing a flat surface with a round edge, this led to results where we could see the curve of the edge.

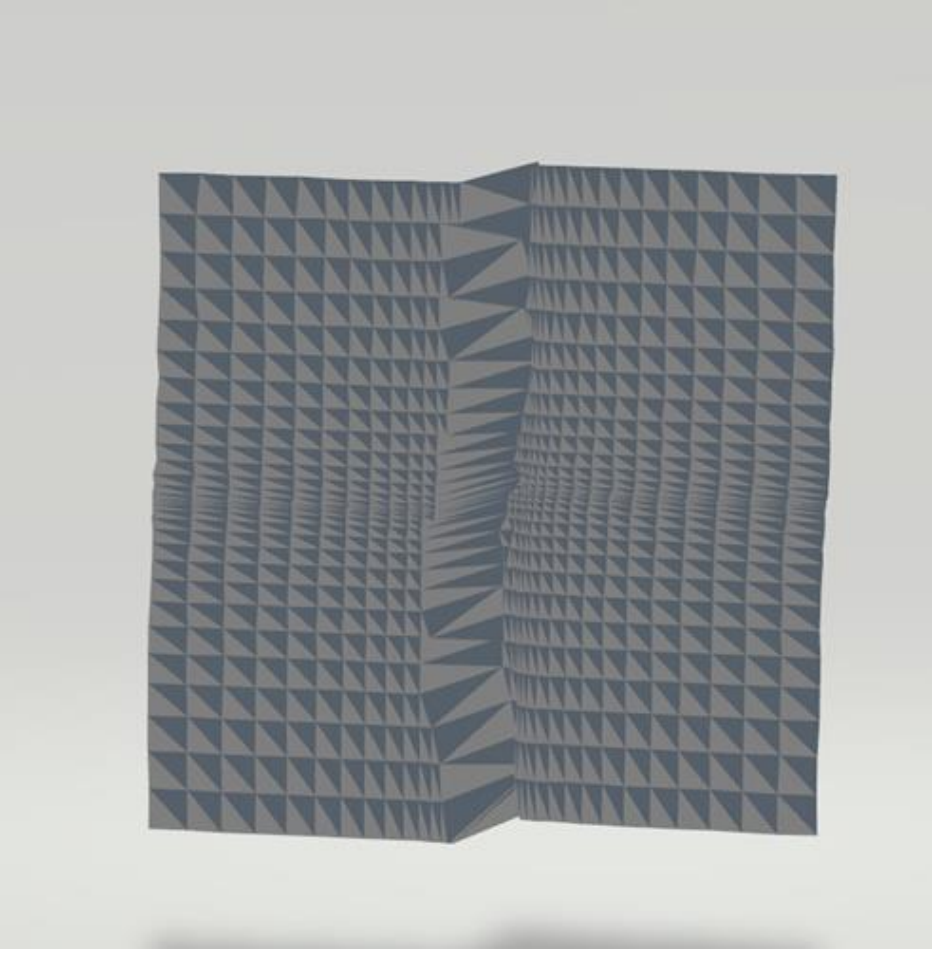

Figure 41-1: Scanning results of a flat edge with flat surfaces

Rows 11 to 20, Column 3

Unprocessed Data	Processed Data
15.3	14.92
15.2	14.84
15.2	14.85
15.1	14.76
15.0	14.67
14.9	14.57
15.0	14.66
14.5	14.16
14.6	14.24
14.5	14.12

Angles used are (abs(Row# - 15)) and (abs(Col# - 15))

Figure 41-2: Scanning results of a flat edge with flat surfaces

As a test we measured just a flat object, we wanted to know if the scanner was able to see a flat surface. This lets us make sure we are scanning properly. As seen through the results below, except for one point in the very center, we have a completely flat image.

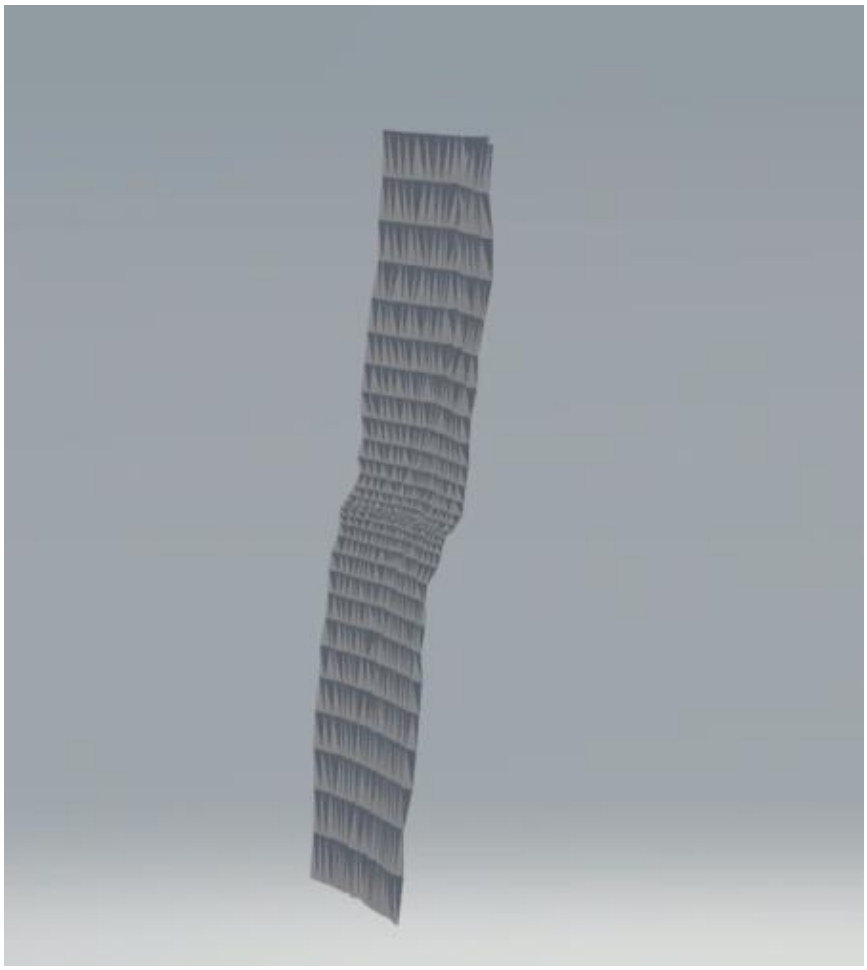

Figure 42: 3D scan of a flat surface.

Next was testing a Flat Object with a Round Edge. For this test we used an index card that we shaped into an object with a round edge. The ultrasonic sensor was able to determine that the object was flat and the edge was round as seen below. One issue with this scan however is that the rounded edge appears sharper than it actually is.

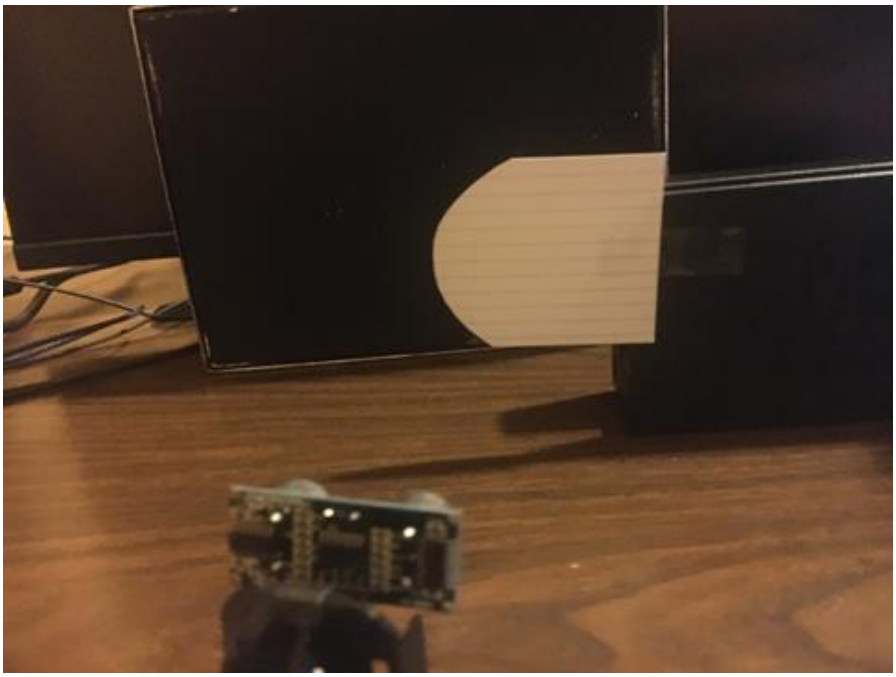

Figure 43-1: Scanned object with a round edge and flat surface

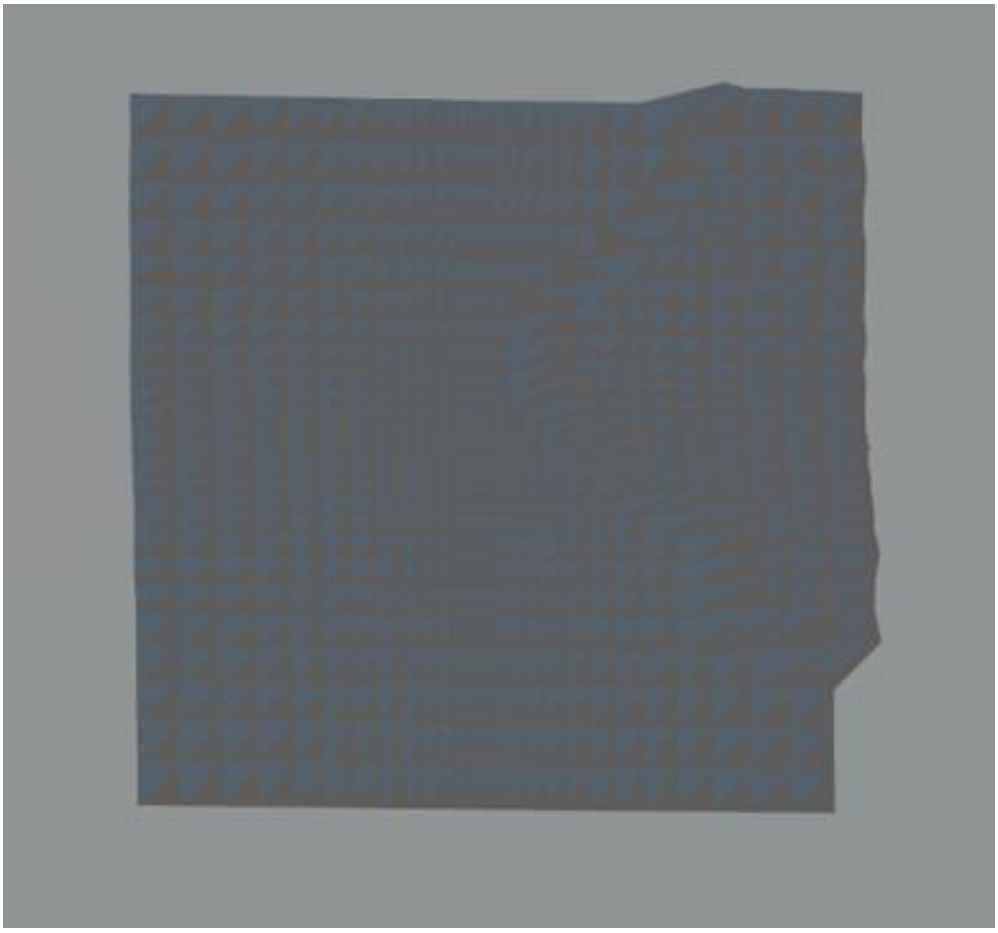

Figure 43-2: Scanning results of a round edge with a flat surface

The third object we scanned was a can of pineapples. This can had a flat edge and a round surface. As you can see below, the scanner was unable to determine that the surface was flat. Furthermore, because of the curvature of the can, it negatively affected the scanners ability to see a flat surface.

Figure 44-1: Object with a flat edge and round surface

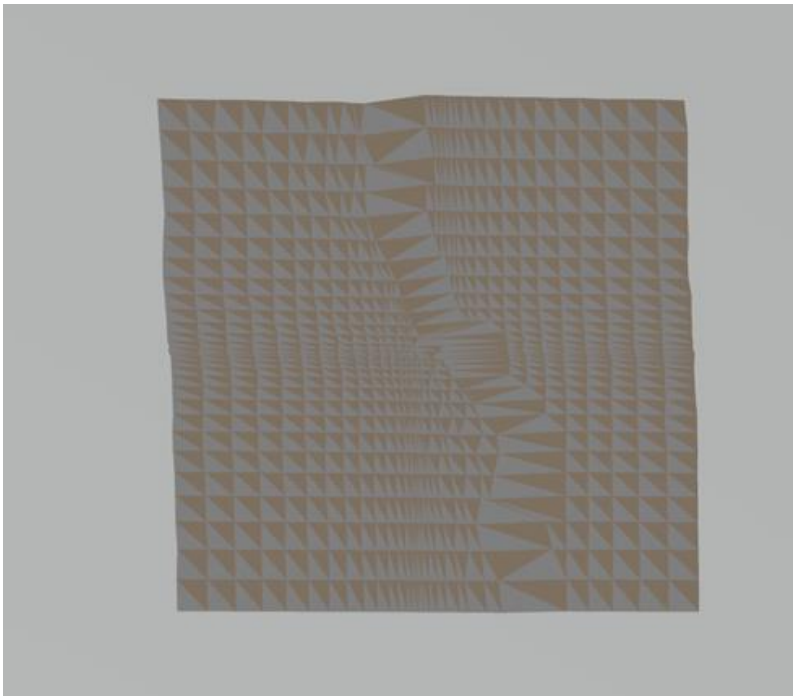

Figure 44-2: Scanning results of a flat edge with a round surface

The last thing we scanned was an object with a round edge and a round surface. Specifically in this case it was a bottle. The bottle as shown below was unable to be scanned properly at all. The scanner was able to detect something was there, but the surface is still shown to be flat, and the edges of the object are incredibly inaccurate.

Figure 45-1: Object with a round edge and round surface

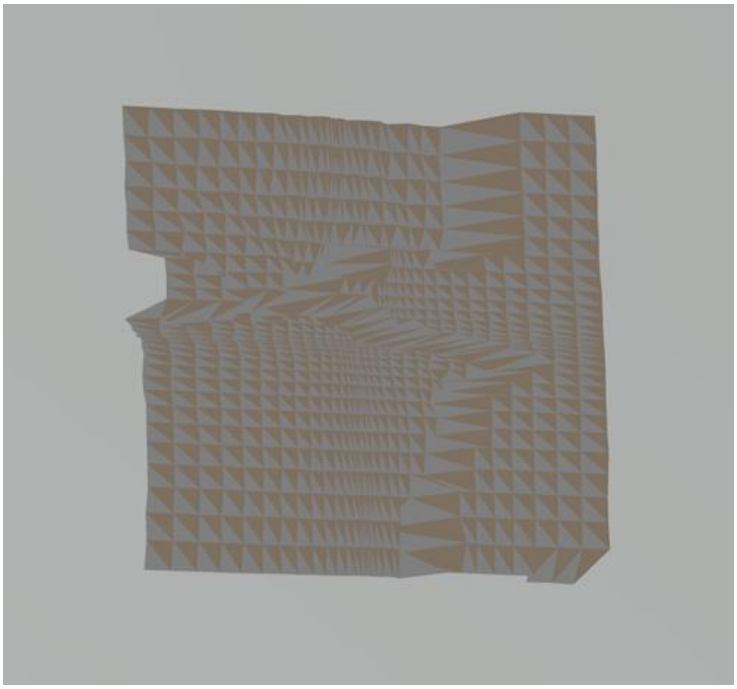

Figure 45-2: Scanning results of a round edge and round object

After a lot of testing, it has been determined that our scanner is able to detect edges, whether they be round or flat. However, the scanner is only able to create flat surfaces, even if the object itself is round. This proves to us that of now our scanner is good at finding the layout of an area, or just being able to detect an objects position. Looking at the results, we were able to determine that the Ultrasonic Sensor is the reason for the inaccuracies of the scans. The Ultrasonic sensor having a beam of 15 degrees caused the scan to be not precise. If we wanted a more precise scan, we would need to use a sensor with a much smaller FOV.

7.5 Conclusion

In the end we learned that no matter what, you should always work on something every day, no matter how small the progress. The more work that is done sooner, the less you have to do in the future. We learned that communication is key in any work environment, no matter if it is in person or online. When trying to communicate with each other, we noticed that many of the members would be offline due to school or work. This led to us leaving more time to check up on our members whenever possible. The last thing we learned was that when working on something, always prepare for mistakes or errors. When finishing something, there would always be one bug here or one error there.

The portable 3D scanner prototype which we assembled shows that with enough time and effort it is definitely possible to create a portable 3D scanner. The only thing we would change is the type of sensor we would use. Different sensors we could use would be Lidar, infrared, or Time of Flight. The reasons for why those weren't options during our time prototyping is because of multiple reasons, the most impactful is that we didn't have enough time. So overall, in terms of feasibility, it is very feasible to create a good 3D environment scanner for a much lower cost than what is on the market. Testing has shown that some scans are really good for something that is worth less than 2% of 3D scanners on the market today. With more time, more improvements could be made, but with what results we have gained we can safely say that it is definitely feasible to create a low-cost 3D Portable Scanner.

7.5 Engineering Specifications

We tested three different Engineering Specifications. The ultrasonic sensor accuracy, the time it takes for the scanner to scan one point, and the scanners' ability to go back to the same point every scan. The ultrasonic sensor has a guaranteed accuracy of half a centimeter, but when tested we noticed an accuracy of +/- 3 millimeters, better than expected. Each point scan took 0.75 seconds to scan. Finally, after multiple tests, we discovered that not every point is scanned the same, we see the scanner scan different points with different distances.

8. Administrative Content

8.1 Milestone Discussion

When it comes to the project milestone this is where we will keep tracking our project progress. This will represent a clear sequence of even that incrementally build up until our project is complete. So, another to keep tracking our progress we will use Trello board. Trello board is a collaboration tool that organizes your projects into board. In Trello board we will be able to know what's being worked on, who's working on what, and where something is in a process, also when something is done. So, right now to start with our milestone discussion, all our major part will be order by the end of July the latest date is Friday July 30th and the testing part will be complete by the end of this semester.

8.2 Budget and Finance Discussions

As we already mentioned, we are the sponsor of our project, therefore all the cost will be from us as a group of developers and sponsors. We had several meetings where we talked about our budget for this project. In last meeting we discussed our we plaining on buying the parts for our project. We came with a solution where one person will be ordering the parts and we will split the total cost with four people. We also talked about the number of parts that we will buy another to stay on track with our project. For the parts that is not too expensive we are plaining on buying two of them and the parts that is not too expensive we are plaining on buying at least three of them. We get that idea from the professor and realize this is a good idea, because of the development of technology the manufacture may decide not to make that product anymore or runout of stock that may cause a delay in our project deliverable, that is the reason why we order more than we will need just so we can have some backup plan.

The total cost of the project is imperative to our study as our intent was to create not only a portable scanner, but one that is affordable that gives our scanner an edge compared to the traditional scanners that already exist in the market. We wanted to ensure that the cost of the scanner was kept as low as possible, but also not at the cost of accuracy or ability. We wanted to find the most optimal way of ensuring quality engineering and design, but also affordability. Many of our component selections took into account their features, but also their price. Our PCB was designed to be small and lightweight, which led to a more compact (but more layered) design.

The cost for our PCB was only increased due to the fact we ordered multiple, in fear of one malfunctioning. On the other hand we had more flexibility in addressing more affordable options on the sensors as SR04 sensors which we chose were extremely powerful for the price compared to the premium URM37 sensors which were nearly 4x more expensive. The flexibility in picking sensors allowed for more budget allocation towards other components that had to be constrained by our

project's core goals such as size and portability of the scanner. The scanner prototype did fit our vision of creating a scanner that is smaller in size, and relatively lightweight. Although it is not made in a way that can be handheld, there is definitely a way to make the scanner in a handheld shape, our main goal for the prototype was to assemble the scanner to just scan and realize our goal of scanning for an object in its FOV.

When ordering the PCB, we ordered 5 boards with 2 of them assembled. The cost of all this was around $113, so the cost of one assembled PCB is about $50. So if added to the chart below, the final total becomes around $300. The total cost for one scanner as we have it would be $185 dollars. This is a much better cost compared to the $10,000 to $20,000 scanners on the market.

Item	Price
5 PCBs with 2 assembled	$113.49
3pcs Ultrasonic Sensors	$9.99
12V NiMH Battery	$29.95
UART and SWD Programming Adapters	$35.00
Jetson Nano 4GB	$99.99
Pan/Tilt Servo kit	$9.99
Total ----------	$298.41

Table 15: Final Budget Chart

9. Final Notes

9.1 Project team

Jean Cestin is a computer engineering student at the University of Central Florida. He is pursuing a career in Computer Hardware.

Sergio Arciniegas is a computer engineering student at the University of Central Florida. He is pursuing a career in Cybersecurity.

Rayan Hamada is a computer engineering student at the University of Central Florida. After graduating in the Fall of 2021, he has been hired for Lockheed Martin in their software department and will be joining them in January.

John Paszynski is an electrical engineering student at the University of Central Florida. He is pursuing a career in the RF/Communications field.

9.2 Possible Improvements

Over the course of testing, we realized that the reason for the inaccuracies in our scanner was a hardware error, more specifically, an ultrasonic sensor error. The ultrasonic sensor has a narrow beam angle of 15 degrees. 15 degrees is essentially half the FOV of our scanner. What caused the issue was that when scanning a point, it would detect the closest point in a 15 degree angle, so when scanning an object with a curve or rounded surface, the ultrasonic sensor will only detect the closest point to the scanner, thus causing the illusion of a round object being flat. The only possible way to improve this is to get a sensor that can measure distances at a much lower FOV. Sensors that we have discovered with low FOV's are Lidar sensors, infrared sensors, and time of flight sensors. With those, with an FOV of one degree or less, could give us a much more accurate scan.

Additionally, another improvement would be to create a dedicated interface for the scanner so that users would be able to interact directly with their scanned images as opposed to having to using a command line in order to interact with the scanner. The interface would also let users edit images, export and save them as well. Having a dedicated interface could then lead to creating a community where users can create profiles and share images, projects etc.

Given the scanner can have Bluetooth connection, another improvement could be to incorporate an iOS app that can function with the scanner. This would allow for users to have even more ease of access to their images immediately after the scanner has taken an image. This would be beneficial for users taking scans in a place where they can not immediately access their computer, and therefore be able to delete images they wish to not keep so that they can take as many scans as they can while they are out, and directly use their phone to access them and export.

Appendices

Appendix A - Copyright Permissions

As a definition, a copyright is a bundle of rights to reproduce, derive, distribute, perform, and display an original creative work. Therefore, we do research about related devices, devices that is similar to what we are doing. Most products we will use will be purchased from a tech company to avoid getting penalties for copyrights.

Appendix B - Datasheets

Appendix C - Bibliography

[1] *Comparing the Different Interface Standards for Embedded Vision*. Automate. (2020, March 17). [Online]. Available: https://www.automate.org/blogs/comparing-the-different-interface-standards-for-embedded-vision.

[2] *Discover Wi-Fi CERTIFIED 6* . wi-fi.org. (n.d.). https://www.wi-fi.org/discover-wi-fi/wi-fi-certified-6.

References

(PDF) Image and ultrasonic sensor fusion for object size detection. (2019, November 15).
ResearchGate. https://www.researchgate.net/publication/337464551_Image_and_Ultrasonic_Sensor_Fusion_for_Object_Size_Detection

3D image processing. (2021, July 19). Synopsys | EDA Tools, Semiconductor IP and Application Security Solutions. https://www.synopsys.com/glossary/what-is-3d-image-processing.html

3D scanner market. (n.d.). Market Research Reports, Marketing Research Company, Business Research by MarketsandMarkets. https://www.marketsandmarkets.com/Market-Reports/3d-scanner-market-119952472.html

3D scanners. (n.d.). Professional 3D scanning solutions | Artec 3D. https://www.artec3d.com/portable-3d-scanners

3D scanning market size, share, growth | Global report, 2026. (n.d.). Fortune Business Insights™ | Global Market Research Report & Consulting. https://www.fortunebusinessinsights.com/3d-scanning-market-102627

3D scanning market, size, share, trends & industry analysis. (n.d.). Allied Market Research. https://www.alliedmarketresearch.com/3D-scanning-market

Basics of programming a UART. (n.d.). ActiveXperts - Network Server and IT Infrastructure monitoring. https://www.activexperts.com/serial-port-component/tutorials/uart/

Compressive 3D ultrasound imaging using a single sensor. (n.d.). Science Advances. https://www.science.org/doi/10.1126/sciadv.1701423

Creaform. (n.d.). PorTable 3D scanners: HandySCAN 3D, GoSCAN 3D & MetraSCAN 3D. Portable & Robot Mounted 3D Scanners and CMM Solutions | Creaform. https://www.creaform3d.com/en/portable-3d-scanners

(n.d.). Google. https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwinuoeZstD0AhUBRjABHbkeBVMQFnoECCUQAQ&url=https%3A%2F%2Fwww.mordorintelligence.com%2Findustry-reports%2F3d-scanners-market&usg=AOvVaw0AUO8VKOU08EkKJRxiyoUG

How to use 3D scanning and 3D printing for reverse engineering. (n.d.). Formlabs. https://formlabs.com/blog/how-to-use-3d-scanning-and-3d-printing-for-reverse-engineering/

Indoor 3D reconstruction using camera, IMU and ultrasonic sensors. (2020, June 29). SCIRP Open

Access. https://www.scirp.org/journal/paperinformation.aspx?paperid=101158

Jetson Nano developer kit. (2021, April 14). NVIDIA Developer. https://developer.nvidia.com/embedded/jetson-nano-developer-kit

Keith Araujo - Epec. (n.d.). Battery standards. Manufacturing That Eliminates Risk and Improves Reliability | Epec. https://www.epectec.com/batteries/battery-standards.html

Lesson 9: UART. (2016, April 24). Simply Embedded. https://www.simplyembedded.org/tutorials/msp430-uart/

Lithium-ion battery standards. (2020, June 30). U.S. Agency for International Development. https://www.usaid.gov/energy/powering-health/technical-standards/lithium-ion-batteries

LPC1768: UART programming. (n.d.). Tutorials. https://exploreembedded.com/wiki/LPC1768:_UART_Programming

Microcontroller programming. (2020, April 7). Build Electronic Circuits. https://www.build-electronic-circuits.com/microcontroller-programming/

(n.d.). OpenScan. https://en.openscan.eu/

Standards & certification FAQ. (n.d.). The American Society of Mechanical Engineers - ASME. https://www.asme.org/codes-standards/publications-information/faq

Top 3 standards for lithium battery safety testing. (2018, June 28). Eurofins E&E North America. https://www.metlabs.com/battery/top-3-standards-for-lithium-battery-safety-testing/

What is 3D image processing? (with picture). (n.d.). wiseGEEK: clear answers for common questions. https://www.wise-geek.com/what-is-3d-image-processing.html