3D Portable Scanner Feasibility Study

Rayan Hamada - CPE
John Paszynski - EE
Jean Cestin - CPE
Sergio Arciniegas - CPE

Team Roles

- Rayan Hamada
  - In charge of Image Creation
- Jean Cestin
  - In charge of component and PCB setup
- John Paszynski
  - In charge of PCB design and testing
- Sergio Arciniegas
  - In charge of Data Processing

Motivation

- We want to help independent modelers, game designers, and 3D hobbyists obtain a portable 3D scanner for a much more affordable price.
- We want to push for innovation in other companies to produce an even better scanner to compete with ours (at better prices).
- We want to improve educational opportunities with 3D scanning through cheaper lab equipment costs.

Goals

- What we want to do is determine whether or not it is possible to create a Portable 3D Environment Scanner, looking at a certain FOV and bringing that environment to the users computer as a 3D image file.
- We want this to be at a much lower cost than other portable 3D scanners on the market, which can range from $10,000 to $20,000.
- The impact we hope to make is to improve upon what is available for 3D scanning so that eventually 3D scanners can become more available to the public

3D Scanners

- There are two types of 3D Scanners
- The first type is a room scanner, in which the scanner looks around and maps an area
  - This is the type of scanner we are creating
- The second is an object scanner, in which an object is placed on a turntable and the scanner scans the object in 360 degrees
  - Due to the adding of extra mechanical components, we decided not to do this.

Scanner Requirements

- 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 $300.

Scanner Requirements

- The device must have an interactive button to capture images within 30 seconds to 1 minute as well as an I/O switch for power.
- The device power will be powered via USB 2.0/USB 3.1.
- The power supply must last for a minimum of 2 hours before recharging.
- The device should be powered by a NiMH 7.2V battery.

Scanner Requirements

- A buck converter is needed to step-down 5V to 3.3V for the microcontroller.
- An ultrasonic sensor must be utilized for depth sensing.
- Ultrasonic sensors should operate within 1 to 3 meters.

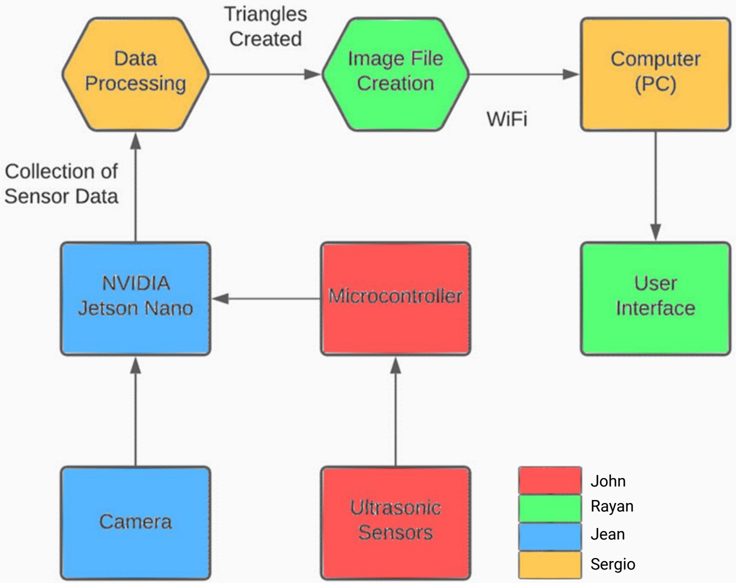
Constraints

- COVID limited the number of in person meetings
- Parts going out of stock caused redesigns in the PCB and the need to look for new PCB Manufacturers
- The time limit for our project left us with less time than desired to include more improved algorithms or better sensor types.
- The hardware design had to be redesigned a second time due to the switch from three ultrasonic sensors to only one ultrasonic sensor.

Standards

- **IEC 60086**: focuses on the standardization of batteries in regard to dimensions, terminal configurations, markings, test methods, performance and safety/environmental aspects.
- **Nasa C style guide**: Nasa states that in its purpose their Software Engineering Laboratory (SEL) recommends code that is in “good style” where it is: Organized, Easy to read, Easy to understand, Maintainable and Efficient
- **ISO 5577:2017** provides a guideline for usage in ultrasonic non-destructive testing that forms a basis for standards and general use

Prototype Flowchart



Camera

- Our original plan for the 3D scanner was to include a camera along with the sensor to collect color for the 3D image.
- There were two reasons why we turned the camera into one of our stretch goals.
  - We wanted to first make sure we could create a working 3D scanner before worrying about color.
  - Adding color to the 3D image would require us to find and learn a new 3D image file format, because the most common 3D image file types don’t support color.

Jetson Nano Development Board

- 2-lane CSI interface for dedicated camera sensors.
- Dedicated GPU for image processing.
- Up to 64GB storage with SD card.
- Easy to use Linux OS to set up the board.

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

Firmware

- Flashing the STM32F1 took a while, we started off with using USB, however since our firmware wasn't initialized, this resulted in errors.
- Moving on to trying to use UART, we once again failed because the UART tool we used drew 5V, however we supplied 3.3V to UART, causing the tool not to work
- Finally, we used SWD using a ST-Link, which we were finally able to use to flash firmware into the microcontroller, making it useable through USB

Software - Microcontroller

- Wiring the microcontroller through I2C to the Jetson Nano, we were able to create request and receive events for the PCB.
- The request event is for collecting ultrasonic sensor data. When requestEvent is called, it call a function which activates the ultrasonic sensor and sends the distance in millimeters.
  - This function is called 10 times, and the average of the data is sent to the Nano
- The receive event happens when the Nano wants to change the positions of the servo's

Reason for Scanning 10 times

- While scanning a single point, we have the ultrasonic sensor scan the distance it sees 10 times.
- The reason why is because while the ultrasonic sensor is scanning, when an object is 14.5 cm away, it will return 14 or 15 cm.
- So by having the sensor scan ten times, we would see something like [14,15,14,14,15,15,15,14,14,15]. Averaging out the data would give us 14.5

Average Differences

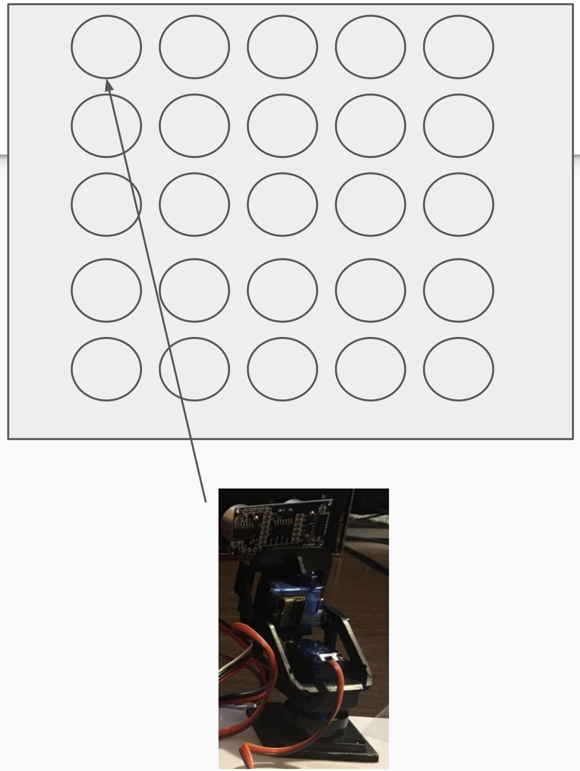
- There is a major difference between having the ultrasonic sensor scan 10 times, verses having the scanner itself scan 10 times.
  - When the servos move during a second scan, there is no guarantee they will move to the exact same position they were in during scan one. This causes slight variations that ruin the scanner and make it look more spikey.
  - Having the ultrasonic sensor scan 10 times rather than going through the scan 10 times saves a lot of time, the servos move a lot less creating a massive difference in scanning time.
- For those reasons we have the ultrasonic sensor collect data 10 times, then have the microcontroller average it out and send it to the Nano

Software - Jetson Nano

- While programming the Nano, we came up with three phases that must be completed to create a 3D image.
- The First Phase is the Data Collection Phase. This is where the Nano sends request and receive events through the I2C bus to the microcontroller. The Nano moves the servos in a 31x31 grid, and at each point collects data from the Ultrasonic sensor

Field of View of Scanner

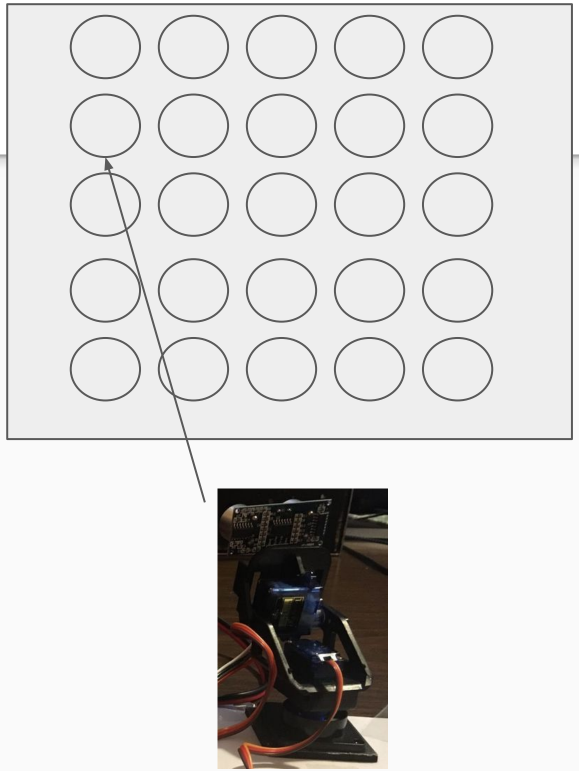
31 x 31



Scanner

Field of View of Scanner

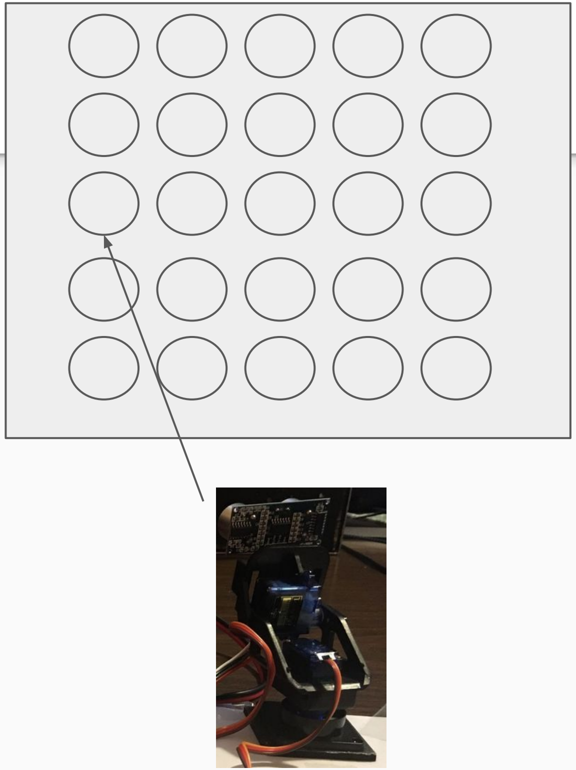
31 x 31



Scanner

Field of View of Scanner

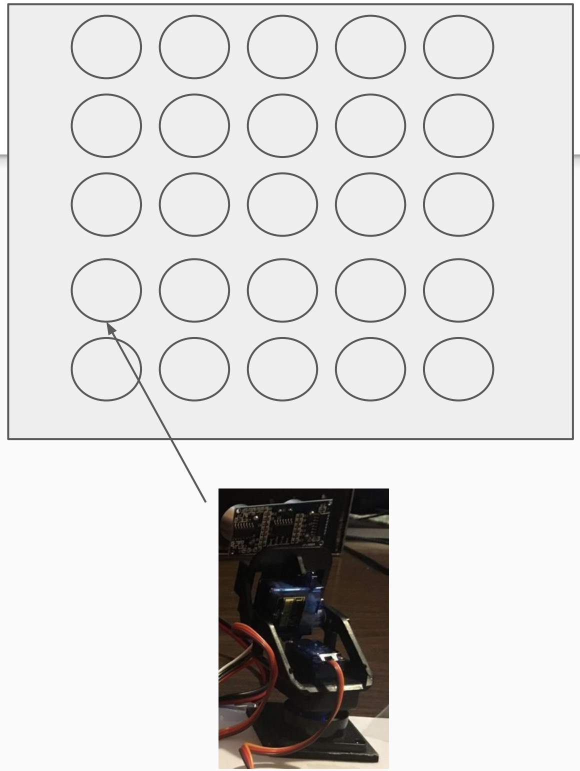
31 x 31



Scanner

Field of View of Scanner

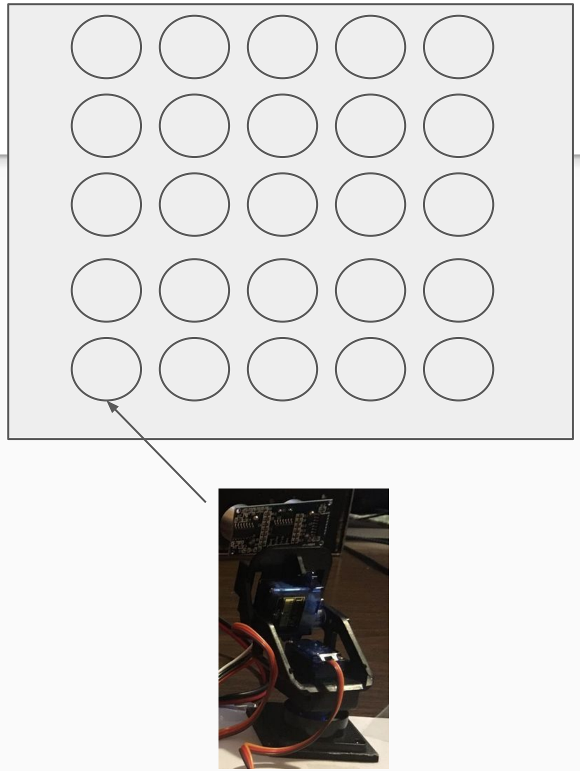
31 x 31



Scanner

Field of View of Scanner

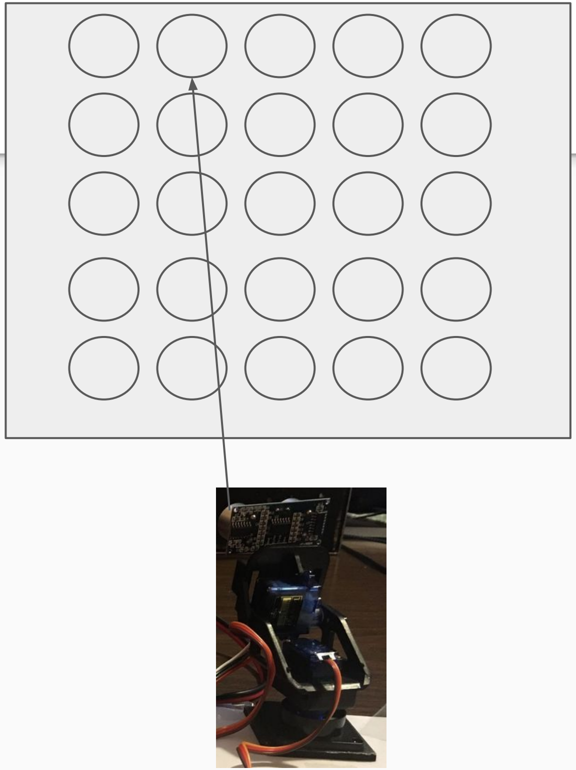
31 x 31



Scanner

Field of View of Scanner

31 x 31

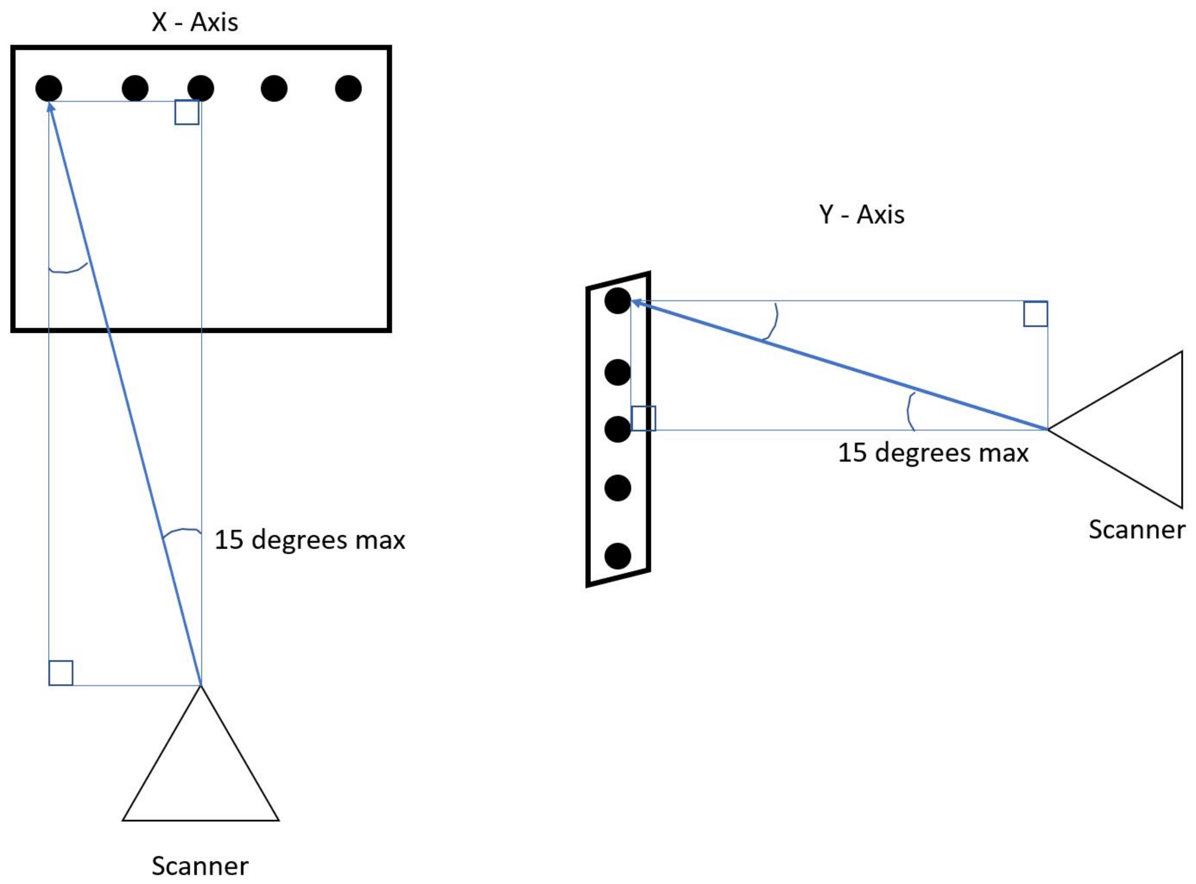


Scanner

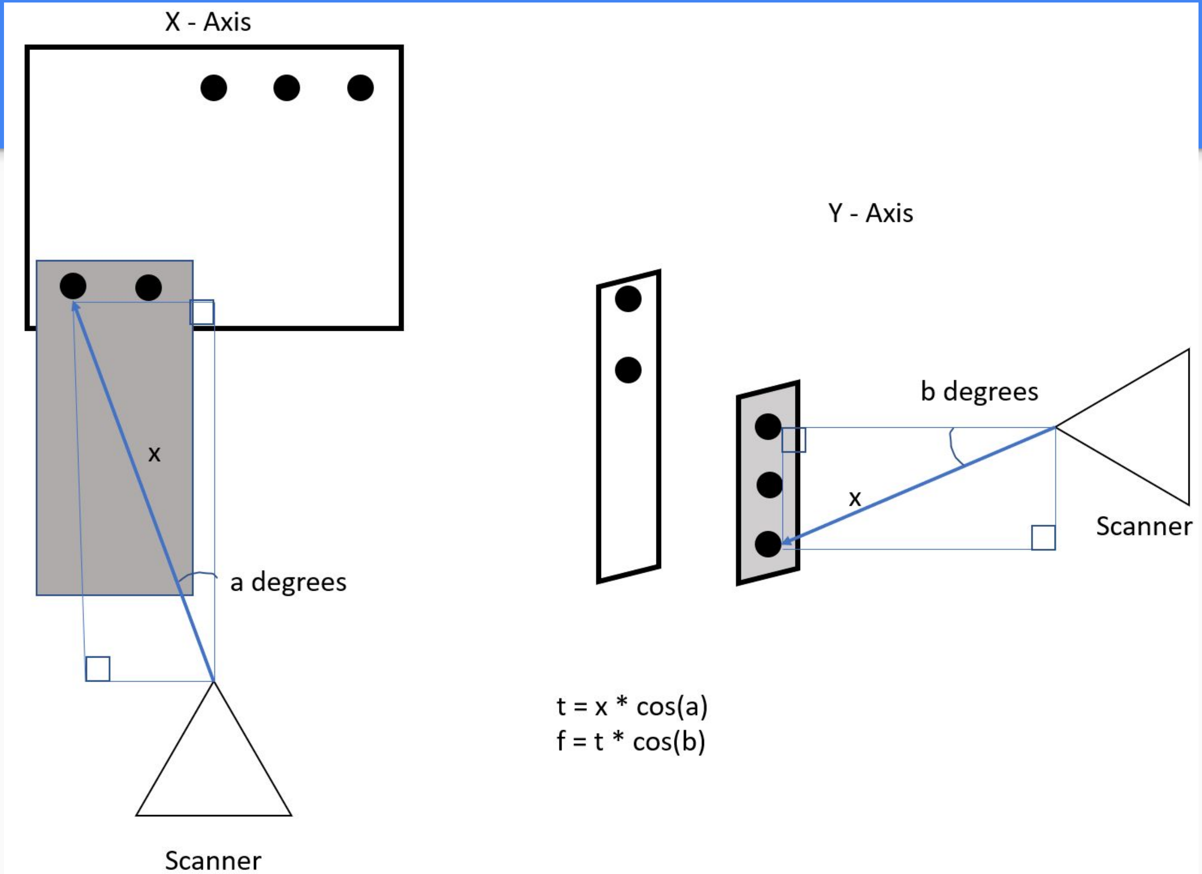
Software - Jetson Nano

- Phase Two is the Data Processing Phase
- Since the Ultrasonic sensor is scanning at an angle, we have to compensate for the fact that the distance scanned might be larger than the actual distance between the scanner and the object, since what is being measured is the hypotenuse.
- Therefore, we use trigonometry to convert the angled data into the real distance
  - By using the position of the data in the array, [15][15] being the center, we can determine the two angles to be used, along with the data scanned as the hypotenuse, we can use COS twice to create the proper data.

Scanning example



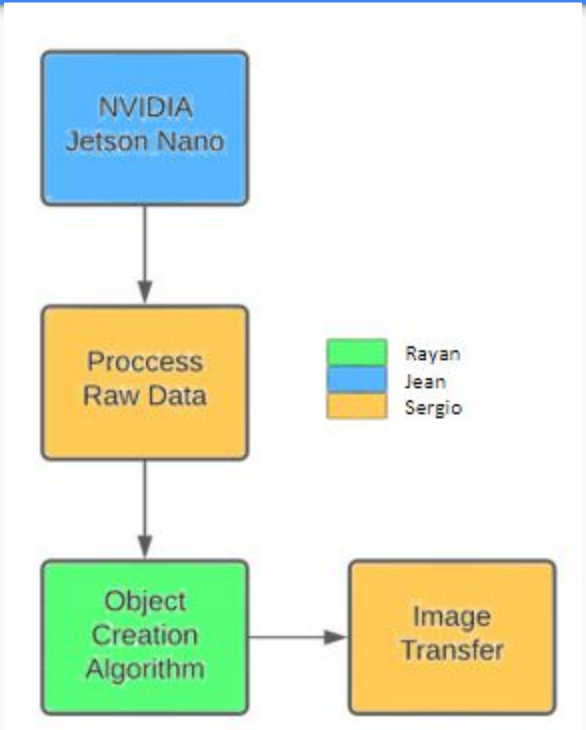
Scanning example



Software - Jetson Nano

- Phase 3 is the Image Creation Phase
- The method for image creation is derived from an AI compressing method called Pooling. Essentially we have a 2x2 filter looking through the 31x31 array of data, the filter goes through the input with a stride of 1. However, instead of finding the max or average data from the 4 values seen by the filter, we use the data to create triangles.
- With the 4 values viewed by the filter, we can create two triangles to be written into the .stl file of the 3D image. The X and Y of the points in the triangles are predetermined depending on where it is on the grid, however the Z will be determined by the values scanned by the ultrasonic sensor.

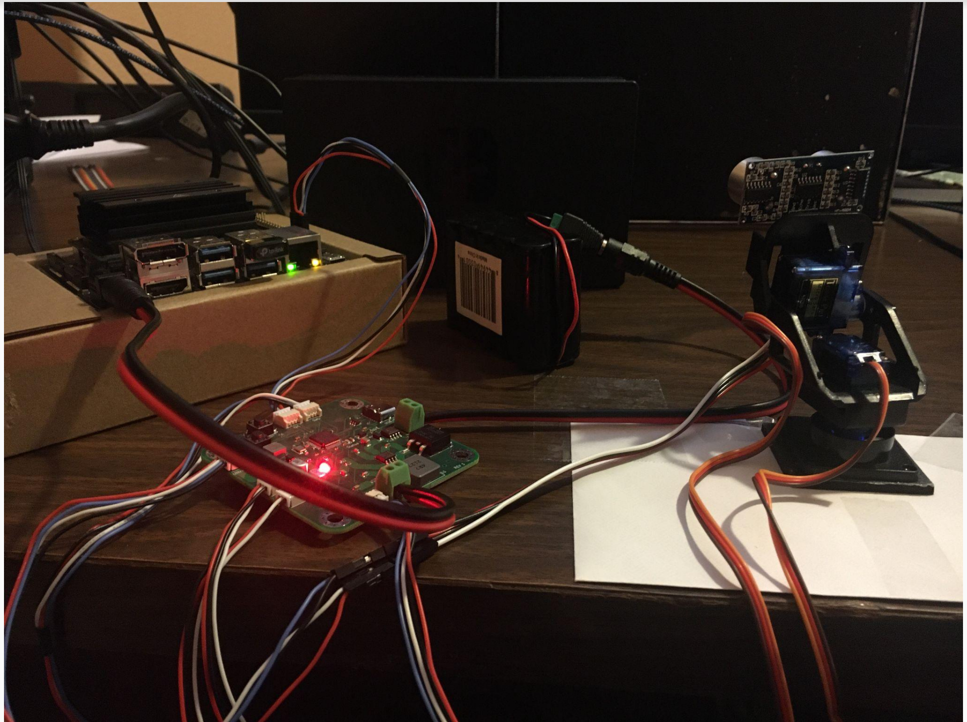
Software Flowchart



Programming Languages

- The Microcontroller
  - Uses C Language
  - A small but very significant part of the programming
  - Simple algorithms used
  - Programmed through the Arduino IDE
- The Jetson Nano
  - Uses Python Language
  - Consists of a majority of the programming written
  - More Complex algorithms used
  - Programmed through JupyterLab

Prototype Design



Ultrasonic Subsystem

- Consists of an Ultrasonic sensors and a microcontroller.
- The HC-SR04 sensors run on 5V logic while the STM32F1 microcontroller runs on 3.3V.
- The sensor will not be mounted on the PCB, rather it would be mounted to a servo kit, which then would be mounted to a chassis.

Ultrasonic Sensor

- The HC-SR04 ultrasonic sensor has a narrow beam
  - Its FOV is 15 degrees
- We selected the HC-SR04 because it was low cost, light weight, and small
  - We kept in mind the portability of the device when selecting

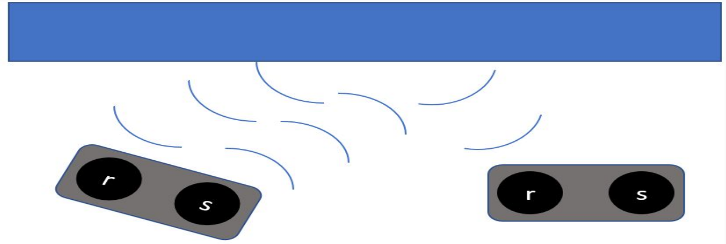
Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

Ultrasonic Subsystem

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	Square	Rectangle	Rectangle
Current Draw	10 to 40mA	15mA	20mA
Pins	5	4	9
Operational Range	2cm - 450cm	2cm - 400cm	2cm - 800cm
Precision	~ 3mm	~ 2mm	~ 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

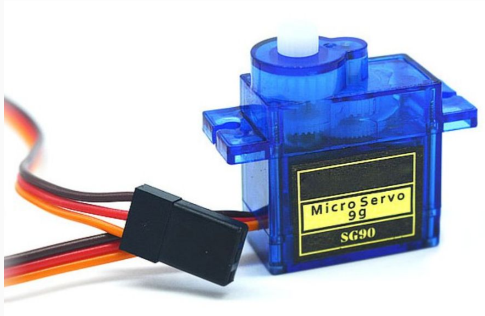
Multiple Ultrasonic Sensors

- One of our stretch goals was to have three ultrasonic sensors on the sensor, rather than just one.
- With multiple ultrasonic sensors, we would be able to create a beam array.
    - When tilting the ultrasonic sensor in one direction a lot, it would be best if we had other ultrasonic sensors catching the sound waves the first ultrasonic sensor sends
    - This is due to the fact that the other ultrasonic sensor would be closer to the beam's path in order to capture the distance more accurately.



Servos

- The two servos we selected were the SG90 Micro Servos
- We selected these servos for many reasons
  - They were light weight, making these great for portability.
  - They were low in price, which was great for our low cost 3D scanner
  - They came with a Pan-Tilt Kit which is exactly what we needed to move our ultrasonic sensor at certain angles

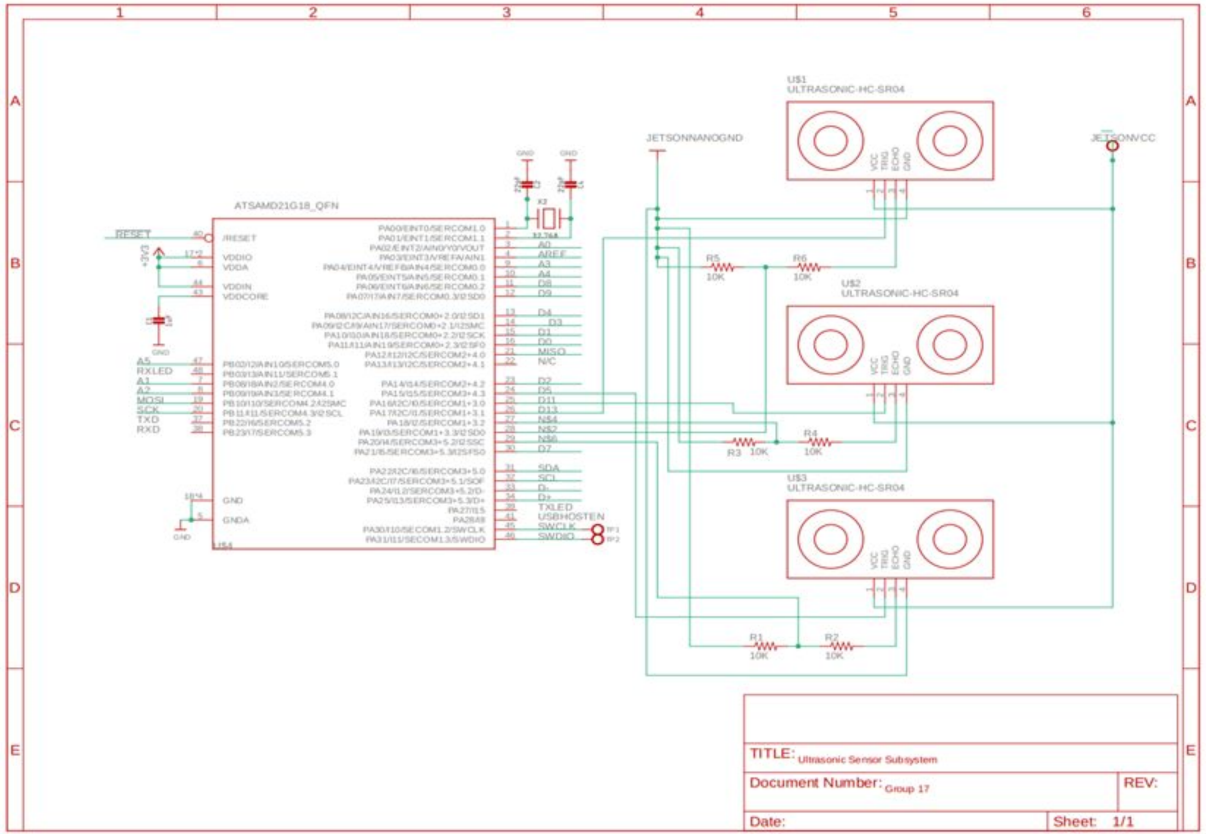


PCB Schematic Design

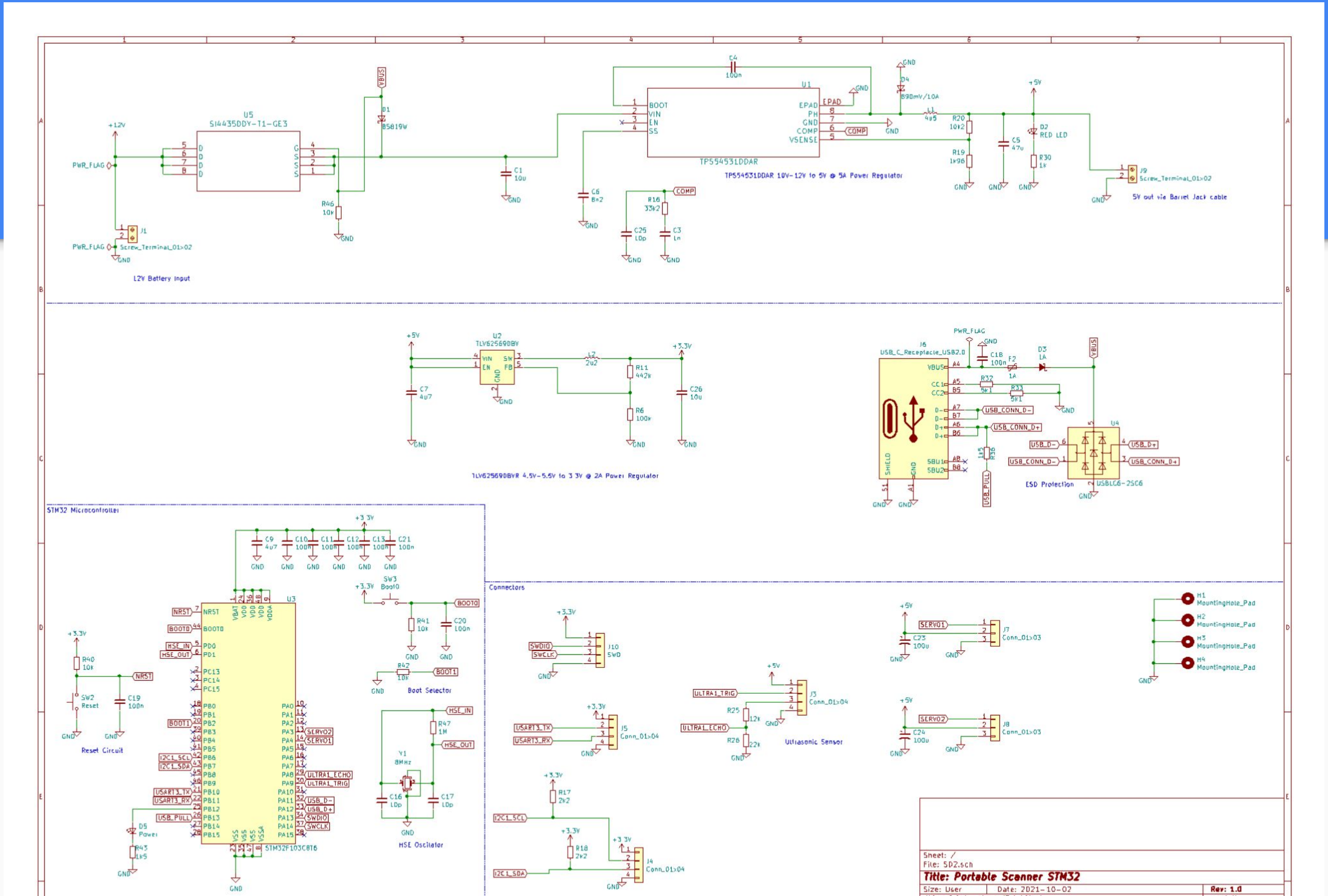
- Texas Instruments TPS56637 Buck convertor will regulate the 12V NiMH Battery to 5V logic and feed into the Jetson Nano female barrel jack port.
- The STM32F103C8T6 will then be powered with 3.3V after 5V goes into the 5V to 3.3V buck converter.
- STM32F103 supports only USB A, while USB C is needed.
- A 4 pin header on the PCB with two 3 pin headers will allow us to connect the sensor and allow the servos to move.

Component	Operating Voltage	Operating Current	Total Power
NVIDIA Jetson Nano	5V	2.5A	12.5W
IMX219-83 Stereo	1.8V	1.23mA	0.0022W
STM32F103C 8T6	3.3V	1.4uA	0.00000462W
HC-SR04	5V	15mA	0.0075W
		Total:	12.51W

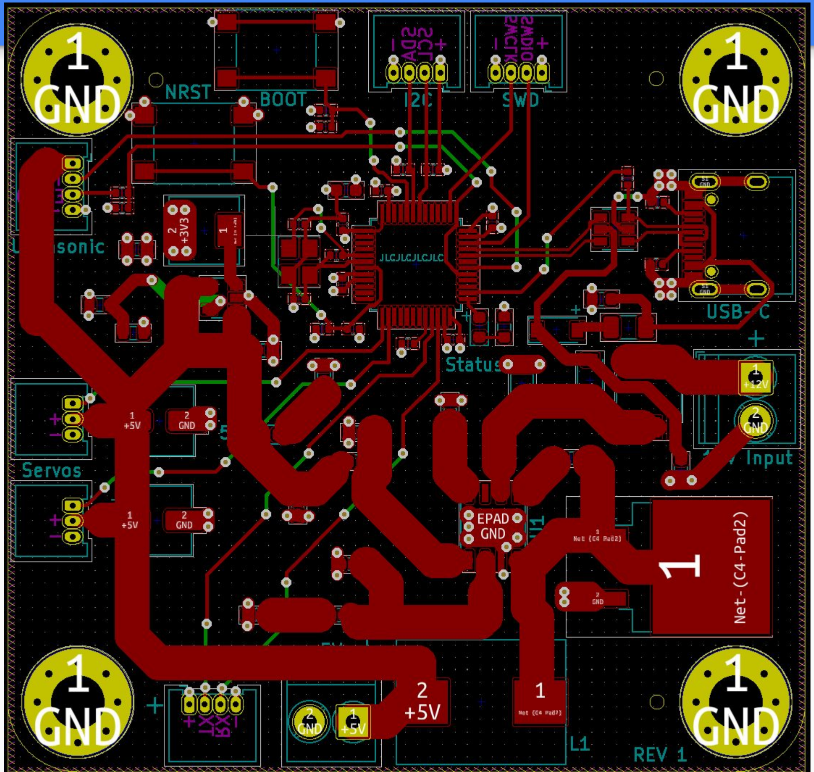
Initial PCB Schematic

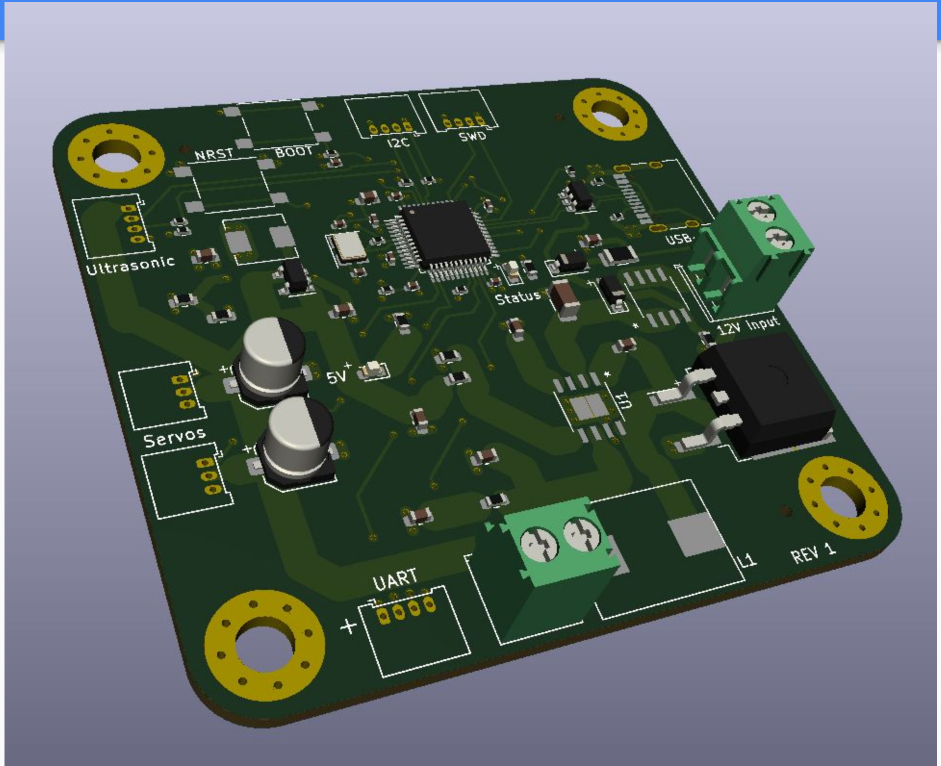


PCB Schematic Revision

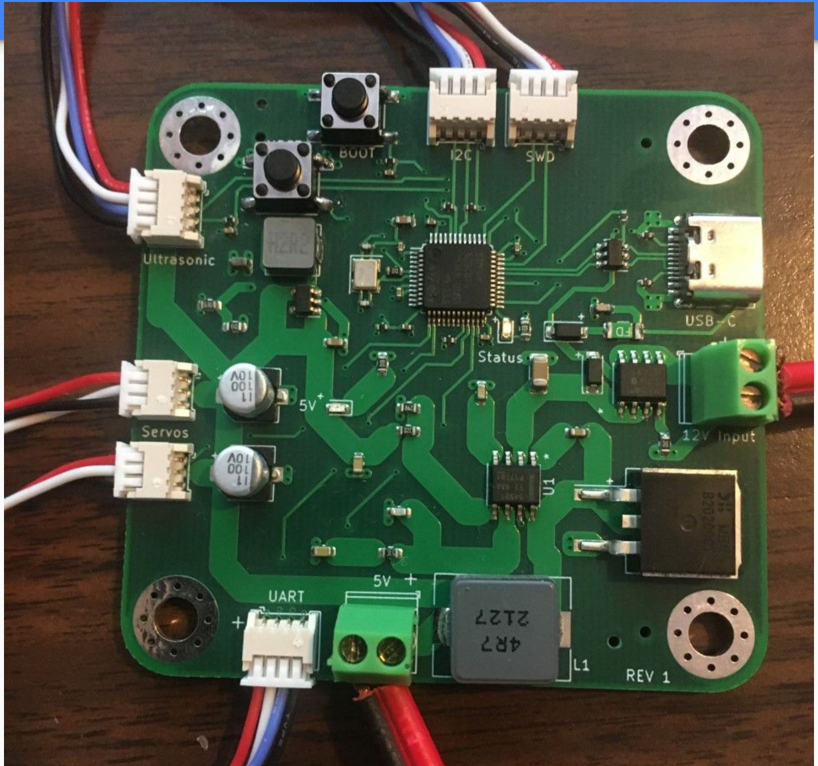


PCB Design


GND
NRST
BOOT
I2C
SWD
Status
USB-C
+12V
Input
Servos
EPAD
GND
Net-(C4-Pad2)
+5V
L1
REV 1


NRST
BOOT
I2C
SWD
Ultrasonic
USB
Status
12V Input
5V
Servos
UART
L1
REV 1

Final PCB

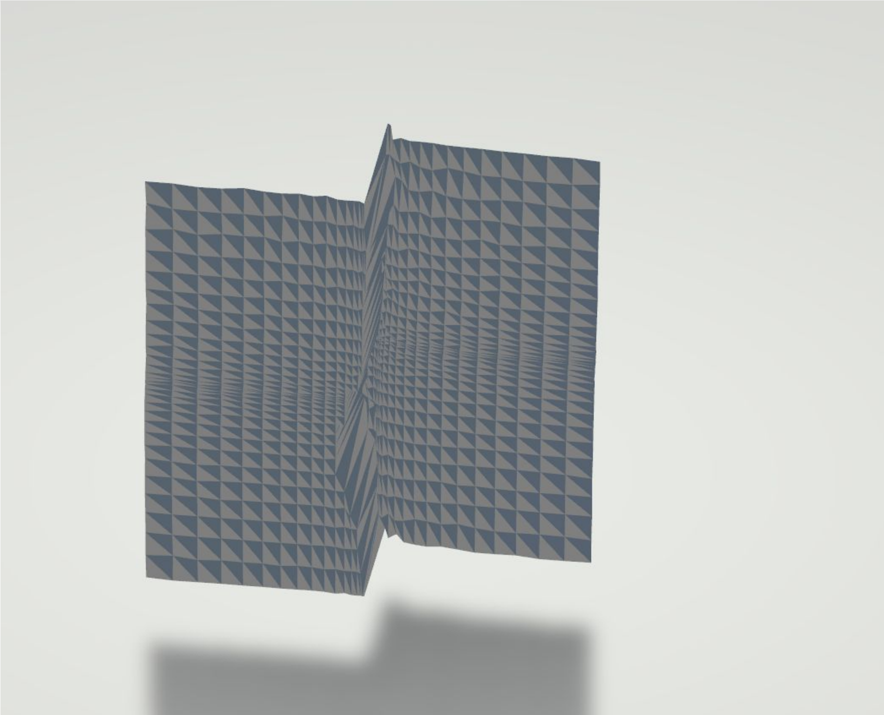


BOOT
I2C
SWD
Ultrasonic
USB-C
Status
5V
Servos
12V Input
U1
UART
5V +
L1
REV 1

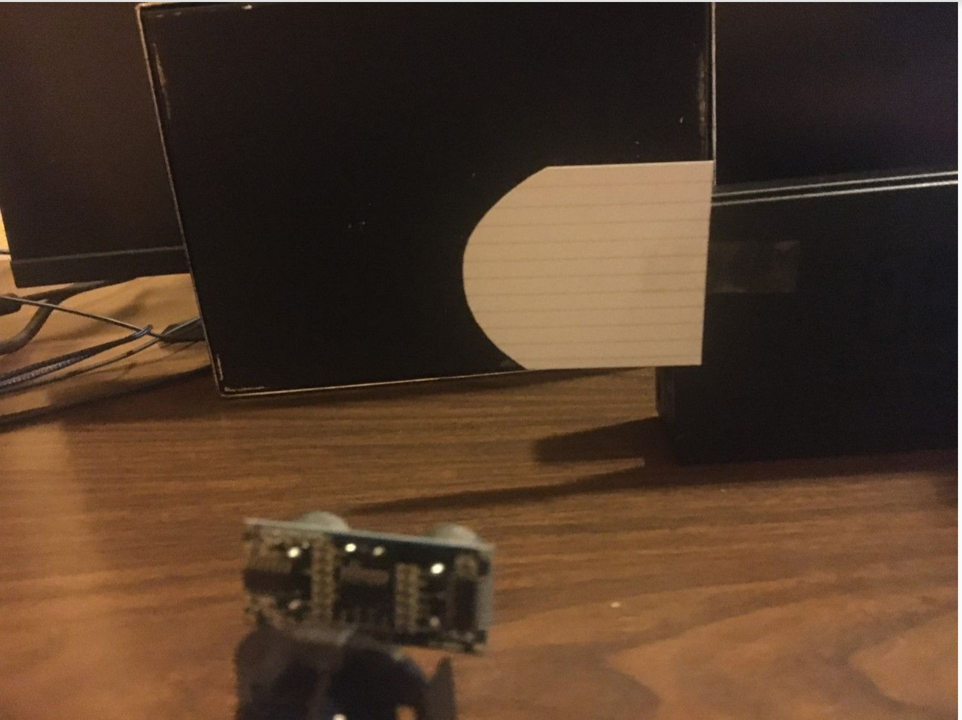
Budget

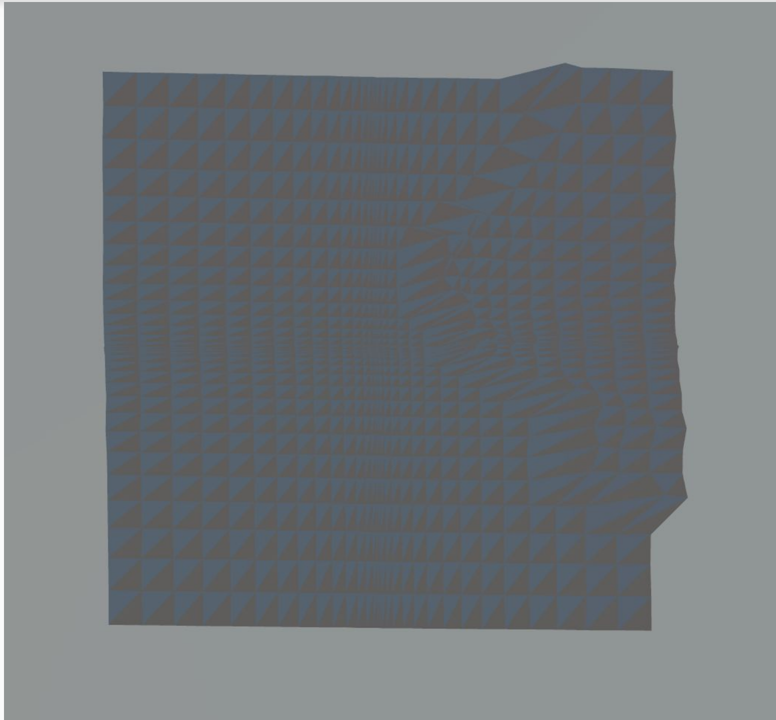
Item	Price
5 PCBs with 2 assembled	$113.49
3pcs Ultrasonic Sensors	$9.99
12V NiMH Battery	$29.95
UART and SWD Programming Adapters	$35.00
Jetson Nano 4GB	$99.99
Pan/Tilt Servo kit	$9.99
Total --	$298.41

3D Scans - Flat Edge, Flat Surface

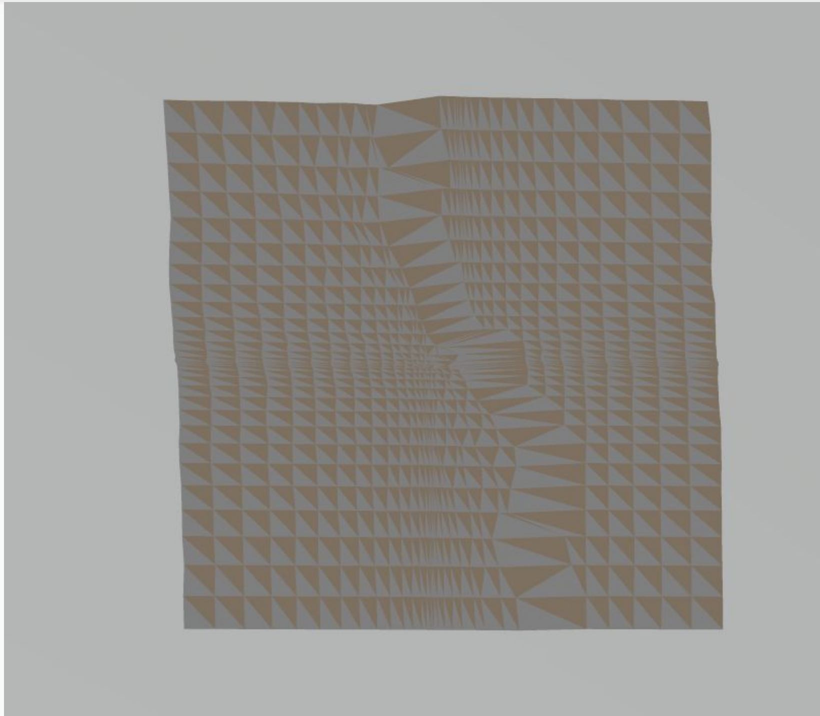


3D Scans - Round Edge, Flat Surface

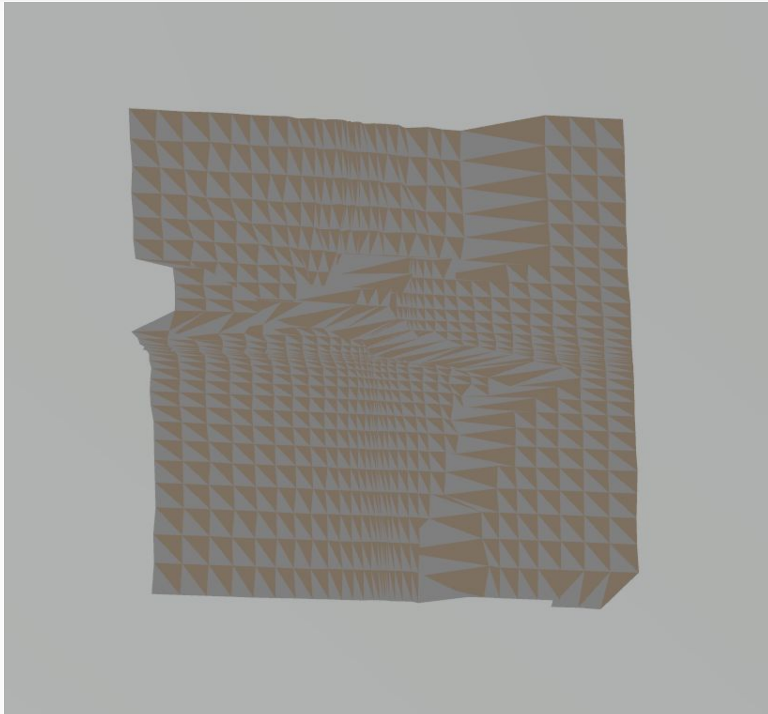




3D Scan - Flat Edge, Round Surface



3D Scan - Round Edge, Round Surface



Testing Results

- The scanner is able to detect edges, whether they be round or flat.
- The scanner is only able to create flat surfaces, even if the object itself is round.
- This makes the scanner good for finding the layout of an area.
- However the ultrasonic sensor was unable to determine the difference between a round surface and a flat one.

Conclusion

- A low cost portable 3D scanner is definitely feasible.
- One thing we would change if we had more time would be to research, program, and test out different sensors.
  - The limiting constraint in our project was the quality of our ultrasonic sensor.
  - With Lidar, Infrared, or Time of Flight sensors, we might have been able to see different results.
- Overall our system worked well with what we have in terms of edge detection.

Our Lessons

- We learned that no matter what, you should work on something every day, no matter how small the progress.
- We learned that communication is key in any work environment, no matter if it is in person or online.
- We learned that when working on something, always prepare for mistakes or errors.