2DX: Microprocessor Systems Final Project

Instructors: Drs. Boursalie, Doyle, and Haddara

Anthony Acosta–acosta3–400201942– 2DX4 –Monday Afternoon) – L01

As a future member of the engineering profession, the student is responsible for performing the required work in an honest manner, without plagiarism and cheating. Submitting this work with my name and student number is a statement and understanding that this work is my own and adheres to the Academic Integrity Policy of McMaster University and the Code of Conduct of the Professional Engineers of Ontario. Submitted by [Anthony Acosta, acosta3, 400201942]

Device Overview: General Description& Block Diagram:

The system is designed as a spatial measurement system using a time-of-flight sensor and a rotary machine. Combining the Time-of-Flight sensor and the stepper motor it provides two degrees of freedom for this application the sensor takes the coordinates of the y-z plane. The x-axis is treated as a fixed axis, and the user must displace the system manually. This is then translated into a visualizer.

Features

This system is designed to recreate a 3D image of its surrounding by using a stepper motor, a lidar, push buttons, and microcontroller. The total project cost under 200 CAD dollars.

- Bus speed is 120Mhz
- Operating voltage 2.97V-3.63V
- C, assembly language and python
- Baud Rate: 115200 bits
- Nominal voltage is 3.3V
- 2 X 12-bit SAR ADC modules
- Communication protocol: I2C
- Memory:
    - 1024Kb Flash memory
    - 256KB SRAM
    - 6KB EEPROM and internal

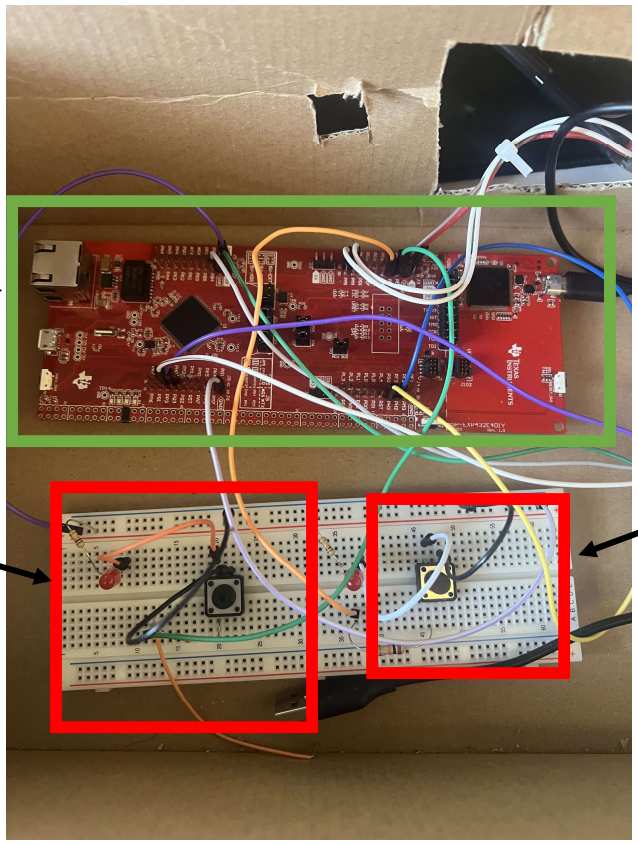
The 3415-POLOLU can measure up to 4 meters and its operating voltages is 2.6V to 5.5V.

General Description

Figure 1: Picture of the entire System

MPS-EXP432E401Y:

The main control source. It is used to communicate with the ToF and stepper motor.



Push button: To start rotating clockwise and capture data

Push button: To start rotating counter clockwise and capture data.

Figure 2: Control System

MOT-28BYJ48: Is to rotate the Time of Flight Sensor. It stops for every 45 degrees till one full rotation is completed. It helps The ToF sensor gather data.

Figure 3: Stepper Motor

3415-POLOLU: A lidar is used to measure the distance in the y-z plane.

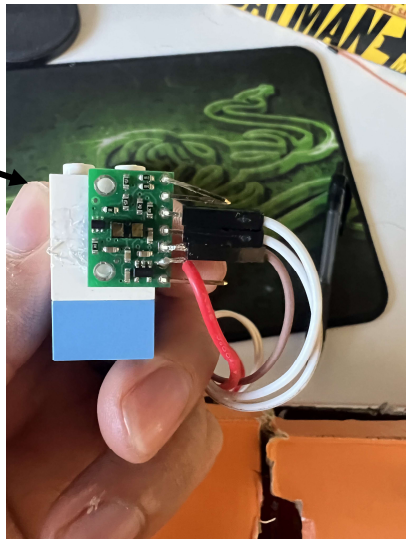
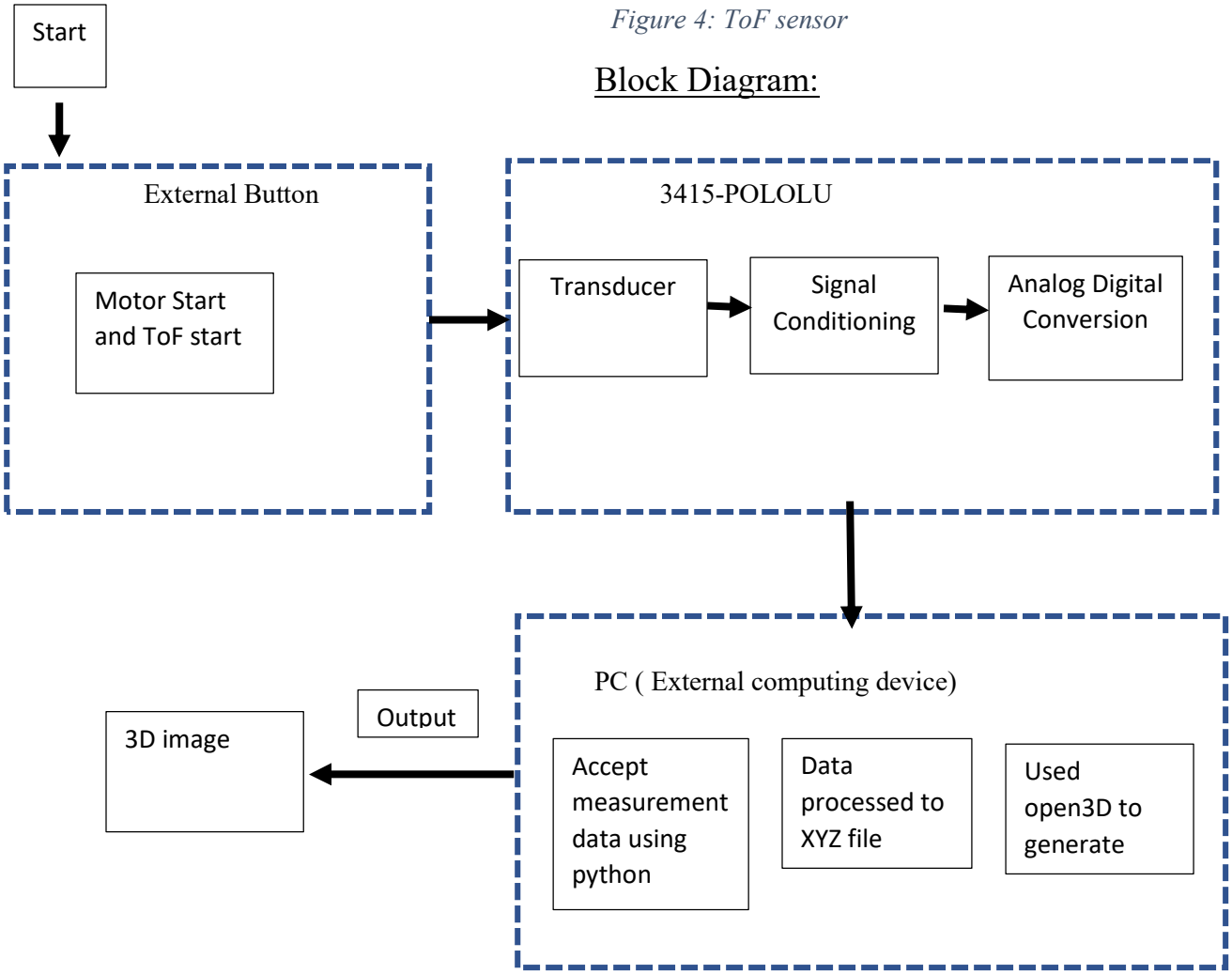


Figure 4: ToF sensor

Block Diagram:



The Time-of-Flight sensors is a transducer that detects distance. Once it has detected values it is then conditioned to match the design specifications of the ADC. After it is conditioned, it is transmitted to the ADC. Once this is complete it is sent to the ADC stored as digital format. This Digital data can now be used to generate a 3D image of a space.

Device Characteristic Table:

Stepper Motor Pins	PH0,PH1,PH2,PH3
ToF pins	PB2, PB3
External push buttons	PE01, PE1
Bus speed	30 MHz
Communication protocols	I2C and UART
Communications speed	115200 bits per second
Communication port	COM3
Python Libraries	NumPy, open3D, pyserial

Detailed description:

The 3415-POLOLU VL53L0x is a lidar that uses infrared light. It can measure the distance by estimating the time it takes for the light to reflect to the sensor. This measurement is described by the equation below.

$$Distance\ (mm) = \frac{Laser\ \ Travel\ \ Time * Speed\ of Light}{2}$$

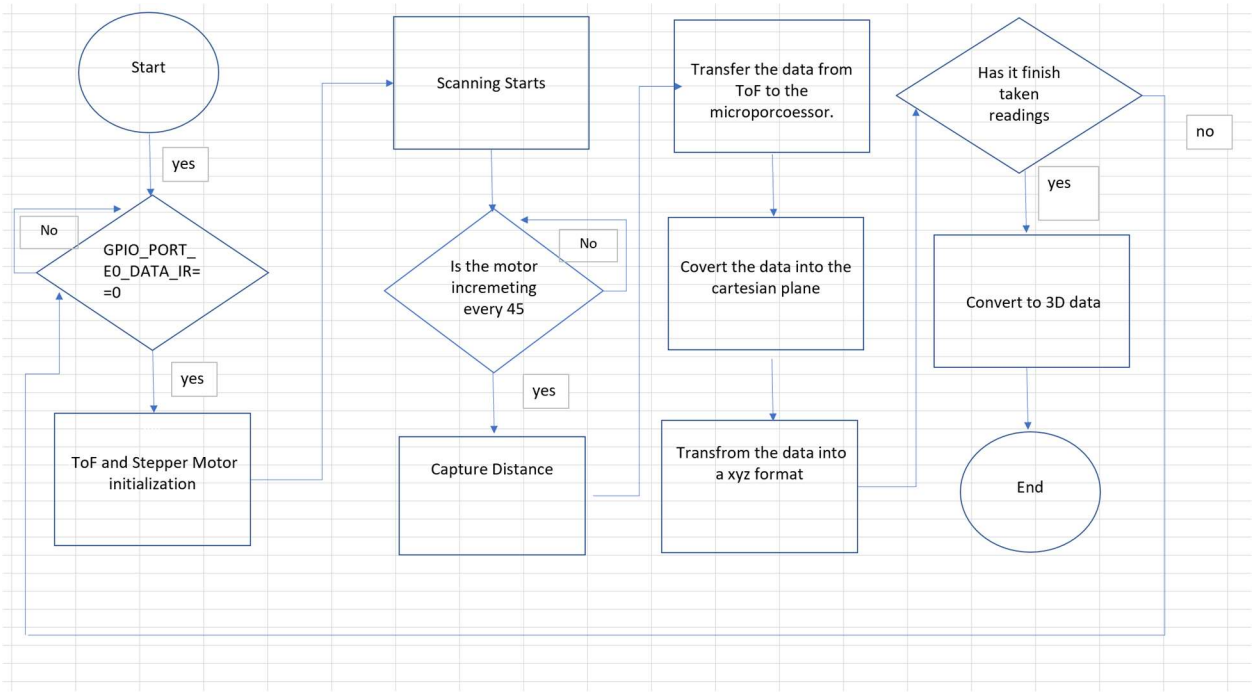
The steps for data acquisition:

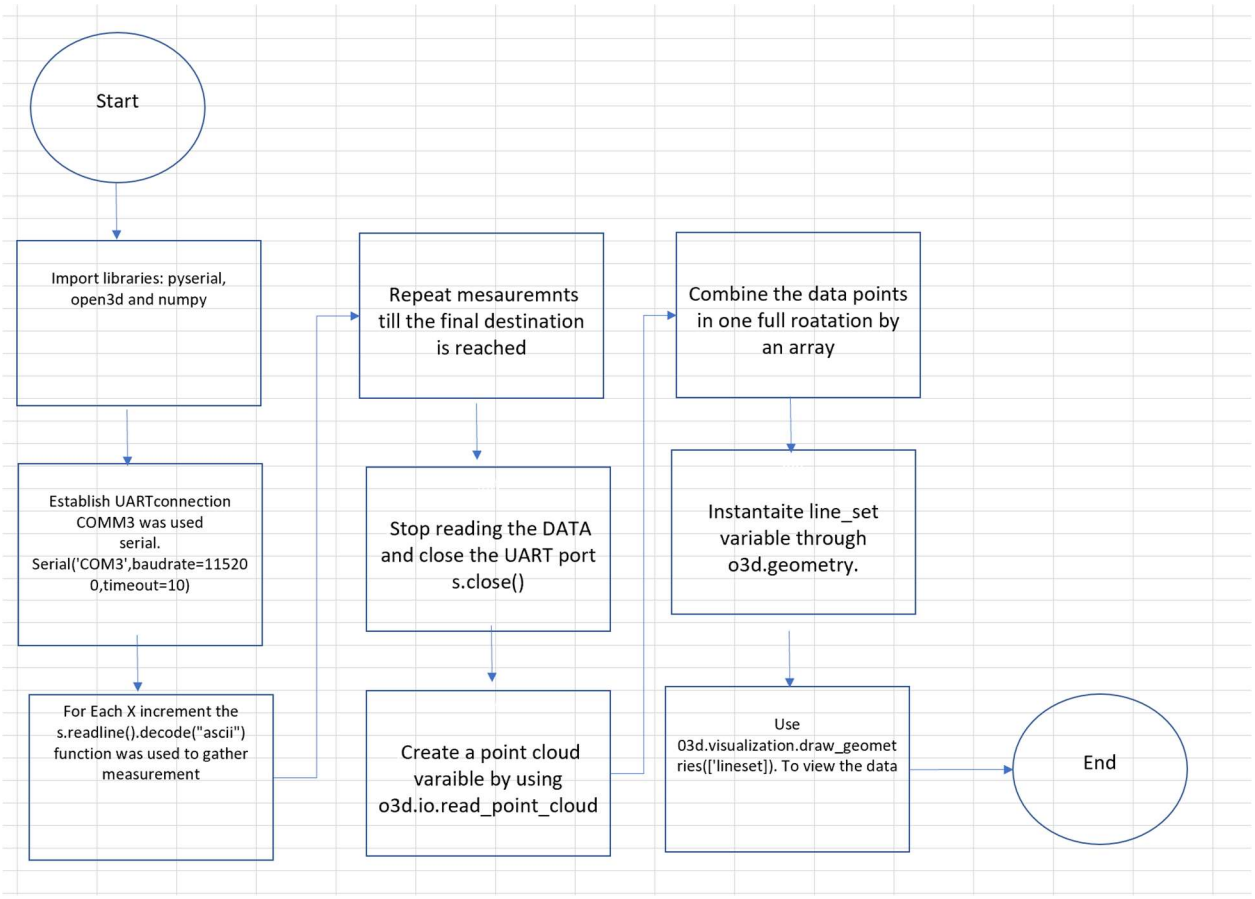
1. Digital input: Two external buttons are used and its set to active high. When the first button is pressed the lidar and motor would start to capture data and it move one full rotation clockwise. When the second button is pressed it would move counter clockwise to detangle the wire. The ports that were used to setup the buttons are port E0 and E1.
2. Data acquisition: This step is all done throughout the Time-of-Flight sensor where the sensor behaves as a transducer, signal conditioning and digital conversion.
3. Data transmission: Once the data is collected by the Time-of-Flight sensor this data is transferred to the microcontroller MSP-EXP432E401Y through I2C at a rate of 100 Kbps. This then transferred to the PC via UART at a baud rate of 115200 bits/second.
4. Data conversion: The data is converted by the MSP-EXP432401y by using basic trigonometry this is described by the formula below. Once it is converted in the cartesian plane it is the process by python to generate a 3D image of the space.

$$x = manual\ dipalcement\ (increased\ by\ 30\ mm\ each\ full\ rotatiuon)$$

$$y = distance * \cos(\alpha)$$
$$z = distance * \sin(\beta)$$
$$Example:$$
$$Distance = 300\ mm\ , Angle = 45,\ \text{x=0mm}$$
$$x\ = manual\ dipalcement\ (increased\ by\ 30\ mm\ each\ full\ rotatiuon)$$
$$y = 300 * \cos(45)$$
$$z = 300 * \sin(45)$$

Note: the math. H file in C uses radians so the angle is converted in radians.





Visualization

The computer that was used is a Microsoft Surface 2 with an operating system of Windows 10 and Python Version 3.9.12.

Computer Specs

- Intel® Core™ i7-8650U
- Nividia GeForce GTX 1060
- Intel® UHD Graphics 620
- 512 SSD
- Two USB A and one USB C

Data visualization program

- Data is received by UART from the microcontroller it is then transported to python to do data processing
- Data storage and processing: To read data pyserial library was used and the port and baud rate is inputted with the *function serial. Serial('COM3',baudrate=115200,timeout=10)* to allow for UART communication. This then converted to the cartesian plane and stored as xyz file format

for open 3D to process. Python was used to process data specifically open 3D. The *o3d.read_point_cloud and o3d.geometry.LineSet* was used to read and visualize the program.

- Data visualization: Open 3D is used to generate a 3D image of the captured data. It is generated by line sets and points which are connected.

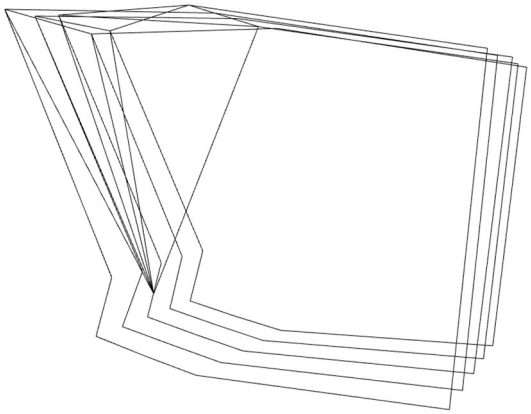


Figure 5 Open3d image

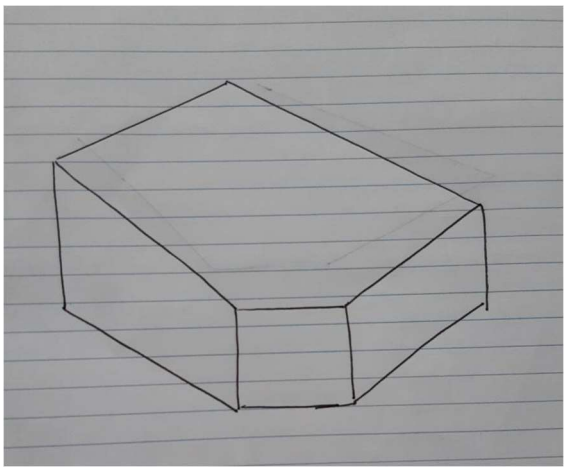


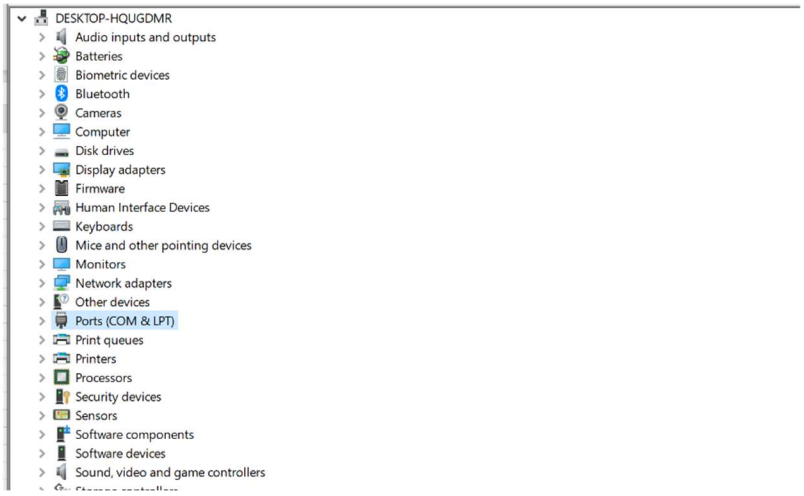
Figure 6 Isometric view

Application Example

1. Install Python any version between 3.7-3.9 is sufficient for this case Python version 3.9.12 is used
2. In the command prompt install the following libraries:
    a. pip install pyserial
    b. pip install numpy
    c. pipi install open3d
3. Check if all the libraries are installed: by typing *pip list* in the command prompt

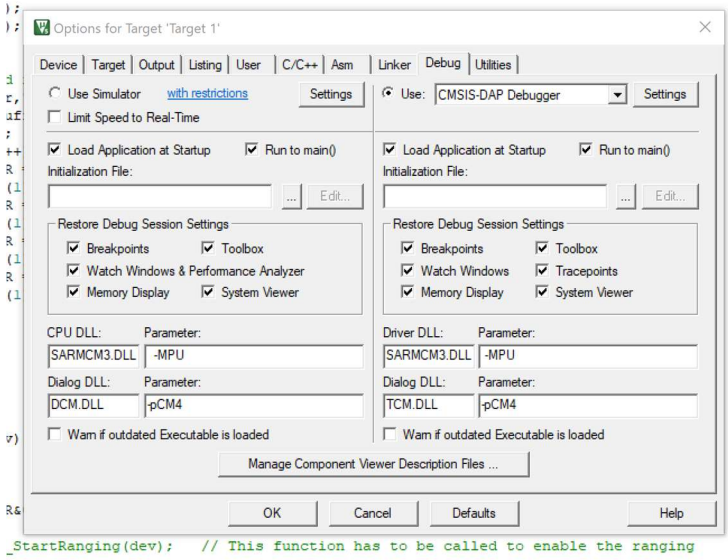
```
C:\Users\antho>pip list
Package                Version
---------------------- ---------
anyio                  3.5.0
argon2-cffi            21.3.0
argon2-cffi-bindings   21.2.0
asttokens              2.0.5
attrs                  21.4.0
Babel                  2.9.1
backcall               0.2.0
beautifulsoup4         4.10.0
bleach                 4.1.0
certifi                2021.10.8
cffi                   1.15.0
charset-normalizer     2.0.12
colorama               0.4.4
debugpy                1.6.0
decorator              5.1.1
defusedxml             0.7.1
deprecation            2.1.0
entrypoints            0.4
executing              0.8.3
idna                   3.3
ipykernel              6.11.0
ipython                8.2.0
ipython-genutils       0.2.0
ipywidgets             7.7.0
```

4. To establish the UART communication between the computer and the microcontroller these steps must be followed.
    a. Check which ports the UART communication is established this can be done by looking at device manager and search for port.



b. Once this is completed open the python script and edit the *function serial. Serial('COM3',baudrate=115200,timeout=10)* to your specifications. The baud rate is remained untouched as this depends on the microcontroller. The COM depends on what you PC has designated as the UART port.

5. Next setup the hardware as shown in the figure above this would require a two-switch button, the TOF sensor and the microcontroller.
6. To get started flash the C code to the microcontroller. Keil was used to flash the code make sure the same debugger is used to flash the program.



7. Run the python script and start placing the device where the starting position is.
8. Use the button that is designated to rotate Clockwise LED 4 would indicate each time it successfully scans (YZ) plane. *The distance is calculated by using trigonometry as stated in the report above.* Wait until it has come to a full rotation.
9. Walk about 30cm and pressed the button that is designated to rotate counter clockwise to detangle the wire. Repeat step 8-9 till finish capturing.
10. Once finished hit the reset button.
11. Open 3D would visualize the data set and generate the image.

```
0.000000 2002.000000 0.000000
0.000000 1901.459668 1899.946087
0.000000 1.660341 2084.999339
0.000000 -449.888668 450.964729
0.000000 -480.999390 0.766066
0.000000 -505.878353 -503.868130
0.000000 -0.979481 -409.998830
0.000000 1860.846591 -1871.248504
300.000000 1995.000000 0.000000
300.000000 1894.385785 1892.877835
300.000000 1.657952 2081.999340
300.000000 -452.713714 453.796532
300.000000 -481.999389 0.767659
300.000000 -510.837945 -508.808013
300.000000 -0.881533 -368.998947
300.000000 1850.269593 -1860.612381
600.000000 2000.000000 0.000000
600.000000 1914.900045 1913.375765
600.000000 1.654767 2077.999341
600.000000 -454.832499 455.920385
600.000000 -481.999389 0.767659
600.000000 -503.044300 -501.045339
600.000000 -1.020094 -426.998782
600.000000 1898.923786 -1909.538545
900.000000 1981.000000 0.000000
900.000000 1900.044891 1898.532436
900.000000 1.650785 2072.999343
900.000000 -456.951284 458.044238
900.000000 -481.999389 0.767659
900.000000 -502.335787 -500.339641
900.000000 -1.053539 -440.998742
900.000000 1779.051137 -1788.995822
1200.000000 2006.000000 0.000000
1200.000000 1903.581832 1902.066562
1200.000000 1.649989 2071.999343
1200.000000 -454.832499 455.920385
1200.000000 -482.999387 0.769251
1200.000000 -502.335787 -500.339641
1200.000000 -1.041594 -435.998756
1200.000000 1840.397728 -1850.685334
```

C:\WINDOWS\py.exe

```
0.000000 2002.000000 0.000000

0.000000 1901.459668 1899.946087

0.000000 1.660341 2084.999339

0.000000 -449.888668 450.964729

0.000000 -480.999390 0.766066

0.000000 -505.878353 -503.868130

0.000000 -0.979481 -409.998830

0.000000 1860.846591 -1871.248504

300.000000 1995.000000 0.000000

300.000000 1894.385785 1892.877835

300.000000 1.657952 2081.999340

300.000000 -452.713714 453.796532

300.000000 -481.999389 0.767659

300.000000 -510.837945 -508.808013

300.000000 -0.881533 -368.998947
```

Limitations:

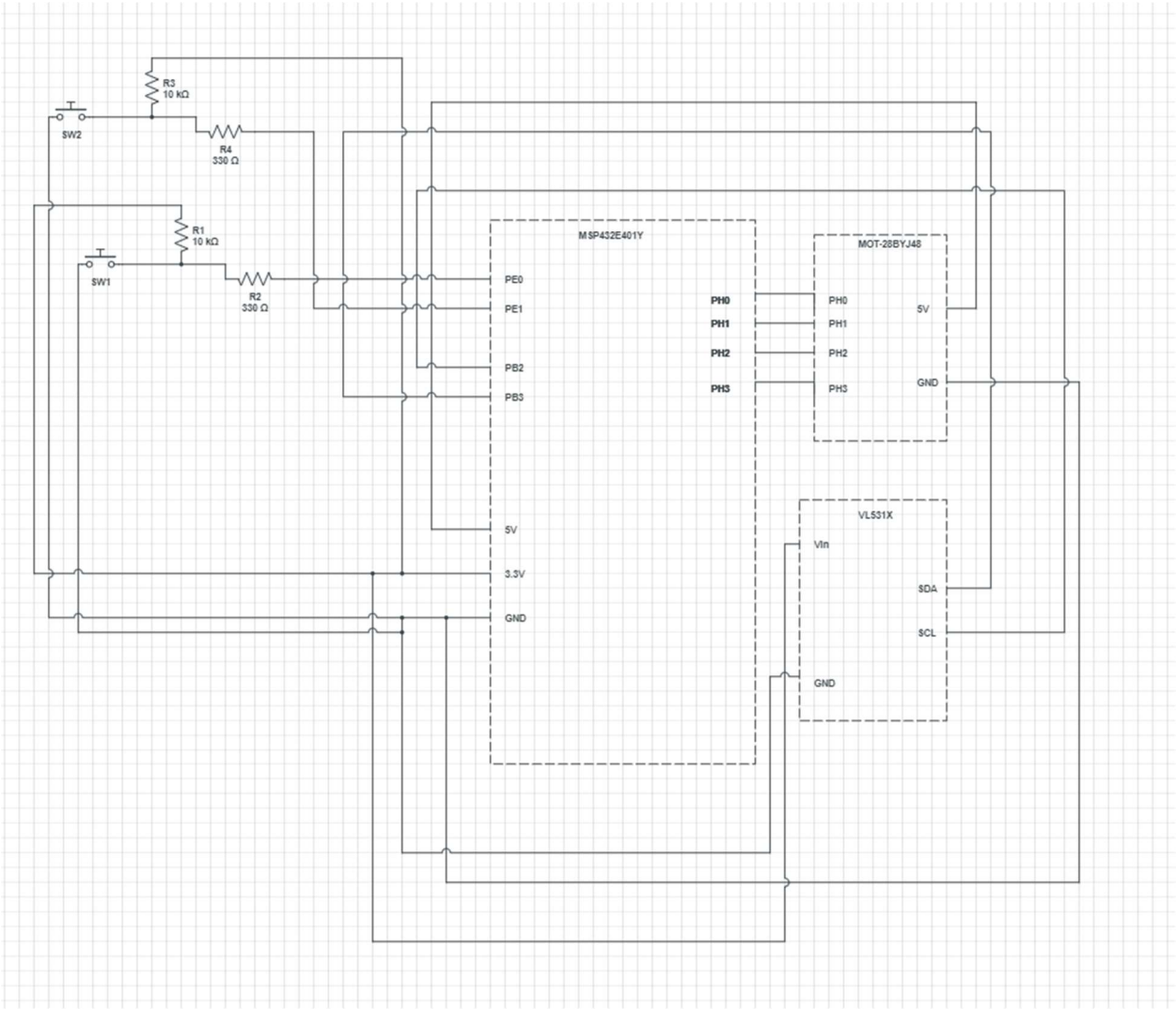
1. The microcontroller is limited to 32 bits hence this limits the precision of the values. The trigonometry function is used using the C library. It accepts a parameter of double and the units are in radians. Due to the fact radians are required we must estimate the angle hence the value won't be precise. In addition to this the trigonometric functions used a double precision parameter whereas the microprocessor only handles 32-bit operations. This causes a longer processing time.
2. $Max\ quantization\ error = \frac{Max\ Distance}{2^n}$

$$6(10^{-5})m = \frac{4\ meters}{2^{16}}$$

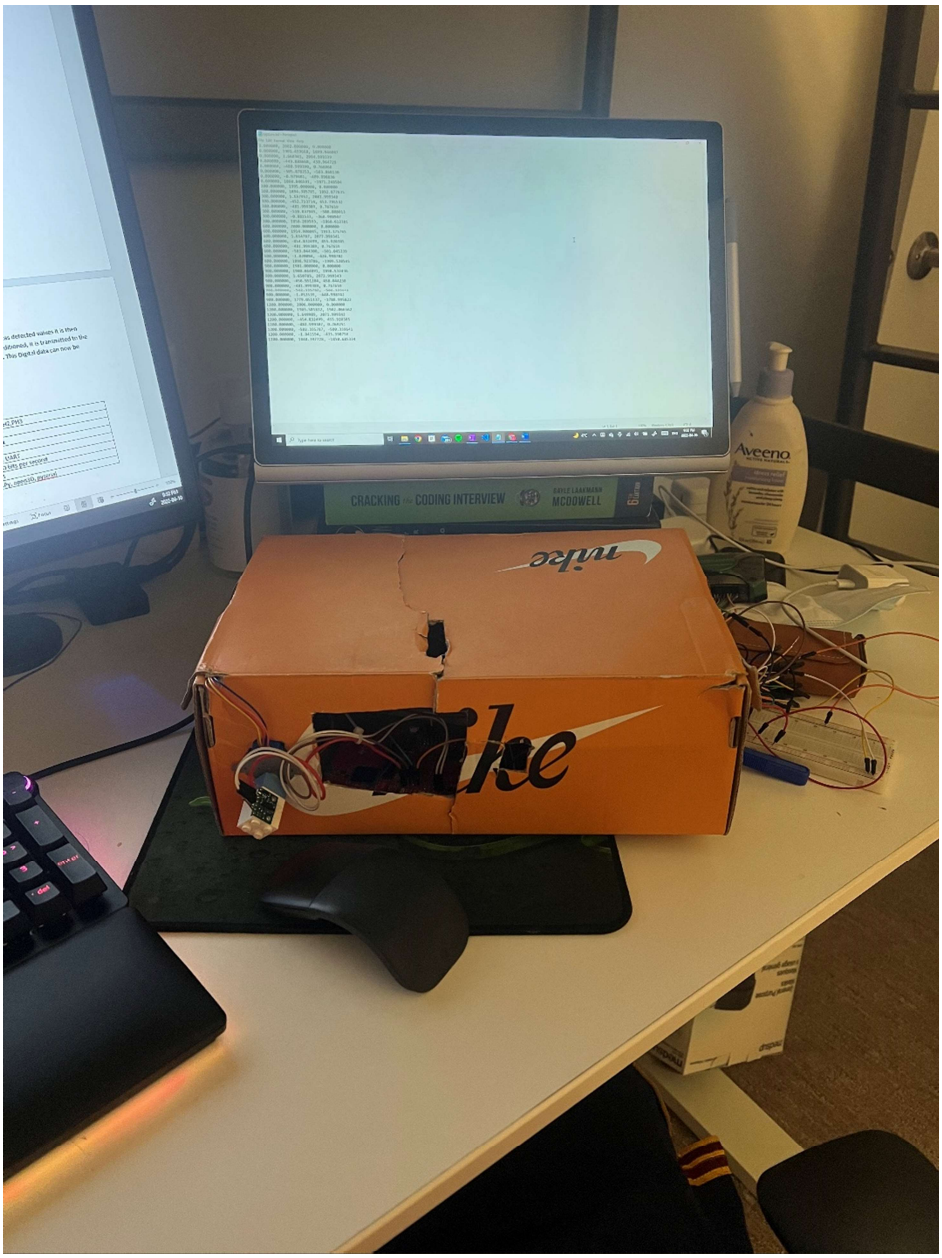
.

3. The max serial communication rate is 115200 bits per second. This was tested by gradually increasing the baud rate until a transmission error occurs.
4. The communication method is serial communication and uses the I2C protocol at 100kbps.
5. The hindering factor of the system is the ToF sensor as the maximum reading speed is cap at 50Hz. This was tested by manually by increasing both the motor and the sensor. It is observed that the motor is turning too fast for the sensor to capture data.

Circuit Diagram



Appendix


CRACKING the CODING INTERVIEW
GAYLE LAAKMANN
MCDOWELL
Aveeno