Comp Eng 2DX3
Final Deliverable Report

Dhanvin Tandon

400526680, tandod2

Date: April 6th, 2025

Statement of Originality: I hereby certify that all the work described within this report is the original work of the author. Any published (or unpublished) ideas and/or techniques from the work of others are fully acknowledged in accordance with the standard referencing practices, and I have obtained the necessary permissions from the company to submit this report.

1A. General Description

The 3D LIDAR Scanning System is an embedded system that is built to produce three-dimensional representations of physical environments by utilizing a time-of-flight (ToF) sensor. The system is able to record distances in multiple 360-degree planes of an orthogonal axis and process the data recorded to generate a 3D model of the covered area. The system includes a MSP432E401Y microcontroller, a stepper motor, and a VL53L1X time-of-flight sensor.

The microcontroller controls all system functions except 3D visualization, provides power to other components, and transmits all data captured to an external device through serial communication. The stepper motor allows the system to achieve 360 degrees range of motion on the Y-Z axis so the mounted time-of-flight sensor is able to make distance measurements along a complete vertical plane, and with fixed distance stepping on the X-axis in order to acquire spatial data layer by layer.

The VL53L1X ToF sensor emits pulses of infrared laser light, calculating object distances by the time required for laser pulses to bounce back to a detector. The sensor calculates the distance based on this time information based on its configuration parameters and transmits the data to the microcontroller using I2C communication protocol.

The device operates with only one user input. The first is a "start scanning" switch via PJ0 that enables the device to scan a single slice of the environment and take 32 measurements. After taking the 32 measurements, the stepper motor will rotate 360 degrees in the opposite direction to return back to the original position. The number of horizontal steps is a hard-coded value but can be changed by the user by simply changing one value in both Keil and Python code.

The system communicates with a PC where the given Python programs are executed. The microcontroller transfers data to the PC through UART, which transfers status data and distance values that are derived by using PySerial. Collected data is visualized through a user-developed Python application that converts the data into Cartesian coordinates, utilizing Open3D to visualize the scanned world in 3D.

1B. List Of Features

Core System:

- Microcontroller: TM4C1294NCPDT with ARM Cortex-M4F core, 256KB SRAM, 1MB Flash, and 120MHz operating frequency
- Stepper Motor Control: Full 360° rotational scanning with 2048 steps/revolution (0.175° step resolution)
- Configurable angular resolution (default 11.25° = 64 steps between measurements)

Time-of-Flight Sensing:

- VL53L1X Sensor: ±5mm accuracy up to 4m range using 940nm VCSEL laser
- 60Hz ranging frequency with I2C communication (100kbps)
- Multi-zone scanning capability for enhanced spatial resolution

Data Acquisition & Processing:

- Scanning Protocol: 32 distance measurements per rotation (11.25° angular resolution)
- 11 vertical steps (adjustable) with 1000mm default layer spacing
- Real-Time Data Streaming: UART transmission at 115200 baud rate
- Structured data packets (8 measurements/chunk) for reliable PC communication

Visualization & Control:

- Python Interface: PySerial for UART data collection
- Open3D-based 3D point cloud rendering with XYZ coordinate conversion
- Interactive visualization with adjustable perspective and scaling
- User Controls: PJ0 (Polling): Initiates scanning process & returns home

System Indicators:

- LED Feedback:
    - D1 (PF0): Scan in progress
    - D2 (PF4): Data transmission active
    - D3 (PN1): Data transmission mode
- Configuration Flexibility: Adjustable scan parameters (steps, layers, delay) via Keil constants
- Python script supports customizable visualization parameters

1C. Block Diagram

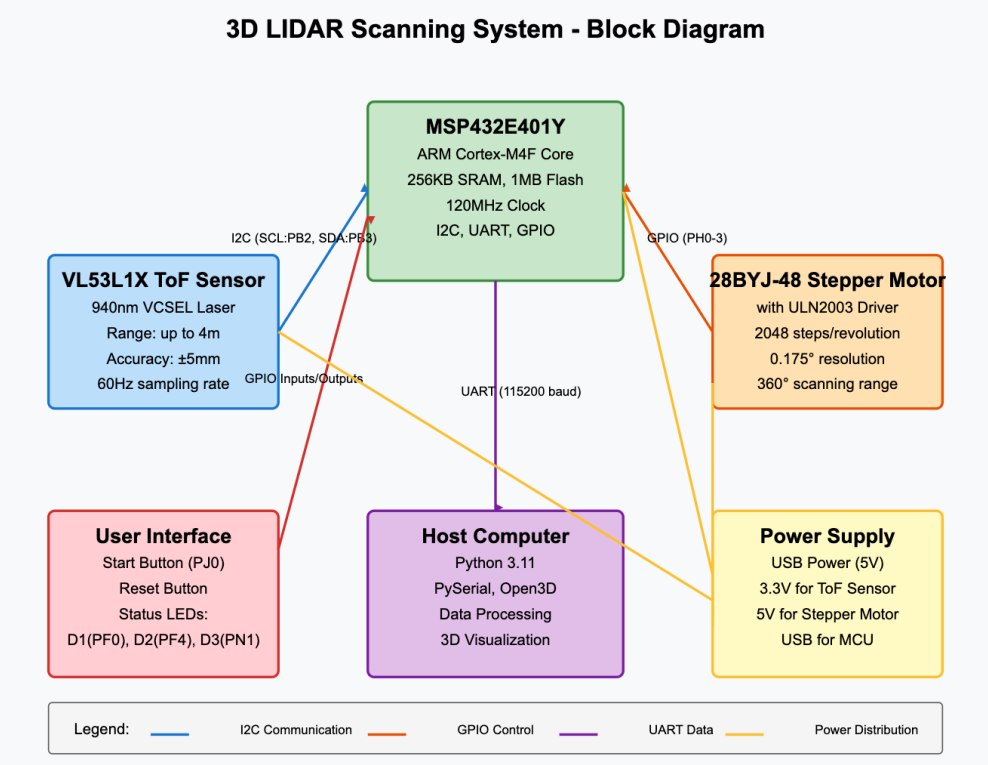

Figure 1 - Overall Block Diagram

2. Device Characteristics Table

Device	Description				
Microcontroller	MSP432E401 Microcontroller				
	Bus Speed		Baud Rate		
	12 MHz		115200 Bd		
ToF Sensor	VL53L1X Time-of-Flight Sensor				
Connections	SCL → PB2	SDA → PB3	VIN → 3V3	GND → GND	
Stepper Motor	28BYJ-48 Stepper Motor and ULN2003 Driver				
Connections	IN1-4 → PH0-3	V+ → 5V	V- → GND		
Host Device	UART		Microcontroller Power		
Connections	COM6		USB Cable		
Other Features	Start Button	Reset Button	Measuring Status LED	Transmitting Status LED	Transmit Mode Status LED
Pins	PJ0	RESET	PF0	PF4	PN1

3A. Distance Measurements

Distance measurements are the responsibility of the VL53L1X Time-of-Flight sensor, taking a measurement at each step as the stepper motor completes a full 360-degree turn. This enables the ToF sensor to measure distances over a complete circular plane.

The VL53L1X ToF sensor operates by transmitting pulses of infrared laser light (940nm VCSEL laser) that reflect from objects in the field of view of the laser. The sensor measures the time it takes for these pulses to return to its detector after striking objects close to it. Distance is measured as follows:

$$Distance = \frac{time \times speed\ of\ light}{2}$$

The sensor computes this timing data based on its configuration and transmits the calculated distance values to the microcontroller via the I2C serial communication protocol. The ToF settings used during system initialization are the default configuration settings provided in the VL53L1 Core API from STMicroelectronics with all functions unchanged from the provided code.

As an initial point of measurement taking, the reset button on the microcontroller must be pushed prior to engaging and booting up all operations on the device. Since the user needs to take measurements, a press on PJ0 will initiate the taking of measurements. While measuring, the stepper motor turns clockwise and pauses after every 11.25 degrees to read a measurement of distance, repeating the same until it has measured to a full 360 degrees. The program takes a total of 32 measurements per cycle. Once the final measurement is taken, the motor returns home automatically by rotating counterclockwise to avoid wire tangling. To future proof the code with a potential future implement of a return-to-home functionality, the code maintains an "angle" counter that is incremented after each motor rotation.

After it has completed one measurement cycle, the system waits for the user to initiate a new measurement using the same PJ0 input. The Time-of-Flight sensor utilizes the synchronous communication mode using I2C with two lines of communication: serial clock line (SCL) that synchronizes the clock and signals the time to begin the measure, and a serial data line (SDA) which sends data in both directions between microcontroller and the ToF.

After collection is finished, the microcontroller transmits the distance information to a PC through UART at a baud rate of 115200 bits per second. All 11 (or any amount set by the user) steps' data is saved in a 2D array, with each group of 32 distances formatted by the microcontroller into four 8-data-point chunks. The Python script (version 3.11) is ready to receive and convert this data from polar to Cartesian coordinates for plotting. The y and z components are calculated using trigonometric functions, where $y = d \times \cos(\theta)$ and $z = d \times \sin(\theta)$, with θ as the rotation angle.

3B. Visualization

3D visualisation of the data is performed by importing the measurements in.xyz file format to Python and treating them as inputs for processing by the Open3D library, which has methods for visualising the data in 3D as a point cloud.

After all the measurements are completed, the Python interface communicates with the microcontroller using the pySerial library. For each cycle, it sends a start signal, and then it receives the four pieces of data and converts them into individual distance values. These values are inserted into a list that is the 3D dataset of 352 points in total (11 steps × 32 angular measurements per step).

The code converts all polar coordinates into Cartesian coordinates using trigonometric relationships. These coordinates are then stored in an.xyz file format which the Open3D library can read. Using the functionalities of Open3D, a point cloud consisting of all points in the visualization is created. This point cloud is also used to create a line set consisting of the points and lines linking them in the visualization. The lines connect all individual points in a slice, with other lines connecting adjacent slices to each other to create a full 3D mesh representation.

4A. Application Note

Device Setup:

1. Components required to set up the device:
    a. MSP432E401Y Microcontroller board
    b. 28NYJ-48 Stepper Motor driver
    c. VL53L1X Time-of-Flight range sensor
    d. ~15 Female-to-female jumper cables for component attachment
    e. Computing device (Windows/macOS/Linux operating system) with Python 3.11 or later
    f. Kiel uVision installed and setup to the MSP432E401Y Microcontroller board
    g. Required Python libraries (pyserial, open3D, numpy)
2. Open Command Prompt as administrator and install required Python packages:
    a. Execute `pip install open3d` and allow it to finish successfully
    b. Execute `pip install pyserial` to add serial communication support
3. Connect the hardware according to the circuit diagram described at the end of the documentation.
4. Plug your computing platform into the microcontroller using the provided USB cable.

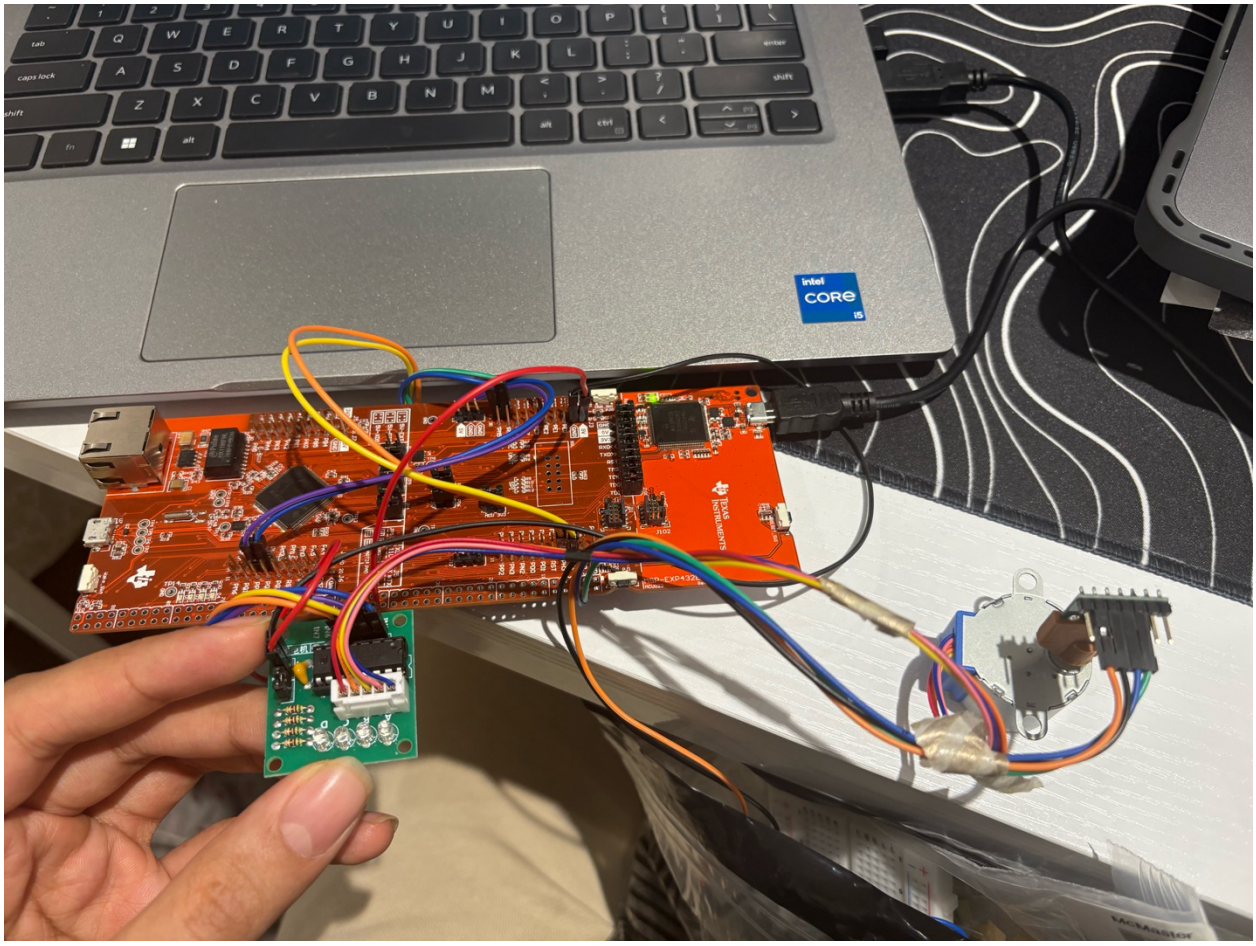

Figure 2 - Physical Built Circuit

Configuration Setup:

1. Open your computer system's Access Device Manager and navigate to the "Ports" tab in order to find the COM port assignment for the microcontroller attached to your system.
2. Open your preferred Python development environment in Python version 3.11and the data-gathering script.
3. Adjust the following script parameters according to your setup requirements:
    a. Alter the COM port designation in the python code to align with the port assignment for the microcontroller
    b. Change the data output file name if needed
    c. Set the incremental movement step size by adjusting the STEPS constant at the top of the file in both the python script and the Kiel uVision script in millimeter units

Operating Procedure:

1. Select an appropriate environment for scanning (considering the sensor effective range limit of 4 meters and reduced accuracy in expansive, open spaces).
2. Position the scanning device on a stable surface that allows linear motion along one axis.
3. Utilize Keil uVision to compile and deploy the firmware to the microcontroller system.
4. Power on the system by pressing the reset button on the microcontroller board.
5. Execute the data collection Python script in your development environment. Inability to see terminal output may be caused by serial communication connectivity problems or driver configurations.
6. When the initialization prompt is received, move the scanning system to your desired scanning position if needed. Keep all cable connections in place during movement.
7. Mount the ToF sensor to the stepper motor using the provided 3d printed attachment.
8. Trigger a scanning cycle by pushing the PJ0 button. Keep the device stationary for the entire scanning process.
9. System status indicator LED D1 will be lit during active scan and turn off upon completion, indicating readiness for the next operations.
10. Once LED D1 is turned off, relocate the device by shifting it one predetermined step distance (about 10 cm) along the horizontal axis.
11. Go back to step 8 and proceed with the scanning sequence until finishing the appropriate number of STEP successive scan cycles.
12. When your measurement sequence has ended, the LED D3 will turn on, indicating data transmission mode.
13. When prompted with "Press Enter to start ToF measurement" in the terminal window, press Enter to start data transfer.
14. Data transmission between computing platform and microcontroller through UART will start. LED D2 will glow while actively transferring data.
15. As soon as LED D2 has gone dark, visualization processing will automatically generate and display a 3D model of data collected.
16. An "Open3D" visualization window will appear displaying the three-dimensional scan data. Navigation of the visualization can be done using standard mouse controls to zoom and rotate.

4B. Expected Output

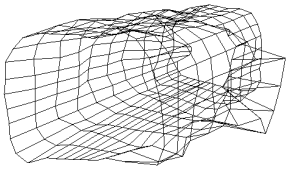

Figure 3 - Open3D Visualization Of Hallway

Figure 4 - Picture Of Hallway

5A. Computational Constraints

The MSP432E401Y microcontroller includes a hardware floating point unit (FPU) to carry out complex floating-point operations and trigonometric functions. However, these tasks require a lot of computing power, making them computationally demanding. To speed things up, the system moves all trigonometric calculations to a Python script running on the host PC instead of the microcontroller. This change greatly improves the speed and efficiency of the entire system.

5B. Sensor Measurement Accuracy

The VL53L1X ToF (Time of Flight) sensor faces some measurement errors because it's digital. This error, called quantization error, happens because the sensor can only measure in set increments. The formula to find the maximum error is:

$$Maximum\ Quantization\ Error\ = \frac{Maximum\ Measurement}{2^n}$$

This sensor has a 16-bit resolution and can measure up to 4 meters (4000mm). With this, the maximum quantization error is:

$$\frac{4000mm}{2^{16}}\ =\ 0.06103mm$$

This number shows the smallest change in distance that the sensor can detect.

5C. Communication Rate Limitations

The system communicates using a serial UART connection, which limits how fast data can be transferred. The current setup uses a baud rate of 115200 bps, but the maximum possible rate for UART is 128000 bps. You can confirm this rate by looking at the PC's device manager port settings, which show the highest speed the connection can handle.

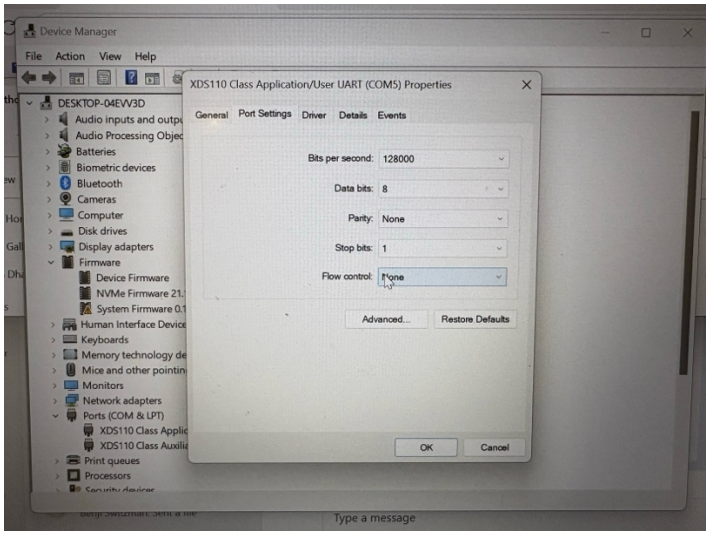

Figure 5 - Communication Rate Limitation

5D. Sensor Interface Constraints

All communication between the ToF sensor and the microcontroller happens through the I2C interface. This protocol limits the data transfer speed to a maximum of 100kbps, which caps how quickly data can be collected from the sensor. Even if the rest of the system is optimized, this speed cannot be exceeded.

5E. System Performance Bottleneck

The stepper motor puts the most significant limitation on system speed. The step is around 3 milliseconds, and it takes 2048 steps for one complete rotation. Thus, one rotation would take around 6.1 seconds. On the other hand, the ToF sensor is considerably faster, taking a single reading every 20 milliseconds and taking 32 readings per rotation—taking a total of around 0.64 seconds. As one can see, this is a big difference, and it is apparent that the stepper motor is the limiting factor in the speed of the system overall.

5E. System Clock Configuration

Setting up the system bus speed involves using the following formula:

$$SYSCLK = (FXTAL/(Q+1)/(N+1)) * (MINT + MFRAC/1024)/(PSYSDIV+1)$$

To achieve a clock speed of 12MHz, we choose N = 24 and MINT = 48, which fits the equation as:

$$12MHz = (25000000/(0+1)/(24+1)) * (48 + 0/1024)/(3+1)$$

This configuration helps ensure that all parts of the system work together at the right pace and stay in sync while keeping compatibility with the communication protocols in use.

6. Circuit Schematic

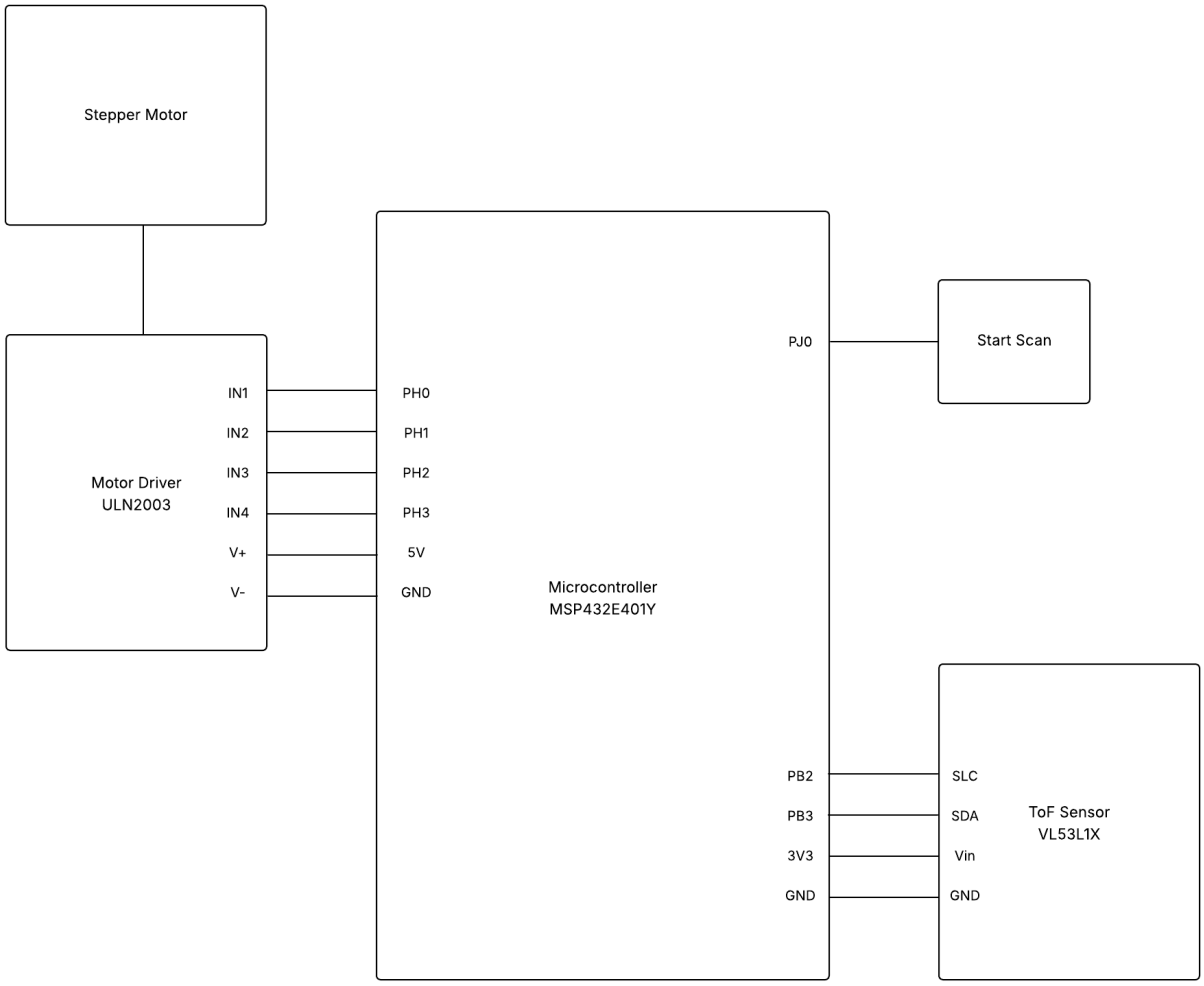

Figure 6 - Circuit Schematic

7. Programming Logic Flowchart

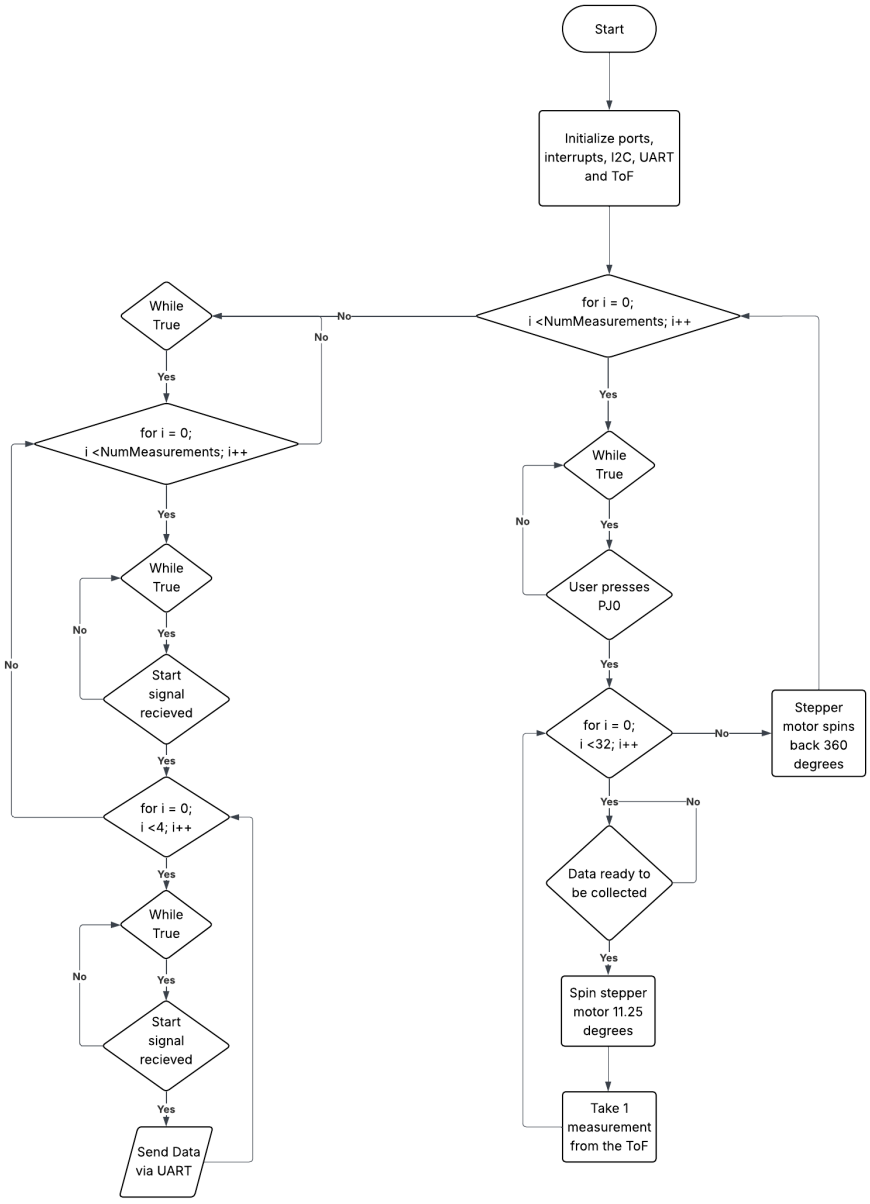

Figure 7 - Kiel Code Flowchart

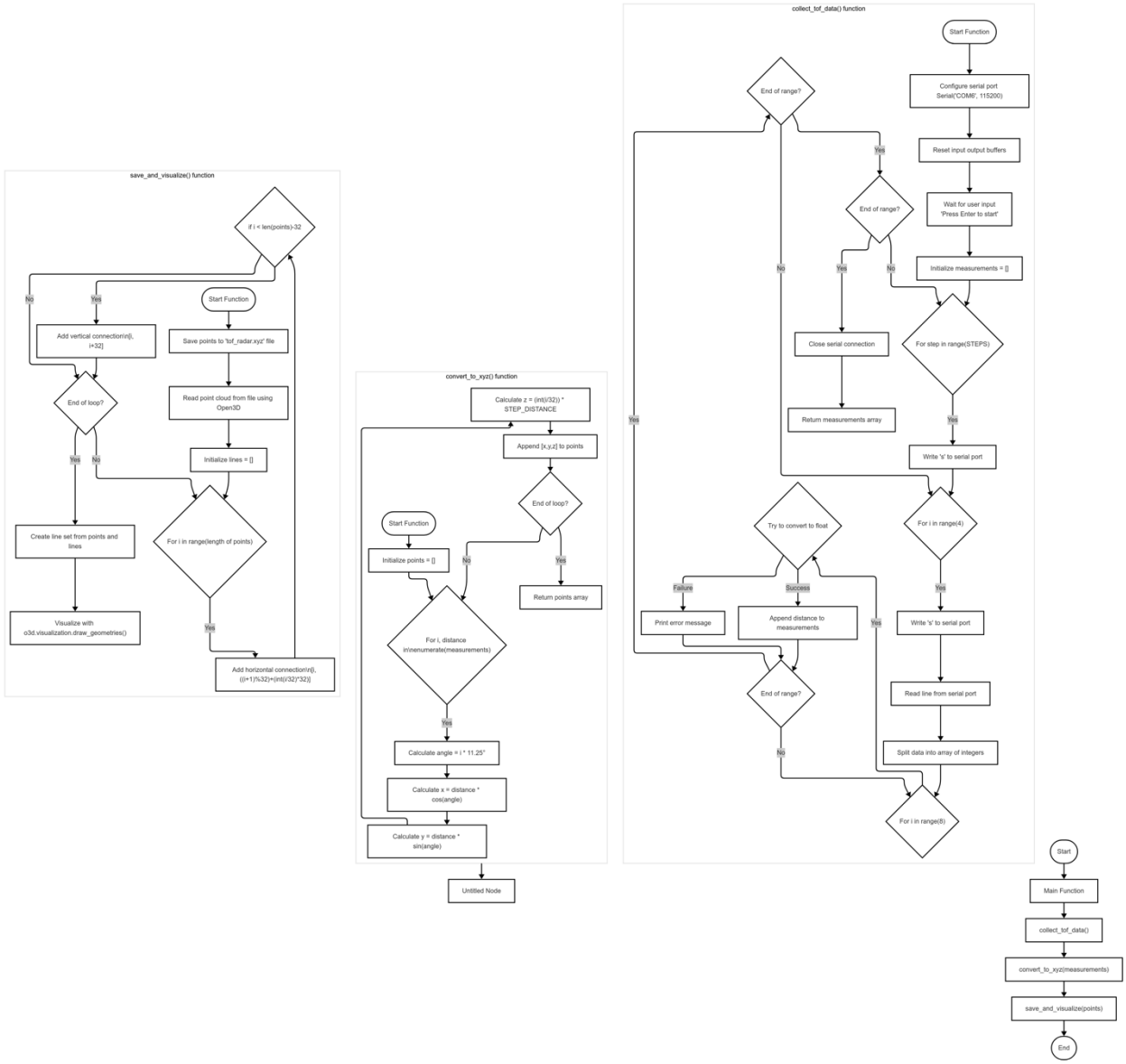

Figure 8 - Python Code Flowchart

8. References

[1] STMicroelectronics, "VL53L1X: Long-distance ranging Time-of-Flight sensor," Datasheet, Rev. 5, Mar. 2023. [Online]. Available: https://www.st.com/resource/en/datasheet/vl53l1x.pdf [Accessed: Apr. 8, 2025].

[2] Texas Instruments, "MSP432E401Y Mixed-Signal Microcontrollers," Datasheet, Rev. D, Jul. 2023. [Online]. Available: https://www.ti.com/lit/ds/symlink/msp432e401y.pdf [Accessed: Apr. 8, 2025].

[3] Texas Instruments, "MSP432E4 SimpleLink Microcontrollers Technical Reference Manual," Reference Manual, Rev. A, Aug. 2023. [Online]. Available: https://www.ti.com/lit/ug/slau723a/slau723a.pdf [Accessed: Apr. 8, 2025].

[4] Computer Engineering 2DX3, Jan. 25 ed. Hamilton, ON, Canada: Department of Electrical and Computer Engineering, McMaster Engineering, 2025. [Accessed: Apr. 8, 2025].