3D LiDAR Scanner Project Report

Device Overview

Features:

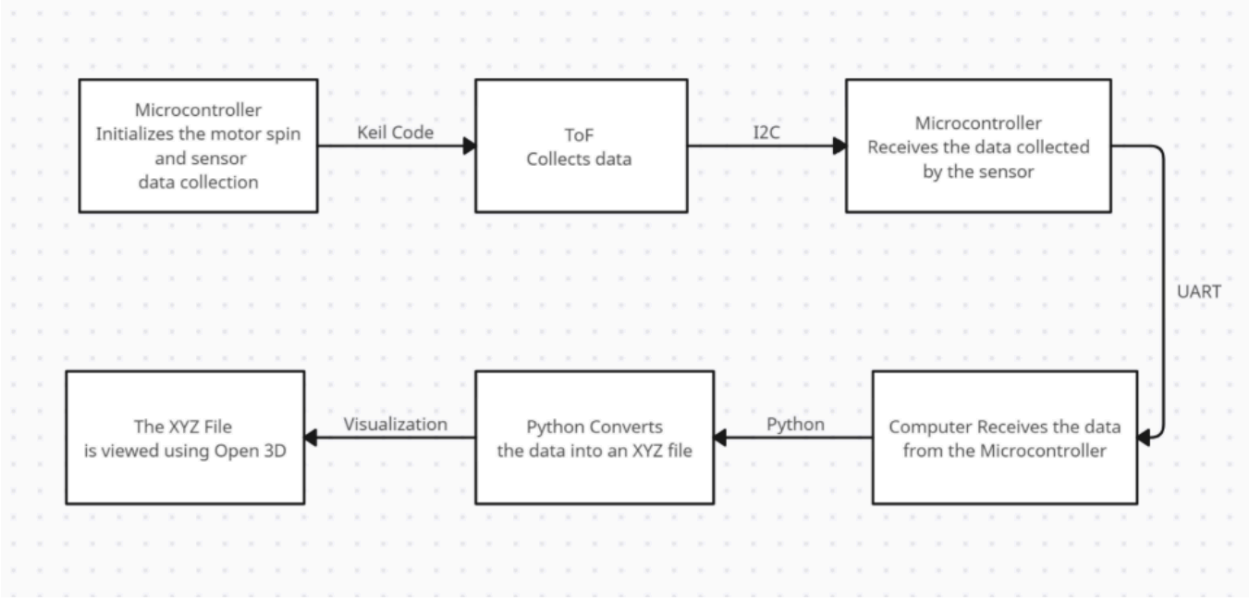
- **Bus Speed**: 20 MHz, set using Phase-Locked Loop (PLL) with PSYSDIV = 23 in firmware configuration.
- **Operating Voltage**:
  - Microcontroller (MSP432E401Y): 3.3V logic, powered via 5V USB
  - VL53L1X ToF Sensor: 2.6–3.5V operating voltage
- **Serial Communication Specs**:
  - **UART0** at 115200 baud (8 data bits, no parity, 1 stop bit) for MCU → PC communication
  - **I2C** at 100 kHz for MCU ↔ VL53L1X ToF Sensor communication
- **Programming Language (PC side)**:
  - Python 3.9 using PySerial (UART), math (trig), and Open3D (3D visualization)
- **Memory (MSP432E401Y)**:
  - 1MB Flash (program + persistent storage)
  - 256KB SRAM (runtime variables and buffers)
- **Microcontroller**: MSP-EXP432E401Y
  - 32-bit ARM Cortex-M4F with integrated DSP
  - Peripherals: UART, I2C, Timers, ADC, PWM, GPIO
- **ToF Sensor (VL53L1X)**:
  - Accurate measurements up to approximately 4 meters.
  - Communicates through I2C with microcontroller
- **Stepper Motor (28BYJ-48)**:
  - 512 steps per full revolution
  - 5V operating voltage
- **Driver Board (ULN2003)**:
  - LED feedback on step coil activation
  - Standard step/direction interface
- **3D Visualization**:
  - UART distance data saved to .xyz format
  - Rendered using Open3D into full point cloud scan
- **User Interaction**:
  - Press-button input (PJ1) toggles scan
  - Onboard LEDs (PF0, PN1) provide visual scan indicators

General Description:

This system is built to perform 3D distance-based environmental scanning using a Time-of-Flight sensor, stepper motor, and embedded control via a microcontroller. The key components include the MSP432E401Y microcontroller, VL53L1X sensor, a 28BYJ-48 stepper motor, and PC-side data handling through Python.

The VL53L1X ToF sensor emits short bursts of infrared light and measures the time it takes for the light to reflect back from the surface. This timing information is converted to distance by the sensor's internal ADC and digital processing hardware. The sensor communicates with the microcontroller through an I2C interface operating at 100 kHz. No external analog circuitry is needed, as the conversion and filtering occur within the sensor chip. The stepper motor is used to physically rotate the sensor to cover a full 360° field. Every 32 motor steps (22.5°), the microcontroller records a distance sample. These discrete angular positions allow the system to sample across the Y-Z plane. The stepper motor is driven via GPIO pins using a four-step wave drive pattern. The MSP432E401Y handles all logic and control tasks. It initializes communication with the sensor, triggers data acquisition, and sends processed readings over UART0. Each distance measurement is sent at 115200 baud to the host PC in real time. A Python script running on the PC reads each UART message and decodes it into distance and angle values. These are transformed into Cartesian (Y, Z) coordinates based on the known rotation step size. The X-coordinate is added frame-by-frame as the user manually moves the device along the X-axis. Once the full set of coordinates is written to a file, another script loads the data into Open3D to render a 3D point cloud for visualization.

Block Diagram:



Device Characteristics:

Parameter	Value
Bus Speed	20 MHz
UART Baud Rate	115200 bps
Python Version	3.9
Measurement Status LED	PF0
UART Tx LED	PN0
Additional Status LED	PN1

Motor Driver Board	Connected MCU Pin
+5V	5V
GND	GND
IN1	PH0
IN2	PH1
IN3	PH2
IN4	PH3

ToF Sensor Pin	Connected MCU Pin
VIN	3.3V
GND	GND
SDA	PB3
SCL	PB2

Distance Measurement

The distance measurement process is fundamental to spatial mapping and involves accurately measuring the distance between the Time-of-Flight sensor and objects within its field of view. This is achieved using the VL53L1X ToF sensor, which is a state-of-the-art sensor designed to measure distance based on the time it takes for emitted light to travel to an object and back. This process is sometimes referred to as ToF, where light pulses are emitted, reflected, and detected to determine the time it took for the round trip.

The VL53L1X sensor operates by emitting infrared laser pulses towards the object. The emitted light travels toward the object and reflects off it. The sensor then detects the time it takes for the light to return. This time-of-flight is the critical measurement that allows for the determination of the distance to the object.

The key principle behind this technology is the speed of light and its predictable, constant value. Using the time of flight, the distance to the object is calculated using the following formula:

Distance= (Speed of Light×Time of Flight) / 2

Where:

- **Speed of Light** is 3×10^8 m/s
- **Time of Flight** is measured in nanoseconds by the sensor.

This calculation provides the distance to the object in millimeters. The sensor is capable of measuring distances up to 4 meters with an accuracy of ±20 mm. The distance is then output as a digital signal.

Once the distance measurement is captured, it undergoes **signal conditioning**. Signal conditioning is the process of enhancing the accuracy of the signal by filtering out noise, amplifying weak signals, and ensuring they are suitable for digital processing. The conditioned signal is then transferred to the microcontroller for further processing.

The MSP432E401Y microcontroller receives the distance data from the VL53L1X sensor via the I2C communication protocol. I2C is a widely used protocol that allows for efficient communication between the sensor and the microcontroller. The I2C protocol uses two wires: the Serial Clock Line (SCL) and the Serial Data Line (SDA), making it a simple but effective communication method. The communication speed is 100 kHz, which is sufficient for the needs of this application. I2C allows the sensor to transfer the distance data to the microcontroller, which then processes the data and prepares it for further use in visualization.

Data Transfer and UART Communication

The processed distance data is then sent from the microcontroller to the PC using UART (Universal Asynchronous Receiver-Transmitter) communication. UART is a serial communication protocol that uses two wires: TX (Transmit) and RX (Receive). It operates asynchronously, meaning it does not require a shared clock signal. Instead, the two devices agree on a common baud rate, in this case 115200 bps, which ensures reliable and fast data transmission. Data through UART is transmitted in a series of 8-bit packets, with one start bit, one stop bit. This ensures that the data is transferred correctly without requiring a dedicated clock signal. This UART connection allows the computer to receive the distance data, process it, and finally visualize it as part of the 3D mapping.

For a full 360° scan, the stepper motor rotates the ToF sensor, collecting data at fixed intervals. The motor performs 512 steps for a complete rotation, and a measurement is taken every 32 steps (every 22.5°). This results in 16 measurements per rotation. This number of measurements provides sufficient resolution to map the surrounding environment with good accuracy. After each full 360° rotation, the sensor's position is incremented along the X-axis by 500 mm (50 cm) to collect data for the next layer of the scan.

Visualization

Visualization of the collected data is an essential aspect of the system. The distance data obtained from the ToF sensor only represents a 1D array of measurements (distance vs. angle). To create a comprehensive 3D map, this data needs to be transformed into Cartesian coordinates (X, Y, Z) to represent the scanned environment accurately.

The 3D visualization process is divided into different stages, which include data conversion, data processing, and rendering.

The first step in visualization is the conversion of the polar coordinates (distance, angle) into Cartesian coordinates. The angle is determined by the step number of the stepper motor, and the distance is directly read from the sensor. To convert these polar coordinates into 3D space:

- **Y-coordinate** is determined using the cosine of the angle:
  $Y = \text{distance} \times \cos(\theta)$
- **Z-coordinate** is determined using the sine of the angle:
  $Z = \text{distance} \times \sin(\theta)$
- **X-coordinate** is independent, as device was moved physically across x-axis

Once these calculations are done, each measurement is represented by a point in 3D space.

After the distance measurements are converted into Cartesian coordinates, the data is stored in an XYZ file format, where each line represents one point in 3D space with its corresponding X, Y, and Z values. The data is then processed using the following Python libraries:

- **PySerial**: This is used to communicate with the microcontroller over UART, ensuring smooth data transfer from the microcontroller to the visualization software.

- **Numpy**: Numpy is used to handle large arrays of data and to perform mathematical operations like trigonometric transformations (sine and cosine) required for coordinate conversion.

- **Open3D**: This library is used for 3D visualization. It takes the XYZ data points and renders them as a 3D point cloud that represents the scanned environment.

Once the data is visualized using Open3D, the user can interact with the 3D map, zoom in/out, and rotate the view to explore the scanned environment. This gives a clear and intuitive representation of the distances measured by the sensor across the full 360° rotation.

Application Note:

System Setup:

To assemble the device, start with connecting the VL53L1X Time-of-Flight (ToF) sensor to the MSP432E401Y microcontroller via I2C. Connect the sensor's SDA and SCL pins to PB3 and PB2 on the microcontroller, respectively. The sensor should be powered by a 3.3V supply from the microcontroller, and the ground (GND) should be connected accordingly. Next, wire the stepper motor to the ULN2003 motor driver. The motor's control pins (IN1 to IN4) should be connected to Port H pins (PH0, PH1, PH2, PH3) on the microcontroller. The motor requires a 5V power supply for operation, which can be supplied through the microcontroller. The onboard button (connected to Port J Pin 1 (PJ1)) will be used to trigger the scanning process. Pressing the button will initiate the 360-degree rotation of the stepper motor, which in turn activates the ToF sensor to collect distance measurements.

Before starting the scan, ensure all hardware components are properly connected. Once the hardware is set up, make sure your Python environment is ready. Install Python 3.9 and ensure all necessary libraries (PySerial, Open3D, Numpy) are installed. Then, navigate to the directory containing the Python script.

Ensure that the correct COM port is chosen on the python code, you can check this on your device settings (For example, I used COM5 in my implementation). Once this is set, the system will be ready to start receiving data from the microcontroller.

System Activation and Monitoring:

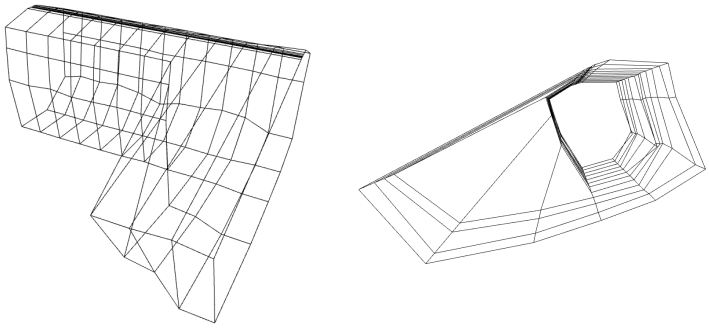
1. Run and Flash the Keil Code to the microcontroller
2. Press the reset button on the microcontroller
3. Run the Python data code, enter how many scans you want to take.
4. The onboard button (PJ1) will trigger the start of the scanning process. Once pressed, the stepper motor will rotate, and the ToF sensor will start collecting data.
5. During the scanning process, the system will take measurements every 22.5 degrees as the motor moves. The onboard LEDs will flash as feedback to show progress during the scan.
6. After completing the full 360-degree rotation (three full scans), the system will automatically stop the motor, and the second LED will turn on as the motor moves the opposite direction, returning to its original position.
7. If you wish to take more than 1 scan, move the sensor down at your selected displacement, (currently 500mm), then press PJ1 again to scan. Perform repeatedly until finished.

Expected Output:

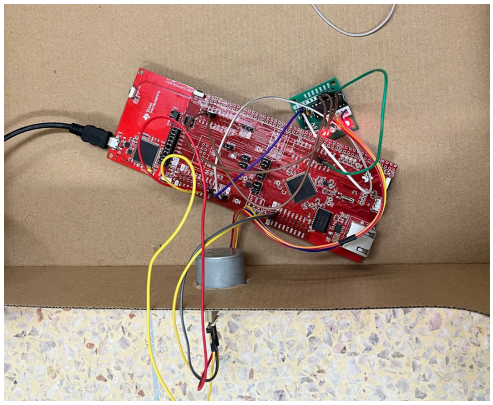
After the scanning process is completed, the **.xyz** file will contain all the XYZ coordinates, representing the scanned space (XYZ coordinates explained earlier in the detailed explanation section). This file is used by Open3D to render the 3D visualization of the environment, which can be explored interactively.

Once all data is collected, you will be able to see a detailed representation of the scanned space. The 3D scan can be rotated, zoomed in, and explored from different angles to gain a better understanding of the spatial relationships in the scanned area.

Picture of Hallway



Scans of Hallway



Physical Circuit

Limitations:

1. Microcontroller Floating-Point and Trigonometric Limitations

The MSP432E401Y microcontroller utilizes a 32-bit Floating-Point Unit (FPU), which provides support for basic floating-point operations. However, this FPU's precision is limited to 32 bits, which can result in small inaccuracies when performing more complex mathematical functions, especially trigonometric calculations like sine and cosine. These errors can impact the accuracy of calculations for distance and angle, particularly in applications where high precision is required, such as with very fine-grained sensor measurements.

2. Maximum Quantization Error for the ToF Module

The VL53L1X ToF sensor used in this system has a 1mm resolution. This means the smallest change in distance that can be detected by the sensor is 1mm. The quantization error associated with the sensor is 1mm, which sets a limit on the accuracy of the distance measurements. For applications requiring detection of smaller distances, this resolution might not be sufficient, and more precise measurements would require a different sensor with higher resolution.

3. Maximum Serial Communication Rate

The system uses UART for communication between the microcontroller and the PC, with the maximum baud rate set to 115200 bps. While this speed is suitable for most tasks, attempts to use higher baud rates, such as 230400 bps, resulted in data transmission errors and communication instability. This suggests that 115200 bps is the maximum stable baud rate for this system, which can limit the speed of data transfer, particularly when dealing with large volumes of data or when real-time performance is needed.

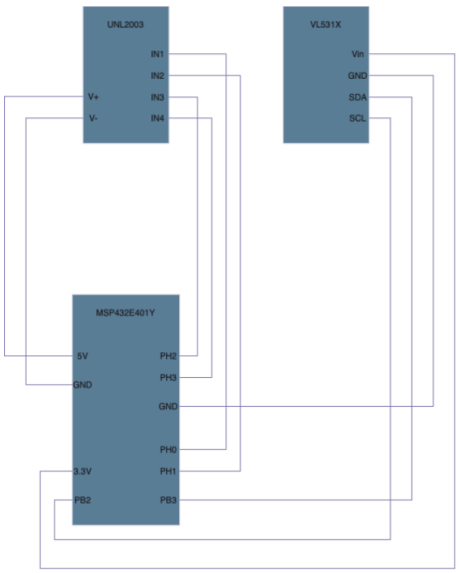
4. Communication Methods and Speed

The communication between the microcontroller and the VL53L1X sensor is handled via I2C, operating at a 100 kHz clock speed. Although I2C supports speeds up to 400 kHz, the system uses a 100 kHz rate for reliability and stability. At higher speeds, communication may become less stable, leading to potential data loss or inaccuracies. This choice ensures that the system works effectively but limits the throughput of data exchange between the microcontroller and the sensor.

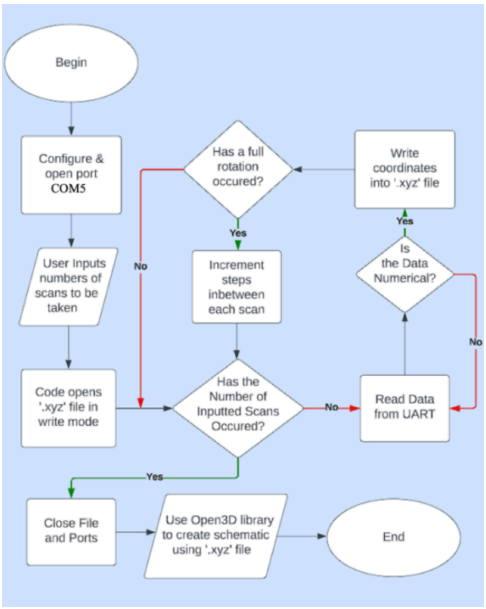
5. Primary System Bottlenecks

The stepper motor and serial communication speed are the primary bottlenecks affecting the performance of the system. The 28BYJ-48 stepper motor requires 512 steps to complete a full 360-degree rotation, which impacts the scanning speed. While this motor is suitable for precise scanning, the relatively slow rotation speed limits how quickly the system can capture data for the entire scan. Additionally, the UART communication speed of 115200 bps restricts the rate at which data can be transmitted from the microcontroller to the PC.

Circuit Schematic:



Python Flowchart:



Keil Flowchart: