2DX4: Microprocessor System Project
Final Project

Instructor: Dr. Haddara, Dr. Hranilovic, Dr. Shirani

Yuming Zhang - L02 - zhany257 - 400132290

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 [Yuming Zhang, zhany257, 400132290]

Demo video
https://drive.google.com/file/d/1cxaWGQ-mAGvTNbSz-GkdtEXfCA76OrDS/view?usp=sharing

Q1 Link:
https://drive.google.com/file/d/1K28y3a_vcP6suFlcZeR7cW1--OpMIZrC/view?usp=sharing

Q2 Link:
https://drive.google.com/file/d/1s4ZEd9gxObm3B9lbcKoyilL_FGTTa8mb/view?usp=sharing

Q3 Link:
https://drive.google.com/file/d/17-0q30WoQVoTgE5KvE9TJtv2Vo4lwZny/view?usp=sharing

Device Overview

Features

- Lidar distance measurement system
    - Applicable for indoor exploration and night time navigation
    - Communicates with a PC with USB
    - Visualization of data via Python program
    - Cost effective and portable
- MSP432E401Y Microcontroller from Texas Instruments
    - ARM Cortex-M4F Processor
    - 12 MHz default clock speed, 8MHz setting for lidar system, easily editable
    - 256KB of SRAM
    - LEDs for status indication every 45 degrees of motor rotation
    - Programmable in C, C++ and Assembly
- VL53L1X Time of Flight sensor
    - Detection range up to 4 meters
    - Small ranging error of 20 millimeter
    - 50Hz ranging frequency
    - 3.5 - 2.6 volt operating range
- 28BYJ-48 stepper motor
    - 512 steps per rotation
    - 12 - 5 volt operating range
- Communication
    - ToF sensor and microcontroller uses I2C serial communication
    - UART serial communication between PC and micro
    - 115200 bps
- Visualization
    - Python3 with Open 3D

General description

The Lidar system is an embedded 3D environment measurement device which uses a time of flight sensor to determine the distance of the area around it. The sensor is attached to a stepper motor which spins 360 degree within a single vertical geometric y and z plane to measure the distance. It requires manual movement along the x axis 20 cm each time for another 360 degree measurement. The distance data is transferred to a connected PC via USB to be recorded and visualized in 3D.

The lidar system includes a microcontroller, time of flight sensor and stepper motor. The microcontroller is the key component of this system, it redistributes power to all other components while also transferring data to the CP through USB. The stepper motor rotates the ToF sensor so that it archives an 360 degree scan. The ToF sensor measures the distance by emitting pulses of infrared light and determines the amount of time the light bounces back to the

sensor, then calculates the distance by converting analog signal to digital signal. The data is sent via I2C. The device must be connected to a PC that can communicate serially using USB to function. A python file is used to translate the collected data to xyz coordinate and visualized with open3D.

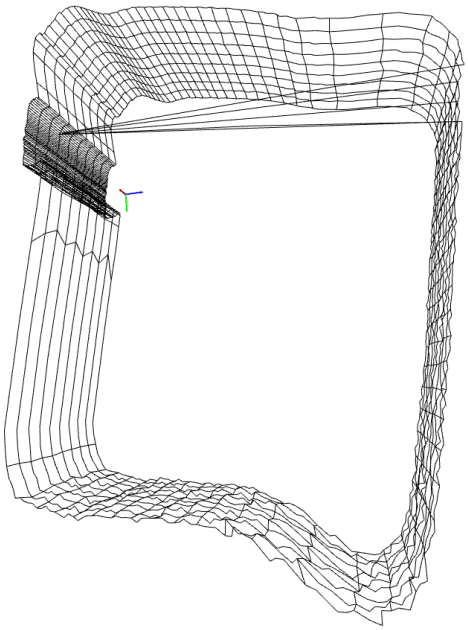

Figure 1. Screenshot of 3D visualization

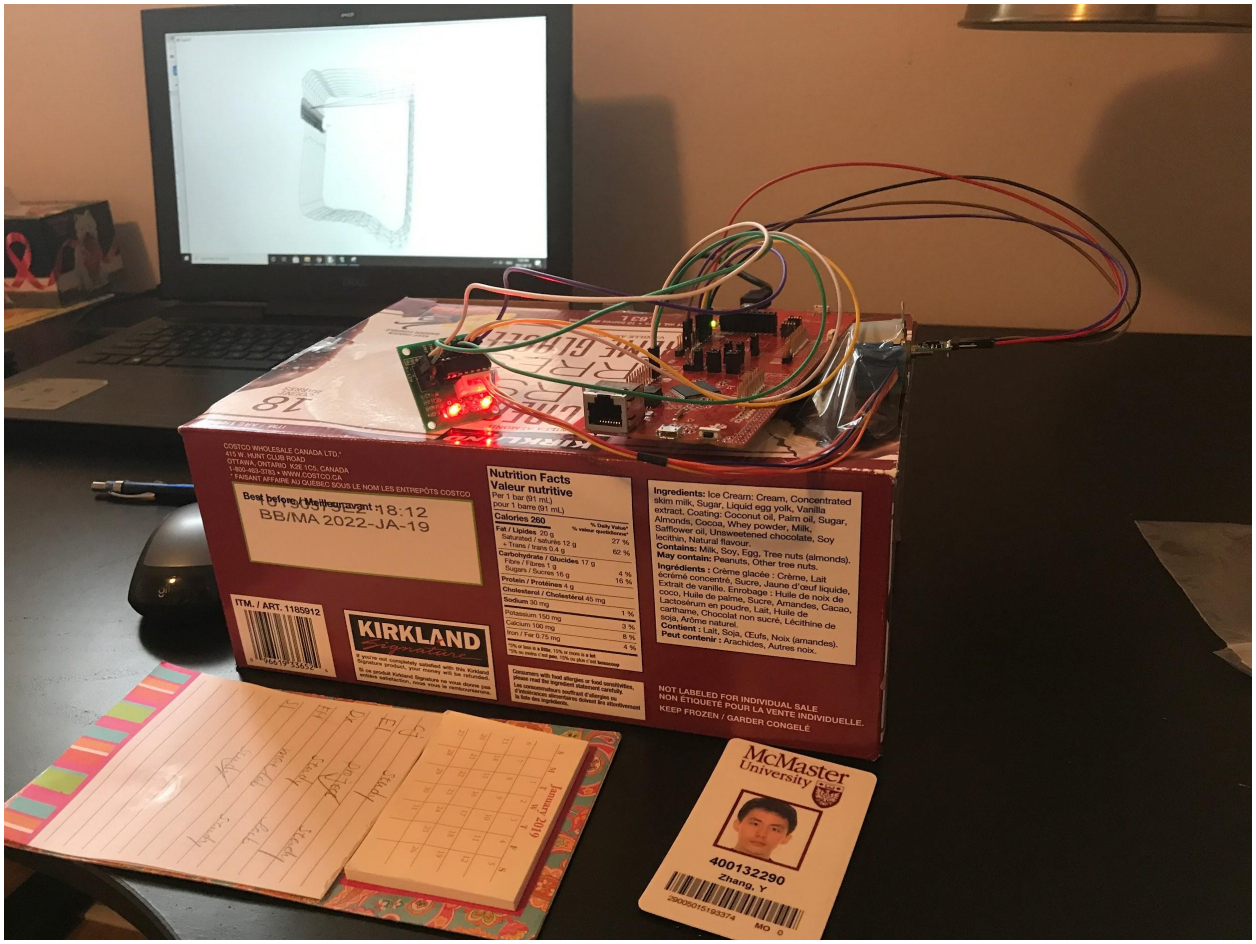

Figure 2. Lidar system connected to a laptop displaying data

Block Diagram

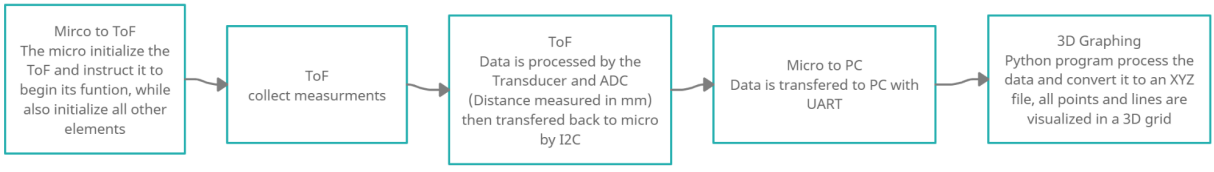

Figure 3. System block diagram

Device Characteristic Table

*pin not mentioned is not connected to anything

Specification	Info/Connections	
Bus Speed	8 MHz	
Baud Rate	115200 bps	
Serial Port	COM3	
Python	Version 3.8.8	
ToF sensor	VL53L1X	MSP432E401Y
	VIN	3.3V
	GND	GND
	SDA	PB3
	SCL	PB2
Motor Driver	ULN2003	MSP432E401Y
	IN1	PE3
	IN2	PE2
	IN3	PE1
	IN4	PE0
	+	5V
	-	GND

Table 1. Technical and connection specification

Detailed Description

Distance measurement

Distance is measured by the VL43L1X time of fight sensor, it is attached to a stepper motor to achieve a 360 degree distance measurement within a single geometric y-z plane.

The VL53L1X time of flight sensor measures distance by emitting a pulse of infrared light, and detects the time at which it takes the light to bounce back to the sencor. The speed of light is known with the common symbol ‘c’, the time ‘t’ for the sensor to detect light is twice the

time for the light to reach an object. Therefore, an equation for the time of flight sensor is D = c*t/2, where D is the distance. To calculate distance, the device first needs to translate the measured analog value into digital, then run it through this equation. Since the time of flight sensor only has an effective range of 4 meters, it has to check if the value is out of range, if it is, then the value is wrong. The Data is transmitted to the microcontroller via I2C. During the Initialization process, I2C functionality of the microcontroller is enabled to communicate with the time of flight sensor. GPIO PJ1, an onboard button is used as a toggle switch to start and pause the data capture process. To enable the button, the interrupt functionality must be activated. When the system is ready to capture data, the program waits for the sensor to capture data. The flight sensor and microcontroller both use the I2C protocol to communicate with each other. The measurement result is stored on the microcontroller and transferred to the PC with UART, then the data is processed by the Python program. After each data transfer, the stepper motor is moved one step in the counterclockwise direction. A counter variable is created to assist the distance indicator LED. Each time the motor moves, the counter is incremented by 1. The distance LED D2 is assigned to flash every 45 degrees of the rotation. If the remainder of the counter variable by 45 is 0, then The LED D2 would flash once. If the motor has moved 512 steps, the stepper motor makes one full rotation clockwise to untangle the wires. The X displacement is increased by 200 mm and all lights are turned off.

A python script is used to process the data received form the microcontroller by using pySerial library. The data is in the form of displacement, distance measured in millimeters and motor angle. The data is processed again and written to an .xyz file, which stores the coordinate of the measurements. The x component of the data is fixed based on the selected displacement, the y and z data are calculated using python methods. The motor requires a total of 512 steps to complete one complete cycle, each step a distance measurement is taken, 360 degree divided by 512 steps result in about 0.7 degree per step. With the angle and distance known , the y and z coordinate can be found by simple trigonometric. Let d be the measured distance, let Θ be the angle. The resulting equation to find y is d*sin(Θ) and z is d*cos(Θ).

Visualization

A 3D graph is used to visualize the data, the data is taken from the .xyz file, a point cloud is created from the data using the open3D library.

The visualization is performed on a windows 10 computer, with an intel core i7 processor, an NVIDIA GenForce GTX 1060 and 12 GB RAM. A script is written on Python 3.8.8, this version is used since some other version does not support open3D. The program utilizes open3D, numPy, and pySerial Python libraries, which include all the methods necessary to assist the collection, processing and visualization of the data. Math and Queue are also imported from the Python libraries to manage the data. Once the program receives data, a comma is used to separate the variables. If the data is invalid, it waits until new data is available. The main function handles the data conversion from cynical coordinates to cartesian coordinates. Open3D is initialized which updates the geometries in real time.

Application example

To set up the lidar detection device, some necessary software is required to be installed on the device computer. The original design process is performed on a windows 10 PC.

- Install python 3.8.8 x64 version, make sure to add path.
- Install pySerial, Open3D, numpy, UART

The device should have all necessary software to run the program

1. Connect PC to microcontroller
2. Connect motor, sensor and microcontroller as shown in the circuit diagram
3. To determine a COM port is being used correctly, run “python -m serial.tools.list_ports -v” in the command prompt while the microcontroller is connected to the PC.
4. Variables ‘num_slice’ and ‘dist_per_slices’ can be changed in the python code to change the measurement parameter.
5. When the code has been modified with the desired parameter, run the python program.
6. Press the reset button to start the program.
7. The program takes time to boot up, wait for all lights to turn off, then press the PJ1 button.
8. Position the sensor into a vertical yz dimension that needs to measure.
9. The motor will start rotating and distance information measured. The LED will flash every 45 degrees. Wait for the motor to stop.
10. Move the system to the next x location and press PJ1 again.
11. Once every data plane user wants to collect is completed, a 3D graph of the data will show up. The graph can only be closed when the program is reset.

Limitation

1)

The microcontroller uses a Floating point unit that can operate 32 bits arithmetic operations. Math functions can be applied to floating variables. Because the is in 32 bits, 64 bits operation will take more instruction to complete.

2)

4000 mm / (2^16) = 6.1*10^-2 mm.

3)

The maximum standard serial communication rate is 128000 bps, this is checked by looking at the UART port in the device manager.

4)

The microcontroller and the time of flight sensor communicate via I2C protocol, the transfer rate is 100kbps with a clock speed of 100kHz.

5)

The primary limitation is the available range of the time of flight sensor. The sensor is always in long distance mode, the minimum timing that will work with in all modes is 33 ms, but the timing for it to reach the maximum 4 meters is 140 ms.

Circuit schematic

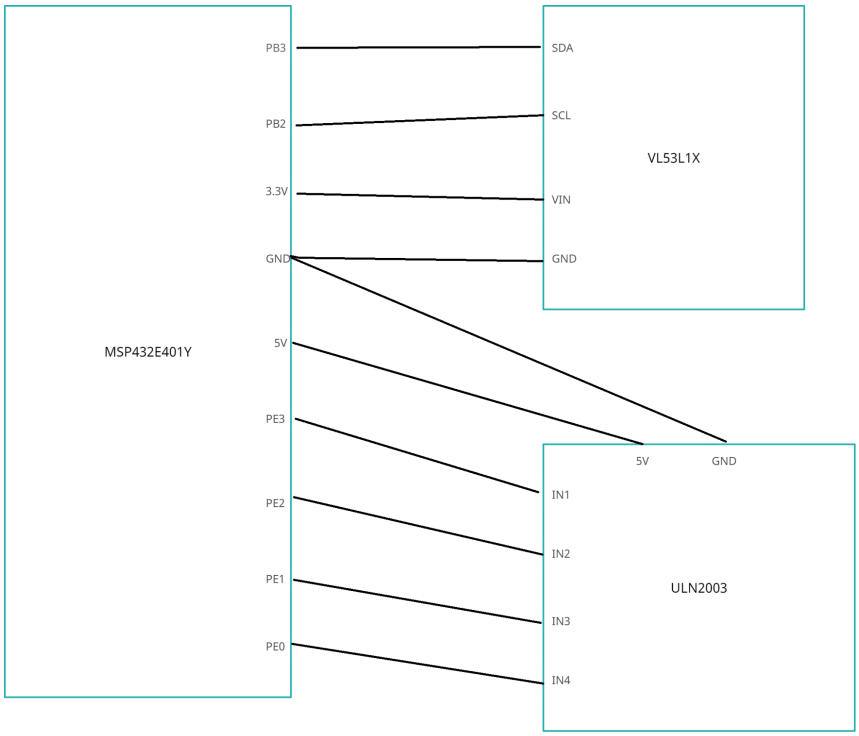

Figure 4. Circuit schematic

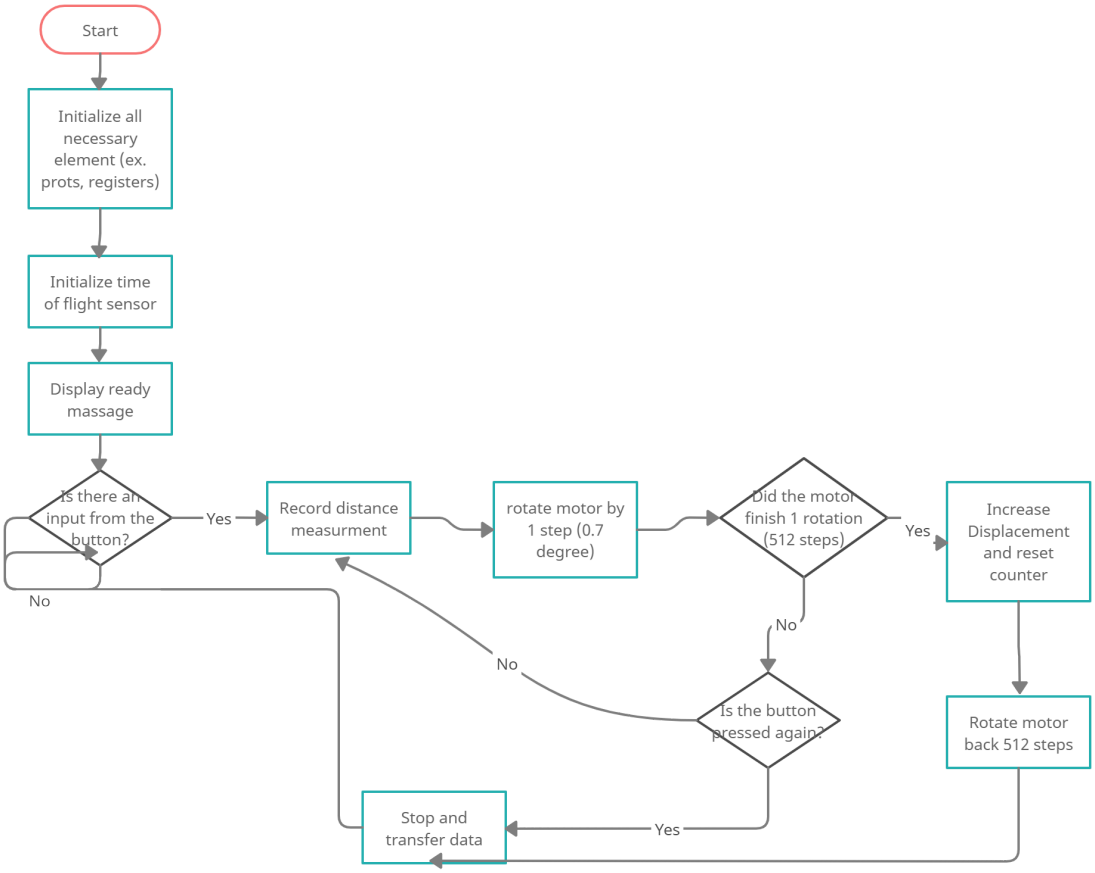

Figure 5. Microcontroller flowchart

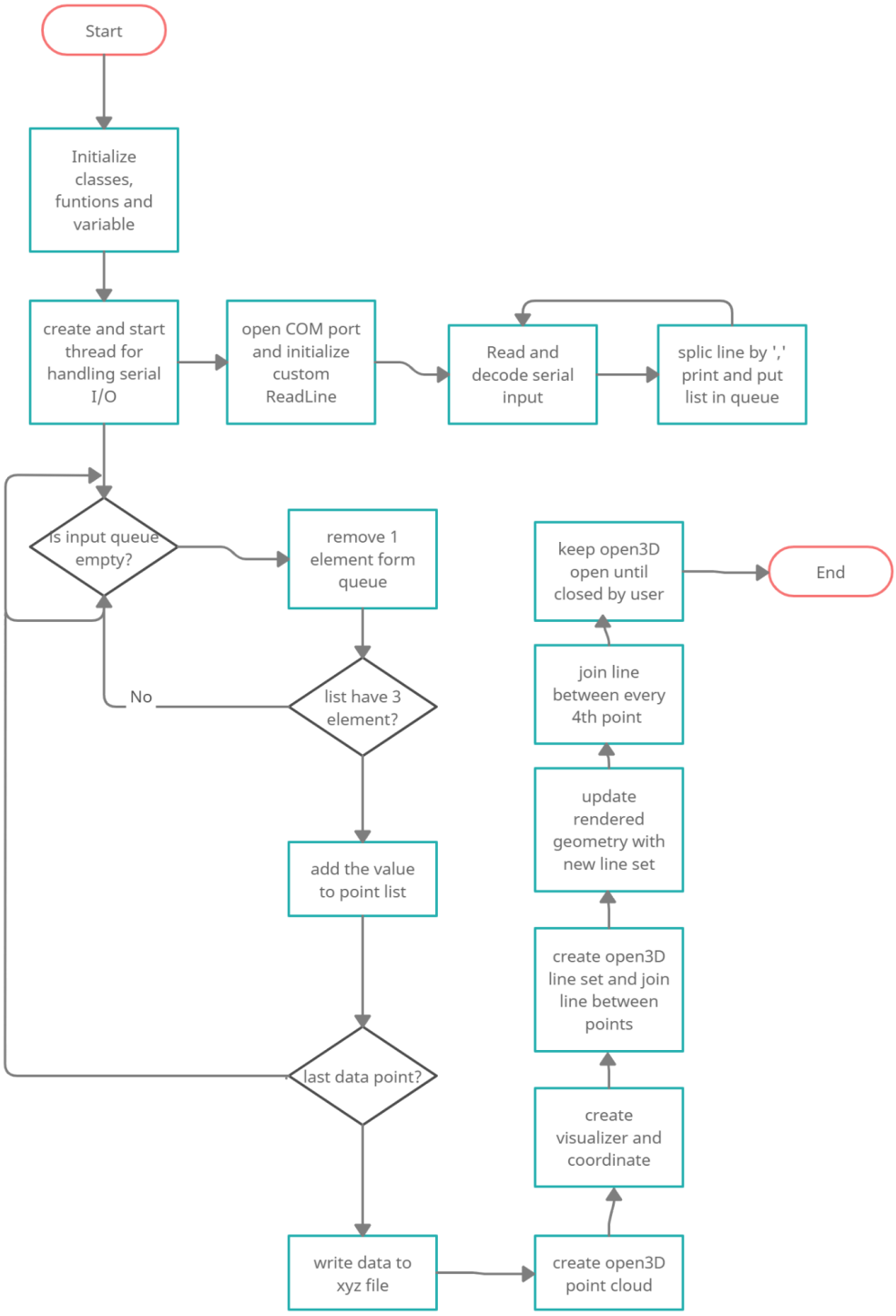

Figure 6. Python flow chart