NC State University

Department of Electrical and Computer Engineering

ECE 785: Spring 2018

Project #3: SONAR Point Cloud Builder

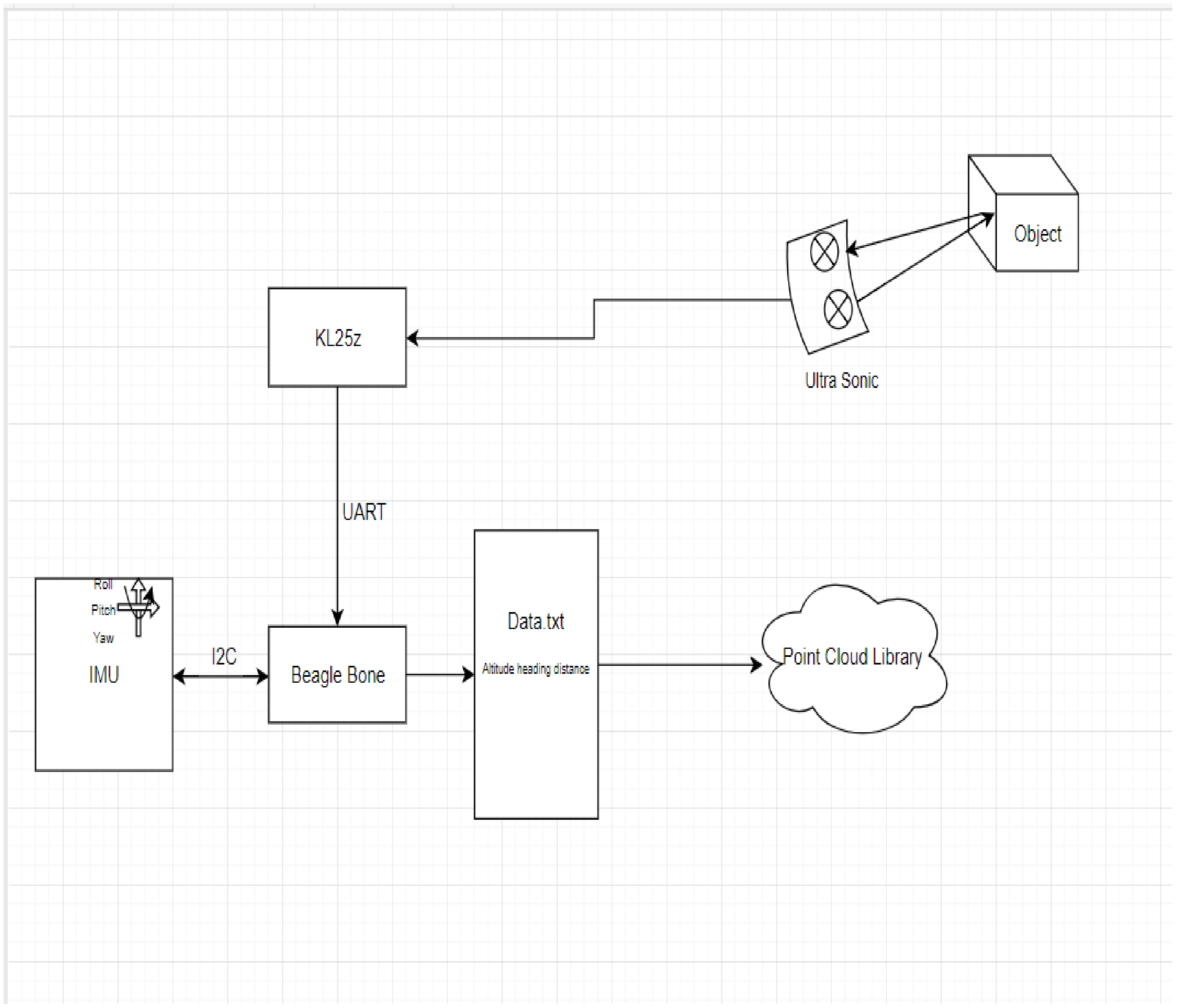
by

Shaunak Chokshi(200194438)

Veerakumar Natarajan (200208042)

Design of Hardware

Overview of Hardware Design



1. Interfacing IMU with Beaglebone black using I2C

I2C-2 is the I2C controller used to interface IMU with Beaglebone black.
Below table represents pin connection between Beaglebone black and IMU

Beaglebone	**IMU**
Vcc	Vcc
Gnd	Gnd
P9.19	SCL
P9.20	SDA

2. Interfacing SONAR with KL25z using ADC

We need to pull down the voltage of Ultrasonic as the output voltage of analog pin is 5V. So, we add a resistor divider in between such that maximum voltage will only be 3.3V as KL25z can only handle values up to 3.3V.

Following tables shows pin connection between KL25z and SONAR

KL25z Pin	**Ultrasonic**
Vcc	Vcc
Gnd	Gnd
PB.0	Trigger
PE.2	Analog Output

3. Interfacing Beaglebone black with KL25z via UART

Below table shows pin connection of Beaglebone black and KL25z via UART.

KL25z	**Beaglebone**
PE.22(Tx)	P9.11(Rx)
PE.23(Rx)	P9.13(Tx)
GND	GND

Design of Software

Overview of Software Implementation

KL25z will read data from SONAR with interval of 25ms and send that value via UART. Beaglebone will wait for the value from UART. Once Beaglebone black read value in UART, it will start determining sensor position by sending IMU value to Madgwick code to determine quaternion components. We use quaternion components to calculate roll, pitch and yaw.

Now we have altitude, heading and distance of obstacle at one point, we will write that value to a file called ‘data.txt’. We repeated this for 1000 times and calculated altitude, heading and distance of obstacle for 1000 points and store that in a file ‘data.txt’.

We used data.txt file to create point cloud and viewed it using PCL library.

1. Calculation of roll, pitch and yaw from IMU (MPU-9250)

We are reading ax, ay, az, gx, gy, gz, mx, my and mz from sensor. Once values are read, we will be passing that values to madgwick code which will return quaternion components.

From the quaternion components, we are calculated roll, pitch and yaw as follow

yaw = atan2f(2.0*(q1*q3 + q0*q0), q0*q0 - q1*q1 - q2*q2 + q3*q3);
pitch = asinf(-2.0*(q0*q3 - q0*q2));
roll = atan2f(2.0*(q1*q2 + q0*q3), q0*q0 + q1*q1 - q2*q2 - q3*q3);

Yaw gives as heading of the IMU and we used value from gyroscope for gyroscope which is not accurate but it give some relative altitude value.

2. Calculation of obstacle distance from SONAR (HCSR04)

Triggering

We have connected the Trigger pin to PortB.0 which is triggered for 12us and the triggering happens timely after 25ms. To have a good timing precision 25ms we are using SysTickTimer and for setting the bit high for 12us we are using PIT Timer.

ADC Calculation

We are sampling the ADC at 48Mhz and we set the threshold of 0.2V and we will be discarding any values below this value. Once we have the value above this value we calculate the time between the peaks using Timer0. Once we have the time we can calculate the distance(distance=t*330/2).

Sending data to Beagle Bone

We are sending data to Beagle Bone using UART2 of KL25z with a baud rate of 115200. Enumerated below are the pin connection between Beagle Bone and KL25z.

3. Communication between Beaglebone black and KL25z

Beaglebone black and KL25z is interfaced via UART to transfer distance value from SONAR to beaglebone black. UART is configured as follows

- Baudrate – 115200
- Data bits – 8

In Beaglebone black, we configured P9_11 and P9_13 as UART by following commands.

$ config-pin p9_11 uart

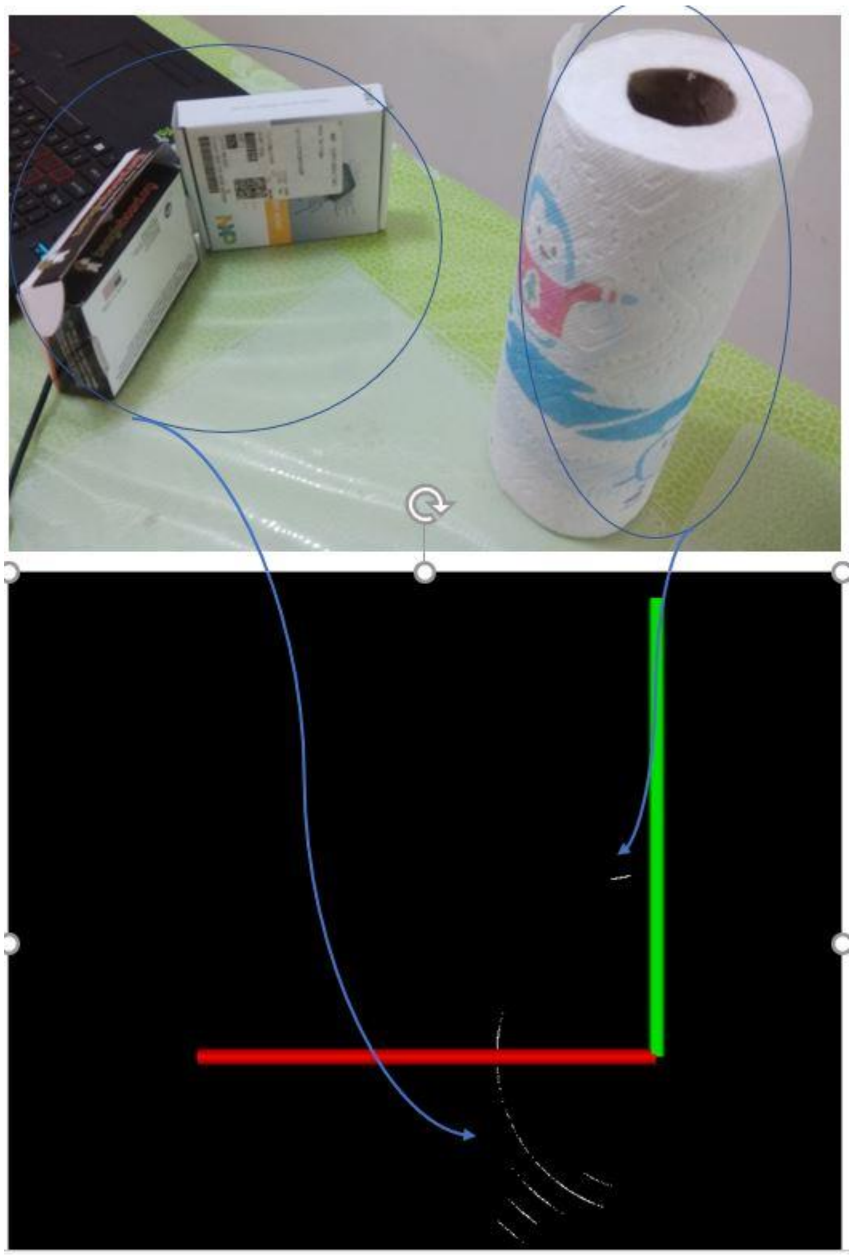
$ config-pin p9_13 uart

4. PCL library

Beaglebone will store data file which has three column and 1000 rows. Following is the structure of data file

```
float data[1000][2] = {
{1.21, 78.0, 2.8},   // {altitude, heading, obstacle_distance}
……
}
```

PCL Output



YouTube link: https://www.youtube.com/watch?v=ih3KZj42JEM&feature=youtu.be

Challenges

1. **Values from IMU are not stable**
   ax, ay, az, gx, gy, gz values are not stable even when IMU is stationary. We are looked into issue, it seems IMU will give value only somewhat close to correct value but with some random noise added to it.

   I passed those values to Madgwick code to check whether quaternion components will become constant or constant. But it still remains fluctuating even IMU is stationary. Then we changed 'betaDef' value from 0.1 with smaller steps. We change that until the values becomes somewhat constant. Finally we fixed betaDef value to 0.5 to correct noise from IMU raw data.

2. **Synchronization between KL25z and Beaglebone.**
   We needed synchronization mechanism to map between SONAR value from KL25z with sensor position value from Beaglebone black. We tried with configuring one pin as interrupt in beaglebone black which is connected to a digital pin in KL25z. Whenever KL25z reads SONAR data, it will trigger that digital pin. Since that pin is configured as interrupt in beaglebone, we can implement reading position sensor data from IMU in ISR.

   But we faced issue with configuring a pin as interrupt in beaglebone. So to provide synchronization, we will be waiting to read data from UART. Once we read data, then we proceed with determining position of IMU.

3. **PCL library**

   We faced few issues in installing PCL library. When we tried to install PCL, we encountered "unmet package dependency" issue. It took some time to resolve all the dependency issues and get the PCL library up and running.

Next Steps

1. **Navigation system for visually impaired people**

   Obstacle detection and warning can improve the mobility as well as the safety of visually impaired people specially in unfamiliar environments. For this, firstly, obstacles are detected and localized and then the information of the obstacles will be sent to the visually impaired people by using different modalities such as voice, tactile, vibration.

   To make this happen, values from IMU sensor should be reliable. We can implement Kalman or particle filter to remove noise from raw sensor data.

2. **Autonomous vehicle and mapping**

   By IMU and SONAR, we can have 3D point cloud of obstacles around you. Using 3D map, we can navigate vehicle with more precision. It is not possible to determine location by GPS always, so whenever GPS fails, we can switch into IMU and SONAR mapping to have better navigation system.