NC State University

Department of Electrical and Computer Engineering

ECE 785: Spring 2018

Topics in Computer Design

Project #3: SONAR Point Cloud Builder

by

Aditya Bhopale (asbhopal)

Viplove Rakheja (vrakhej)

Shrirang Deshpande(skdeshpa)

Introduction:

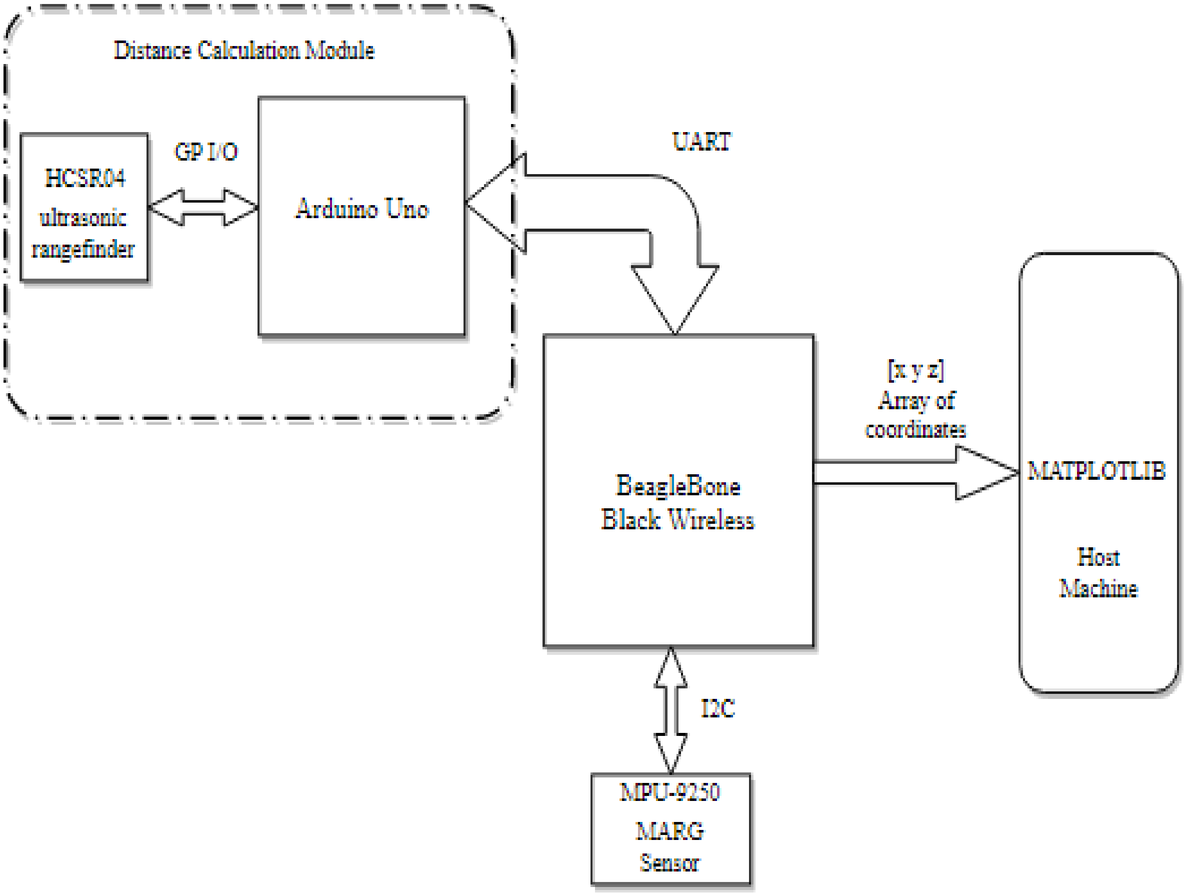
The project deals with developing a system that determines the location of object using an ultrasonic sensor ranging module and a 9 degree-of-freedom MARG (magnetic field, angular rate and gravity) sensor. The coordinates/location of the object thus obtained are displayed over Point Cloud Library in order to generate a 3D model of the object.

Hardware used:

1. BeagleBone Black Wireless
2. Arduino Uno
3. HCSR04 (Ultrasonic Rangefinder)
4. MPU-9250 (MARG Sensor)

Description:

The Block diagram describing the different module operation is shown below:


Distance Calculation Module
HCSR04 ultrasonic rangefinder
GP I/O
Arduino Uno
UART
BeagleBone Black Wireless
[x y z] Array of coordinates
MATPLOTLIB
Host Machine
I2C
MPU-9250 MARG Sensor

Distance Calculation module:

This module consist of two hardware devices: Arduino Uno and HCSR04 (ultrasonic rangefinder/SONAR). The module is designed such that the input to HCSR04 is fed through Arduiono Uno board to the trigger pin of HCSR04, which makes the module generate pulses from the transmitter and the output is hacked from the Analog pin of the HCSR04 instead of echo pin and those samples that we got from the analog pin are used to calculate the distance of the object depending on the number of samples above a certain threshold (which was determined by observing the output using Logic Analyzer) the distance was computed using the below calculations :

$$distance = duration * 0.034/2$$

$$ret\_val = (duration + (count1 * delay\_samp * 0.001)) * (0.034/2)$$

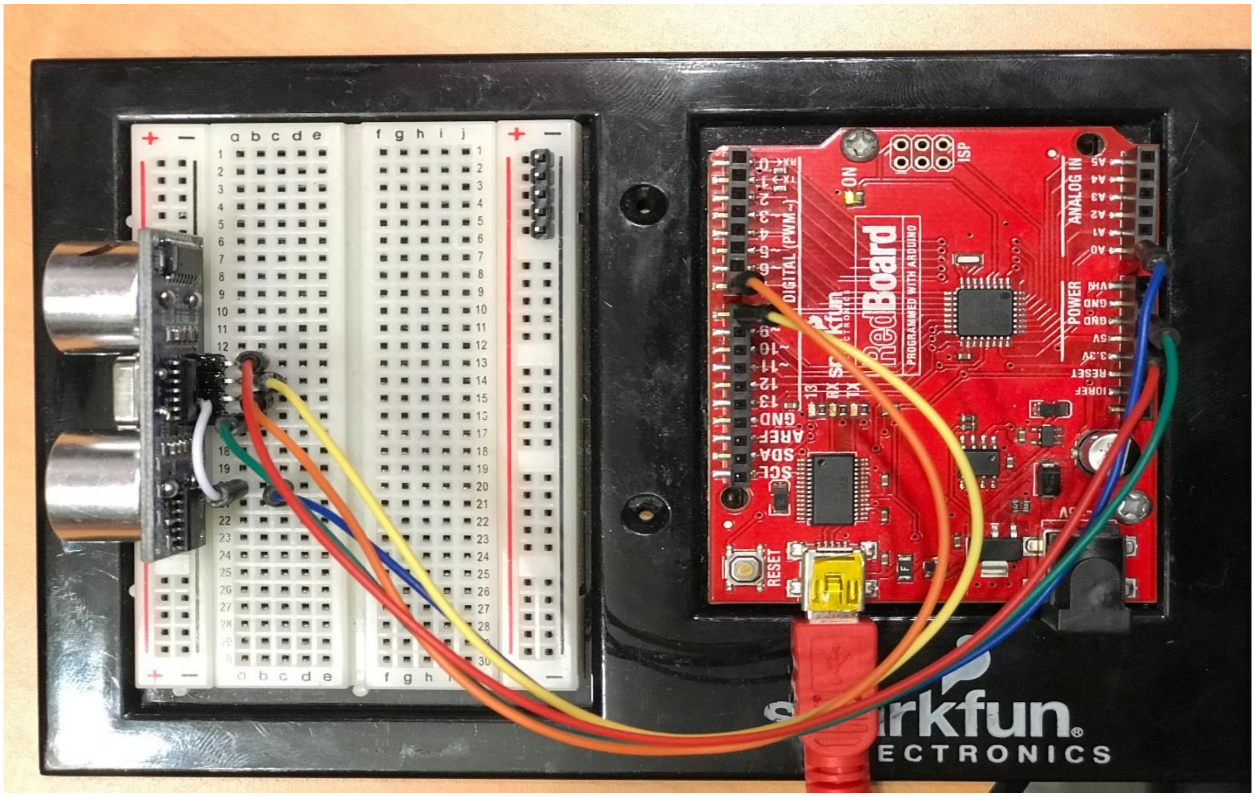
***duration : period for which the echo pin is high*

***ret_val : value from the analog pin after threshold*

The distance thus calculated is sent to the BeagleBone black board using serial communication. The triggering of the HCSR04 module is designed such that after waiting for 2000 samples, the Arduino will send the make the trigger high again to collect next samples. The module is designed such that Arduino board will continuously keep on producing the distance value and keep sending it to Beaglebone board.

Connections :

1) Trigger (HCSR04) to Pin 8 (Arduino)
2) Analog pin (HCSR04) to Port A0 (Arduino)
3) VCC and GND (HCSR04) to 5V and GND (Arduino) respectively



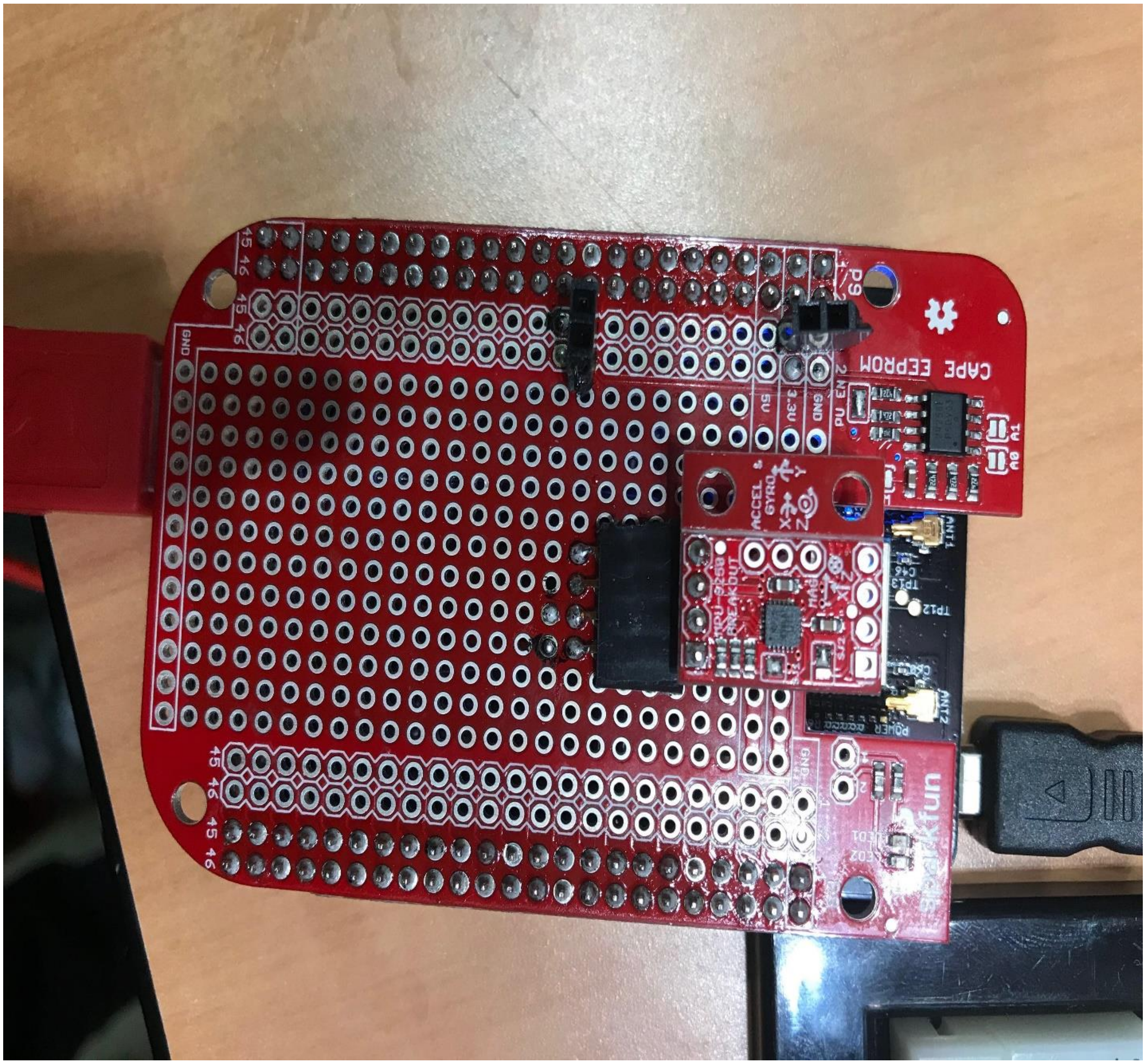
Interfacing MARG Sensor with BeagleBone:

MPU-9250 (MARG Sensor) is used to determine the 9-axis of freedom of the object. MARG sensor provided data for 3-axis accelerometer, 3-axis gyrometer and 3-axis magnetic field (Compass). MARG sensor communicated with BeagleBone and vice versa using I2C bus.

In order to make the MARG sensor produce all the 9 correct values, various register were being configured using BeagleBone. The data received from the MARG sensor was communicated to BeagleBone using the I2C interface again.

Ports used for I2C:

1) GND (MARG Sensor) to Pin 1 of Port 9 (BeagleBone)
2) VCC (MARG Sensor) to Pin 3 of Port 9 (BeagleBone)
3) SCL (MARG Sensor) to Pin 19 of Port 9 (BeagleBone)
4) SDA (MARG Sensor) to Pin 20 of Port 9 (BeagleBone)



Generating the coordinates of object:

In order to generate the coordinates of the object, the values achieved from the MARG sensor are then passed to MARG filter by calibrating the values that we got from MARG sensor to the range specified.

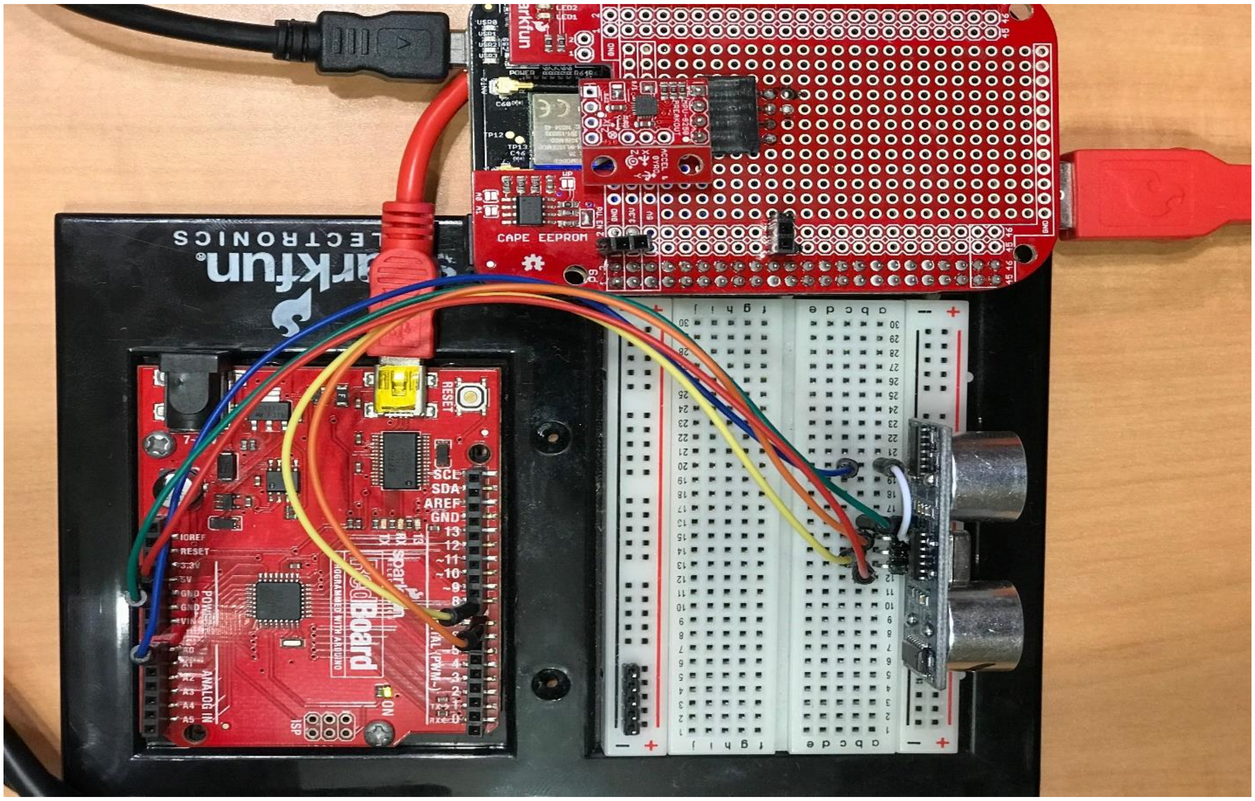
After getting the calibrated values, these values are passed to the Madgwick's MARG filter in order to determine the object's relative location and direction by generating the quaternion coordinates namely q0, q1, q2 and q3 depending upon the values received from the MARG sensor. Now, these quaternion values obtained are used to generate the rotation matrix values i.e. *roll, pitch* and *yaw*.

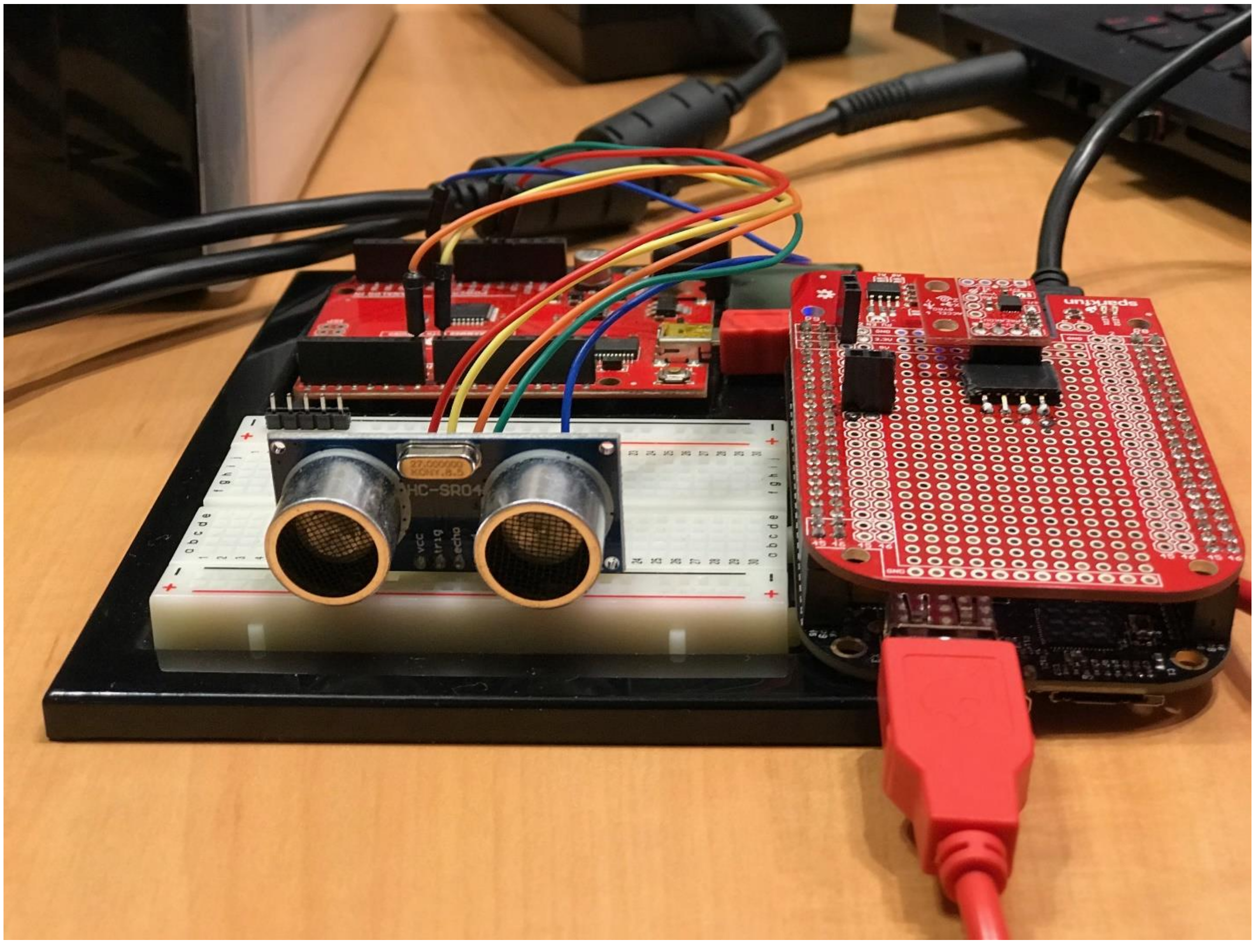
$$roll = atan2f((2.0 * (\mathrm{q0} * \mathrm{q1} + \mathrm{q2} * \mathrm{q3}))/(1.0 - 2.0 * (\mathrm{q1} * \mathrm{q1} + \mathrm{q2} * \mathrm{q2})))$$
$$pitch = asinf(2.0 * (q0 * q2 - q3 * q1))$$
$$yaw = atan2f((2.0 * (q0 * q3 + q1 * q2))/1.0 - 2.0 * (q2 * q2 + q3 * q3))$$

These rotation matrix values are used to generate the Cartesian coordinates of the object position. The distance value comes from the *Distance Calculation Module.*

$$x = distance * \cos(pitch) * \sin(yaw)$$
$$y = distance * \cos(pitch) * \cos(yaw)$$
$$z = distance * \sin(pitch)$$

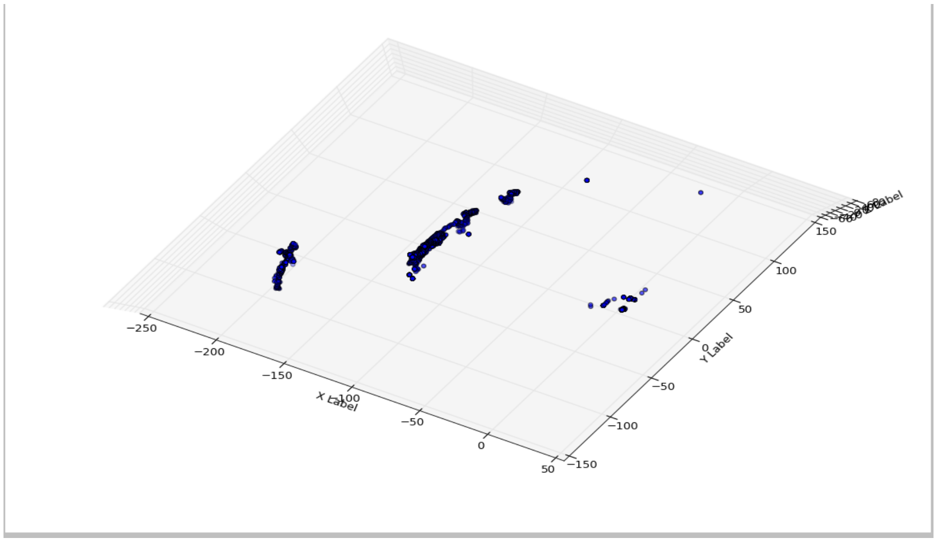
**Pitch and yaw are in radians.*

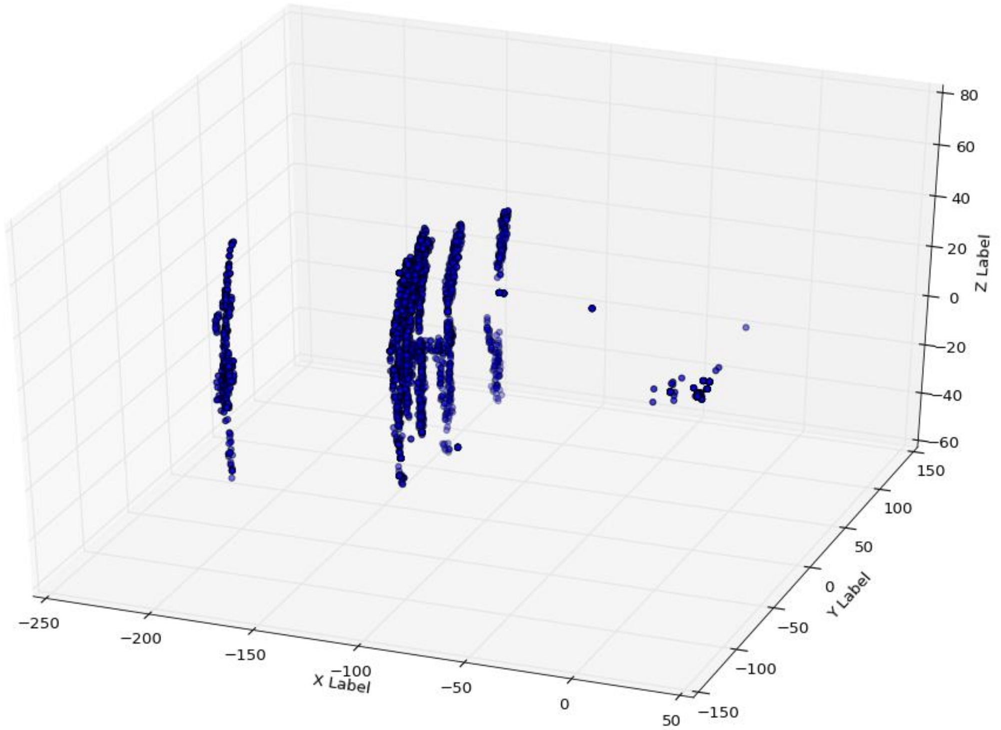




Generating the Point Cloud:

After generating the cartesian [x,y,z] coordinates of the position of object, these values are stored into a file and that file is run on the host machine using MATPLOTLIB to generate the 3D image of the object.





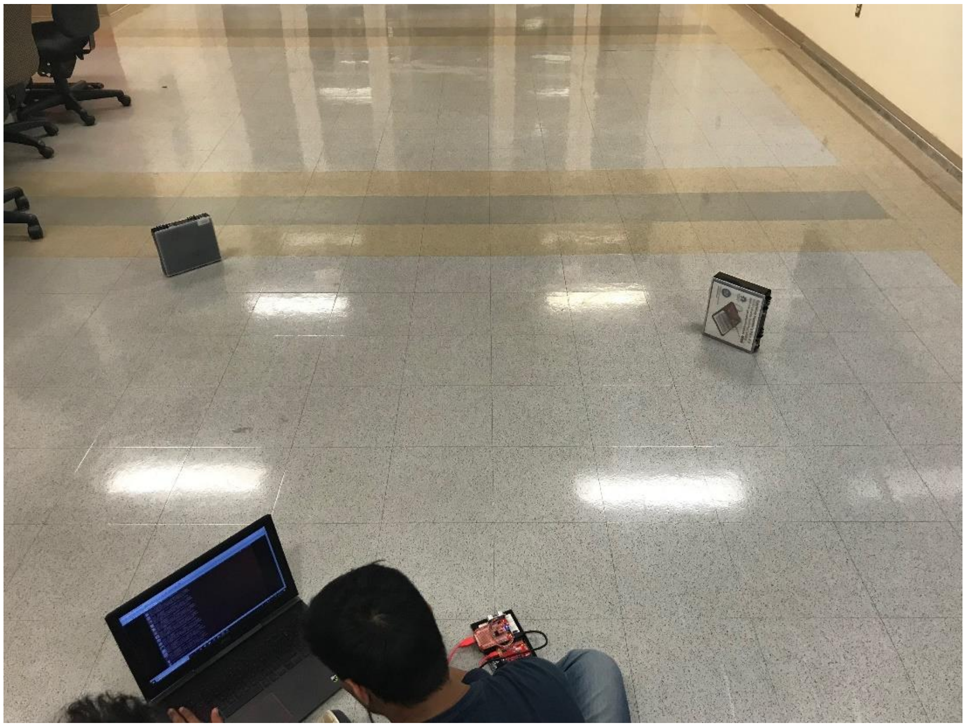


Fig. Testing environment

Steps to run the Design:

1) Use dmesg | tail to know the port number at which the Arduino is connected to Beaglebone.
2) Set the port accordingly in the code.
3) Make the file in directory "IMU First".
4) Upload the code in the directory "Ultra" by using command "make upload".
5) Whenever desired values are obtained, kill the code, and copy the generated file in python folder for point cloud generation.
6) Run the python script to obtain the plot for point cloud.

Challenges Encountered:

There were various challenges faced during the project:

1) Taking the output of the SONAR from the analog pin and using those values to compute the distance of the object.
2) Sending the value of distance calculated from Arduino to BeaglBone using UART.
3) Calibrating the values of MARG sensor in order to compensate the hard ion losses and making the values to be generated in input range for the Madgwick's MARG filter to generate the quaternion values.

Next Steps:

1) Incorporating the HCSR04 with BeagleBone along with the MARG sensor in order to avoid use of Arduino specially only for getting distance value from HCSR04.
2) The system is for now operated manually by moving the system and tracing the object to generate the coordinate matrix. Rather than that, it can be designed on an auto rotator system in order to reduce the errors in the value obtained from MARG sensor.
3) The system can be made faster and more responsive by using threads to simultaneously communicate with the HCSR04 sensor and MARG sensor.
4) If using multiple development boards, then the communication can be made more sensitive by communicating through handshake protocols and thus removing the unwanted values and get more precise results.
5) Removing the loose wire connections and using PCB instead.

Conclusion:

The design ran successfully and the values of *roll, pitch and yaw* were successfully generated from the quaternion coordinates generated from the values of MARG sensor without any errors. The values were constantly changing as the orientation of the system changed. Also, the HCSR04 sensor was successfully integrated with Arduino and the values that were taken from the analog pin of HCSR04 was successfully used to capture the distance of object and also differentiating the values between nearby and far object. The communication between the Arduino and BeagleBone was successful and the distance value was used to obtain the values of coordinates of the object.

The link to watch the system working operation :

http://www.youtube.com/watch?v=0DHaMRvyqhk