Proceedings of IOE Graduate Conference, 2017
Volume: 5 ISSN: 2350-8914 (Online), 2350-8906 (Print)

Indoor Odometry and Point Cloud Mapping

Prabhat Sanu Ligal [a], Bikram Acharya [b], Pradeep Bajracharya [c], Prasun Shrestha [d], Pratik Pokharel [e], Sharad Kumar Ghimire [f]

[a, b, c, d, e, f] *Department of Electronics and Computer Engineering, Pulchowk Campus, IOE, TU, Nepal*

Corresponding Email: [a] ligal3245@gmail.com, [b] aacharya.bikram@gmail.com, [c] bajra.pradeep@gmail.com, [d] shresthaprasun1@gmail.com, [e] prokharelprateik1933@gmail.com, [f] skghimire@ioe.edu.np

Abstract

Indoor localization and mapping is an important problem with many applications such as emergency response, architectural modeling, and historical preservation. In this project, a metrically accurate, GPS-denied, indoor 3D static mapping system was developed using a moveable base coupled with three degree of freedom IMU. The experiment was performed by scanning a flat apartment which closely resembled the actual test environment. First the distance of various elements in the surrounding was collected using the LiDAR module along with the orientation of LiDAR Lite. This is termed as scanning mode of the system. After full scan of an area, the system was manually switched to Odometry mode wherein the system was moved to another location for new scan. The distance travelled as well as the direction of travel was noted. Secondly, the data is processed to obtain the position of different points in 3D space. The vast data set is stored in Point Cloud format. The point cloud is processed and stitched by removing the outliers, clustering error-free data and segmenting data to different planes. The scanning and odometry are two different modes that are better operated separately. This is due to the fact that scanning takes time and simultaneous movement registered erroneous data. This increased the number of outliers and hence produced an anomalous map. The toggle between two modes are maintained using a manual switch which provided more accurate data on separate implementation.

Keywords

Point Cloud – 3D Scanning – Odometry – Indoor Mapping – LiDAR

1. Introduction

With the advancement in technology, the need for indoor mapping and localization method for applications such as emergency response, architectural modelling and historical preservation is seen more crucial than ever. The existing mapping techniques include using simple 360 degree capture of an environment using a camera but this doesn't give complete information of the environment [1]. Furthermore, they need to have completely or properly lit environment. Next, the use of multiple sensor and their fusion is also an option. This creates a chaotic collaboration with complex integration and algorithm development which may lead to expensive cost. In any case, the existing systems do not provide a complete mapping of the environment with desired information. There are already developed Light Range Finders (LRFs) to assist mapping using remote sensing with light radiation but they are expensive. This creates an ever growing need for a low cost indoor mapping and localization module .

The objective is to develop a metrically accurate, GPS-denied, indoor 3D static mapping system using a moveable base coupled with 3 DOF sensors and LiDAR. There are two stages implemented in the scanning and mapping approach. First, there is scanning and odometry which takes place one after another in a loop but not simultaneously. This is due to the fact that scanning takes time to get the data and simultaneous movement will produce erroneous data. The toggle between two modes are maintained by using a manual switch which provides more accurate data thus the system is a static system. In scanning mode, the distance of various elements in the surrounding was collected using the LiDAR module in raster scan pattern. The orientation of LiDAR Lite at time of measure in

scanning mode is noted. Then, the system was manually switch to Odometry mode to move the system to desired location. The distance travelled as well as direction change in terms of angle is noted using the potentiometer and IMU respectively. The position of system is also determined by using dead reckoning.

Secondly the data is transmitted to computer via Serial interface to visualize the position of different points in 3D space along with odometry information. The 3D point cluster is stored in Point Cloud format.

Finally, since the point cloud is erroneous, the point cloud is processed through thresholding to remove the obvious error points caused due to limitation of sensor. As the purpose of the project is to obtain measurable data for further calculation, the point cloud is processed for filtering as well as segmentation to detail the output which is displayed in the screen for visualization.

The experiment performed by scanning a flat apartment gives the output which closely resembles the actual test environment. The Odometry gives distance and direction of motion which on visualization closely resembles the actual path travelled. One limitation of the system originates from the use of magnetometer to calculate the direction of path travelled. Scanning is restricted only to even terrains and high magnetic field in the surrounding or change in magnetic declination over a larger distance introduces offset in the map [1].

2. System Overview

The overall system can be divided into two part: acquisition hardware that computes the linear distance travelled by the system in different direction and software which computes those information to convert into point cloud. The basic idea behind point cloud mapping is LiDAR detect the linear distance to different points in the surrounding. To create an all-round map, LiDAR module is rotated in sweep motion such that it can scan in every direction. Due to physical limitation of actuator, the module cannot rotate in every direction but field of view is limited to 180 degree yaw (horizontal) and 135 degree pitch (vertical). This is considered as a hemisphere. In figure 1 the block diagram represent how the whole system work together. IMU and rotatory sensor were used to detect the change in location based on previous location. The servos are connected to LiDAR module to scan linear distance along with field of view. Arduino Mega serves as the controller for IMU, rotatory sensor, servos and LiDAR as well as raw data processing and serial communication. Laptop and computer communicate with controller using serial control protocol. After retrieval of raw data software computes into point cloud mapping which represent real spatial coordinates.

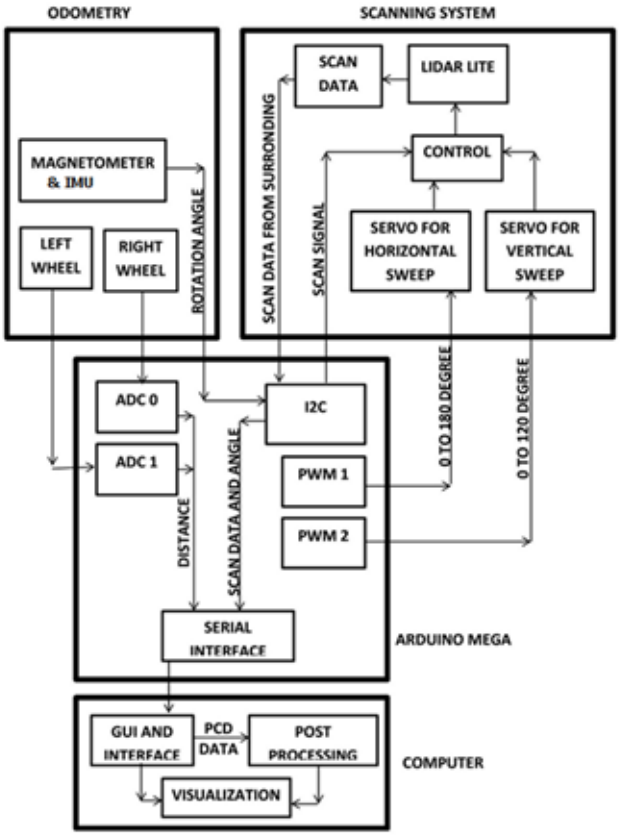

Figure 1: System Block Diagram

2.1 Hardware Overview

The Scanning system comprises of a LiDAR Lite module and two servo motors connected such that the motors give double axis rotation freedom to the LiDAR Lite, upto 90 degrees vertically and 180 degrees horizontally. Fundamental princilple of Light Detection and Ranging (LiDAR) is calcuate the distance travelled based on the time of flight of reflected light rays [2]. The LiDAR Lite takes 0.02 seconds for data acquisition and each full scan gives a hemisphere scan that merely takes 5 minutes.

2.1.1 Scanning System

LiDAR is a remote sensing method which uses light in form of a pulsed laser to measure distances. It measures distance on time of flight (TOF) principle [3]. Time of flight is the duration taken by light emitted by the LiDAR to incident on the object of interest and reflect back to the LiDAR. The LiDAR Lite takes 0.02 seconds for data acquisition. The motor with vertical axis gives horizontal sweep to the LiDAR Lite and after each sweep, the motor with horizontal axis rotates by 1 degrees giving new vertical orientation to the LiDAR Lite. The process continues until both the motors reach their maximum angle values. Thus each full scan gives a hemisphere scan.

2.1.2 Odometry

The change in direction and distance travelled by scanning system is determined using odometry. Odometry is the use of motion sensors to determine information such as relative position with respect to the starting position of the system[4]. The overall scanning system is mounted on a four- wheeled framework. The wheel is coupled with a continuous potentiometer of 10 Mohm. The wheel coupled to the potentiometer has a circumference of 22.5 cm. The rotation of wheel changes the resistance of potentiometer coupled to it. This is interpreted using the ADC value in the Arduino. So, depending on how much the potentiometer resistance is changed, the distance is estimated. Initially, the ADC value read is converted to zero and considered as reference point. Then the ADC values are categorized into four quadrants each of width 256 ADC value. The end points 0 and 1023 are in the first and fourth quadrants respectively. Each ADC value is compared with previous value and depending on the increment or decrement of values based on the previous values the direction of rotation is obtained. By coupling with the direction of motion these ADC end points are used to determine count of wheel in particular direction. In this configuration though, decreased ADC values refers to forward rotation and vice versa. The direction of path travelled is determined as the angle of rotation of the system about the horizontal plane. The angle of rotation is determined using IMU. The IMU is fit to match the center of the system and it helps in calculation of path travelled or yaw. It has fluctuation of 1 to 2 degree during rotation so the threshold is managed to 2 degrees.

The information obtained from odometry is transmitted to the computer to determine the position of the system related to the original position. The position is calculated based on the distance that it travels and the angle at which it currently moves as well as the previous position. This process is called dead reckoning. The distance is calculated every time ADC value for the wheel changes. The distance obtained is compared with the distance available for five consecutive times. The data is transmitted provided the distances are same for each instance. The data transmitted includes format (‘O’, distance, angle) where ‘O’ represents the fact that the data being forwarded is from Odometry. After each distance travel, the distance is reset to zero for new reading.

2.2 Software Overview

The scanned data is sent to the computer from Arduino via USB communication. The baud rate and the USB port where data arrives is defined at setup i.e., 9600 baud rate. When the data arrives, it is classified into different category based on the header of the packet formed for serial communication. The data sent by scanner to the computer is called Point Cloud data. A point cloud is a set of data points in some coordinate system. In this case the scanned data is in spherical coordinates system where the position of a point is specified by three numbers: the radial distance of that point from a fixed origin (r), its polar angle measured from a fixed zenith direction (theta), and the azimuth angle (phi) of its orthogonal projection on a reference plane that passes through the origin and is orthogonal to the zenith, measured from a fixed reference direction on that plane. The user interface for viewing output data, is created on QT platform using OPENGL library. GUI provides the options for viewing angle changes, file handling and point cloud rendering in 3D space. The point cloud coordinates are converted into rectangular coordinates system as (x, y, z). For visualization, Point Cloud Library offers visualization library. Basic post-processing like filtering and segmentations are done for effective analysis of the result. Filtering is used for removal of any points as noise data, and segmentation is used to cluster the related points together for the formation of relative object. Also the

distance between two points can be determined from GUI using the point pick feature.

3. Integrating Odometry And Scanning

The overall system highly depends upon integrating odometry system which estimates the current location of the system and scanning system which scans to return the distance to different points the surrounding. It is crucial to match these two modules otherwise there is an offset between different scans. Since scan time is very slow, our system was not able to scan and move at the same time. Hence system is toggled between scanning and odometry mode manually. IMU and linear distance sensor is integrated to estimate current location from previous location. This approach is also known as dead reckoning. The use of 3DOF IMU provides orientation information which allowes Odometry to calculate current coordinate and look at vector.

3.1 Interial Measurment Unit

To derive the orientation and direction of the system IMU module is used. IMU system is basically comprised of accelerometer, gyroscope and magnetometer [5]. Angular rate obtained from gyroscope is fused with gravity vector of accelerometer to fix angular drift issue. Finally magnetometer magnetic vector is integrate to achieve accuracy and direction at which module is pointed to.

3.2 Dead Reckoning

Dead reckoning or dead-reckoning is the process of calculating one's current position by using a previously determined position, or fix, and advancing that position based upon known or estimated speeds over elapsed time and course [6]. The mathematical relation used for dead reckoning differs from one system of implementation to another. The car-like steering used in this robot, turns in a curvature of constant and known radius, while differential drive robots can make very narrow turns, it can even spin in its position. For a steering type dead reckoning, the process of keeping track of the position deals with three variables: X and Y Cartesian coordinates and the angle (A) of the nose of the robot.

Dead reckoning can give the best available information on position but it is subjected to significant errors due to many factors as speed and direction. As an example, if displacement is measured by the number of rotations of a wheel, any discrepancy between the actual and assumed diameter, due perhaps to the degree of inflation and wear, will be a source of error. This therefore leads to accumulation of error in long run.

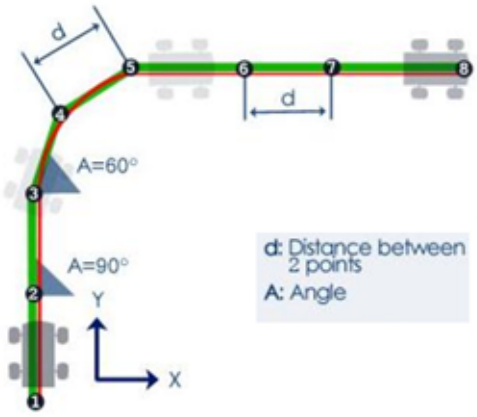

Figure 2: Dead Reckoning

Dead reckoning can give the best available information on position but it is subjected to significant errors due to many factors as speed and direction. As an example, if displacement is measured by the number of rotations of a wheel, any discrepancy between the actual and assumed diameter, due perhaps to the degree of inflation and wear, will be a source of error. This therefore leads to accumulation of error in long run.

3.3 Scanning Sweep

LiDAR module can only provide one dimensional data point which represents distance between incident points to origin of module. LiDAR module is attached to servos such that it can rotate around X axis and Z axis. The field of view for scanning module is 180 degree yaw and 90 degrees pitch. To obtain full field of view the servo sweep from left to right and after completion horizontal then next servo changed it direction by 1 degree pitch at the speed of 50 degree/sec. This process repeats until point cloud for field of view is not completed. In following figure green curve plan represents the FOV. Due to physical limitation of servo and construction material, at high speed the servos cause vibration and instability. Due to vibration

movement LiDAR distance calculated variance increase and was more prone to error. Finally scan rate was reduced to half to improve accuracy.

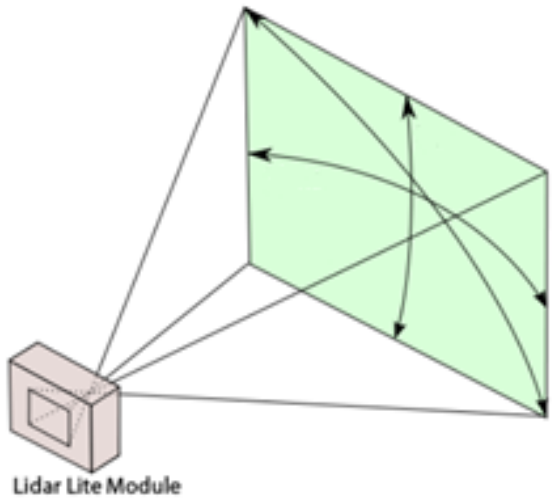

Figure 3: Scanning Field Of View

After calculating linear distance for each angle in sweep scan, point cloud is generated by computing the distance and orientation at that instance of time. The compute point is translated by system current coordinates; it helps to map the point cloud in correct position.

Figure 4: Stable case of servo (Left) and Jitter case of servo(right) as shown by laser pointer

4. Point Cloud Analysis

Laser scans generate point cloud datasets of varying point densities. Higher point density represents the higher detailing of Surface [7]. Sparse point cloud or low point densities are created either by sensor/ measurement error or by less scanning rate. Point Cloud gives the information about planes, curvature and other 3D surface features. Our system provides 50 points per second and scanning for 5 minutes gives half hemisphere i.e 15000 points. Sparse points should be removed by statistical analysis for better calculation and better result.

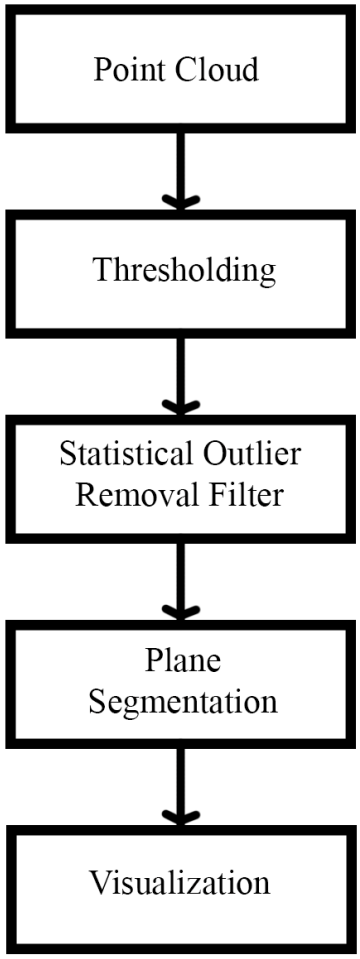

Figure 5: Software Block Diagram

4.1 Thresholding

The point cloud data generated by the system includes unwanted points due to measurement error. Thresholding is the simplest method for segmentation of unwanted points created beyond the limitation of system/sensor. The maximum range for our system/sensor is 30m and minimum range is 0.025 m. Points generated beyond this range is clipped by default in order to get statistically important points for calculation.

4.2 Statistical Outlier Removal Filter

The target of this analysis is to be able to view the indoor mapping that includes planar surface. Sparse points obtained is usually the error that can occur due to measurement error, light interference, multiple reflection usually in corners (planes in angle less than 90) and duplicate signal detection. Sparse points can be filtered by performing statistical analysis on each point's neighborhood, and trimming those which do not meet a certain criteria. Our sparse outlier removal filter is based on the computation of the distribution of point to neighbor's distances in the input dataset. For each point, we compute the mean distance from it to all its neighbors. By assuming that the resulted distribution is Gaussian with a mean and a standard deviation, all points whose mean distances are outside an interval defined by the global distances mean and standard deviation can be considered as outliers and trimmed from the dataset [8]. Removing the sparse points provides the higher density point cloud so that the 3D surface is represented better.

4.3 Plane Segmentation

The purpose of this scanning is to identify various 3D surface for measurement and calculation. Higher density point cloud obtained from Statistical Outlier Removal Filter gives better representation of 3D surface. Plane detection is the initial step to ensure separation of point cloud having similar geometrical nature. A visually appealing and distinct representation of the surrounding is obtained by applying color gradient to different planes viz. wall and ceilings in our testing [9]. For any points in the cluster set that was within the threshold distance (10 cm) from the place, the point was included into the plane. More accurate measurement and calculation is possible only after segmentation applied. RANSAC (RANdom SAmple Consensus) algorithm is widely used for plane detection in point cloud data. The principle of RANSAC algorithm consists to search the best plane among a 3D point cloud [10]. In the same time, it reduces the number of iterations, even if the number of points is very large. For this purpose, it selects randomly three points and it calculates the parameters of the corresponding plane. Then it detects all points of the original cloud belonging to the calculated plane, according to a given threshold. Afterwards, it repeats these procedures N times; in each one, it compares the obtained result with the last saved one. If the new result is better, then it replaces the saved result by the new one.

5. Experiment

The assessment of the system output (metrically accurate map) is objective of the following experiment. Each experiment is repeated on the period of development. For each experiment, system is connected to software which listens to the serial communication. Software estimates the location and processes the point cloud based on the result generated by the device. The system is not fully automated so the user has to move it around and change scanning and odometry mode. The following 6 illustrates the floor plan where experiments was conducted and it should be noted that the floor plan image is not drawn in scale.

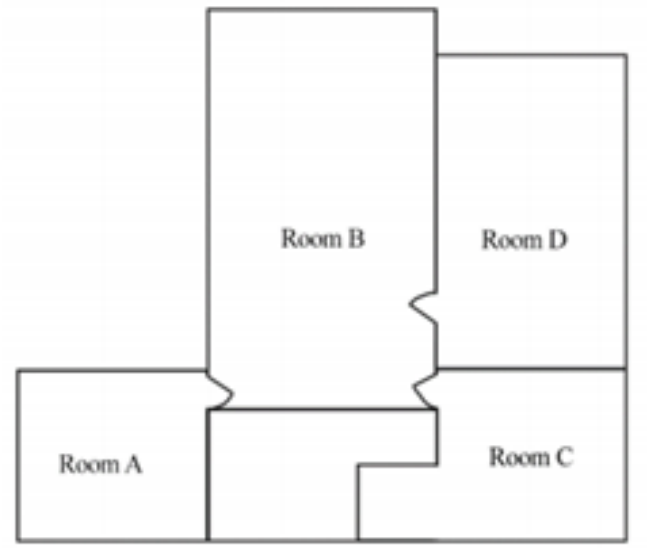

Figure 6: Floorplan

5.1 Indoor Test

The experiment was performed in a closed flat apartment where three of four rooms were scanned viz. Rooms A, B, C and D. The figure 7 illustrates the point cloud scanned from the room D as shown in figure 6. It can be noticed in figure 7 that portion of the room is a vacant space which was due to open door during the experiment.

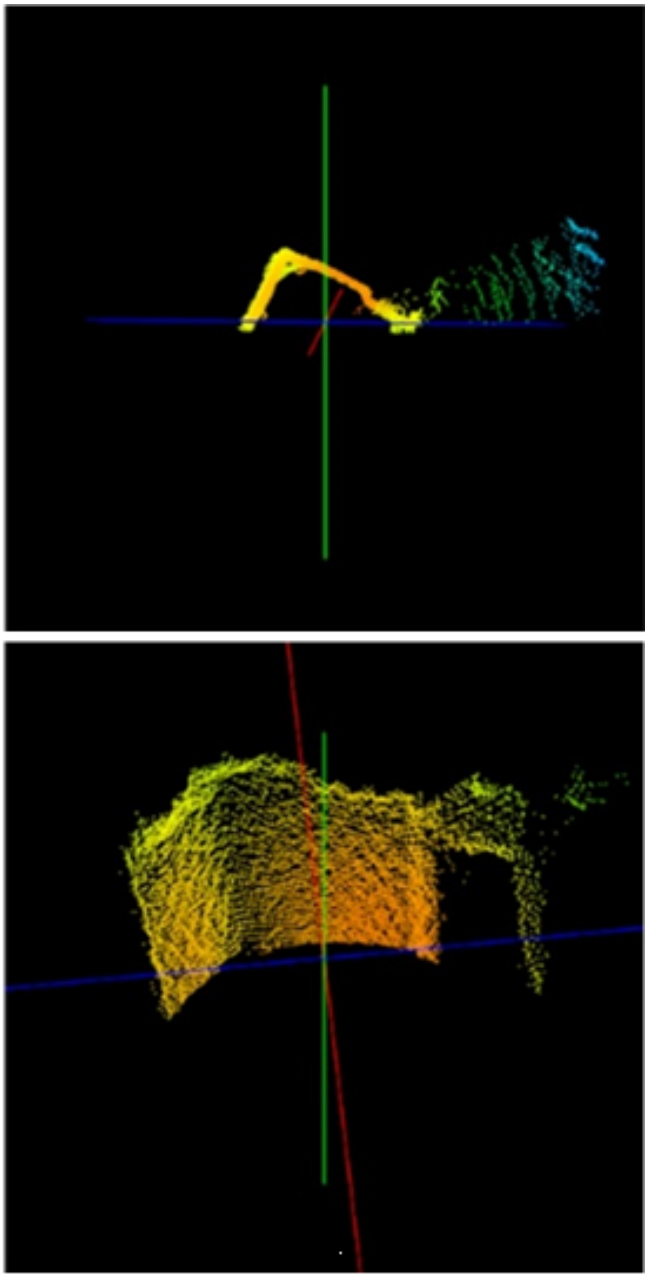

Figure 7: Point Cloud Mapping for Room D (Top: Top view , Bottom : Projections Views)

The figure 8 illustrates the comparison between actual photo of the room with point cloud mapping. Vacant space in figure 8 is due to opened door space. Another noticeable structure is the sink counter, Sink counter space is observable and closer than back wall.

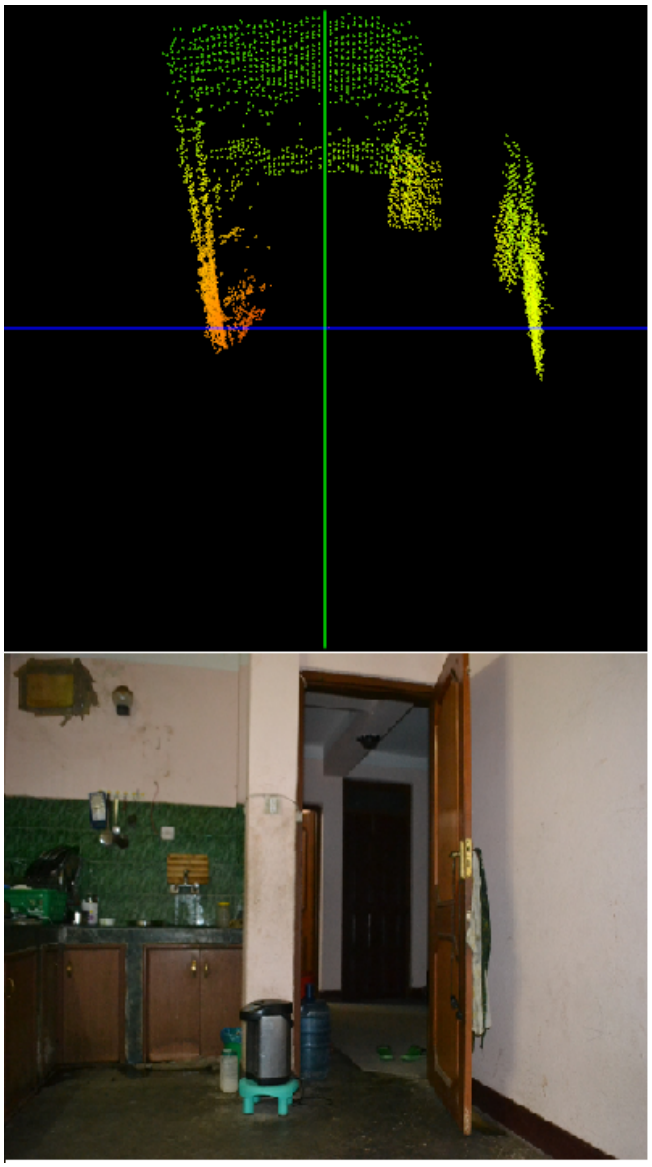

Figure 8: Point Cloud Mapping for Room C (Bottom: Acutal Image of Room C)

5.2 Odometry Test

Core feature of this project is to enabling continous scanning from different positions. System can start scan from one region and continously merge scan. It avoid additional stiching step. This feature allows seamless integration of point clouds and generate full scan for any shape floor region.

The scanning was started at Room C and then the system was moved to Room B and then to Room A. The movement was recorded and plotted along with the scan of the room. In figure 9 white line represents the path

traced.

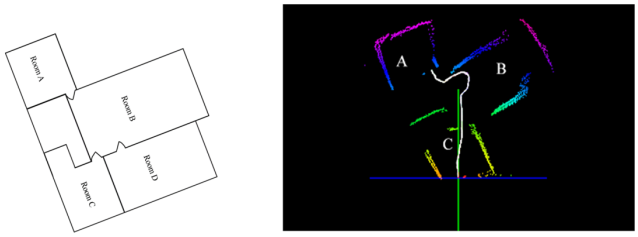

Figure 9: Top view of Odometry Output

Without the lack of the odometry data, point cloud of the same wall does not align. Since the device is moved, point cloud is off by displacement unit. Figure 10 illustrate the correction of the offset by integrating odometry displacement metric.

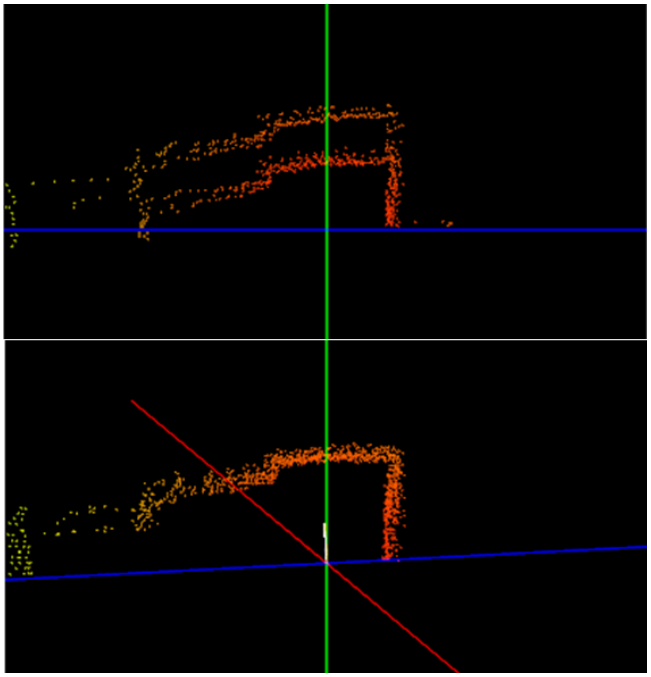

Figure 10: Top view of Wall (Top : Without Odometry Bottom: With Odometry)

5.3 Accuracy of experiment data

The scanned data had outliers generated due to few limitations in the system. The servos used for scan produced jitters because of the fact that interrupt-driven servo routines didn't actually give a very stable output pulse. Besides this, the potentiometer used for tracing the path taken by the system was not exactly linear. This meant that the path travelled by the system had some error in the distance measured. The outliers produced were removed using filter in the software which reduced the cumulative errors to 1.538%, measurement of a distance of 650cm was shown as 640cm. Another major application would be measurement based on captured point cloud. After capturing point cloud, software can calculate distance between any points. In figure 11, segmented point is represented by different colors and is clearly separates the indoor features such as wall and ceiling.

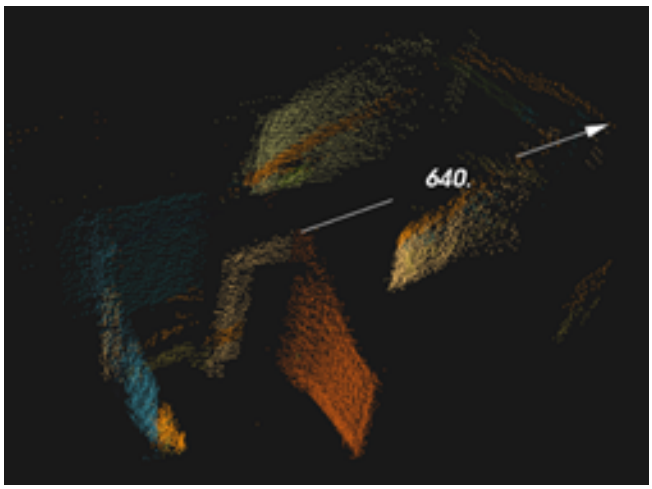

Figure 11: Calculating Distance from Point Cloud

6. Conclusion

Indoor point cloud mapping has a promising application and can provide spatial information in real scale. On another hand, point cloud capture and odometry for practical application needs to be optimized. Currently system has very slow scan rate which can be improved. Point cloud data seems be match the real world coordinate distance with error margin below 5%. Point cloud data can be further processed and filtered to remove outlier coordinates. Filtered point cloud caused false segmentation detection rate to minimum. The simple features such as wall, ceiling can be extracted from the point cloud.

References

[1] R. Li. Mobile mapping: An emerging technology for spatial data acquisition. 1997.

[2] T. Luttel M. Himmelsbach, A. Muller and H.-J. Wuunsche. Lidar-based 3d object perception. 2008.

[3] Larry Li. Time-of-flight camera. Texas Instrument, 2014.

[4] J. J. Leonard T. Whelan, M. Kaess and J. McDonald. Robust real-time visual odometry for dense rgb-d mapping.

[5] Antonio Ramon Jimenez, Fernando Seco Granja, Jose Carlos Prieto, and Jorge I. Guevara Rosas. Indoor pedestrian navigation using an ins/ekf framework for yaw drift reduction and a foot-mounted imu. 2010.

[6] Cliff Randell, Chris Djiallis, and Henk L. Muller. Personal position measurement using dead reckoning. In *SEMWEB*, 2003.

[7] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). *2011 IEEE International Conference on Robotics and Automation*, pages 1–4, 2011.

[8] S. Sotoodeh. Outlier detetcion in laser scanner point clouds. september 2006.

[9] Bertrand Douillard, James Patrick Underwood, Noah Kuntz, Vsevolod Vlaskine, Alastair James Quadros, Peter Morton, and Alon Frenkel. On the segmentation of 3d lidar point clouds. *2011 IEEE International Conference on Robotics and Automation*, pages 2798–2805, 2011.

[10] F. Tarsha-Kurdi, Tania Landes, and Pierre Grussenmeyer. Hough-transform and extended ransac algorithms for automatic detection of 3d building roof planes from lidar data. 2007.