BACHELOR OF SCIENCE IN COMPUTER SCIENCE

AND ENGINEERING

3D Indoor Depth Mapping Using SIFT Feature Based ICP Registration

Authors:

Nazmus Sakib (114431)

Araf Farayez (114432)

Department of Computer Science and Engineering (CSE)

Islamic University of Technology (IUT)

Organization of Islamic Cooperation (OIC)

Gazipur-1704, Bangladesh

October 2015

3D Indoor Depth Mapping Using SIFT Feature Based ICP Registration

Authors:

Nazmus Sakib (114431)

Araf Farayez (114432)

Supervisor:
Md. Hasanul Kabir, PhD
Associate Professor
Department of Computer Science and Engineering (CSE)

Co-supervisor:
Ferdous Ahmed
Lecturer
Department of Computer Science and Engineering (CSE)

A thesis submitted to the Department of Computer Science and Engineering (CSE) in partial fulfilment of the Requirements for the Award of Bachelor of Science in Computer Science and Engineering (B.Sc. Engg. (CSE))

Department of Computer Science and Engineering (CSE)

Islamic University of Technology (IUT)

Organization of Islamic Cooperation (OIC)

Gazipur-1704, Bangladesh

October 2015

Declaration of Authorship

This is to certify that the work presented in this thesis is the outcome of the analysis and investigation carried out by Nazmus Sakib and Araf Farayez under the supervision of Dr. Md. Hasanul Kabir and Ferdous Ahmed in the Department of Computer Science and Engineering (CSE), IUT, Dhaka, Bangladesh. It is also declared that neither of this thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Authors:

_____

Nazmus Sakib
Student ID – 114431

_____

Araf Farayez
Student ID – 114432

Supervisor:

_____

Dr. Md. Hasanul Kabir
Associate Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)

Co -Supervisor:

_____

Ferdous Ahmed
Lecturer
Department of Computer Science and Engineering
Islamic University of Technology (IUT)

Abstract

3D mapping is one of the challenging research areas of Computer Vision and Robotics. Mappings are often done with mobile robots. One of the best approach is the use of SLAM (Simultaneous Localization and Mapping) where a robot roams around and builds the map and also localizes its position on the currently built map. The robot needs the sense of depth of the environment and often equipped with depth sensor. It also needs Odometry data from the accelerometer or encoder set on the wheels. There are some other solutions where the map and localization is based on the features extracted and builds the map based on the depth images and use of loop closure by adjusting the error in perception. In this thesis work we used feature based mapping which does not need any Odometry data and the map can be built based on RGB image and Depth data. We used Kinect, a very popular depth sensor which is effective in this process under certain constraints. Here in our work we gathered the RGB and Depth data simultaneously and then chose suitable frames and found key features with SIFT algorithm. The extracted features in the RGB images has corresponding 3D co-ordinates. The 3d co-ordinates have been found through intensive experiments done on calibrating the Kinect sensor and finding the best fit value of parameters to give more accurate 2D spatial points corresponding to each frame captured. The matched 3D points were then applied in Iterative closest Point (ICP) algorithm as initial points to merge the two depth images. When two images are merged they were saved in separated storages and another frame is extracted and registered with the previous ones. In this methodology multiple frames were stitched in 3D spatial co-ordinates. After forming the indoor map the map was evaluated with respect to the lengths of the objects formed in the map and the shape of the map along different 2D planes by comparing their area. Different version of ICP give different result in time complexity and accuracy. It was found IRLS (Iteratively Reweighted Least Square) ICP gave better results. The methodology we proposed can be implemented at faster time and fewer constraints and mapping do not depend on the movement noise of the robot. So the whole process is simpler and robust and can be used in indoor mapping, object detection and security purposes.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Building rich 3D maps of environments is an important task for mobile robotics, with applications in navigation, manipulation, semantic mapping and telepresence. So far, to reconstruct 3D model with fine geometric accuracy and rich visual information is the goal which is pursued unremittingly by researchers. Relative technologies have been developed vigorously by some communities including Photogrammetric, Computer Vision and Robotics. The applications of research results in the Digital City, Robot Navigation and other fields have demonstrated wide application prospect. However, due to the high complexity of real scene, to achieve this goal still seems far beyond our reach. In practice geometric fine 3D indoor models are still rarely used. On the other hand, with the development of Digital Earth technology, urban 3D modeling technology has been mature to some extent. But indoor models of large public places have not entered public's view yet. However, because of the importance of indoor navigation, it is imperative for 3D reconstruction to shift from outdoor to indoor context. Now there have been some kinds of indoor3D reconstruction methods.

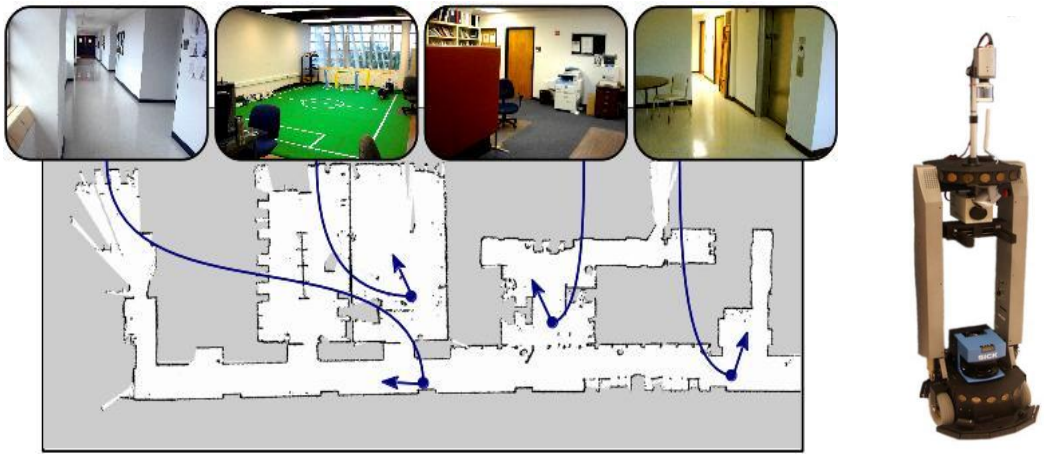

Figure 1.1: SLAM with mobile robot

Figure 1.2: Demonstration of 3D maps

Simultaneous localization and mapping (SLAM) is an algorithm that allows a mobile robot to form a map of an unknown environment and locate itself within this map. SLAM can be implemented with Mobile Robot .It's also known in the literature as Concurrent Localization and Mapping (CLM).We can place a robot in an unknown location in an unknown environment and have the robot incrementally build a map of this environment while simultaneously using this map to compute vehicle location. This process is often called Robot Pose Tracking in many of the papers [1]. The benefit to fusing a mapping conjunction with other means of recording movement is that it allows the algorithm to correct positioning estimates and detect and recover from errors regarding sensor data. It is also concerned with, on a secondary basis in this research area, navigation using the map. Applications include ocean surveying, oil pipeline inspection, mine exploration, crime scene investigation and military operations

1.2 Problem Statement

Simultaneous Localization and Mapping is a long process which includes use of lots algorithms to implement different steps [1]. The steps can be divided into 2 major parts: **Localization** and **Mapping**. **Localization** includes: Moving the robot, observing the environment, Detecting Landmarks, Associate new landmarks with previously observed landmarks, the robot's motion from the distance between associated landmarks, Correct the new landmarks' position for the estimated robot's motion, Add corrected landmarks to the map and repeat. **Mapping** part includes extracting Key

features, registration of 3d cloud, Loop Closure Detection to align the different part of map segments.

Our focus is on 3d mapping. We developed an algorithm which makes 3d map of indoor. At first it takes **Depth** and **RGB** data from Kinect sensor. It preprocess the data and convert raw data into x, y, z spatial co-ordinates. Next similar features are extracted using SIFT algorithm and Homography using RANSAC. Then the cloud points are registered using a version of ICP [3] next the elements in the map is evaluated to check its feasibility to be used in SLAM.

1.3 Research Challenges

In the Simultaneous Localization and Mapping (SLAM) problem a mobile robot has to be able to autonomously explore the environment with its on-board sensors, gain knowledge about it, interpret the scene, build an appropriate map and localize itself relative to this map

- Map Representation
  - Sparse set of landmarks (beacons, salient visual features)
  - Dense representation (stereo, dense depth data)

In [4], the convergence of a filter which estimates the robot configuration and the absolute location of the landmarks by adopting a Kalman filter (absolute map filter, AMF), is theoretically proven. However, the proof is based on a perfect statistical knowledge of the error of each sensor and also on the hypothesis of a linear observation

- Unstructured 3d Environment :
  - The 3d environment around us is not smooth and uniform.
  - The reflecting surface should be a solid in order to reflect IR data for depth measurement.
- Data association/Map convergence:
  - Adopt an optimal filter (accordingly to the dynamics and the observation) [5]
  - Use the best statistical model to characterize the error of the adopted sensor readings (it is better avoiding the use of sensors whose error is roughly known (statistically) in the estimation process)
- Dynamic Environments

  - World is not static
  - Doors / open close
  - Human / material motion
  - Dynamic object detection
- Informed Sensing
  - Can sensor parameter be estimated
  - Is Auto calibration of sensor possible

After the first precise mathematical definition of the stochastic map [6] early experiments ([7], [8]), have shown the quality of fully metric simultaneous localization and map building: the resulting environment model permits highly precise localization that is only bounded by the quality of the sensor data

- Erroneous Odometry data : For automatic mapping this makes the global consistency of the map difficult to maintain in large environments where the drift in the Odometry becomes too important
- Computational Requirement : According to Wikipedia "Researchers and experts in artificial intelligence have struggled to solve the SLAM problem in practical settings: that is, it required a great deal of computational power to sense a sizable area and process the resulting data to both map and localize." The complexities can be divided into 2 parts
  - Time :
  - Complexity

A 2008 review [9] of the topic summarized: "SLAM is one of the fundamental challenges of robotics. But it seems that almost all the current approaches cannot perform consistent maps for large areas, mainly due to the increase of the computational cost and memory."

1.4 Thesis Objective

Autonomous navigation and mapping is very popular nowadays. Google car is one such example. SLAM has become a prominent methodology to solve the autonomous movement and map generation. The related problems of data association and computational complexity are among the problems yet to be fully resolved. For example

the identification of multiple confusable landmarks. A significant recent advance in the feature-based SLAM literature involved the re-examination of the probabilistic foundation for Simultaneous Localization and Mapping (SLAM) where it was posed in terms of multi-object Bayesian filtering with random finite sets. It provide superior performance to leading feature-based SLAM algorithms in challenging measurement scenarios with high false alarm rates and high missed detection rates without the need for data association.

Our thesis aimed at proposing a method which can be implemented as part of the steps of SLAM with

- Feature based localization instead of Odometry data
- No requirement of data associatively
- Calculation of angle from static pose with feature
- Justification of use of combination of SIFT-ICP to extract feature and depth map merging.

1.5 Thesis Contributions

Already we have discussed about the research challenges in this field and our objective. And we have undertaken different type of procedures and but succeeded at few of them.-

- We generated partial map with static translation and at heuristic speed of rotation.
- We computed angle of rotation based on RGB based SIFT between 2 images and computed the angle of variation between the images.
- Plotting the depth points in 2D plane at an angle derived in the above steps we get a 2D map of the room.
- We have investigated several existing methods for 3d map generation for SLAM identifying their weakness and solved those with our method.
- In the comparison step we evaluated the performance of depth point clouds of different ICP and found the best ICP which ensures good match and helps discarding bad match between cloud points.

- For performance analysis, we implemented our method and figured out the performance using real time dataset and measured the accuracy.
- Comparative analysis of our method with others have also been done for getting the comparative performance with respect to other.

1.6 Organization of thesis

The rest of thesis will be organized as follows:

In Chapter 2 we present the literature review of existing methods and their performance, strong point and weak points. In Chapter 3, we propose our verification method. There we discuss about the overall idea of our proposal and step by step implementation process. In Chapter 4, experimental set-up, experimental result and performance analysis of our method with various challenge is shown. Besides with other methods a comparative analysis is also shown. Lastly, in Chapter 5, we conclude the thesis contribution and shows the future scopes for further developing the proposed method.

Chapter 2

Literature Review

2.1 3D Mapping

We are familiarized ourselves with methods for 3D registration of images captured with cheap RGB-D sensors such as the Kinect. A Kinect mounted on a moving robot could replace both its RGB camera and its range finder. Another application would be affordable 3D reconstructions of objects or indoor scenes. For these applications we need to register consecutive frames to the same global coordinate space. In effect, we need to find the transformation that the camera made in between the captured frames.

2.2 Simultaneous localization and mapping

Simultaneous localization and mapping (SLAM) is an algorithm that allows a mobile robot to form a map of an unknown environment and locate itself within this map [1]. SLAM can be implemented with Mobile Robot. It's also known in the literature as Concurrent Localization and Mapping (CLM).We can place a robot in an unknown location in an unknown environment and have the robot incrementally build a map of this environment while simultaneously using this map to compute vehicle location. This process is often called Robot Pose Tracking in many of the papers. The benefit to fusing a mapping conjunction with other means of recording movement is that it allows the algorithm to correct positioning estimates and detect and recover from errors regarding sensor data. It is also concerned with, on a secondary basis in this research area, navigation using the map.

The general steps of SLAM are as follows [25]:

1. Move robot
2. Observe environment
3. Detect landmarks
4. Associate new landmarks with previously observed landmarks
5. Estimate the robot's motion from the distance between associated landmarks

6. Correct the new landmarks' position for the estimated robot's motion
7. Add corrected landmarks to the map and repeat

In order to implement such an algorithm three pieces of hardware are necessary: a mobile robot, a means to observe the environment and a computer. Of these essential pieces of hardware, the means to observe the environment is the most specific to SLAM. Observing the robot's environment can be implemented in several ways. Examples are sonar, monocular cameras, and laser range finders. Laser range finders provide the most accurate observations but have historically been the most expensive until the advent of the Microsoft Kinect. With the Microsoft Kinect it is now possible to get an array of depth data (640 x 480) that is accurate to the millimeter.

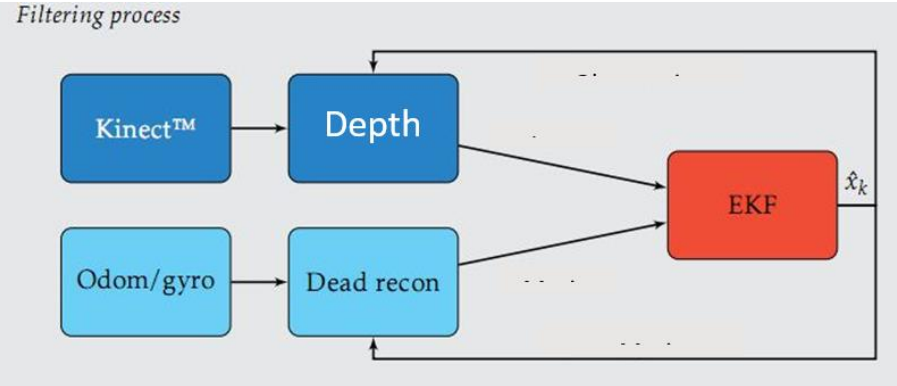

Figure 2.1: Odometry based SLAM

2.2.1 Landmark Extraction:

Landmark extraction may be achieved via a variety of methods. This is a critical part of the algorithm as it is used as an input to the EKF algorithm [30]. The method used is generally determined by the type of landmark to be extracted in the environment and the type of sensor used to detect these.

2.2.1.1 Spike Landmarks:

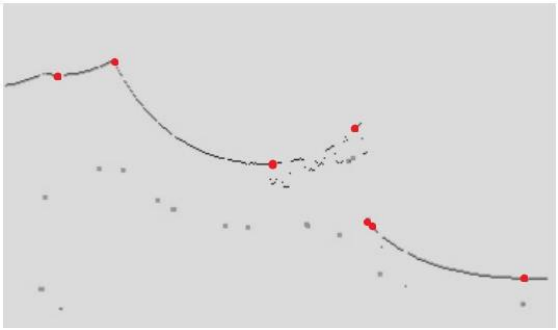

Figure 2.2: Spike Landmarks

Spike landmark extraction is a simple algorithm concerned with landmark extraction from laser or depth range data. In scanning systems where scans yield multiple values within a certain angle of scanning, this algorithm tries to find extreme differences in the values read by the scanners [26]. This happens when the distance measured at one angle is largely different to the distance measured at the next angle. This indicates that there is a geometric change between the angles, which can be interpreted as a landmark.

Equation:

$$m = \frac{\Delta f(a)}{\Delta a} = \frac{f(a+h) - f(a)}{(a+h) - (a)} = \frac{f(a+h) - f(a)}{h}$$

Due to the method in which this extraction algorithm works, it has an inherent flaw. It requires that the environment be highly populated with landmarks. As it requires a high degree of change between ranges scans, which means that it will not work in an environment with either zero or smooth edges [10].

2.2.1.2 RANSAC:

Random Sampling Consensus is a method of landmark extraction that works by trying to identify lines from range scans. This can be particularly useful in Indoor environments as walls are easily identified by this method [26]. RANSAC associates range readings with lines and is known as a form of scan matching. This is done by taking a random sample of range values from an unassociated scan and then using a least squares algorithm to try and find a line that best fits these readings. Once the line has been found, the algorithm will count the number of range values that match that line and if this count is above a consensus threshold, these values are then associated with a line [27].

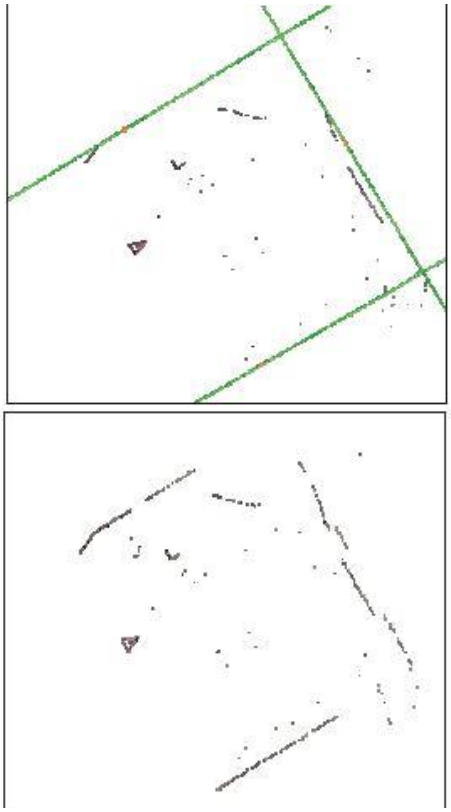

Figure 2.3: RANSAC

Algorithm:

While

- there are still unassociated laser readings,
- and the number of readings is larger than the consensus,
- and we have done less than N trials.

Do

- Select a random laser data reading.
- Randomly sample S data readings within D degrees of this laser data reading (for example, choose 5 sample readings that lie within 10 degrees of the randomly selected laser data reading).
- Using these S samples and the original reading calculate a least squares best fit line.
- Determine how many laser data readings lie within X centimeters of this best fit line.
- If the number of laser data readings on the line is above some consensus C do the following:
    - Calculate new least squares best fit line based on all the laser readings determined to lie on the old best fit line.
    - Add this best fit line to the lines we have extracted.
    - Remove the number of readings lying on the line from the total set of unassociated readings.

End Loop

This algorithm can thus be tuned based on the following parameters:

N – Max number of times to attempt to find lines.

S – Number of samples to compute initial line.

D – Degrees from initial reading to sample from.

X – Max distance a reading may be from line to get associated to line.

C – Number of points that must lie on a line for it to be taken as a line

2.2.2 Landmark Association:

The next step of SLAM is to associate these newly found landmarks with previously observed landmarks that are believed to be the same. We have also referred to this as re-observing landmarks [26].

- Might not re-observe landmarks every time step.
- Might observe something as being a landmark but fail to ever see it again.
- Might wrongly associate a landmark to a previously seen landmark.

As stated in the landmarks chapter it should be easy to re-observe landmarks. As such the first two cases above are not acceptable for a landmark. In other words they are bad landmarks. Even if you have a very good landmark extraction algorithm you may run into these so it is best to define a suitable data-association policy to minimize this.

$$r = k * dX + m * d\theta + l$$

The value of k, m will vary from robot to robot.

2.3 Localization:

In the SLAM process Projective geometry is use instead of Euclidian geometry. Cameras generate a projected image of the world. Euclidian geometry is suboptimal to describe the central projection and the mathematics becomes complex. Whereas Projective geometry uses homogenous coordinates by which points at infinity can be represented using finite coordinates.

2.3.1 Recursive Bayes Filter:

To find the best state estimate, different types of Recursive Bayes Filters are used during localization. Recursive Bayes Filter is derived using a belief

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t})$$

Where,

$x_t = state\ in\ time\ t$

$z_t = observation\ at\ time\ t$

$u_t = motion\ control\ at\ time\ t$

Here Markov assumption is used to state that the current state estimate depends only on the previous state estimate and all previous states can be ignored.

The resultant equation has two parts:

- Prediction step:

$$\overline{bel}(x_t) = \int p(x_t|u_t, x_{t-1})\ bel(x_{t-1})\ dx_{t-1}$$

- Correction step:

$$bel(x_t) = \eta\ p(z_t|x_t)\ \overline{bel}(x_t)$$

2.3.2 Motion Models:

Mainly two types of motion models are widely used in designing robots:

1. Odometry-based
2. Velocity-based

2.3.2.1 Standard Odometry-based Model:

In this model, motion controls are executed with the help of wheel encoders and/or Gyro sensors.

Robot moves from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$

Odometry information $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$

$$\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$$
$$\delta_{rot1} = atan2(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$$
$$\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$$

In case of systems with Gaussian noise, $u = \mathcal{N}(0, \Sigma)$.

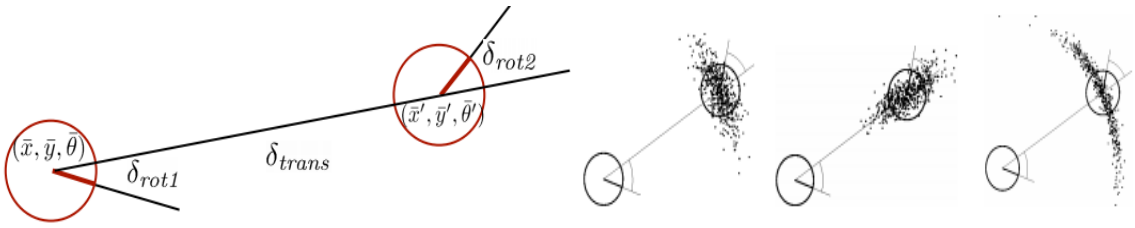

Figure 2.4: Demonstration of Odometry equation and Gaussian noise

2.3.2.2 Velocity-based Model:

In this model, motion controls are executed with the help of wheel encoders and/or Gyro sensors.

Robot moves from (x, y, θ) to (x', y', θ')

Odometry information$u = (v, w)$

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v}{\omega}\sin\theta + \frac{v}{\omega}\sin(\theta + \omega\Delta t) \\ -\frac{v}{\omega}\sin\theta + \frac{v}{\omega}\sin(\theta + \omega\Delta t) \\ \omega\Delta t \end{pmatrix}$$

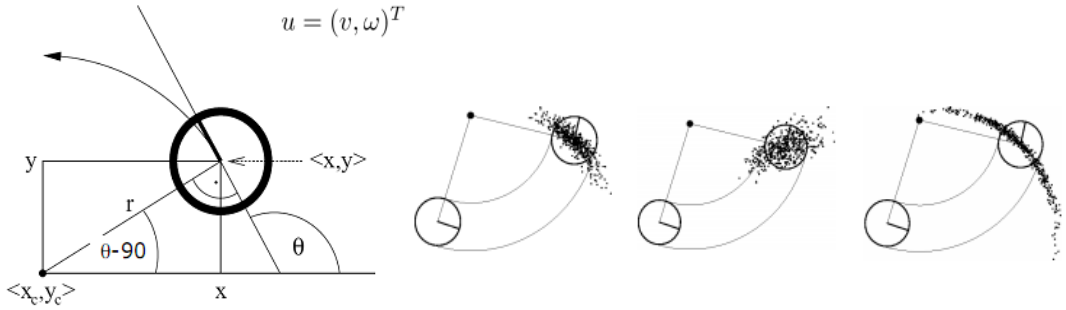

Figure 2.5: Gaussian noise generated during rotation

2.3.3 Sensor Models:

Mostly used sensors in SLAM are:

1. Laser scanner
2. Kinect sensor
3. Range-Bearing sensor

The different types of sensor model are:

1. Beam-Endpoint model
   - Beam of laser is transmitted and the endpoint is recorded as the beacon
2. Ray-cast model
   - First obstacle along the line of sight is recorded

2.3.4 Kalman Filter

Kalman filter estimates states and observations in a linear system. The motion and observation models are defined by,

- Motion model:

  $x_t = A_t x_{t-1} + B_t u_t - \epsilon_t$
- Observation model:

  $z_t = C_t x_t + \delta_t$

Here,

$A_t = State\ transition\ matrix \quad (n * n)$

$B_t = Maps\ control\ input\ into\ state\ vector \quad (N * l)$

$C_t = Maps\ state\ vector\ to\ an\ observation\ z_t (k * n)$

Under Gaussian noise of the environment the models are defined by,

- Motion model:

$$p(x_t \mid u_t, x_{t-1}) = \det(2\pi R_t)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t)\right)$$

- Observation model:

$$p(z_t \mid x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\right)$$

Here,

$\mu_t = State\ estimate$

$\Sigma_t = State\ and\ Landmark\ covariance$

$Q_t = Measurement\ noise\ covariance$

$R_t = Process\ noise\ covariance$

But the real world is non-linear with angular motion and measurements. Kalman filter produces inaccurate estimations when calculated with a non-linear and non-Gaussian distribution. To solve this problem Extended Kalman Filter is introduced.

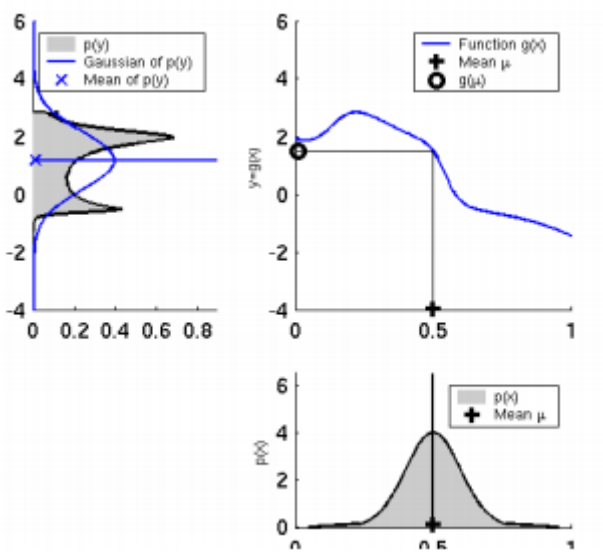

Figure 2.6: EKF under Gaussian noise

2.3.5 Extended Kalman Filter:

This filter uses first order Taylor expansion to linearize Kalman Filter. The linearization function creates a tangential curve of the non-Gaussian distribution.

Jacobian of Prediction

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \underbrace{\frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}}_{=: G_t} (x_{t-1} - \mu_{t-1})$$

Jacobian of Correction

$$h(x_t) \approx h(\bar{\mu}_t) + \underbrace{\frac{\partial h(\bar{\mu}_t)}{\partial x_t}}_{=: H_t} (x_t - \bar{\mu}_t)$$

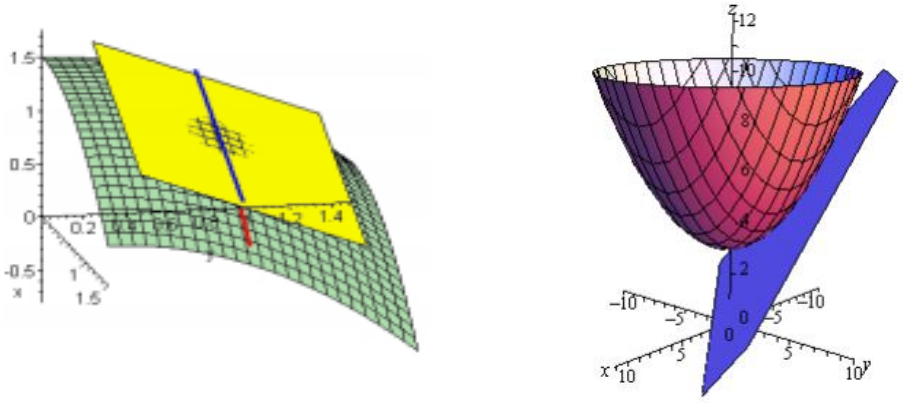

Figure 2.7: Jacobian model, courtesy Wikipedia

The Jacobian Matrix is a non-square matrix of dimension $m * n$ to map the derivative of the linearization function to the state vector.

Given, a vector-valued function

$$g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{pmatrix}$$

The Jacobian matrix is defined as,

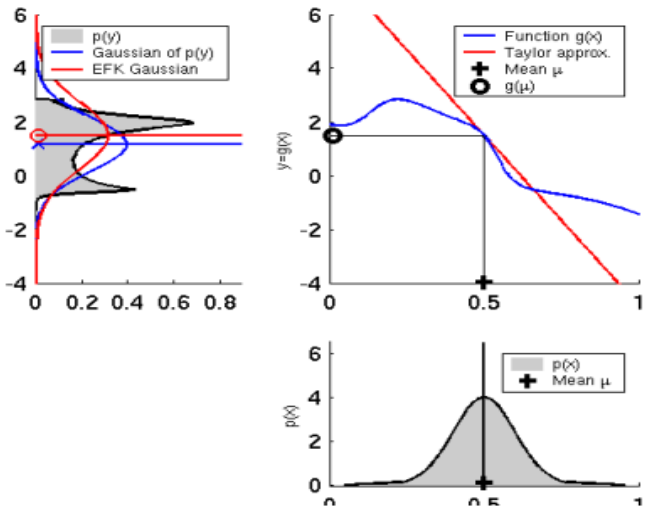

Figure 2.8: Comparison between Taylor and EKF

$$G_x = \begin{pmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial g_m}{\partial x_1} & \frac{\partial g_m}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_n} \end{pmatrix}$$

Finally the linearized model leads to,

- Motion model

$$p(x_t \mid u_t, x_{t-1}) \approx \det(2\pi R_t)^{-\frac{1}{2}} \exp\Big(-\frac{1}{2} (x_t - g(u_t, \mu_{t-1}) - G_t (x_{t-1} - \mu_{t-1}))^T R_t^{-1} (x_t - \underbrace{g(u_t, \mu_{t-1}) - G_t (x_{t-1} - \mu_{t-1})}_{\text{linearized model}}) \Big)$$

- Observation model:

$$p(z_t \mid x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp\Big(-\frac{1}{2} (z_t - h(\bar{\mu}_t) - H_t (x_t - \bar{\mu}_t))^T Q_t^{-1} (z_t - \underbrace{h(\bar{\mu}_t) - H_t (x_t - \bar{\mu}_t)}_{\text{linearized model}}) \Big)$$

2.4 Feature Based Mapping

2.4.1 Mapping Using SURF and Loop Closure

In this type approach, the underlying structure of the map is a graph with nodes and links. The nodes save Odometry poses for each location in the map. The nodes also contain visualization information like laser scans, RGB images, depth images and visual words [15] used for loop closure detection. The links store rigid geometrical transformations between nodes. There are two types of links: neighbor and loop closure. Neighbor links are added between the current and the previous nodes with their Odometry transformation. Loop closure links are added when a loop closure detection is found between the current node and one from the same or previous maps. The approach [17] involves combining two algorithms, loop closure detection [16] and graph optimization [18], through a memory management process [16] that limits the number of nodes available from the graph for loop closure detection and graph optimization, so that they always satisfy online requirements.

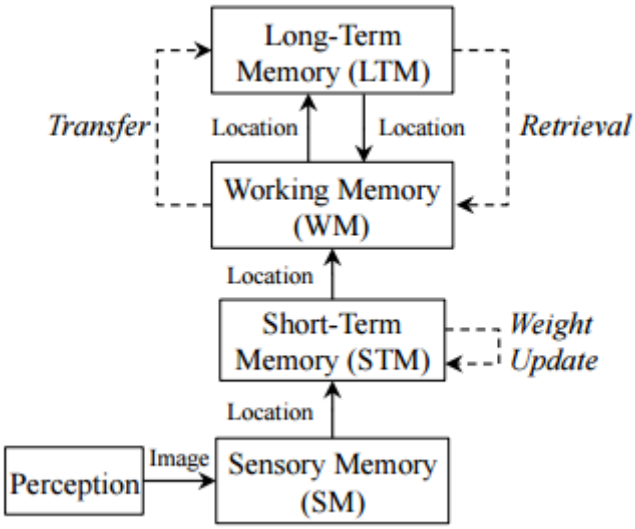

Figure 2.9: Memory management in RTAB

The approach is based on a memory management method, which limits the number of locations used for loop closure detection so that the computation time remains under real-time constraints. The idea consists of keeping the most recent and frequently observed locations in a Working Memory (WM) used for loop closure detection, and transferring the others into a Long-Term Memory (LTM). When a match is found between the current location and one stored in WM, associated locations stored in LTM can be updated and remembered for additional loop closure detections. The pseudo code for mapping is stated below

Algorithm 1 RTAB-Map

$time \leftarrow$ TIMENOW() ▷ TIMENOW() returns current time
$I_t \leftarrow$ acquired image
$L_t \leftarrow$ LOCATIONCREATION(I_t)
if z_t (of L_t) is a bad signature (using T_{bad}) **then**
    Delete L_t
else
    Insert L_t into STM, adding a neighbor link with L_{t-1}
    Weight Update of L_t in STM (using $T_{\text{similarity}}$)
    if STM's size reached its limit (T_{STM}) **then**
        Move oldest location of STM to WM
    end if
    $\boldsymbol{p}(S_t|L^t) \leftarrow$Bayesian Filter Update in WM with L_t
    Loop Closure Hypothesis Selection ($S_t = i$)
    if $S_t = i$ is accepted (using T_{loop}) **then**
        Add loop closure link between L_t and L_i
    end if
    Join $trash$'s thread ▷ Thread started in TRANSFER()
    RETRIEVAL(L_i) ▷ LTM $\rightarrow$ WM
    $pTime \leftarrow$ TIMENOW() $- time$ ▷ Processing time
    if $pTime > T_{\text{time}}$ **then**
        TRANSFER() ▷ WM $\rightarrow$ LTM
    end if
end if

2.4.2 3D Indoor Mapping With SIFT [19]

Many scanning are performed from different station to obtain occlusion free 3D object model. Laser scanner data (point cloud) is in local coordinate system center of which is the laser scanner. In this case, all point clouds must be registered into common coordinate system to visualizing 3D model of the object. Generally first point cloud was selected as a reference and the others are registered into its coordinate system. Many methods have been developed for registration of point clouds. The most popular method is iterative closest point (ICP) method (Chen and Medioni, 1992; Besl and McKay, 1992). Another one is least square 3D image matching method (Guen and Akca, 2005). In addition, registration can be performed with object details extracted from point clouds (Deveau et al., 2004; Briese and Pfeifer, 2008). The registration methods of point clouds were investigated with details in Salvi et al. (2007).

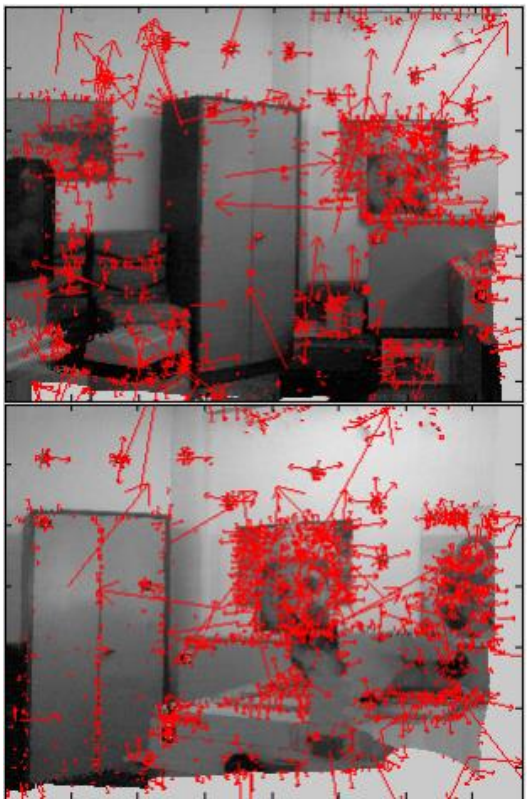

Figure 2.10: SIFT key points with translation and rotation

Laser scans have been made as overlapping beginning of the first point cloud. In that case, to combine all point clods, the registration must be performed relation to reference point clouds. The registration parameters have been computed by laser scanning points with different techniques in overlapping area. In this study, pair-wise registration was performed by intensity image created from laser scanner data. Key points from intensity images were extracted and matched by SIFT method. After outlier points were detected by statistical method, point cloud coordinates for each conjugate key points were determined using range image data. The registration parameters were computed with these corresponding points and residuals on SIFT points were computed. Then the results of the method were compared with the ICP.

The key points are summarized as follows:

- Indoor mapping with SIFT
- Use only SIFT feature to merge depth image
- No mobile robot/ Odometry
- Autonomous registration
- Result compared with ICP

The algorithm procedure can be stated as follows:

- Laser scans have overlapping beginning of the first point cloud
- Registration must be performed relation to reference point clouds.
- Pair-wise registration was performed
- Intensity image created from laser scanner data.
- Intensity images were extracted and matched by SIFT method

Chapter 3

Proposed Method: 3D Indoor Depth Mapping Using SIFT Feature Based ICP Registration

3.1 Skeleton of Proposed Method

The following figure shows the overall steps of the proposed method:

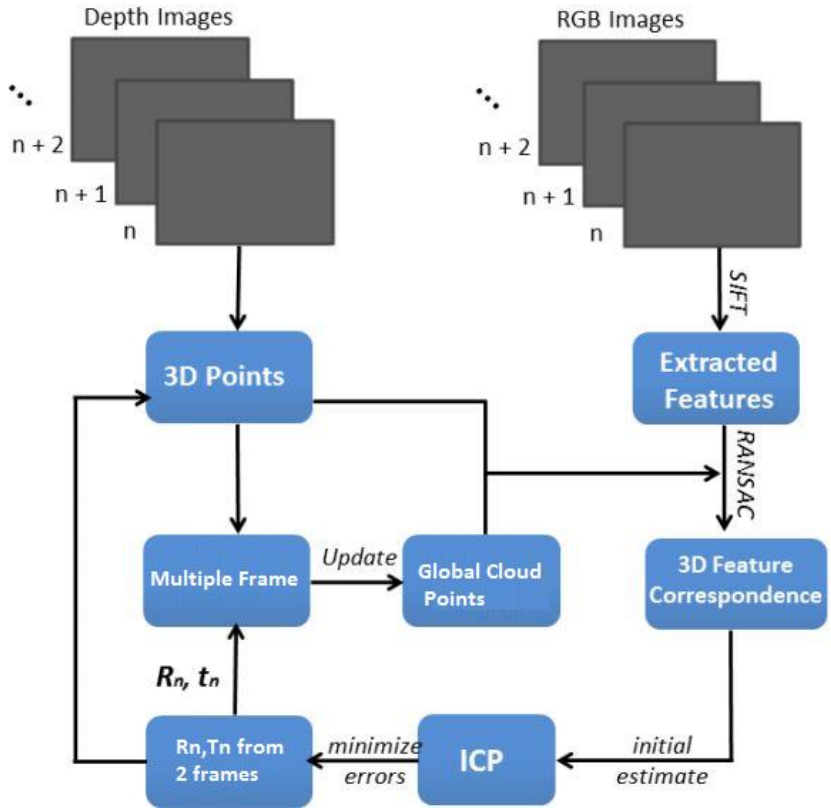

Figure 3.1: The basic steps of proposed method

At first we take the Depth image and the RGB image simultaneously. The RGB images are run through SIFT algorithm and they give us extracted features. With the use of RANSAC the homography between the points are found. Then the matched points are mapped back to their 3d co-ordinates and sent to ICP algorithm. Since ICP algorithm works on 3d geometric model on some initial random guess. But when the matched points are sent for initial estimate it gives better performance [12]. The points also carry significant common geometric pattern. The algorithm returns 2

matrices and the merged data points. The matrices are Rn and Tn which are rotation matrix and translation matrix. Therefore if the data points are Data1, Data2 then the projection of Data2 on Data1 is,

Merged Data = Rn*Data1 + Tn

Then data points are saved in the storage and the algorithm usually runs for every 2 frames and merging the depth images on the basis on 3d cloud point registration.

3.2 Acquiring Depth Data

At first the data is taken from the Kinect camera and data consist of 2 parts one depth and another image. Since RGB data minimum rate in MatLab API is 640*480 so for computational purpose they are converted into 320*240 same as depth image and saved in database. Both depth data and RGB data comes as video input frame and are saved in double variable format. So for each of 320*240 we get pixel wise depth value returned by the Kinect.

3.3 Pre-Processing

The pre-processing is an essential part of the Mapping without appropriate RGB and Depth data feature based SLAM is almost impossible. The purpose is to make the world co-ordinate ready for feature extraction and registration. Also conversion of raw data to cm/m gives us better perspective and easy calculation of error and accuracy.

3.3.1 Conversion from 3D to 2D data

Procedure:

When depth images taken they have certain resolution of [X Y] and Z is the depth value for each correspondent [X Y]. So the pixel co-ordinate has to be converted into linear length with respect to the depth value. So mid of the depth image is taken as (0,0) then the left side has negative length in X axis and right side has positive and for Y the upper part is positive and lower part is negative. The equation for conversion is as follows:

- Raw Data converted to depth matrix, Z'
- X and Y matrices calculated

$$X_{i,j} = \left(j - \frac{w}{2}\right) * S * (u_t + MinD)$$

$$Y_{i,j} = \left(i - \frac{h}{2}\right) * S * (u_t + MinD)$$

$$w = width$$

$$h = height$$

$$MinD = Minimum\ Active\ Distance\ of\ Kinect$$

$$S = Scale\ Factor$$

With the help of above formula the 3d co-ordinates of depth data under homographic plane is found. Here the value of S=.0021 and MinD=10 are taken with trial and error basis.

3.3.2 Conversion to Reduced Cloud Point

During feature extraction or cloud point conversion the data set is reduced to lower size to faster computation. Careful measures were taken so that necessary information is not lost. The reduced cloud point was achieved in 2 formats.

1. Resizing the raw depth image
2. Sampling: the cloud point was taken as sampled.

The MatLab scatter plot is very slow to show the 3d plot data in real time. So sampling the data showed the 3d frame properly. But lots of points remain vacant. While the reduced size shows continuous data but the smaller version of it.

3.4 Feature Extraction of the Key-points

In this step the feature descriptor is calculated for each of the key-point found in the previous step. We used SIFT (Scale-Invariant Feature Transform) for calculating the descriptor so that descriptors are highly distinctive, invariant as possible to variations such as changes in viewpoint and illumination [13]. Before calculating the descriptor for each of the key-point *Gaussian* weighting around center is done in order to provide some weight that is inversely proportional to the distance from the key-point. Next a 16X16 neighborhood around the key-point is taken. It is divided into 16 sub-blocks of 4X4 size. For each sub-block, 8 bin orientation histogram is created. So a total of 128 bin values are available. The overall step may be shown as the following image:

The first image shows that the Gaussian has been applied around the point of interest and rest two images show how the 128 bins are found as we said in the above. Finally the 128 bins value is represented in vector as key-point descriptor for the usage in next step of our proposed method.

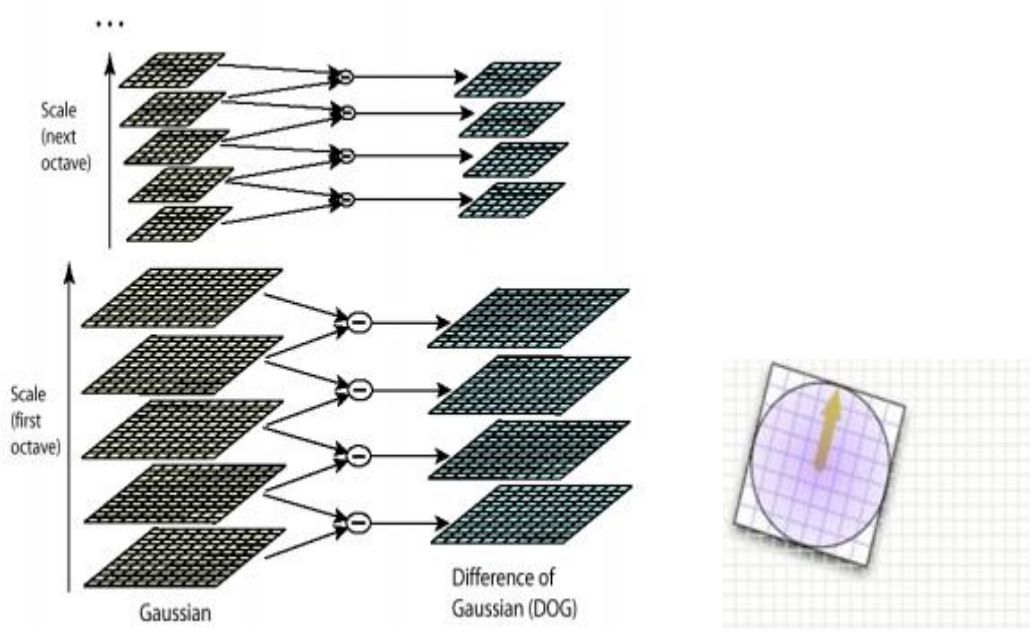

Figure 3.2: Gaussian Filter

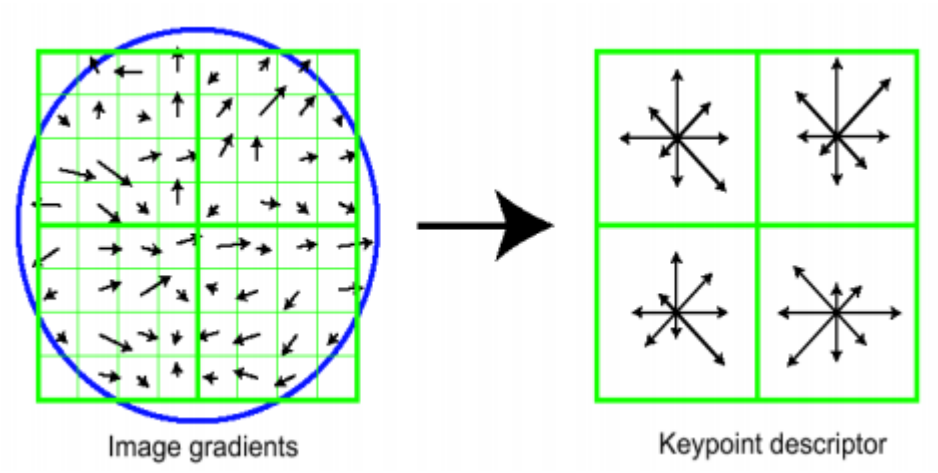

Figure 3.3: Feature (Descriptor) extraction

Given a Gaussian-blurred image

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

where

$$G(x, y, \sigma) = 1/(2\pi\sigma^2) \exp^{-(x^2+y^2)/\sigma^2}$$

is a variable scale Gaussian, the result of convolving an image with a difference-of-Gaussian filter

$$G(x, y, k\sigma) - G(x, y, \sigma)$$

is given by

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \qquad (1)$$

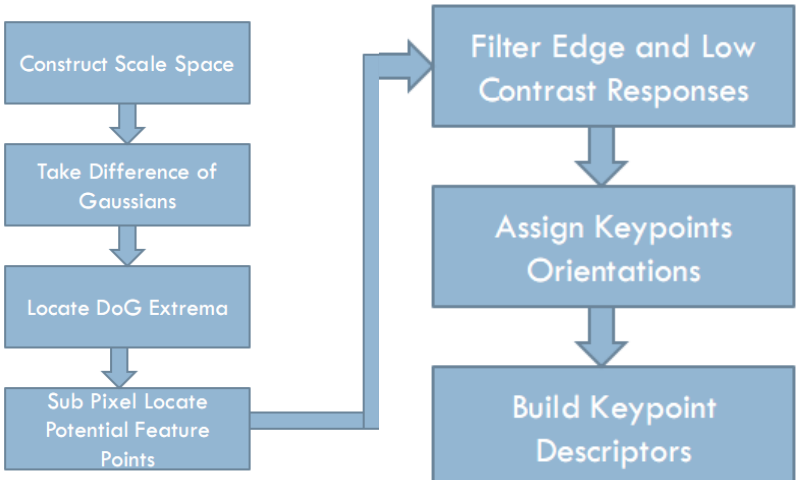

Figure 3.4: Steps of SIFT algorithm

A summarization of our SIFT algorithm is as follows:

1. Load Image
2. Resample Image to double size
3. for each octave
    - create Gaussian blur intervals
    - create difference of Gaussian intervals
    - compute edges for each interval
4. end for
5. Search each object for stable extrema
6. Create key points at dominant orientations of extrema
7. for each key point
    - rotate sample grid to key point orientation
    - sample region and create descriptor
8. end for
9. optionally save pyramid images
10. save output images
11. save descriptors

After running the algorithm, these features are matched to the SIFT feature database. Each of key point specifies 4 parameters: 2D location, scale, and orientation. To increase recognition robustness Hough transform can be used. Next each key point votes for consistent with the key points. Locations in the Hough accumulator that accumulate at least 3 votes are selected as candidate object/pose matches. A verification step matches the training image for the hypothesized object/pose to the image using a least-squares to the hypothesized location

3.5 Iterative Closest Point

Iterative Closest Point (ICP) is an algorithm employed to minimize the difference between two clouds of points [3]. ICP is often used to reconstruct 2D or 3D surfaces from different scans, to localize robots and achieve optimal path planning (especially when wheel Odometry is unreliable due to slippery terrain).

In the algorithm, one point cloud, the reference, or target, is kept fixed, while the other one, the source, is transformed to best match the reference. The algorithm iteratively revises the transformation (combination of translation and rotation) needed to minimize the distance from the source to the reference point cloud.

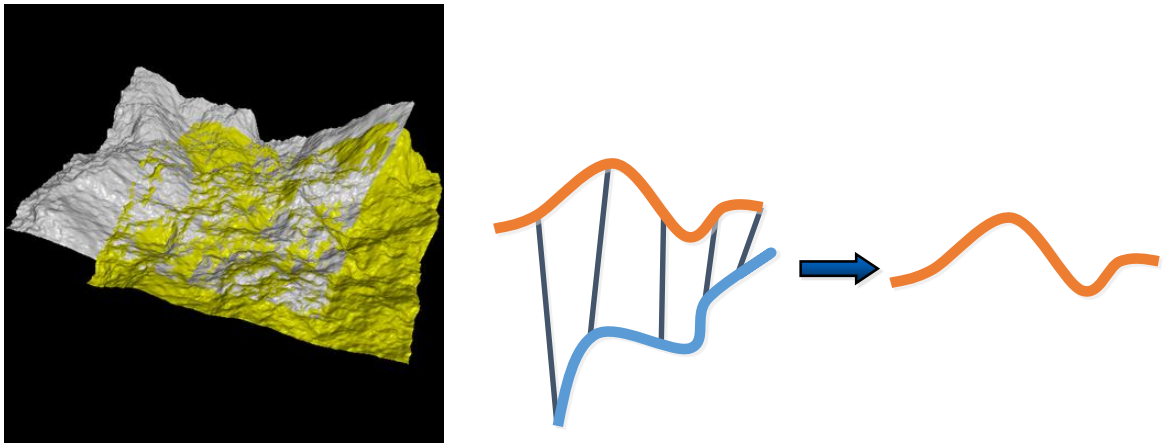

Figure 3.5: Merging 3D cloud points

Inputs: reference and source point clouds, initial estimation of the transformation to align the source to the reference (optional), criteria for stopping the iterations.

Output: refined transformation.

Essentially, the algorithm steps are:

1. Select random/initial points
2. For each point in the source point cloud, find the closest point in the reference point cloud with Euclidean or K-d tree
3. Estimate the combination of rotation and translation using a mean squared error cost function that will best align each source point to its match found in the previous step. The cost function is

$$E \coloneqq \sum_{i} (R_{p_i} + t - q_i)^2$$

4. Transform the source points using the obtained transformation.
5. Iterate (re-associate the points, and so on).

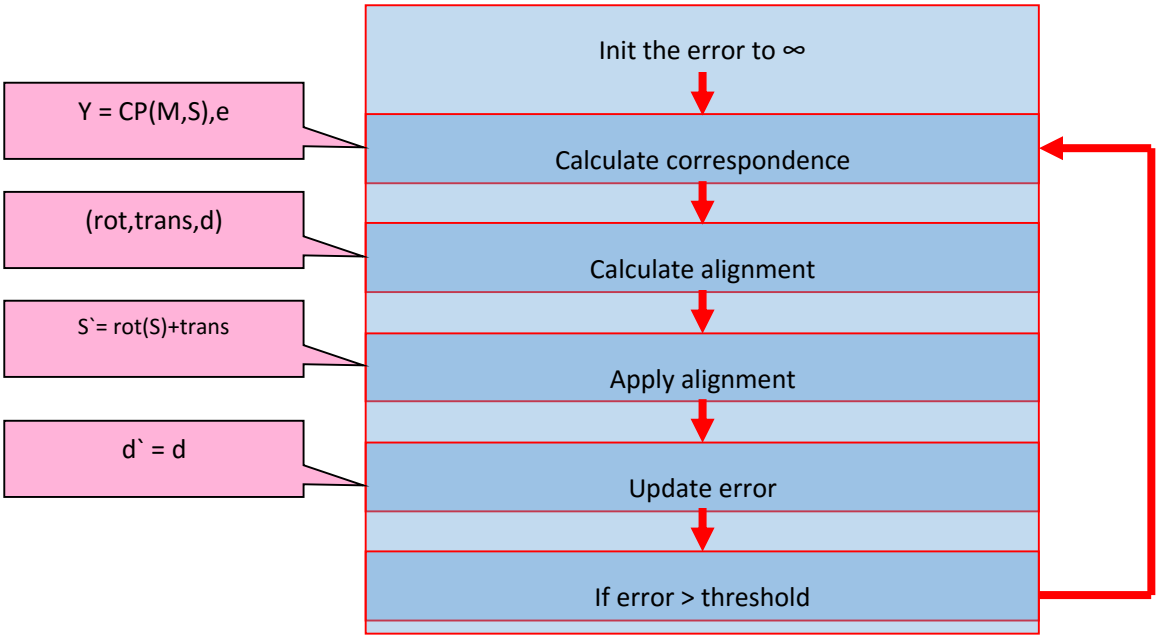

Figure 3.6: ICP Steps

Finally the ICP returns a transformation matrix and translation vector with which 2 depth images can be registered.

3.5.1 Iteratively Reweighted Least Squares ICP

The algorithm that we analyze is a version of the ICP algorithm that uses iteratively re-weighed least squares (IRLS), which has previously been touched upon in e.g. [20]. IRLS is widely used in robust linear regression. From a computational point of view this method is very similar to the method of weighting of point-pairs, but the weights are obtained from an M-estimation criterion. As calling of the residuals is needed in order to define what a large residual is. In each iteration we are determining a scale parameter for fixed residuals. An update of the rigid body transformation is then found while the scale parameter is kept unchanged. The standard deviation of the error of the "good" data that represents what it is supposed to represent pretty well is as summed to be known. It should be given from knowledge about the acquirement of the data or by some other assumptions.

A weight function w is defined as

$$w(r) = \begin{cases} \dfrac{\psi(r)}{r} & \text{if } r \neq 0, \\ \lim\limits_{r \to 0} \dfrac{\psi(r)}{r} = \psi'(0) & \text{if } r = 0. \end{cases}$$

Chapter 4

Experimental Result and Performance Analysis

4.1 Experimental Setup

We have implemented the proposed model and compared with a number of existing approaches. All those implementation and simulation have been done using MATLAB R2013a on a personal computer of 2.1 GHz processors with 8 GB main memory. We have taken the corridor depth and RGB image of our dormitory. We have taken different data with and without translation of camera. Since we have omitted the Odometry data for localization we used a Tripod to capture data.

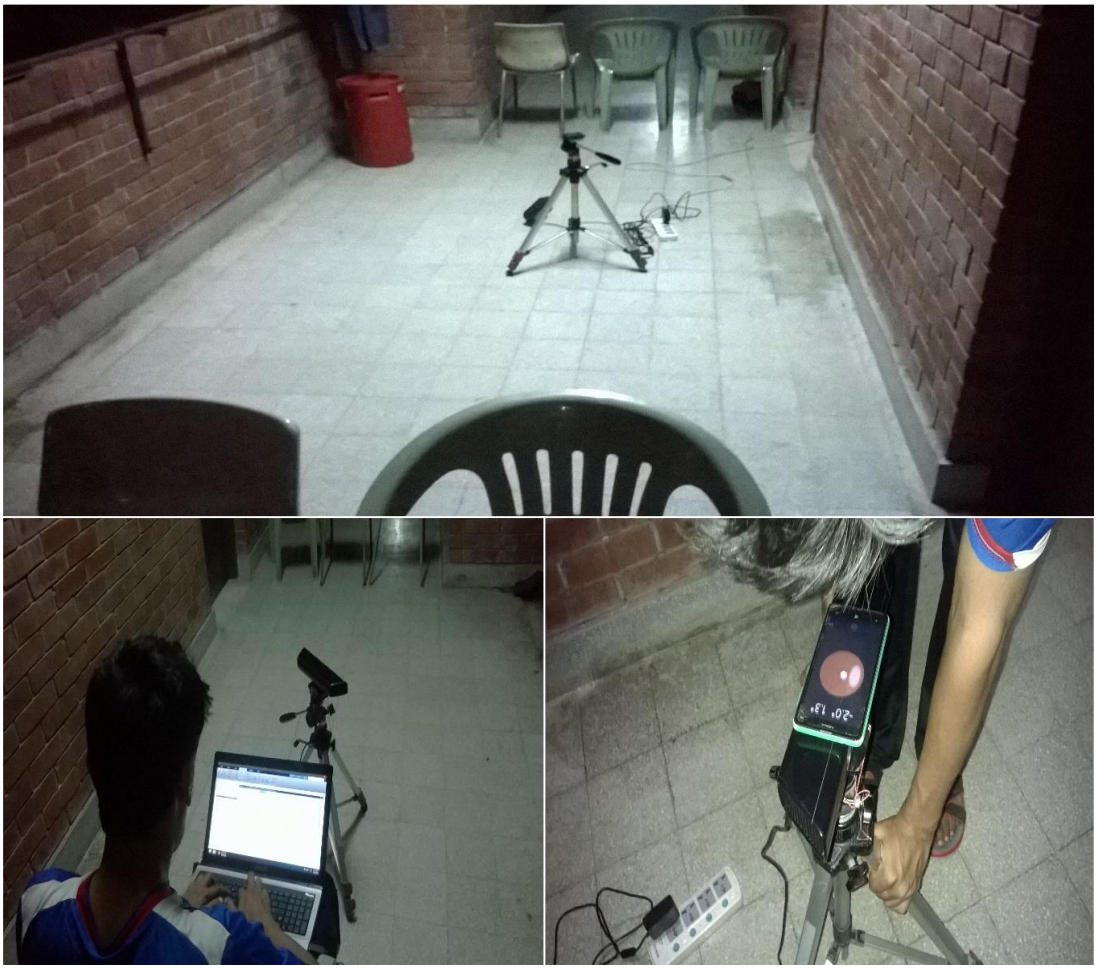

Figure 4.1: Collecting Dataset

The datasets were all generated by us. A separate program was written in MatLab to gather the data and store. Initially data were collected from 3 different locations.

4.2 Data Set

The data set was formed by taking RGB and depth images of 3 different locations. They are (a) Data set table tennis (b) Dataset Student Center of IUT (c) Data set of corridor.

The data sets have following characteristics:

1. The objects in the data was stationary.
2. There were no moving objects.
3. Lighting effect was constant.
4. Data set of environment was captured at different view of angle and translation.
5. All data were stored offline.
6. Both video and static data was gathered

The following table shows the Mean and variance of each of the RGB and Depth.

Figure 4.2: Dataset of Table Tennis

Figure 4.3: Dataset of Student Center

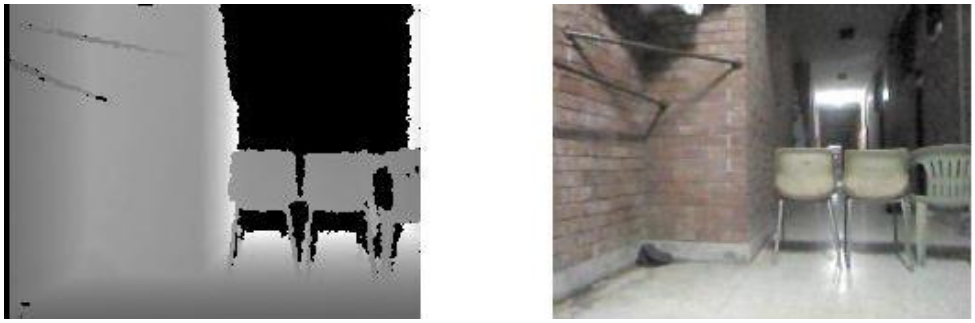

Figure 4.4: Dataset of Corridor

4.3 Intermediate Result (Features Detected)

In figure 4.5, features were extracted from two RGB images and a line was drawn from each of the corresponding matches. It has been found that many corresponding features do not match. So a nearest neighbor search was applied.

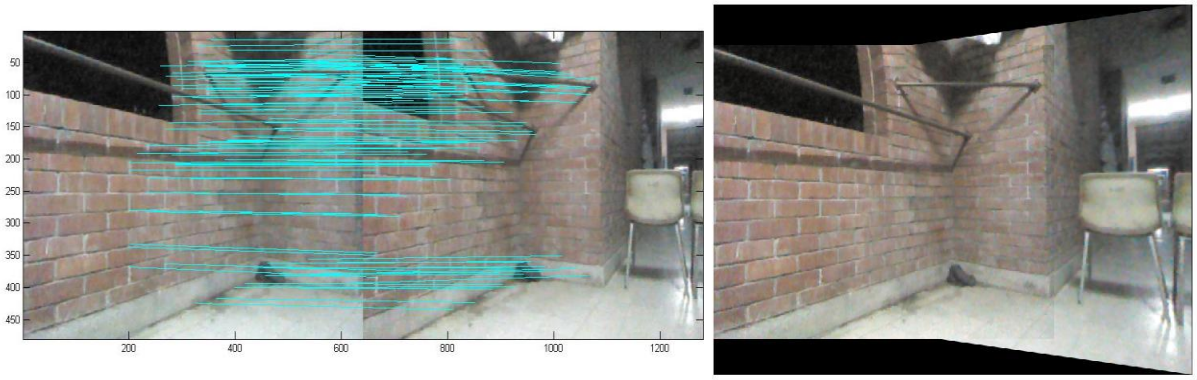

Figure 4.5: Feature Detected and Merged in Data Set 1

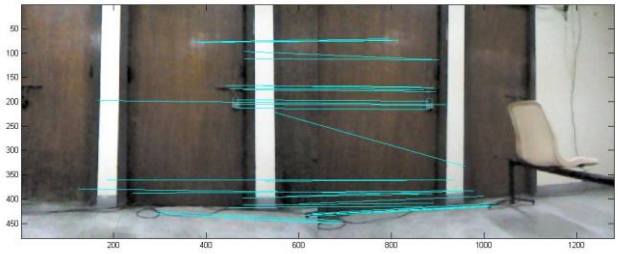

Figure 4.6: Feature Detected in Data Set 2

Figure 4.7: Image merged in Data Set 2

In figure 4.6 same algorithm was applied against different dataset and the stitched image is demonstrated in figure 4.7.

4.4 Final Result (Registered 3D Map)

In figure 4.8 two cloud points of two different frames are plotted with blue and red marker. Figure 4.9 demonstrates registered cloud points between two frames based on initial key feature points and geometric patterns with the help of ICP.

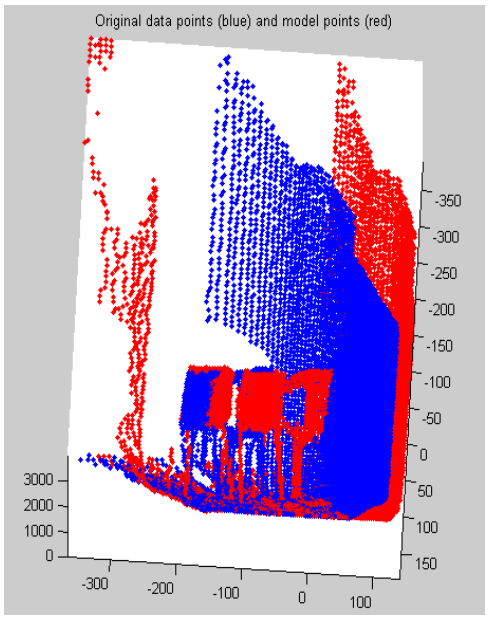

Figure 4.8: Unregistered Data points

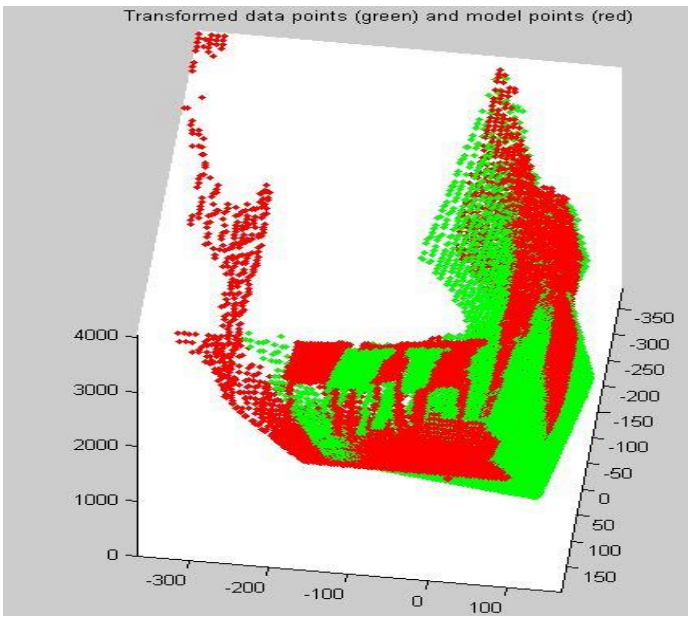

Figure 4.9: Registered Data points

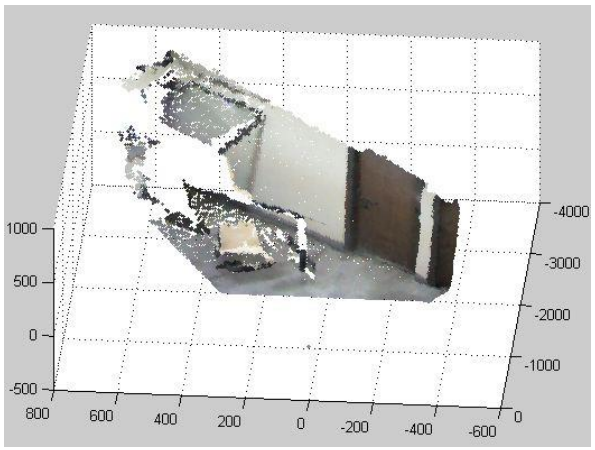

Figure 4.10: Map of Student Center Corner

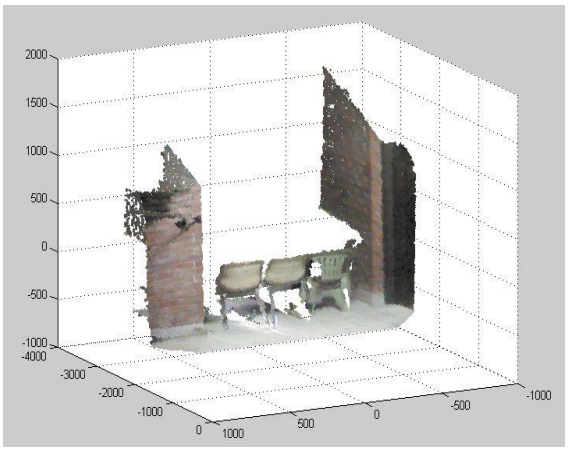

Figure 4.11: Map of Corridor

4.5 Performance Measurement

We have measured the performance by calculating accuracy of the objects found in the map. Accuracy means the percentage of derived length/area with respect to original objects. We calculate the accuracy by the following formula:

$$\text{Accuracy } = \frac{Derived\ Length}{Exact\ length/area}$$

Higher accuracy means a less possibility of error. In order to be the map more accurate we need good Odometry data. So during translation of camera we also compute the Odometry and angle as required and consequently provided the data to the algorithm to generate the map. We discussed about various challenge in this sector in Chapter 1.

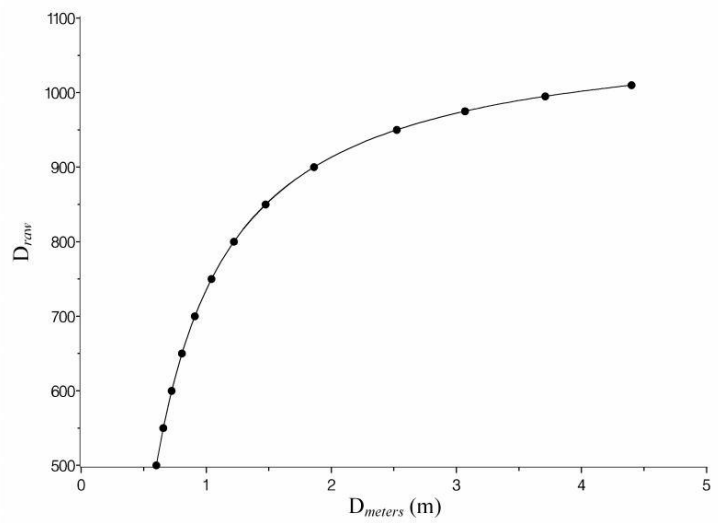

Figure 4.12: Kinect Raw Sensor data vs Real Distance data

The relation between Kinect raw depth data with the increase of object from the camera position is not linear. So the data is usually multiplied with a constant called Multiplier Constant S to acquire the Depth in cm against raw value returned by it. The choice of S also do not follow any linear equation. Though by experiment it revealed that the S can be predicted to have better result for higher degree polynomial function.

Here we show the performance of our proposed method in those challenges.

4.6 Linear Length Accuracy computation

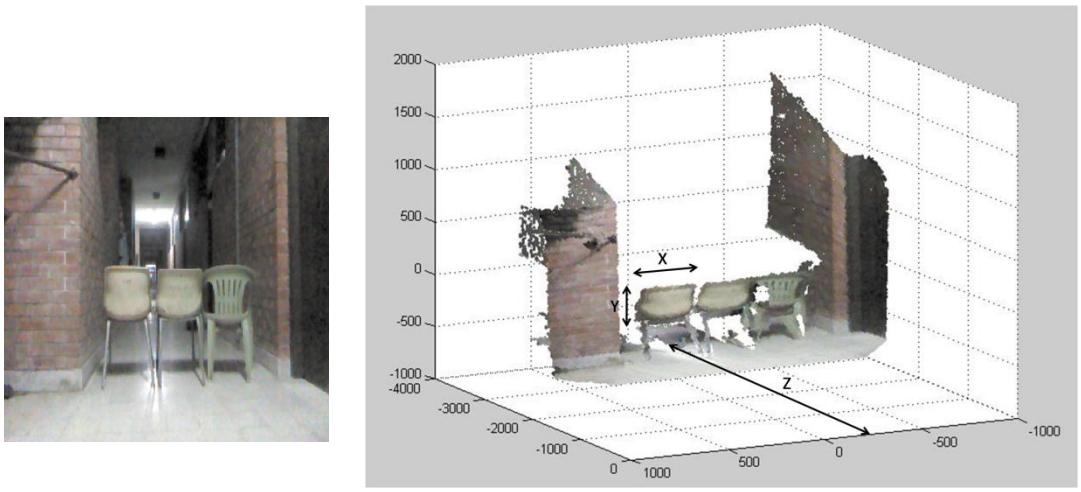

Figure 4.13: Object measurement

The accuracy of the map was generated by calculating the real distance between the points and the plane from which camera captures the image. In the map each point has 3 points x, y, z so distances of the object were counted by Euclidean distance formula as,$d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2 + (z1 - z2)^2}$.

And the accuracy was measured as

$$acc = 1 - (real\ distance - derived\ distance)/100$$

Table 4-2

Object	Multiplier S	Real Distance(Z,X,Y)	Derived Distance	Accuracy (%)
1	.0021	210	209.7	99.7
		40	33.5	98.5
		35	33.6	98.6
1	.0026	210	206.33	95
		40	38.75	98.10
		35	33.25	96.875
2	.0027	210	205	95
		50	46.4	96.4
		70	64.56	94.56
2	.0021	210	208	98
		50	44.9	94.9
		70	66.28	96.28
3	.0031	220	217	97
		35	33.2	98.2
		50	49.5	99.2
3	.0025	220	214.2	94.2
		35	31.32	96.32
		50	49.5	98.69
4	.019	321.8	218.9	98.2
		75.3	75.11	99.81
		55.5	51.5	96
4	.021	321.8	319.2	97.4
		75.3	72.8	97.5
		55.5	52.2	96.7
5	.022	330	327.89	97.89
		140	138.5	98.5
		110	102	92
5	.029	330	321	91
		140	134.59	94.59
		110	108	98.2
Overall Accuracy				96.772 %

The table 4.2 shows of the objects found in map in length. Length, width and distance from plane of camera were taken from real world measurement and also calculated from generated map. Accuracy was computed between the real world and derived measurement. During the calculation from the map, Kinect was calibrated for different values of S (described in section 3.3.1). The result is 96.772% which is acceptable.

4.7 Area Accuracy computation

The accuracy of the map can be estimated with respect to area .Each 3d map can be projected on the 2d surface .The area by the surface can be more accurate if the shape is real. In this way taking projection of the map or taking area of different plane and comparing with original cross sectional surface can give us a parametric measure of shape of the acquired map.

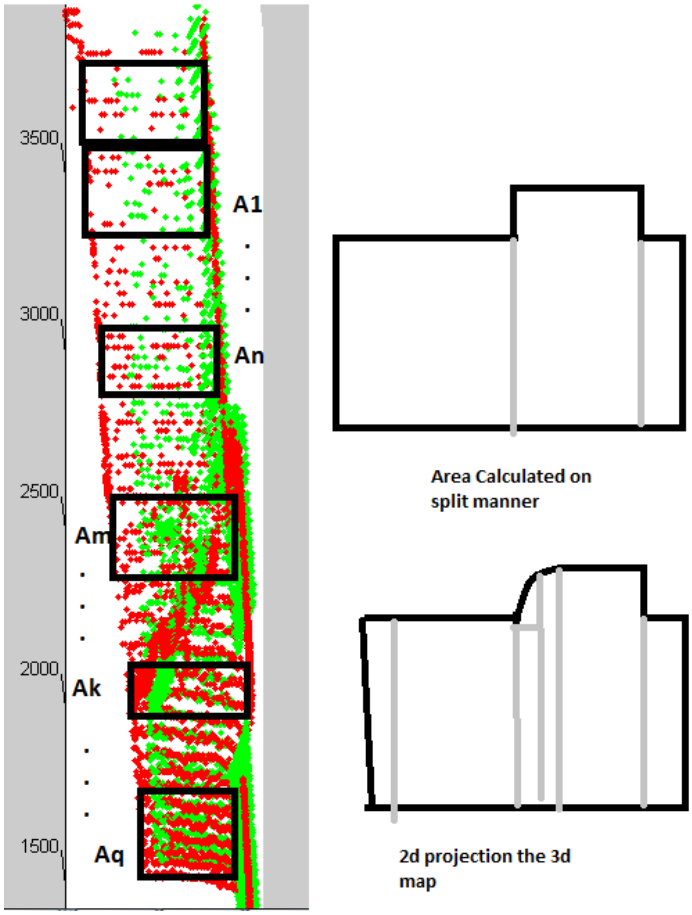

Figure 4.14: Accuracy computation using area

In the above figure (4.14) the area of the 3d map has been segmented in to A1,..An,...Am,...Ak, Aq sum of which gives A. The A was calculated in this manner and real value of the area and an accuracy of **89%** was found.

4.8 Runtime Analysis

Our data set was used to find the metrics between 3 variants of ICP. The result shows comparison of IRLS ICP [3], ICP using K-D tree, ICP (standard using random points), ICP (standard using matched points)

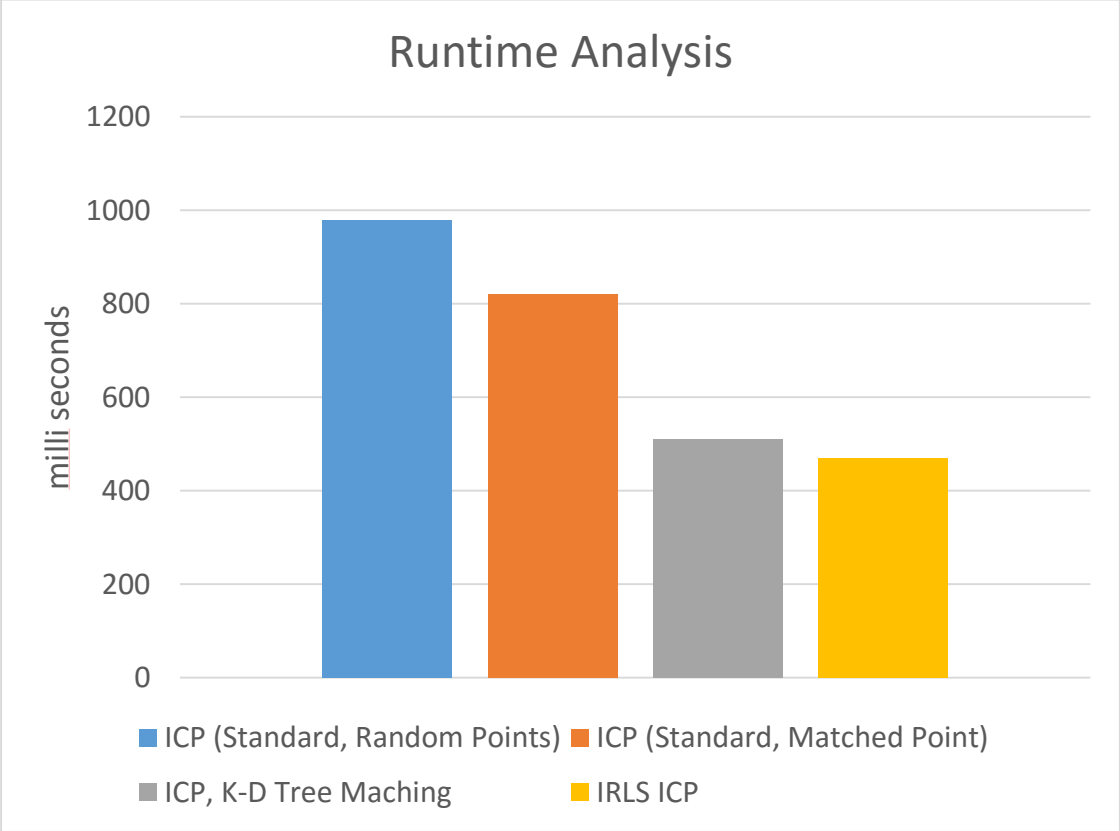

Figure 4.1: Runtime Analysis

Chapter 5

Conclusion

5.1 Summary of Contributions

Our work verified the use of SIFT based feature extraction and ICP based registration to be used in SLAM. It has been found that use of IRLS ICP gives best performance which can reduce the time complexity of SLAM algorithm to be implemented real time. Also it infers the fact that use of feature based SLAM can reduce dependency on Odometry. We have also found that we can generate the map even without a robot by just translating the camera to new position having common geometric pattern.

5.2 Limitations and Future Work

The scatter plot of the MatLab is very slow to merge and view multiple images. So use of C++ or C# based 3D plotting will show the whole map of the surrounding. We can make the system online by sending data to a server running our algorithm and a robot with low computation may carry the camera and connect with the server wirelessly. This concept can also be extended for UAV carrying camera and communication module and thus will give us 3D map of remote places form aerial view point.

Bibliography

[1] Hugh Durrant, Whyte. *Simultaneous Localization and Mapping (SLAM): Part 1 the Essential Algorithms*, Robotics & Automation Magazine, IEEE, 2006

[2] J. Besl and N. McKay, *A method for registration of 3-d shapes*, IEEE Trans. Pattern Anal. Mach. Intell., vol. 14, pp. 239–256, Feb 1992.

[3] S. Rusinkiewicz and M. Levoy, *Efficient variants of the icp algorithm*, International Conference on 3-D Digital Imaging and Modeling, 2001, pp. 145 –152, 2001.

[4] Dissanayake, Newman, Clark, Durrant-Whyte and Csorba, *A Solution to the Simultaneous Localization and Map Building (SLAM) problem*, IEEE Trans. On Rob. And Aut. Vol 17, No.3, June 2001

[5] Agostino Martinelli, Nicola Tomatis and Roland Siegwart, *Open Challenges in SLAM: An Optimal Solution Based on Shift and Rotation Invariants*, Swiss Federal Institute of Technology Lausanne (EPFL) CH-1015 Lausanne, Switzerland

[6] Smith, Self, et al. (1988), *Estimating uncertain spatial relationships in robotics and Uncertainty in Artificial Intelligence*, Elsevier Science Pub: 435-461.

[7] Crowley, J.L., (1989). *World Modeling and Position Estimation for a Mobile Robot Using Ultrasonic Ranging*. IEEE International Conference on Robotics and Automation (ICRA), Scottsdale, AZ.

[8] J.J. Leonard, H.F. Durrant-Whyte, *Directed Sonar Sensing for Mobile Robot Navigation*, Kluwer Academic Publishers, Dordrecht, 1992.

[9] Brynjolfsson, Erik; McAfee, Andrew (Jan 20, 2014). *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies*. W. W. Norton & Company. p. 52.ISBN 9780393239355.

[10] S.Riisgard, M.R.Blas. *A tutorial approach to Simultaneous Localization and Mapping*. Massachusetts Institute of Technology, Cambridge, MA, 2005.

[11] B Amberg, T Vetter, *Optimal landmark detection using shape models and branch and bound*. Computer Vision (ICCV), 2011 IEEE

[12] Agostino Martinelli, Nicola Tomatis and Roland Siegwart, *Open Challenges in SLAM: An Optimal Solution Based on Shift and Rotation Invariants*. Swiss Federal Institute of Technology, Lausanne (EPFL),CH-1015 Lausanne, Switzerland

[13] Baumberg, A. 2000. *Reliable feature matching across widely separated views*. Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina, pp. 774-781.

[14] Per Bergström, *Robust registration of point sets using iteratively reweighted least squares*, Springer Science+Business Media New York 2014

[15] J. Sivic and A. Zisserman, *Video Google: A text retrieval approach to object matching in videos*, in Proc. 9th Int. Conf. on Computer Vision, Nice, France, 2003, pp. 1470–1478.

[16] M. Labbe and F. Michaud, *Appearance-based loop closure detection for online large-scale and long-term operation*, IEEE Transactions on Robotics, vol. 29, no. 3, pp. 734–745, 2013.

[17] M. Labbé and F. Michaud, *Online Global Loop Closure Detection for Large-Scale Multi-Session Graph-Based SLAM*, in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014.

[18] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard, *Efficient estimation of accurate maximum likelihood maps in 3D*, in Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2007, pp. 3472–3478.

[19] Cihan Altuntas. *An Experimental Study on Registration Three-Dimensional Range Images using RAND and Intensity Data*, Selcuk University, Engineering and Architectural Faculty, Geomatic Engineering, 42075, Selcuklu, Konya, Turkey, caltuntas@selcuk.edu.tr

[20] Stewart, C., Tsai, C.L., Roysam, B. *The dual boot strap iterative closest point algorithm with application to retinal image registration*. IEEE Trans. Med. Imaging 22(11), 1379–1394(2003)

[21] R.Biswas, B.Limketkai, S.Sanner, and S.Thrun. *Towards object mapping in non-stationary environments with mobile robots*. Proceedings of the International Conference on Intelligent Robots and Systems (IROS), 2002, pp. 1014–1019.

[22] B.S.Blackmore, H. Have, S. Fountas. *A specification of behavioral requirements for an autonomous tractor*. Automation technology for off-road equipment, edited by Zhang, Q. ASAE Publication, 2002, pp. 33-42.

[23] A. Elfes. *Using occupancy grids for mobile robot perception and navigation*. IEEE Computer, 6 1989. pp. 46–57.

[24] D.Haehnel, D.Schulz and W.Burgard. *Map building with mobile robots in populated environments*. Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2002, pp. 496–501.

[25] B. Yamauchi, A. Schultz, W. Adams. *Mobile robot exploration and map building with continuous localization*. Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998, pp. 3715–3720.

[26] J.Ryde, H.Hu. *Fast Circular Landmark Detection for Cooperative Localization and Mapping*. Proceedings of the 2005 IEEE International Conference on Robotics and Automation Barcelona, Spain, April 2005, pp. 2756-2761.

[27] C.Wang and C.Thorpe. *Simultaneous localization and mapping with detection and tracking of moving objects*. IEEE International Conference on Robotics and Automation, 2002, pp. 2918–2924.

[28] C.C.Wang, C.Thorpe, S.Thrun. *Online Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects: Theory and Results from a Ground Vehicle in Crowded Urban Areas*. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2003, pp. 842-849.

[29] D.Wolf, S.Sukhatme. *Towards Mapping Dynamic Environments*. Proceedings of the International Conference on Advanced Robotics (ICAR), 2003, pp. 594-600.