Place Recognition and Factor Graph Localization for Mobile Robots using Google Indoor Street View

by

©Kusal B. Tennakoon

A Thesis submitted to the School of Graduate Studies in partial fulfillment of the requirements for the degree of

M.Eng

Faculty of Engineering and Applied Science

Memorial University of Newfoundland

May 2021

St. John's Newfoundland

Abstract

This thesis develops an indoor localization system for mobile robots using Google indoor street view.

The proposed localization system consists of two main modules. The first is a place recognition system based on Google's indoor street view. Its purpose is to determine the position of the robot in terms of the node on the street view map that is closest to the robot's actual location. It is achieved by comparing an image captured by the robot's camera against the indoor street view images. In order to achieve the best accuracy possible, the input image is compared to every street view image. The system employs two verification stages. The first stage is based on the visual similarity among the images. The best five images that qualified through this stage become candidate images for the second verification stage. In this stage, the geometric consistency between the images is assessed. The image that passes this test with the highest similarity score is considered a match with the input image. The proposed place recognition system is tested on different data sets and the performance is assessed using standard evaluation metrics.

The second part is the main module of the proposed localization system. It is a graph-based estimation module that incorporates the odometry data, visual feedback, and motion data that eventually is solved via optimization techniques. The result is the estimates of the robot's locations at specified intervals along its journey. It uses odometry data to interpret connections among successive poses. Also, it uses visual information from the robot's camera to establish constraints between the robot's poses and the map. It is achieved through constraints derived using two images, one from the robot's camera and one from the node in the map that matches best with the image from the robot's camera. The localization system is designed to minimize the drift caused by the odometer. The system is simulated and then tested for a data set captured at the Memorial University of Newfoundland engineering building basement. The performance of the system is evaluated using standard error metrics.

Acknowledgements

I wish to express my sincere gratitude to my supervisor, Dr. Oscar de Silva, for consistently guiding, tolerating all my mistakes and setbacks.

I wholeheartedly appreciate the support from the co-supervisors, Dr. George Mann and Dr. Raymond Gosine, for their advice which proved monumental towards the success of my study.

The physical and technical support from the Faculty of Engineering and Applied Science of Memorial University of Newfoundland is immeasurable. Without their help, this thesis would not have been a reality.

I would like to appreciate the financial support from Dr. Raymond Gosine and my funding sources, namely the Natural Science and Engineering Research Council (NSERC) and the graduate scholarship by the Memorial University of Newfoundland. If not for their financial support this thesis would not have reached this far.

I wish to express my gratitude to Mr. Paul Bourke from the University of Western Australia for his consultation on image transformations.

I would acknowledge the constant love and appreciation from my parents, brother and the love of my life, for holding onto me during hard times.

I wish to appreciate the support and motivation from my labmates and my friends, who stood by my side when I was in need.

Finally, I would like to appreciate the support of my colleague Mr. Ravindu Thalagala, in accomplishing the hardest challenges encountered.

The assistance and moral support that you all provided during my study are highly recognized.

Table of Contents

List of Tables

List of Figures

Nomenclature

Coordinate frames

$\{\boldsymbol{B}\}$	Robot body-fixed frame
$\{\boldsymbol{C}\}$	Camera coordinate frame
$\{\boldsymbol{M}\}$	Map fixed frame
$\{\boldsymbol{W}\}$	World coordinate frame

Greek letters

$\boldsymbol{\lambda}$	Eigen value
$\boldsymbol{\Sigma}$	Covariance matrix
$\boldsymbol{\sigma}$	Standard deviation
$\boldsymbol{\theta_H}$	Horizontal field of view angle
$\boldsymbol{\theta_V}$	Vertical field of view angle

Operators

$\boldsymbol{E(X)}$	Cost function of the set of variables X
$\boldsymbol{p(\Theta)}$	Joint probability density of the random variable Θ

Roman letters

$\boldsymbol{A}$	Measurement Jacobian matrix
$\boldsymbol{c}$	Image center in camera coordinates
$\boldsymbol{f}$	Focal length

$\boldsymbol{H_{FOV}}$	Vertical field of view distance
$\boldsymbol{H_{ij}}$	Jacobian of the measurement function h
$\boldsymbol{H}$	Image height
$\boldsymbol{K}$	Camera intrinsic matrix
$\boldsymbol{P}$	Probability density function
$\boldsymbol{R}$	Rotation matrix
$\boldsymbol{t}$	Translation vector
$\boldsymbol{W_{FOV}}$	Horizontal field of view distance
$\boldsymbol{W}$	Image width
$\boldsymbol{T}$	Transformation matrix

Groups & Sets

$\boldsymbol{\mathcal{E}}$	Set of all edges of a factor graph
$\boldsymbol{\mathcal{N}(\phi_i)}$	The set of variable nodes adjescent to a factor ϕ_i
$\boldsymbol{\mathcal{U}}$	Set of all factors of a factor graph
$\boldsymbol{\mathcal{V}}$	Set of all variables of a factor graph
$\boldsymbol{\Xi}$	Set of incremental pose coordinates
$\boldsymbol{GL(3)}$	General linear group of order 3

Superscripts & Subscripts

${}^{\boldsymbol{C}}\boldsymbol{O_{AB}}$	O of reference frame B w.r.t frame A expressed in frame C
$\boldsymbol{O^A}$	O expressed in reference frame A
$\boldsymbol{O^T}$	Transpose of O
$\boldsymbol{O_x}$	x coordinate of O
$\boldsymbol{O_y}$	y coordinate of O
$\boldsymbol{O_z}$	z coordinate of O
$\boldsymbol{O_B^A}$	O of reference frame B w.r.t reference frame A

Abbreviations

API	Application Programming Interface
ASIFT	Affine Scale Invariant Feature Transform
BoRF	Bag-of-Raw-Features
BoW	Bag-of-Words
BRIEF	Binary Robust Independant Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CNN	Convolutional Neural Network
COLD	COSY Localization Database
EKF	Extended Kalman filter
FACT	Fast Adaptive Colour Tags
FLANN	Fast Library for Approximate Nearest Neighbors
FOV	Field of view
FSH	Feature Stability Histogram
GDOP	Geometric Dilution Of Precision
GiST	Generalized Search Tree
GPS	Global positioning system
GPU	Graphics Processing Unit
GTM	Graph Transformation Matching
GTSAM	Georgia Tech Smoothing And Mapping
HIF	Hierarchical Inverted File

HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradients
IMM	Integrated Multi Model
IMU	Inertial measurement unit
iSAM	Incremental Smoothing And Mapping
LIDAR	Light Detection and Ranging
LM	Lavenberg-Marquardt
MAP	Maximum A Posteriori
MAV	Micro-aerial vehicle
MHE	Moving Horizon Estimation
MSCKF	Multi State Constraint Kalman Filter
MUN	Memorial University of Newfoundland
OACH	Orientation Adjacency Coherence Histogram
ORSA	Optimized Random Sampling Algorithm
OVV	Online Visual Vocabulary
PCA	Principal Component Analysis
PDL	Powell's Dog Leg
PIRF	Position-Invariant Robust Features
PTM	Probabilistic Topological Maps
RANSAC	Random Sample And Consensus
RMSE	Root Mean Square Error
ROS	Robot Operating System
RTAB-Map	Real-Time Appearance-Based Mapping
SAD	Sum of Absolute Differences
SAM	Smoothing And Mapping
SFM	Structure From Motion

SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization And Mapping
SMART	Sequence Matching Across Route Traversals
SURF	Speeded Up Robust Features
TF-IDF	Term Frequency Inverse Document Frequency
UWB	Ultra Wide Band
V-SLAM	Visual Simultaneous Localization And Mapping
VINS	Visual Inertial Navigation System
VLAD	Vector of Locally Aggregated Descriptors
VLD	Virtual Line Descriptors
VT&R	Visual Teach and Repeat

Chapter 1

Introduction

In this chapter, the motivation for this study is presented, followed by an overview of the available place recognition based localization methods and their associated limitations. The thesis's problem statement is formulated, and the sub-problems that are addressed in this thesis are introduced. Finally, the objectives and expected contributions are highlighted, followed by the organization of the thesis.

1.1 Motivation

Mobile robots have proven success in several application areas that are considered safety critical for human operators. Examples include first response squads in emergency situations [1], search and rescue [2], exploration of abandoned mines [3–5], investigating pipelines [6–8], autonomous planetary exploration [9–11], surveillance [12, 13] and neutralization of explosives [14–16]. Another avenue is assisting tasks that humans alone take a long time to accomplish, such as construction of structures [17–19], mapping sites [20, 21] transportation of goods [22], building material [23, 24], visual yield mapping and precision farming [25, 26].

An interesting development of robotic localization and mapping is what is known as *Simultaneous Localization And Mapping* (SLAM). It refers to the case where a robot continuously builds a map of its surroundings while simultaneously determining its location in it. When visual information obtained through cameras is used for SLAM, it is referred to as *Visual Simultaneous Localization And Mapping* (V-SLAM) [27]. The availability of visual information proves to be a considerable advantage due to the

richness of the data provided. It, when combined with the low cost of cameras and the advancement of camera technology over the years, makes them excellent candidates as sensors for robots engaged in SLAM. Besides, adding a camera to a robot minimizes the necessity of other high-cost localization support sensors, reducing the weight and power consumption, and enhancing its operating time. It has been shown that visual-SLAM can be autonomously executed even in *Global positioning system* (GPS) denied environments [28]. Several V-SLAM systems has been since developed in both research and commercial applications, hence V-SLAM can be now considered as a better alternative when executing SLAM indoors [29, 30], when compared with *Light Detection and Ranging* (LIDAR) based [31], or external positioning aid based [32, 33] systems.

The topic of SLAM can be categorized based on the underlying architecture. Discussed in sections 2.1.1, 2.1.2, 2.1.3 and 2.1.4 are the principal branches along which SLAM has evolved over the years.

1.1.1 ISL building wide localization system

The intelligent systems lab (ISL) of Memorial University of Newfoundland (MUN) is developing a building-wide navigation system for multi-robot applications. The system comprises of *Micro-aerial vehicle*s (MAVs) equipped with embedded low complexity stable *Visual Inertial Navigation System* (VINS) which serve as the odometers, and external navigation aids (*Ultra Wide Band* (UWB) and place recognition) to assist with periodic corrections of the platforms. The system requires a stable and robust back end[1] that is simple, quickly deployable, and can incorporate a multitude of different sensors with minimal reconfiguration. As an example, if the robot is to enter an area with poor lighting, or lack of UWB coverage, and perhaps outdoor transitions where GPS becomes available for a while the localization method should be able to capture all this information in a common framework easily.

This thesis's main objective is to develop a visual loop closure back end and establish it in an autonomous platform to be used as a research tool for mobile robot research

[1] The back end is where the estimation and loop closure take place. The front end acquires data from the sensors and abstracts them into mathematical forms the back end can use. For further details, refer section 2.2.2

activities. This thesis is specifically targeted for the development of the mobile robot navigation system discussed above. For this purpose, the problem can be simplified by assuming a pre-existing map from Google street view. Hence, visual localization in the presence of a pre-existing map applies. The details of the system are discussed in section 2.2.

1.2 Problem Statement

Developing a place recognition and factor-graph based indoor localization system using Google indoor street view presents several challenges. When the system is built around a monocular camera, the complexity of the task increases further. Some of the challenges encountered can be listed as follows.

1. Absence of indoor maps that satisfies the requirements of the desired system, since Google's indoor street view is not well established as its outdoor counterpart
2. The equirectangular[2] [34] projection of the street view images makes all the conventional feature matching methods not directly transferable.
3. The abundance of repetitive and self-similar structures increases the possibility of perceptual aliasing.
4. Lack of robust, unique features makes feature matching a challenge.
5. Frequent changes in indoor environments due to human movement and displacement of objects make a comparison against a pre-constructed map challenging.
6. Conventional camera calibration methods are not directly transferable since the intrinsic parameters (especially the focal length) obtained through such methods do not relate well to equirectangular images.

[2] Equirectangular projection is a way of representing spherical images as rectangular panoramas. It converts a sphere into a rectangular Cartesian grid where the rectangular grid cells are of the same size, shape, and area. Equirectangular images are severely distorted towards the periphery, and the amount of distortion decreases radially towards the center. It is also known as simple cylindrical, equidistant cylindrical and rectangular projection.

7. Using monocular cameras leaves no way of extracting depth information from the images. It makes utilizing the image data for imposing constraints on the factor graph complicated.

8. Searching through large databases of images is exhaustive and time-consuming, thus, directly impacting the localization and loop closure detection.

9. Performing *Structure From Motion* (SFM) on all the available images presents computational resource limitations and thus is an overkill in this research.

In this thesis, we focus on the following three problems.

1.2.1 Problem I: Factor graph localization with Google street view

Using factor graphs has become a preferred method in mobile robot localization [35, 36], mainly because both measurement and motion models can be incorporated into a single graph. It is a systematic approach in using and can be solved by conventional graph solvers such as g2o [37]. A majority of localization studies based on using factor-graphs, rely on constructing the topological map on the fly. As an example, the popular VINS-Mono [38] system builds its visual map during navigation. Building a map in itself is a computationally demanding procedure which can be minimized by using pre-existing public maps of an environment.

This thesis proposes using a factor-graph for localization, which exploits Google street view by using it for topological place recognition. The thesis establishes the constraints on the factor graph using two image views between the map and the robot. Using an already available map is much more efficient for our concerned application. The visual updates necessary for corrections should be made efficiently using image views between robot poses and map nodes.

1.2.2 Problem II: Place recognition using Google street view

Using Google street view for localization is a recent trend in localization [39–45]. Many of the studies conducted are focused on the aspect of place recognition [44–46] whereas several other studies have considered autonomous localization of a robot [39, 40].

One of the studies is the Google street view based urban localization study by Salarian et al. [46]. Their primary focus is accurately localizing a mobile device using a monocular camera, GPS module, an IMU, and Google street view. The key underlying idea of the method is image feature-based scene recognition.

The urban localization study by Yu et. al [43] uses a monocular camera and street view for localization. This approach exploits the topometric nature of Google street view for localization. This approach proposes place recognition as a primary selection step, which is then directed to a metric localization step. It also modifies the conventional *Bag-of-Words* (BoW) matching process by considering multiple database constraints.

Majdik et al. [40] studies Google street view for autonomous robot localization. The work presents a solution to urban localization by solving the *air-ground matching* problem. It identifies the failure of conventional image feature matching due to substantial distortions present in the equirectangular street view images and proposes a solution by generating virtual views. This study also identifies the limitations of *Random Sample And Consensus* (RANSAC) for outlier rejection and proposes an alternative approach based on k-*Virtual Line Descriptors* (VLD).

Agarwal et al. [39] present a similar study for ground robot applications. This study addresses the problem of significant distortions in Google street view images through the use of perspective images downloaded directly from Google's servers.

The above studies are all considering outdoor applications. Work in [41] proposes an approach for indoor localization using google street view, but the study uses a strategy other than visual place recognition.

Direct adoption of an outdoor localization system to an indoor environment is not trivial since indoor localization involves challenges that are not encountered in outdoor localization. It includes the unavailability of GPS due to the loss of signal strength [47], inability to use the compass due to distortions (Hard-iron and soft-iron) [48], lack of unique landmarks, the abundance of repetitive and self-similar structures, illumination deficits, and frequent changes in the environment. There are no performance evaluations conducted for Google indoor street view based visual topological localization to the best of the author's knowledge. Therefore, evaluating such a system's performance

for an indoor environment is the main emphasis of this thesis.

1.2.3 Problem III: Limitations in benchmark datasets for Google indoor street view localization

As was discussed earlier, few studies have focussed on using Google street view for indoor localization [41]. There are datasets for validating visual topological indoor localization systems such as the *COSY Localization Database* (COLD) [49], which has been used for studies such as the recent study by Thoma et al. [50]. The images it constitutes are either perspective or omnidirectional. Hence, this work opts to create and upload our own Google street view dataset with ground truth since Google's indoor street view is of limited and sparse availability for MUN engineering. As we expect to perform mobile robot localization studies at this location, a reliable map upload to Google servers is necessary. With the rapid growth of Google indoor mapped sites, we expect that the proposed localization architecture will find broader applicability. Furthermore, the experimental procedure followed for creating reliable reference maps can be replicated at different sites to update the indoor street view database periodically.

1.3 Objectives and Expected Contributions

In order to address the three problems regarding visual topological indoor localization discussed under section 1.2, the thesis considers the following main objectives.

1.3.1 Objective I : Developing a factor-graph for a visual topological localization system

Task 1 Proving the concept underlying the proposed strategy through MATLAB simulations.

Task 2 Developing a ROS C++ based simulator that is capable of exploiting the established software libraries yielding the full potential of the proposed system. Developing a visualization platform, along with the necessary software tools.

Task 3 Developing a pose recovery system capable of estimating the relative change in camera pose between two image views.

1.3.2 Objective II : Developing a place recognition system

Task 1 Developing a Google street view based brute force place recognition system that predicts the node in the topological map that closely resembles an input query image.

Task 2 Evaluating the performance of the place recognition system through standard performance metrics by executing it on a set of street view snippets, for an outdoor scenario.

Task 3 Testing the place recognition system on a data set of MUN engineering basement, analyzing the performance indoors, and assessing viable improvements.

1.3.3 Objective III : Validating the factor-graph based localization system for an indoor data set

Task 1 Configuring Seekur Jr. to suit the experiments conducted by installing necessary software libraries.

Task 2 Collecting a validation data set by traversing Seekur Jr. around the corridor of the MUN Engineering building basement.

Task 3 Validating the performance of the factor-graph based localization system under simulated essential matrix constraints.

Task 4 Validating the performance of the factor-graph based localization system under actual essential matrix constraints.

Accomplishing these three objectives results in a Google street view based topological place recognition system and factor-graph based visual localization system, with preliminary module wise validation for an indoor environment. Combining the necessary sensor modules completes the initial step towards the building-wide indoor navigation system for mobile robots that the intelligent systems lab (ISL) of Memorial University of Newfoundland (MUN) is developing.

1.4 Organization of the Thesis

This section illustrates how the thesis is organized. The main chapters and a briefing on their content provide insight to a reader interested in delving into the details.

Chapter 1 : Introduction presents an overview of the research area, highlights the research statement, and elaborates on the objectives and critical contributions of the study.

Chapter 2 : Background presents a literature review of similar or related work in topological SLAM and localization.

Chapter 3 : Factor Graph presents the work carried out in developing an optimization-based back end to the localization system.

Chapter 4 : Place Recognition System presents the work carried out in producing a visual topological place recognition system.

Chapter 5 : Experimental Evaluation presents experimental evidence substantiating the validity and quality of the work produced throughout the research.

Chapter 6 : Conclusion presents the conclusions rested upon from the conducted study, viable improvements identified, future directions to pursue, and the contributions made by the author.

Chapter 2

Background

In this chapter, several branches of SLAM are presented, which is followed by a description of the overview of a visual-topological SLAM system, its back end, and the front end. The rest of the chapter presents the state of the art related to visual-topological localization and visual place recognition.

2.1 Principal branches of SLAM

2.1.1 Outdoor vs Indoor SLAM

Based on the operating environment, SLAM divides into two main domains: outdoor SLAM [51], and indoor SLAM [52, 53]. Indoor SLAM primarily differs from outdoor SLAM due to particular challenges that are associated with indoor SLAM. Some examples are unavailability of GPS, inability to use the compass (due to high magnetic disturbance inside structures), high abundance of repetitive and self-similar structures, lack of features of the environment, and frequent changes in the background (e.g., movement of people and re-location of objects). Often more than one of these challenges co-occur.

2.1.2 Filter-based vs Optimization-based SLAM

Based on the method of estimation used, SLAM can be categorized as filter-based methods and optimization-based methods. Filter based methods tend to distinctly

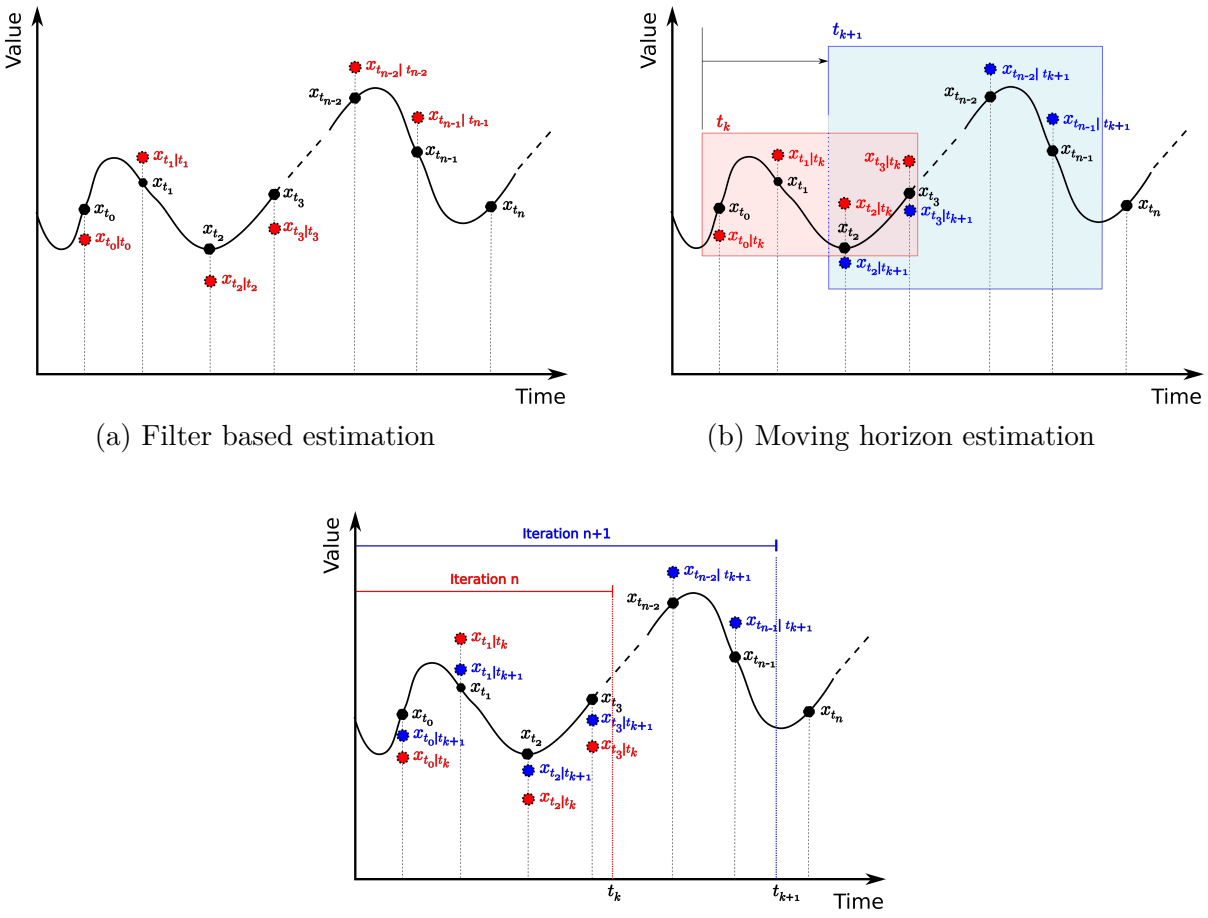
separate the motion model and the measurement model of the problem. Since filter-based methods are usually application-specific, they require re-tuning every time they come across new applications. Filter based methods often tend to be unstable during a lack of sensor data and sometimes completely fail in the absence of it. Over the years, *Extended Kalman filter* (EKF)-based methods such as the *Multi State Constraint Kalman Filter* (MSCKF) [54] have produced results almost as accurate as those of optimization-based methods. But it is not to mention that the salient snags of filter-based methods are inevitable in them as well. However, filter-based methods are still preferred despite the shortcomings they involve. As filter-based methods estimate one pose at a time (figure 2.1(a)), unlike optimization methods, an increase in the number of variables has minimal impact on the speed of execution and computational cost. Therefore, filter-based methods, in general, are favored for applications that involve relatively fast dynamics but posses lower computational capacities.

Conversely, optimization-based methods represent the SLAM problem as a least-squares minimization problem. The advantage of optimization-based methods is that the implementations are more stable to execute than filtering based techniques with minimal needs for tuning of the system, resulting in better accuracy than the filter-based methods. Unlike filter-based methods, optimization methods consider both the measurement and observation models as *factors* of specific forms, which equally contributes to the optimization process. A *factor* in this regard, represents probabilistic information about the variables derived from measurements and prior knowledge, which imposes a constraint between two variables (poses of the robot).

Maximum A Posteriori (MAP) estimation has been proven to be more accurate than conventional methods of non-linear filtering. MAP estimation can be performed following two methods. In the first type, the estimation is performed on a window of data selected from all the data available (figure 2.1(b)). The data within the estimation window is changed at every iteration by shifting the window (estimation horizon) forward in time. This method is known as *Moving Horizon Estimation* (MHE). In the second type, the estimation is performed on all the data that is available (figure 2.1(c)). Factor graph optimization and bundle adjustment are examples for the latter [55, 56]

Figure 2.1 illustrates the difference between a filter, MHE, and an optimization

method acting on the same set of variables over time. $\mathbf{x_{t_m}}$ represents the actual state observed at time t_m and $\mathbf{x_{t_m}|_{t_n}}$ represents the value of x_{t_m} estimated at time t_n.



(a) Filter based estimation

(b) Moving horizon estimation

(c) Optimization based estimation

Figure 2.1: Filter based, moving horizon and optimization based estimation performed on the same set of state variables. In the above figures, black colour indicates the actual states vs time. In figure 2.1(a), every state is estimated at a time specific for it and only one variable is estimated at a given time as indicated by the red coloured points. Also, the estimations are made in the order the variables occur in time. In figure 2.1(b), the red and blue rectangles indicate the estimation window (horizon) at time t_k and t_{k+1} respectively, while red and blue points indicate the estimations made at t_k and t_{k+1} respectively. Here, the points contained withing the estimation window are estimated at the same instance. Hence, variables x_{t_2} and x_{t_3} are re-estimated at time t_{k+1}. In figure 2.1(c), red and blue colours indicate the estimates made during the iterations n and $n+1$ which occurs at times t_k and t_{k+1} respectively. Here, every variable is re-estimated within every iteration. Hence, variables x_{t_0} - x_{t_3} are re-estimated

Note

In the above figures, the positions of the points above the time-axis do not reflect their values. The points are located within the figures to maximize clarity. However, the positions along the time-axis do reflect their position in time.

2.1.3 Metric vs Topological vs Semantic SLAM

In SLAM, a map can be loosely defined as a geometric representation of the landmarks within a robot's surroundings and the relationships among them. In general, maps used for SLAM do not portray the geometric or visual attributes of the landmarks itself. Instead, they indicate the boundaries of the objects present, space they occupy in the surrounding or the positions of the landmarks, and the connectivity among them. It serves as an aid for the robot to determine its position within its surroundings. It also helps to identify the revisiting of locations that have already been visited once. The latter scenario is known as *loop closure*.

Maps generated in SLAM are fourfold. They are metric maps, topological maps [57], topometric (a hybrid of metric and topological maps also known as topological-metric) maps [58], and semantic maps [59]. On the basis of the map involved, SLAM has evolved into four sub-fields, namely Metric SLAM [60], Topological SLAM [61–63], Topometric SLAM [64] and Semantic SLAM [65, 66].

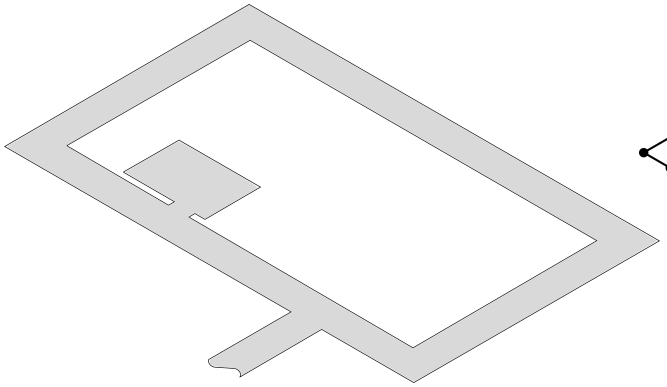
Metric maps (figure 2.2(a)) represent the environment exactly as it is in terms of geometry. The distances, angles, shapes, areas, and other geometric information of the environment are stored in the map. Topological maps represent the environment as a collection of distinguishable landmarks at selected locations and the connectivity among those. These are referred to as *nodes* and *edges* respectively. Topometric maps use the best of both worlds by having their upper level as a topological map with their nodes directed to metric maps at a lower level. Google maps are a well-known example for this category (Depending on how it is used, Google maps can be considered as a topological map as well). Semantic maps capture the geometric information in the environment and classify them under semantic labels.

Metric maps used for SLAM are either *landmark-based maps* or *occupancy-grids*. Landmark-based maps represent the environment as a set of sparse landmarks. In contrast, occupancy grids' representation is in the form of a cluster of discrete cells, each to which a probability of occupancy is assigned. They are more difficult to build and maintain. The abundance of information in metric maps turns out to be advantageous for tasks such as obstacle avoidance. But, it comes at the cost of large storage requirements, increased search time, and prone to incorrect matches during loop closure detection.

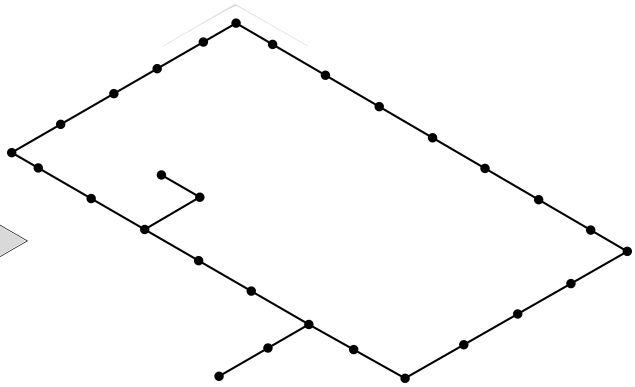
On the contrary, topological maps (figure 2.2(b)) are much simpler, compact, requires less storage space, and scales better. The presence of topological information improves the number of correct matches during loop closure detection with a reduction in search time. In the presence of location priors, the search time can be reduced further. Topological maps are often preferred in V-SLAM, in which loop closure detection is based on vision-based place recognition. Although the lack of metric information seems a disadvantage, topological maps have proven its practical applicability by facilitating successful place recognition even with weak image data [67, 68].

Semantic maps (figure 2.2(c)) are constructed by overlaying the geometric and visual-appearance information of the objects in the environment onto a map built similar to a metric map. The additional geometric information improves the accuracy of the map, while the visual-appearance information makes it a better qualitative representation of the surrounding. These maps are beneficial in applications for which the geometric information contained in a traditional map may become insufficient.

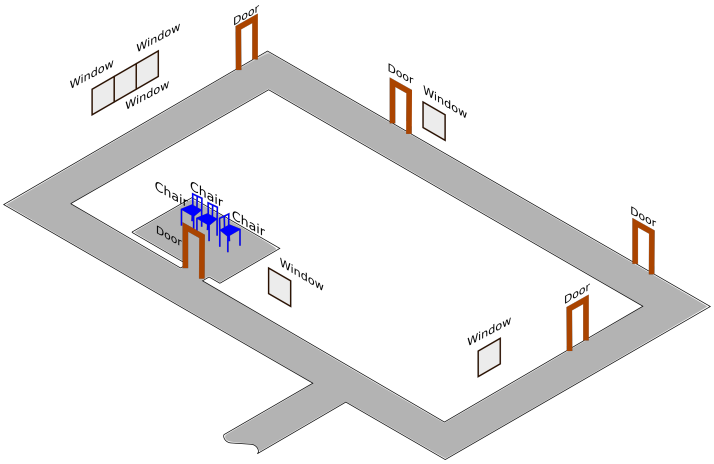
Google Street View is a topological map spanning the entire earth. Google Indoor Street View provides its users the freedom to create their indoor maps and uploading them to the Google server. Thus, it can serve visual topological SLAM as a pre-existing or third-party map. Google Street View and other third-party map-based localization have been a topic of interest in recent literature [39, 40, 69]. A pre-existing map eliminates the mapping module from the SLAM problem. It relieves a portion of the computational burden imposed on the system providing it with more room for accurate localization. Eliminating the mapping module transforms the SLAM problem into a localization problem, i.e., finding the platform's location, using the sensor measurements, and a pre-existing map of the environment.



(a) Metric map



(b) Topological map



(c) Semantic map

Figure 2.2: Metric, topological and semantic maps of the same environment

2.1.4 Graph-based SLAM

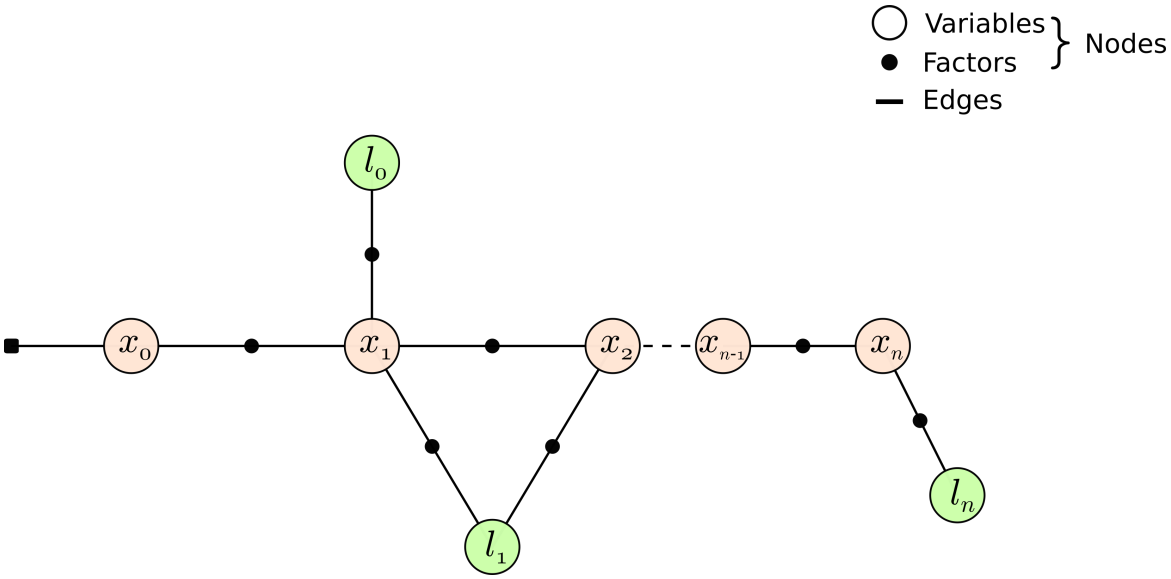
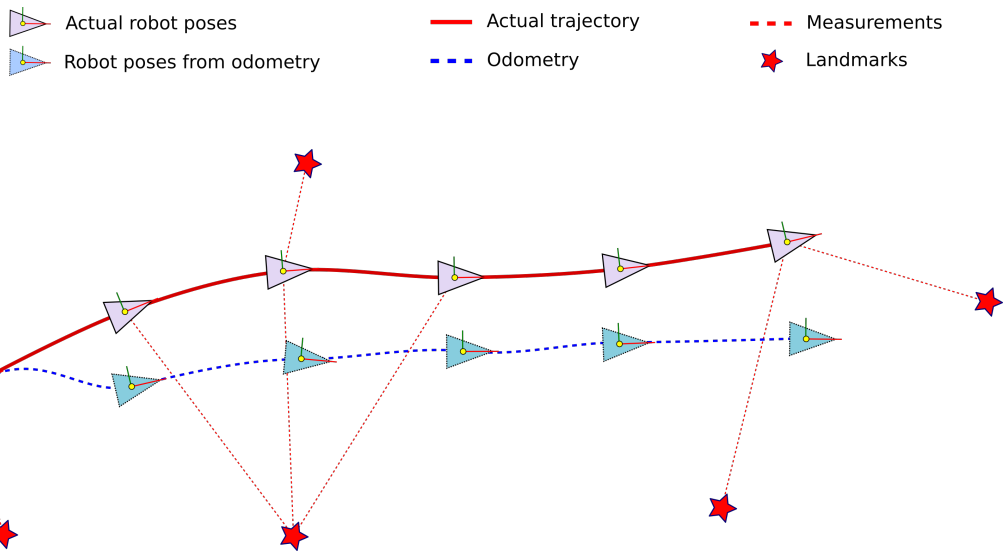
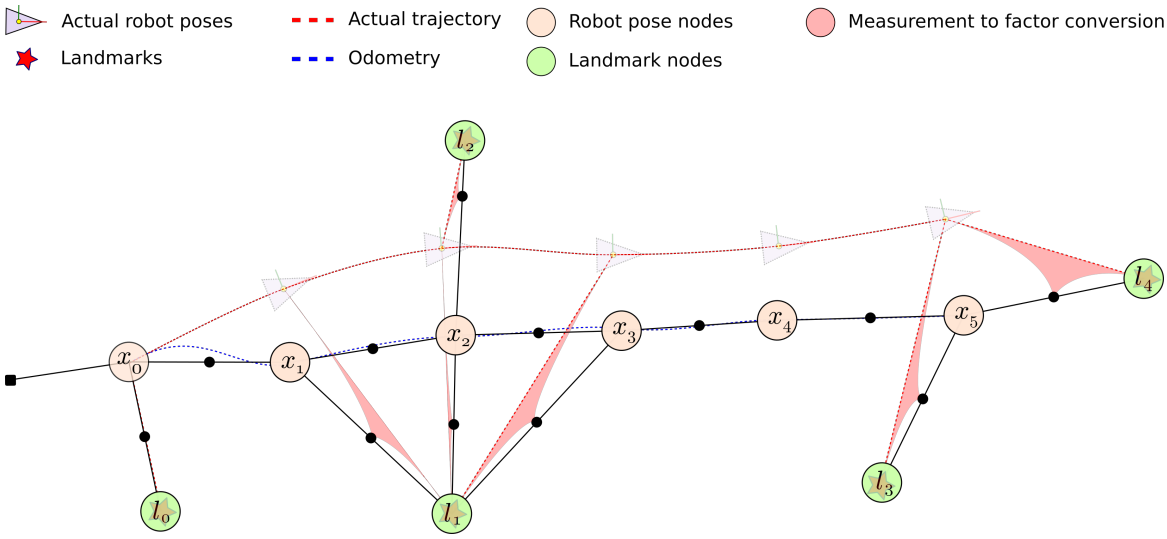


Figure 2.3: Structure of a typical factor graph

In graph-based SLAM [70], the robot's trajectory is considered as a sequence of distinct poses constrained via measurements obtained by various sensors (figure 2.4(a)). The unknown robot poses or landmark positions are represented as the nodes of a directed graph. The edges of the graph represent spatial constraints among the robot poses and landmark positions. These graphs, in general, appear as *factor graphs* (figure 2.3). A variant of this is a *pose graph*, in which the variables are the unknown poses along the robot's trajectory with factors denoting the constraints among poses (figure 2.4(b)). Due to the similarity in the representation, topological maps suit well for graph-based SLAM.

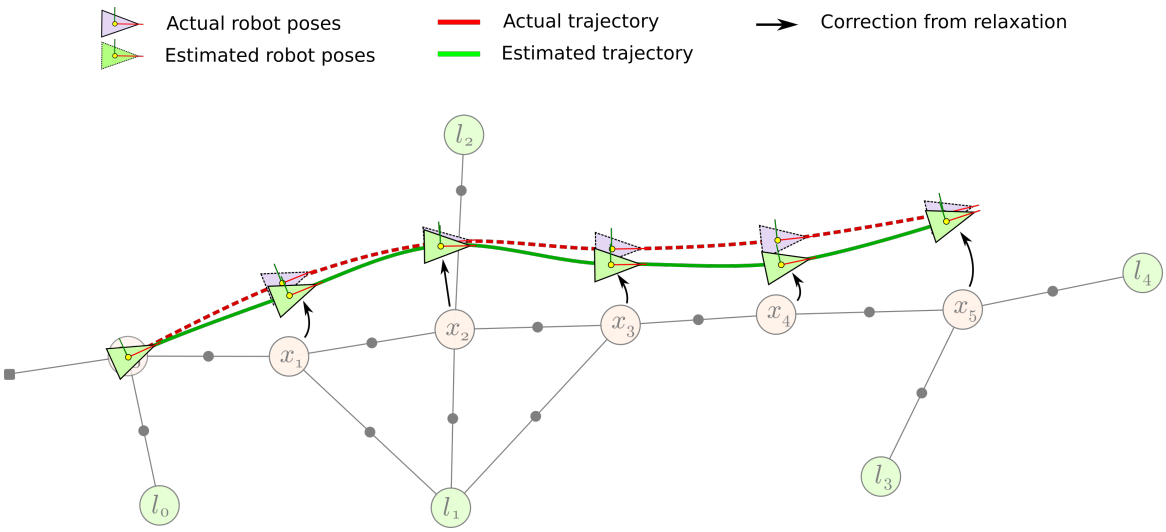


(a) Actual trajectory, odometry and landmarks



(b) Construction of the factor graph

Figure 2.4: The complete factor-graph process. In figure 2.4(b) the unknown robot poses and known landmark positions are indicated by x_0 - x_5 and l_0 - l_4 respectively. Although the constraints between x_2-l_1 and x_2-l_2 seems to be approximately equal, in figure 2.4(a) x_2-l_1 is shorter than x_2-l_2. Thus, x_2 is pulled towards l_2 while being pushed away from l_1. During the solution process node x_2 displaces to a position at which this push-pull effect is neutral. In this process node x_2 pulls nodes x_1 and x_3 towards l_2 and away from l_1. This is known as *relaxation*. The new positions of the nodes represents the estimated poses of the robot. From figure 2.4(c) (page 17), it can be seen that node x_2 which experiences the highest number of constraints results in the best estimate while node x_4 which has no constraints other than odometry experiences the highest deviation. Node x_0 which is a prior is identical to the actual robot pose.



(c) Relaxation of a factor graph

Figure 2.4: The complete factor-graph process

Optimization techniques are used in solving/relaxing (figure 2.4(c)) graph-based SLAM problems. Over the years a number of graph solvers GTSAM [71], Olson [72], TreeMap [73], Square Root SAM [74], iSAM [75], TORO [76], Sparse Pose Adjustment [77], iSAM2 [78], g2o [37], SLAM++ [79] and Ceres [80] have come into existence. Those are capable of solving graphs with very large numbers of nodes (variables) within a modest time that is suitable for robotic implementations. These algorithms are also capable of receiving topological maps as inputs.

Note

For the rest of the thesis, all discussions related to SLAM will be made in the context of visual topological SLAM.

2.2 Overview of a visual-topological-SLAM system

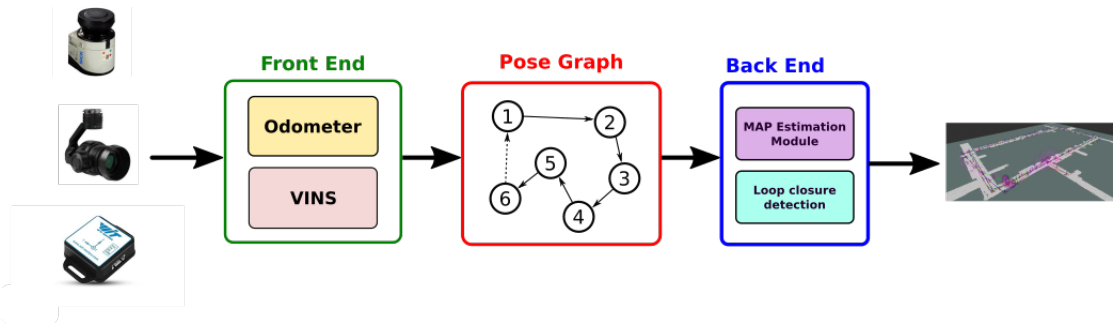


Figure 2.5: Structure of a visual topological SLAM system

A graphical overview of a SLAM system is presented in figure 2.5. It can be seen, that the architecture of a SLAM system, can be broadly classified into one of two categories, namely the *front end* and the *back end*. The front end is responsible for collecting data received from the sensors and fitting those into models that can be used for MAP estimation in the back end. Also, the front end is responsible for taking care of odometry. In the case of a wheeled robot, odometry can be achieved using the wheel encoders, and in the case of an aerial platform, a VINS can be used to serve the function of an odometer. The back end, on the other hand, performs inference on the processed data, deriving estimations. The following is a detailed discussion of the above.

2.2.1 The front end

The purpose of the front end is to capture the data received through the sensors, extract relevant types and amounts of data, and abstract those into mathematical forms that the back end can easily manipulate. It also handles the task of associating the sensor measurements to landmarks in the surrounding, where necessary. This task, in particular, is known as *data association*. In graph-based SLAM methods, the front end is also responsible for constructing, updating, and pruning the graph as well as estimating odometry factors using filters such as VINS, INS, and odometers. In visual SLAM, the front end has to execute additional tasks such as feature extraction, feature tracking, place recognition. Concerning optimization, the front end does the job of initializing the estimator functioning in the back end.

The data association task entails two distinct phases; *short-term data association* and *long-term data association* [81]. Short term data association deals with consecutive sensor data. Feature tracking and odometry belong to this category. In contrast, long term data association connects freshly obtained measurements to previously identified landmarks. This procedure is referred to as *loop closure*. It is the most crucial, yet it is the most challenging step to accomplish. There often exists a feedback loop connecting the back end to the front end, aiding the tasks such as loop closure detection, map updating, and factor-graph construction.

2.2.2 The back end

The SLAM back-end [82] is where a solution to a factor graph is determined. As mentioned earlier, the SLAM problem is formulated as a Maximum A Posteriori estimation problem. Maximum A Posteriori refers to, the "*set of variables that maximizes their belief, for a given a set of measurements and priors*". It can also be described as the set of variables that the conditional probability of the variables for a given set of measurements and priors is maximum. In the case of SLAM, the variables are the robot poses and the landmark positions. In the case of localization, the variables are only the robot poses. The prior knowledge and relative measurements (between previous poses or landmarks) available are represented as the *factors* of the factor graph. Once the measurement and motion models of the system are known, the MAP estimation problem can be described as a least-squares minimization problem, which can then be solved using an optimization algorithm such as *Lavenberg-Marquardt* (LM) [83] or *Powell's Dog Leg* (PDL) [84].

2.3 Visual topological localization

Vision-based topological localization and mapping method employ visual place recognition as the mechanism for loop closure. A camera extracts useful visual information from the environment, and then those are compared against a known set of information that belongs to the same environment the robot navigates. Visual topological localization and mapping has appeared in different forms depending on the type of feature descriptor used, the detector- descriptor combination used, search method used, and by the kind of third-party map employed for the purpose.

Singh and Kosecka [85], in their work, have used the *Generalized Search Tree* (GiST) descriptors with images of urban environments captured through an omnidirectional camera. The panoramas they obtained are represented by four perspective images with a *Field of view* (FOV) of 90°. The gist descriptors of the panoramas are extracted by applying the algorithm on these four images. They have introduced a novel similarity measure for assessing the similarity between the image panoramas when using the gist descriptor and evaluated its efficiency in loop closure detection. Murillo et al. [86] have conducted a similar system by incorporating a version of gist descriptor known as Omni-gist with catadioptric cameras. They have made improvements on the similarity measure of the descriptors proposing a hierarchical topological mapping and localization system. Rituerto et al. [87], in their work, have used Omni-gist for semantic labeling in building indoor topological maps. They have classified the nodes and the edges of the map as *places* or *transitions*, respectively. This place classifier was later integrated with a *Hidden Markov Model* (HMM). Liu et al. [88] used a dimensionality reduction on the Gist descriptor using *Principal Component Analysis* (PCA), which improved the efficiency of the system. The loop closure detection was based on a particle filter that used these descriptors for the update step. They have shown that using only a few particles, a high recall at 100% precision can be achieved.

Liu et al. [89] applied the *Fast Adaptive Colour Tags* (FACT) descriptor for indoor topological mapping using omnidirectional cameras. The observation that the vertical edges in indoor environments divide the environment into meaningful cuts was used to introduce a region descriptor known as a *tag*. It is dictated by the average color value of each cut and the width of the region. These tags are then concatenated into a vector known as a scene descriptor. The 2D Euclidean distance between the color descriptors and recursive comparison among the region widths was used for scene matching. The authors presented an upgraded version of FACT known as DP-FACT in [90].

Milford et al. [91], in their work, presented a novel method named SeqSLAM, which is based on *Sum of Absolute Differences* (SAD), which they showed invariant to weather and over season changes. In [67], they showed that route recognition is achievable even by using as low as few bits per image. They also conducted studies as to how the length of the sequences affected the performance if SeqSLAM. Pepperell et al. [92] proposed a method named *Sequence Matching Across Route Traversals*

(SMART), which was better at handling substantial changes in perceptual change and translational pose. This improvement was achieved through a development made to seqSLAM by incorporating self-motion information and new image matching methods.

An exciting contribution is a work by Wu et al. [93]. They introduced a novel method for loop closure, which they claim can detect loop closures in a map of 20 million key locations. Surprisingly, the underlying methodology was a straightforward image representation which was based on Gaussian smoothing and Otsu's thresholding.

Among the recent work based on local descriptors, Zhang et al. [94] introduced a method known as *Bag-of-Raw-Features* (BoRF). It was a scale based feature selection method, where sets of such features that can be matched among consecutive images provided a representation of a location. Nevertheless, this method had a drawback that the number of features escalated with every added image, which was arduous for the linear search process. The authors later had overcome this by introducing a kd-tree-based indexing structure. Johns et al. [95] presented a method of constructing maps spanning the space between nodal images by detecting landmarks across multiple images. This method was capable of generating dense continuous topological indoor maps without compromising speed. The authors also presented a probabilistic indoor localization system based on the properties of the landmarks recognized.

Kawewong et al. [96] presented an incremental appearance-based SLAM method known as PIRF-Nav, which was based on a novel type of features called *Position-Invariant Robust Features* (PIRF), that the authors had invented as part of their previous work [97]. Despite the improvements made, it had a significant setback in terms of high computational cost. The work of Tongprasit et al. [98] too involved PIRF, but with some modifications to improve computational cost. They came with an improved SLAM algorithm called PIRF-Nav 2, which proved to be 12 times faster than the original version, but with a minor decline in recall percentage. All the above work on PIRF was based on omnidirectional images.

Valgren et al. [99] made an importing discovery while investigating on long-term outdoor topological localization. In their comparison between SIFT and SURF, they figured out that SURF was the better option for outdoor topological localization. In a similar study, Ascani et al. [100] concluded that in terms of indoor topological

localization, SIFT was the better option.

Bacca et al. [101] used a model that was influenced by human memory. The models were implemented using *Feature Stability Histogram* (FSH), which records the frequency in which each feature has been observed. An improved version of this is presented in [102], applying it for SLAM.

Romero et.al [103] proposed a method based on *Graph Transformation Matching* (GTM) for constructing topological maps. In this approach graphs of grouped invariant features, extracted from image segments, are matched considering the description and structure of the features.

Garcia et al. [104] proposed an appearance-based visual mapping and localization method with a discrete Bayes filter running in its core. They came up with a new image similarity measure based on the number of feature matches and the associated distances. They improved the running time of their method by utilizing a randomized kd-tree-based indexing system. The redundancies of the generated maps were eliminated using a framework they presented in [105], which saved storage spaces and boosted the speed of the localization system.

Upon understanding the pros and cons of each feature detector and descriptor, some researchers have made attempts to use the best of both worlds by combining different detectors and descriptors. Wang et al. [106] combined Harris-Laplace features and *Scale Invariant Feature Transform* (SIFT) descriptors in creating a global descriptor named *Orientation Adjacency Coherence Histogram* (OACH). Their approach was creating two separate databases for OACH Moreover, SIFT, for coarse and fine localization, respectively. The procedure was to isolate a set of candidate images during the global localization phase and use those in the fine localization phase. The verification step involved the typical RANSAC based fundamental matrix estimation. Another approach by Chapoulie et al. [107] developed an outdoor loop closure detection algorithm by using SIFT as local features and their distribution histogram as local features. These were then merged within a Bayes filter to detect loop closure candidates. This approach stands out from the rest since the images used in the localization process were spherical. Similar work was carried out by Wang et al. [108], where they combined Harris detector and SIFT the descriptor in their proposed local recognition

system, which was instead designed for monocular cameras. Korrapati et al. [109] presented an outdoor topological mapping module by using *Vector of Locally Aggregated Descriptors* (VLAD), which allowed them to create maps consisting of over 11,000 images.

2.4 Visual place recognition

Visual Bag-of-Words based methods are another category under which localization and mapping techniques can be studied. Bag-of-Words is a text-retrieval technique that was adapted to robotics. It works by extracting features from images, retaining only the feature descriptors by dropping the image representation, and quantizing the descriptors into clusters to which a number is assigned. This number is referred to as a *visual word*. Bag-of-Words is used as a technique for improving the execution time of localization systems. By using kd-tree-based search methods and optimized search methods such as *Fast Library for Approximate Nearest Neighbors* (FLANN), the execution time can be improved further. Therefore, this concept has been adapted in a considerable amount of work-related to localization [95, 105, 110–126].

A recent trend in localization is the use of third-party maps. A unique case among these is the use of Google Street View. Although Google Street View has been used for place recognition related research, to the best of our knowledge, Majdik et al. [40] was the first of its kind in localization. They successfully conducted an outdoor localization in an urban environment using a MAV. This study resulted in a couple of crucial discoveries.

Although feature descriptors such as SIFT [127], *Binary Robust Invariant Scalable Keypoints* (BRISK) [128] and *Binary Robust Independant Elementary Features* (BRIEF) [129] are invariant to scale, rotation, and a certain degree of affine transformation, they are vulnerable to large viewpoint changes ($\theta > 45°$). Due to the significant distortions present in Google Street View images, those cannot be used directly in creating visual vocabularies. Work in [40] provided a solution to this by using *virtual views* out of the Street View images. Another observation was the issue of using RANSAC for outlier rejection. The authors claim that algorithms such as RANSAC [130] works

robustly, only for outlier ratios less than 50%. But, in the case of Google Street View, the outlier ratio can be as high as 90%. They concluded that a solution to this is to use modified versions of RANSAC such as *Optimized Random Sampling Algorithm* (ORSA) [131] or *Virtual Line Descriptors* (VLD) [132].

Another application of Google Street View in localization is the work by Agarwal et al. [39], in which they used a ground robot for metric localization. Their approach was to download perspective images of known heading and FOV from Google Street View images in such a way that the images did overlap at their boundaries and spanned 360°. They had used a Google provided *Application Programming Interface* (API) [133] for this. Their solution was modeled a two-phase non-linear least squares estimation, which resulted in sub-meter accuracy. Similar work can be seen in [42,43].

The method proposed in this thesis uses SIFT + BRIEF as the detector-descriptor combination. SIFT features were chosen due to its invariance to translation, scale, and rotation. BRIEF descriptor was chosen, as it is one of the fastest feature descriptors in existence. *Georgia Tech Smoothing And Mapping* (GTSAM) library was used to construct the factor-graph underlying the system since it is equipped with all the tools necessary to implement the intended system.

To summarize, SLAM can be discussed under different branches based on the environment of operation, the method of estimation used, the type of map used and graph-SLAM. Visual-topological SLAM has become an interesting domain for many in the research community over the years. A recent trend in this area is the use of third party maps to eliminate the mapping module, converting the SLAM problem into a localization problem. Several attempts have been made in utilizing Google street view for localization. But these are mainly outdoor studies. Besides, the most successful use of Google street view has been achieved for visual place recognition. However, using Google indoor street view for visual-topological localization appears to be an area that has not been adequately studied. The method proposed in this thesis is to use visual place recognition and factor-graphs together with Google indoor street view in developing an indoor localization system.

2.5 Camera - Camera pose transformation

Shown in figure 2.6 are two tripods placed apart from one another, each to which a camera is attached. T_1, T_2, C_1 and C_2 are the body-fixed frames of the tripod 1, tripod 2, camera 1 (mounted on tripod 1) and camera 2 (mounted on tripod 2) respectively. Figure 2.7 provides a simplified version of figure 2.6 indicating all the coordinate frames and the transformations among those.

Figure 2.6: Tripod 1 (Left), tripod 2 (Right) and the cameras

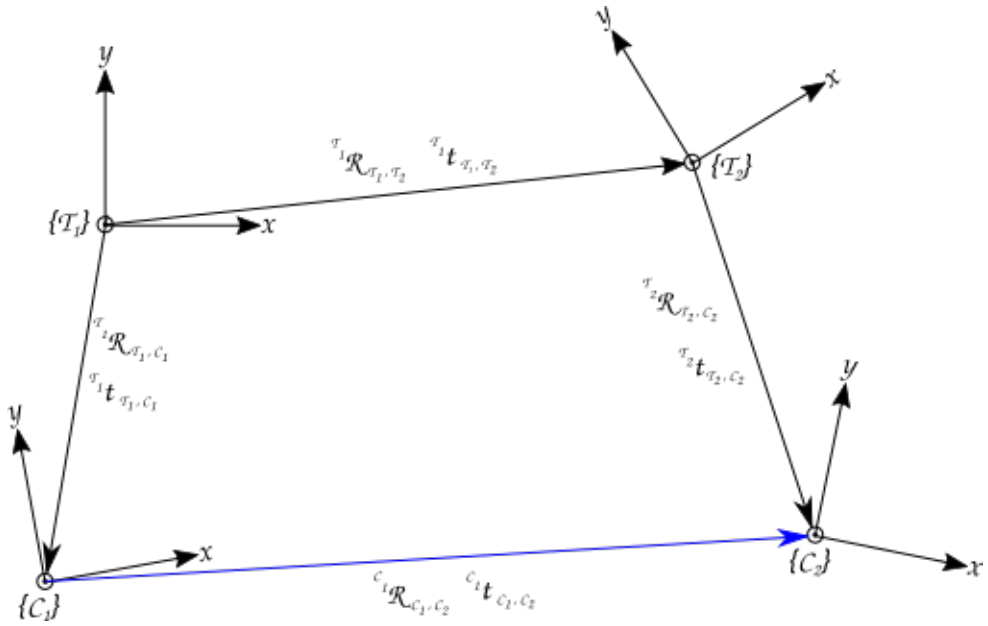


Figure 2.7: Coordinate frames and transformations

${}^{T_1}R_{T_1,C_1}$, ${}^{T_1}R_{T_1,T_2}$, ${}^{T_2}R_{T_2,C_2}$, ${}^{T_1}t_{T_1,C_1}$, ${}^{T_1}t_{T_1,T_2}$, ${}^{T_2}t_{T_2,C_2}$, ${}^{C_1}R_{C_1,C_2}$ and ${}^{C_1}t_{C_1,C_2}$ are defined as follows.

${}^{T_1}t_{T_1,C_1}$ - The position of C_1 w.r.t T_1 expressed in T_1.

${}^{T_1}t_{T_1,T_2}$ - The position of T_2 w.r.t T_1 expressed in T_1.

${}^{T_2}t_{T_2,C_2}$ - The position of C_2 w.r.t T_2 expressed in T_2.

${}^{C_1}t_{C_1,C_2}$ - The position of C_2 w.r.t C_1 expressed in C_1.

${}^{T_1}R_{T_1,C_1}$ - The orientation of C_1 w.r.t T_1 expressed in T_1.

${}^{T_1}R_{T_1,T_2}$ - The orientation of T_2 w.r.t T_1 expressed in T_1.

${}^{T_2}R_{T_2,C_2}$ - The orientation of C_2 w.r.t T_2 expressed in T_2.

${}^{C_1}R_{C_1,C_2}$ - The orientation of C_2 w.r.t C_1 expressed in C_1.

The position and orientation of the cameras w.r.t to the tripods are known, and the relative position and orientation of the tripod two w.r.t to tripod one can be measured. Thus, the position and orientation of the camera two w.r.t the camera one as expressed in camera 1 (indicated by the red color arrow in figure 2.6) can be derived from equations 2.1 and 2.2 respectively.

$$
\begin{aligned}
{}^{C_1}t_{C_1,C_2} &= {}^{C_1}t_{C_1,T_1} + {}^{C_1}t_{T_1,C_1} + {}^{C_1}t_{T_1,C_1} && (2.1)\\
{}^{C_1}R_{C_1,C_2} &= \left({}^{T_1}R_{T_1,C_1}\right)^T \left({}^{T_1}R_{T_1,T_2}\right)\left({}^{T_2}R_{T_2,C_2}\right) && (2.2)
\end{aligned}
$$

where,

$$
\begin{aligned}
{}^{C_1}t_{C_1,T_1} &= -\left({}^{T_1}R_{T_1,C_1}\right)^T \left({}^{T_1}t_{T_1,C_1}\right)\\
{}^{C_1}t_{T_1,C_1} &= \left({}^{T_1}R_{T_1,C_1}\right)^T \left({}^{T_1}t_{T_1,T_2}\right)\\
{}^{C_1}t_{T_2,C_2} &= \left({}^{T_1}R_{T_1,C_1}\right)^T \left({}^{T_1}R_{T_1,T_2}\right)\left({}^{T_2}t_{T_2,C_2}\right)
\end{aligned}
$$

2.6 Equivalent intrinsic matrix

In the pinhole camera model shown in figure 2.8, θ_H is is the horizontal field-of-view, θ_V is the vertical field-of-view, f is the focal length, W_{FOV} is the horizontal field-of-view in length units, d_{FOV} is the object distance(distance to the object from the optic center) in length units and W and H be the width and height of the image formed respectively [70].

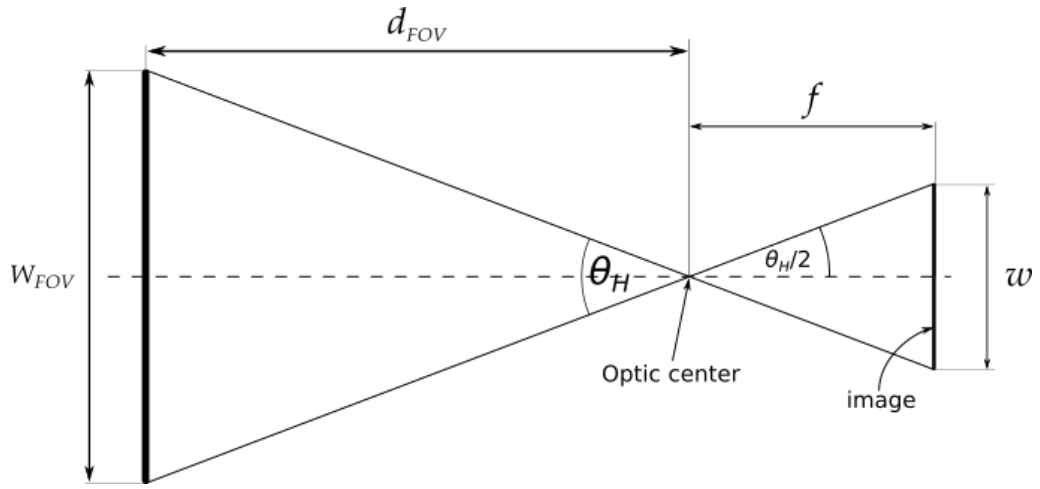


Figure 2.8: Image formation in a pin hole camera model

From basic trigonometry,

$$\tan\left(\frac{\theta_H}{2}\right) = \frac{\frac{W}{2}}{f} = \frac{W_{FOV}}{d_{FOV}} \tag{2.3}$$

rearraging the terms,

$$f = \frac{W}{2\tan\left(\frac{\theta_H}{2}\right)} \tag{2.4}$$

If the focal length of the camera is the same for both vertical and horizontal directions (i.e. $f_x = f_y$), similar to equation 2.4,

$$f = \frac{H}{2\tan\left(\frac{\theta_V}{2}\right)} \tag{2.5}$$

Thus, from equations 2.4 and 2.5, the constraint between θ_H and θ_V, for having the same focal length can be derived to be that shown in equation 2.6.

$$\theta_V = 2.tan^{-1}\left[\left(\frac{H}{W}\right)tan\left(\frac{\theta_H}{2}\right)\right] \tag{2.6}$$

Let $r = \left(\frac{H}{W}\right)$ be the *aspect ratio* of the image. Then, from equation 2.6,

$$\theta_V = 2.tan^{-1}\left[r.tan\left(\frac{\theta_H}{2}\right)\right] \tag{2.7}$$

Thus, the intrinsic matrix equivalent to a given image, K can be represented in matrix

form, as shown by equation 2.8.

$$K = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.8}$$

where $c_x = \frac{W}{2}$ and $c_y = \frac{H}{2}$.

Chapter 3

Factor Graph

The success of a navigation system depends on its ability to utilize the available measurements in combination with prior knowledge, producing an accurate estimation of its current state. This chapter presents the process of correcting the trajectory traced by a robot, that is different from the actual trajectory due to sensor noises and odometer drift. The problem is very similar to a PoseSLAM, except for the availability of a pre-constructed map.

Here, an overview of factor graphs is presented, which is to be followed by a couple of underlying principles of factor-graph based methods. Next, a detailed discussion of the followed procedure is presented, followed by its algorithm structure. Then, two custom-built simulators are presented which is succeeded by the results of the simulations conducted through each of them, and the conclusions rested upon those.

3.1 Introduction

3.1.1 Bayesian Networks

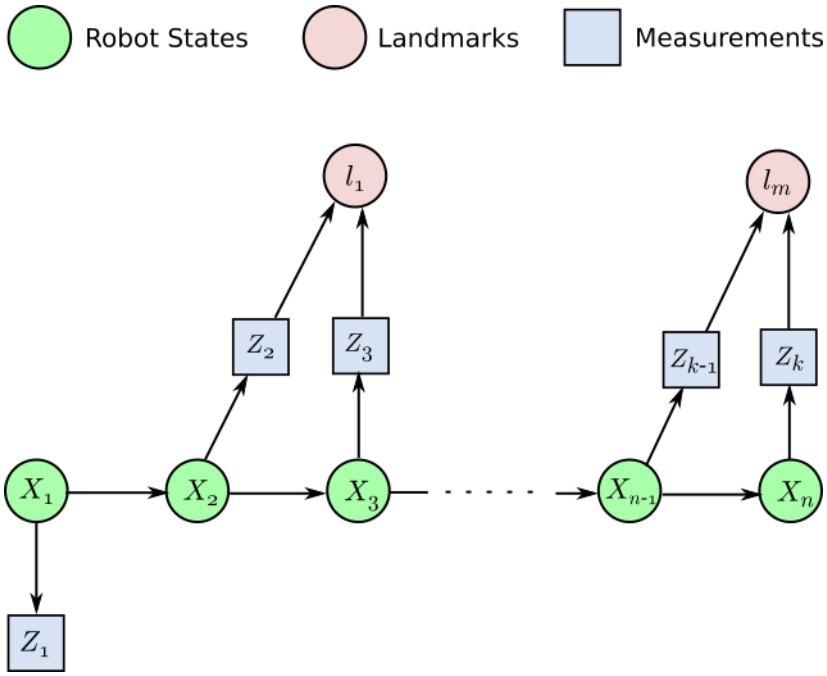


Figure 3.1: Bayes' net of a general SLAM problem

Bayes net is a directed graphical model in which nodes represent variables θ_j. If an entire set of random variables of interest is denoted by $\Theta = \{\theta_1, \theta_2, \ldots \theta_n\}$, a Bayes net defines a joint probability density $p(\Theta)$ over the entire set Θ is given by,

$$p(\Theta) \triangleq \prod_j p(\theta_j|\pi_j) \tag{3.1}$$

where $p(\theta_j|\pi_j)$ is the conditional density associated with node θ_j and π_j is the assignment of values to the parents of θ_j. Factorization of the joint density can be achieved by exploiting the graph structure (the *node-parent* relationship). For a SLAM problem the random variable of interest are $\Theta = \{X, Z\}$, where X is the unknown poses and landmarks and Z are the measurements. Thus, the joint density $p(X, Z)$ can be represented as a product of conditional densities.

$$
\begin{aligned}
p(X,Z) &= p(x_1, x_2, \ldots x_n, l_1, l_2, \ldots, l_m, z_1, z_2, \ldots, z_k) \\
&= p(x_1)p(x_2|x_1)p(x_3|x_2)\ldots p(x_n|x_{n-1}) \\
&\times \; p(l_1)p(l_2)\ldots p(l_m) \\
&\times \; p(z_1|x_1) \\
&\times \; p(z_2|x_1, l_1)p(z_3|x_2, l_1)\ldots p(z_k|x_n, l_m) \qquad (3.2)
\end{aligned}
$$

The factorization shown in equation 3.2 represents four distinct types factors. $p(x_1)p(x_2|x_1)p(x_3|x_2)\ldots p(x_n|x_{n-1})$ is a Markov chain on $x_1, x_2, \ldots, x_n$. The conditional densities $p(x_{t+1}|x_t)$ can represent either prior knowledge or can be derived through control inputs. $p(l_m)$ on the landmarks are Prior densities. This is an important factor in cases where a prior map is available, but can be ignored if not. The conditional denisty $p(z_1|x_1)$ corresponds to the absolute pose measurement of the first pose x_1. This ties the graph structure to the physical environment preventing a floating graph structure. It does not necessarily have to be the first pose measurement although in most of the practical cases it is. $p(z_2|x_1, l_1)p(z_3|x_2, l_1)\ldots p(z_k|x_n, l_m)$ represents the bearing factors on the landmarks $l_1, l_2, \ldots, l_m$ from the poses $x_1, x_2, \ldots, x_n$

3.1.2 Factor Graphs

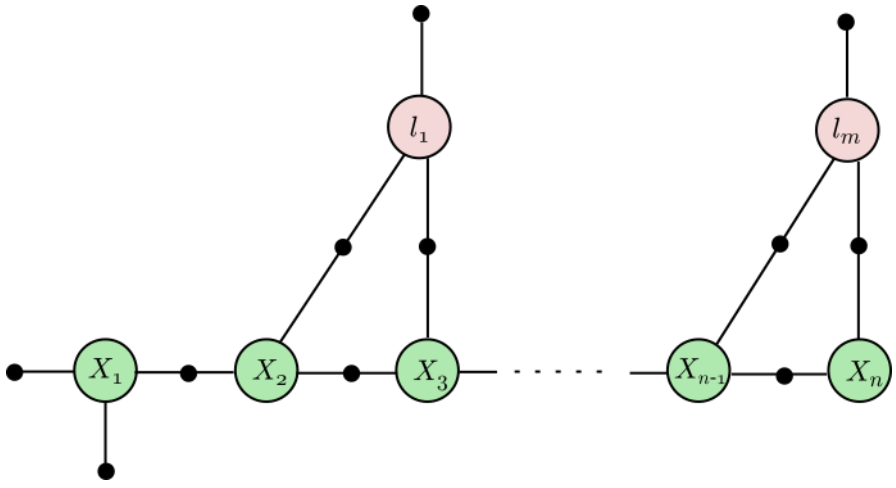


Figure 3.2: Factor graph of the Bayes net in figure 3.1

A factor graph is a graph structure that represents unknown variables connected to factors that encode probabilistic information. Factor Graphs can be used to represent the joint probability as a product of factors. They can be used to represent any factored function over a set of variables. Generally, any Bayes net can be converted into a factor graph.

In Factor graphs, unkonwn variables (Poses and landmarks) and measurements are represented using two distinct types of nodes. Factor graph is a bipartite graph $F = (\mathcal{U}, \mathcal{V}, \mathcal{E})$ with two types of nodes which are factors $\phi_i \in \mathcal{U}$ and variables $x_j \in \mathcal{V}$. Edges $e_ij \in \mathcal{E}$ always lie between a facor node and a variable node. The set of variable nodes adjescent to a factor ϕ_i is denoted ny $\mathcal{N}(\phi_i)$. The factorization of a global function $\phi(X_1, X_2, X_3, \ldots, X_n)$ by a factor graph can be represented as follows.

$$\phi_i(X_1, X_2, X_3, \ldots, X_n) = \prod \phi_i(\mathcal{X}_i) \tag{3.3}$$

The Maximum A Posteriori inference on a factor graph, is to maximize the product given by equation 3.3, which is nothing but the value of the factor graph.
Thus, similarly equation 3.2 can be rewritten as a factor graph factorization as in equation 3.4.

$$\begin{aligned} \phi(l, x) &= \phi(x_1)\phi(x_2|x_1)\phi(x_3|x_2)\ldots\phi(x_n|x_{n-1}) \\ &\times \phi(l_1)\phi(l_2)\ldots\phi(l_m) \\ &\times \phi(z_1|x_1) \\ &\times \phi(z_2|x_1, l_1)\phi(z_3|x_2, l_1)\ldots\phi(z_k|x_n, l_m) \end{aligned} \tag{3.4}$$

where, $\phi(l, x) \triangleq \phi(l_1, l_2, \ldots, l_m, x_1, x_2, \ldots, x_n)$
The mathematical formulation of the 2D Pose SLAM problem can be divided into four parts: Derivation of the measurement function, formulation of SLAM as an optimization problem, Linearization, and Solving.

3.1.3 PoseSLAM

In PoseSLAM [71], the concern is regarding all the poses in the trajectory of the robot that needs to be reconstructed. In such cases, typically, two types of factors

are involved: unary factors and binary factors. Unary factors depend only on the current pose, whereas binary factors depend on two successive poses. Pose priors and absolute pose measurements such as GPS are examples of unary factors, while relative pose constraints derived through vision and odometry are examples of binary factors. In the factor graph shown in figure 3.3, $f_0(x_1)$ represents a unary factor. It is an absolute measurement and anchors the factor graph to the physical world. Such factors are known as prior factors. Factors represented as $f(x_t, x_{t+1})$ are relative measurements obtained through odometry. An important constraint in a PoseSLAM problem is the loop closure. In figure 3.3, this is denoted by $f(x_5, x_2)$. It occurs when the robot recognizes a previously visited location through vision or laser range finders and computes the pose constraint between the initial and current poses.

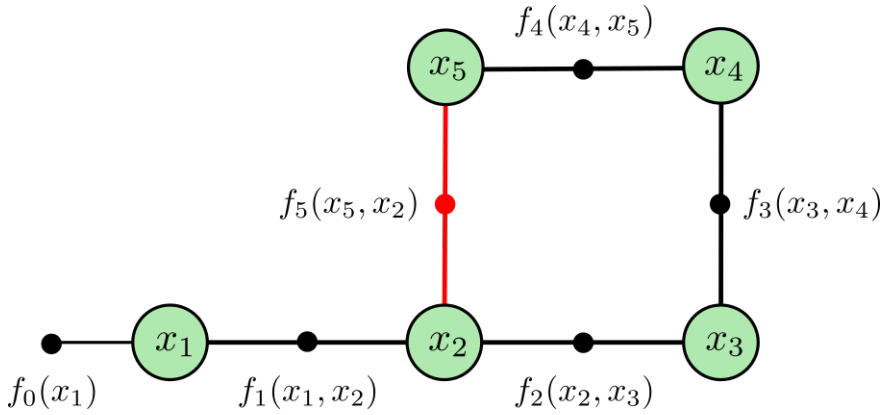


Figure 3.3: The trajectory which was optimized using unary factors.

The objective function of a PoseSLAM encodes both the measurement model and the motion model of the system, and can be written as,

$$\begin{aligned} \Xi^* &= \underset{\Xi}{\operatorname{argmin}} \Bigg\{ \sum_i \|h(x_i) + H_i\xi_i - z\|^2_\Sigma \\ &+ \sum_k \|g(x_i, x_j) + F_i\xi_i + G_j\xi_j - z\|^2_\Sigma \Bigg\} \end{aligned} \tag{3.5}$$

where $\Xi \triangleq \{\xi_i\}$ is the set of incremental pose coordinates, h is the unary measurement function, g is the binary measurement function, and H_i, F_i and G_j are the respective jacobians.

The solver minimizes the objective function over local coordinates of all the poses to determine the optimal value for Ξ.

3.2 Preliminaries

3.2.1 The measurement function

Consider two poses $\xi_1 = (x_1, y_1, \theta_1)^T$ and $\xi_2 = (x_2, y_2, \theta_2)^T$. Let $\Delta\xi = (\Delta x, \Delta y, \Delta z)$ be the relative pose measurement resulting from the difference between ξ_1 and ξ_2. Also, let $\xi_1, \xi_2 \in GL(3)$. Let,

$$T_1^w = \begin{bmatrix} R_1 & t_1 \\ 0 & 1 \end{bmatrix}$$

$$T_2^w = \begin{bmatrix} R_2 & t_2 \\ 0 & 1 \end{bmatrix}$$

where the translation $t_i = (x_i, y_i)$ and the rotation matrix R_i,

$$R_i = Rot(\theta_i) \triangleq \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) \\ sin(\theta_i) & cos(\theta_i) \end{bmatrix}$$

The relative pose between ξ_1 and ξ_2 is,

$$\begin{aligned} T_2^1 &= (T_1^w)^{-1} T_2^w \\ &= \begin{bmatrix} R_1 & t_1 \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} R_2 & t_2 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} R_1^T & -R_1 t_1 \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} R_2 & t_2 \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} R_1^T R_2 & R_1^T (t_2 - t_1) \\ 0 & 1 \end{bmatrix}^{-1} \end{aligned}$$

From $R_1^T = Rot(-\theta_1)$ and $R_1^T R_2 = Rot(\theta_2 - \theta_1)$, the measurement function $h(\xi_1, \xi_2)$,

$$\widehat{\Delta\xi} = h(\xi_1, \xi_2) = \begin{bmatrix} Rot(-\theta_1)(t_2 - t_1) \\ \theta_2 - \theta_1 \end{bmatrix} \tag{3.6}$$

Thus, equation 3.6 can be used to used to determine the relative pose measurement between ξ_1 and ξ_2 when imposing constraints among them.

3.2.2 SLAM as an optimization problem

Suppose there are many relative pose measurements, $\Delta\xi_i$. To determine the optimal set of poses, the following objective function can be defined.

$$E(X) = \frac{1}{2}\sum_i \|h(\xi_1, \xi_2) - \Delta\xi_i\|^2 \tag{3.7}$$

where $\Delta\xi_i$ is the measured relative pose beteween ξ_1, ξ_2.
By minimizing this cost function the optimal set of pose $X = \{\xi_j\}_{j=1}^n$ can be determined.

3.2.3 Linearization

The cost function 3.7 cannot be optimized as it is due the non-linear nature of $h(,)$. Thus, using Taylor series expansion $h(,)$ can be linearized as shown below.

$$h(\xi_1 \oplus \delta_1, \xi_2 \oplus \delta_1) = h(\xi_1, \xi_2) \oplus \{H_1\delta_1 + H_2\delta_@\} \tag{3.8}$$

where $\delta_1, \delta_2 \in \mathbb{R}^3$ are pose updates and H_1 and H_2 are 3×3 Jacobian matrices, where as $\xi \oplus \delta$ is defined in the coordinate frame of ξ such that,

$$\xi \oplus \delta = \begin{bmatrix} t + Rot(\theta)\delta t \\ \theta + \delta\theta \end{bmatrix}$$

$$\begin{bmatrix} Rot(\theta + \delta\theta) & t + Rot(\theta)\delta t \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} Rot(\theta) & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} Rot(\delta\theta) & \delta t \\ 0 & 1 \end{bmatrix}$$

It can be shown that the Jacobians,

$$H_1 = \begin{bmatrix} R_2^T R_1 & Rot(-\frac{\pi}{2})R_2^T(t_1 - t_2) \\ 0 & 1 \end{bmatrix} \tag{3.9}$$

$$H_2 = \begin{bmatrix} I & 0 \\ 0 & 1 \end{bmatrix} \tag{3.10}$$

3.2.4 Solution

From equations 3.8 and 3.7, a linar objective function of δ , $E(\delta)$ can be derived.

$$\begin{aligned} E(q) &= \frac{1}{2}\sum_i \|h(\xi_{j1},\xi_{j2}) + H_{j1}\delta_{j1} + H_{j2}\delta_{j2} - \Delta\xi_i\|^2 \\ &= \frac{1}{2}\sum_i \|A_i\delta - b_i\|^2 \\ &= \frac{1}{2}\|A\delta - b\|^2 \end{aligned}$$

where $A_{3m\times 3n}$ is the measurement Jacobian matrix such that

$$A = [\ A_1\ A_2\ \dots\ A_i\ \dots\ A_n\ ]^T$$

where A_i is a 3 $\times$ 3m matrix such that,

$$A_i = [\ \dots\ H_{j1}\ \dots\ H_{j2}\ \dots\ ]^T$$

and b is a 3m $\times$ 1 matrix consisting of n no. of 3 $\times$1 prediction errors b_i where,

$$b_i = h(\xi_{j1},\xi_{j2}) - \Delta\xi_i$$

Two popular optimization algorithms are used to solve problems of this nature: LM optimizer and the PDL optimizer. In general PDL is a better option than LM since the computational cost of LM algorithm increases if a step gets rejected. However, on the other hand, for applications where the system is under-constrained (insufficient measurements) or numerically poorly constrained, LM is a better option [71].

3.2.5 Covariance ellipsoid

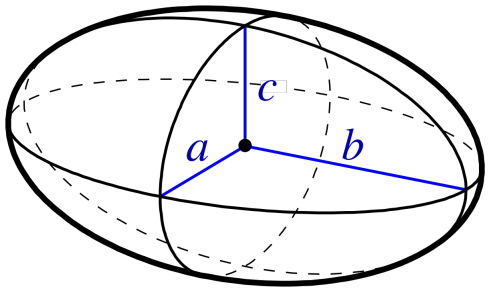


Figure 3.4: Semi-axes of an ellipsoid

Let ${}^{W}\Sigma$ be covariance matrix for the position of the robot expressed in the world frame and σ_x, σ_y, σ_z be the standard deviation in position along X, Y and Z coordinate axes respectively.

$$ {}^{W}\Sigma = \begin{pmatrix} {\sigma_x}^2 & \sigma_x\sigma_y & \sigma_x\sigma_z \\ \sigma_y\sigma_x & {\sigma_y}^2 & \sigma_y\sigma_z \\ \sigma_z\sigma_x & \sigma_z\sigma_y & {\sigma_z}^2 \end{pmatrix} $$

An axis aligned origin centered ellipsoid can be geometrically represented as,

$$ \left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 + \left(\frac{z}{c}\right)^2 = 1 \tag{3.11} $$

where a, b and c be the semi-axis lengths(radii) along X, Y, Z axes respectively(figure 3.4). If the ellipsoid represents positional uncertainty, the semi-axis lengths will be the standard deviations parameterized by a scale factor k.

$$ \therefore \left(\frac{x}{\sigma_x}\right)^2 + \left(\frac{y}{\sigma_y}\right)^2 + \left(\frac{z}{\sigma_z}\right)^2 = k \tag{3.12} $$

Here, k is referred to as the *Mahalanobis radius* of the ellipsoid. Since k is the sum of the squares of three Gaussian distributed data samples, if follows a chi-square(χ^2) distribution. Thus, the value of k corresponding to a confidence of p satisfies

$$ P(z > k) = p \tag{3.13} $$

with the number of *degrees of freedom* equal to 3 [134]. It can also be shown that the eigenvalues of it give the semi-axis lengths of the covariance ellipsoid for an unscaled

covariance matrix. In contrast, the directions of its axes are given by the corresponding eigenvectors.
Therefore, if the eigenvalues and eigenvectors of ${}^{W}\Sigma$ are λ_1, λ_2, λ_3 and $\underline{v_1}$, $\underline{v_2}$, $\underline{v_3}$ respectively, the semi-axis lengths of the covariance ellipsoid of ${}^{W}\Sigma$ for a confidence of p are given by $k.\sqrt{\lambda_1}$, $k.\sqrt{\lambda_2}$ and $k.\sqrt{\lambda_3}$ and its orientation can be represented by the rotation matrix $[\underline{v_1} \;\; \underline{v_2} \;\; \underline{v_3}]^T$.
If the covariance matrix of the robot position is expressed in the body frame as ${}^{B}\Sigma$ and the robot's orientation at that position with respect to the world is ${}^{W}R_B$, then, ${}^{W}\Sigma$ and ${}^{B}\Sigma$ are related as shown by equation 3.14

$$ {}^{W}\Sigma \;=\; ({}^{W}R_B)\;({}^{B}\Sigma)\;({}^{W}R_B)^T \tag{3.14} $$

3.3 Methodology

The approach taken relied on a couple of requirements. First, the existence of a pre-constructed map was assumed. Next, it was assumed that the starting position of the robot is known. In terms of measurements, the existence of odometric measurements, either by wheel encoders of a VINS, was assumed. For the simulations, a trajectory similar to that of figure 3.5 was used. These contributed to the factor graph as prior factors and between factors, respectively.

First, a non-linear factor graph comprised of map nodes and robot poses was created. These can be seen in figures 3.7 and 3.8 respectively. The graph was initialized with the robot poses with appropriately tuned odometer noise figures to simulate the drifted odometer path. These were utilized to initialize the factor graph to simulate the robot motion with odometric measurements.

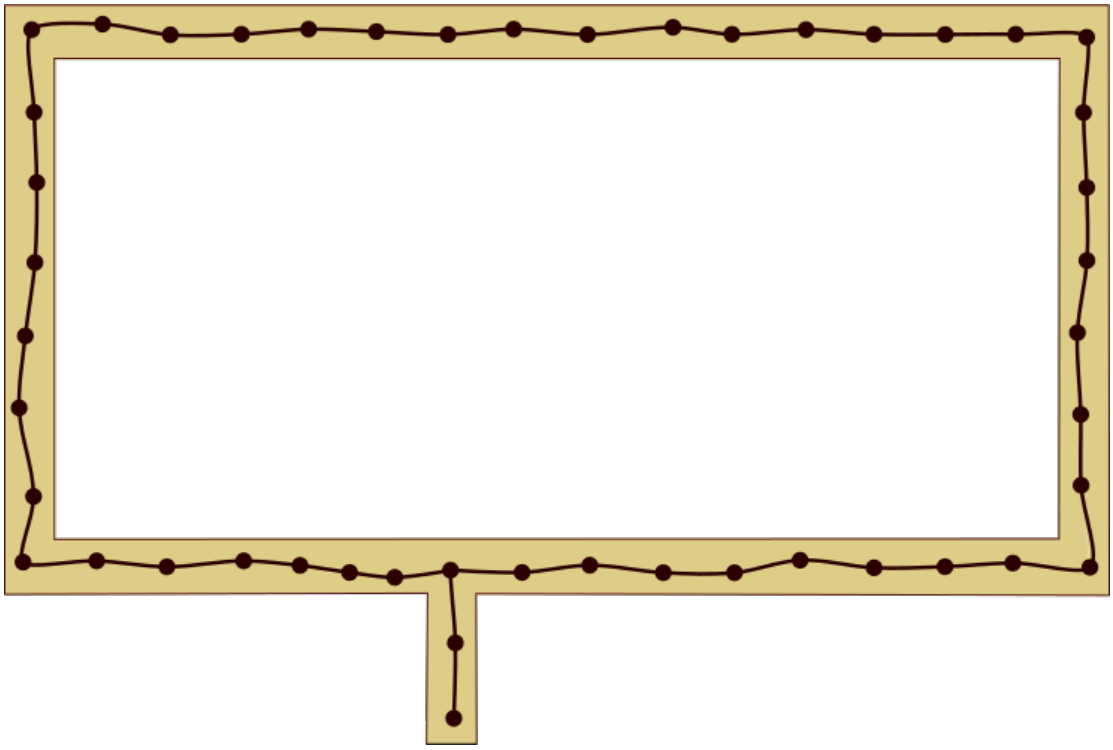


Figure 3.5: The trajectory used in simulations

In order to enforce a correction on the robot trajectory, some form of feedback is required. Given the fact that not many sensors are reliable for indoor applications and operate indoors and outdoors alike, a camera is used to obtain visual feedback of the robot's location. The visual feedback is passed through the place recognition algorithm discussed in chapter 4 and the map node to which it shows the closest resemblance is identified. The essential matrix constraint was identified as the most appropriate constraint that would tie up visual feedback between a map node and a robot pose since this is one of the more straightforward means of achieving this without solving for the structure, i.e., the map.

The relative translation being up to a scale does not affect at all since the Essential Matrix Constraint in GTSAM, uses the translation unit vector and not the translation vector by itself.

Although prior factors, between factors and initial values, are added at every node along the way, that is not the case for the Essential Matrix constraint. Since its purpose is not to define the connections among the nodes of the graph, but to tie the

robot poses to the map nodes, taking visual feedback at every couple of nodes seems more than sufficient(figure 3.9).

Overall, the factor graph functions as follows. The factor graph begins by adding all the pre-known map nodes , defining their poses as prior factors, and then adding the starting pose of the robot as a prior factor, for both of which the noise models are predefined. Next, it initializes the map nodes and the starting pose of the robot. Then, as long as the robot is moving and the sensor information is available, it keeps adding nodes at robot poses and initializing them at frequent intervals, while defining between factors among consecutive nodes. The decision to add a node is based on whether the robot has traveled a threshold distance or turned by a threshold angle, relative to the previous consecutive node. At predefined intervals, an image is captured through the robot's camera, which is then passed onto the place recognition system. The place recognition system determines the map node closest to the current robot pose, as described in chapter 4. The essential matrix between the captured image and the Street View image of the closest node is computed using SFM (Figure 3.6). By decomposing this essential matrix, the relative pose between the map node's camera and the robot's camera is determined. This pose is then transformed into a pose between the robot and the map node. Using this relative pose an Essential Matrix Constraint is established between the two. As soon as this step is completed, the factor graph is solved, and the corrected robot poses are determined.

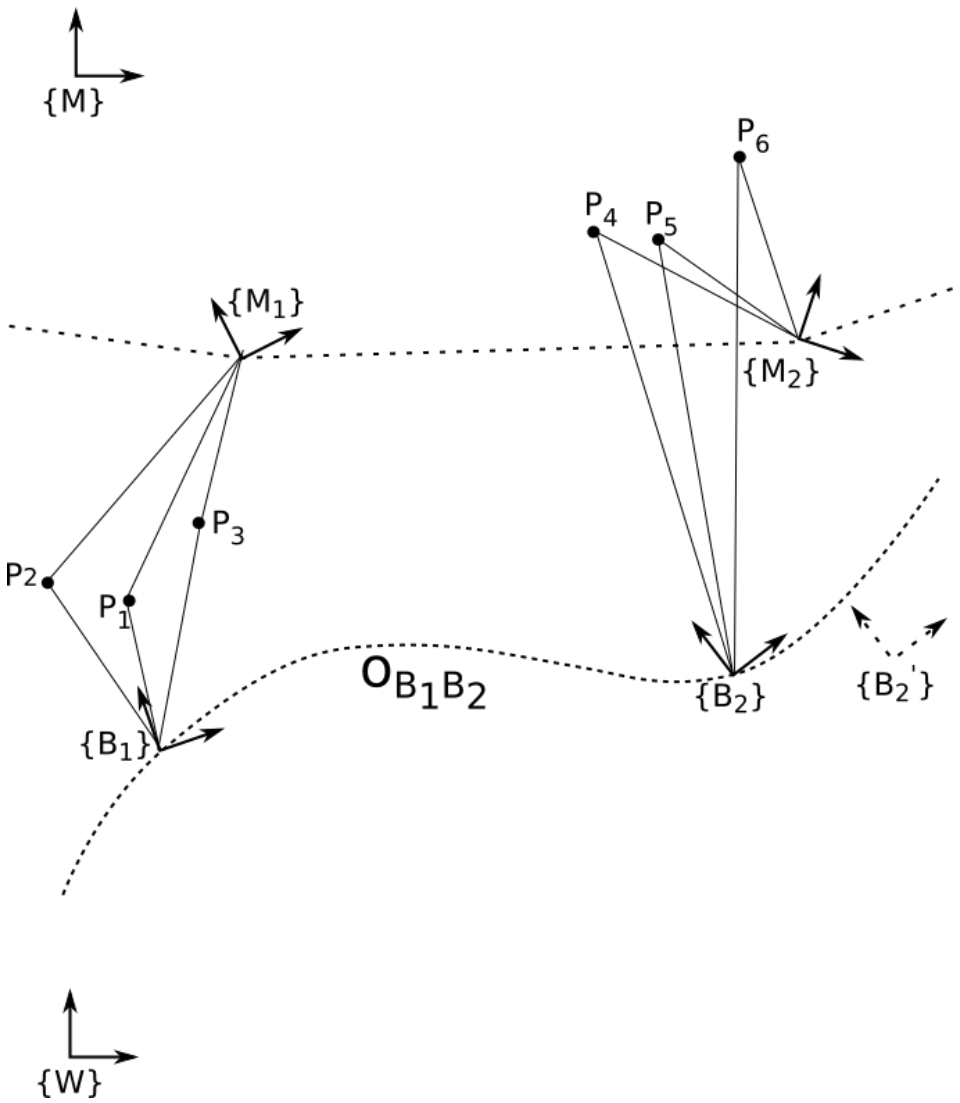


Figure 3.6: Establishing an essential matrix constraint between the robot and the map using SFM. B_1 and B_2 are two robot poses as determined using odometry. $O_{B_1B_2}$ is the odometry between B_1 and B_2. M_1 and M_2 are the map nodes closest to B_1 and B_2 respectively. B_2' is the ground truth of pose B_2. P_1-P_6 are the triangulated feature points between the respective image views.

3.4 The Algorithm

The procedure discussed under section 3.3 can be represented as an algorithm as presented in algorithm 3.1 below.

Algorithm 3.1 Factor Graph

Require: **Map** - the set of map poses
Odom - Odometry data
Cam - Camera image feed
NM - Noise models of **Map**, **Odom** and **Cam**

create $\boldsymbol{FG}$ ▷ *Create non-linear factor graph*
$\boldsymbol{FG} \xleftarrow{prior\ factor}$ **Map** nodes ▷ *Add map nodes as prior factors*
$\boldsymbol{FG} \xleftarrow{prior\ factor}$ Robot initial pose ▷ *Add robot's initial pose as a prior factor*
initialize $\boldsymbol{FG}$ ▷ *Define initial values for map nodes and the robot pose*
$N_{robot\ poses} \leftarrow 0$
while (**Odom** & **Cam** & **NM** are avaialable) **do**
    d $\leftarrow$ **Odom** ▷ *Calculate translation w.r.t previous pose*
    $\theta \leftarrow$ **Odom** ▷ *Calculate rotation w.r.t previous pose*
    if (d $\geq d_{thresh}$ **or** $\theta \geq \theta_{thresh}$) **then**
        $\boldsymbol{FG} \xleftarrow{Pose}$ Robot pose ▷ *Add the robot's current pose*
        $\boldsymbol{FG} \xleftarrow{between\ factor}$ (d,θ) ▷ *Add the relative pose between the current and previous poses as a between factor*
        initialize $\boldsymbol{FG}$ ▷ *Define an initial value for the new robot pose*
        $N_{robot\ poses} \leftarrow N_{robot\ poses} + 1$
    end if
    if ($N_{robot\ poses} = 5$) **then**
        image $\leftarrow$ **Cam** ▷ *Capture an image from the robot's camera*
        Place Recognition System $\leftarrow$ image
        Essential_Mat $\leftarrow$ ***Place Recognition System***
        ${}^{Cam1}\boldsymbol{Pose}_{Cam2} \Leftarrow$ Essential_Mat ▷ *Extract the relative pose between the map and robot cameras*
        ${}^{Robot}\boldsymbol{Pose}_{Map} \Leftarrow {}^{Cam1}\boldsymbol{Pose}_{Cam2}$ ▷ *Compute the relative pose between the robot and the closest map node*
        $\boldsymbol{FG} \xleftarrow{essential\ matrix\ constraint} {}^{Robot}\boldsymbol{Pose}_{Map}$ ▷ *Add the relative pose as an essential matrix constraint*
        calculated robot poses $\leftarrow$ solve $\boldsymbol{FG}$
        return calculated robot poses
        $N_{robot\ poses} \leftarrow 0$
    end if
end while

3.5 MATLAB simulator

In order to verify the concept described under section 3.3, a simulation study was carried out. For that, a simulator was created. MATLAB became the choice for this, for several reasons.

1. It is easier to work with in comparison to other options.
2. It is easier to debug the systems in MATLAB.
3. It has many options for visualization with a variety of built-in drawing and plotting tools.
4. A MATLAB wrapper for the GTSAM library was already available which has built-in functions for visualization

The map discussed under section 4.5.1 was generated in MATLAB as shown in figure 3.7. The robot trajectory once reproduced in MATLAB, appeared as in figure 3.8. The procedure discussed under section 3.3 was executed just as explained, with the only exception being the Essential Matrix Constraint. Once this was completed, the factor graph resembled that shown in figure 3.9.

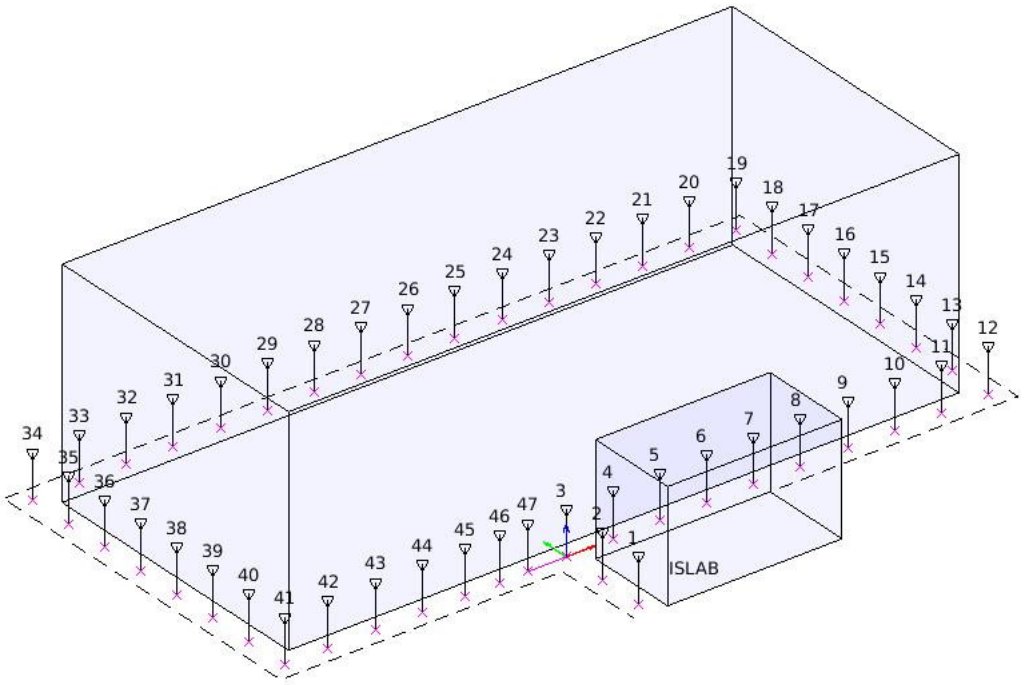


Figure 3.7: The map of *Memorial University of Newfoundland* (MUN) Engineering basement as seen in the MATLAB simulator

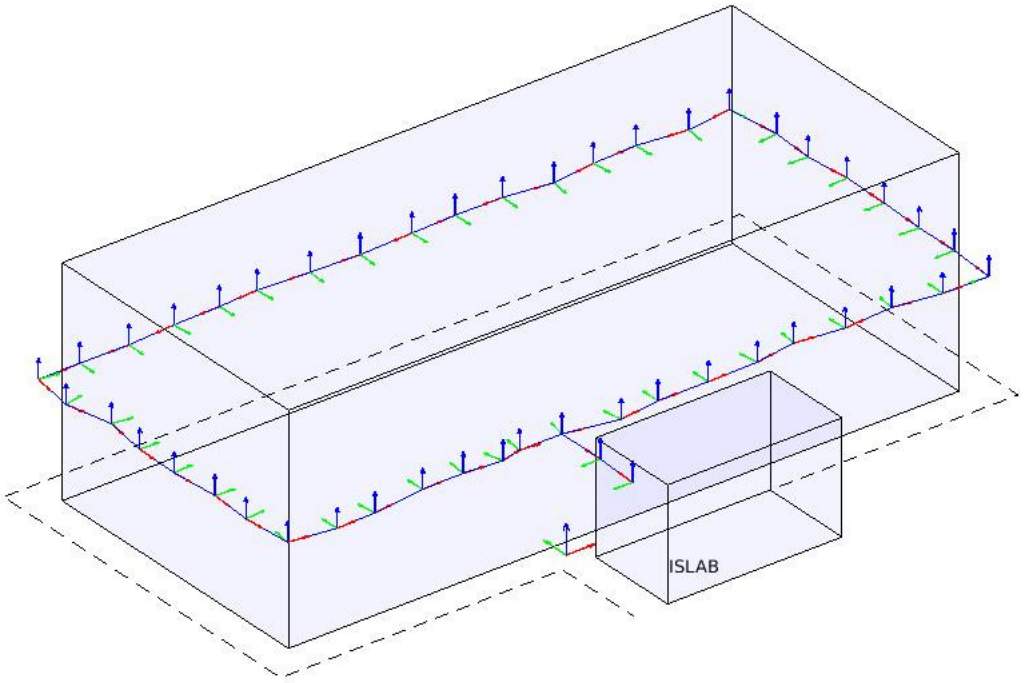


Figure 3.8: The hypothetical robot trajectory as seen in the MATLAB simulator

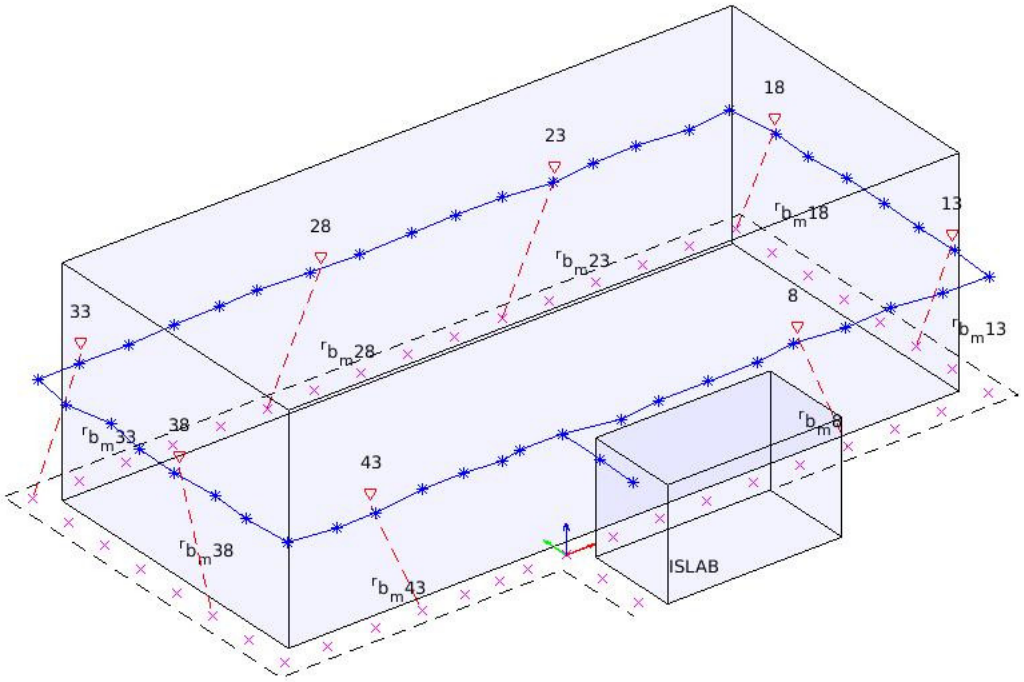


Figure 3.9: The complete factor graph as seen in the MATLAB simulator

The exception mentioned earlier was due to the absence of the Essential Matrix

Constraint in the MATLAB wrapper for GTSAM at the time. It was mitigated by utilizing the Between Factor. According to the GTSAM documentation, the Essential Matrix is parameterized similar to Pose3, with the translation being a unit vector. The relative pose between the map and robot nodes were placed between the two nodes, as a between factor. It is not entirely accurate but serves its purpose in proving the concept. Since place recognition was tested on its own as a separate system, this step was bypassed using manually generated Essential matrices. The simulator was initially developed as a 2D simulator, but once things were adequately figured out, it was upgraded into a 3D simulator.

3.6 ROS C++ simulator

GTSAM has a complete and well-established library written in C++, which comprises all the necessary data types and factors, including the Essential Matrix constraint. The limitations of the MATLAB version discussed under section 3.5 compelled testing the system in C++ as well. After figuring out the structure of the system using MATLAB, a similar simulator was developed using C++. A fundamental limitation of C++ is the lack of visualization tools. Neither did the C++ version of GTSAM offer any options for visualization, unlike its MATLAB counterpart. Thus, the attention was turned towards *Robot Operating System* (ROS) for the following reasons.

1. A catkin wrapper for GTSAM was available.
2. ROS has a visualization tool(rviz) with numerous features.
3. The simulator can later be converted into implementation with minimal modifications.

Initially, the simulation that was conducted in MATLAB was replicated line-for-line in ROS kinetic C++. Once the things were figured out such that the results produced were identical to those by the MATLAB simulator, the between factors simulating the Essential Matrix constraints were replaced by actual Essential Matrix constraints. Once everything was in order, the simulator has upgraded a level by making it modular. The critical steps in the factor graph process were separated into individual nodes to function parallelly. The nodes were connected through ros-topics. Important repetitive tasks, such as the addition of factors and solving the factor graph, were introduced as ros-services. When necessary, they are triggered by

sending service-requests to the servers. A central node was set in charge of handling the complete system. A schematic of the upgraded system is presented in figure 3.10.

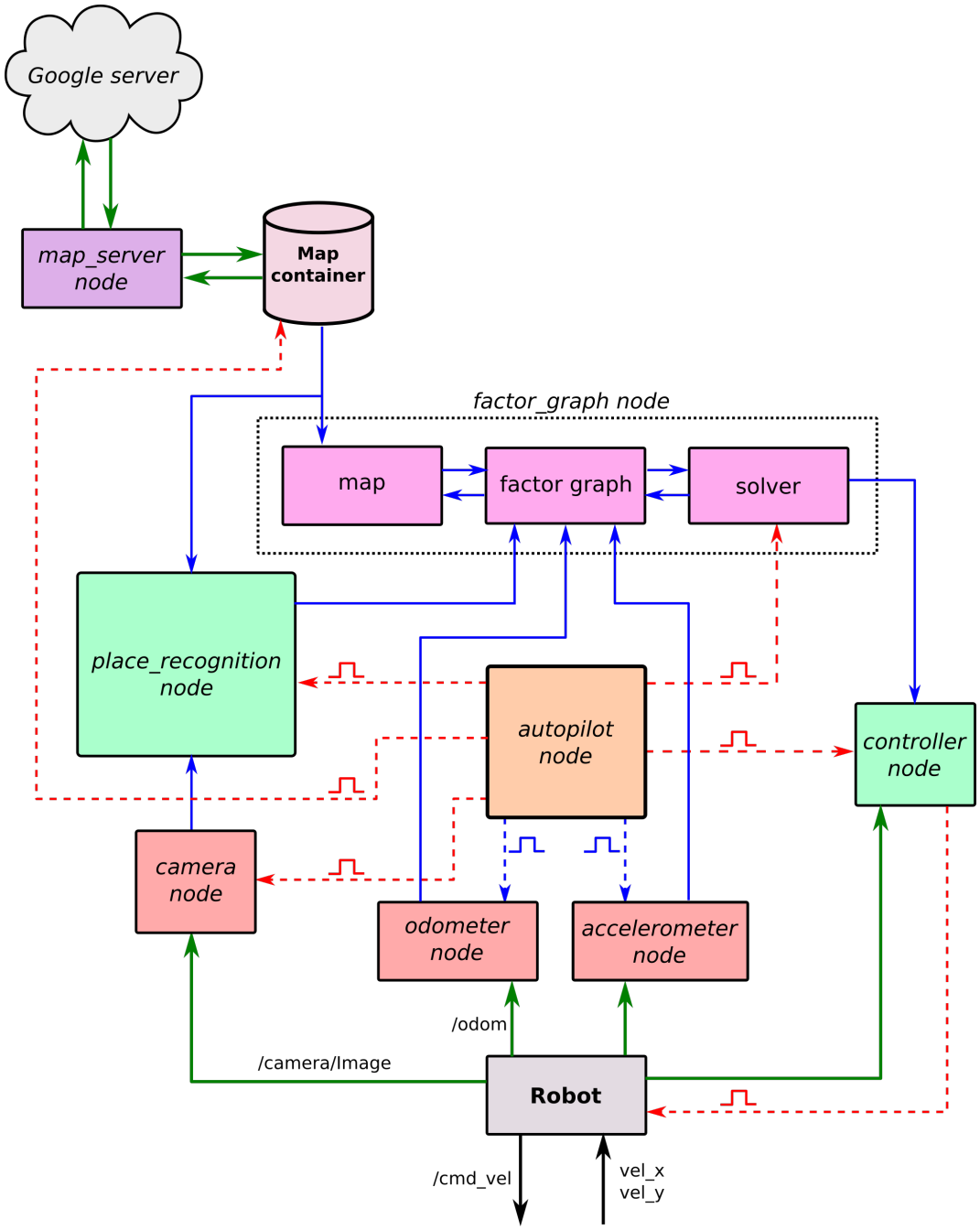


Figure 3.10: The schematic of the ROS C++ simulator

This upgrade proved useful for the following reasons.

1. The nodes can be reconfigured without impacting the rest of the system

2. New sensors and functionalities can be added to the system without sabotaging the existing system

3. The system can be easily extended onto a robotic platform, with minimal changes

4. Debugging the system becomes comparatively easier.

The rosgraph of the simulator is shown in figure 3.11

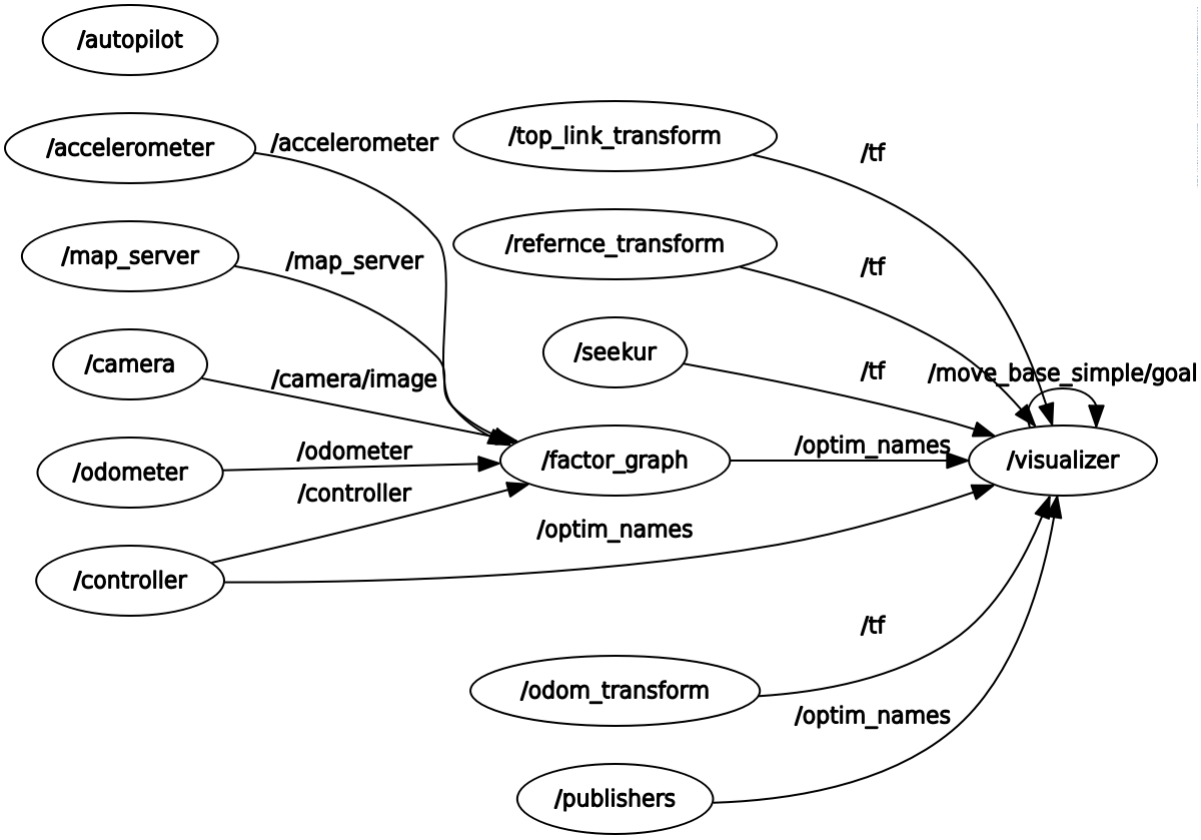


Figure 3.11: rosgraph of the simulator

The Visualizer

Although ROS has a visualization package(rviz) and the data types necessary for visualization, a custom visualizer that meets the requirements had to be created from scratch. Using the basic shapes from the *ros-visualization-msg* package, geometries from the *ros-geometry-msgs* package, geometric operations (such as transformation, rotation, angle conversions) from the *ros-tf* package and the sensor datatypes from

the *ros-sensor-msgs* package, a visualizer that is capable of displaying the map, robot trajectory, odometer, visual feedback, camera image feed and the solution to the factor graph, including the poses, connections and covariance ellipsoids were created. The ros-graph of the visualizer is shown in figure 3.12.

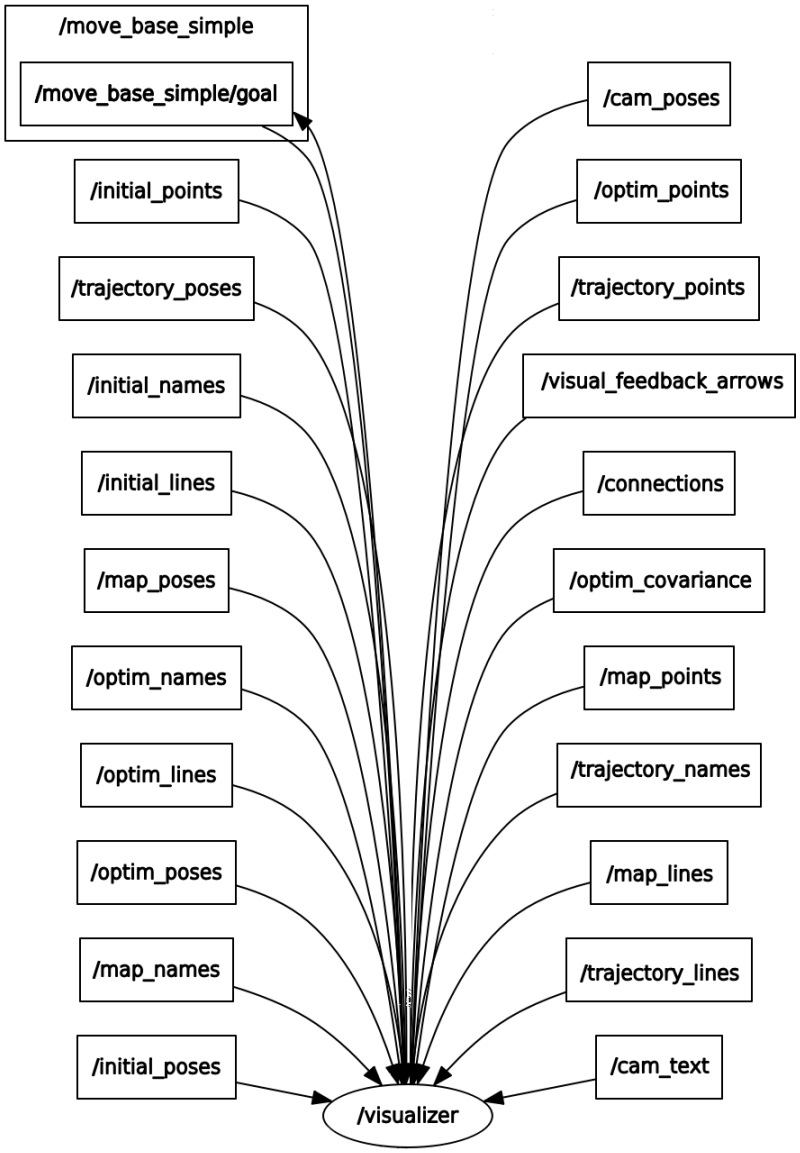


Figure 3.12: rosgraph of the visualizer

The Covariance Ellipsoids

Although the visualization tools in ROS provided a couple of options for visualizing covariance ellipses, it does not provide an option to visualize multiple ellipses. Hence, a custom covariance ellipsoid feature had to be developed. The MATLAB version of GTSAM included an option for plotting 3D trajectories together with the covariance ellipsoids. The MATLAB function for plotting the mentioned covariance ellipsoids was used as a reference [135]. *ros-visualization-msgs* and *ros-ecl-statistics* packages were used in the process. The ellipsoids are scaled to represent a 3σ(3 standard deviations) confidence boundary, which corresponds to 99% uncertainty. The scaling factor for 99.74% uncertainty was found to be 11.82 [136].

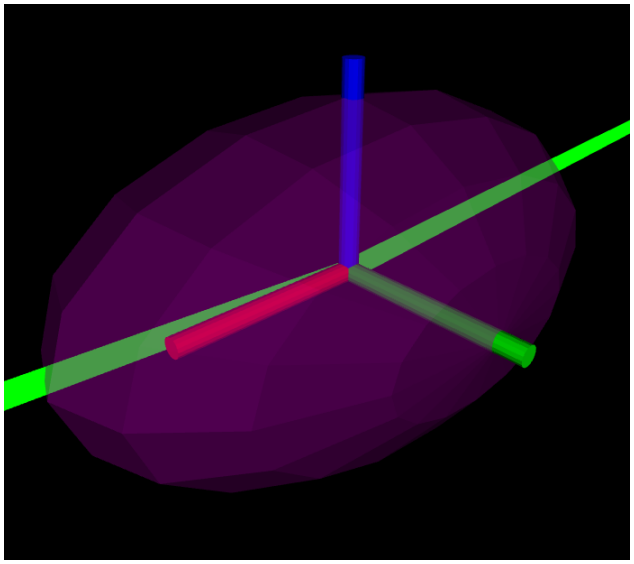


Figure 3.13: The covariance ellipsoid as seen in the ROS simulator

3.7 Results

3.7.1 MATLAB simulation

The result of the MATLAB simulation is presented in figure 3.14. It can be seen that, for the most part, the solution to the factor graph closely resembles the actual trajectory. Thus, it supports the proposed concept of factor graph and place recognition based indoor localization.

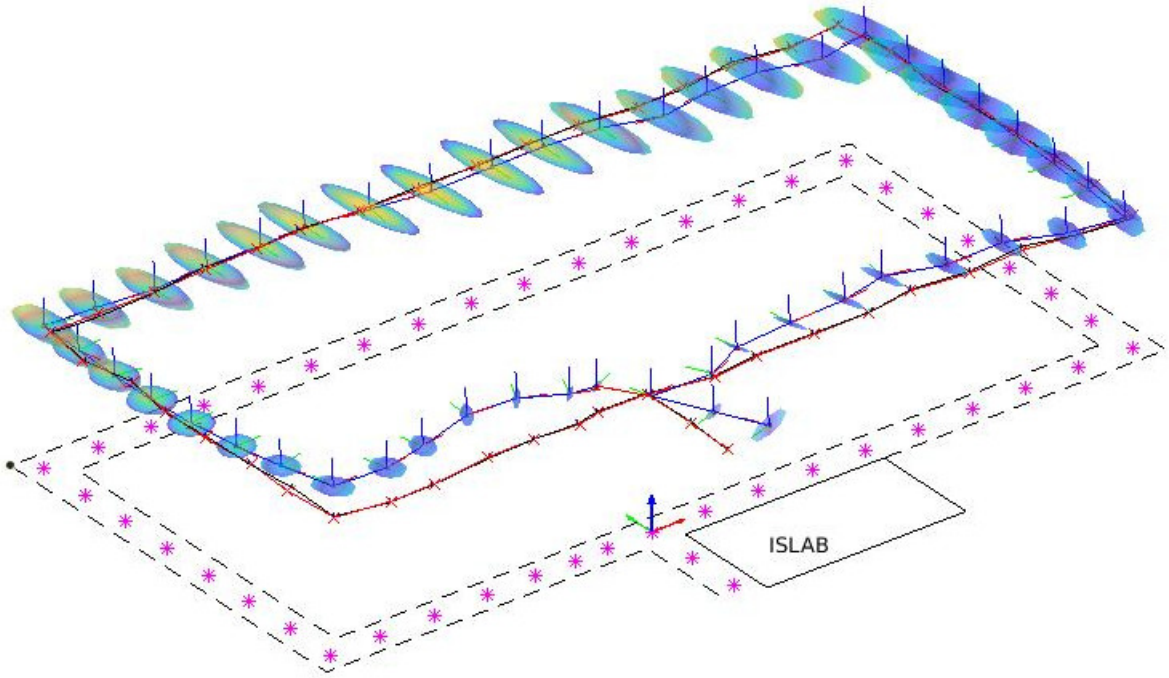


Figure 3.14: Solution of the factor graph

3.7.2 ROS C++ simulation

A graphical comparison of the ground truth and the odometry of the simulated trajectory is presented in figure 3.15. It represents the drift in odometry in comparison to the ground truth. Essential Matrix Constraints were added to the factor graph at every other robot pose along the robot's trajectory and are represented by the red color arrows shown in figure 3.16, connecting a robot pose to its relevant node in the map.

Figure 3.18 provides a visual comparison between the ground truth and the solution to the factor graph. The solution is in good agreement with the ground truth. The error ellipsoids corresponding to the solved poses represents a confidence region of 99%. It can be observed that the actual robot poses lies within the confidence region of the solved robot poses. It signifies that, with the proper essential matrix constraints, the factor graph estimates the actual robot pose with significant accuracy. The degree of correction on odometry drift by the factor graph is shown in figure 3.17.

Table 3.1 presents a quantitative picture of the localization process. The maximum

error in position is close to 2 m, a significant contributor to this could be the last robot pose ($45^{\texttt{th}}$ pose), which is constrained only through odometry. A similar explanation can be provided regarding the maximum error in orientation as well. The *Root Mean Square Error* (RMSE) for both position and orientation is less than 1 m and 2°, respectively. If necessary, these errors can be further reduced by utilizing better quality equipment in the case of practical implementation.

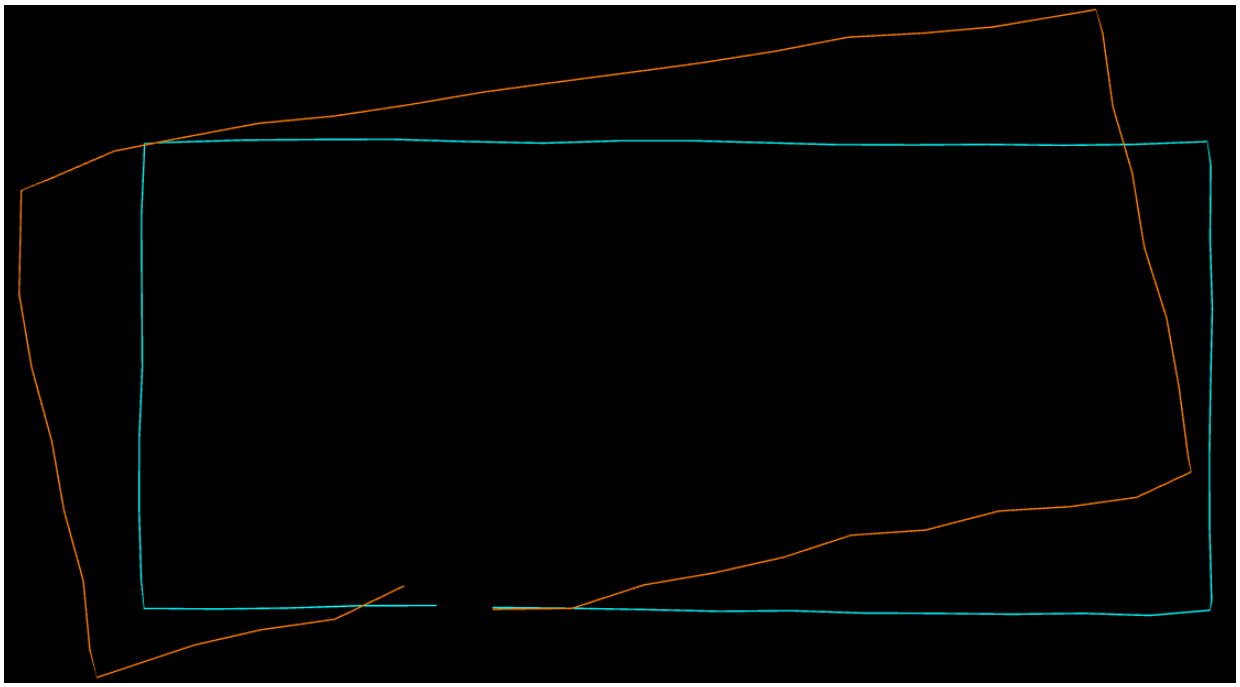


Figure 3.15: Odometry (orange) and ground truth (blue)

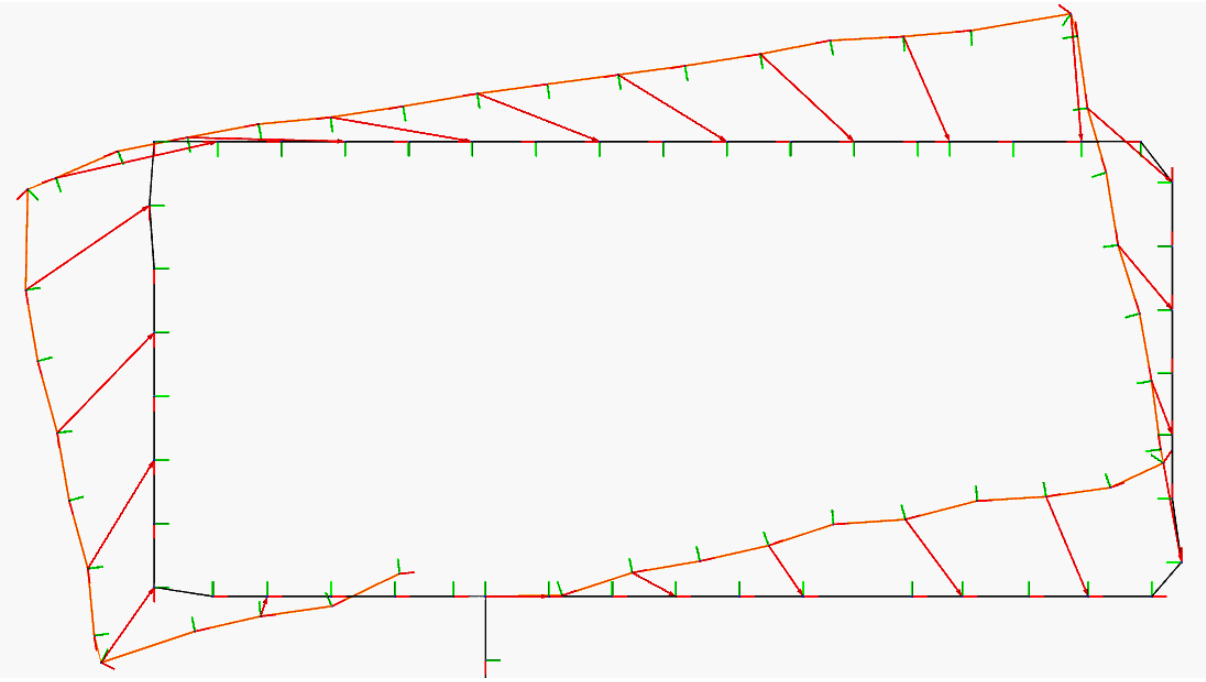


Figure 3.16: Robots poses (orange), map nodes (black) and Essential Matrix Constraints (red arrows)

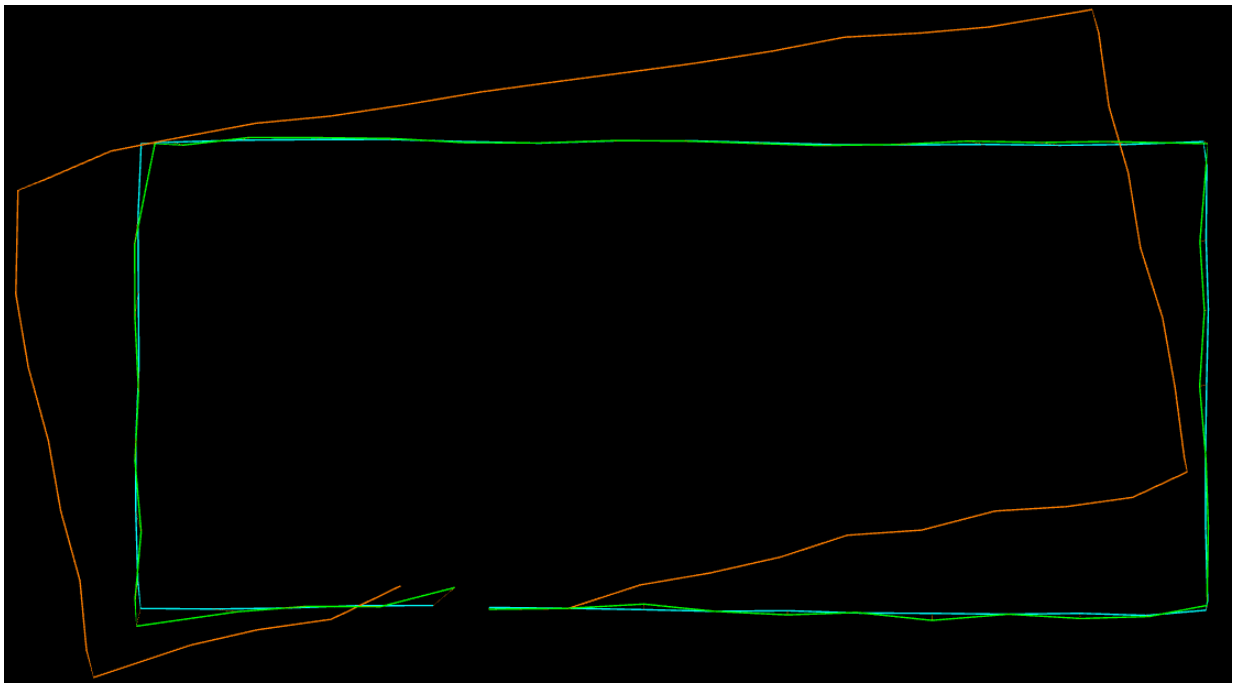


Figure 3.17: Odometry (orange), ground truth (blue) and the solution[1] (green)

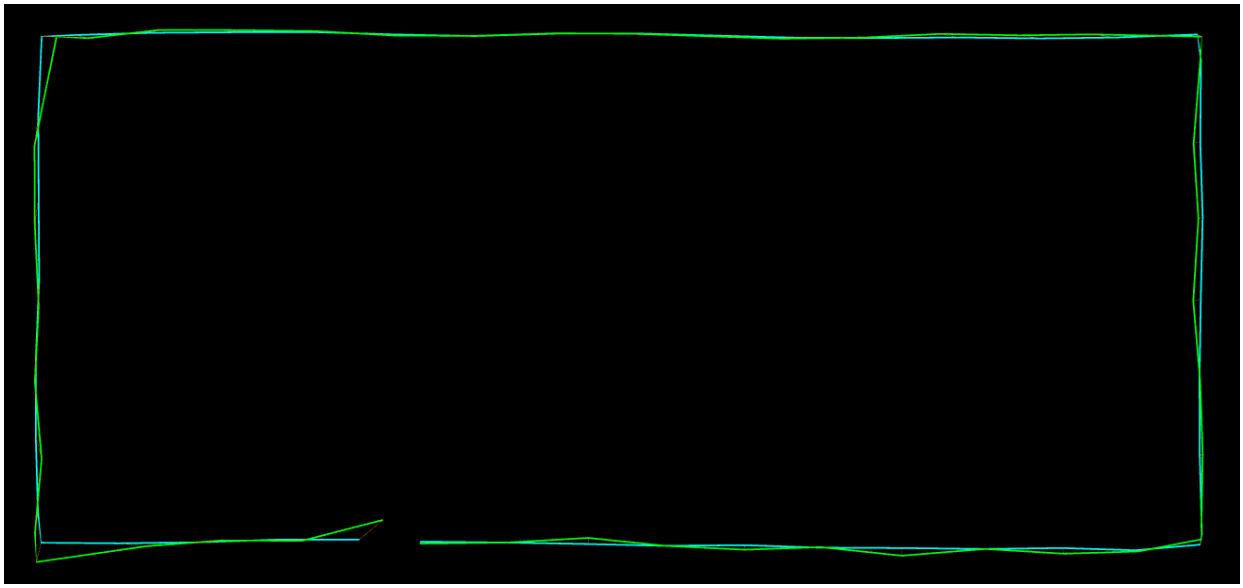


Figure 3.18: Ground truth (blue) and the solution (green)

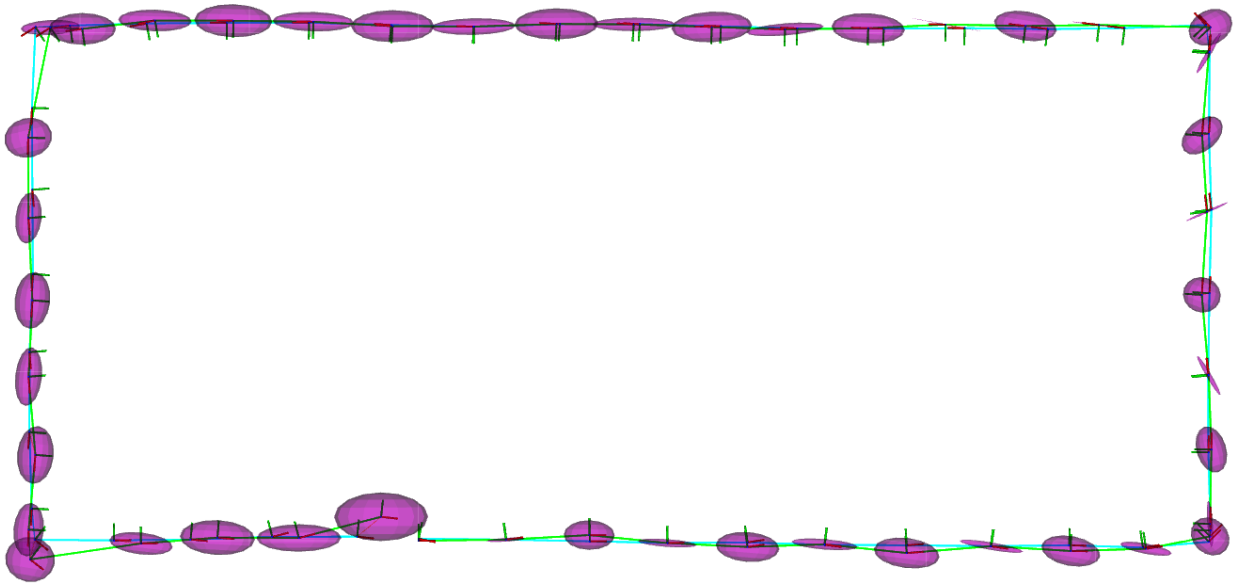


Figure 3.19: Error ellipsoids for a confidence of 99% (3 standard deviations)

[1]The solution is almost invisible since it overlaps with the ground truth

Table 3.1: Overall localization error

Error	Position (m)	Orientation (°)
Min	0.0195	0.001 24
Max	1.836	11.183
RMSE	0.906	1.683

3.8 Conclusion

The MATLAB simulation result shown in figure 3.14 suggested that using factor graphs and place recognition for indoor localization is possible. However, since this simulation's objective was to understand the behavior of the factor graph, a separate place recognition system was assumed. It can be seen that the uncertainty of the solved poses gradually increases halfway along the robot's trajectory and gradually decreases as the robot completes the loop. It can be seen that the uncertainty along the Y-axis has propagated faster than that along X-axis and Z-axis. It is due to the yaw of the robot being the only rotation, whereas the roll and pitch along the trajectory are assumed to be zero. It can also be seen that the solution to the factor graph shows a deviation from the actual trajectory[2] towards the beginning and the end. There are a few possible causes for this. The factor graph is pivoted near map node 3, although the trajectory begins at map node 1. Hence, the solution is fixed at node 3. Visual feedback constraints are added at every fifth node. The result could have been better if more constraints were added. The starting point of the trajectory is not constrained despite it been a prior. Finally, a loop closure constraint has been added between nodes 47 and 3. However, according to the odometry of the robot, the loop is not closed. In satisfying all of the above constraints during optimization, the solution might have deviated from the actual.

However, for the ROS C++ simulation, the factor graph is pivoted at node 1. Visual feedback constraints between robot poses and map nodes are added at every other

[2]The actual trajectory begins at map node 1, goes through node 3, and completes an anti-clockwise loop returning to node 3.

node, and no loop closure constraint is added. Besides, the trajectory used for the simulation is much more accurate than that used for the MATLAB simulation. It is because the former was derived via a simulation whereas the latter was derived hypothetically.
Hence, the results for the ROS C++ simulation is much more consistent and closely resembles the robot's trajectory.

The uncertainty along the trajectory in the ROS C++ simulation has propagated along the X-axis compared to the MATLAB simulation. The reason for this might be the uncertainty of the essential matrix constraint added. Since the translation component of the essential matrix is correct only up to a scale, the uncertainty along the direction of translation (in this case the X-axis) is comparatively higher.

Throughout the first half of the trajectory, the uncertainty ellipsoids at the nodes where the visual feedback constraints are added, are very small and slim. However, throughout the rest of it, a stark difference in size and shape can be seen. Although the result has not been significantly affected by this, it is worthwhile looking into.

The uncertainty of the last pose is the largest as the only constraint on it is the between factor with its previous pose.

Chapter 4

Place Recognition System

An essential step in the provision of visual feedback is establishing a clear sense of the current position. In robotics literature, this process is termed as *Place Recognition*. This chapter provides an overview of Place Recognition, followed by the preliminaries of place recognition. Next, the proposed place recognition system is presented in detail. Finally, the results of the conducted tests are presented, followed by the conclusions that were derived upon those.

4.1 Introduction

Place recognition also known as *Place Localization* has been an active research area for years due to its potential applications in numerous domains of engineering.
There are two main methods for image retrieval process: *Brute-force* method and the Visual BoW [137] method. In the brute-force method, each query image is compared with each reference image available in the database for visual similarity. The accuracy of this method is very high and results in the best accuracy that can be achieved by any image retrieval method. The downside of this method is the high computational cost and the time associated with the retrieval process. Also, both the computational cost and the retrieval time increases linearly with the number of reference images.

Visual BoW is a popular image representation method where feature descriptors are used to denote salient features that occur in an image. This method has gained popularity due to its simplicity and the improvement in the speed of the comparison process. The general practice is to quantize these descriptors into clusters and assign a

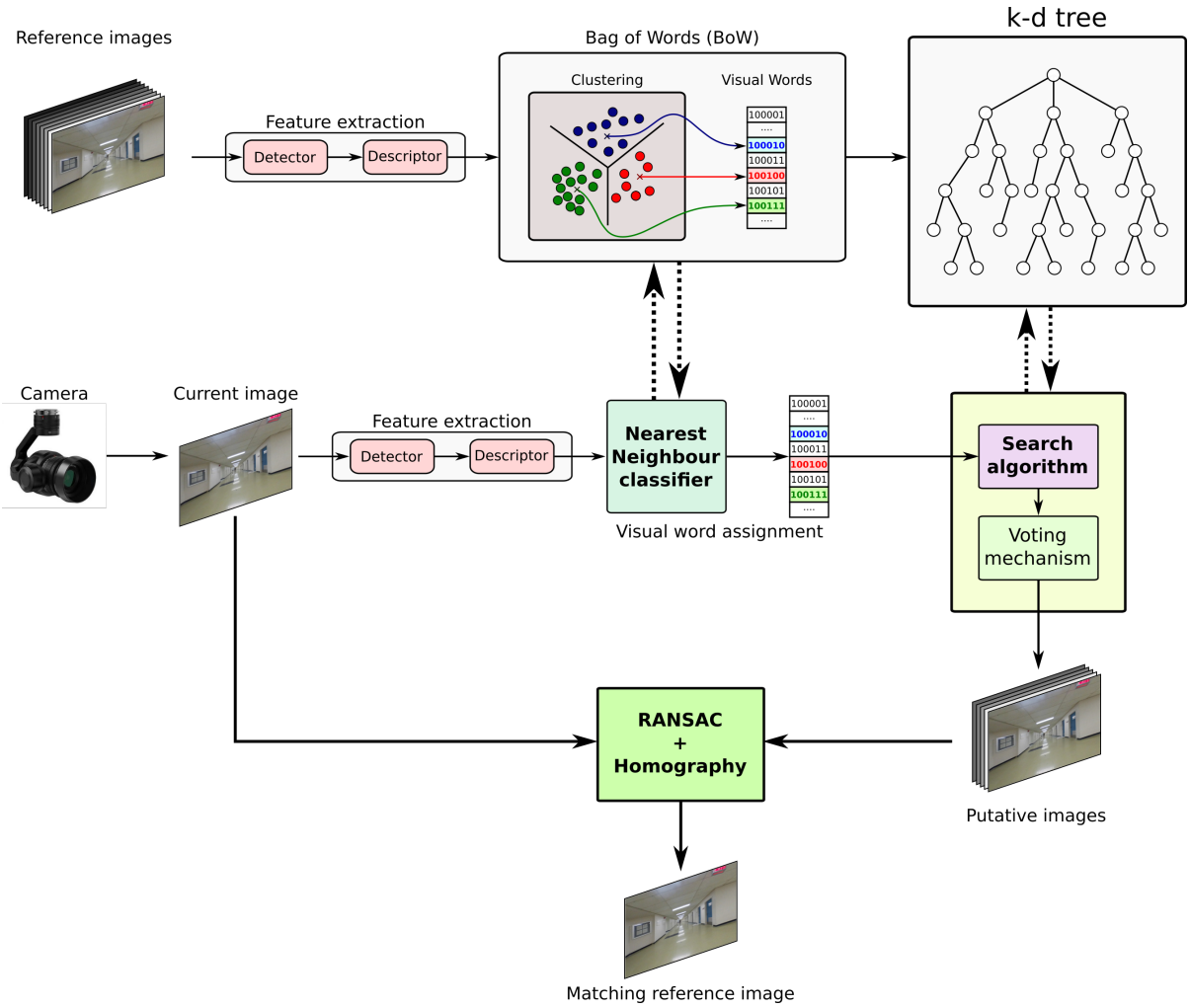


Figure 4.1: Schematic of a typical place recognition system

label to each cluster known as the Visual Word. The advantage of doing this is that it improves the speed of comparison at a slight loss of accuracy. However, the drawback is that the quantization errors (due to quantization artifacts during clustering), creates a potential for error in the retrieval process. These errors may even cause perceptual aliasing. Using smaller bin sizes and using raw features as visual words are two ways of minimizing this issue.

Two kinds of images are involved in the retrieval process, namely, reference images and query images (query images used that are used for testing are referred to as test images). Reference images refer to a set of pre-existing images that represents the environment of interest. These images can be perspective, omnidirectional, or

equirectangular. It is against the reference images that a query image is compared to estimate the location that it represents in a map. A query image is an image of an environment, captured using the active camera (the camera that is handheld or fixed onto a robot, which is used to capture images of the environment it exists), of which the location while capturing the image needs to be estimated for a map of the same environment.

The retrieval of images that match a given query image can be done either through a linear search or tree data structure. Linear search algorithms (e.g., search using the inverted table [40]) shows better accuracy at the cost of computation time. On the other hand, using tree data structures result in fast algorithms at a slight cost of accuracy. K-d tree is the most used tree structure for image retrieval due to its simplicity. Both online and offline k-d trees can be seen in applications. Online k-d trees build the tree simultaneously with the localization process. Offline k-d trees build the tree before the localization process begins.

The objective here is to predict the node of the pre-existing map , which is closest to a given query image. The approach taken here is "brute force," i.e., each query image is compared with every reference image. The set of reference images comprises 30 equirectangular street view images that correspond to the 30 nodes of the considered map. The comparison is performed through the extraction and matching of image features by using the SIFT detector. The preferred feature detector-descriptor combination is SIFT detector with BRIEF descriptor. SIFT features are preferred due to its invariance to translation, scale, and rotation. BRIEF descriptors are preferred over other options as they are faster and performs better in matching, due to its series of optimizations.

4.2 Preliminaries

4.2.1 Performance Indicators

In literature, the performance of classification algorithms is evaluated using several indicators. The performance of the place recognition system was evaluated using precision, recall, and F1-score. These indicators are calculated from specific values

related to a data point of interest extracted from the confusion matrix of the classification.

Confusion Matrix

The confusion matrix of the place recognition test takes the following form.

Table 4.1: The confusion matrix of the place recognition test

	Node 1	**Node 2**	...	**Node n**
Node 1	$c_{1,1}$	$c_{1,2}$	...	$c_{1,n}$
Node 2	$c_{2,1}$	$c_{2,2}$	...	$c_{2,n}$
⋮	⋮	⋮	⋱	⋮
Node n	$c_{n,1}$	$c_{n,2}$	...	$c_{n,n}$

As shown in table 4.2, from the Confusion Matrix shown under table 4.1, four parameters can be defined with regards to a node of interest, node i.

Table 4.2: The confusion matrix of Node i

	Node i	**Not Node i**
Node i	**TP**	**FN**
Not Node i	**FP**	**TN**

True Positives(TP) Total number of test images from node i, that has been classified under the node i.

False Positives(FP) Total number of test images from all other nodes, that has been classified under the node i.

True Negatives(TN) Total number of test images not from from node i, that has not been classified under node i.

False Negatives(FN) Total number of test images from node i, that has not been classified under the node i.

Accuracy

Accuracy indicates the number of true classifications as a fraction of the total number of classifications. The total number of true classifications contains consists of both true positive predictions and false-negative classifications. Hence, accuracy alone does not represent a fair indication of performance.

$$Accuracy \ = \ \frac{TP + FN}{TP + FN + FP + FN} \tag{4.1}$$

Precision

Precision indicates the number of actually positive classifications (True Positive) as a fraction of the total number of positive classifications. The total number of positive classifications consists of both True Positive and False Negative classifications.

$$Precision \ = \ \frac{TP}{TP + FP} \tag{4.2}$$

Recall

Recall indicates the number of actually positive labels as a fraction of the total number of positive labels.

$$Recall \ = \ \frac{TP}{TP + FN} \tag{4.3}$$

F1 Score

F1 Score is the Harmonic mean between precision and recall values. The following formula defines the F1 Score.

$$F1 \ Score \ = \ \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{4.4}$$

4.3 Methodology

4.3.1 Brute force place recognition algorithm

First, a maximum of 100 feature points of the query image is extracted and are matched against the feature points of every reference image. It is followed by Lowe's ratio test[1] on every feature match. By the end of this step, only the strong feature matches are retained, and the rest is filtered out. Next, the five reference images that exhibit the highest number of strong feature matches are chosen as putative matches. As the number of images chosen as putative increases, the possibility of the correct match being a putative too increases. However, with a higher number of putative, the computational cost increases unnecessarily. Hence, the number of putative images should be chosen such that an acceptable trade-off between the two is established. Thus, the number of putative images was empirically determined as five.

Table 4.3: Thresholds used for RANSAC

Parameter	Value
Confidence threshold(%)	99.9
Max. re-projection error(px)	1
Maximum iterations	2000

Next, a second stage verification is performed on the putative matches, where a RANSAC based homography check is conducted on each putative match, and the number of inliers is determined. The parameters and values used for RANSAC are presented in table 4.3. The geometric similarity between the two images is assessed using the *5-point algorithm* [138], which is faster and more accurate than the widely used 8-point algorithm [139]. It requires a minimum of five points for a successful calculation. But, the more points used, the better the result gets. However, setting a significantly high value as the minimum threshold for the number of inliers

[1]The best match and the second-best match of each query feature (feature points of the query image) are compared with one another based on distance. If the best match is 0.7 (as empirically determined by David.G.Lowe in [127]) times closer than the second-best match, then the best match is considered to be a strong feature match. Otherwise, it is considered a weak feature match.

is impractical since it may end up rejecting specific images that display sufficient geometric resemblance. Thus, ten was empirically determined as a suitable minimum threshold, i.e., a putative match qualifies as a valid match only if it comprises a minimum of 10 inliers. For those putative matches that qualify, a similarity score is calculated using equation 4.5.

$$Similarity\ score\ =\ \frac{2 \times no.\ of\ inliers}{no.\ of\ images\ +\ no.\ of\ matches} \tag{4.5}$$

Then, the putative matches that score less than a pre-determined threshold are filtered out. Out of the remaining, the node corresponding to the putative match, which records the highest similarity score is predicted as the node closest to the considered query image.

The place recognition system was tested for outdoor and indoor environments using snippets[2] from outdoor and indoor street view images, respectively. Snippets of varying sizes were selected so as not to capture similar regions of the images. e.g., the sky, which is visually consistent within an image as well as between images.

As the outdoor test dataset, a total of 300 snippets comprised of 10 snippets from each outdoor street view image were used. The snippets are well illuminated, have no self-similar structures, and have a significant amount of unique features. Hence, this dataset helped assess the performance of the place recognition system under more favorable conditions.

As the indoor test dataset, a total of 1470 snippets comprising 30 snippets from each indoor street view image were used. These snippets contain illumination variations, repetitive and self-similar structures, and lack unique features. Hence, this dataset helped in assessing how the place recognition system behaved under less favorable conditions common in indoor environments.

Finally, two sets of perspective images captured from an iPhone SE were used to test the algorithm. A confusion matrix similar to 4.1 was created from which the performance indicators presented under section 4.2.1 were computed.

[2]Rectangular sub-images cropped from the original image

4.4 The Algorithm

The procedure discussed under section 4.3.1 can be represented as an algorithm is presented in algorithm 4.1

Algorithm 4.1 Place Recognition

Require: **Map** $= \{m_1, m_2, \ldots, m_n\}$ ▷ *Reference images*

Test $= \{q_1, q_2, \ldots, q_i, \ldots, q_n\}$ ▷ *Query (test) images*

1: **for** $q_i \in$ **Test** **do**

2: $d_q \leftarrow$ SIFT + BRIEF of q_i ▷ *Extracting descriptors*

3: **for all** $m_i \in$ **Map** **do**

4: $d_m \leftarrow$ SIFT + BRIEF of m_i ▷ *Extracting descriptors*

5: $N_{match}\{m_i\} \leftarrow$ Ratio test $\leftarrow$ Match d_q and d_m ▷ *Matching features and determining the no. of strong feature matches*

6: **end for**

7: $M_{putatives} \leftarrow \{m_i \in \textbf{Map} \mid \underset{m_i}{argmax}\ N_{match}\{m_i\}\}$ ▷ *Selecting m_i with the highest number of matching features*

8: **for all** $m_i \in M_{putatives}$ **do**

9: inliers $\leftarrow$ RANSAC + Homography ▷ *Outlier rejection using RANSAC*

10: **if** inliers ≥ 10 **then**

11: Compute $P_{match}(m_i)$ ▷ *Using equation 4.5*

12: **if** $P_{match}(m_i) \geq \theta_{ransac}$ **then**

13: $M_{best} \leftarrow m_i$ ▷ *Selecting m_i's that are potential best matches for q_i*

14: **end if**

15: **end if**

16: **end for**

17: **return** $\{m_i \in M_{best} \mid \underset{m_i}{argmax}\ P_{match}(m_i)\}$ ▷ *Selecting m_i with the highest P_{match} as the best match for q_i*

18: **end for**

4.5 Results

4.5.1 Map of MUN Engineering building basement

A Google Indoor Street View map was created using fourty seven 360° images (figure 4.2) that were captured by a Samsung Gear 360 camera, along the MUN Engineering basement corridor, at pre-selected locations which are approximately 4.5 m (15 feet) apart from one another. The locations later become the nodes for this topological map. These images were uploaded to the Google server thus creating a map that would later serve as the pre-constructed map that is used for Localization. As the nodes were selected premeditatively, approximate positions of the nodes relative to the map's origin are known. For simplicity, the absolute position of the origin was selected to be $(0, 0, 0)$. The origin was also chosen as the first node (Node 1) of the map.

Figure 4.2: 360° image corresponding to node1 captured by Samsung Gear 360

A node map of the *Memorial University of Newfoundland* (MUN) Engineering basement similar to that shown in figure 4.3 created. All the map nodes were declared as prior factors with Node 1 as the origin.

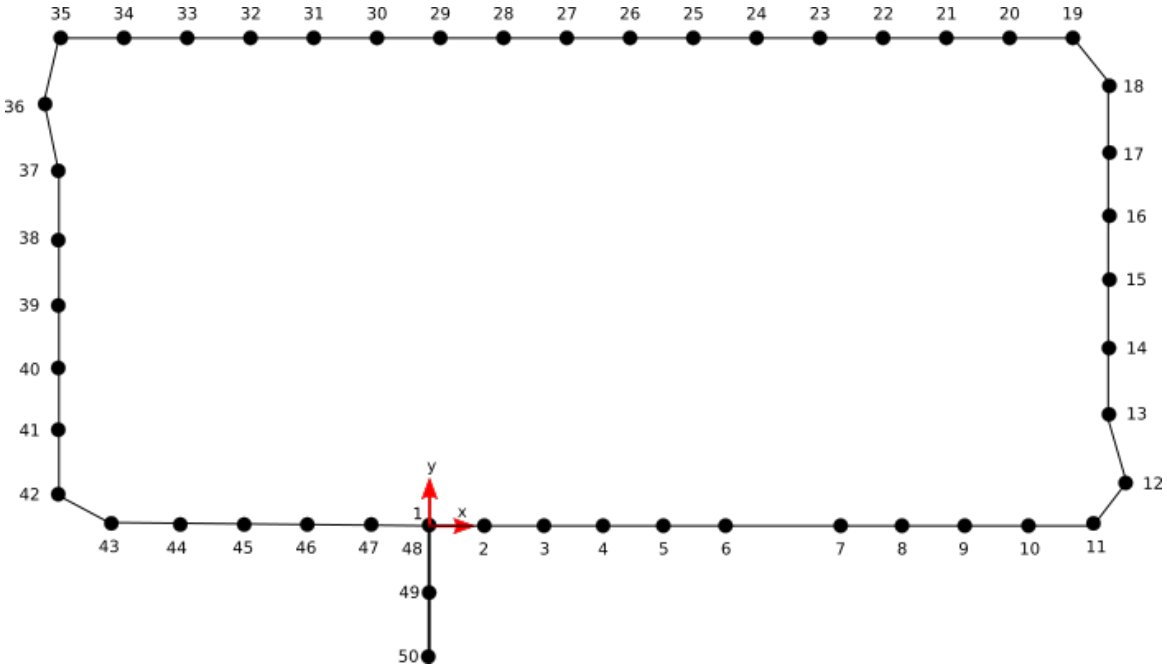


Figure 4.3: The node map of the MUN Engineering building basement.

4.5.2 Place recognition

By looking at the precision and recall values for the outdoor test case, in table 4.4, it can be seen that when using snippets, the algorithm has performed well in predicting the closest node. The high accuracy of 96.74% substantiates this. The precision values suggest that almost all the predicted nodes were correct. However, a precision value for row number 6 cannot be calculated.

Table 4.4: Performance of the place recognition algorithm for outdoor images

Node	Precision	Recall	F1 Score	Node	Precision	Recall	F1 Score
1	1·000	0·1	0·182	16	1·000	0·2	0·333
2	1·000	0·1	0·182	17	1·000	0·2	0·333
3	1·000	0·3	0·462	18	1·000	0·2	0·333
4	1·000	0·5	0·667	19	1·000	0·1	0·182
5	1·000	0·7	0·824	20	1·000	0·5	0·667
6	–	0·0	–	21	1·000	0·6	0·750
7	0·750	0·3	0·429	22	1·000	0·1	0·182
8	1·000	0·5	0·667	23	1·000	0·8	0·889
9	1·000	0·2	0·333	24	0·800	0·4	0·533
10	1·000	0·3	0·429	25	0·750	0·3	0·429
11	1·000	0·4	0·571	26	1·000	0·3	0·462
12	1·000	0·2	0·333	27	1·000	0·1	0·182
13	1·000	0·2	0·333	28	1·000	0·1	0·182
14	1·000	0·4	0·571	29	1·000	0·3	0·462
15	1·000	0·2	0·333	30	1·000	0·3	0·462

In figure 4.4, it can be seen that for true label 6, there are no predicted labels. Therefore, both the true and false positive values become zero causing the value of precision to become indeterminate.

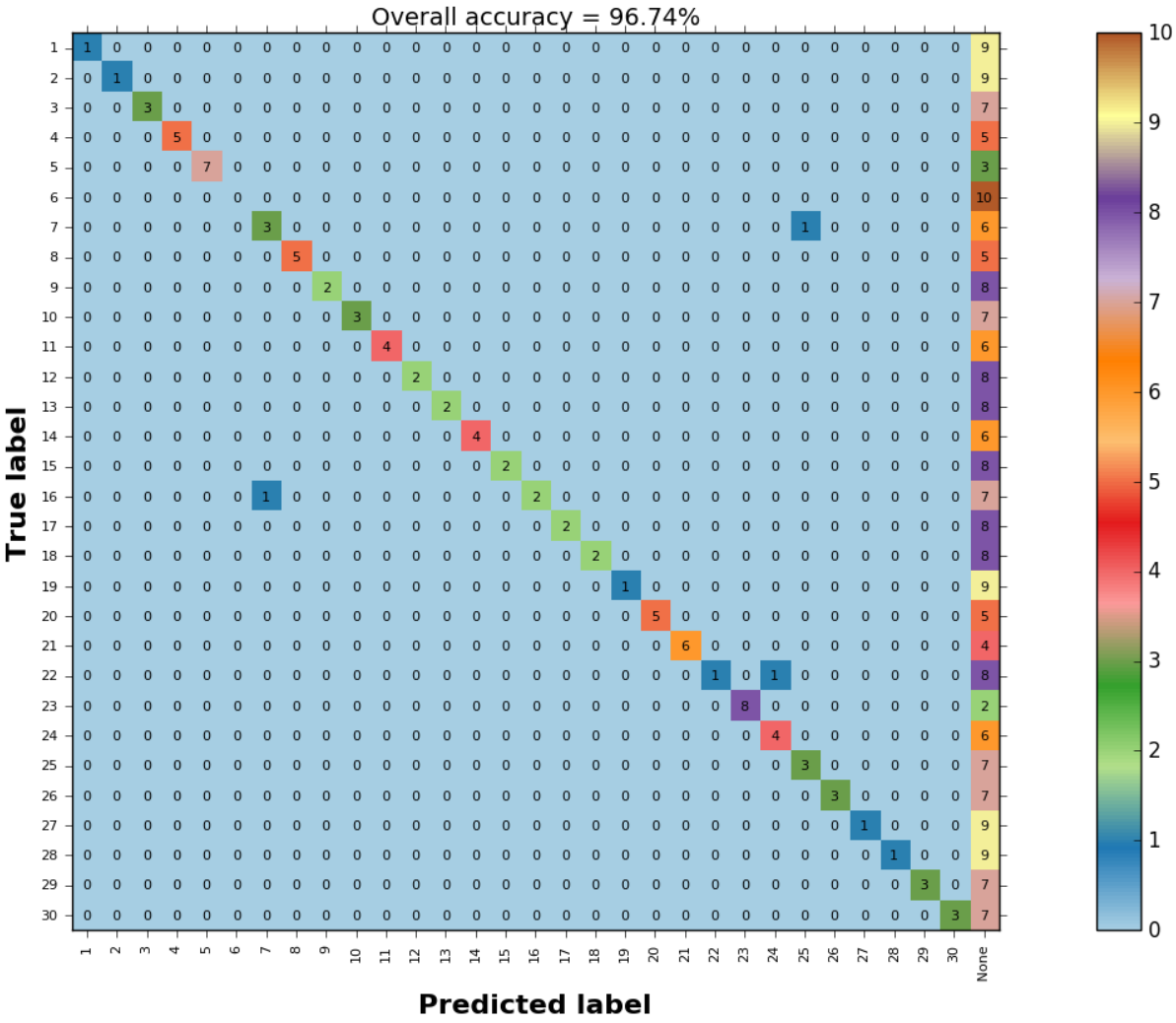


Figure 4.4: Confusion Matrix for outdoor snippets. The values along the leading diagonal indicate the number of true positives. The last column indicates the number of images for each case, which is inconclusive. However, in this case, inconclusive is not considered as an incorrect match. Hence, inconclusive cases are excluded in computing the overall accuracy, i.e., the total number of true positive matches as a fraction of the total number of matches.

However, for the indoor test case, the accuracy has drastically dropped to 34.75%. From table 4.5, it is obvious that for most of the nodes, the precision, recall, and F1-score values are significantly lower in comparison to the outdoor case.

Table 4.5: Performance of the place recognition algorithm for indoor images

Node	Precision	Recall	F1 Score
1	0·500	0·100	0·167
2	0·800	0·133	0·229
3	0·105	0·067	0·082
4	1·000	0·133	0·235
5	–	0·000	–
6	1·000	0·067	0·125
7	–	0·000	–
8	0·222	0·067	0·103
9	0·667	0·067	0·121
10	–	0·000	–
11	0·000	0·000	–
12	0·091	0·033	0·049
13	–	0·000	–
14	–	0·000	–
15	0·600	0·100	0·171
16	0·250	0·067	0·105
17	0·500	0·233	0·318
18	0·909	0·333	0·488
19	0·538	0·233	0·326
20	0·571	0·133	0·216
21	0·857	0·200	0·324
22	1·000	0·067	0·125
23	0·357	0·167	0·227
24	–	0·000	–
25	0·429	0·100	0·162
26	0·333	0·033	0·061
27	0·667	0·067	0·121
28	1·000	0·033	0·065
29	1·000	0·033	0·065
30	1·000	0·033	0·065
31	–	0·000	–
32	1·000	0·133	0·235
33	0·200	0·033	0·057
34	0·132	0·167	0·147
35	1·000	0·100	0·182
36	0·429	0·100	0·162
37	0·667	0·200	0·308
38	1·000	0·233	0·378
39	0·000	0·000	–
40	1·000	0·067	0·125
41	–	0·000	–
42	1·000	0·033	0·065
43	0·047	0·200	0·076
44	0·550	0·367	0·440
45	1·000	0·033	0·065
46	0·000	0·000	–
47	1·000	0·367	0·537
48	1·000	0·067	0·125
49	0·500	0·033	0·062

From the confusion matrix presented in figure 4.5, it can be seen that apart from predicted label 43 for a majority of the test cases, the algorithm has either produced the correct prediction or rejected the incorrect ones. The visible pattern along the leading diagonal and the last column of the confusion matrix substantiates this. Although a few mismatches can be noticed scattered throughout the confusion matrix, a stark trend can be seen for predicted label 43.

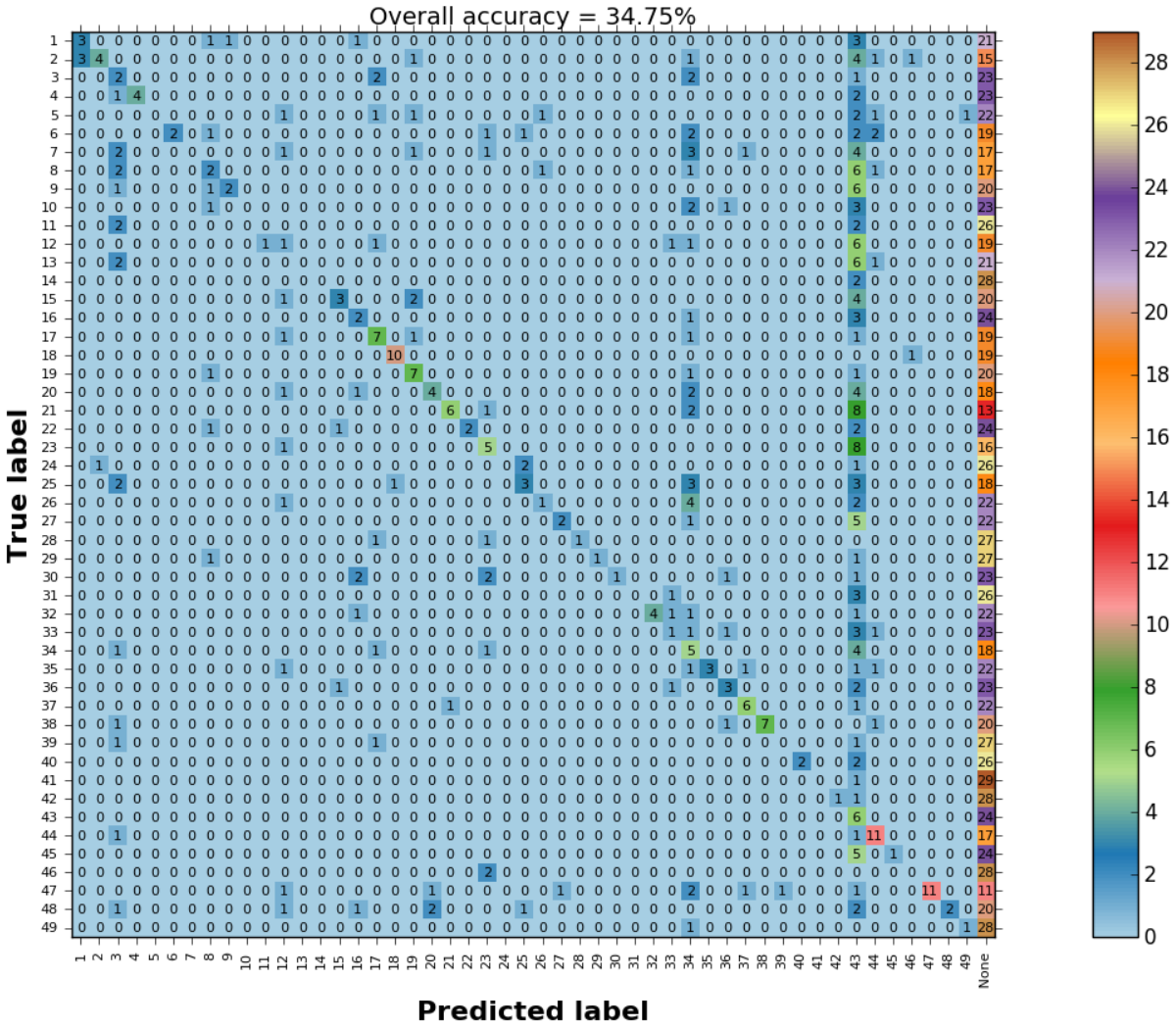


Figure 4.5: Confusion Matrix for indoor snippets

Upon investigation, it was determined to be a consequence of perceptual aliasing. The trial image corresponding to node 43 consists of elements similar in appearance to elements present in other nodes. These contribute towards visually correct yet

physically incorrect predictions. A few examples of this is shown in figures 4.6(a), 4.6(b), 4.6(c) and 4.6.

(a) The reference image matches incorrectly to a snippet 7 nodes ahead of it

(b) The reference image matches incorrectly to a snippet 10 nodes ahead of it

Figure 4.6: Feature matching between different snippets (left) and reference image of node 43 (right)

(c) The reference image matches incorrectly to a snippet 14 nodes ahead of it

(d) The reference image matches incorrectly to a snippet 15 nodes ahead of it

Figure 4.6: Feature matching between different snippets (left) and the reference image of node 43 (right). The matches are a result of the features present in the environment of a certain node showing closer resemblance to that of node 43. This phenomenon is known as *perceptual aliasing* and is an inevitable challenge faced during indoor localization.

When tested using the perspective images captured by an iPhone SE, the algorithm has failed to produce correct predictions in both outdoor and indoor cases. The reason behind this is that the significant distortions present in the reference images (street view images) causing even the scale and rotation invariant feature detectors to fail.

A solution for this is virtual view generation and to use feature extractors that focus on unique features that occur across images. Such a feature extractor that suits an indoor environment better can be constructed by utilizing net-VLAD [140] or a similar *Convolutional Neural Network* (CNN)-based feature extractor.

4.6 Conclusion

The reasonably low values for recall in table 4.4 suggests that the number of false negatives is substantially high. It implies that most of the input images corresponding to a particular node are not identified as belonging to that node. However, from the confusion matrix in figure 4.4, it is evident that these are not incorrectly predicted either. The algorithm was unable to predict a node for those images. The lack of strong features in the image and the failure of the second stage verification could be contributing factors toward it. However, the low values for both precision and recall values for the indoor test case (table 4.5) signify that the number of true positives is low, and the number of false positives being high. But, as discussed under section 4.5.2, this is mainly due to perceptual aliasing. Thus, it can be concluded that the algorithm is capable of predicting the node closest to a given query image.

Nevertheless, when using actual images instead of snippets, the majority of the results are mismatched. The reason for this is the difference in nature between the query and the reference images. The query images are perspective while the reference images are equirectangular. Hence, direct feature matching between the two would result in errors, as shown in figure 4.7. Figures 4.7(a), 4.7(b), and 4.7(c) clearly illustrate how directly matching the two can result in the same query image being matched to different reference images. It is known as perceptual aliasing.

A solution to address the massive difference in viewpoint among the actual and street view images is to generate a perspective image (virtual view) out of the equirectangular reference image and then match it against the query image. Another reason for the erroneous results is the tremendous difference in scale between the query and the reference images (figure 4.8). The scale difference could be too much for the BRIEF descriptor to handle. A solution to this could be to re-scale the query image and the region of interest of the reference image to a comparable scale before matching. Using

image feature detectors such as *Affine Scale Invariant Feature Transform* (ASIFT) that are invariant to affine transformation can improve the performance further. The performance of the place recognition system for indoor environments can be improved by training a place recognition feature extractor suitable for indoor environments such as net-VLAD. This work is currently under progress and will also be addressed as part of future work.

(a) The query image matches incorrectly to a reference image 5 nodes behind the actual

(b) The query image matches incorrectly to a reference image 2 nodes behind the actual

Figure 4.7: Incorrect matches of the same actual image (left) with multiple street view images (right)

(c) The query image matches incorrectly to a reference image 9 nodes ahead of the actual

Figure 4.7: Incorrect matches of the same actual image (left) with multiple street view images (right)

Figure 4.8: Illustration of the disparity between a query image(left) and a reference image(right)

Chapter 5

Experimental Evaluation

This chapter presents experimental evidence as a validation of the facts presented in the preceding chapters. Starting with an introduction to the conducted experiments followed by a few preliminaries related to the experiments, this chapter provides a complete description of the equipment and resources used, the methodology, the generated results and the derived conclusions in succession.

5.1 Introduction

Experimental evaluation is a crucial step in developing any robotic system. It is how the validity of that system under real-world conditions can be assessed. The way to accomplish it by executing the system on a data set relevant to the scenario is captured in the desired environment, under the required conditions. The ground truth for the data set must be available. It is against the ground truth to which the result from the test is compared against. The closer the result resembles the ground truth, the better, and vice-versa.

A quantitative analysis of any deviations from the ground truth and the confidence region of the outcome is necessary for deciding how accurately the result resembles the ground truth. In localization studies, the degree of deviation or the localization error is expressed in terms of *Root Mean Square Error* (RMSE). It is a valuable indicator of the mean localization error along a robot trajectory. A smaller value for RMSE indicates an accurate localization system. However, a higher value for RMSE has two possible causes. Either the entire result has more or less deviated

from the ground truth. Otherwise, most of the result is fairly accurate, while a couple of massive deviations contribute to the overall error. Therefore, often it helps to have an understanding of the maximum and minimum values for the localization error as well.

In terms of the confidence region, the robot's estimated locations must lie well within the estimation's confidence region. It can be easily evaluated by using the covariance information of the estimations. The confidence region is generally defined for three confidence levels based on the assumption that the error distribution is a zero-mean Gaussian-normal distribution. They are 68.3% confidence or one-standard deviation (σ), 95.4% confidence or two-standard deviations (2σ), and 99.7% confidence or three-standard deviations (3σ) as shown in figure 5.1. For a successful localization, the actual robot position should at least lie withing a 3σ confidence region.

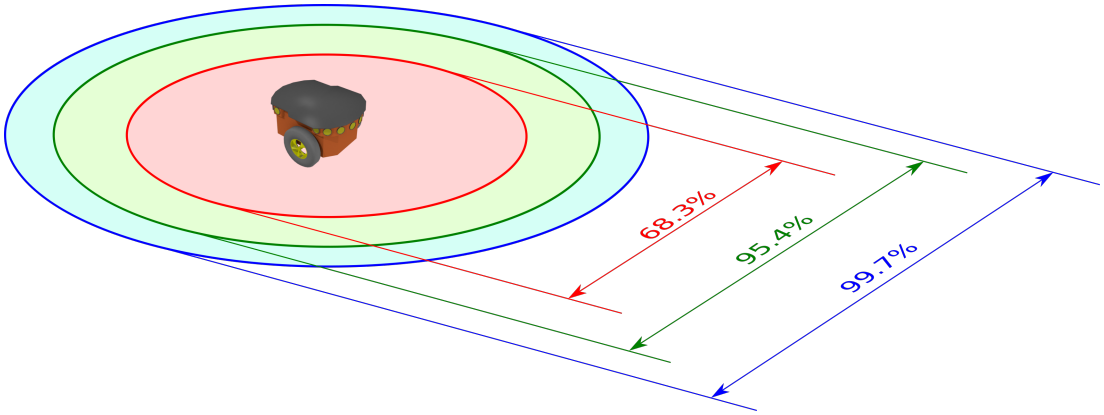


Figure 5.1: Confidence regions of a robot's location

5.2 Equipment and Resources

5.2.1 RICOH THETA S 360° camera

The RICOH THETA S(figure 5.2) is a commercially available 360° camera. Among many other options, this became the choice of interest mainly since it is ROS compatible. Besides, it is compact and mobile. The specifications of the camera, too, met the requirements for the conducted experiments. A couple of relevant specifications of the camera can be found in table 5.1. Further information on the product can be found at [141]

Figure 5.2: RICOH THETA S 360° camera

Table 5.1: Specifications of RICOH THETA S

Specification	**Value**
Resolution	1.5 MP
Sensor size	1/2.3 inch(6.2mm × 4.6mm)
Lens	7.3mm eq. Non-Zoom
ISO range	100 - 1600
Shutter	1/6400 - 60 sec
Max. aperture	2.0
ROS compatible	Yes

5.2.2 Seekur Jr.

Seekur Jr. (figure 5.3) is a 4-wheel drive skid steer differential drive robot. It comes equipped with an assortment of sensors and software making it an excellent platform for localization, Mapping and SLAM related research. It can be controlled wirelessly through a computer or a tethered joystick. Several safety features such as obstacle avoidance, drive-safe mode, and emergency stop make it compliant and safe for indoor use. A couple of Seekur Jr. specifications specific and important in this research can be found in table 5.2. Further information on the Seekur Jr. can be found in [142].

Figure 5.3: Seekur Jr. mobile robot

Table 5.2: Important specifications of Seekur Jr. Mobile robot

Specification	Value
Dimensions	105mm × 840mm × 500mm (lwh)
Weight	70 kg (1 battery)
Steering and Suspension	4-wheel skid steering
Maximum Speed	1.2 mph
Runtime	3 hrs
Payload	50 kg
Sensors	SICK LMS 100 Laser rangefinder
	840-tick/rev Hall effect sensors on traction motors
	MobileRanger C3D obstacle avoidance system
Software	Seekur server operating system
	ARIA API(Custom modified version of ARIA)
	Mapper3 indoor mapping software

5.3 Methodology

5.3.1 Setting up the 360° camera, Seekur Jr. and the Visualizer

The 360° camera

A ROS node was written for the camera to acquire a stream of images in real-time in equirectangular form. It was later used to acquire the visual feedback necessary for the factor graph.

Seekur Jr.

The libraries AriaClientDriver [143] (A custom made driver based on ARIA APL), Mapper3, ARNL server and gmapping necessary for controlling and remotely operating Seekur Jr, and generating maps using it, were installed. The ros package *ros-amcl* was installed to obtain the ground truth from laser scan data.

The Visualizer

The visualizer described under section 3.6 was upgraded by introducing the following features.

1. Display maps created using Seekur Jr. in the form of Occupancy Grids.
2. Display raw Odometry data of Seekur Jr. captured through the wheel encoders.
3. Display laser scan data from Seekur Jr. in the form of point clouds.
4. Display the ground truth of Seekur Jr.calculated through the *ros-amcl* package
5. Display the particle cloud of the particle filter used by amcl.
6. Display the map frame, Odom frame, and all other coordinate frames present in the system(e.g., Seekur Jr.'s base coordinate frame, the laser rangefinder's coordinate frame), in the form of a transformation tree (*tf tree*).

Using this upgraded version of the visualizer and actual data obtained via Seeker Jr.'s sensors, the simulator discussed under section 3.6 was turned into an implementation of the system discussed under section 3.3.

5.3.2 Developing and Testing the experimentation platform

In order to conduct experiments on Essential Matrix accuracy, an experimentation platform was required. Thus, a system to calculate the relative pose among two given was developed. The system is based on the principle of structure-from-motion.
First, an intrinsic matrix for the images is formed using intrinsic parameters, either pre-known, calculated, or obtained through camera calibration. Next, using the lens distortion parameters, the images are undistorted. Then, from the undistorted images, a maximum of 1500 SIFT features are extracted and matched. It is followed by

Lowe's ratio test on the matching features, retaining the strong feature matches , and filtering out the rest. Then, the retained features are normalized[1] [144] using the intrinsic matrix formed earlier. The normalized features are then subjected to a RANSAC + Homography test, through which the Essential Matrix for the two images is determined. The parameters and values used for RANSAC are presented in table 5.3. The number of inliers and the inlier ratio was carefully monitored to ensure that they are more than ten and 20%, respectively. Finally, by decomposing this Essential Matrix into a rotation matrix and an up-to-scale translation (or a translation unit vector), the relative pose between the two camera poses that correspond to the input images can be determined.

Table 5.3: Thresholds used for RANSAC

Parameter	Value
Confidence threshold(%)	99.9
Max. reprojection error(px)	0.0001
Maximum iterations	2000

Testing the system

Two images of the same location in the MUN Engineering basement was acquired at two different points that were Eight feet apart, with a relative turn of 30° between the points. It was performed using an iPhone SE attached to a tripod. A major concern was that the camera and the tripod axes were non-intersecting as seen in figure 5.4.

[1]Normalization refers to transforming the feature point coordinates to have a mean equal to zero and a variance equal to one. It is done in order to make the tasks involving the feature points robust to image noise

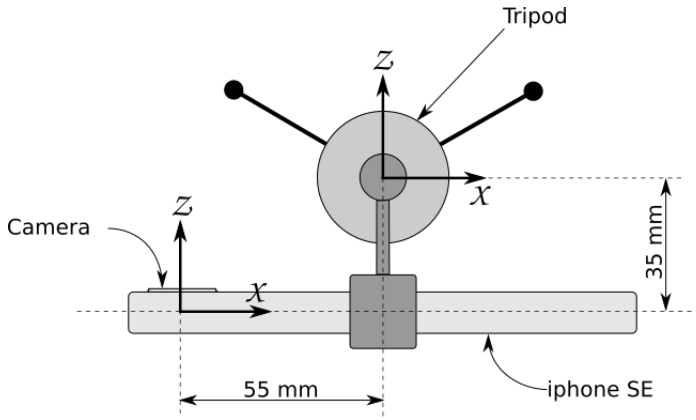


Figure 5.4: The tripod-camera arrangement

Thus, the relative pose between the camera instances was quite different from that of the tripod poses, as shown in figure 5.5

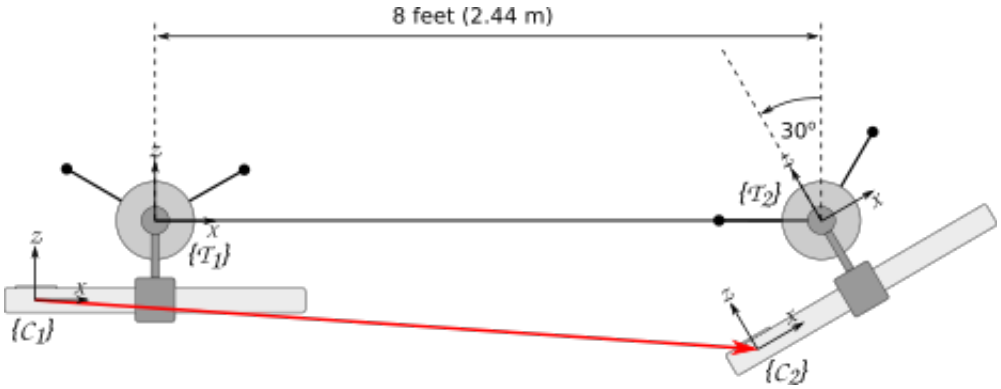


Figure 5.5: The camera and tripod poses between the points

The relative pose between the camera instances was calculated using equations 2.1 and 2.2. The measured and the calculated relative poses presented side-by-side in table 5.4 with the inliers between the images illustrated in figure 5.6.

Table 5.4: Comparison between measured and calculated values

Parameter	Measured	Calculated[2]
x translation	2.463 m	1.000
y translation	0.000 m	0.000
z translation	−0.023 m	0.000
Roll	0.0 °	0.2 °
Pitch	−30.0 °	−30.4 °
Yaw	0.0 °	0.5 °

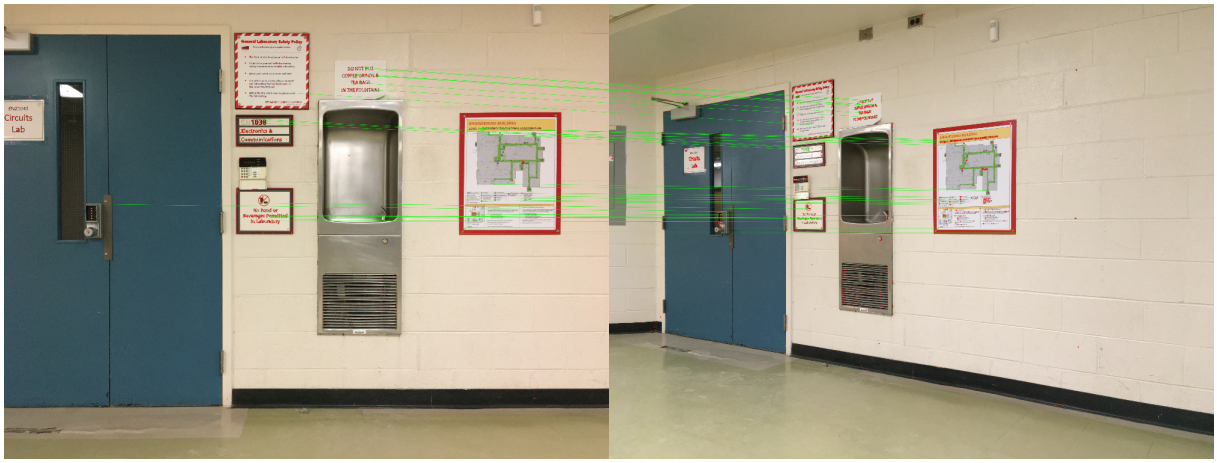


Figure 5.6: Inliers and Outliers between the test images

The test run results provided significant proof regarding the system's accuracy, deeming the suitability of it for the upcoming experiments.

5.3.3 Camera Calibration

For the test described under section 5.3.2, the intrinsic parameters of the iPhone SE used were required. Thus, it was calibrated using a checkerboard target (figure 5.7). The intrinsic parameters and distortion coefficients obtained are tabulated in table

[2]The calculated translation is in the normalized form

5.5. The extrinsic parameter visualization and the reprojection error plot for the calibration are shown in figures 5.8 and 5.9.

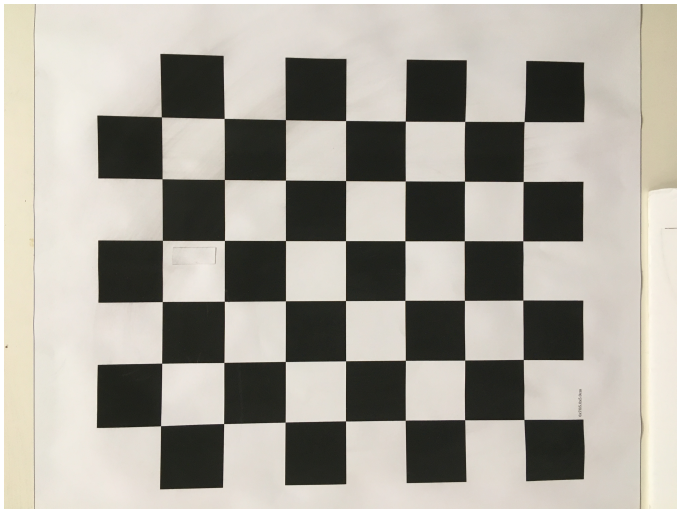


Figure 5.7: The checkerboard target used for calibration

Table 5.5: Intrinsic parameters and distortion coefficients of the iphone SE

Parameter	Value
Focal length(f_x)	3431.7
Focal length(f_y)	3429.3
Image centre	(2009.8 , 1517.9)
Tangential distortion coeffcients	(0.0 , 0.0)
Radial distortion coeffcients	(0.0869 , -0.0761)

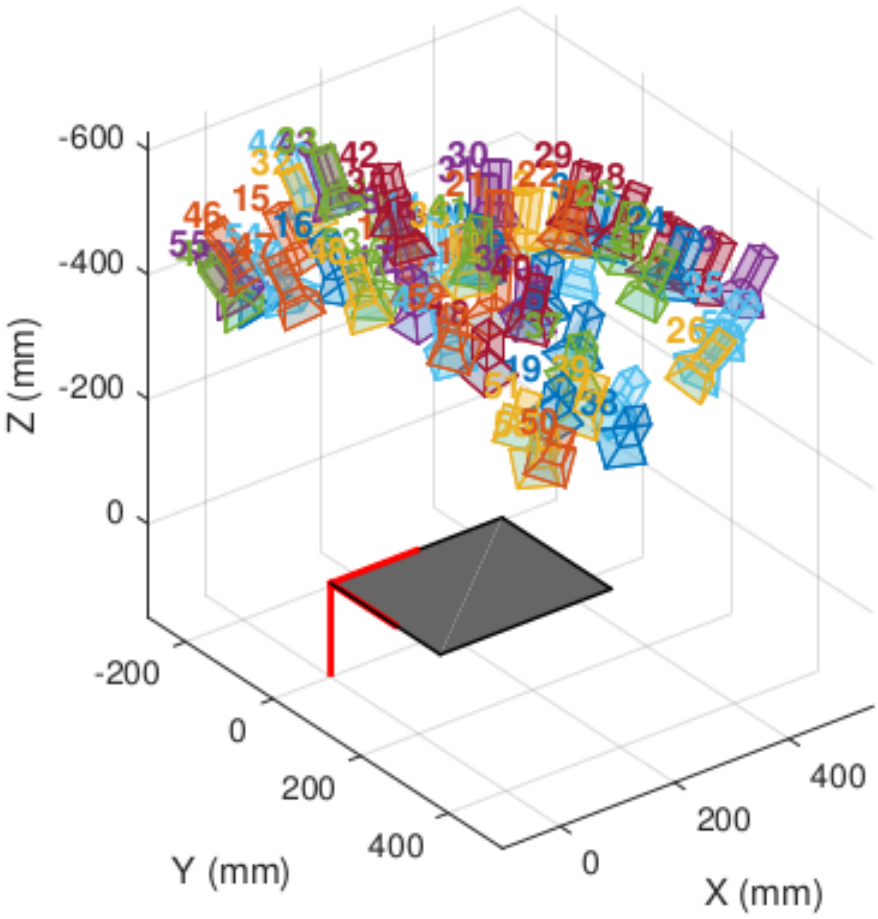


Figure 5.8: Extrinsic parameter visualization

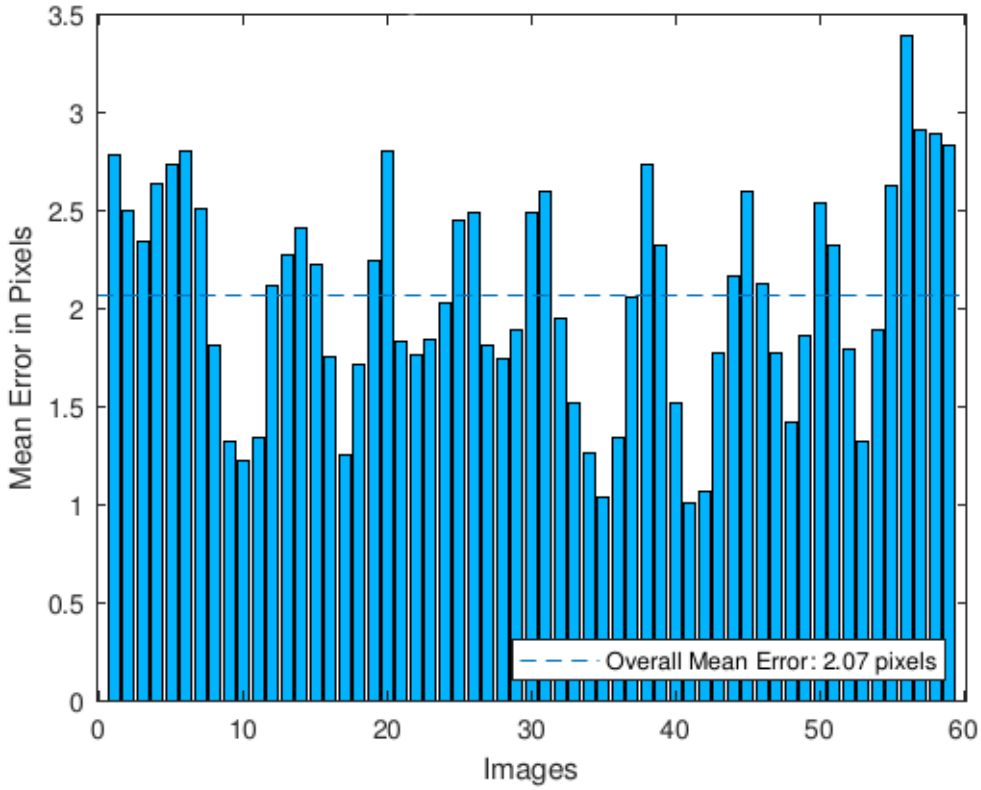


Figure 5.9: Reprojection error for the images

5.3.4 Mapping the MUN Engineering building basement and collecting ros-bags

A map of the MUN Engineering basement in the form of an *Occupancy Grid* was required for calculating the ground truth. Using Seekur Jr., its laser rangefinder and Mapper3 software, a map of the basement shown in figure 5.10, was generated. This map was further refined such that it overlaps correctly with the node map of the basement (figure 4.5.1), and the map shown in figure 5.11 was obtained.

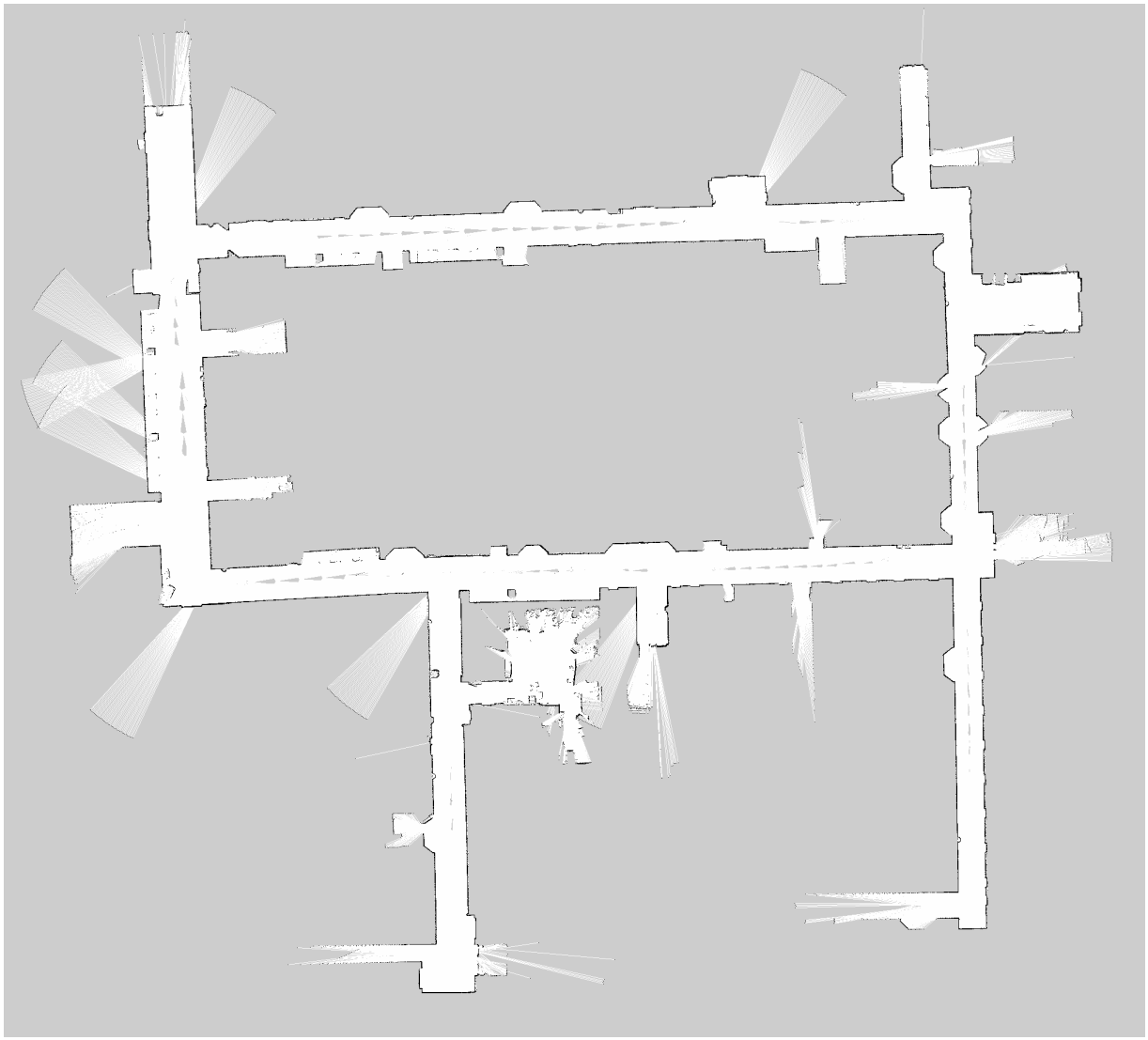


Figure 5.10: Map of the MUN Engineering basement generated using Seekur Jr.

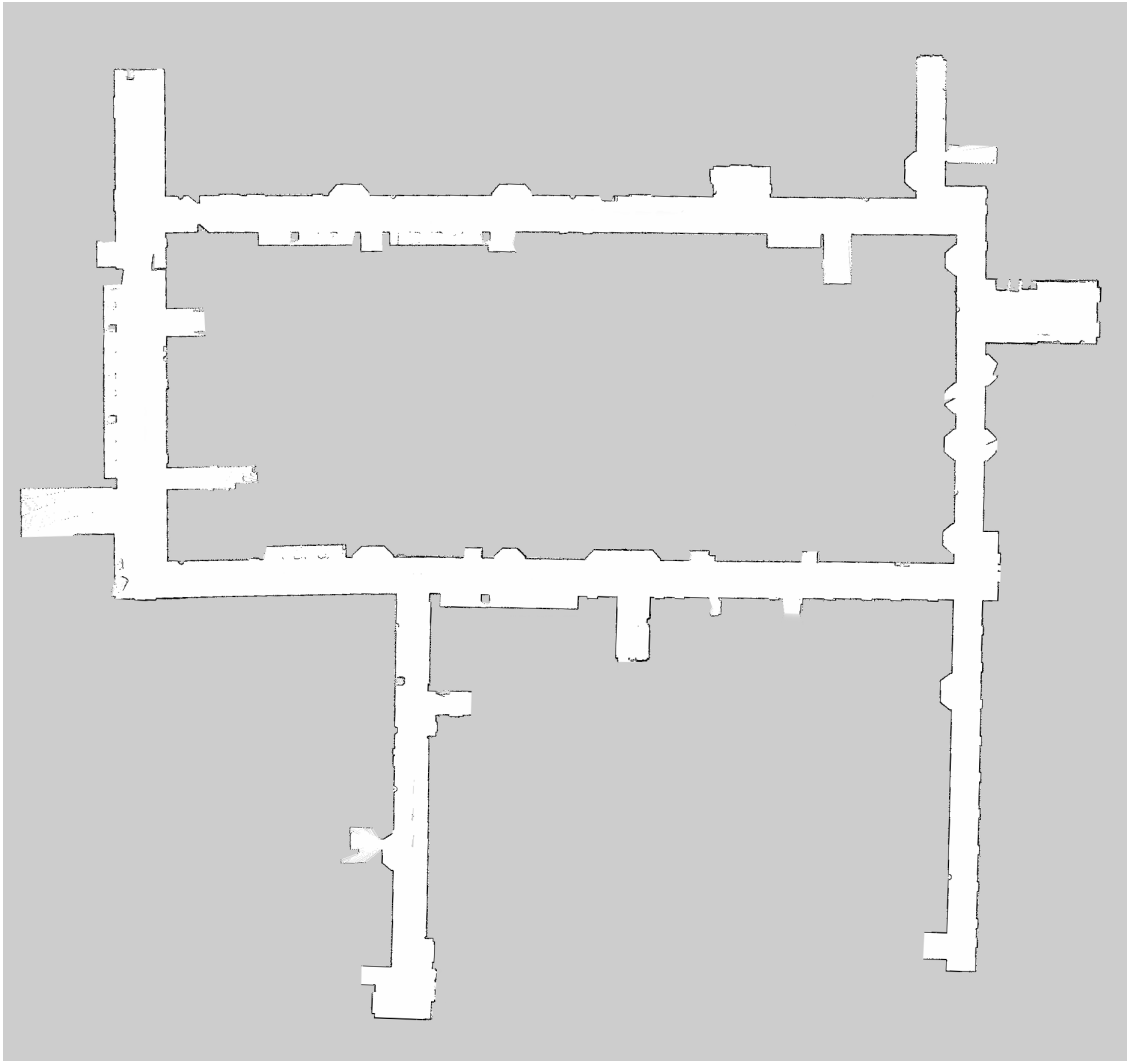


Figure 5.11: Map of basement after refinement

Once this was complete, a rosbag was captured while navigating the Seekur Jr. around the corridor, returning to the starting point, thus making an anti-clockwise loop. The rosbag captured consists of the odometry data, laser scan data and a 360° image stream captured using the RICOH THETA camera mounted on top of it.

5.3.5 The place recognition module

The place recognition module required to determine the node on the map closest to a given robot pose is being developed separately. At this stage, it is not yet at a level to effectively combine with the rest of the system. Therefore, the place recognition at

the current stage is performed manually.

5.3.6 Evaluating Essential Matrix accuracy

To evaluate the accuracy of the Essential matrix derived using virtual views, two virtual views, generated from two different Google Street View images were required. Using a freely available online tool [145], two virtual views, 640 px × 640 px in size and differ by 60° in viewpoint, were generated from the Street View images of two adjacent nodes of the MUN Engineering basement. Next, the field-of-view of the images was determined to be 90° using equation 2.3. Then, an equivalent intrinsic matrix for the images were determined using equations 2.4 and 2.8. Finally, the images were fed to the platform discussed under section 5.3.2. In this case, however, since the rotation of the camera(viewpoint) was about its axis, correcting the poses were unnecessary.

5.3.7 Evaluating Factor Graph with simulated measurements

After a series of successful simulations, the next step was to evaluate how well the factor graph responds to actual data. Since a *ros-bag* of the Seekur Jr. navigating the MUN Engineering basement was already available (section 5.3.4), it was utilized as a dataset to run a a couple of offline tests. The distance and angle thresholds for the factor graph, i.e., the minimum distance and angle by which the robot should move relative to the preceding node, for a new node to be added to the factor graph, was arbitrarily fixed at 5m and 45° respectively. The ground truth of the robot was estimated using the laser scan data from the ros bag, and a particle filter from the *ros-amcl* package. As the objective here was to test how well the factor graph itself performs under actual data, the Essential Matrix constraints were computed manually and emplaced in the graph, rather than having them computed automatically using the feedback captured from the image stream.

5.4 Results

5.4.1 Essential Matrix accuracy

Figure 5.12 shows the resulting inliers and outliers when the two virtual views mentioned under section 5.3.6 are used to estimate the relative pose shift among the two views.

It provides insight as to how the algorithm has filtered outliers and retained inliers. Additionally, table 5.6 presents a comparison between the measured and calculated relative translation and rotation. The algorithm has been capable of estimating the relative pose with decent accuracy. However, this should be further verified using several repeatability studies. The translation unit vector has been precisely estimated while the orientation is accurate to within 2 °, which would more than suffice for all practical purposes.

Figure 5.12: Inliers (green) and Outliers (red) between the matched virtual views

Table 5.6: Comparison between measured and calculated values

Parameter	Measured	Calculated
x translation[3]	−1.0	−1.0
y translation	0.0	0.0
z translation	0.0	0.0
Roll	0.0 °	1.5 °
Pitch	60.0 °	59.0 °
Yaw	0.0 °	1.4 °

5.4.2 Factor Graph with simulated measurements

A visual comparison between the odometry and the ground truth of Seekur Jr. is presented in figure 5.13. In comparison to the ground truth, the drift in the odometer is evident in the figure, which is the one outlined in blue.

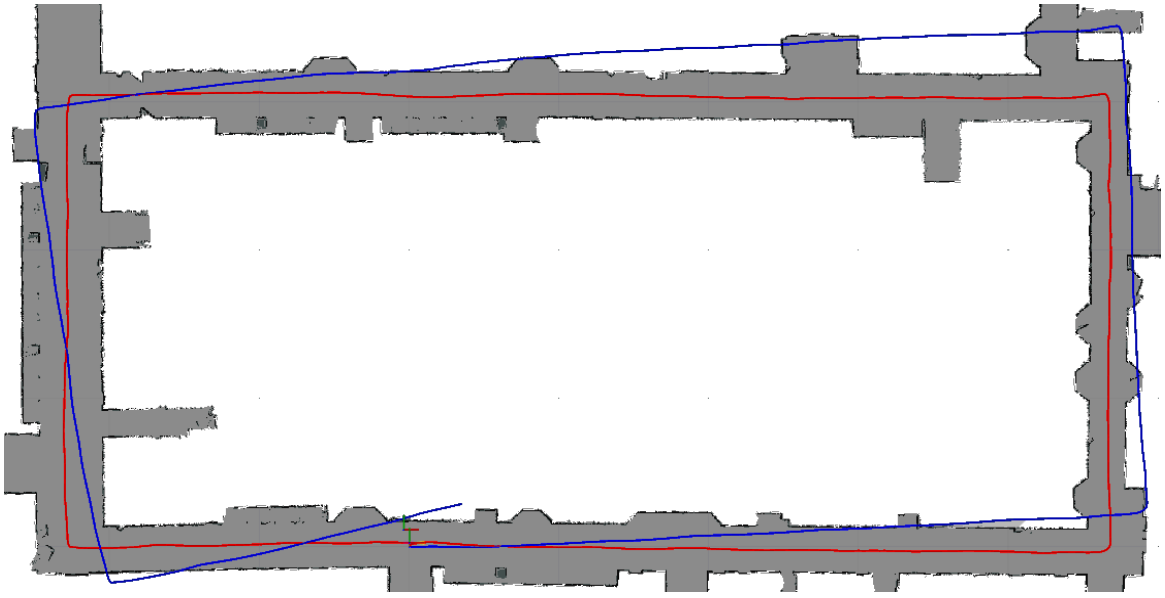


Figure 5.13: Odometry (blue) and ground truth (red)

[3]The x, y, z translations are in the normalized form

Like the simulations, essential matrix constraints were administered at every other node, as illustrated in figure 5.14.

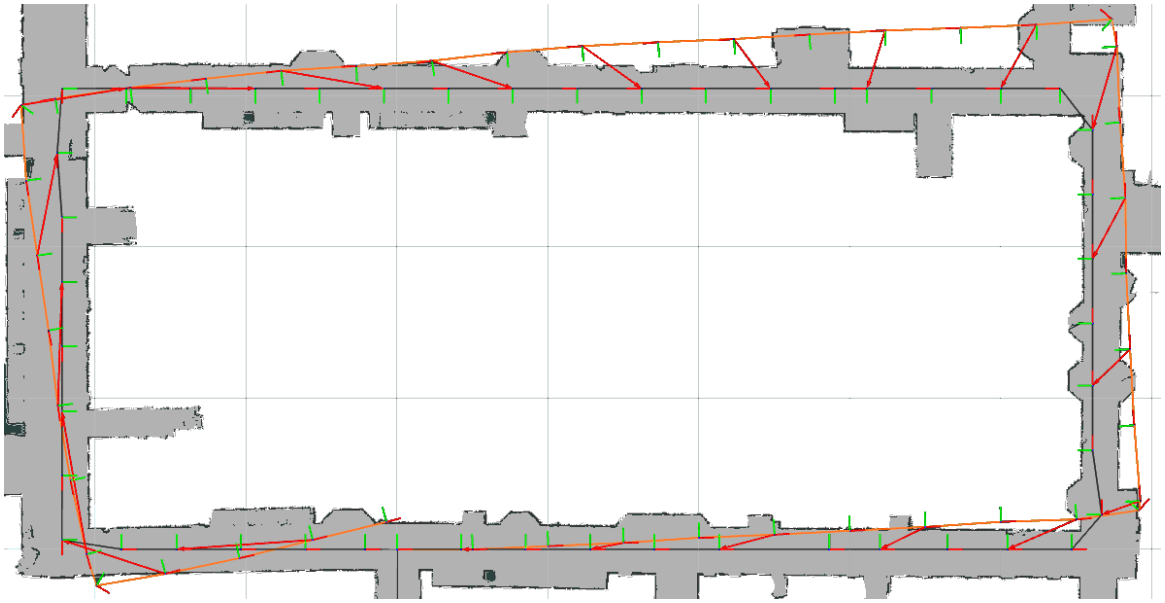


Figure 5.14: Robots poses (orange), map nodes (black) and essential matrix constraints (red arrows)

Figure 5.15 compares the factor graph solution with the ground truth of Seekur Jr. The solution is in good agreement with the ground truth in this particular experiment.

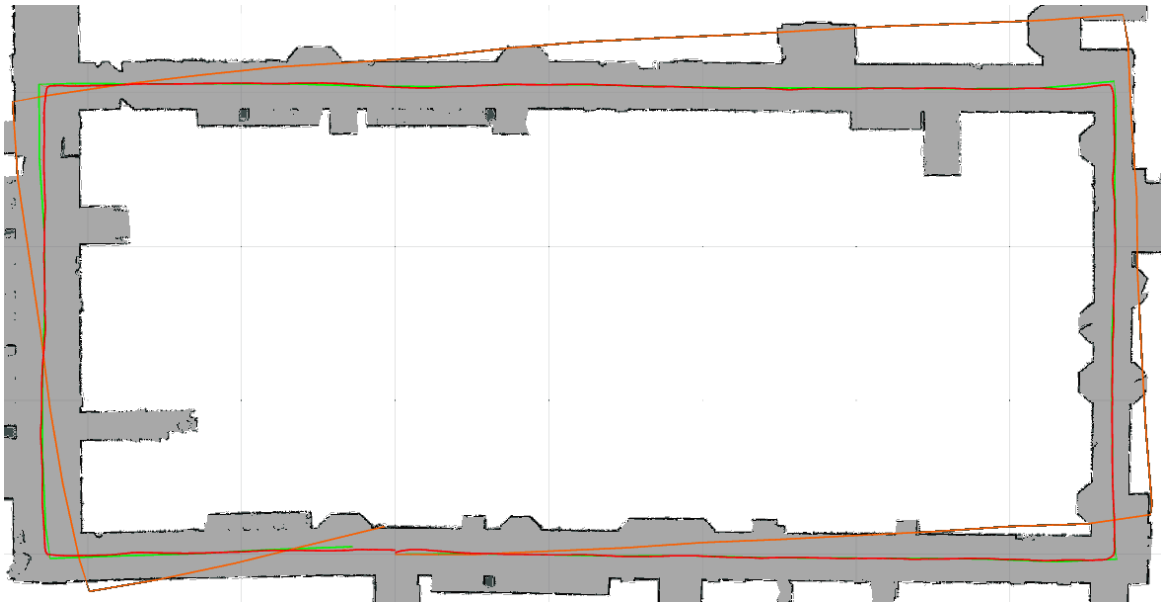


Figure 5.15: Odometry (orange), ground truth (red) and the solution[4] (green)

The error ellipsoids for 99% confidence illustrated in figure 5.16 shows a consistent estimation of the poses, i.e., the error ellipsoids are significantly smaller, yet, the actual robot poses are situated within the confidence region of the estimated robot poses.

Figure 5.16: Error ellipsoids for a confidence of 99% (3 standard deviations)

Furthermore, the error statistics presented under table 5.7 provides quantitative performance results regarding the performance of the factor graph for the considered experimental scenario.

Table 5.7: Overall localization error

Error	**Position (m)**	**Orientation (°)**
Max	0.899	8.292
Min	0.0244	0.002 52
RMSE	0.382	1.783

[4]The solution is almost invisible since it overlaps with the ground truth

5.5 Conclusion

Figure 5.12 shows the inliers and outliers between the two virtual views used to verify the essential matrix accuracy. It is obvious that the pose recovery system has successfully filtered out the outliers and estimated the relative pose between the two images accurately. However, the rotation recovered has a maximum error of 2°. There are a couple of factors contributing to this error. Although figure 5.12 consists of a decent number of inliers, when examined carefully, it is visible that a significant number of feature matches that are inliers have been incorrectly categorized as outliers. For an accurate pose recovery, the number of strong feature matches has to be high in addition to the inlier ratio being high. It ensures that the inlier ratio is more reliable and less sensitive to perturbations, thus, providing a much better insight into the pose recovery process. It can be accomplished by extracting and matching dense features rather than sparse features. It is an aspect that can be worked on as future work.

Figures 5.15, 5.16 and table 5.7 show that factor graphs can be successfully used for indoor localization of mobile robots when an accurate place recognition system is available.

For this study, the essential matrices are simulated since the results presented are a module wise evaluation. The accuracy of the essential matrix is evaluated separately under section 5.3.6. However, a system-level evaluation combining everything is needed for a reasonable validation of the performance. It remains as an activity to be continued as part of future work.

Chapter 6

Conclusion

The objectives of this thesis were developing a factor-graph for a visual topological localization system, developing a place recognition system and validating the factor-graph based localization system for an indoor data set.

For the first objective, a proof of concept was made through MATLAB simulations, which was later developed into a ROS C++ based simulator constituting of all the necessary tools and features. The conducted simulations established a firm ground for the proposed indoor localization system. Also, a pose recovery system was developed for estimating the relative pose shift of a camera between two image views. The simulations provided significant insight regarding when to add new nodes to the factor graph, when to establish constraints using visual feedback, how to do it, and how frequently, and when to solve the factor graph. Besides, it also served in determining the threshold values for specific parameters of the system.

Under the second objective, a Google street view based brute-force place recognition system was developed. The performance of the place recognition system was assessed by applying it for a street view snippets and evaluating the precision, recall, and F1-score values. The system was tested further by applying it to outdoor and indoor environments using both snippets and actual images. The results obtained for snippets achieved an accuracy of 96.74% for outdoor datasets, while for indoor datasets, an accuracy of 34.75% was achieved. However, the results for the actual images suggested that matching them directly against equirectangular street view images mostly causes the closest node to be unidentified and seldom results in incorrect matches. Thus, a

necessity for virtual views of the street view images was observed. A place recognition feature extractor for indoor environments such as net-VLAD is required to improve the performance in indoor environments.

Under the third objective, the factor-graph was validated using a custom data set, and the pose recovery system was validated using virtual views of indoor street view images. The pose recovery system exhibited an acceptable performance, and so did the factor-graph. The solution to the factor graph was in close agreement with the ground truth, and the actual robot poses were located within the confidence region of the solved poses.

Numerous challenges were encountered in accomplishing the above objectives. The lack of indoor street view maps and proper testing and validation datasets was one of the significant limitations identified. Thus, custom street view maps and validation data sets had to be created, which were long and tedious processes.

Another limitation encountered was the absence of sufficient documentation and examples for specific software libraries used in this research, which was under development. Besides, the version incompatibilities of specific dependencies of those libraries consumed a significant amount of time to get familiar with the library. Also, the unavailability of some required functionalities in the MATLAB versions of the libraries imposed limitations on the initial simulations conducted.

The laptop's hardware limitations for the simulations and experiments caused the simulations to run for hours and, on some occasions, days. Specific alterations had to be made to the operating system, which on a couple of occasions resulted in system crashes which required reinstalling the operating system and all the required software libraries and drivers.

The lack of visualization tools for C++ was one of the most significant limitations encountered in this research. Thus, own visualization tools had to be created along with a few other function libraries. It too consumed a significant amount of time

Finding the proper drivers for Seekur Jr. was another hurdle that was faced during this research. The company that manufactured Seekur Jr no longer exists. Hence,

obtaining the drivers necessary for Seekur Jr. was a hectic task which required days of surfing the internet.

Another limitation was the lack of tools to convert equirectangular images into perspective images. Although a couple of software tools were available, the perspective images they produced contained a substantial amount of noise

The lack of studies that use Google street view for indoor localization created a deficit for benchmark results to compare against. Hence, it was unable to obtain a comparative evaluation regarding the performance of the system.

The university's lockdown due to COVID-19 pandemic imposed limitations on university access, which caused difficulties in obtaining certain measurements related to the Seekur Jr. and the MUN engineering basement corridor. Thus, the values for these measurements were estimated where necessary. It also hindered the use of alternate methods that could have otherwise be used to replace the simulated essential matrix constraints with actual ones.

The objectives of this thesis were achieved up to a certain extent. The following limitations of the proposed approach can be identified.

1. The place recognition is not accurate enough to be combined with the rest of the system. Virtual views, better descriptors, and *Bag-of-Words* (BoW) should be used for execution.

2. The factor graph should be evaluated with the inputs coming from a place recognition system.

3. Performance on changing conditions of lighting and dynamic environments should be evaluated along with metrics on the computational time required for graph solutions.

However, a significant challenge that still prevails is combining the place recognition system, the factor-graph, and the pose recovery system into a single autonomous system. Additionally, incorporating virtual views and visual bag-of-words based search methods are potential improvements towards a better system. It poses a

directive for future work related to the proposed indoor localization system. Furthermore, applications of the localization system in automating tasks such as indoor patrol and creating indoor street view maps are some other areas that are worth looking into.

The author addresses the challenges encountered while developing a fully autonomous indoor localization as part of his doctoral studies. Upon its completion, the author intends to forward a paper for a journal publication. The software, hardware, and visualization tools developed for this project can be re-used for future research. The knowledge and skills acquired during the project will act as a stepping stone for the future work by the author.

Ultimately, developing a Google indoor street view and factor graph-based fully autonomous indoor localization system would open numerous avenues in the future, such as building-wide localization, indoor mapping, indoor patrolling, security, and surveillance. The ease of extending to use a multitude of sensors, efficient use of hardware, reduced requirement for tuning, ability to easily migrate between indoor and outdoor environments, and ability to effectively utilize sensor information available at any given time can be seen as advantages of using such a system for indoor localization.

Bibliography

[1] F. E. Schneider and D. Wildermuth, "Using robots for firefighters and first responders: Scenario specification and exemplary system description," in *2017 18th International Carpathian Control Conference, ICCC 2017*, 2017, pp. 216–221.

[2] J. S. Jennings, G. Whelan, and W. F. Evans, "Cooperative search and rescue with a team of mobile robots," in *International Conference on Advanced Robotics, Proceedings, ICAR*, jul 1997, pp. 193–200.

[3] S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Hahnel, M. Montemerlo, A. Morris, Z. Omohundro, C. Reverte, and W. Whittaker, "Autonomous exploration and mapping of abandoned mines: Software architecture of an autonomous robotic system," *IEEE Robotics and Automation Magazine*, vol. 11, no. 4, pp. 79–91, 2004.

[4] S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker, "A system for volumetric robotic mapping of abandoned mines," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 3, 2003, pp. 4270–4275.

[5] H. Miura, A. Watanabe, M. Okugawa, S. Kurahashi, M. Kurisu, and T. Miura, "Field experiment report for verification of abandoned lignite mines by robotic exploration system," in *Journal of Robotics and Mechatronics*, vol. 30, no. 6, 2018, pp. 1004–1013.

[6] L. Zhang, Y. Du, and A. Cao, "The design of natural gas pipeline inspection robot system," in *2015 IEEE International Conference on Information and Automation, ICIA 2015 - In conjunction with 2015 IEEE International Conference on Automation and Logistics*, 2015, pp. 843–846.

[7] J. Y. Choi, H. Lim, and B. J. Yi, "Semi-automatic pipeline inspection robot systems," in *2006 SICE-ICASE International Joint Conference*, 2006, pp. 2266–2269.

[8] F. Yuan and L. Wang, "The design and study of the drainage pipelines dredging robot," in *2010 International Conference on Computing, Control and Industrial Engineering, CCIE 2010*, vol. 1, 2010, pp. 17–20.

[9] M. J. Schuster, C. Brand, S. G. Brunner, P. Lehner, J. Reill, S. Riedel, T. Bodenmüller, K. Bussmann, S. Büttner, A. Dömel, W. Friedl, I. Grixa, M. Hellerer, H. Hirschmüller, M. Kassecker, Z. C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler, "The LRU Rover for Autonomous Planetary Exploration and Its Success in the SpaceBotCamp Challenge," in *Proceedings - 2016 International Conference on Autonomous Robot Systems and Competitions, ICARSC 2016*, 2016, pp. 7–14.

[10] F. Cordes, S. Planthaber, I. Ahrns, T. Birnschein, S. Bartsch, and F. Kirchner, "Cooperating reconfigurable robots for autonomous planetary sample return missions," in *Proceedings of the 2009 ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots, ReMAR 2009*, 2009, pp. 665–673.

[11] P. Fiorini, "Ground mobility systems for planetary exploration," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 1, 2000, pp. 908–913.

[12] K. S. Hwang, K. J. Park, D. H. Kim, S. S. Kim, and S. H. Park, "Development of a mobile surveillance robot," in *ICCAS 2007 - International Conference on Control, Automation and Systems*, 2007, pp. 2503–2508.

[13] K. Kim, S. Bae, and K. Huh, "Intelligent surveillance and security robot systems," in *Proceedings of IEEE Workshop on Advanced Robotics and its Social Impacts, ARSO*, 2010, pp. 70–73.

[14] B. Wei, J. Gao, J. Zhu, and K. Li, "Design of a large explosive ordnance disposal robot," in *2009 2nd International Conference on Intelligent Computing Technology and Automation, ICICTA 2009*, vol. 3, 2009, pp. 403–406.

[15] S. Prabakaran, S. Sharon Rosy, and S. Shakena Grace, "Identification of explosives and disaster prevention using intelligent robots," in *International Conference on "Emerging Trends in Robotics and Communication Technologies", INTERACT-2010*, 2010, pp. 209–212.

[16] A. K. Bin Motaleb, M. B. Hoque, and M. A. Hoque, "Bomb disposal robot," in *2016 International Conference on Innovations in Science, Engineering and Technology (ICISET)*, 2017, pp. 1–5.

[17] Lindsey, Quentin and Mellinger, Daniel and Kumar, Vijay, "Construction of cubic structures with quadrotor teams," in *Robotics: Science and Systems*, vol. 7, Los Angeles, CA, USA, jun 2012, pp. 177–184.

[18] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction of cubic structures with quadrotor teams," in *Robotics: Science and Systems*. Los Angeles, CA, USA: MIT Press, jun 2012, vol. 7, pp. 177–184. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6301066

[19] F. Augugliaro, E. Zarfati, A. Mirjan, and R. D'Andrea, "Knot-tying with flying machines for aerial construction," in *IEEE International Conference on Intelligent Robots and Systems*, vol. 2015-Decem, sep 2015, pp. 5917–5922.

[20] K. Singh and K. Fujimura, "Map making by cooperating mobile robots," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2, 1993, pp. 254–259.

[21] A. Y. Hata and D. F. Wolf, "Outdoor mapping using mobile robots and laser range finders," in *CERMA 2009 - Electronics Robotics and Automotive Mechanics Conference*, 2009, pp. 209–214.

[22] e. R. Wurman, R. D'Andrea, M. T. Barbehenn, A. E. Hoffman, and M. C. Mountz, "A method for transporting inventory items includes moving," jul 2018.

[23] N. Michael, J. Fink, and V. Kumar, "Cooperative manipulation and transportation with aerial robots," *Autonomous Robots*, vol. 30, no. 1, pp. 73–86, jan 2011. [Online]. Available: https://doi.org/10.1007/s10514-010-9205-0

[24] R. Ritz and R. D'Andrea, "Carrying a flexible payload with multiple flying vehicles," in *IEEE International Conference on Intelligent Robots and Systems*, nov 2013, pp. 3465–3471.

[25] Anthony Stentz, Mark Ollis, Kerien Fitzpatrick, Regis Hoffman, and William Whittaker, "Vision-based crop line tracking for harvesters," apr 1997.

[26] J. Das, G. Cross, C. Qu, A. Makineni, P. Tokekar, Y. Mulgaonkar, and V. Kumar, "Devices, systems, and methods for automated monitoring enabling precision agriculture," in *IEEE International Conference on Automation Science and Engineering*, vol. 2015-Octob, aug 2015, pp. 462–469.

[27] G. Silveira, E. Malis, and P. Rives, "An efficient direct approach to visual SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 969–979, oct 2008.

[28] S. Weiss, D. Scaramuzza, and R. Siegwart, "Monocular-SLAM-based navigation for autonomous micro helicopters in GPS-denied environments," *Journal of Field Robotics*, vol. 28, no. 6, pp. 854–874, 2011. [Online]. Available: http://dx.doi.org/10.1002/rob.20412

[29] B. Steder, G. Grisetti, C. Stachniss, and W. Burgard, "Visual SLAM for flying vehicles," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1088–1093, 2008.

[30] S. Y. Hwang and J. B. Song, "Monocular vision-based SLAM in indoor environment using corner, lamp, and door features from upward-looking camera," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 10, pp. 4804–4812, 2011.

[31] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular camera localization in 3D LiDAR maps," in *IEEE International Conference on Intelligent Robots and Systems*, vol. 2016-Novem. IEEE, 2016, pp. 1926–1931.

[32] J. Biswas and M. Veloso, "WiFi localization and navigation for autonomous indoor mobile robots," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 4379–4384.

[33] J. J. Robles, "Indoor localization based on wireless sensor networks," *AEU - International Journal of Electronics and Communications*, vol. 68, no. 7, pp. 578–580, 2014.

[34] "Equirectangular—Help | ArcGIS for Desktop." [Online]. Available: https://desktop.arcgis.com/en/arcmap/10.3/guide-books/map-projections/equirectangular.htm

[35] H. P. Chiu, S. Williams, F. Dellaert, S. Samarasekera, and R. Kumar, "Robust vision-aided navigation using Sliding-Window Factor graphs," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 46–53.

[36] J. Dong, M. Mukadam, F. Dellaert, and B. Boots, "Motion planning as probabilistic inference using Gaussian processes and factor graphs," in *Robotics: Science and Systems*, vol. 12, 2016, p. 4.

[37] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 3607–3613.

[38] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[39] P. Agarwal, W. Burgard, and L. Spinello, "Metric localization using Google Street View," in *IEEE International Conference on Intelligent Robots and Systems*, vol. 2015-Decem, sep 2015, pp. 3111–3118.

[40] A. L. Majdik, Y. Albers-Schoenberg, and D. Scaramuzza, "MAV urban localization from Google street view data," in *IEEE International Conference on Intelligent Robots and Systems*, nov 2013, pp. 3979–3986.

[41] S. V. Ramani and Y. N. Tank, "Indoor Navigation on Google Maps and Indoor Localization Using RSS Fingerprinting," *International Journal of Engineering Trends and Technology*, vol. 11, no. 4, pp. 171–173, 2014.

[42] X. Yan, Z. Shi, and Y. Zhong, "Vision-based Global Localization of Unmanned Aerial Vehicles with Street View Images," in *Chinese Control Conference, CCC*, vol. 2018-July. IEEE, 2018, pp. 4672–4678.

[43] L. Yu, C. Joly, G. Bresson, and F. Moutarde, "Monocular urban localization using street view," in *2016 14th International Conference on Control, Automation, Robotics and Vision, ICARCV 2016*. IEEE, 2017, pp. 1–6.

[44] A. R. Zamir and M. Shah, "Accurate image localization based on google maps street view," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6314 LNCS, no. PART 4. Springer, 2010, pp. 255–268.

[45] H. Sadeghi, S. Valaee, and S. Shirani, "2DTriPnP: A Robust Two-Dimensional Method for Fine Visual Localization Using Google Streetview Database," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4678–4690, 2017.

[46] M. Salarian, A. Manavella, and R. Ansari, "Accurate localization in dense urban area using Google street view images," in *IntelliSys 2015 - Proceedings of 2015 SAI Intelligent Systems Conference*. IEEE, 2015, pp. 485–490.

[47] M. B. Kjærgaard, H. Blunck, T. Godsk, T. Toftkjær, D. L. Christensen, and K. Grønbæk, "Indoor positioning using GPS revisited," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6030 LNCS. Springer, Berlin, Heidelberg, 2010, pp. 38–56. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-12654-3_3

[48] "AN-000011 COMPASSES/MAGNETOMETERS MAGNETIC DISTORTION Additive Fields and the Effects of Hard Iron Materials," InvenSense, Inc., Tech. Rep., 2014. [Online]. Available: https://invensense.tdk.com/developers/wp-content/uploads/sites/2/2015/04/Compass-Magnetometer-AN-000011v1-0.pdf

[49] A. Pronobis and B. Caputo, "COLD: The CoSy localization database," *International Journal of Robotics Research*, vol. 28, no. 5, pp. 588–594, may 2009. [Online]. Available: http://www.pronobis.pro/publications/pronobis2009ijrr

[50] J. Thoma, D. P. Paudel, A. Chhatkuli, T. Probst, and L. V. Gool, "Mapping, localization and path planning for image-based navigation using visual features and map," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2019-June, 2019, pp. 7375–7383.

[51] P. Newman, D. Cole, and K. Ho, "Outdoor SLAM using visual appearance and laser ranging," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2006, may 2006, pp. 1180–1187.

[52] S. Shen, N. Michael, and V. Kumar, "Autonomous indoor 3D exploration with a micro-aerial vehicle," in *Proceedings - IEEE International Conference on Robotics and Automation*, may 2012, pp. 9–15.

[53] Shen, Shaojie and Michael, Nathan and Kumar, Vijay, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *Proceedings - IEEE International Conference on Robotics and Automation*, may 2011, pp. 20–25.

[54] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[55] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Factor graph based incremental smoothing in inertial navigation systems," in *15th International Conference on Information Fusion, FUSION 2012*. IEEE, 2012, pp. 2154–2161.

[56] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *International workshop on vision algorithms*. Springer, 1999, pp. 298–372.

[57] E. Remolina and B. Kuipers, "Towards a general theory of topological maps," *Artificial Intelligence*, vol. 152, no. 1, pp. 47–104, 2004.

[58] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artificial Intelligence*, vol. 99, no. 1, pp. 21–71, 1998.

[59] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 915–926, 2008.

[60] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," *Proceedings of the National Conference on Artificial Intelligence*, vol. 593598, pp. 593–598, 2002.

[61] H. Choset and K. Nagatani, "Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 2, pp. 125–137, 2001.

[62] B. Kuipers and Y. T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Robotics and Autonomous Systems*, vol. 8, no. 1-2, pp. 47–63, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/092188909190014C

[63] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2, apr 2000, pp. 1023–1029.

[64] J. L. Blanco, J. A. Fernández-Madrigal, and J. González, "Toward a unified Bayesian approach to hybrid metric - Topological SLAM," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 259–270, 2008.

[65] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray, "Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2007, pp. 4102–4107.

[66] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. M. Montiel, "Towards semantic SLAM using a monocular camera," in *IEEE International Conference on Intelligent Robots and Systems*, 2011, pp. 1277–1284.

[67] C. Martin, "How low can you go?" *Fire Prevention and Fire Engineers Journals*, vol. 32, no. DEC., pp. 766–789, 2006.

[68] Y. Latif, G. Huang, J. Leonard, and J. Neira, "An Online Sparsity-Cognizant Loop-Closure Algorithm for Visual Navigation," in *Robotics: Science and Systems*, 2015.

[69] D. Wilbers, C. Merfels, and C. Stachniss, "Localization with sliding window factor graphs on third-party maps for automated driving," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, 2019, pp. 5951–5957.

[70] M. Whitty, *Robotics, Vision and Control. Fundamental Algorithms in MATLAB*. Springer, 2012, vol. 39, no. 6.

[71] F. Dellaert, "Factor Graphs and {GTSAM}," *CP&R Technical Report ; GT-RIM-CP&R-2012-002*, no. GT-RIM-CP&R-2012-002, sep 2012. [Online]. Available: http://tinyurl.com/gtsam.

[72] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2006, 2006, pp. 2262–2269.

[73] U. Frese and L. Schroder, "Closing a Million-Landmarks Loop," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 5032–5039.

[74] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006. [Online]. Available: https://doi.org/10.1177/0278364906072768

[75] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[76] G. Grisetti, C. Stachniss, and W. Burgard, "Nonlinear constraint network optimization for efficient map learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 428–439, 2009.

[77] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient sparse pose adjustment for 2D mapping," in *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 2010, pp. 22–29.

[78] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 3281–3288.

[79] L. Polok, V. Ila, M. Solony, P. Smrz, and P. Zemcik, "Incremental Block Cholesky Factorization for Nonlinear Least Squares in Robotics," in *Robotics: Science and Systems*, 2016, pp. 328–336.

[80] S. Agarwal, K. Mierle, and Others, "Ceres Solver," 2020. [Online]. Available: http://ceres-solver.org

[81] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, dec 2016.

[82] R. Valencia and J. Andrade-Cetto, *SLAM back-end*, 2018, vol. 119, pp. 25–52.

[83] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[84] M. J. D. Powell, "An efficient method for finding the minimum of a function of several variables without calculating derivatives," *The Computer Journal*, vol. 7, no. 2, pp. 155–162, 1964.

[85] G. Singh and J. Košecká, "Visual Loop Closing using Gist Descriptors in Manhattan World," in *Robotics and Automation (ICRA) Omnidirectional Robot Vision Workshop, IEEE International Con- ference on*, 2010, pp. 4042–4047. [Online]. Available: https://cs.gmu.edu/~kosecka/Publications/SinghKoseckaICRA2010.pdf

[86] A. C. Murillo, P. Campos, J. Kosecka, and J. J. Guerrero, "Gist vocabularies in omnidirectional images for appearance based mapping and localization," *People.Csail.Mit.Edu*, vol. RSS Omnivi, 2010. [Online]. Available: http://people.csail.mit.edu/koch/omnivis2010/papers/paper-13-OMNIVIS2010.pdf%5Cnpapers2://publication/uuid/80D59851-77F6-473D-96CF-99FDF8985A15

[87] A. Rituerto, A. C. Murillo, and J. J. Guerrero, "Semantic labeling for indoor topological mapping using a wearable catadioptric system," *Robotics and Autonomous Systems*, vol. 62, no. 5, pp. 685–695, 2014.

[88] Y. Liu and H. Zhang, "Visual loop closure detection with a compact image descriptor," in *IEEE International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1051–1056.

[89] M. Liu, D. Scaramuzza, C. Pradalier, R. Siegwart, and Q. Chen, "Scene recognition with omnidirectional vision for topological map using lightweight adaptive descriptors," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*. IEEE, 2009, pp. 116–121.

[90] M. Liu and R. Siegwart, "DP-FACT: Towards topological mapping and scene recognition with color for omnidirectional camera," in *Proceedings - IEEE International Conference on Robotics and Automation*. Ieee, 2012, pp. 3503–3508.

[91] M. J. Milford and G. F. Wyeth, "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1643–1649.

[92] E. Pepperell, P. I. Corke, and M. J. Milford, "All-environment visual place recognition with SMART," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 1612–1618.

[93] J. Wu, H. Zhang, and Y. Guan, "An efficient visual loop closure detection method in a map of 20 million key locations," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 861–866.

[94] H. Zhang, "BoRF: Loop-closure detection with scale invariant visual features," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 3125–3130.

[95] E. Johns and G. Z. Yang, "Global localization in a dense continuous topological map," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1032–1037.

[96] A. Kawewong, N. Tongprasit, S. Tangruamsub, and O. Hasegawa, "Online and incremental appearance-based slam in highly dynamic environments," *International Journal of Robotics Research*, vol. 30, no. 1, pp. 33–55, 2011.

[97] A. Kawewong, S. Tangruamsub, and O. Hasegawa, "Position-invariant robust features for long-term recognition of dynamic outdoor scenes," *IEICE Transactions on Information and Systems*, vol. E93-D, no. 9, pp. 2587–2601, 2010.

[98] N. Tongprasit, A. Kawewong, and O. Hasegawa, "PIRF-Nav 2: Speeded-up online and incremental appearance-based SLAM in an indoor environment," in *2011 IEEE Workshop on Applications of Computer Vision, WACV 2011*. IEEE, 2011, pp. 145–152.

[99] C. Valgren and A. Lilienthal, "SIFT , SURF and Seasons : Long-term Outdoor Localization Using Local Features," in *Proceedings of the European Conference on Mobile Robots ECMR*, vol. 128, no. February, 2007, pp. 1–6. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.2497{&}rep=rep1{&}type=pdf

[100] A. Ascani, E. Frontoni, A. Mancini, and P. Zingaretti, "Feature group matching for appearance-based localization," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. IEEE, 2008, pp. 3933–3938.

[101] B. Bacca, J. Salví, and X. Cufí, "Probabilistic appearance-based mapping and localization using the feature stability histogram," *Frontiers in Artificial Intelligence and Applications*, vol. 232, no. 10, pp. 51–60, 2011.

[102] B. Bacca, J. Salvi, and X. Cufí, "Long-term mapping and localization using feature stability histograms," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1539–1558, 2013.

[103] A. Romero and M. Cazorla, "Topological SLAM using omnidirectional images: Merging feature detectors and graph-matching," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6474 LNCS, no. PART 1. Springer, 2010, pp. 464–475.

[104] E. Garcia-Fidalgo and A. Ortiz, "Probabilistic appearance-based mapping and localization using visual features," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7887 LNCS. Springer, 2013, pp. 277–285.

[105] Garcia-Fidalgo, Emilio and Ortiz, Alberto, "On the use of binary feature descriptors for loop closure detection," in *19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2014*. IEEE, 2014, pp. 1–8.

[106] J. Wang and Y. Yagi, "Efficient topological localization using global and local feature matching," *International Journal of Advanced Robotic Systems*, vol. 10, no. 3, p. 153, 2013.

[107] A. Chapoulie, P. Rives, and D. Filliat, "A spherical representation for efficient visual loop closing," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 335–342.

[108] J. Wang and Y. Yagi, "Robust location recognition based on efficient feature integration," in *2012 IEEE International Conference on Robotics and Biomimetics, ROBIO 2012 - Conference Digest*. IEEE, 2012, pp. 97–101.

[109] H. Korrapati, F. Uzer, and Y. Mezouar, "Hierarchical visual mapping with omnidirectional images," in *IEEE International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 3684–3690.

[110] R. Paul and P. Newman, "FAB-MAP 3D: Topological mapping with spatial and visual appearance," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 2649–2656.

[111] E. Johns and G. Z. Yang, "Dynamic scene models for incremental, long-term, appearance-based localisation," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 2731–2736.

[112] D. Gálvez-López and J. D. Tardós, "Real-time loop detection with bags of binary words," in *IEEE International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 51–58.

[113] Gálvez-López, Dorian and Tardós, Juan D., "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[114] R. Mur-Artal and J. D. Tardós, "Fast relocalisation and loop closing in keyframe-based SLAM," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 846–853.

[115] A. Ranganathan and F. Dellaert, "Online probabilistic topological mapping," *International Journal of Robotics Research*, vol. 30, no. 6, pp. 755–771, 2011.

[116] C. Cadena, D. Gálvez-López, J. D. Tardós, and J. Neira, "Robust place recognition with stereo sequences," in *IEEE Transactions on Robotics*, vol. 28, no. 4. IEEE, 2012, pp. 871–885.

[117] T. A. Ciarfuglia, G. Costante, P. Valigi, and E. Ricci, "A discriminative approach for appearance based loop closing," in *IEEE International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 3837–3843.

[118] A. Majdik, D. Gálvez-López, G. Lazea, and J. A. Castellanos, "Adaptive appearance based loop-closing in heterogeneous environments," in *IEEE International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1256–1263.

[119] S. Achar, C. V. Jawahar, and K. Madhava Krishna, "Large scale visual localization in urban environments," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 5642–5648.

[120] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3921–3926.

[121] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat, "Real-time visual loop-closure detection," in *2008 IEEE international conference on robotics and automation*. IEEE, 2008, pp. 1842–1847.

[122] A. Angeli, D. Filliat, S. Doncieux, and J. A. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1027–1037, 2008.

[123] A. Angeli, S. Doncieux, J. A. Meyer, and D. Filliat, "Incremental vision-based topological SLAM," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. Ieee, 2008, pp. 1031–1036.

[124] 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.

[125] T. Nicosevici and R. Garcia, "Automatic visual bag-of-words for online robot navigation and mapping," *IEEE Transactions on Robotics*, vol. 28, no. 4, pp. 886–898, 2012.

[126] L. Murphy and G. Sibley, "Incremental unsupervised topological place discovery," in *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 1312–1318.

[127] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, nov 2004. [Online]. Available: https://doi.org/10.1023/B:VISI.0000029664.99615.94

[128] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary Robust invariant scalable keypoints," in *Proceedings of the IEEE International Conference on Computer Vision*, nov 2011, pp. 2548–2555.

[129] M. Calonder, V. Lepetit, M. Özuysal, T. Trzcinski, C. Strecha, and P. Fua, "BRIEF: Computing a local binary descriptor very fast," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1281–1298, jul 2012.

[130] R. Szeliski, "Robust least squares and RANSAC," in *Computer Vision: Algorithms and Applications*. Springer, 2010, ch. 6, pp. 318–319.

[131] L. Moisan and B. Stival, "A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix," *International Journal of Computer Vision*, vol. 57, no. 3, pp. 201–218, 2004.

[132] Z. Liu and R. Marlet, "Virtual line descriptor and semi-local matching method for reliable feature correspondence," in *BMVC 2012 - Electronic Proceedings of the British Machine Vision Conference 2012*. BMVA Press, 2012, pp. 16.1—-16.11.

[133] Geo-location APIs. [Online]. Available: https://cloud.google.com/maps-platform/

[134] R. Eisele. How to plot a covariance error ellipse|Computer Science & Machine Learning. [Online]. Available: https://www.xarg.org/2018/04/how-to-plot-a-covariance-error-ellipse/

[135] D. Bharat. (2014) gtsam/covarianceEllipse3D.m. [Online]. Available: https://github.com/devbharat/gtsam/blob/master/matlab/%2Bgtsam/covarianceEllipse3D.m

[136] P. S. Maybeck, *Stochastic models, estimation, and control*, 3rd ed., ser. Mathematics in Science and Engineering. Elsevier Science Inc., 1979, vol. 3.

[137] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Perception*. MIT Press, 2011, pp. 137–234. [Online]. Available: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6277376

[138] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756–770, jun 2004.

[139] H. C. Longuet-higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, no. 5828, pp. 133–135, 1981. [Online]. Available: https://www.nature.com/articles/293133a0

[140] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN Architecture for Weakly Supervised Place Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 6, 2018, pp. 1437–1451.

[141] Product | RICOH THETA S. [Online]. Available: https://theta360.com/en/about/theta/s.html

[142] MobileRobots, *Seekur Jr Operations Manual ®*.

[143] M. P. I. King. (2018) Ros_Kinetic_AriaClientDriver. [Online]. Available: https://github.com/Matthew-PI-King/Ros_Kinetic_AriaClientDriver.git

[144] A. Gartia, "Computer Vision Project." [Online]. Available: https://www.cc.gatech.edu/classes/AY2016/cs4476_fall/results/proj3/html/sdai30/index.html

[145] T. Orlita. Online Viewer for Street View™. [Online]. Available: https://istreetview.com/CAoSLEFGMVFpcE9ZVGxXYjd5MHFZRml0anY5bkk1ejVrYVg0ckVjT1hrY3gwUkFj