VISUAL NAVIGATION FOR ROBOTS IN URBAN AND INDOOR ENVIRONMENTS

A Dissertation

by

YAN LU

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Dezhen Song
Committee Members,	Ricardo Gutierrez-Osuna
	Dylan Shell
	Wei Yan
Head of Department,	Dilma Da Silva

August 2015

Major Subject: Computer Engineering

ABSTRACT

As a fundamental capability for mobile robots, navigation involves multiple tasks including localization, mapping, motion planning, and obstacle avoidance. In unknown environments, a robot has to construct a map of the environment while simultaneously keeping track of its own location within the map. This is known as simultaneous localization and mapping (SLAM). For urban and indoor environments, SLAM is especially important since GPS signals are often unavailable. Visual SLAM uses cameras as the primary sensor and is a highly attractive but challenging research topic. The major challenge lies in the robustness to lighting variation and uneven feature distribution. Another challenge is to build semantic maps composed of high-level landmarks. To meet these challenges, we investigate feature fusion approaches for visual SLAM. The basic rationale is that since urban and indoor environments contain various feature types such points and lines, in combination these features should improve the robustness, and meanwhile, high-level landmarks can be defined as or derived from these combinations.

We design a novel data structure, multilayer feature graph (MFG), to organize five types of features and their inner geometric relationships. Building upon a two view-based MFG prototype, we extend the application of MFG to image sequence-based mapping by using EKF. We model and analyze how errors are generated and propagated through the construction of a two view-based MFG. This enables us to treat each MFG as an observation in the EKF update step. We apply the MFG-EKF method to a building exterior mapping task and demonstrate its efficacy.

Two view based MFG requires sufficient baseline to be successfully constructed, which is not always feasible. Therefore, we further devise a multiple view based algorithm to construct MFG as a global map. Our proposed algorithm takes a video stream as input, initializes and iteratively updates MFG based on extracted key frames; it also refines robot localization and MFG landmarks using local bundle adjustment. We show the advantage of our method by comparing it with state-of-the-art methods on multiple indoor and outdoor datasets.

To avoid the scale ambiguity in monocular vision, we investigate the application of RGB-D for SLAM. We propose an algorithm by fusing point and line features. We extract 3D points and lines from RGB-D data, analyze their measurement uncertainties, and compute camera motion using maximum likelihood estimation. We validate our method using both uncertainty analysis and physical experiments, where it outperforms the counterparts under both constant and varying lighting conditions.

Besides visual SLAM, we also study specular object avoidance, which is a great challenge for range sensors. We propose a vision-based algorithm to detect planar mirrors. We derive geometric constraints for corresponding real-virtual features across images and employ RANSAC to develop a robust detection algorithm. Our algorithm achieves a detection accuracy of 91.0%.

To my parents, wife and son

ACKNOWLEDGEMENTS

I would like to thank everyone who has helped me throughout my PhD study at A&M.

The first one I want to thank is my advisor, Dr. Song, for his guide and support in the past five years. He brought me to robotics research, and gradually broadened my perspective over the field. He teaches me what is high quality research and how to do it. He is always willing to spend time with me discussing research plans, ideas, approaches, writing, etc. Especially, his patience in revising my paper drafts iteration by iteration greatly improved my technical writing. Dr. Song helped me not only in research, but also in many other aspects such as life and career planning. He has set up a good model for me to learn, and I will always be grateful to him.

I sincerely thank Drs. Haifeng Li, Jingtai Liu, Yiliang Xu, and Jingang Yi for being my great research collaborators and friends. My thank also goes to my friendly and supportive lab mates/alumni Chang-Young Kim, Wen Li, Ji Zhang, Matthew Hielsberg, Joseph Lee, Chieh Chou, Hsin-min Cheng, Xinran Wang, Madison Treat, and Rui Liu.

Last but not least, I would like to thank my advisory committee members Drs. Ricardo Gutierrez-Osuna, Dylan Shell and Wei Yan for their valuable time, insightful suggestions, and inspiring discussions.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

Robots are changing our life. Nowadays robots are not only operating in factories or outer space, but also participating in people's daily activities. For example, by Feb. 2014 over 10 million Roomba robotic vacuums have been sold worldwide, cleaning rooms for people. Self-driving cars, which would greatly reduce traffic accidents and congestions, are getting closer and closer to real life, thanks to the continuous efforts of big companies like Google. Four U.S. states have passed laws permitting autonomous cars, including Nevada, Florida, California, and Michigan.

For any mobile robots, navigation is a fundamental capability. Robot navigation is a combination of multiple tasks including localization, mapping, motion planning, obstacle avoidance, etc. Localization and mapping answers two basic questions for a robot: "where am I" and "what is the world like", respectively. Motion planning finds a path for a robot to move from its initial configuration to goal configuration. Obstacle avoidance keeps a robot from collision with objects.

Robot Navigation has been a popular research field in the past decades [1, 2]. For navigation, robots use sensors to perceive the world, including range sensors and passive sensors. Ultrasonic range sensors are inexpensive at the cost of low angular resolution. Laser range finders are expensive and not eye safe, despite high angular resolution. A common problem of range sensors is that they do not capture much information other than range, such as material and texture. On the other hand, cameras are not only inexpensive, but also able to capture rich texture and color information about the world. Meanwhile, cameras are becoming unprecedentedly available to everyone with the spread of mobile devices like cellphone and tablets. All these facts motivate us to study camera-based navigation, i.e., visual navigation.

In early robotics works, localization is conducted with respect to a map of known environment. As robotic sensors are inherently noisy, probabilistic approaches (e.g., Kalman filter) are widely adopted to solve the localization problem. However, when a robot explores a previously-unknown environment with no map available, the chicken and egg problem arises: localization requires a known map, but mapping relies on accurate location information. This leads to the simultaneous localization and mapping (SLAM) problem, i.e., a robot constructs a map of an unknown environment while simultaneously keeping track of its own location within the map. For navigation in urban and indoor environments, SLAM is especially important since GPS signals are often blocked/reflected by tall buildings.

SLAM has attracted extensive interest and research since early 1990's. People have applied different kinds of sensors and proposed various techniques for SLAM. At present, laser-based SLAM algorithms can produce highly accurate results and are relatively mature. On the contrary, visual SLAM (i.e. vision-based SLAM) is still facing a lot of challenges, despite its fast progress in the past decade.

The major challenge for visual SLAM lies in the robustness to lighting variation and feature distribution. The core element of visual SLAM algorithms is to estimate the relative motion between two images based on commonly observed scene/objects. To do so, one needs to find out the correspondence between images, which can be either pixel-wise or between visual features such as interest points and edges. Unfortunately, the correspondence quality can be easily challenged by lighting variations and uneven feature distributions, which directly degrades visual SLAM performance. For example, pixel-wise matching algorithms usually assume photo-consistency, which becomes invalid under lighting change. Moreover, most visual SLAM algorithms are built upon a single type of visual feature for simplicity. However, when that type of feature is unevenly distributed or even absent in the scene, the estimation becomes subject to degenerated situations or failure.

Another limitation of current visual SLAM algorithms is that the resulted map is usually composed of simple landmarks, which are constructed from low-level visual features. The most popular type of visual feature is points (also known as interest points or key points). While various point detection algorithms exist including SIFT, SURF and FAST, the generated map just consists of sparse point landmarks, without much semantic meaning. This hinders higher-level navigation tasks such as approaching an office door. It is thus desirable to build a map of higher-level landmarks such as planes or even objects. High-level landmarks can not only make a map more semantic but also enable more robust feature matching or place recognition.

To meet these challenges, we investigate feature fusion approaches in this dissertation. The basic rationale is that in urban and indoor environments there exist various feature types such points and lines, which have different properties; in combination, these features should improve the robustness of visual SLAM, and furthermore, high-level landmarks can be defined as or derived from their combinations, making maps more semantic.

We have designed a novel data structure, multilayer feature graph (MFG), which not only incorporates five types of features ranging from points to planes, but also models the geometric relationships between these feature types. We have prototyped MFG using a two view-based construction algorithm [3] to demonstrate its potential for facilitating visual navigation.

In this dissertation, we first extend the application of MFG to image sequence-based robotic mapping by using EKF. To be specific, we build a sequence of two view based MFGs from each pair of adjacent frames and analyze how errors are generated and propagated in the construction process of each MFG. We derive closed form solutions for error distributions, and the error analysis enables us to treat each MFG as an observation for the EKF at each iteration. Based on projective geometry of pinhole camera, we derive the observation models that complete the EKF framework. We have implemented the MFG-

EKF method and applied it to an automatic building exterior mapping task [4]. We have tested the algorithm using image data from outdoor man-made environments. Experiment results show that building facades are successfully constructed with mean relative error of plane depth less than 4.66%.

However, two view based MFG requires sufficient baseline to be successfully constructed. This is not always feasible in real applications. This motivates us to devise a multiple view based algorithm to construct a single MFG which serves as a global map. As a result, our proposed heterogeneous landmark-based visual navigation algorithm takes a video stream as input, initializes and iteratively updates MFG based on extracted key frames; it also refines robot localization and MFG landmarks using local bundle adjustment [5, 6]. We present pseudo code for the algorithm and analyze its complexity. We evaluate our method and compare it with state-of-the-art methods using multiple indoor and outdoor datasets. In particular, on the KITTI dataset our method reduces the translational error by 52.5% under urban sequences where rectilinear structures dominate the scene.

Monocular visual SLAM inevitably suffers scale drift due to depth ambiguity. This can be easily avoided by using an RGB-D camera, which provides pixel-wise depth measurements for color images. While most RGB-D SLAM algorithms use feature points, we investigate how to extract 3D lines from RGB-D data. More importantly, we propose an RGB-D odometry algorithm robust to lighting variation and uneven feature distribution by fusing point and line features. We extract 3D points and lines from RGB-D data, analyze their measurement uncertainties, and compute camera motion using maximum likelihood estimation. We prove that fusing points and lines produces smaller motion estimate uncertainty than using either feature type alone. In experiments our method outperforms the competing algorithms under both constant and varying lighting conditions.

Besides localization and mapping, we also investigate obstacle avoidance. In urban and indoor environments, specular objects like mirrors challenge almost every type of robot sensors including laser range finds and sonar arrays. This is because light and sound signals simply bounce off the surfaces and do not return to receivers. We propose a method for this planar mirror detection problem using two views from an on-board camera [7]. We derive geometric constraints for corresponding real-virtual features across two views. We address an issue that popular feature detectors, such as scale-invariant feature transform (SIFT), are not reflection invariant by combining a secondary reflection with an affine scale-invariant feature transform (ASIFT). We employ a RANSAC framework to develop a robust mirror detection algorithm. The algorithm is tested under both in-lab and field settings, and it achieves an overall detection accuracy rate of 91.0%.

The rest of this dissertation is organized as follows. Section 2 reviews literature related to this dissertation. In Section 3, we present the MFG-based EKF framework and its application to building exterior mapping. Section 4 presents the heterogeneous landmark-based visual SLAM algorithm using multiple view based MFG. Section 5 presents our robust RGB-D odometry algorithm fusing point and line features. In Section 6, we report the planar mirror detection algorithm for obstacle avoidance. Section 7 concludes the dissertation and discusses future work directions.

2. LITERATURE REVIEW

Robot navigation is a combination of tasks including localization, mapping, motion planning, and obstacle avoidance. The works in this dissertation mainly relate to localization, mapping and obstacle avoidance.

Robotic localization is the problem of estimating robot poses (position and orientation) with respect to a given map. Because robotic sensors are inevitably subject to measurement noise, probabilistic approaches are widely adopted for localization, such as EKF [8], probability grids [9], and particle filter [10]. When navigating in unknown environments, localization and mapping have to be conducted in the same time, which is known as the simultaneous localization and mapping (SLAM) problem. A solution to the SLAM problem is considered as the key for mobile robots to be truly autonomous. Early works [11] show that a consistent solution of SLAM requires a joint estimate of the robot poses and every landmark. Later on people further realize that SLAM is actually a convergent problem [12]. Representative approaches to the SLAM problem include EKF-based , particle filter-based (e.g. FastSLAM [13]), and information filter-based [14] methods.

Visual SLAM. Various types of sensors have been applied to SLAM, including ultrasonic sensors, Lidar, cameras. The works in this dissertation utilize cameras primarily, and thus belong to the visual SLAM category. Visual SLAM is also commonly referred to as structure from motion (SFM) or visual odometry in computer vision domain. In this dissertation, we consider these terms to be interchangeable. There exist two prevalent categories of approaches for visual SLAM, one based on sequential filtering (e.g. [15]) with its root in the traditional SLAM research, the other based on bundle adjustment (BA) [16] which is a standard optimization technique in computer vision. BA essentially estimates all camera poses and landmark parameters in a big nonlinear optimization problem, which is usually computationally expensive. Local bundle adjustment is a technique proposed to allow online processing by limiting the camera poses and landmarks to be within a window of latest frames.

The basic element of a typical visual SLAM system is pairwise motion estimation, i.e. recovering the relative 3D rigid transformation between two camera frames. This can be solved by using either pixel-to-pixel registration (i.e. dense methods) or sparse feature matches. Compared with dense methods, sparse features are less sensitive to perspective and/or illumination changes. Various interest point feature detection and/or description algorithms have been adopted in visual SLAM such as SIFT [17] and speeded-up robust feature (SURF) [18]. Other feature types are also studied for visual SLAM, such as line segments [19–22], straight lines [23], vanishing points [24], and planes [25–28]. However, points are still the most commonly used feature due to its simplicity; moreover, most visual SLAM works are built on a single feature type. As a result, existing methods are not very robust to large lighting variations and uneven feature distributions. The works in this dissertation focus on exploiting the combinational power of different feature types in order to achieve better accuracy and robustness. We propose a novel data structure to organize different types of visual features, based on which we respectively develop EKF-based and LBA-based visual SLAM algorithms and evaluate them using real-world data.

RGB-D odometry. A regular RGB camera only measures angular information of objects, with the depth information missing. This produces a scale ambiguity in monocular visual SLAM, and further results in the notorious scale drift problem [29]. The recent emergence of RGB-D cameras (e.g. Microsoft Kinect) solves the scale ambiguity/drift issue to a great extent. This is because RGB-D cameras provide pixel-wise depth data for every color image. To perform SLAM using RGB-D data, point cloud registration methods (e.g. ICP [30]) can be applied [31], but they are easily hindered by degenerated cases. Traditional visual SLAM algorithms are also applicable to RGB-D SLAM with slight modifications [32, 33], which are still sensitive to lighting variations and uneven feature distribution. Our work investigates how to fuse point and line features in RGB-D SLAM to improve the system robustness. We prove that this feature fusion approach produces smaller estimation uncertainty than using either feature alone.

Obstacle avoidance. Besides localization and mapping, obstacle avoidance is another important research problem in navigation. To cope with this problem, range data from ultrasonic [34] or laser [35] range finders are commonly used to detect obstacles. In general range sensors are able to reliably detect normal obstacles like chairs and boxes, except for specular surfaces (e.g. mirrors). This is because sound or light does not return to the receivers after specular reflections most of the time. Our work detects planar mirrors using a vision-based algorithm. Our algorithm not only detects the existence of planar mirrors, but also estimates their poses relative to the camera.

3. AUTOMATIC BUILDING EXTERIOR MAPPING USING MULTILAYER FEATURE GRAPH AND EXTENDED KALMAN FILTER*

We are developing visual algorithms to assist building exterior survey using a mobile robot, which can greatly assist building energy retrofitting. The task requires a robot to map building facades with its on-board camera while the robot travels. Existing navigation methods often utilize low level landmarks, such as feature points and point clouds, and cannot directly provide information for build facades, which can be viewed as high level landmarks. Actually, the high level landmarks, such as primary planes and salient lines, have distinctive advantages over low level features. Bearing clear geometric meaning, high level landmarks are less sensitive to different lighting conditions and varying shadows where low level features are often challenged. High level landmarks are ubiquitous in modern urban areas where rectilinear objects dominate camera field of view. Humans are used to navigating in unknown environments by effectively using high level landmarks as reference. However, robots still have difficulty to utilize advantages of high level landmarks due to challenges in feature recognition and correspondence.

In 2012, we [3] proposed a two-view MFG (multilayer feature graph) as a scene understanding and knowledge representation method for robot navigation. An MFG is constructed from overlapping and dislocated two views and contains five different features ranging from raw key points to planes and vanishing points in 3D. Here we build our high level landmark-based maps (see Fig. 3.1) by employing MFG as observations in an EKF framework. We analyze how errors are generated and propagated in the MFG construction process, which characterizes observation errors in the EKF. We derive closed form

*Reprinted with permission from "Automatic building exterior mapping using multilayer feature graphs" by Y. Lu, D. Song, Y. Xu, A. G. A. Perera, and S. Oh, 2013. *IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 162-167, Copyright © 2013 IEEE.

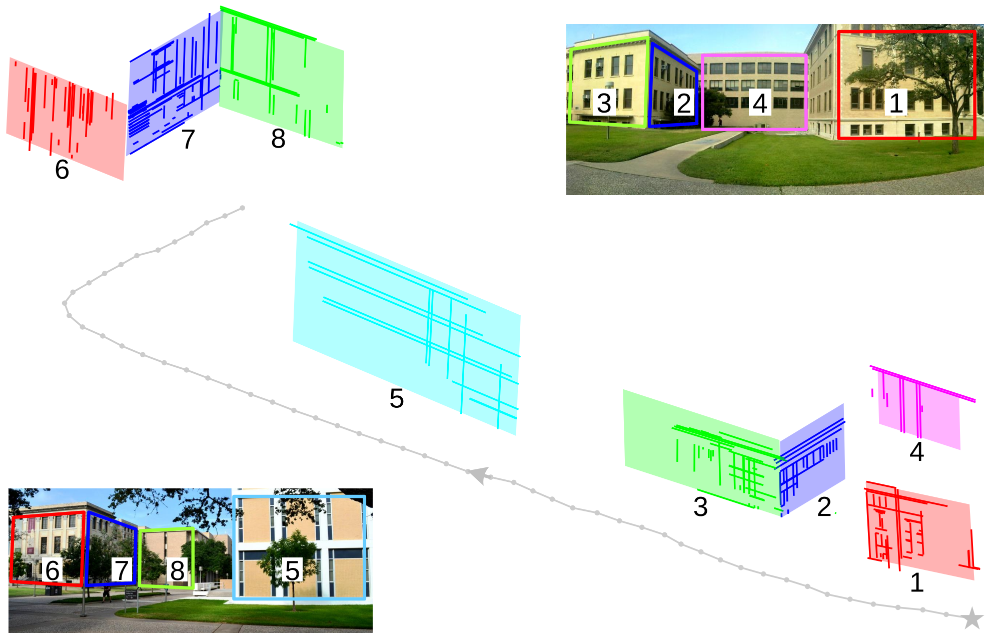


Figure 3.1: A sample output of high level landmarks and the robot trajectory after the mapping process in 3D view. The system is able to recognize primary planes from building facades and their corresponding co-planar lines as high level landmarks. The numbered corresponding building facades are also color coded in the top right and bottom left images.

solutions for error distributions. Based on projective geometry, we derive the observation models to complete the EKF framework. We have implemented and tested our MFG-EKF method at three different sites. Experimental results show that high level landmarks are successfully constructed in modern urban environments with mean relative plane depth errors less than 4.66%.

3.1 Related Work

Robotic mapping with high level landmarks relates to a broad body of research in SLAM and visual odometry including different sensor configurations and different landmark selections.

Depending on costs, payload limitation, and navigation environments, the most common sensors for robot navigation include sonar arrays [1], laser range finders [36, 37], depth cameras [32], regular cameras [38, 39] , or their combinations [40, 41]. Mapping tasks are often conducted under the SLAM framework [42]. As a partially observable Markovian decision process, SLAM infers system states based on the sensory input using different filters and loop closure techniques. The system states usually include both landmarks and robot states whereas landmarks are the representation of the physical world. For example, landmarks are point clouds if a laser ranger finder or a depth camera is the primary sensor. In vision-based SLAM, SIFT feature points or its variants [15] and line features [19, 23] are often employed as landmarks.

Our work belongs to the vision-based SLAM category where one or more cameras are the primary navigation sensor. Recently, many researchers realize landmark selections can make a difference in SLAM and visual odometry performance. Lower level landmarks [15], such as Harris corners and SIFT points, are relatively easy to use due to their geometric simplicity, which share many geometric properties with traditional point clouds used for laser range finders. However, point features are merely mathematical singularities in color, texture, and geometric space. They can be easily influenced by lighting and shadow conditions. Realizing the limitation, recent efforts focus on developing high level landmarks such as lines/edges/line segments [20, 22]. Zhang et al. [43] use vertical lines and floor lines in a monocular SLAM and build a 3D line-based map in an indoor corridor environment.

More recent sophisticated methods combine multiple features such as points, lines, and planes. Gee et al. [25] incorporate 3D planes and lines into visual SLAM framework. Martinez et al. [26] propose a monocular SLAM algorithm that unifies the estimation of point and planar features. These works have demonstrated the robustness of high level landmarks and inspired this work. Observe that the existing works only treat different landmarks as isolated geometric objects, without exploring the inner relationship between them. The treatment simplifies the SLAM problem formulation but cannot fully utilize the power of high level landmarks.

3.2 Problem Definition

Consider a robot equipped with a single camera navigates in an unknown environment. The robot attempts to estimate high level landmarks such as building facades or salient edges from input image frames. The basic assumptions are,

a.1 The robot operates in a largely static modern urban environment with rectilinear structures, which is the prerequisite for MFG.

a.2 The onboard camera is pre-calibrated and has a known intrinsic matrix K.

a.3 The initial step of robot movement is known for reference. Otherwise, estimates would be up to scale. The assumption can be relaxed if a stereo camera is available.

In our approach, adjacent raw image frame pairs are first employed to construct MFG [3] sequence. Let I_k be the k-th ($k \in \mathbb{N}$) image frame and $\mathcal{M}_k$ $(k \geq 1)$ be the MFG constructed from frames I_k and I_{k-1}. Fig. 3.2 illustrates that MFG is a data structure composed of five layers of feature nodes: *key points*, *line segments*, *ideal lines*, *primary planes* and *vanishing points*; edges between nodes of different layers represent geometric relationships including *adjacency*, *collinearity*, *coplanarity*, and *parallelism*.

The resulting MFG sequence, $\{\mathcal{M}_k, k \geq 1\}$, is considered as the input to the problem. Denote $\{C_k\}$ the camera coordinate system (CCS) associated with I_k. MFGs assist us in identifying high level landmarks such as 3D planes and their associated coplanar lines in physical space. However, the planes and lines from $\mathcal{M}_k$ are represented w.r.t. $\{C_k\}$, which cannot be directly used as global landmarks. Define the world coordinate system (WCS), $\{W\}$, to coincide with $\{C_0\}$. Now we are ready to define our problem.

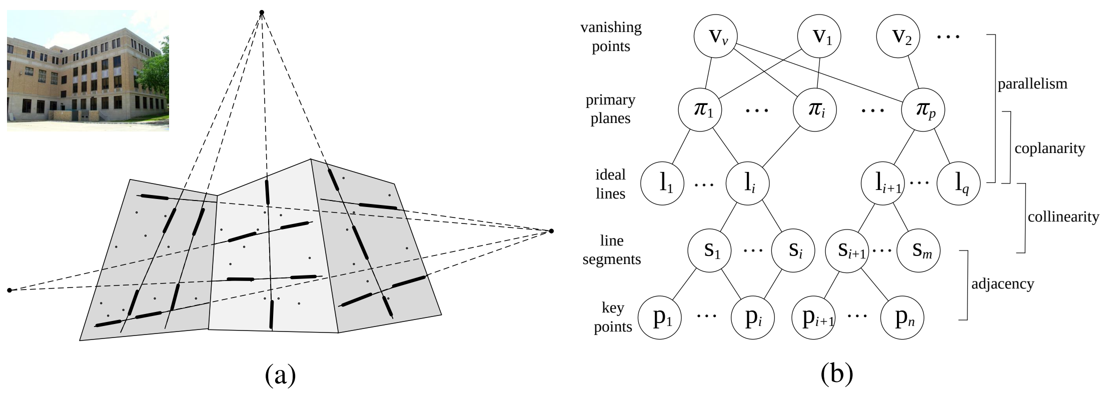


Figure 3.2: (a) A illustration of the multilayer feature graph. (b) MFG structure.

Definition 1. *Given MFG sequence $\{\mathcal{M}_k : k \geq 1, k \in \mathbb{N}\}$, map high level landmarks including 3D planes and coplanar lines in $\{W\}$, and assess the uncertainty of the mapping process by deriving error covariance matrices for each landmark.*

To solve the landmark mapping problem, we employ an EKF-based approach. In this approach, MFG $\mathcal{M}_k$ can be considered as a generalized observation at time k. Therefore, we need to understand how errors are distributed in the construction process of $\mathcal{M}_k$, which can serve as the observation error in the EKF. With the observation error derived, the landmark errors can be estimated by combining process errors using the EKF. Therefore, the problem is solved in two steps with the first step being the uncertainty analysis of MFG.

3.3 Observation Error: Uncertainty in MFG

Our previous work [3], has shown how to construct MFG using a feature fusion method. However, the uncertainty of each feature layer is yet to be analyzed. Here we detail the uncertainty for each layer of MFG in a bottom-up manner.

3.3.1 Error Modeling of Raw Features

The MFG construction algorithm takes two images I and I' as input and outputs a feature graph of five layers, as illustrated in Fig. 3.2(b). In MFGs, key points and line segments are raw features directly detected from images using SIFT [17] and LSD [44], while ideal lines, primary planes and vanishing points represent high level features constructed from raw features. MFGs also include feature correspondences between two views.

Note that I and I' actually represent I_k and I_{k-1} in the continuous image sequence, respectively. Here we drop k and $k-1$ from notations for simplicity. Furthermore, we attach a superscript $'$ to variables associated with I'. As a convention, we use a $\sim$ on top of a homogeneous vector to denote its inhomogeneous counterpart throughout this section.

For each key point $\mathbf{p}_i$ in I, we model its measurement error as an independent and identically distributed (i.i.d.) zero-mean isotropic Gaussian noise with variance σ^2 in each axis

$$\mathrm{Cov}(\tilde{\mathbf{p}}_i) = \sigma^2 \mathrm{I}_2, \ \ \forall i \tag{3.1}$$

where I_2 is a 2×2 identity matrix.

For each line segment $\mathbf{s}_i$ in I, denote its two endpoints by $\mathbf{e}_{i1}$ and $\mathbf{e}_{i2}$. Define $\mathbf{u}_{i\|}$ and $\mathbf{u}_{i\perp}$ to be two unit vectors parallel and perpendicular to the line segment, respectively (see Fig. 3.3). We model the error of $\mathbf{e}_{i1}$ (the same for $\mathbf{e}_{i2}$) as an independent 2D Gaussian with its covariance matrix to be diagonal in the coordinate system defined by $\mathbf{u}_{i\|}$ and $\mathbf{u}_{i\perp}$ as below

$$\Sigma_{i\|\perp} = \begin{bmatrix} \sigma_{i\|}^2 & 0 \\ 0 & \sigma_{i\perp}^2 \end{bmatrix}, \tag{3.2}$$

where $\sigma_{i\perp}$ and $\sigma_{i\|}$ are the standard deviations of $\mathbf{e}_{i1}$ in directions of $\mathbf{u}_{i\perp}$ and $\mathbf{u}_{i\|}$, respectively. $\sigma_{i\perp}$ is usually much smaller than $\sigma_{i\|}$. We have observed that $\sigma_{i\perp}$ usually is inversely correlated to the line segment length. Furthermore, $\sigma_{i\perp}$ also has a lower bound of σ_p due

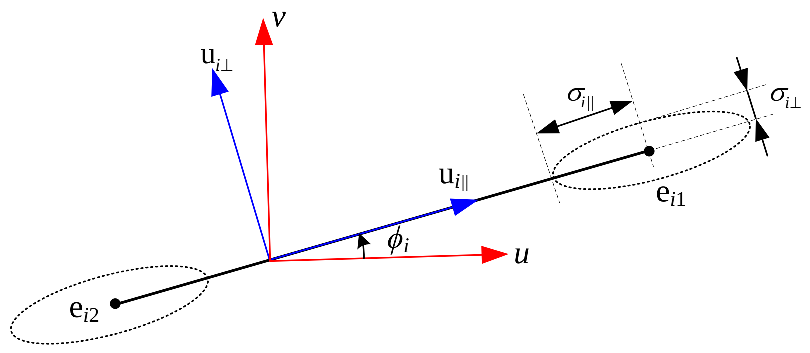


Figure 3.3: Uncertainty of line segment endpoints.

to pixelization error. Thus, we model the endpoint error as follows,

$$\sigma_{i\|} = \sigma_{\|}, \qquad \sigma_{i\perp} = \frac{\sigma_{i\|}}{\|\mathbf{s}_i\|} + \sigma_p, \quad \forall i, \tag{3.3}$$

where $\sigma_{\|}$ and σ_p are constant and independent of i, and $\|\mathbf{s}_i\|$ denotes the length of $\mathbf{s}_i$. The parameters for the models can be determined using Monte Carlo simulation. Projecting (3.2) back to the image coordinate system (ICS), we have

$$\text{Cov}(\tilde{\mathbf{e}}_{i1}) = R(\phi_i)\Sigma_{i\|\perp}R(\phi_i)^{\mathsf{T}} \tag{3.4}$$

where ϕ_i is the angle between $\mathbf{u}_{i\|}$ (Fig. 3.3) and u-axis, and $R(\phi_i) = \begin{bmatrix} \cos\phi_i & -\sin\phi_i \\ \sin\phi_i & \cos\phi_i \end{bmatrix}$. Note that the error model in (3.2-3.4) for line segments may differ for different line detectors. However, the rest of our analysis still applies.

With error distributions of raw features obtained, we are ready to analyze high level features such as ideal lines and primary planes.

3.3.2 Error Analysis of Ideal Lines

In the MFG construction process, an ideal line $\mathbf{l}_i$ is obtained by fitting a straight line through endpoints of a set of m_i collinear line segments $\{\mathbf{s}_j : 1 \leq j \leq m_i\}$. The i-th ideal line in I can be parameterized in terms of angle θ_i and intercept ρ_i with the following homogeneous format in ICS,

$$\mathbf{l}_i = [\cos\theta_i, \sin\theta_i, \rho_i]^\mathsf{T} \tag{3.5}$$

such that $u\cos\theta_i + v\sin\theta_i + \rho_i = 0$ holds for any point (u, v) on $\mathbf{l}_i$. Since the fitting process employs maximum likelihood estimation (MLE) to obtain optimal solution $[\theta_i^*, \rho_i^*]^\mathsf{T}$, we have the following lemma.

Lemma 1. *Given collinear line segments set $\{\mathbf{s}_j\}$ with their endpoint covariance matrices in (3.4), if MLE is employed to estimate $[\theta_i^*, \rho_i^*]^\mathsf{T}$, the resulting $\mathbf{l}_i$ can be approximated by a Gaussian with a mean vector of $[\cos\theta_i^*, \sin\theta_i^*, \rho_i^*]^\mathsf{T}$ and a covariance matrix of,*

$$\mathit{Cov}(\mathbf{l}_i) = J\ \mathit{Cov}(\theta_i^*, \rho_i^*)\ J^\mathsf{T}, \tag{3.6}$$

where $J = \begin{bmatrix} -\sin\theta_i^* & \cos\theta_i^* & 0 \\ 0 & 0 & 1 \end{bmatrix}^\mathsf{T}$ *and $\mathit{Cov}(\theta_i^*, \rho_i^*)$ is given by (3.9).*

Proof. Let us formulate this MLE problem first. This MLE problem simultaneously seeks for line parameters $[\theta_i, \rho_i]^\mathsf{T}$ and the corrected line segment endpoints, denoted by $\underline{\mathbf{e}}_{j\tau}$, $\tau = 1, 2$. To ensure $\underline{\mathbf{e}}_{j\tau}^\mathsf{T}\mathbf{l}_i = 0$, we use the parametrization

$$\tilde{\underline{\mathbf{e}}}_{j\tau}(t_{j\tau}) = \begin{bmatrix} -\rho_i\cos\theta_i \\ -\rho_i\sin\theta_i \end{bmatrix} + t_{j\tau} \begin{bmatrix} \sin\theta_i \\ -\cos\theta_i \end{bmatrix} \tag{3.7}$$

where $t_{j\tau} \in \mathbb{R}$ is the only free parameter for $\underline{\tilde{\mathbf{e}}}_{j\tau}$ that we need to estimate.

Define a parameter vector that is to be estimated as $\Theta_L = [\Theta_{L1}^\mathsf{T}, \Theta_{L2}^\mathsf{T}]^\mathsf{T}$ with $\Theta_{L1} = [\theta_i, \rho_i]^\mathsf{T}$ and $\Theta_{L2} = [t_{11}, t_{12}, \cdots, t_{m_i 1}, t_{m_i 2}]^\mathsf{T}$. Define a measurement vector that incorporates measurement data as $\Omega_L = [\tilde{\mathbf{e}}_{11}^\mathsf{T}, \tilde{\mathbf{e}}_{12}^\mathsf{T}, \cdots, \tilde{\mathbf{e}}_{m_i 1}^\mathsf{T}, \tilde{\mathbf{e}}_{m_i 2}^\mathsf{T}]^\mathsf{T}$. Define a measurement function $f_L(\cdot)$ that maps from parameter space to measurement space, which is straightforward to be obtained from (3.7).

The MLE problem now becomes

$$\arg\min_{\Theta_L} (\Omega_L - f_L(\Theta_L))^\mathsf{T} \Sigma_{\Omega_L}^{-1} (\Omega_L - f(\Theta_L)), \tag{3.8}$$

where

$$\Sigma_{\Omega_L} = \begin{bmatrix} \mathrm{Cov}(\tilde{\mathbf{e}}_{11}) & & 0 \\ & \ddots & \\ 0 & & \mathrm{Cov}(\tilde{\mathbf{e}}_{m_i 2}) \end{bmatrix}_{2m_i \times 2m_i}.$$

The above optimization problem can be solved by the Levenberg-Marquardt algorithm (LMA). Denote the optimal estimate by $[\Theta_{L1}^{*\mathsf{T}}, \Theta_{L2}^{*\mathsf{T}}]^\mathsf{T}$. The covariance of $[\theta_i^*, \rho_i^*]^\mathsf{T}$ can be computed following the method in Chapter 5 of [45] as below

$$\mathrm{Cov}(\Theta_{L1}^*) = (U - WV^{-1}W^\mathsf{T})^\dagger, \tag{3.9}$$

where $U = A^\mathsf{T}\Sigma_{\Omega_L}^{-1}A$, $V = B^\mathsf{T}\Sigma_{\Omega_L}^{-1}B$, $W = A^\mathsf{T}\Sigma_{\Omega_L}^{-1}B$,

$$A = \partial f_L / \partial \Theta_{L1}^*, B = \partial f_L / \partial \Theta_{L2}^*,$$

and $()^\dagger$ indicates the pseudo-inverse operation.

Due to the fact that endpoints of line segments are conformal to independent Gaussian distributions, the property of MLE ensures that $[\theta_i^*, \rho_i^*]^\mathsf{T}$ is asymptotically normal (Thm.

3.1 in pp. 2143 of [46]). Hence we can use a 2D Gaussian to approximate its distribution. Furthermore, through error forward propagation approximation, we arrive at (3.6). Lem. 1 is proved. □

3.3.3 Error Analysis of Primary Planes

In an MFG based on two views, a primary plane $\boldsymbol{\pi}_i$ is a 3D plane represented by a 4D homogeneous vector in the CCS associated with I. Furthermore, if $\boldsymbol{\pi}_i$ does not pass through the camera center (which is often the case in practice), we can have

$$\boldsymbol{\pi}_i = [\tilde{\boldsymbol{\pi}}_i^{\mathsf{T}}, 1]^{\mathsf{T}}, \tag{3.10}$$

where $\tilde{\boldsymbol{\pi}}_i$ is a 3×1 vector for the inhomogeneous representation of $\boldsymbol{\pi}_i$.

Based on the coplanarity relationship in an MFG, each plane $\boldsymbol{\pi}_i$ can be associated with p_i coplanar point correspondences $\{\mathbf{p}_{ij} \leftrightarrow \mathbf{p}'_{ij} : j = 1, \cdots, p_i\}$, and q_i coplanar line correspondences $\{\mathbf{l}_{i\kappa} \leftrightarrow \mathbf{l}'_{i\kappa} : \kappa = 1, \cdots, q_i\}$. These feature correspondences satisfy a homography induced by $\boldsymbol{\pi}_i$

$$\mathbf{p}'_{ij} = \mathrm{H}_i \mathbf{p}_{ij} , \quad \mathbf{l}_{i\kappa} = \mathrm{H}_i^{\mathsf{T}} \mathbf{l}'_{i\kappa}, \tag{3.11}$$

where

$$\mathrm{H}_i = K(\mathrm{R} - \mathbf{t}\tilde{\boldsymbol{\pi}}_i^{\mathsf{T}})K^{-1}, \tag{3.12}$$

and R and $\mathbf{t}$ are the rotation matrix and translation vector between two views, respectively.

Eqs. (3.11 and 3.12) suggest a method of computing $\tilde{\boldsymbol{\pi}}_i$ based on R and $\mathbf{t}$. However, if R and $\mathbf{t}$ are simply derived from epipolar geometry without considering the planar structure information, the solution is not optimal, and neither is $\tilde{\boldsymbol{\pi}}_i$. Inspired by the method from [47], we estimate all $\tilde{\boldsymbol{\pi}}_i$'s, R and $\mathbf{t}$ simultaneously by employing all geometric features (i.e., key points and ideal lines) and constraints (i.e., epipolar constraint and homography) under an MLE framework. Define $\Theta_{P1} = [\tilde{\boldsymbol{\pi}}_1^\mathsf{T}, \cdots, \tilde{\boldsymbol{\pi}}_i^\mathsf{T}, \cdots]^\mathsf{T}$. Supposing Θ_{P1}^* is the MLE output of Θ_{P1}, we have the following lemma.

Lemma 2. *Given that key point errors follow i.i.d. isotropic Gaussian distributions with covariance matrices in (3.1) and line segment endpoints follow independent Gaussian distributions with covariance matrices in (3.4), if MLE is employed to estimate all $\tilde{\boldsymbol{\pi}}_i$'s for primary planes, then the distribution of each $\tilde{\boldsymbol{\pi}}_i$ can be approximated by a Gaussian distribution with the following mean and covariance matrix,*

$$\tilde{\boldsymbol{\pi}}_i^* = T_i\, \Theta_{P1}^* \tag{3.13}$$

$$Cov(\tilde{\boldsymbol{\pi}}_i^*) = T_i\, Cov(\Theta_{P1}^*)\, T_i^\mathsf{T} \tag{3.14}$$

where $T_i = [\mathbf{0}_{3,3i-3} : \mathrm{I}_3 : \mathbf{0}_{3,3(p-i)+6}]$, $Cov(\Theta_{P1}^*)$ is derived in a way similar to that in (3.9), $\mathbf{0}_{a,b}$ is an $a \times b$ zero matrix, and I_3 is a 3×3 identity matrix.

Proof. Similar to the proof of Lem. 1, let us formulate the MLE problem first. This MLE problem estimates planes ($\tilde{\boldsymbol{\pi}}_i$'s), relative pose ($\mathrm{R}$ and $\mathbf{t}$) as well as corrected feature correspondences simultaneously. Define a measurement vector

$$\Omega_P = [\cdots, \tilde{\mathbf{p}}_{ij}^\mathsf{T}, \tilde{\mathbf{p}}_{ij}'^\mathsf{T}, \cdots, \theta_{i\kappa}, \rho_{i\kappa}, \theta_{i\kappa}', \rho_{i\kappa}', \cdots, \tilde{\mathbf{p}}_r^\mathsf{T}, \tilde{\mathbf{p}}_r'^\mathsf{T}, \cdots]^\mathsf{T}$$

where $[\theta_i, \rho_i]$ are the parameters of $\mathbf{l}_i$, and $\{\tilde{\mathbf{p}}_r \leftrightarrow \tilde{\mathbf{p}}_r' : r \geq 1\}$ denote the point correspondences that are not associated with any plane,
and a parameter vector $\Theta_P = [\Theta_{P1}^\mathsf{T}, \Theta_{P2}^\mathsf{T}]^\mathsf{T}$ with

$$\Theta_{P2} = [\alpha, \beta, \gamma, \mathbf{t}^\mathsf{T}, \cdots, \tilde{\mathbf{p}}_{ij}^\mathsf{T}, \cdots, \theta_{i\kappa}', \rho_{i\kappa}', \cdots, \tilde{\mathbf{P}}_r^\mathsf{T}, \cdots]^\mathsf{T},$$

where α, β and γ are the Euler angles of R, $\tilde{\mathbf{P}}_r$ is the 3D point corresponding to $\mathbf{p}_r$.

Now, we define the measurement function for each type of feature correspondence accordingly as below.

- *for point correspondences on plane $\boldsymbol{\pi}_i$:*

$$\tilde{\mathbf{p}}_{ij} = \tilde{\mathbf{p}}_{ij}, \qquad \tilde{\mathbf{p}}'_{ij} = \frac{(\mathrm{H}_i \mathbf{p}_{ij})_{1:2}}{(\mathrm{H}_i \mathbf{p}_{ij})_3},$$

  where H_i is defined in (3.12).

- *for line matches on plane $\boldsymbol{\pi}_i$:* First, let us define a function

$$g(\mathbf{l}) = [\theta, \rho]^\mathsf{T} \tag{3.15}$$

  that maps a line vector $\mathbf{l} = [\cos\theta, \sin\theta, \rho]^\mathsf{T}$ to its parameters. Now the measurement function for line correspondences is

$$\begin{bmatrix} \theta'_{i\kappa} \\ \rho'_{i\kappa} \end{bmatrix} = \begin{bmatrix} \theta'_{i\kappa} \\ \rho'_{i\kappa} \end{bmatrix}, \quad \begin{bmatrix} \theta_{i\kappa} \\ \rho_{i\kappa} \end{bmatrix} = g\Big(\frac{\mathrm{H}_i^\mathsf{T} \mathbf{l}'_{i\kappa}}{\|(\mathrm{H}_i^\mathsf{T} \mathbf{l}'_{i\kappa})_{1:2}\|} \Big).$$

- *for point correspondences not on any plane:*

$$\tilde{\mathbf{p}}_r = \frac{(\mathsf{P}\mathbf{P}_r)_{1:2}}{(\mathsf{P}\mathbf{P}_r)_3}, \quad \tilde{\mathbf{p}}'_r = \frac{(\mathsf{P}'\mathbf{P}_r)_{1:2}}{(\mathsf{P}'\mathbf{P}_r)_3},$$

  where $\mathsf{P} = K[\,\mathrm{I}_3 \mid 0\,]$, $\mathsf{P}' = K[\,\mathrm{R} \mid \mathbf{t}\,]$.

The MLE problem is formulated the same as (3.8) except that Σ_{Ω_P} is obtained from (3.1) and (3.9). Let the optimal estimate be $\Theta_P^* = [\Theta_{P1}^{*\mathsf{T}}, \Theta_{P2}^{*\mathsf{T}}]^\mathsf{T}$. $\mathrm{Cov}(\Theta_{P1}^*)$ can be computed in the similar way as in (3.9). Since the rest of the proof is similar to that in the proof of Lem. 1, we skip the details here. Hence Lem. 2 is proved. □

3.4 EKF based Mapping with MFG

Two-view based MFGs only provide local information of high level features. In order to build a global map in $\{W\}$, EKF is employed to estimate the posterior of landmarks as well as a robot trajectory.

3.4.1 System State Representation

In the EKF framework, we maintain and keep updating a system state $\mathbf{y}_k$, which is composed of the robot state $\mathbf{x}_k$ and 3D landmarks.

The robot state is defined as

$$\mathbf{x}_k = [\mathbf{r}_k^\mathsf{T}, \mathbf{q}_k^\mathsf{T}, \boldsymbol{\nu}_k^\mathsf{T}, \boldsymbol{\omega}_k^\mathsf{T}]^\mathsf{T}, \tag{3.16}$$

where $\mathbf{r}_k$ is a 3D location in $\{W\}$, $\mathbf{q}_k$ is an orientation quaternion w.r.t. $\{W\}$, $\boldsymbol{\nu}_k$ is a velocity vector in $\{W\}$, and $\boldsymbol{\omega}_k$ is an angular velocity vector in $\{C_k\}$.

In $\mathbf{y}_k$, we use $\tilde{\boldsymbol{\pi}}_i^W$ to represent the i-th 3D plane landmark in $\{W\}$. To represent a 3D line, general methods like Plücker coordinates would need as many as 6 parameters. However, a 3D vector is sufficient in this work since our method is only interested in coplanar lines associated with landmark planes. Supposing a landmark line resides on plane $\tilde{\boldsymbol{\pi}}_i^W$, then there exists an one-to-one mapping between this line and its projection on image plane I_0, which is actually a 2D homography induced by $\tilde{\boldsymbol{\pi}}_i^W$. Let us denote $\mathbf{l}_j^k$ the projection of the j-th landmark line on I_k. Then, we can use $\mathbf{l}_j^0$ to fully represent the j-th landmark line in $\mathbf{y}_k$ since we already have $\tilde{\boldsymbol{\pi}}_i^W$ in $\mathbf{y}_k$.

As a result, the complete system state can be written as

$$\mathbf{y}_k = \left[\mathbf{x}_k^\mathsf{T}, \cdots, (\tilde{\boldsymbol{\pi}}_i^W)^\mathsf{T}, \cdots, (\mathbf{l}_j^0)^\mathsf{T}, \cdots\right]^\mathsf{T}. \tag{3.17}$$

3.4.2 EKF Formulation

In an EKF framework, a process model and an observation model need to be specified for the prediction and update steps, respectively.

3.4.2.1 Process Modeling

We follow the conventional assumptions of "constant velocity, constant angular velocity" in [15] for camera motion to formulate the process model as follows,

$$\mathbf{x}_{k+1} = \begin{bmatrix} \mathbf{r}_{k+1} \\ \mathbf{q}_{k+1} \\ \boldsymbol{\nu}_{k+1} \\ \boldsymbol{\omega}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_k + \boldsymbol{\nu}_k \Delta t \\ \mathbf{q}_k \times \mathbf{q}(\boldsymbol{\omega}_k \Delta t) \\ \boldsymbol{\nu}_k \\ \boldsymbol{\omega}_k \end{bmatrix},$$

where $\mathbf{q}(\boldsymbol{\omega}_k \Delta t)$ denotes the quaternion defined by the angle-axis rotation vector $\boldsymbol{\omega}_k \Delta t$, and Δt is the time interval between two steps. Note this is just a partial model for the system state in (3.17) while the rest of states of $\mathbf{y}_k$ are landmark states. Since landmarks are assumed to be static, their corresponding states remain unchanged in the prediction step.

3.4.2.2 Observation Modeling

An observation function maps the system state to landmark observations. For a plane landmark $\tilde{\boldsymbol{\pi}}_i^W$, the observation produced by $\mathcal{M}_k$ is its representation $\tilde{\boldsymbol{\pi}}_i^k$ in $\{C_k\}$. Define a matrix ${}^W_k T$ that transforms a 3D point (of homogeneous format) from $\{C_k\}$ to $\{W\}$ as

$${}^W_k T = \begin{bmatrix} R(\mathbf{q}_k) & \mathbf{r}_k \\ 0 & 1 \end{bmatrix}_{4\times 4}, \tag{3.18}$$

where $R(\mathbf{q}_k)$ represents the rotation matrix defined by $\mathbf{q}_k$. For a primary plane landmark $\boldsymbol{\pi}_i^k$, we know that $\boldsymbol{\pi}_i^k = {}_k^W T^\mathsf{T} \boldsymbol{\pi}_i^W$. This implies the observation to be

$$\tilde{\boldsymbol{\pi}}_i^k = \frac{\left({}_k^W T^\mathsf{T} \boldsymbol{\pi}_i^W\right)_{1:3}}{\left({}_k^W T^\mathsf{T} \boldsymbol{\pi}_i^W\right)_4} \tag{3.19}$$

where $(V)_a$ denotes the a-th element of vector V, and $(V)_{a:b}$ denotes the sub vector of V indexed from a to b.

For a line landmark $\mathbf{l}_j^0$, its observation from $\mathcal{M}_k$ is $\mathbf{l}_j^k$. Supposing $\mathbf{l}_j^0$ lies on plane $\tilde{\boldsymbol{\pi}}_i^W$, $\mathbf{l}_j^k$ can be computed from $\mathbf{l}_j^0$ via a homography [45]

$$\mathbf{l}_j^k = \frac{(\mathrm{H}_i^k)^{-\mathsf{T}} \mathbf{l}_j^0}{\left\| \left((\mathrm{H}_i^k)^{-\mathsf{T}} \mathbf{l}_j^0\right)_{1:2} \right\|} \tag{3.20}$$

where $\mathrm{H}_i^k = K \left[R^{-1}(\mathbf{q}_k) + R^{-1}(\mathbf{q}_k)\mathbf{r}_k \left(\tilde{\boldsymbol{\pi}}_i^W\right)^\mathsf{T} \right] K^{-1}$ and $\|\cdot\|$ is L^2 norm.

Eqs. (3.19 and 3.20) fully determine the observation function. It is worth noting that the covariance matrices of observation noise have been presented in Lems. 1 and 2. EKF also provides covariance of landmarks in its covariance update and prediction steps. Since this is a standard EKF procedure, we skip details here.

3.4.3 Landmark Initialization and Management

Since two views are needed to establish an MFG, the system should start at $k = 1$ when $\mathcal{M}_1$ is constructed and landmark planes and lines are added to $\mathbf{y}_1$. Starting from $\mathbf{y}_1$, the system enters the prediction and update loops. As the robot travels farther, new landmarks may be discovered and added to the system state. Because the MFG output is the landmark representation in the current CCS, it needs to be transformed to $\{W\}$ before augmenting the system state. This coordinate transformation is represented by ${}_k^W T$ or the inverse of H_i^k as shown in (3.18-3.20).

3.5 Experiments

We have implemented the proposed method using Matlab on a Desktop PC. The camera used in the experiment is a pre-calibrated Nikon D5100 camera equipped with a 18 mm lens, which ensures a horizontal field of view of $60°$. Images are down-sampled to a resolution of 800×530 pixels. We have conducted two experiments: uncertainty test and field mapping test.

3.5.1 Uncertainty Test

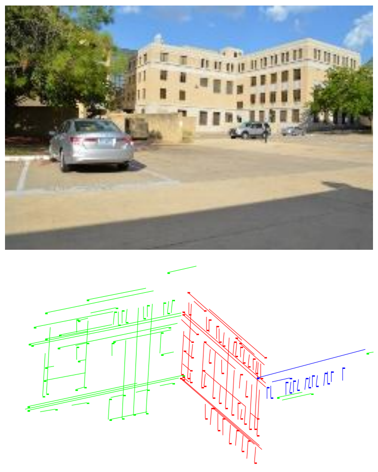
The purpose of this experiment is to verify how the estimation uncertainty of landmarks changes as more images entering our system. A sequence of 14 images has been taken while the camera was carried by a person walking towards a building. The starting point is around 40 meters away from the building. Images have been captured every $1 \sim 2$ meters approximately with the first step length known to be 1.5 meters.

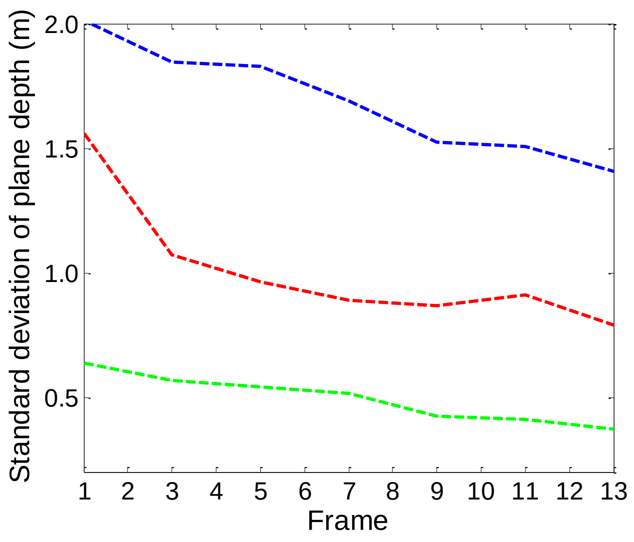
The upper image in Fig. 3.4(a) shows a sample of the image sequence and the lower line drawing in Fig. 3.4(a) shows the 3D landmarks constructed from the image sequence. Each plane and its coplanar line segments are coded in the same color. Fig. 3.4(b) demonstrates that the standard deviation of the depth of each landmark plane (using the same color coding as that in the lower line drawing in Fig. 3.4(a) decreases as the frame number increases.

3.5.2 Field Mapping Test

Table 3.1: FIELD MAPPING TEST RESULTS.

site	dist. (m)	#frame	#plane	#line	ε_d (%)		ε_a (°)		ε_L (pixel)	
					mean	std. dev.	mean	std. dev.	mean	std. dev.
1	216	55	8	197	3.48	2.91	1.77	2.34	0.52	0.26
2	156	40	6	231	4.66	3.27	0.83	3.75	0.39	0.22
3	180	36	7	225	4.09	3.96	1.65	3.09	0.47	0.31


(a)



(b)

Figure 3.4: (a) A sample view (upper) and constructed 3D landmarks (lower). (b) Standard deviations of plane depth vs #frames.

In the second experiment, we have tested our method in the field including three sites on Texas A&M University campus as shown in Fig. 3.5. At each site, the camera follows a pre-defined route. Images are taken every 4 meters approximately while the first step has been known to be exact 4.0 meters as a reference. The distance traveled and the numbers of frames collected at each site are shown in columns 2 and 3 of Tab. 3.1. As shown in the table, our method was able to successfully recognize high level landmarks including

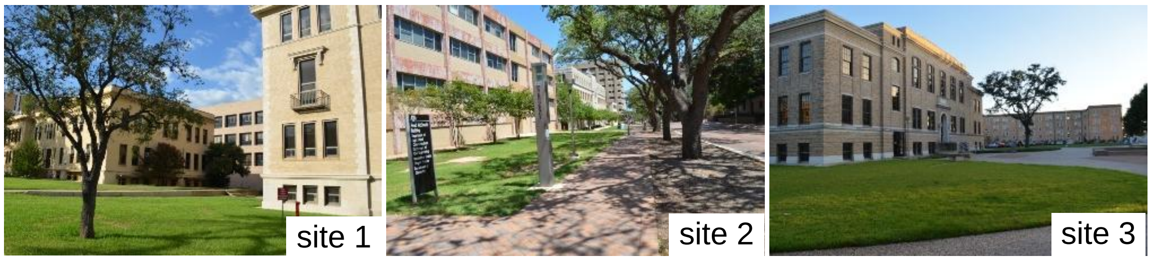


Figure 3.5: Experiment sites.

primary planes (col. 3) and their coplanar line segments (col. 4). Fig. 3.1 actually shows a 3D visualization of the map of high level landmarks constructed from data of site 1, where coplanar lines are color coded according to underlying planes.

We employ three error metrics to assess landmark mapping accuracy. ε_d and ε_a are defined for evaluating planes, and ε_L is defined for assessing lines. Suppose plane $\tilde{\boldsymbol{\pi}}_i^W$ is introduced to the map since the k_i-th frame I_{k_i}. Let d_i denote the true plane depth of $\tilde{\boldsymbol{\pi}}_i^W$ in $\{C_{k_i}\}$ obtained using a BOSCH GLR225 laser distance measurer with a range up to 70 m and measurement accuracy of ± 1.5 mm. Define $\hat{d}_i$ as the estimated value of d_i from EKF output. Then a relative metric for plane depth error is defined as

$$\varepsilon_d = \frac{1}{N} \sum_{i=1}^{N} \frac{\|d_i - \hat{d}_i\|}{d_i}, \tag{3.21}$$

where N is the number of landmark planes. Similarly, define ε_a to be the angular error metric for plane normal. It is worth noting that there exists global drifting error between $\{C_{k_i}\}$ and $\{W\}$, which will be addressed in future loop closure stage. Here we focus on ε_d and ε_a after the plane landmark appears in the camera field of view.

To evaluate a line landmark's estimation accuracy, we consider a re-projection error in ICS. Suppose $\mathbf{l}_j^0$ is added to the map since the k_j-th frame. Let $\hat{\mathbf{l}}_j^k$ be the re-projection of $\mathbf{l}_j^0$ in I_{k_j}, and $\mathbf{e}_h^{(j)}$ be the h-th observed endpoint of line segment in I_{k_j} associated with $\hat{\mathbf{l}}_j^k$. Then the error metric for lines is defined based on the distance between observed line segment endpoints and re-projected line in local frame:

$$\varepsilon_L = \frac{1}{M} \sum_{j=1}^{M} \frac{1}{N_j} \sum_{h=1}^{N_j} d_\perp(\mathbf{e}_h^{(j)}, \hat{\mathbf{l}}_j^k), \tag{3.22}$$

where $d_\perp(\cdot)$ represents the distance from a point to a line, M is the number of line landmarks and N_j is the number of line segment endpoints associated with $\hat{\mathbf{l}}_j^k$.

Tab. 3.1 shows the mapping results based on the three metrics. It is clear that our method successfully maps the high level landmarks. However, since loop closure has not been performed, the estimated camera trajectory inevitably suffers from drifting error, which will be addressed in the future work.

3.6 Conclusions

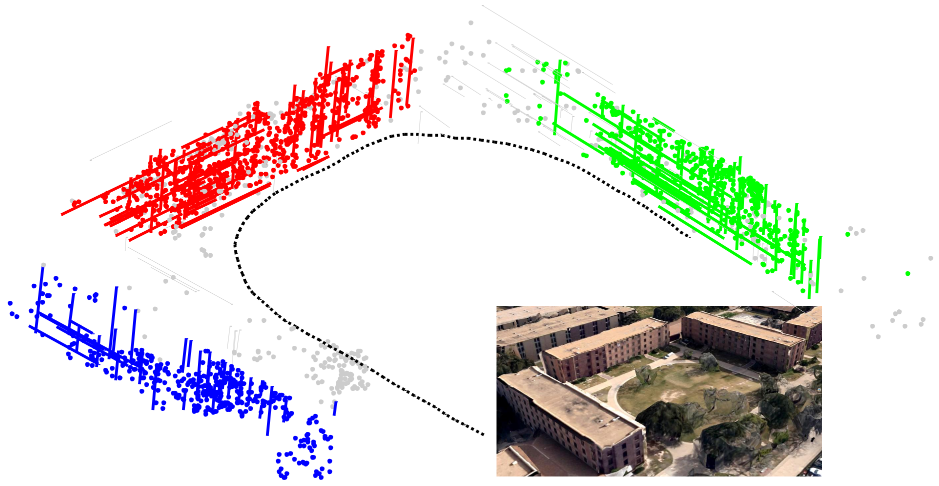
We developed a method to allow a mobile robot to perform mapping of building facades by enabling high level landmark mapping. The method incorporated a multiple layer feature graph into an EKF framework. We analyzed how errors are generated and propagated in the MFG construction process, which are used as observation error models in the EKF. We derived closed form solutions for error distribution to quantify the observation errors. Based on projective geometry, we derived observation models to complete the EKF framework. We implemented and tested the system at three different sites. Experiment results have shown that high level landmarks are successfully constructed in a modern urban environment with mean relative plane depth error less than 4.66%.

4. VISUAL NAVIGATION USING HETEROGENEOUS LANDMARKS AND UNSUPERVISED GEOMETRIC CONSTRAINTS*

While the MFG-EKF method in Section 3 is able to map the building facades from image sequences, the camera trajectory estimation is subject to obvious drifting as the travel distance increases. One reason is that the linearization step in EKF leads to inconsistencies due to the high nonlinearity in projective camera models. Recent studies [4] show that bundle adjustment-based SLAM approaches can produce better accuracy than EKF-based methods. Another limitation of the MFG-EKF method is its dependence on two view-based MFG. However, two view-based MFG needs a sufficient baseline, which is not always feasible. This motivates us to design a multiple view based MFG algorithm for visual SLAM using bundle adjustment.

In this section, we continue utilizing heterogeneous visual features and their inner geometric constraints to assist robot navigation, which is managed by a multiple view based MFG. Our method extends the local bundle adjustment-based SLAM framework by explicitly exploiting heterogeneous features and their geometric relationships in an unsupervised manner. The proposed heterogeneous landmark-based visual navigation (HLVN) algorithm takes a video stream as input, initializes and iteratively updates MFG based on extracted key frames, and refines robot localization and MFG landmarks. We present the algorithm pseudo code and analyze its complexity. We evaluate our method and compare it with state-of-the-art methods using multiple indoor and outdoor datasets. In particular, on the KITTI dataset our method reduces the translational error by 52.5% under urban sequences where rectilinear structures dominate the scene.

*Reprinted with permission from "Visual navigation using heterogeneous landmarks and unsupervised geometric constraints" by Y. Lu and D. Song, 2015. *IEEE Transcations on Robotics*, vol. 31, no. 3, pp. 736-749, Copyright © 2015 IEEE.



(a)

Figure 4.1: Sample result of our algorithm, and a Google Earth^{TM}view of the same site from a similar perspective. Coplanar landmarks (points and lines) are coded in the same color, while general landmarks are in gray color. The dotted line is the estimated camera trajectory.

4.1 Related Work

Visual navigation using heterogeneous landmarks mainly relates to two research fields: 3D reconstruction and SLAM.

In computer vision and graphics, 3D reconstruction has been a very popular topic for research as well as commercial applications. Besides regular cameras, sensors used for 3D reconstruction also include laser range finder [48] and more often, aerial cameras [49]. Google Earth and Microsoft Virtual Earth are successful showcases for 3D reconstruction of city models [50]. Following the taxonomy of Seitz et al. [51], 3D reconstruction algorithms are categorized into the following classes: voxel approaches [52], level-set techniques [53], line segment matching [54], polygon mesh methods [55], and algorithms that compute and merge depth maps [56]. Unlike those methods, our work does not pursue a full scale reconstruction. This is because 3D reconstruction usually needs repetitive scene scanning, which is often not the main task for robots.

In robotics research, the most common external sensors for robot navigation include sonar arrays, laser range finders, GPS, cameras and their combinations. SLAM is the typical framework employed in robot navigation [42, 57]. In SLAM, the physical world is represented as a collection of landmarks. For example, point clouds serve as landmarks when a laser range finder is the primary sensor [58]. In particular, our work belongs to the visual SLAM category, where cameras provide the main sensory input [59–62].

There are two prevalent methodologies in visual SLAM: the bundle adjustment (BA) approaches (e.g., [59]) rooted in the structure from motion area in computer vision, and the filtering methods (e.g. [15]) originated from the traditional SLAM field of robotics research. Strasdat et al. have analyzed the advantages of each method in [63]. For both methods, various camera configurations/modalities have been studied, including a monocular camera [64], a stereo camera [39, 65, 66], an omnidirectional camera [67], a camera with range sensors [68, 69], and an RGB-D camera [32, 70].

Besides methodology and sensor configuration, another critical issue in visual SLAM is scene representation. For example, point clouds and sparse feature points [71] are often employed as landmarks in a map. Recently, many researchers have realized that landmark selection is an important factor in visual odometry and SLAM [72]. Lower level landmarks such as corners [73] and SIFT points [17] are relatively easy to use due to their geometric simplicity. However, point features are merely mathematical singularities in color, texture and/or geometric spaces. They are difficult to interpret and use for scene understanding or human-robot interaction. They are also easily influenced by lighting and shadow conditions. To overcome these shortcomings, higher level landmarks have received more and more attention for visual SLAM, such as line segments [19–22], straight lines [23], vanishing points [24], and planes [25–28].

These works have demonstrated the advantages of higher level landmarks in robustness and accuracy, but they either treat these landmarks as isolated objects, or partially explore the inner relations between them. This treatment simplifies the SLAM problem formulation but cannot fully utilize the power of heterogeneous landmarks. Very recently, Tretyak et al. present an optimization framework for geometric parsing of image by jointly using edges, line segments, lines, and vanishing points [74]. However, this method has not been applied to navigation yet.

4.2 Problem Formulation

4.2.1 Assumptions and Notations

Consider a mobile robot navigating in a previously unknown environment with a monocular camera. We make two assumptions here:

a.1 The robot operates in a largely static man-made environment with rectilinear structures consisting of parallel lines which are not necessarily in orthogonal directions.

a.2 The camera is pre-calibrated with its radial distortion removed.

Let us define the following notations,

$\mathcal{V}$ Input camera video,

I_k k-th key frame extracted from $\mathcal{V}$, $I_k \in \mathcal{V}, k \in \mathbb{N}$,

$\{C_k\}$ 3D camera coordinate system for I_k, a right-handed coordinate system with its origin at the camera optical center, its Z-axis coinciding with the optical axis and pointing to the forward direction of the camera, and its X-axis and Y-axis parallel to the horizontal and vertical directions of the CCD sensor plane, respectively,

$\{I_k\}$ 2D image coordinate system for I_k, with its origin on the image plane and its u-axis and v-axis parallel to the X-axis and Y-axis of $\{C_k\}$, respectively,

$\{W\}$ 3D world coordinate system,

K Camera calibration matrix,

R_k Camera rotation matrix at I_k with respect to $\{W\}$,

t_k Camera translation vector at I_k with respect to $\{W\}$,

P_k Camera projection matrix, $\mathrm{P}_k = \mathrm{K}\ [\mathrm{R}_k \,|\, \mathrm{t}_k]$,

$\mathrm{R}_{k_2}^{k_1}$ Relative rotation matrix between I_{k_1} and I_{k_2} defined as $\mathrm{R}_{k_2}^{k_1} = \mathrm{R}_{k_2}\mathrm{R}_{k_1}^{-1}$

$\mathrm{t}_{k_2}^{k_1}$ Relative translation between I_{k_1} and I_{k_2} defined as $\mathrm{t}_{k_2}^{k_1} = \mathrm{t}_{k_2} - \mathrm{R}_{k_2}^{k_1}\mathrm{t}_{k_1}$,

$X_{i:j}$ Collection defined as $X_{i:j} = \{X_k | i \le k \le j\}$,

$\mathbf{m}_k$ A 2D MFG (defined later) constructed for I_k,

$\mathcal{M}_k$ 3D MFG (defined later) constructed based on $I_{0:k}$,

$\mathbb{E}^n$ n-dimensional Euclidean space,

$\mathbb{P}^n$ n-dimensional projective space, and

$\mathbf{X}$ A homogeneous vector, $\mathbf{X} = [\tilde{\mathbf{X}}^\mathsf{T}, 1]^\mathsf{T}$, where $\tilde{\mathbf{X}}$ denotes the inhomogeneous counterpart of $\mathbf{X}$. $\mathbf{X} \in \mathbb{P}^n \Rightarrow \tilde{\mathbf{X}} \in \mathbb{E}^n$.

We abuse "=" to denote real equality and up-to-scale equality for inhomogeneous and homogeneous vectors, respectively.

4.2.2 Multilayer Feature Graph

As illustrated in Fig. 4.2, we redesign multilayer feature graph for organizing heterogeneous features and their inner geometric relations. MFG includes the following type of nodes.

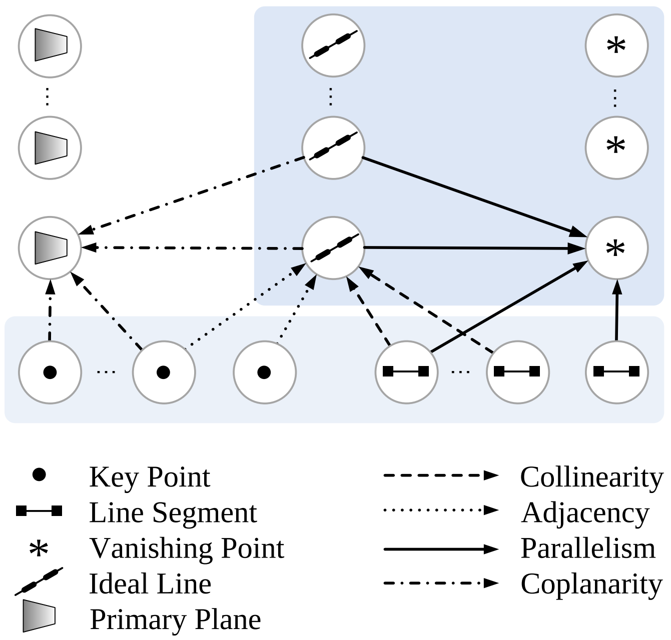


Figure 4.2: The whole graph represents a 3D MFG, and the shaded regions jointly represent a 2D MFG. Geometric relationships between nodes are represented by edges of different line types.

1. **Key points** represent point features. We refer to point features detected from images as 2D key points, which only reside in image space. Thus the set of 2D key points detected in I_k is denoted by $\{\mathbf{p}_{i,k} \in \mathbb{P}^2, i = 1, 2, \cdots\}$. We refer to spatial points as 3D key points, and represent them with respect to $\{W\}$ by $\{\mathbf{P}_j \in \mathbb{P}^3, j = 1, 2, \cdots\}$. The observation of $\mathbf{P}_j$ in I_k, if existing, is denoted by $\mathbf{p}_{j(k)}$. Therefore, if $\mathbf{p}_{i,k}$ is the observation of $\mathbf{P}_j$ in I_k, then we have $\mathbf{p}_{j(k)} = \mathbf{p}_{i,k}$ by definition. Note that the subscript convention used in naming $\mathbf{p}_{i,k}$ and $\mathbf{p}_{j(k)}$ also applies to other types of features in this section.

2. **Line segments** represent finite linear objects. We denote a 2D line segment in I_k by $\mathbf{s}_{i,k} = [\mathbf{d}_{i1,k}^\mathsf{T}, \mathbf{d}_{i2,k}^\mathsf{T}]^\mathsf{T}$, where $\mathbf{d}_{i1,k}$ and $\mathbf{d}_{i2,k}$ are the endpoints. We represent a 3D line segment in $\{W\}$ by $\mathbf{S}_i = [\mathbf{D}_{i1}^\mathsf{T}, \mathbf{D}_{i2}^\mathsf{T}]^\mathsf{T}$.

3. **Ideal lines** (defined later in Definition 3) represent infinite lines. A 2D ideal line in I_k is represented by $\mathbf{l}_{i,k} \in \mathbb{P}^2$. We represent a 3D ideal line by $\mathbf{L}_i = [\mathbf{Q}_i^\mathsf{T}, \mathbf{J}_i^\mathsf{T}]^\mathsf{T}$, where $\mathbf{Q}_i \in \mathbb{P}^3$ is a finite 3D point located on $\mathbf{L}_i$, and $\mathbf{J}_i \in \mathbb{P}^3$ is an infinite 3D point defining the direction of $\mathbf{L}_i$. The observation of $\mathbf{L}_i$ in I_k is denoted by $\mathbf{l}_{i(k)}$.

4. **Vanishing points** represent particular directions of parallel 3D lines. We denote a 2D vanishing point in I_k by $\mathbf{v}_{i,k}$, and a 3D vanishing point in $\{W\}$ by $\mathbf{V}_i$. $\mathbf{V}_i \in \mathbb{P}^3$ is an infinite 3D point, and its observation in I_k is denoted by $\mathbf{v}_{i(k)}$.

5. **Primary planes** represent dominant planar surfaces (e.g. building facades) and only exist in 3D space. We denote a primary plane by $\mathbf{\Pi}_i = [\mathrm{n}_i^\mathsf{T}, d_i]^\mathsf{T}$, where $\mathrm{n}_i \in \mathbb{E}^3$ and $d_i \in \mathbb{R}$, such that $\mathbf{X}^\mathsf{T}\mathbf{\Pi}_i = 0$ for any point $\mathbf{X}$ on the plane.

MFG exists in both $\{I_k\}$ and $\{W\}$. In $\{I_k\}$, we name it as a 2D MFG, which consists of 2D key points, 2D line segments, 2D ideal lines, and 2D vanishing points as its nodes. The geometric relationships between 2D features, including adjacency, collinearity, and parallelism, are represented by the edges of 2D MFG. A 2D MFG effectively summarizes the feature information of a frame. Thus we construct a 2D MFG for each key frame I_k and denote it by $\mathbf{m}_k$. In Fig. 4.2, the shaded regions jointly represent the structure of a 2D MFG. The top shaded region consists of raw features that are directly extracted from images, while the lower shaded region includes features that need to be abstracted from raw features.

In $\{W\}$, we define a 3D MFG, which contains 3D key points, 3D line segments, 3D ideal lines, 3D vanishing points, and primary planes as its nodes. The edges of 3D MFG represent geometric relationships including collinearity, parallelism, and coplanarity. There is only one 3D MFG in $\{W\}$ and we use $\mathcal{M}_k$ to denote the 3D MFG constructed/updated based upon $I_{0:k}$.

4.2.3 Problem Definition

Our ultimate goal is to construct a 3D MFG from an input video. To achieve this goal, we utilize an iterative method to solve the following problem.

Definition 2. *Given video $\mathcal{V}$, MFG $\mathcal{M}_{k-1}$, and historical camera poses $\{\mathrm{R}_{0:k-1}, \mathrm{t}_{0:k-1}\}$ for $k \geq 1$, select key frame I_k, estimate camera pose $\{\mathrm{R}_k, \mathrm{t}_k\}$, refine $\{\mathrm{R}_{0:k}, \mathrm{t}_{0:k}\}$, and update the nodes and edges of $\mathcal{M}_{k-1}$ to obtain $\mathcal{M}_k$.*

4.3 System Design and Multilayer Feature Graph

Fig. 4.3 shows our system architecture, where the main blocks are shaded by different colors. The system takes a video as input and proceeds iteratively. During each iteration, the system selects a key frame I_k, extracts a 2D MFG $\mathbf{m}_k$, and finds 2D feature correspondences between $\mathbf{m}_k$ and $\mathbf{m}_{k-1}$, which are used to estimate camera pose and establish 3D features for $\mathcal{M}_k$. The last step of each iteration performs LBA to jointly refine camera poses and 3D MFG features.

To start, we select the first video frame as key frame I_0. We let $\mathcal{M}_0 = \emptyset$ and $\{W\}$ coincide with $\{C_0\}$.

4.3.1 Key Frame Selection

Given a video, it is necessary to select a set of key frames for motion estimation and 3D reconstruction. The basic principle is to find a good balance between two needs: 1) wide baseline to avoid ill-posed epipolar geometry problems and 2) sufficient overlap of scene between key frames. Based on existing methods (e.g. [64]), we use the following criteria for key frame selection when $k \geq 1$. Given I_{k-1} and $\mathcal{M}_{k-1}$, a video frame I is chosen as key frame I_k if it satisfies:

1. the number of 2D point matches between I_{k-1} and I is not less than a threshold N_2,

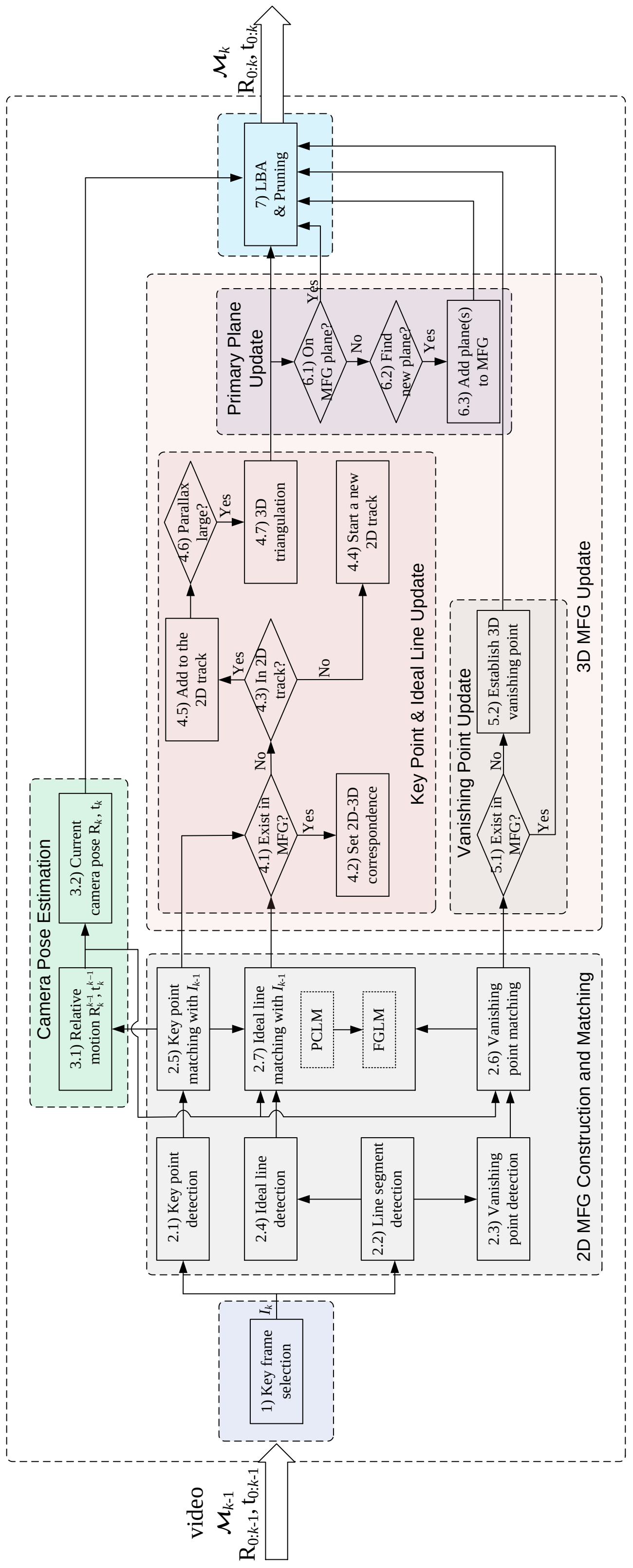


Figure 4.3: System diagram.

2. for $k \geq 2$, the number of 3D key points (from $\mathcal{M}_{k-1}$) observable in I is not less than a threshold N_3,

3. for $k \geq 2$, the rotation angle between I_{k-1} and I is not larger than a threshold τ_R, and

4. there are as many video frames as possible between I_{k-1} and I.

4.3.2 2D MFG Construction and Matching

From I_k we construct a 2D MFG $\mathbf{m}_k$, and match $\mathbf{m}_k$ with $\mathbf{m}_{k-1}$ for $k \geq 1$ to establish 2D-2D matching for heterogeneous features. We discuss the extraction and matching for each type of features separately.

4.3.2.1 ***Key Points***

We detect 2D key points from I_k using the corner detector proposed in [73], though alternatives such as SIFT are also applicable. We track 2D feature points across frames using the iterative Lucas-Kanade method with pyramids [75]. Thus, putative key point correspondences between I_k and I_{k-1} (for $k \geq 1$) are readily obtained from the tracking result (see Box 2.5 in Fig. 4.3). To remove false matching, the putative matches are fed into a five-point algorithm-based RANSAC [76] to estimate the essential matrix E. We also compute the relative camera rotation R_k^{k-1} by decomposing E. Note that although the relative motion estimation (in Box 3.1) belongs to the “camera pose estimation” block in Fig. 4.3, it is indeed conducted as soon as putative key point correspondences are available.

4.3.2.2 ***Line Segments***

We detect 2D line segments from I_k using LSD [44] (see Box 2.2 in Fig. 4.3). Line segments provide more information in addition to key points, but line segment matching is hard due to the lack of distinctive descriptors and the instability of endpoint detection. However, we use line segments to find vanishing points.

*4.3.2.3 **Vanishing Points***

We detect vanishing points from 2D line segments using RANSAC (see Box 2.3 in Fig. 4.3). In a 2D MFG, each vanishing point has a set of child line segments, which are actually parallel to each other in 3D space.

To find correspondences between two vanishing point sets, $\{\mathbf{v}_{i,k-1}|i = 1, \cdots\}$ and $\{\mathbf{v}_{j,k}|j = 1, \cdots\}$, we compute

$$\theta_{ij} = \cos^{-1}(|(\mathrm{K}^{-1}\mathbf{v}_{i,k-1})^{\mathsf{T}}\mathrm{R}_k^{k-1}\mathrm{K}^{-1}\mathbf{v}_{j,k}|), \forall i, j, \tag{4.1}$$

which represents the angle between the two vanishing point directions in 3D.

Let $\theta^*_{\cdot j} = \min_\iota(\theta_{\iota j})$, and $\theta^*_{i\cdot} = \min_\iota(\theta_{i\iota})$. We claim $\mathbf{v}_{i,k-1} \leftrightarrow \mathbf{v}_{j,k}$ as a correspondence if it holds that

$$\theta_{ij} = \theta^*_{\cdot j} = \theta^*_{i\cdot} \leq \tau_\theta, \tag{4.2}$$

where τ_θ is a user-specified upper bound. Fig. 4.4 shows an example of vanishing point matching result.

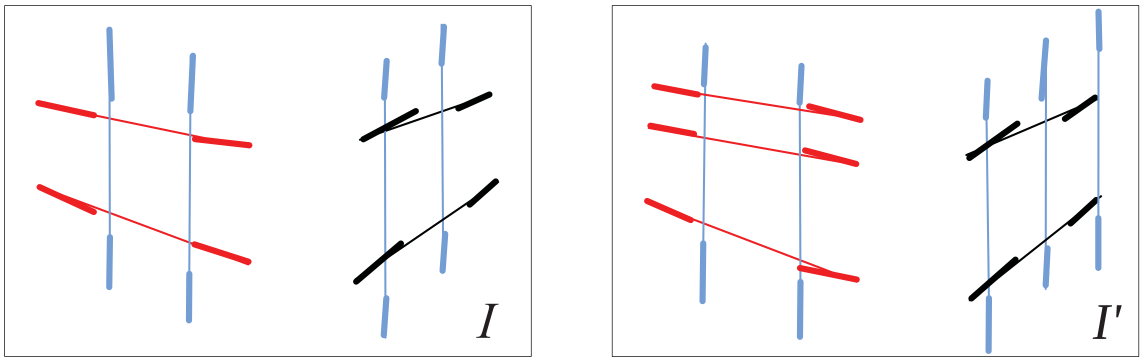


Figure 4.4: An example of vanishing point matching. The line segments and ideal lines associated with the same vanishing points are drawn in the same color.

4.3.2.4 Ideal Lines

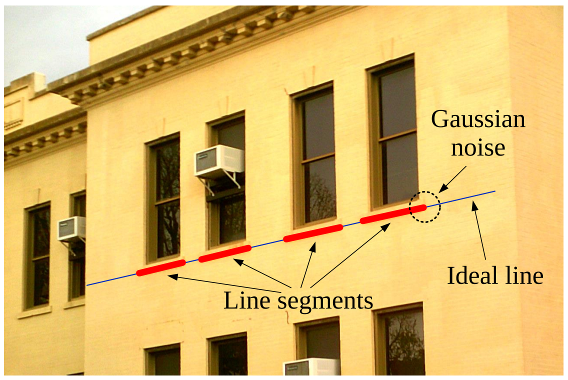
As illustrated in Fig. 4.5(a), line segments tend to be short and fragmented. However, there are usually many small groups of collinear line segments which come from the same 3D linear structure. If we fuse the information of collinear line segments to form a single line, the accuracy and robustness should be improved. Therefore, we introduce ideal lines.

Definition 3 (Ideal Line)**.** *An ideal line is defined as a real or virtual line passing through a set of collinear line segments.*

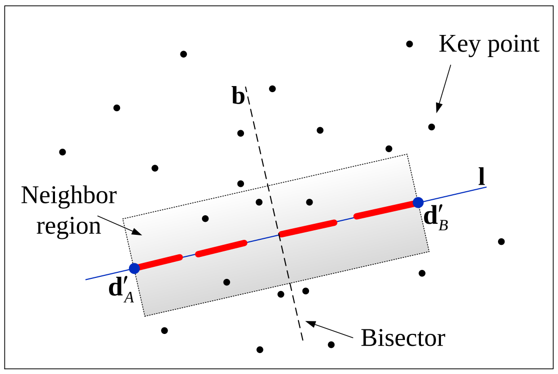
We detect 2D ideal lines from line segments using sequential RANSAC (see Box 2.4 in Fig. 4.3). To reduce the problem size, we group line segments by vanishing point and detect ideal lines group by group. After a set of collinear line segments $\{\mathbf{s}_i\}$ is found, we compute the ideal line as

$$\mathbf{l}^* = \operatorname*{argmin}_{\mathbf{l}} \sum_i \sum_{j=1}^{2} d_\perp(\mathbf{d}_{ij}, \mathbf{l})^2, \tag{4.3}$$

where $d_\perp(\cdot,\cdot)$ denotes the perpendicular distance from a point to a line in 2D. This method



(a)



(b)

Figure 4.5: (a) An example of an ideal line. (b) An example of the adjacency between key points and ideal lines.

is effectively the maximum likelihood estimation (MLE) when each line segment endpoint is subject to an isotropic Gaussian noise as illustrated in Fig. 4.5(a). In MFG, a line segment must have only one ideal line as its parent node, and an ideal line may have multiple line segment nodes as its children.

We consider two ideal lines from different images to be matched if they correspond to the same 3D line. Since vanishing point matching is done, we narrow down the ideal line matching problem by only considering lines connected to the same vanishing point. We present a two-stage approach to ideal line matching here (see Box 2.7 in Fig. 4.3).

Stage 1: Point Correspondence-based Line Matching

In Stage 1, we adopt a point correspondence-based line matching (PCLM) method proposed by [77]. To apply this method, we first introduce the neighbor region of an ideal line. For an ideal line $\mathbf{l}$, let $\mathbf{d}_A$ and $\mathbf{d}_B$ be the two farthest line segment endpoints associated with $\mathbf{l}$, and $\mathbf{d}'_A$ and $\mathbf{d}'_B$ be the projections of $\mathbf{d}_A$ and $\mathbf{d}_B$ onto $\mathbf{l}$, respectively. Let $\mathbf{b}$ be the perpendicular bisector of line segment $\mathbf{d}'_A\mathbf{d}'_B$, as illustrated in Fig. 4.5(b). We define the neighbor region of $\mathbf{l}$ to be

$$\lambda(\mathbf{l}) := \left\{ \mathbf{x} \in \mathbb{P}^2 : d_\perp(\mathbf{x}, \mathbf{b}) \le \frac{\|\mathbf{d}'_A\mathbf{d}'_B\|}{2}, d_\perp(\mathbf{x}, \mathbf{l}) \le \frac{\sigma_b}{2} \right\}, \tag{4.4}$$

where $\|\mathbf{d}'_A\mathbf{d}'_B\|$ is the length of line segment $\mathbf{d}'_A\mathbf{d}'_B$, and $\sigma_b = \min\{100, (\text{image width})/12\}$. In Fig. 4.5(b), $\lambda(\mathbf{l})$ is the rectangular area enclosed by dotted lines. If a key point $\mathbf{p} \in \lambda(\mathbf{l})$, then we say $\mathbf{p}$ is adjacent to $\mathbf{l}$ and they are connected in 2D MFG.

The intuition of the PCLM method is that for an ideal line match $\mathbf{l}_{i,k-1} \leftrightarrow \mathbf{l}_{j,k}$, point correspondences in their neighbor regions must satisfy

$$\frac{\mathbf{l}_{i,k-1}^{\mathsf{T}}\mathbf{p}_{a,k-1}}{\mathbf{l}_{i,k-1}^{\mathsf{T}}\mathbf{p}_{b,k-1}} = \frac{\mathbf{l}_{j,k}^{\mathsf{T}}\mathbf{p}_{c,k}}{\mathbf{l}_{j,k}^{\mathsf{T}}\mathbf{p}_{d,k}} \tag{4.5}$$

where $\mathbf{p}_{a,k-1} \leftrightarrow \mathbf{p}_{c,k}$ and $\mathbf{p}_{b,k-1} \leftrightarrow \mathbf{p}_{d,k}$ are two pairs of point correspondences satisfying $\mathbf{p}_{a,k-1}, \mathbf{p}_{b,k-1} \in \lambda(\mathbf{l}_{i,k-1})$, and $\mathbf{p}_{c,k}, \mathbf{p}_{d,k} \in \lambda(\mathbf{l}_{j,k})$.

The PCLM method provides very accurate matching result, but tends to fail when the neighbor regions are textureless. To handle this issue, in Stage 2 we use an F-guided line matching (FGLM) method [78], which utilizes the fundamental matrix F. To be specific, we take the line matches found by PCLM out of the candidate sets, and apply FGLM to the remaining ideal lines. The fundamental matrix is computed as $\mathrm{F} = \mathrm{K}^{-\mathsf{T}}\mathrm{E}\mathrm{K}^{-1}$.

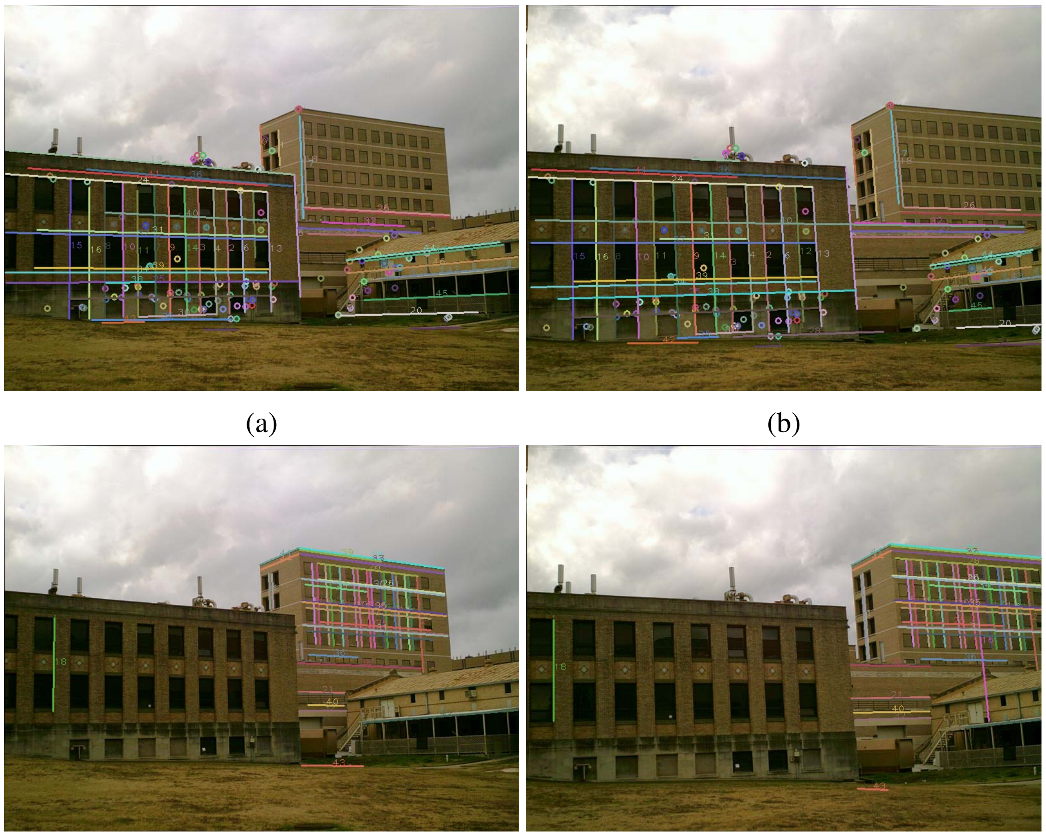
Stage 2: F-Guided Line Matching

Since the FGLM method essentially works with line segments, we treat an ideal line as an augmented line segment. To be specific, we only use the part of ideal line inside the neighbor region (for example, $\mathbf{d}'_A\mathbf{d}'_B$ in Fig. 4.5(b)) since this is the part that bears most appearance information.

For a point $\mathbf{x}$ on the augmented line segment of $\mathbf{l}_{i,k-1}$, we find its correspondence $\mathbf{x}' = \mathbf{l}_{j,k} \times (\mathrm{F}\mathbf{x})$, assuming $\mathbf{l}_{i,k-1}$ and $\mathbf{l}_{j,k}$ correspond to each other. The basic idea of FGLM is to compute the matching score of a pair of line segments as the average of the individual correlation scores for the points (pixels) of the lines. Although FGLM does not produce very accurate results, but it is complementary to the PCLM method. As an example, Fig. 4.6(a) and 4.6(b) show the matching result of the PCLM method, and Fig. 4.6(c) and 4.6(d) show the result of the FGLM method. It is obvious that the two-stage matching approach is able to find more ideal line correspondences.

4.3.3 Camera Pose Estimation

With 2D feature correspondences obtained, estimating the 6 degrees of freedom (DoF) camera pose R_k and t_k for I_k is a key step for constructing and updating 3D MFG. Existing methods (e.g. [59]) usually solve this problem using 3-point algorithm based on the 3D-2D correspondences $\{\mathbf{P}_i \leftrightarrow \mathbf{p}_{i(k)}\}$ between known 3D points and their observations in


(a) (b)

(c) (d)

Figure 4.6: An example of our two-stage approach to ideal line matching. (a) and (b): Ideal line matches found in Stage 1 by the PCLM method, each pair of line match is plotted in the same color, and small circles represent point correspondences used by PCLM; (c) and (d): Additional matches found by the FGLM method in Stage 2. (Best viewed in color)

I_k. This method omits those 2D-2D correspondences between I_{k-1} and I_k whose 3D positions are unknown yet. This omission will lead to large estimation error when observed 3D points are few. Various approaches exist to handle this issue, e.g. using Kalman filtering [79] or three-view constraints [80]. A good fit for our system is a method proposed by Tardif et al. [67] that decouples the estimation of R_k from t_k in two steps. We adopt this method with the modification as follows.

Step 1: Compute essential matrix E as described in Section 4.3.2.1. Decompose

E to recover the relative camera rotation R_k^{k-1} and translation t_k^{k-1}, with $\|\mathrm{t}_k^{k-1}\|$ unknown.

Step 2: Compute the translation distance $\|\mathrm{t}_k^{k-1}\|$ using 3D-2D correspondences through a RANSAC process where only one correspondence is needed for a minimal solution. This completes the 6 DoF estimation.

In the Step 2 of [67], Tardif et al. estimate the full 3 DoFs of t_k^{k-1} using two 3D-2D correspondences for a minimal solution. This difference can be justified by the different cameras used - an omnidirectional camera in [67] with 360° horizontal field of view (HFOV) vs. a regular camera we use with $40°-80°$ HFOV. Narrower HFOV results in fewer observable 3D landmarks in view and thus fewer 3D-2D correspondences, especially in a turning situation. Therefore, we choose to reduce the problem dimension in Step 2 to fit our needs.

It is worth noting that when $k = 1$, we do not need Step 2, but set $\|\mathrm{t}_k^{k-1}\| = 1$. This fixes the scale of the following estimations.

4.3.4 3D MFG Update

We initialize $\mathcal{M}_k$ by letting $\mathcal{M}_k = \mathcal{M}_{k-1}$ and then perform 3D MFG update for $\mathcal{M}_k$ using 2D information just obtained. This is a process of associating 2D features in $\mathbf{m}_k$ with 3D landmarks in $\mathcal{M}_k$ and introducing new 3D landmarks into $\mathcal{M}_k$. We present details for each type of landmarks as follows.

4.3.4.1 ***Key Point Update***

Key point update involves associating image observations to existing 3D key points, and establishing new 3D key points using new 2D key point correspondences (see Boxes 4.1-4.6 in Fig. 4.3).

A 2D point correspondence must have sufficient parallax to be used for computing a

3D point. Here we define the parallax of a 2D key point correspondence $\mathbf{p}_{i,k_1} \leftrightarrow \mathbf{p}_{j,k_2}$ as

$$\rho(\mathbf{p}_{i,k_1}, \mathbf{p}_{j,k_2}) := \langle \mathrm{K}^{-1}\mathrm{H}_\mathrm{r}\mathbf{p}_{i,k_1}, \mathrm{K}^{-1}\mathbf{p}_{j,k_2} \rangle \tag{4.6}$$

$$\text{with } \mathrm{H}_\mathrm{r} = \mathrm{K}\mathrm{R}_{k_2}^{k_1}\mathrm{K}^{-1} \tag{4.7}$$

where H_r represents a rotational homography [45], $\langle \cdot, \cdot \rangle$ indicates the angle between two vectors, and $\mathrm{R}_{k_2}^{k_1}$ has been computed in Section 4.3.3.

For a 2D key point correspondence $\mathbf{p}_{i,k-1} \leftrightarrow \mathbf{p}_{j,k}$,

- if it is a re-observation of key point $\mathbf{P}_\iota$, we make the association by letting $\mathbf{p}_{\iota(k)} = \mathbf{p}_{j,k}$.
- if it is a newly discovered point, compute its parallax $\rho(\mathbf{p}_{i,k-1}, \mathbf{p}_{j,k})$ using (4.6). If $\rho(\mathbf{p}_{i,k-1}, \mathbf{p}_{j,k}) > \tau_\rho$ where τ_ρ is a parallax threshold, we triangulate it and add the 3D point to $\mathcal{M}_k$ as a new key point. Otherwise, we set up a new 2D key point track $\mathcal{Q}_q = \{\mathbf{p}_{i,k-1}, \mathbf{p}_{j,k}\}$ to keep track of it for potential triangulation in the future. A 2D key point track is a collection of 2D key points corresponding to a 3D point whose position is not computed yet due to insufficient parallax.
- if it is an observation of an existing 2D key point track $\mathcal{Q}_q$, we append it to the track $\mathcal{Q}_q = \mathcal{Q}_q \cup \{\mathbf{p}_{j,k}\}$, and check whether $\mathcal{Q}_q$ can be converted to a 3D key point. To do this, we compute the parallax between $\mathbf{p}_{j,k}$ and each of the rest points in $\mathcal{Q}_q$. If anyone is larger than τ, we compute a 3D point from all points in $\mathcal{Q}_q$ and add it to $\mathcal{M}_k$; $\mathcal{Q}_q$ is then deleted.

*4.3.4.2 **Vanishing Point Update***

Vanishing point update is straightforward (see Boxes 5.1-5.2 in Fig. 4.3). Given a 2D vanishing point $\mathbf{v}_{i,k}$, if it is a re-observation of existing $\mathbf{V}_j$, let $\mathbf{v}_{j(k)} = \mathbf{v}_{i,k}$. Otherwise, establish a new vanishing point node $\mathbf{V}_j = [\mathbf{v}_{i,k}^{\mathsf{T}}\mathrm{R}_k, 0]^{\mathsf{T}}$. It is trivial but important to update the edges between ideal lines and vanishing points whenever a new ideal line or vanishing point node is added.

4.3.4.3 ***Ideal Line Update***

Before presenting the ideal line update algorithm, we need to define the parallax for ideal lines. Generally speaking, parallax has not been clearly defined for lines. Here we propose a heuristic parallax measurement for ideal lines by leveraging their line segment endpoints. For a 2D ideal line correspondence $\mathbf{l}_{i,k_1} \leftrightarrow \mathbf{l}_{j,k_2}$, define

$$\varrho(\mathbf{l}_{i,k_1}, \mathbf{l}_{j,k_2}) := \frac{1}{n}\sum_{\iota=1}^{n} \rho(\mathbf{d}_{\iota,k_1}, \mathbf{d}_{\iota,k_2}^{+}) \tag{4.8}$$

where $\{\mathbf{d}_{\iota,k_1} | \iota = 1, \cdots, n\}$ denotes the endpoints of line segments that support $\mathbf{l}_{i,k_1}$, and $\mathbf{d}_{\iota,k_2}^{+}$ is the perpendicular foot of $\mathbf{d}'_{\iota,k_2} := \mathrm{H}_\mathrm{r}\mathbf{d}_{\iota,k_1}$ on $\mathbf{l}_{j,k_2}$ in I_{k_2}, as illustrated in Fig. 4.7.

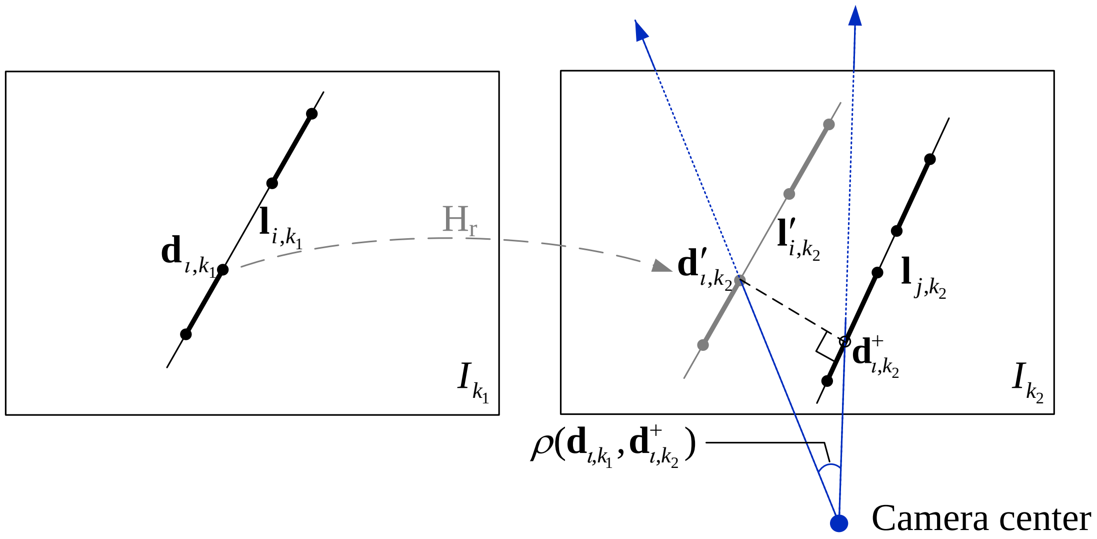


Figure 4.7: Illustration of parallax computation for 2D ideal lines. H_r is a rotational homography defined in (4.7). Bold lines are supporting line segments of the underlying (thin) ideal line. $\rho(\mathbf{d}_{\iota,k_1}, \mathbf{d}_{\iota,k_2}^{+})$ is the parallax between points $\mathbf{d}_{\iota,k_1}$ and $\mathbf{d}_{\iota,k_2}^{+}$.

The rationale is that we want to reward line correspondences which have larger distance in their perpendicular direction. If $\mathbf{l}'_{i,k_2} := \mathrm{H}_\mathrm{r}^{-\mathsf{T}}\mathbf{l}_{i,k_1}$ overlap with $\mathbf{l}_{j,k_2}$, their parallax should be zero.

With the parallax defined, the ideal line update is performed in a similar fashion to the key point case (i.e. Boxes 4.1-4.6 in Fig. 4.3), and thus skipped here.

Remark 1. *3D Line segments are also updated in this process. Since a line segment always has an ideal line parent, when a 2D ideal line is converted to 3D, its associated line segments are also converted to 3D. Their 3D positions are computed based on the 3D ideal line parameters.*

*4.3.4.4 **Primary Plane Update***

Detecting primary planes is of great importance for robot navigation. Here we detect primary planes by finding coplanar 3D key points and ideal lines using RANSAC. To be specific, let $\mathcal{C}$ be a collection of 3D key points and ideal lines which are not yet associated with any primary plane. We briefly describe two key steps of RANSAC below.

1. Compute a plane candidate $\mathbf{\Gamma}$ from a minimal solution set, which could include either 3 key points, or 2 parallel ideal lines, or 1 key point plus 1 ideal line.

2. $\forall c \in \mathcal{C}$, compute a consensus score $f(c, \mathbf{\Gamma})$ as follows.

$$f(c,\mathbf{\Gamma}) = \begin{cases} \delta_\perp(c,\mathbf{\Gamma}) & \text{if } c \text{ is a key point} \\ \frac{1}{n}\sum_{i=1}^{n}\delta_\perp(D_i,\mathbf{\Gamma}) & \text{if } c \text{ is an ideal line} \end{cases} \tag{4.9}$$

   where $\delta_\perp(\cdot,\cdot)$ denotes the perpendicular distance from a point to a plane in 3D, and $\{D_i | i = 1, \cdots, n\}$ is the set of 3D endpoints associated with ideal line c. Therefore, if c is an ideal line, $f(c, \mathbf{\Gamma})$ is the average of the distances from its associated line segment endpoints to $\mathbf{\Gamma}$.

If the size of the largest consensus set is greater than a threshold N_{cp}, we add the corresponding plane candidate to $\mathcal{M}_k$ as a primary plane, and establish edges between it and the key points and ideal lines in the consensus set. To control the problem size, we do not include all 3D key points or ideal lines in $\mathcal{C}$. Instead, we only take into account those recently established landmarks. Here we enforce $|\mathcal{C}| \leq$450.

Moreover, when new 3D key points or ideal lines are established, we check if they belong to existing primary planes using the metric defined by (4.9) and add edges accordingly. An ideal line may have two parent primary planes if it is a boundary line.

4.4 LBA and Pruning

4.4.1 LBA with Geometric Constraints

After the 3D MFG is updated, we further refine the estimated camera poses and 3D landmarks jointly using LBA (see Box 7 in Fig. 4.3). Inspired by [67], we use w latest key frames to bundle adjust m latest camera poses and MFG nodes established since I_{k-m+1}, with $w \geq m$ usually. To account for the various feature types and geometric constraints in MFG, we define cost functions accordingly.

4.4.1.1 Key Points

Denote the reprojection of key point $\mathbf{P}_i$ in I_k by $\hat{\mathbf{p}}_{i(k)} := \mathrm{P}_k\mathbf{P}_i$. Recall that the observation of $\mathbf{P}_i$ in I_k is $\mathbf{p}_{i(k)}$. Usually $\hat{\mathbf{p}}_{i(k)} \neq \mathbf{p}_{i(k)}$ due to image noise. Here we assume $\mathbf{p}_{i(k)}$ is subject to a zero-mean Gaussian noise $\mathcal{N}(\mathbf{0}, \Lambda_{\mathbf{p}})$.

Define the cost function for $\mathbf{P}_i$ in I_k to be

$$\mathcal{C}_p(\mathbf{P}_i, k) = (\tilde{\hat{\mathbf{p}}}_{i(k)} - \tilde{\mathbf{p}}_{i(k)})^\mathsf{T} \Lambda_{\mathbf{p}}^{-1} (\tilde{\hat{\mathbf{p}}}_{i(k)} - \tilde{\mathbf{p}}_{i(k)}). \tag{4.10}$$

4.4.1.2 Ideal Lines & Collinearity

Denote the reprojection of ideal line $\mathbf{L}_i$ in I_k by $\hat{\mathbf{l}}_{i(k)} := (\mathrm{P}_k\mathbf{Q}_i) \times (\mathrm{P}_k\mathbf{J}_i)$. Since the observation of $\mathbf{L}_i$ in I_k, i.e. $\mathbf{l}_{i(k)}$, is estimated from its supporting line segments $\{\mathbf{s}_{\iota,k} | \iota = 1, \cdots\}$, we directly treat these line segments as its observations for cost function definition. The measurement noise of 2D line segments can be modeled in various ways. Here we adopt a simple but well-accepted modeling [81], which assumes each line segment endpoint is subject to a zero-mean Gaussian noise $\mathcal{N}(\mathbf{0}, \sigma_d^2\mathbf{I}_2)$, where σ_d is a scalar, and $\mathbf{I}_2$ is a 2×2 identity matrix.

Define the cost function for $\mathbf{L}_i$ in I_k as

$$\mathcal{C}_l(\mathbf{L}_i, k) = \sum_{\iota} \sum_{j=1}^{2} \left(\frac{d_\perp(\tilde{\mathbf{d}}_{\iota j,k}, \hat{\mathbf{l}}_{i(k)})}{\sigma_d} \right)^2. \tag{4.11}$$

This cost function effectively captures the *collinearity* constraint between ideal lines and line segments.

4.4.1.3 Vanishing Points & Parallelism

Let the reprojection of vanishing point $\mathbf{V}_i$ in I_k be $\hat{\mathbf{v}}_{i(k)} := \mathrm{P}_k\mathbf{V}_i$. The observation of $\mathbf{V}_i$ in I_k is $\mathbf{v}_{i(k)}$ which is the intersection of 2D line segments from the same parallel group. Since $\mathbf{v}_{i(k)}$ is estimated from line segments, its estimation covariance $\Lambda_{\mathbf{v}_{i(k)}}$ is easily derived as well [81]. Define the cost function for $\mathbf{V}_i$ in I_k by

$$\mathcal{C}_v(\mathbf{V}_i, k) = (\hat{\mathbf{v}}_{i(k)} - \mathbf{v}_{i(k)})^\mathsf{T} \Lambda_{\mathbf{v}_{i(k)}}^{-1} (\hat{\mathbf{v}}_{i(k)} - \mathbf{v}_{i(k)}). \tag{4.12}$$

In particular, for all ideal lines $\{\mathbf{L}_j\}$ connected to $\mathbf{V}_i$ in MFG, we enforce $\mathbf{L}_j = [\mathbf{Q}_j^\mathsf{T}, \mathbf{V}_i^\mathsf{T}]^\mathsf{T}$ such that these lines are strictly parallel. Recall that $\mathbf{Q}_j$ is a finite point on $\mathbf{L}_j$. This parameterization and cost function (4.12) together account for the *parallelism* relationship in MFG.

4.4.1.4 Primary Planes & Coplanarity

Primary plane $\mathbf{\Pi}_i$ has neither reprojection nor direct observation in image space. Therefore, we define its cost function by leveraging 3D key points and ideal lines, respectively. For key point $\mathbf{P}_j$ and primary plane $\mathbf{\Pi}_i$, define

$$\mathcal{C}_\pi(\mathbf{P}_j, \mathbf{\Pi}_i) = \begin{cases} [\delta_\perp(\mathbf{P}_j, \mathbf{\Pi}_i)]^2 & \text{if } \mathbf{P}_j \in \mathbf{\Pi}_i \\ 0 & \text{otherwise} \end{cases} \tag{4.13}$$

where $\mathbf{P}_j \in \mathbf{\Pi}_i$ indicates that $\mathbf{P}_j$ is connected with $\mathbf{\Pi}_i$ in MFG.

For ideal line $\mathbf{L}_j$ and primary plane $\mathbf{\Pi}_i$, define

$$\mathcal{C}_\pi(\mathbf{L}_j, \mathbf{\Pi}_i) = \begin{cases} \frac{1}{n}\sum_{\iota=1}^{n}[\delta_\perp(\mathbf{D}_\iota, \mathbf{\Pi}_i)]^2 & \text{if } \mathbf{L}_j \in \mathbf{\Pi}_i \\ 0 & \text{otherwise} \end{cases} \tag{4.14}$$

where $\{\mathbf{D}_\iota | \iota = 1, \cdots, n\}$ denote the endpoints of all the line segments that support $\mathbf{L}_j$.

Eqs. (4.13) and (4.14) represent the *coplanarity* constraint in MFG.

4.4.1.5 Overall Metric

Denote the last m camera poses by $\mathcal{S}_c^k = \{\mathrm{R}_i, \mathrm{t}_i | i = k-m+1, \cdots, k\}$, and the last m key frames by $\mathcal{I}^k = \{I_i | i = k-m+1, \cdots, k\}$. The key points to be refined in LBA are those that are observed in at least one frame of $\mathcal{I}^k$, and we denote them by $\mathcal{S}_p^k$. Similarly we define $\mathcal{S}_l^k$ and $\mathcal{S}_v^k$ for ideal lines and vanishing points, respectively. The primary planes to be refined are those that have edges connected to key points from $\mathcal{S}_p^k$ or ideal lines from $\mathcal{S}_l^k$, and we denote them by $\mathcal{S}_\pi^k$.

The cost function of LBA is defined as a weighted sum of the costs of MFG features/constraints,

$$\mathcal{C}_{\text{LBA}}(\mathcal{M}_k) = \sum_{\kappa=k-w+1}^{k} \left[\eta_p \sum_{\mathbf{P}\in\mathcal{S}_p^k} \mathcal{K}_{\delta_p}\big(\mathcal{C}_p(\mathbf{P},\kappa)\big) + \eta_l \sum_{\mathbf{L}\in\mathcal{S}_l^k} \mathcal{K}_{\delta_l}\big(\mathcal{C}_l(\mathbf{L},\kappa)\big) + \eta_v \sum_{\mathbf{V}\in\mathcal{S}_v^k} \mathcal{K}_{\delta_v}\big(\mathcal{C}_v(\mathbf{V},\kappa)\big) \right] \tag{4.15}$$

$$+ \eta_\pi \sum_{\mathbf{\Pi}\in\mathcal{S}_\pi^k} \left[\sum_{\mathbf{P}\in\mathcal{S}_p^k} \mathcal{K}_{\delta_\pi}\big(\mathcal{C}_\pi(\mathbf{P},\mathbf{\Pi})\big) + \sum_{\mathbf{L}\in\mathcal{S}_l^k} \mathcal{K}_{\delta_\pi}\big(\mathcal{C}_\pi(\mathbf{L},\mathbf{\Pi})\big) \right]$$

where η_p, η_l, η_v and η_π are the weights for key points, ideal lines, vanishing points and primary planes, respectively, and $\mathcal{K}_\delta(\cdot)$ is a robust kernel function with parameter δ. Currently the weights are chosen empirically, and the kernel function is the Huber loss defined as

$$\mathcal{K}_\delta(e^2) = \begin{cases} e^2 & \text{for } |e| < \delta, \\ 2\delta e - \delta^2 & \text{otherwise.} \end{cases} \tag{4.16}$$

The LBA problem at time k is

$$\min_{\mathcal{S}_c^k,\mathcal{S}_p^k,\mathcal{S}_l^k,\mathcal{S}_v^k,\mathcal{S}_\pi^k} \mathcal{C}_{\text{LBA}}(\mathcal{M}_k). \tag{4.17}$$

This problem is solved using the Levenberg-Marquardt (LM) algorithm [45], and the solution is used to refine camera poses $\mathcal{S}_c^k$ and $\mathcal{M}_k$ nodes including key points $\mathcal{S}_p^k$, ideal lines $\mathcal{S}_l^k$, vanishing points $\mathcal{S}_v^k$ and primary planes $\mathcal{S}_\pi^k$.

4.4.2 MFG Pruning

False data association inevitably results in erroneous estimation in the 3D MFG. We thus constantly prune the MFG after performing LBA. We start with key point pruning.

Recall that $\mathbf{p}_{i(k)}$ represents the observation of $\mathbf{P}_i$ in I_k. Here we define a set $\mathcal{S}_{ob}(\mathbf{P}_i) = \{\mathbf{p}_{i(\kappa)}, \kappa \geq 0\}$ to contain all the detected observations for $\mathbf{P}_i$. The key point pruning procedure is summarized in Algorithm 8. The basic idea is that a key point outlier resulted from false matching must have observations inconsistent with its reprojections. After removing inconsistent observations, if a key point has very few surviving observations, then we consider it as mis-estimated.

The ideal line pruning procedure is similar to Algorithm 8 except that $e_{i\kappa} = \mathcal{C}_l(\mathbf{L}_i, \kappa)$. Ideal line mis-estimation results not only from false line matching between images but also from wrong association with vanishing points. In the latter case, the true direction of a 3D line is not parallel to the assigned vanishing point. Since we enforce the estimated line direction to be parallel to the vanishing point (see Section 4.4.1.3), the line reprojections must have discrepancies with observations. This allows the pruning algorithm to detect this kind of mis-estimation.

At current stage, we do not prune vanishing points and primary planes because 1) they are rarely mis-estimated, and 2) the Huber loss functions in (4.15) allow the LBA to be robust to such mis-estimations.

Algorithm 4.1: Key Point Pruning

```
  Input  : $\mathcal{V}$, $\mathcal{M}_k$, $\mathrm{R}_{0:k}$, $\mathrm{t}_{0:k}$
  Output: $\mathcal{M}_k$
1 for $\mathbf{P}_i \in \mathcal{S}_p^k$ do
2     for $\mathbf{p}_{i(\kappa)} \in \mathcal{S}_{ob}(\mathbf{P}_i)$ do
3         Compute $e_{i\kappa} = \mathcal{C}_p(\mathbf{P}_i, \kappa)$ using (4.10);
4         if $e_{i\kappa} > \varepsilon$ then
5             Remove $\mathbf{p}_{i(\kappa)}$ from $\mathcal{S}_{ob}(\mathbf{P}_i)$;
6     if $|\mathcal{S}_{ob}(\mathbf{P}_i)| < N_{ob}$ then
7         Remove $\mathbf{P}_i$ and the associated edges from $\mathcal{M}_k$;
8 return $\mathcal{M}_k$
```

4.5 Algorithms

The HLVN algorithm is summarized in Algorithm 18 to facilitate our analysis. Let the image resolution of I_k be r pixels. Then detecting 2D key points and line segments can be done in $O(r)$ time [44, 73]. Suppose on average we detect n_p 2D key points, n_s 2D line segments, n_l 2D ideal lines, and n_v 2D vanishing points in each image. Obviously, $r > n_p$, $r > n_s \geq n_l > n_v$. Usually, n_v is very small and can be considered as constant. Primary plane update takes $O(1)$ time because $|\mathcal{C}|$ is bounded by constant (see Section 4.3.4.4). MFG pruning takes $O(n_T(|\mathcal{S}_p^k| + |\mathcal{S}_l^k|))$ time, where n_T denotes the average number of observations for a 3D key point (or ideal line). In general visual navigation, we can bound n_T by a large constant. Thus, MFG pruning has a time complexity $O(w(n_p+n_s))$ because $|\mathcal{S}_p^k| < wn_p$ and $|\mathcal{S}_l^k| < wn_s$.

The most computationally-expensive step is LBA, which refines a parameter vector of dimension

$$d_p = 3|\mathcal{S}_p^k| + 6|\mathcal{S}_l^k| + 3|\mathcal{S}_v^k| + 4|\mathcal{S}_\pi^k| + 7|\mathcal{S}_c^k|.$$

This is because we use a 3-vector for a 3D key point, a 6-vector for a 3D ideal line, a 3-vector for a vanishing point, a 4-vector for a primary plane and a 7-vector (unit quaternion for rotation) for a camera pose. Since $|\mathcal{S}_v^k| < wn_v$, $|\mathcal{S}_\pi^k| < |\mathcal{S}_p^k|$ and $|\mathcal{S}_c^k| < w$, we have $d_p = O(w(n_p+n_s))$. Similarly, the observation vector's dimension is $d_o = O(w(n_p+n_s))$. In each iteration of LM, the computational complexity is $O(w^3(n_p + n_s)^3)$ for a dense matrix solver. According to [82], the total iterations needed by LM is upper-bounded by $O(1/\epsilon^2)$, with a stopping criterion $\|\nabla\mathcal{C}_{\text{LBA}}\| \leq \epsilon$.

Theorem 1. *The computational complexity of the HLVN algorithm is* $O(r + w^3(n_p + n_s)^3/\epsilon^2)$.

Algorithm 4.2: HLVN Algorithm

```
    Input  : $\mathcal{V}$, $\mathcal{M}_{k-1}$, $\mathrm{R}_{0:k-1}, \mathrm{t}_{0:k-1}$
    Output: $\mathcal{M}_k$, $\mathrm{R}_{0:k}, \mathrm{t}_{0:k}$
 1  Select key frame $I_k$;                                   $O(r + n_p^2)$
 2  Detect 2D key points from $I_k$;                          $O(r)$
 3  Detect 2D line segments from $I_k$;                       $O(r)$
 4  Detect 2D ideal lines for $I_k$;                          $O(n_s^2)$
 5  Detect vanishing points for $I_k$;                        $O(n_s^2)$
 6  Match key points between $I_{k-1}$ and $I_k$;             $O(n_p^2)$
 7  Compute epipolar geometry;                                $O(n_p)$
 8  Match vanishing points between $I_{k-1}$ and $I_k$;       $O(1)$
 9  Match ideal lines between $I_{k-1}$ and $I_k$;            $O(n_s^2)$
10  Estimate camera pose $\mathrm{R}_k, \mathrm{t}_k$;        $O(n_p^2)$
11  $\mathcal{M}_k = \mathcal{M}_{k-1}$;                      $O(1)$
12  Update 3D key points;                                     $O(n_p)$
13  Update 3D ideal lines;                                    $O(n_s)$
14  Update 3D vanishing points;                               $O(1)$
15  Update 3D primary planes;                                 $O(1)$
16  Perform LBA on $\mathcal{M}_k$;                           $O(w^3(n_p + n_s)^3/\epsilon^2)$
17  Prune $\mathcal{M}_k$;                                    $O(w(n_p + n_s))$
18  return $\mathcal{M}_k$, $\mathrm{R}_{0:k}, \mathrm{t}_{0:k}$
```

4.6 Experiments

We have implemented our algorithm using C++ [83]. We first validate the proposed two stage line matching (TSLM) approach on real image data. Then we evaluate the visual odometry performance of HLVN under both indoor and outdoor scenarios and compare it with state-of-the-art algorithms.

4.6.1 Line Matching Test

Ideal lines play a pivotal role in MFG. A good matching algorithm for ideal lines should find as many matches as possible while maintaining high accuracy. Here we validate our TSLM approach by comparing it with PCLM, the state-of-the-art in line matching.

We have collected 20 pairs of images covering a variety of man-made scenes (available

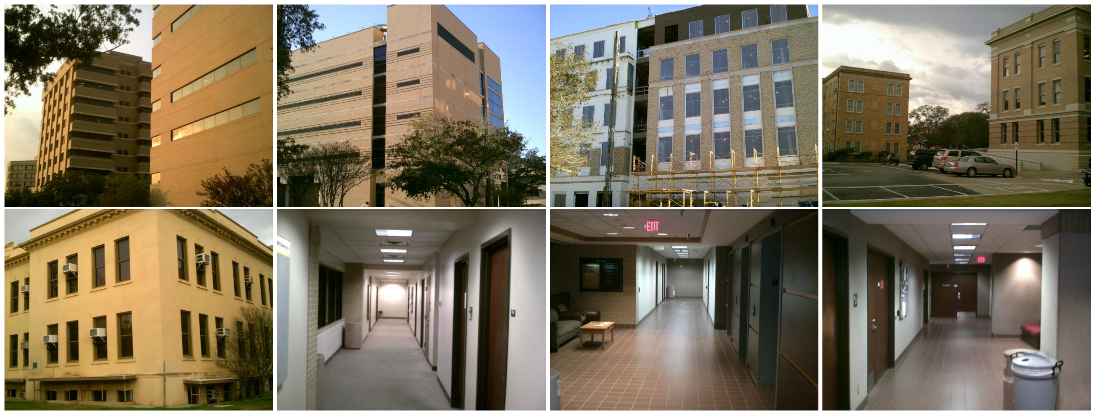


Figure 4.8: Sample images used for line matching test.

on line [83]). Due to the wide baselines in the image data, SIFT matches are utilized as point correspondences for both line matching methods. Table 4.1 shows the number of total matches (TM) and the true positive rate (TPR) obtained by TSLM and PCLM, respectively. On average, TSLM is able to find 62.5% more line matches than PCLM while achieving a TPR of 92.3%. The significant increase in line match number can greatly benefit the MFG construction process. The slight decrease in TPR is not a big problem because false matches are handled by other procedures such as MFG pruning.

4.6.2 Visual Odometry Test

We now focus on the visual odometry performance of our HLVN algorithm. For comparison, we have chosen the following two state-of-the-art algorithms in feature-based monocular visual odometry/SLAM.

- PTAM: Parallel Tracking and Mapping [71], one of the most successful BA based visual SLAM algorithms, and

- 1-Point-EKF: 1-Point RANSAC-based EKF-SLAM [60], a representative sequential filtering based visual odometry method.

Table 4.1: IDEAL LINE MATCHING RESULTS

No.	PCLM		TSLM	
	#TM	TPR	#TM	TPR
1	48	97.9%	93	97.8%
2	84	96.4%	125	88.8 %
3	54	92.6%	73	94.5%
4	51	86.2%	62	87.1%
5	83	90.4%	125	89.6%
6	58	96.5%	87	94.3%
7	62	93.5%	82	90.2%
8	74	89.2%	110	90.9 %
9	50	94.0%	84	96.4%
10	62	98.3%	86	97.7%
11	29	96.5%	51	94.1%
12	55	98.1%	76	97.4%
13	42	95.2%	87	96.6%
14	79	97.4%	104	95.2%
15	22	86.3%	80	76.3%
16	35	97.1%	74	87.8%
17	29	100%	56	91.1%
18	13	84.6%	28	82.1%
19	25	100%	68	97.1%
20	6	100%	19	89.5%
Avg.	48	93.7%	78	92.3%

Both algorithms above provide open-source code, allowing more fair comparisons. Although system parameter settings largely depend on real applications, we list the relevant parameter values used in our experiments in Table 4.2.

*4.6.2.1 **Evaluation Metric***

To evaluate the localization accuracy, we adopt the widely used absolute trajectory error (ATE) [60] as explained below.

Since the ground truth and the estimation of camera poses are usually represented in different coordinate systems, we need to align them before computing ATE. Let $\mathrm{g}_k^{W'}$ be

Table 4.2: PARAMETER SETTINGS

Parameter	Value	Description
N_2	50	2D point match number
N_3	7	3D key point number
τ_R	15°	rotation angle
τ_θ	10°	vanishing point angle
τ_ρ	0.9°	parallax threshold
N_{cp}	100	coplanar feature number
w	10	LBA window size
m	8	LBA pose number
η_p	1	key point weight
η_l	1	ideal line weight
η_v	15	vanishing point weight
η_π	100	primary plane weight
δ_p	1	Huber kernel size
δ_l	3	Huber kernel size
δ_v	1	Huber kernel size
δ_π	1	Huber kernel size
ε	4	reprojection error
N_{ob}	2	observation number

the ground truth of camera position at time k in a coordinate system $\{W'\}$ and $\mathbf{r}_k^W$ the estimated one in $\{W\}$. We need to find a similarity transformation that maps $\mathbf{r}_k^W$ to $\{W'\}$:

$$\mathbf{r}_k^{W'} := s\mathbf{R}_W^{W'}\mathbf{r}_k^W + \mathbf{t}_W^{W'}, \tag{4.18}$$

where the transformation is defined by rotation matrix $\mathbf{R}_W^{W'}$, translation vector $\mathbf{t}_W^{W'}$ and scaling factor s. The similarity transformation is obtained by minimizing $\sum_k \|\mathbf{r}_k^{W'} - \mathbf{g}_k^{W'}\|^2$. The ATE ε_k at time k is then defined as: $\varepsilon_k = \|\mathbf{r}_k^{W'} - \mathbf{g}_k^{W'}\|$. We compute the root-mean-square-error (RMSE) of ATE's over all the time indexes, as well as the standard deviation (SD), maximum (Max), and the ratio between RMSE and the trajectory length for comparison purpose.

4.6.2.2 ***Datasets***

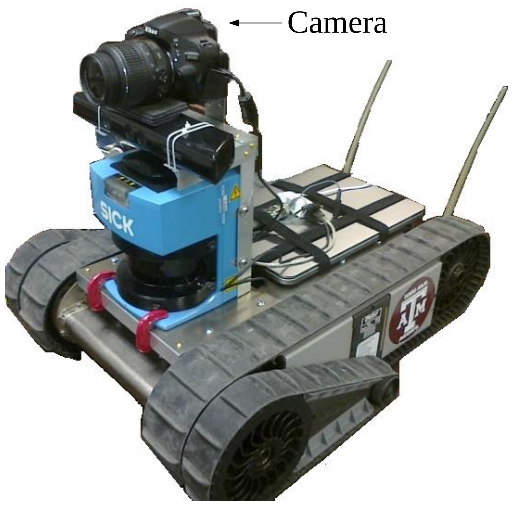
We evaluate the aforementioned three methods on two indoor and one outdoor sequence, as described below.

- Bicocca sequence: a subsequence of over 4,500 images from the Bicocca-2009-02-25b session of the Rawseeds datasets [84]. The images are recorded in a library using a camera with 79° HFOV at a resolution of 320×240. The trajectory has an approximate length of 77 m, with ground truth provided.

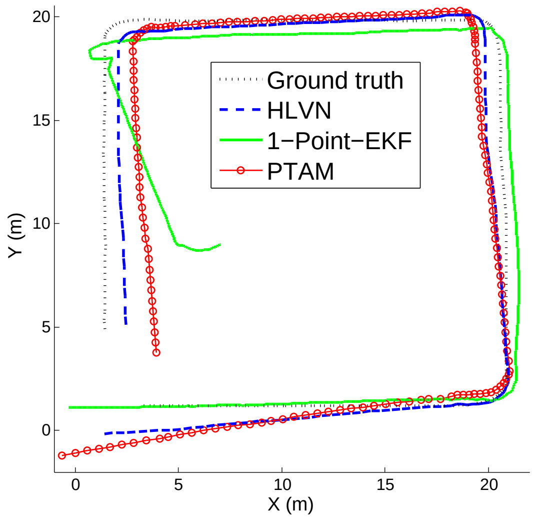
- HRBB4 sequence: a sequence of 12,000 images collected in an office corridor environment by ourselves (available on line [83]). The images are recorded using a camera (with 65° HFOV) mounted on a PackBot (see Fig. 4.9(a)). In our experiment, we reduce the image resolution from 1920×1080 to 640×360 for faster computation. The robot trajectory has an approximate length of 70 m as illustrated in Fig. 4.9(b).

  The ground truth of camera poses is obtained by using artificial landmarks - markers posted along the lower parts of walls. The 3D positions of the 4 outmost corners of each marker are manually measured, and their projections in images are manually selected. Based on these 3D-2D point correspondences, the camera pose of each frame is recovered using a PnP (perspective-n-point) solver (e.g. [85]). According to our test, this method achieves an accuracy of ±1 cm in camera position, owing to the following facts.

  - The marker locations are designed such that the camera clearly sees at least 3 or 4 markers most of the time.
  - The marker corners' 3D positions are carefully measured using a BOSCH GLR225 laser distance measurer with a range up to 70 m and measurement



(a) PackBot



(b) Trajectories

Figure 4.9: HRBB4 sequence. (a) Camera and robot. (b) Trajectory estimates aligned with the ground truth using similarity transforms as described in (4.18).

accuracy of ±1.5 mm.

– High resolution images (i.e. 1920×1080) are used to suppress image noise when finding the projections of marker corners.

Due to their minimal sizes in images, these markers do not bring much influence to the approaches in our test.

- Malaga6 sequence: a sequence of over 4,600 urban images from the 6th section of the Málaga Stereo and Laser Urban Data Set [86]. Although the original dataset provides stereo images, we only use the left channel at a resolution of 800×600. The trajectory length is over 1,200 m with GPS data available.

4.6.2.3 ***ATE***

We have fine-tuned parameters for PTAM and 1-Point-EKF with best efforts. The results below represent their best performance in our test.

Table 4.3: ABSOLUTE TRAJECTORY ERRORS

(a) BICOCCA

Method	RMSE (m)	SD (m)	Max (m)	RMSE/trajectory
1-Point-EKF	2.23	1.10	5.20	2.90%
PTAM	0.93	0.35	1.97	1.21%
HLVN	0.88	0.28	1.86	1.14%

(b) HRBB4

Method	RMSE (m)	SD (m)	Max (m)	RMSE/trajectory
1-Point-EKF	1.99	1.17	6.94	2.84%
PTAM	1.61	0.87	4.65	2.30%
HLVN	0.85	0.41	2.45	1.21%

(c) MALAGA6

Method	RMSE (m)	SD (m)	Max (m)	RMSE/trajectory
1-Point-EKF	77.16	44.68	175.98	6.43%
PTAM	—	—	—	—
HLVN	14.10	6.23	45.03	1.18%

Table 4.3(a) shows the ATE's on Bicocca for each method. Both PTAM and HLVN outperform 1-Point-EKF, which manifests the superiority of BA approaches over filtering. On the other hand, the difference between PTAM and HLVN is almost negligible. This is because Bicocca is recorded in an environment with rich texture (on average, 478 corner points detected per image), and the camera HFOV is relatively wide. This favors key point-based approaches like PTAM, whereas not allowing HLVN to demonstrate much advantage.

In contrast HRBB4 is a much more challenging dataset. Despite its larger image resolution than Bicocca, HRBB4 only detects 355 corner points per image on average due to the textureless scene. The robot also makes sharp turns (i.e., small translation along with large rotation) in HRBB4, which easily leads to large scale drift. The narrower HFOV in HRBB4 further increases the difficulty. Nonetheless, Table 4.3(b) shows that HLVN outperforms both competing approaches with an RMSE of 0.85 m, 1.21% of the overall trajectory length. Specifically, the RMSE of HLVN is 47.2% less than that of PTAM. As shown in Fig. 4.9(b), HLVN suffers less scale drift at sharp turns than PTAM and 1-Point-EKF by leveraging more types of features.

Designed for small-scale indoor use, PTAM is not quite applicable to Malaga6. In fact, we are not even able to have PTAM completely process Malaga6 because of tracking failure. In Table 4.3(c), HLVN outperforms 1-Point-EKF again by exploiting heterogeneous features and LBA. This demonstrates the benefit of HLVN for visual navigation in urban environments.

4.6.2.4 ***Feature Contributions***

Fig. 4.10 illustrates the values of each component in (4.15) over key frames of the Bicocca and HRBB4 sequences, respectively. We see how much each type of feature contributes to the LBA problem dynamically. As a result of the rich texture in Bicocca, the contribution of key points dominates all other costs throughout the sequence. On the other hand, the contributions of key points and ideal lines are mostly comparable to each other in HRBB4. This phenomenon shows that HLVN is adaptive in the sense that the contribution of each type of feature varies as the scene structure changes.

For a further insight of feature contributions, we investigate the performance of our system under different combinations of feature types using the following variants of HLVN:

- PT: using only key points,
- PT+LN: using key points and ideal lines,
- PT+VP: using key points and vanishing points,
- PT+PL: using key points and primary planes,
- PT+LN+VP: using key points, ideal lines and vanishing points,

- PT+LN+PL: using key points, ideal lines and primary planes,
- PT+VP+PL: using key points, vanishing points and primary planes.

Table 4.4 shows the ATE's resulted from these HLVN variants on the three datasets. We find that the inclusion of more feature types helps reduce the ATE's, though the improvement may vary in different scenarios. For example, the error reduction brought by more feature types is less significant on Bicocca than on HRBB4 or Malaga6; this conforms to the observation in Fig. 4.10(a) that key points dominate the overall cost of LBA throughout Bicocca. Unfortunately it is hard to judge the relative importance between individual feature types in general since it is essentially a scene-dependent problem. Nonetheless, the result implies that exploiting more features types and geometric constraints, whenever available, effectively reduces the overall estimation error. This justifies our choice of fusing heterogeneous landmarks for visual navigation in man-made environments, despite higher computational demands.

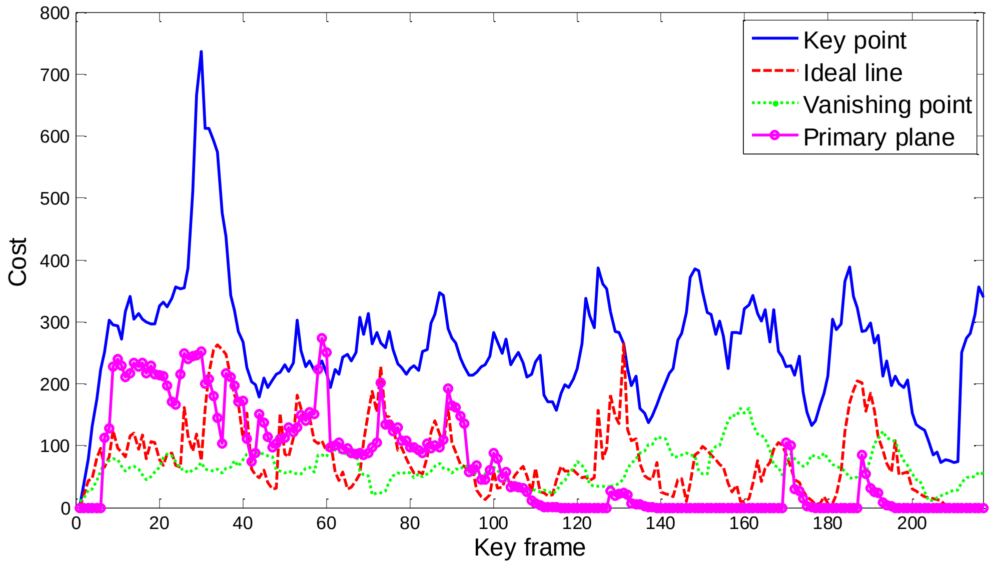
4.6.2.5 ***Time Consumption***

The LBA of HLVN is implemented using g^{2}o (general graph optimization) [87], which allows to leverage sparse optimization solvers. Our current implementation of HLVN

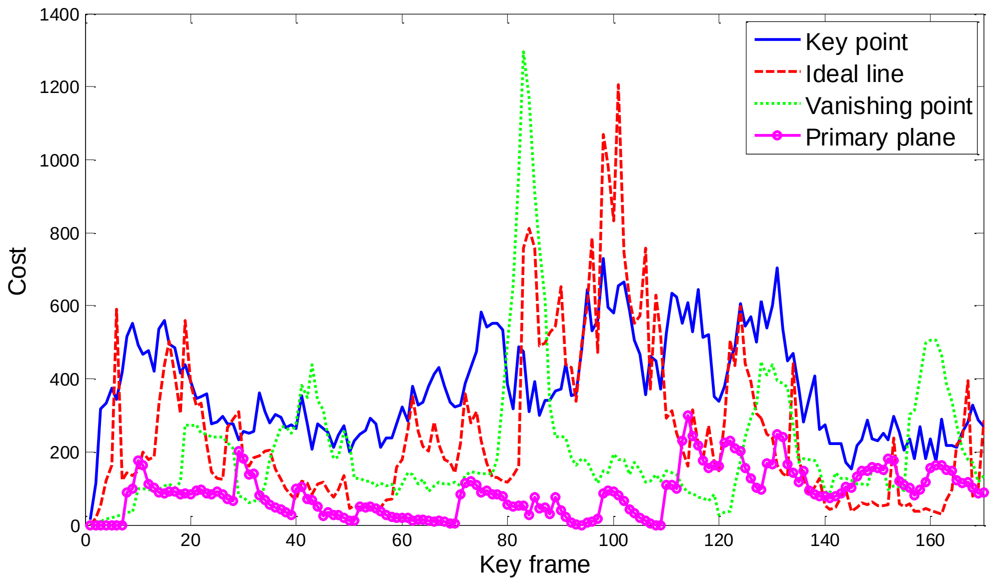
Table 4.4: ATE'S (M) OF HLVN VARIANTS

Variant	Bicocca	HRBB4	Malaga6
PT	1.04	2.05	39.68
PT+LN	0.98	1.42	25.24
PT+VP	1.01	1.56	22.08
PT+PL	0.96	1.62	25.71
PT+LN+VP	0.95	1.33	17.85
PT+LN+PL	0.91	1.17	19.37
PT+VP+PL	0.92	1.26	19.52
HLVN	0.88	0.85	14.10

The ATE's in the Malaga6 column are larger than other columns because Malaga6 is an outdoor sequence with a longer trajectory.



(a) Bicocca



(b) HRBB4

Figure 4.10: Contributions to LBA costs by different features. The costs are dimensionless.

is single-threaded and not yet optimized. The computation time on a desktop with an Intel Core i7-3770 CPU is reported in Table 4.5. Although relatively slow, HLVN can be accelerated for real time use in at least 3 ways, i.e., optimizing the implementation, using graphics processing units or more powerful CPUs, and parallelizing the algorithm. We expect that the algorithm can run in real time in the near future.

4.6.3 Test on KITTI Odometry Dataset

The KITTI odometry dataset [88] contains 22 image sequences, 11 of which (i.e. sequences 00-10) are provided with ground truth and thus used for our test. For general autonomous driving testing, this dataset covers various scenarios including urban, countryside and highway roads. In our experiment, however, only sequences 00 and 07 are of particular interest because they have rectilinear buildings dominating the scene, which conforms to our assumption **a.1**. Due to the lack of feature heterogeneity on other sequences, our method is not expected to outperform other approaches. For comparison, we choose the following point-based methods

- VISO2-M: the monocular visual odometry algorithm associated with the dataset [89], and
- SCG: a state-of-the-art large-scale monocular system proposed by Song, Chandraker and Guest [90], referred to as SCG here. SCG is a top-ranked monocular algorithm on the KITTI odometry benchmark.

Table 4.5: RUN TIME OF HLVN

Sequence	Duration	Run time	n_p	n_s	#Key frame
Bicocca	153 s	290 s	478	192	218
HRBB4	400 s	210 s	355	250	170
Malaga6	231 s	600 s	518	413	267

As defined in Section 4.5, n_p and n_s are the average numbers of 2D key points and line segments detected from each image, respectively. Duration means video length, and run time means computation time.

The evaluation metric provided by the dataset (see [88] for detail) requires estimated trajectories to be in real-world scale. Therefore, we have augmented our system with a ground plane detection component to remove scale ambiguity by assuming a fixed camera height. Similar to [89, 90], in each iteration our algorithm finds point correspondences within a pre-defined image region between key frames, reconstructs 3D points by triangulation, and detects ground plane from these points using RANSAC.

As highlighted in Table 4.6, on sequences 00 and 07 our method achieves clearly smaller translational errors than SCG, and rotational errors of the same level. To be specific, our method reduces the respective translational errors on sequences 00 and 07 by at least 52.5%. As expected, our method outperforms the counterparts on these two sequences by exploiting heterogeneous landmarks and their geometric relationships. Fig. 4.11 illustrates the estimated trajectories for sequences 00 and 07 by our method. Meanwhile, on other sequences our method yields comparable translational errors with SCG, despite slightly increased rotational errors. In fact, our translational errors on sequences 05 and 06 are also substantially lower than SCG thanks to the sporadic presence of rectilinear structures in the imagery. Sequence 01 is not included in Table 4.6 because its fast speed (up to 90 km/h) fails the ground plane detection for all three methods.

To summarize, our method significantly outperforms the counterpart in urban scenarios (e.g. sequences 00 and 07) where rectilinear buildings dominate the scene. Importantly, this is exactly the scenario where visual SLAM is of great importance due to the fact that GPS signals are often blocked or reflected by tall buildings.

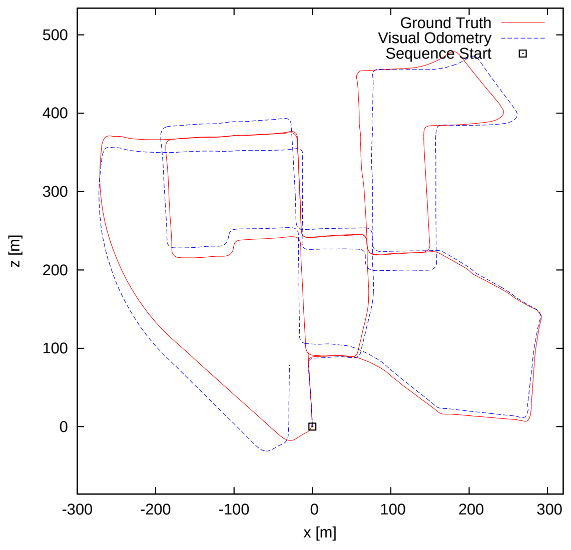
4.7 Conclusions

We presented a method utilizing heterogeneous visual features and their inner geometric constraints to assist robot navigation in man-made environments. This was managed by a multilayer feature graph. Our method extended the LBA framework by explicitly

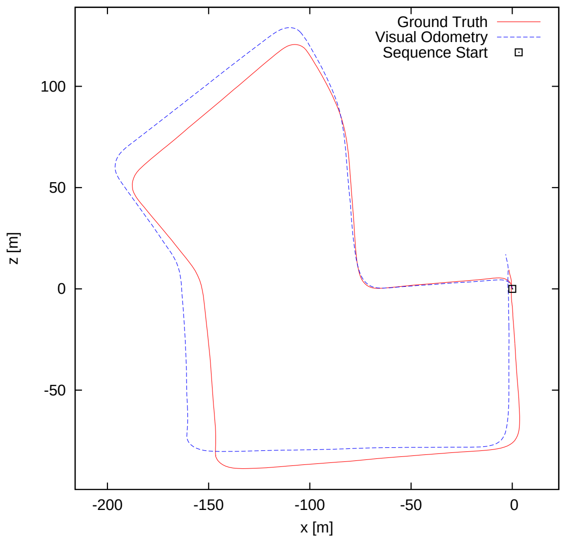
Table 4.6: COMPARISON ON KITTI DATASET

Seq	VISO2-M Rot (deg/m)	VISO2-M Trans (%)	SCG Rot (deg/m)	SCG Trans (%)	Ours Rot (deg/m)	Ours Trans (%)
00	0.0369	12.62	0.0142	7.14	0.0151	4.39
02	0.0194	3.71	0.0097	4.34	0.0122	5.60
03	0.0288	9.05	0.0093	2.90	0.0122	3.71
04	0.0163	7.58	0.0064	2.45	0.0088	2.74
05	0.0575	12.74	0.0107	8.13	0.0188	4.93
06	0.0275	3.71	0.0108	7.56	0.0181	4.09
07	0.1235	25.77	0.0234	9.92	0.0199	4.71
08	0.0369	16.88	0.0122	7.29	0.0171	6.69
09	0.0227	3.94	0.0096	5.14	0.0231	5.27
10	0.0596	29.36	0.0121	4.99	0.0119	4.43

Highlighted rows indicate urban sequences with rectilinear buildings dominating the scene, which allow our method to stand out by design.



(a) Sequence 00



(b) Sequence 07

Figure 4.11: Estimated trajectories by our method for sequences 00 and 07 in the KITTI dataset.

exploiting heterogeneous features and their geometric relationships in an unsupervised manner. The algorithm took a video stream as input, initialized and iteratively updated

MFG based on extracted key frames, and refined robot localization and MFG landmarks. We presented the algorithm pseudo code and analyzed its computation complexity. Physical experiments showed that our algorithm outperformed state-of-the-art approaches on datasets where rectilinear structures dominate the scene.

5. ROBUST RGB-D ODOMETRY USING POINT AND LINE FEATURES

For GPS-denied indoor environments, visual odometry is an attractive, low-cost alternative to laser-based robot localization approaches. The recent emergence of RGB-D cameras (e.g. Kinect) significantly enhances visual odometry performance by providing pixel-wise depth measurements. Besides the relative short range and the limited accuracy of existing RGB-D technologies, the main challenges come from *large lighting condition variations* and *uneven feature distributions*. The former directly hinders direct approaches (e.g. the dense method in [91]) which are based on the photo-consistency assumption. The latter often corrupts feature tracking quality in feature-based approaches.

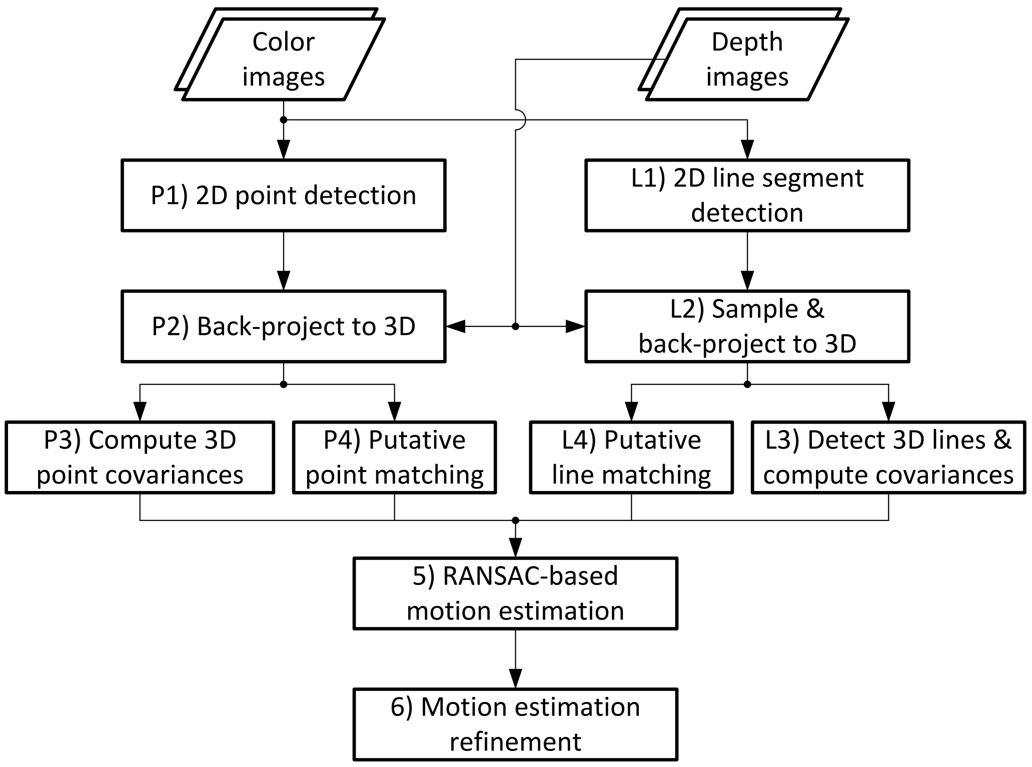

Color images
Depth images
P1) 2D point detection
L1) 2D line segment detection
P2) Back-project to 3D
L2) Sample & back-project to 3D
P3) Compute 3D point covariances
P4) Putative point matching
L4) Putative line matching
L3) Detect 3D lines & compute covariances
5) RANSAC-based motion estimation
6) Motion estimation refinement

Figure 5.1: System diagram.

Building on feature-based approaches and with the challenges in mind, we propose a robust RGB-D odometry method by fusing point and line features (see Figure 5.1). Line features are abundant indoors and less sensitive to lighting variation than points. On the other hand, points provide less position ambiguity than that of lines if sufficient observations are available. Effectively combining those desirable properties would increase both accuracy and robustness for an odometry method. We extract 3D points and lines from RGB-D data and analyze their measurement uncertainties. We provide a framework that seamlessly fuses points and lines by adopting a RANSAC-based motion estimation, followed by an MLE-based motion refinement. Under Gaussian noise assumption, we prove that fusing points and lines results in smaller uncertainty in motion estimation than using either feature type alone.

Our method has been evaluated on real-world data in experiments. We compare its performance with state-of-the-art methods including a keypoint-based approach and a dense visual odometry algorithm. Our method outperforms the counterparts under both constant and varying lighting conditions. Specifically, our method achieves an average translational error that is 34.9% smaller than the counterparts, when tested using public datasets.

5.1 Related Work

This work belongs to visual odometry, which estimates camera trajectories (or poses) from a sequence of images. Visual odometry is considered as a subproblem of visual SLAM.

Many visual odometry works have been developed using regular passive RGB cameras as the primary sensor, in monocular [60], stereo [92], or multi-camera settings. To improve accuracy, researchers study visual odometry from different perspectives. For example, Strasdat *et al.* [93] analyze two prevalent approaches to visual SLAM and find that bundle adjustment (e.g. [71]) produces more accurate results than sequential filtering (e.g. [94]). Due to depth ambiguity, monocular visual odometry inevitably suffers from scale drift, which can be easily avoided by using an RGB-D camera [95]. Besides accuracy, robustness is another critical issue but lags behind in visual odometry development. Lighting variation and uneven feature distribution are two main challenges for robustness.

Lighting variations caused by either natural or artificial lighting challenges both direct visual odometry and feature-based methods [96]. Although direct approaches can achieve superior accuracy by doing pixel-wise registration [28, 91, 97–99], their fundamental assumption on photo-consistency makes them sensitive to lighting condition changes. In the feature-based category, data-driven approaches are proposed to learn lighting-invariant descriptors [62] and matching functions [100] for robust matching of feature points. However, point features are also prone to illumination variations at the detection stage. On the other hand, the detection of edge and line features is less sensitive to lighting changes by nature. Edges [20], line segments [19,21] and lines [23] have been applied to visual odometry/SLAM, though their accuracy is usually not comparable with that of point features. In addition to regular approaches, RGB-D odometry can also utilize point could registration methods, which originate from Lidar-based SLAM. This kind of method [31] is invariant to lighting changes, but the problem is that it easily fails in degenerated cases, e.g. when a plane dominates the scene.

Meanwhile, uneven feature distribution hinders all feature-based visual odometry algorithms. In RGB-D odometry, points are the most popular type of visual feature. For example, in Henry *et al.*'s RGB-D mapping system [32], keypoints are extracted from RGB images and back-projected into 3D using depth data. Endres *et al.* [33] present an open-source RGB-D SLAM system based on point features. These approaches can be drastically affected if the distribution of point features is largely uneven, e.g. when textureless surfaces dominate the scene. To overcome this shortcoming, other types of features are investigated in RGB-D odometry. Points and planes are jointly utilized in

Taguchi *et al.*'s work [101], which uses any combination of three primitives of points and planes as a minimal set for initial pose estimation in RANSAC. Planes are adopted as the primary feature in [102] for visual odometry, and points are utilized only when the number of planes is insufficient. In [103] planes are employed as the only feature for SLAM. However, the applications of these methods are limited to plane-dominant environments. A 3D edge-based approach is proposed by Choi *et al.* [104], which treats 3D edges as an intelligently-downsampled version of point clouds and applies ICP for registration. However, this method does not take measurement uncertainties of 3D edges into account. Here we choose line features in combination with point features to improve the robustness of RGB-D odometry. We analyze measurement uncertainties of 3D features to maximize the accuracy.

5.2 Problem Description and System Overview

We assume the RGB-D camera to

a.1 be pre-calibrated, with lens distortion removed, and

a.2 have its depth image pixel-wisely synchronized with the corresponding color image.

Define an RGB-D frame at time k to be $F_k := \{I_k, D_k\}$, where I_k and D_k denote the color and depth images, respectively. The local coordinate system of F_k is the same as the RGB camera coordinate system (right-handed, Z-axis passing the camera center pointing forward, and X-axis pointing rightward). We define our problem as follows.

Problem 1. *Given an RGB-D sequence* $\{F_k\}, k \geq 1$, *estimate the camera pose of each frame with respect to a world coordinate system.*

To solve Problem 1, we estimate camera motion using adjacent RGB-D pairs, namely, F and F'. We compute a 3D rigid transformation between F and F'.

Our system (see Figure 5.1) mainly consists of feature detection and motion estimation.

In the feature detection stage, for each RGB-D frame we detect point and line features from the color image, back-project them to 3D, and analyze their uncertainties in parallel. In the motion estimation stage, we find feature matching between two RGB-D frames and estimate the relative motion using points and lines in a joint manner.

5.3 Feature Detection & Uncertainty Analysis

Errors inevitably enter the system at the feature detection stage. Eventually, the errors propagate to motion estimation results. For a deep understanding of the system accuracy, we begin with uncertainty analysis for each feature type.

5.3.1 Point Detection & Uncertainty Analysis

Detection. Given an RGB-D frame F, we first detect a set of 2D points from color image I using interest point detection algorithms such as SURF [18] (see Box P1, Figure 5.1). Then we find the depth values, if available, for these 2D points from D. Supposing a 2D point $\mathbf{p} = [u, v]^\mathsf{T}$ in I has depth d, its 3D position w.r.t. F is computed as follows (see Box P2, Figure 5.1)

$$\mathbf{P} := \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} (u - c_u)d/f_c \\ (v - c_v)d/f_c \\ d \end{bmatrix}, \tag{5.1}$$

where $[c_u, c_v]^T$ and f_c are the principal point and focal length of the RGB camera, respectively.

Uncertainty. As a function of $\left[\mathbf{p}^\mathsf{T}, d\right]^\mathsf{T}$, $\mathbf{P}$ has a measurement uncertainty depending on the error distribution of $\left[\mathbf{p}^\mathsf{T}, d\right]^\mathsf{T}$. The noise distribution of $\mathbf{p}$ is modeled as a zero-mean Gaussian with covariance $\sigma_p^2 \mathbf{I}_2$, where $\mathbf{I}_2$ is a 2×2 identity matrix. The measurement error of d is determined by many factors such as the imaging sensor, depth interpolation algorithm, and depth resolution. Taking the Kinect for example, it is commonly agreed that the depth noise is a quadratic function of the depth itself [105]. Specifically, the standard deviation (SD) σ_d of d is modeled as

$$\sigma_d = c_1 d^2 + c_2 d + c_3, \tag{5.2}$$

where c_1, c_2 and c_3 are constant coefficients. We set $c_1 = 2.73\times10^{-3}$, $c_2 = 7.4\times10^{-4}$, and $c_3 = -5.8\times10^{-4}$ in our experiments and the unit of d is meter [105].

Assuming the measurement noise of $\mathbf{p}$ is independent of that of d, we have

$$\mathrm{cov}\left(\begin{bmatrix}\mathbf{p}\\ d\end{bmatrix}\right) = \begin{bmatrix}\sigma_p^2\mathbf{I}_2 & \mathbf{0}_{2\times1}\\ \mathbf{0}_{1\times2} & \sigma_d^2\end{bmatrix}, \tag{5.3}$$

where $\mathrm{cov}(\cdot)$ indicates the covariance matrix of a random variable, and $\mathbf{0}_{m\times n}$ means a zero matrix of size $m \times n$. Under first-order approximation, we have (see Box P3, Figure 5.1)

$$\mathrm{cov}(\mathbf{P}) = J_P \mathrm{cov}\left(\begin{bmatrix}\mathbf{p}\\ d\end{bmatrix}\right) J_P^{\mathsf{T}}, \tag{5.4}$$

where $J_P = \frac{\partial \mathbf{P}}{\partial(\mathbf{p},d)} = \begin{bmatrix} d/f_c & 0 & (u_j - c_u)/f_c \\ 0 & d/f_c & (v_j - c_v)/f_c \\ 0 & 0 & 1 \end{bmatrix}.$

5.3.2 Line Detection & Uncertainty Analysis

In this section, we introduce a 3D line detection method by considering cues from both color and depth data. To handle RGB-D data noise, we also present how to optimally estimate the detected 3D lines and analyze the uncertainties of the estimates. Our method starts from 2D line detection.

5.3.2.1 Line Detection in 2D and 3D

Under the pinhole camera model, lines remain straight when projected from 3D to images. Therefore, to detect 3D lines we first detect their projections in the color image I (see Box L1, Figure 5.1). As long as a 3D line is visible in I, it appears as a 2D line segment. Here we employ the line segment detector LSD [44] to extract a set of 2D line segments $\mathcal{S}_{\mathrm{2D}} = \{\mathbf{s}_i | i = 1, 2, \cdots\}$ from I. Each line segment is represented by two endpoints $\mathbf{s}_i = \left[\mathbf{a}_i^{\mathsf{T}}, \mathbf{b}_i^{\mathsf{T}}\right]^{\mathsf{T}}$.

A naive way to obtain the 3D position of a 2D image line segment is to back-project its two endpoints to 3D using the depth map. However, this method does not work well in practice for two reasons: 1) *Depth corruption*: depth information is not always available, especially on object boundaries when the depth is discontinuous, and 2) *Perspective ambiguity* because a line segment in $\mathcal{S}_{\mathrm{2D}}$ does not necessarily correspond to a line segment in 3D as a result of the perspective projection. This ambiguity cannot be resolved by only checking the 3D positions of the two endpoints of the 2D line segment. This suggests that we should inspect more depth information of a 2D line segment to avoid the aforementioned issues. As a line segment consists of infinite number of points, we propose a sampling based method for 3D line detection.

Sampling. Given a 2D line segment $\mathbf{s}$, we sample n_s points evenly spaced on $\mathbf{s}$ as illustrated in Figure 5.2. In all experiments, we set $n_s = \min(100, \lfloor \|\mathbf{s}\| \rfloor)$, where $\|\mathbf{s}\|$ denotes the length of $\mathbf{s}$ (in pixels) and $\lfloor \cdot \rfloor$ is the floor function. We discard the sample points whose depths are unavailable, back-project the remaining points to 3D (see Box L2, Figure 5.1) using (5.1), and compute their 3D uncertainties using (5.4).

The 3D sample points obtained above are not necessarily from a 3D line, and even if they are, they may contain outliers due to large depth errors. As illustrated in Figure 5.2, we apply RANSAC to detect the existence of 3D line segments and filter out outliers

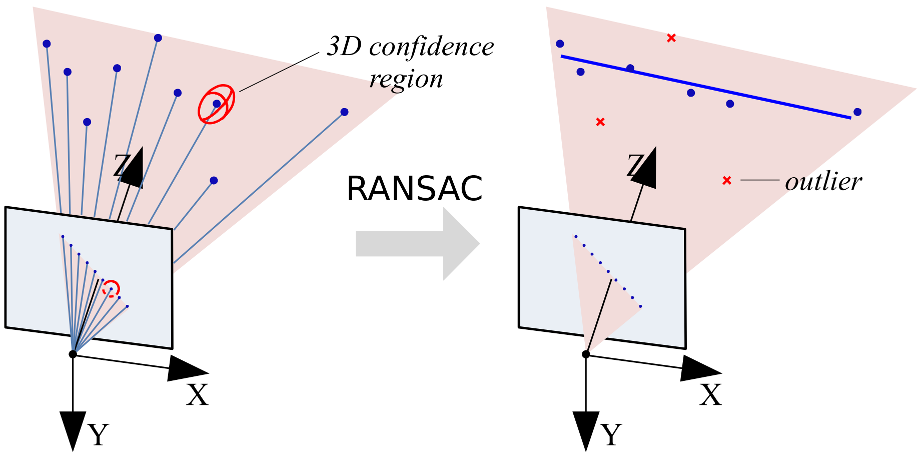


Figure 5.2: Sampling-based 3D line segment estimation. From a 2D line segment, n_s evenly-spaced points are sampled. The sample points are back-projected to 3D using depth information. Then a 3D line segment is fitted to these 3D points using RANSAC and Mahalanobis distance-based optimization.

(see Box L3, Figure 5.1). For brevity, we skip every detail of RANSAC but the error metric used for identifying inlier/outlier. Given a 3D line and a 3D point observation (subject to measurement noise), we utilize the Mahalanobis distance between them to evaluate whether the point is an observation of a point on the line. Mahalanobis distance is widely used in computer vision because it produces the optimal estimate under Gaussian assumptions [45]. For completeness, we briefly describe how to compute the Mahalanobis distance.

Mahalanobis distance. Let $\mathbf{P}$ be a 3D point measurement with covariance Σ_p and $\mathbf{L}$ be an infinite 3D line. The Mahalanobis distance from $\mathbf{P}$ to $\mathbf{L}$ is defined as

$$d_{\text{MAH}}(\mathbf{P}, \mathbf{L}) = \min_{\mathbf{Q} \in \mathbf{L}} \sqrt{(\mathbf{P} - \mathbf{Q})^\top \Sigma_p^{-1} (\mathbf{P} - \mathbf{Q})}, \tag{5.5}$$

where $\mathbf{Q} \in \mathbf{L}$ indicates an arbitrary point lying on line $\mathbf{L}$. To derive $d_{\text{MAH}}(\mathbf{P}, \mathbf{L})$, let $\mathbf{A}$ and $\mathbf{B}$ be two reference points on $\mathbf{L}$. Write $\mathbf{Q} = \mathbf{A} + \lambda(\mathbf{B} - \mathbf{A}), \lambda \in \mathbb{R}$. The minimization problem in (5.5) is then equivalent to minimizing the following univariate quadratic function.

$$\min_{\lambda} \begin{array}{c} \lambda^2(\mathbf{B}-\mathbf{A})^{\mathsf{T}}\Sigma_p^{-1}(\mathbf{B}-\mathbf{A}) + 2\lambda(\mathbf{B}-\mathbf{A})^{\mathsf{T}}\Sigma_p^{-1}(\mathbf{A}-\mathbf{P}) \\ +(\mathbf{A}-\mathbf{P})^{\mathsf{T}}\Sigma_p^{-1}(\mathbf{A}-\mathbf{P}) \end{array}$$

The optimal value of λ yields the optimal $\mathbf{Q}$ for (5.5) $\mathbf{Q}^* = \mathbf{A} - \frac{(\mathbf{B}-\mathbf{A})^{\mathsf{T}}\Sigma_p^{-1}(\mathbf{A}-\mathbf{P})}{(\mathbf{B}-\mathbf{A})^{\mathsf{T}}\Sigma_p^{-1}(\mathbf{B}-\mathbf{A})}(\mathbf{B}-\mathbf{A})$. Let

$$\boldsymbol{\delta}(\mathbf{P},\mathbf{L}) = \mathbf{P} - \mathbf{Q}^*, \tag{5.6}$$

and then we have,

$$d_{\text{MAH}}(\mathbf{P},\mathbf{L}) = \sqrt{\boldsymbol{\delta}(\mathbf{P},\mathbf{L})^{\mathsf{T}}\Sigma_p^{-1}\boldsymbol{\delta}(\mathbf{P},\mathbf{L})}. \tag{5.7}$$

5.3.2.2 Line Uncertainty under MLE

Suppose the size of the largest consensus set returned by the aforementioned RANSAC process is n_{con}. Recall that n_s points are originally sampled from the 2D line segment s. If n_{con}/n_s is below a threshold τ (0.6 in all experiments), it implies that we do not have sufficient depth information to retrieve the 3D position of the line segment s. If $n_{\text{con}}/n_s \geq \tau$, we proceed to apply MLE to obtain the 3D line segment.

Let the largest consensus set be $\{\mathbf{G}_i | i = 1, \cdots, n_{\text{con}}\}$ with $\mathbf{G}_1$ and $\mathbf{G}_{\text{con}}$ being the two extremities. We parametrize the 3D line segment $\mathbf{L} = \left[\mathbf{A}^{\mathsf{T}}, \mathbf{B}^{\mathsf{T}}\right]^{\mathsf{T}}$ to be estimated by two 3D points associated with $\mathbf{G}_1$ and $\mathbf{G}_{\text{con}}$. Define a measurement error function

$$w(\mathbf{L}) = \begin{bmatrix} \mathbf{G}_1 - \mathbf{A} \\ \boldsymbol{\delta}(\mathbf{G}_2, \mathbf{L}) \\ \vdots \\ \boldsymbol{\delta}\left(\mathbf{G}_{n_{\text{con}-1}}, \mathbf{L}\right) \\ \mathbf{G}_{n_{\text{con}}} - \mathbf{B} \end{bmatrix}, \tag{5.8}$$

where $\boldsymbol{\delta}(\cdot,\cdot)$ is defined in (5.6). The MLE of $\mathbf{L}$ is obtained as follows,

$$\mathbf{L}^* = \min_{\mathbf{L}} w(\mathbf{L})^\mathsf{T} \Sigma_w^{-1} w(\mathbf{L}), \tag{5.9}$$

where $\Sigma_w = diag\big(\text{cov}(\mathbf{G}_1), \cdots, \text{cov}\left(\mathbf{G}_{n_\text{con}}\right)\big)$ is a blockwise diagonal matrix. This problem is then solved using the Levenberg-Marquardt (LM) algorithm. From back-propagation of covariance [45], we obtain the covariance of the MLE (see Box L3, Figure 5.1)

$$\text{cov}(\mathbf{L}^*) = \left(J_w^\mathsf{T} \Sigma_w^{-1} J_w\right)^{-1}, \tag{5.10}$$

where $J_w = \frac{\partial w}{\partial \mathbf{L}}\Big|_{\mathbf{L}=\mathbf{L}^*}$.

5.4 Motion Estimation & Uncertainty Analysis

With point and line features detected and the understanding of their error covariance, we are ready to perform overall camera motion estimation for the adjacent frame pair and analyze the uncertainty of the estimation.

5.4.1 Putative Feature Matching and RANSAC-based Motion Estimation

Once 3D points and lines are detected from F and F', we need to find feature correspondences between frames. As both 3D points and lines have associated 2D points and lines, we do feature matching using 2D features (see Boxes P4 and L4, Figure 5.1). We compute the SURF descriptors [18] for points and the MSLD descriptors [106] for lines, respectively. We adopt nearest-neighbour method in descriptor space for the putative matching.

As putative matching inevitably contains false matching, we use RANSAC to filter out outliers and estimate the 3D rigid transformation (see Box 5, Figure 5.1). Up to this point, points and lines have been processed in parallel. However, in each iteration of RANSAC, a minimal set of data is randomly sampled for a motion estimate; a seamless fusion of points and lines should allow mixed features in a minimal set. For this purpose, we consider the four possible types of minimal sets as follows,

- **3 point** matches. This can be trivially solved using methods like [107, 108].
- **3 line** matches. In fact the 3D rigid transformation can be recovered by only 2 line matches using an SVD-based method [109]. Considering that this method is very sensitive to noise, we sample 3 line matches.
- **1 point + 2 line** matches. In each frame, we orthogonally project the point onto the 2 lines, respectively, which converts this case to a case of "3 point" matches.
- **2 point + 1 line** matches. We choose one point and orthogonally project it onto the line in each frame, converting this case to a case of "3 point" matches.

5.4.2 MLE of Motion and Uncertainty Analysis

The RANSAC process results in a largest consensus set of feature matches consisting of both point matches and line matches. We refine the motion estimate with the whole consensus set of point and line matches using MLE (see Box 6, Figure 5.1). We prove that the MLE of 3D rigid transformation obtained using both types of feature correspondences has smaller uncertainty than that obtained using either type of feature correspondence alone. We start by deriving the uncertainties for the MLE of motion obtained using points and lines separately before fusing them.

5.4.2.1 Point-based Motion Estimation

Let the set of 3D point correspondences between F and F' be $\{\mathbf{P}_i \leftrightarrow \mathbf{P}'_i, i = 1, \cdots, n\}$, where $n \geq 3$. Denote the covariance of $\mathbf{P}_i$ and $\mathbf{P}'_i$ by Σ_i and Σ'_i, respectively. The rigid body transformation $\mathbf{T}$ is parametrized by a six-vector $\boldsymbol{\xi}$. $\mathbf{T}$ can also represented by rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$,

$$\mathbf{T}(\mathbf{X}) := \mathbf{R}\mathbf{X} + \mathbf{t}, \tag{5.11}$$

where $\mathbf{X}$ is a 3D point.

To achieve an MLE of motion, the underlying 3D point landmarks must be estimated simultaneously. Let $\mathbf{X}_i$ be the 3D point with respect to F that underlies the measurements $\mathbf{P}_i$ and $\mathbf{P}'_i$. The parameter vector to be estimated is thus defined as $\mathbf{p} = \left[\boldsymbol{\xi}^\mathsf{T}, \mathbf{X}_1^\mathsf{T}, \cdots, \mathbf{X}_n^\mathsf{T}\right]^\mathsf{T}$.

Define a measurement error function

$$h(\mathbf{p}) = \begin{bmatrix} \mathbf{h}_\mathbf{p} \\ \mathbf{h}'_\mathbf{p} \end{bmatrix}, \tag{5.12}$$

where $\mathbf{h}_\mathbf{p} = \begin{bmatrix} \mathbf{X}_1 - \mathbf{P}_1 \\ \vdots \\ \mathbf{X}_n - \mathbf{P}_n \end{bmatrix}$ and $\mathbf{h}'_\mathbf{p} = \begin{bmatrix} \mathbf{T}(\mathbf{X}_1) - \mathbf{P}'_1 \\ \vdots \\ \mathbf{T}(\mathbf{X}_n) - \mathbf{P}'_n \end{bmatrix}$.

The MLE of motion solves the following problem

$$\min_{\mathbf{p}} h(\mathbf{p})^\mathsf{T} \Sigma_h^{-1} h(\mathbf{p}), \tag{5.13}$$

where $\Sigma_h = diag\left(\Sigma_1, \cdots, \Sigma_n, \Sigma'_1 \cdots, \Sigma'_n\right)$. Lemma **??** provides the estimation uncertainty of motion.

Lemma 3. *Under Gaussian noise assumption, the point feature-based MLE of rigid trans-*

formation $\boldsymbol{\xi}$ obtained by (5.13) has covariance

$$C_P = \left(H_h^A - H_h^B {H_h^D}^{-1} {H_h^B}^{\mathsf{T}}\right)^{-1}, \tag{5.14}$$

where

$$H_h^A = \sum_{i=1}^n \left(\frac{\partial \mathbf{T}(\mathbf{X}_i)}{\partial \boldsymbol{\xi}}\right)^{\mathsf{T}} \Sigma_i'^{-1} \frac{\partial \mathbf{T}(\mathbf{X}_i)}{\partial \boldsymbol{\xi}} \ ,$$

$$H_h^B = \begin{bmatrix} \left(\frac{\partial \mathbf{T}(\mathbf{X}_1)}{\partial \mathbf{X}_1}\right)^{\mathsf{T}} \Sigma_1'^{-1} \frac{\partial \mathbf{T}(\mathbf{X}_1)}{\partial \boldsymbol{\xi}} \\ \vdots \\ \left(\frac{\partial \mathbf{T}(\mathbf{X}_n)}{\partial \mathbf{X}_n}\right)^{\mathsf{T}} \Sigma_n'^{-1} \frac{\partial \mathbf{T}(\mathbf{X}_n)}{\partial \boldsymbol{\xi}} \end{bmatrix}^{\mathsf{T}}, \quad \textit{and}$$

$$H_h^D = diag\Bigg(\Sigma_1^{-1} + \left(\frac{\partial \mathbf{T}(\mathbf{X}_1)}{\partial \mathbf{X}_1}\right)^{\mathsf{T}} \Sigma_1'^{-1} \frac{\partial \mathbf{T}(\mathbf{X}_1)}{\partial \mathbf{X}_1} \ , \quad \cdots, \ \Sigma_n^{-1} + \left(\frac{\partial \mathbf{T}(\mathbf{X}_n)}{\partial \mathbf{X}_n}\right)^{\mathsf{T}} \Sigma_n'^{-1} \frac{\partial \mathbf{T}(\mathbf{X}_n)}{\partial \mathbf{X}_n} \Bigg).$$

Proof. By back-propagation of covariance, the MLE of $\mathbf{p}$ has covariance [45]

$$\mathrm{cov}\left(\mathbf{p}\right) = \left(J_h^{\mathsf{T}} \Sigma_h^{-1} J_h\right)^{-1}, \tag{5.15}$$

with

$$J_h = \frac{\partial h}{\partial \mathbf{p}} = \left[\begin{array}{c|c} J_h^A & J_h^B \\ \hline J_h^C & J_h^D \end{array}\right]$$

$$= \left[\begin{array}{c|ccc} \mathbf{0} & \mathbf{I}_3 & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_3 \\ \hline \frac{\partial \mathbf{T}(\mathbf{X}_1)}{\partial \boldsymbol{\xi}} & \frac{\partial \mathbf{T}(\mathbf{X}_1)}{\partial \mathbf{X}_1} & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \ddots & \mathbf{0} \\ \frac{\partial \mathbf{T}(\mathbf{X}_n)}{\partial \boldsymbol{\xi}} & \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{T}(\mathbf{X}_n)}{\partial \mathbf{X}_n} \end{array}\right]_{6n \times 3n+6}$$

where $\mathbf{I}_3$ is a 3×3 identity matrix, and $\mathbf{0}$ indicates a zero matrix of a context-determined size hereafter.

With $\Sigma_h^{-1} = diag\big(\Sigma_1^{-1}, \cdots, \Sigma_n^{-1}, \Sigma_1'^{-1} \cdots, \Sigma_n'^{-1}\big)$, we derive

$$J_h^{\mathsf{T}} \Sigma_h^{-1} J_h = \left[\begin{array}{c|c} H_h^A & H_h^B \\ \hline {H_h^B}^{\mathsf{T}} & H_h^D \end{array}\right].$$

Performing blockwise matrix inversion on $J_h^{\mathsf{T}} \Sigma_h^{-1} J_h$ yields (5.14). $\square$

5.4.2.2 Line-based Motion Estimation

Let the set of 3D line correspondences between F and F' be $\{\mathbf{L}_i \leftrightarrow \mathbf{L}_i', i = 1, \cdots, m\}$, where $m \geq 3$. Recall that $\mathbf{L}_i = \left[\mathbf{A}_i^{\mathsf{T}}, \mathbf{B}_i^{\mathsf{T}}\right]^{\mathsf{T}}$, $\mathbf{L}_i' = \left[\mathbf{A}_i'^{\mathsf{T}}, \mathbf{B}_i'^{\mathsf{T}}\right]^{\mathsf{T}}$. For simplicity, we denote $\Lambda_i = \text{cov}(\mathbf{L})$, $\Lambda_i' = \text{cov}(\mathbf{L}_i')$ (see (5.10)).

For MLE of motion, the underlying 3D line landmarks must be estimated simultaneously. Let $\mathbf{Y}_i = \left[\boldsymbol{\alpha}_i^{\mathsf{T}}, \boldsymbol{\beta}_i^{\mathsf{T}}\right]^{\mathsf{T}}$ be the 3D line with respect to F that underlies the measurements $\mathbf{L}_i$ and $\mathbf{L}'_i$. The parameter vector to be estimated is thus defined as $\mathbf{q} = [\boldsymbol{\xi}^\mathsf{T}, \mathbf{Y}_1^\mathsf{T}, \cdots, \mathbf{Y}_m^\mathsf{T}]^\mathsf{T}$.

Recall $\boldsymbol{\delta}(\mathbf{A}_i, \mathbf{Y})$ is a 3-vector function. Define

$$\boldsymbol{\eta}(\mathbf{L}_i, \mathbf{Y}_i) = \left[\boldsymbol{\delta}(\mathbf{A}_i, \mathbf{Y}_i)^\mathsf{T}, \boldsymbol{\delta}(\mathbf{B}_i, \mathbf{Y}_i)^\mathsf{T}\right]^\mathsf{T}. \tag{5.16}$$

Define a measurement error function

$$g(\mathbf{q}) = \begin{bmatrix} \mathbf{g}_l \\ \mathbf{g}'_l \end{bmatrix}, \tag{5.17}$$

where $\mathbf{g}_l = \begin{bmatrix} \boldsymbol{\eta}(\mathbf{L}_1, \mathbf{Y}_1) \\ \vdots \\ \boldsymbol{\eta}(\mathbf{L}_m, \mathbf{Y}_m) \end{bmatrix}$, $\mathbf{g}'_l = \begin{bmatrix} \boldsymbol{\eta}\left(\mathbf{L}'_1, \mathbf{T}(\mathbf{Y}_1)\right) \\ \vdots \\ \boldsymbol{\eta}\left(\mathbf{L}'_m, \mathbf{T}(\mathbf{Y}_m)\right) \end{bmatrix}$, and $\mathbf{T}(\mathbf{Y}_i) := \left[\mathbf{T}(\boldsymbol{\alpha}_i)^\mathsf{T}, \mathbf{T}(\boldsymbol{\beta}_i)^\mathsf{T}\right]^\mathsf{T}$.
The MLE solves the following problem

$$\min_{\mathbf{q}} g(\mathbf{q})^\mathsf{T} \Sigma_g^{-1} g(\mathbf{q}), \tag{5.18}$$

where $\Sigma_g = diag\left(\Lambda_1, \cdots, \Lambda_m, \Lambda'_1, \cdots, \Lambda'_m\right)$.

Lemma 4. *Under Gaussian noise assumption, the line feature-based MLE of rigid transformation $\boldsymbol{\xi}$ obtained by (5.18) has covariance*

$$C_L = \left(H_g^A - H_g^B H_g^{D^{-1}} H_g^{B^\mathsf{T}}\right)^{-1}, \tag{5.19}$$

where

$$\begin{aligned}
H_g^A = & \ \sum_i \left(\frac{\partial \boldsymbol{\eta}(\mathbf{L}_i', \mathbf{T}(\mathbf{Y}_i))}{\partial \boldsymbol{\xi}}\right)^{\mathsf{T}} \Lambda_i'^{-1} \frac{\partial \boldsymbol{\eta}(\mathbf{L}_i', \mathbf{T}(\mathbf{Y}_i))}{\partial \boldsymbol{\xi}} \ , \\
H_g^B = & \begin{bmatrix} \left(\frac{\partial \boldsymbol{\eta}(\mathbf{L}_1', \mathbf{T}(\mathbf{Y}_1))}{\partial \mathbf{Y}_1}\right)^{\mathsf{T}} \Lambda_1'^{-1} \frac{\partial \boldsymbol{\eta}(\mathbf{L}_1', \mathbf{T}(\mathbf{Y}_1))}{\partial \boldsymbol{\xi}} \\ \vdots \\ \left(\frac{\partial \boldsymbol{\eta}(\mathbf{L}_m', \mathbf{T}(\mathbf{Y}_m))}{\partial \mathbf{Y}_m}\right)^{\mathsf{T}} \Lambda_m'^{-1} \frac{\partial \boldsymbol{\eta}(\mathbf{L}_m', \mathbf{T}(\mathbf{Y}_m))}{\partial \boldsymbol{\xi}} \end{bmatrix}^{\mathsf{T}} , \\
H_g^D = & \ diag(\mathbf{C}_1, \cdots, \mathbf{C}_m), \ \textit{and for } i = 1, \cdots, m \\
\mathbf{C}_i = & \ \left(\frac{\partial \boldsymbol{\eta}(\mathbf{L}_i, \mathbf{Y}_i)}{\partial \mathbf{Y}_i}\right)^{\mathsf{T}} \Lambda_i^{-1} \frac{\partial \boldsymbol{\eta}(\mathbf{L}_i, \mathbf{Y}_i)}{\partial \mathbf{Y}_i} \\
& + \left(\frac{\partial \boldsymbol{\eta}(\mathbf{L}_i', \mathbf{T}(\mathbf{Y}_i))}{\partial \mathbf{Y}_i}\right)^{\mathsf{T}} \Lambda_i'^{-1} \frac{\partial \boldsymbol{\eta}(\mathbf{L}_i', \mathbf{T}(\mathbf{Y}_i))}{\mathbf{Y}_i} .
\end{aligned}$$

Proof. By back-propagation of covariance, the MLE of $\mathbf{q}$ has covariance [45]

$$\text{cov}(\mathbf{q}) = \left(J_g^{\mathsf{T}} \Sigma_g^{-1} J_g\right)^{-1}, \tag{5.20}$$

where

$$J_g = \frac{\partial g}{\partial \mathbf{q}} = \left[\begin{array}{c:c} J_g^A & J_g^B \\ \hdashline J_g^C & J_g^D \end{array}\right]_{12m \times 6 + 6m}$$

$$= \left[\begin{array}{c:ccc}
\mathbf{0} & \frac{\partial \boldsymbol{\eta}(\mathbf{L}_1, \mathbf{Y}_1)}{\partial \mathbf{Y}_1} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial \boldsymbol{\eta}(\mathbf{L}_m, \mathbf{Y}_m)}{\partial \mathbf{Y}_m} \\
\hdashline
\frac{\partial \boldsymbol{\eta}(\mathbf{L}_1', \mathbf{T}(\mathbf{Y}_1))}{\partial \boldsymbol{\xi}} & \frac{\partial \boldsymbol{\eta}(\mathbf{L}_1', \mathbf{T}(\mathbf{Y}_1))}{\partial \mathbf{Y}_1} & \mathbf{0} & \mathbf{0} \\
\vdots & \mathbf{0} & \ddots & \mathbf{0} \\
\frac{\partial \boldsymbol{\eta}(\mathbf{L}_n', \mathbf{T}(\mathbf{Y}_n))}{\partial \boldsymbol{\xi}} & \mathbf{0} & \mathbf{0} & \frac{\partial \boldsymbol{\eta}(\mathbf{L}_m', \mathbf{T}(\mathbf{Y}_m))}{\partial \mathbf{Y}_m}
\end{array}\right]$$

With $\Sigma_g^{-1} = diag\big(\Lambda_1^{-1}, \cdots, \Lambda_m^{-1}, \Lambda_1'^{-1} \cdots, \Lambda_m'^{-1}\big)$, we derive

$$J_g \Sigma_g^{-1} J_g = \left[\begin{array}{c:c} H_g^A & H_g^B \\ \hdashline H_g^{B\mathsf{T}} & H_g^D \end{array}\right].$$

Performing blockwise matrix inversion on $J_g \Sigma_g^{-1} J_g$ yields (5.19). $\square$

5.4.2.3 Motion Estimation Using Points and Lines

Now we are ready to fuse points and lines for the MLE of motion. Given $\{\mathbf{P}_i \leftrightarrow \mathbf{P}_i', i = 1, \cdots, n\}$ and $\{\mathbf{L}_i \leftrightarrow \mathbf{L}_i', i = 1, \cdots, m\}$, we formulate an MLE problem that jointly estimates the rigid motion, point landmarks and line landmarks. The parameter vector to be estimated is defined as

$$\mathbf{r} = \left[\boldsymbol{\xi}^\mathsf{T}, \mathbf{X}_1^\mathsf{T}, \cdots, \mathbf{X}_n^\mathsf{T}, \mathbf{Y}_1^\mathsf{T}, \cdots, \mathbf{Y}_m^\mathsf{T}\right]^\mathsf{T}.$$

Define a measurement error function

$$f(\mathbf{r}) = \begin{bmatrix} \mathbf{f_p} \\ \mathbf{f_l} \end{bmatrix}, \tag{5.21}$$

where $\mathbf{f_p} = \begin{bmatrix} \mathbf{X}_1 - \mathbf{P}_1 \\ \vdots \\ \mathbf{X}_n - \mathbf{P}_n \\ \mathbf{T}(\mathbf{X}_1) - \mathbf{P}_1' \\ \vdots \\ \mathbf{T}(\mathbf{X}_n) - \mathbf{P}_n' \end{bmatrix}$ and $\mathbf{f_l} = \begin{bmatrix} \boldsymbol{\eta}(\mathbf{L}_1, \mathbf{Y}_1) \\ \vdots \\ \boldsymbol{\eta}(\mathbf{L}_m, \mathbf{Y}_m) \\ \boldsymbol{\eta}(\mathbf{L}_1', \mathbf{T}(\mathbf{Y}_1)) \\ \vdots \\ \boldsymbol{\eta}(\mathbf{L}_m', \mathbf{T}(\mathbf{Y}_m)) \end{bmatrix}$. The MLE of $\mathbf{r}$ is obtained by solving the following problem

$$\min_{\mathbf{r}} f(\mathbf{r})^{\mathsf{T}} \Sigma_f^{-1} f(\mathbf{r}), \tag{5.22}$$

where $\Sigma_f = diag(\Sigma_h, \Sigma_g)$.

Lemma 5. *Under Gaussian noise assumption, the MLE of rigid transformation $\boldsymbol{\xi}$ based on both point and line features obtained by (5.22) has covariance*

$$C_H = \left(H_h^A + H_g^A - H_h^B {H_h^D}^{-1} {H_h^B}^{\mathsf{T}} - H_g^B {H_g^D}^{-1} {H_g^B}^{\mathsf{T}}\right)^{-1}. \tag{5.23}$$

Proof. By back-propagation of covariance, the MLE of $\mathbf{r}$ has covariance [45]

$$\text{cov}(\mathbf{r}) = \left(J_f^{\mathsf{T}} \Sigma_f^{-1} J_f\right)^{-1} \tag{5.24}$$

where $J_f = \frac{\partial f}{\partial \mathbf{r}} = \left[\begin{array}{ccc} J_h^A & J_h^B & \mathbf{0} \\ J_h^C & J_h^D & \mathbf{0} \\ \hline J_g^A & \mathbf{0} & J_g^B \\ J_g^C & \mathbf{0} & J_g^D \end{array}\right]$. With $\Sigma_f^{-1} = diag\left(\Sigma_h^{-1}, \Sigma_g^{-1}\right)$, we derive

$$J_f^{\mathsf{T}} \Sigma_f^{-1} J_f = \left[\begin{array}{c|cc} H_h^A + H_g^A & H_h^B & H_g^B \\ \hline {H_h^B}^{\mathsf{T}} & H_h^D & \mathbf{0} \\ {H_g^B}^{\mathsf{T}} & \mathbf{0} & H_g^D \end{array}\right].$$

Performing blockwise matrix inversion on $J_f^{\mathsf{T}} \Sigma_f^{-1} J_f$ yields (5.23).

□

With Lemmas 3, 4 and 5 introduced, we are ready to present the following theorem that justifies the benefit of fusing point and line features for RGB-D odometry.

Theorem 2. *Under Gaussian noise assumption, fusing points and lines produces smaller uncertainty in the MLE of pairwise motion than using each feature alone. Specifically, for the MLE covariances, C_P, C_L and C_H, obtained from using points, lines, and points plus lines respectively, we have*

$$\lambda_i(C_H) < \lambda_i(C_P),\ \lambda_i(C_H) < \lambda_i(C_L), 1 \leq i \leq 6 \tag{5.25}$$

where $\lambda_i(\cdot)$ denotes the i-th largest eigenvalue.

Proof. Let us write $M_1 \succ M_2$ if matrices M_1 and M_2 are real symmetric and $M_1 - M_2$ is positive definite.

By comparing (5.23) with (5.14) and (5.19), we find $C_H = \left(C_P^{-1} + C_L^{-1}\right)^{-1}$, which is equivalent to

$$C_H^{-1} = C_P^{-1} + C_L^{-1}. \tag{5.26}$$

It holds that $C_P \succ 0$, $C_L \succ 0$ since they are both covariance matrices, which further implies $C_P^{-1} \succ 0, C_L^{-1} \succ 0$.

As a result, we have

$$C_H^{-1} - C_P^{-1} = C_L^{-1} \succ 0 \text{ and } C_H^{-1} - C_L^{-1} = C_P^{-1} \succ 0,$$

which means $C_H^{-1} \succ C_P^{-1}$ and $C_H^{-1} \succ C_L^{-1}$. According to Theorem 7.7.3 in [110],

$$C_H^{-1} \succ C_P^{-1} \Leftrightarrow C_H \prec C_P, \text{ and } C_H^{-1} \succ C_L^{-1} \Leftrightarrow C_H \prec C_L,$$

which further leads to (5.25) per Corollary 7.7.4 in [110]. □

5.5 Experiments

We have implemented our method in C++ and named it Point and Line based Visual Odometry (PLVO). We evaluate PLVO under both varying and constant lighting, and compare it with the following state-of-the-art algorithms:

- Kpoint: a representative keypoint based visual SLAM algorithm [33], open source software, referred to as Kpoint here.
- DVO: a recent dense visual SLAM method [91], open source software.
- Edge: the latest edge-based RGB-D method [104], referred to as Edge here. Edge is only compared on public dataset because it is not open source.

We start with the evaluation under varying lighting.

5.5.1 Test Under Varying Lighting

To evaluate PLVO under real-world scenarios, we record RGB-D data at 30 FPS by hand-holding a Kinect and walking in typical indoor environments, including corridors, staircases and halls. The trajectory lengths, listed in Table 5.1, range from 41 m to 86 m,

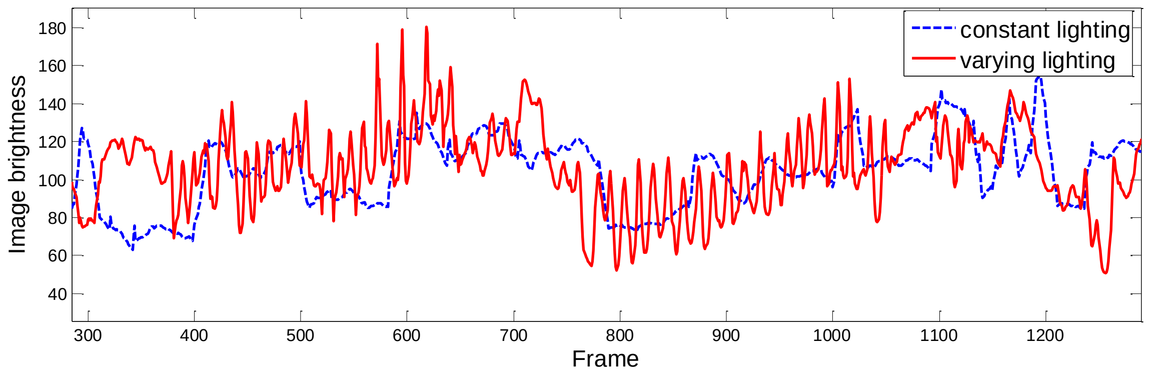


Figure 5.3: Example of image brightness change over time under constant/varying lighting (from Corridor-C). Here image brightness means the average intensity of an image.

which are sufficient for indoor testing. At each site, we record a pair of sequences under constant and varying lighting, respectively. Lighting variations are generated by constantly adjusting and/or swinging a hand-held dimmable LED light panel (Polaroid PL-LED350). Figure 5.3 shows an example of the effect of varying lighting - while the image brightness (i.e. the average intensity of an image) varies over time even under constant lighting, the fluctuation of image brightness is significantly more intense under varying lighting. This brings great challenge for feature tracking.

We enforce the starting and ending points of each sequence to be at the same position. As a result, we define a trajectory endpoint drift (TED) to be the Euclidean distance between the two endpoints of an estimated trajectory, which serves as our evaluation metric. For fair comparison, loop closure is disabled for Kpoint and DVO since it is beyond the scope of this work. Table 5.1 shows the test results, where we use bold font to indicate the best result in each row. As Kpoint allows using various point detectors, we report the best result for each sequence obtained by respectively applying SIFT [17], SURF and ORB [111]. From Table 5.1, we see that PLVO achieves the least TED on the majority of sequences, under both constant and varying lighting conditions. This clearly demonstrates the accuracy advantage of PLVO, as well as its robustness against lighting variations.

5.5.2 Test on TUM Dataset Under Constant Lighting

We also evaluate our method under constant lighting using the TUM FR1 dataset [112], which is most frequently studied in the literature. The FR1 dataset consists of 9 sequences with high-precision ground truth provided, mainly covering desktop and office scenarios.

The evaluation metric used here is the relative pose error (RPE) proposed in [112]. For a given interval Δ, the RPE at time instant i is defined as

$$\mathbf{E}_i := \left(\mathsf{Q}_i^{-1}\mathsf{Q}_{i+\Delta}\right)^{-1}\left(\mathsf{P}_i^{-1}\mathsf{P}_{i+\Delta}\right), \tag{5.27}$$

Table 5.1: TED (IN METERS)

Site (travel distance)	Lighting	Kpoint	DVO	PLVO
Corridor-A (82 m)	constant	**4.36**	7.10	7.50
	varying	16.68	15.41	**12.20**
Corridor-B (77 m)	constant	8.25	7.56	**5.28**
	varying	12.75	12.96	**5.15**
Corridor-C (86 m)	constant	6.53	6.12	**5.70**
	varying	7.30	5.93	**3.46**
Staircase-A (52 m)	constant	4.04	2.26	**2.13**
	varying	4.47	3.17	**2.41**
Staircase-B (45 m)	constant	5.77	**1.72**	4.50
	varying	**3.12**	3.35	6.41
Staircase-C (41 m)	constant	4.51	13.87	**2.74**
	varying	8.79	16.00	**1.86**
Entry-Hall (54 m)	constant	1.53	**1.31**	1.63
	varying	3.78	6.59	**3.70**
Auditorium (53 m)	constant	5.78	2.39	**1.86**
	varying	6.74	10.66	**4.44**
Classroom (45 m)	constant	2.47	3.48	**1.93**
	varying	2.58	4.73	**2.16**

where $\mathsf{Q}_i \in \mathrm{SE}(3)$ and $\mathsf{P}_i \in \mathrm{SE}(3)$ are the i-th ground truth and estimated poses, respectively. Specifically, we compute the root mean squared error (RMSE) of the translational RPE and that of the rotational RPE over the sequence.

Table 5.2 contains two comparison results. On the left part, we compare PLVO with Kpoint and Edge, where the RPE is computed with $\Delta = 1$ *frame* in (5.27). The errors of Kpoint are computed using their published resulting trajectories [113]. The errors of Edge are directly excerpted from [104]. For each sequence, the first and second rows represent the translational and rotational errors, respectively. It shows that PLVO outperforms its counterparts. We compute an average error over all sequences weighted by their frame numbers. Our method achieves the smallest average errors. Specifically, the average translational and rotational errors of PLVO are 42.5% and 28.3% smaller than the second best one (Edge), respectively.

Table 5.2: RMSE OF RPE ON TUM FR1 SEQUENCES

Seq.	Kpoint	Edge	PLVO	PLVO	DVO
(#Frame)	error per frame			error per sec	
360 (745)	13.8 mm 0.63 deg	**11.2** mm 0.55 deg	**11.2** mm **0.45** deg	**84** mm	119 mm
desk (575)	11.7 mm 0.73 deg	**8.6** mm 0.70 deg	10.8 mm **0.60** deg	39 mm	**30** mm
desk2 (614)	17.6 mm 1.07 deg	**8.9** mm 0.7 deg	11.5 mm **0.64** deg	**54** mm	55 mm
floor (1214)	3.7 mm 0.29 deg	15.7 mm 0.47 deg	**3.5** mm **0.28** deg	**24** mm	90 mm
plant (1112)	20.7 mm 1.25 deg	6.9 mm 0.49 deg	**5.1** mm **0.34** deg	**24** mm	36 mm
room (1332)	13.7 mm 0.63 deg	6.2 mm 0.48 deg	**5.3** mm **0.36** deg	49 mm	**48** mm
rpy (687)	12.1 mm 0.91 deg	**7.2** mm 0.67 deg	9.1 mm **0.63** deg	52 mm	**43** mm
teddy (1395)	25.4 mm 1.45 deg	36.5 mm 0.92 deg	**11.5** mm **0.47** deg	**50** mm	67 mm
xyz (788)	5.8 mm **0.35** deg	**4.7** mm 0.41 deg	5.3 mm **0.35** deg	**22** mm	24 mm
weighted mean	14.4 mm 0.83 deg	13.4 mm 0.60 deg	**7.7** mm **0.43** deg	**43** mm	58 mm

We compare PLVO with DVO separately in the right part of Table 5.2 because only translational errors are reported in [91] with its unit being m/s, i.e. $\Delta = 1$ *sec* in (5.27). PLVO produces an average translational error which is 34.9% smaller than DVO.

5.6 Conclusions

To improve visual odometry robustness, we proposed an RGB-D camera based method by fusing point and line features. We proved that fusing these two types of features produced smaller uncertainty in the MLE of relative motion than using either feature type alone. Our method was evaluated on real-world data in experiments. We compared its performance with state-of-the-art methods Kpoint, Edge and DVO, under both constant and varying lighting. Our method exhibited both superior robustness to lighting change and better accuracy in either settings.

6. ROBUST RECOGNITION OF MIRRORED PLANES USING TWO VIEWS*

The previous sections have presented localization and mapping algorithms for mobile robots in urban and/or indoor environments. Another critical capability needed for autonomous navigation is obstacle avoidance. Unfortunately, in man-made environments highly reflective surfaces, such as glassy building exterior and mirrored walls, challenge almost every type of sensors including laser range finders, sonar arrays, and cameras. This is because light and sound signals simply bounce off the surfaces which become invisible to the sensors. Therefore, detecting these surfaces is necessary for robots to avoid collisions in navigation.

In this section we report a method for this new planar mirror detection problem (PMDP) using two views from an on-board camera. First, we derive geometric constraints for corresponding real-virtual features across two views. The constraints include 1) the mirror normal as a function of vanishing points of lines connecting the real-virtual feature point pairs and 2) the mirror depth in closed form format derived from a mirror plane-induced homography. We also address the issue that popular feature detectors, such as SIFT, are not reflection invariant by combining a secondary reflection with an affine scale-invariant feature transform (ASIFT). Based on the results, we employ a RANSAC framework to develop a robust mirror detection algorithm. We have implemented the algorithm and tested it under both in-lab and field settings. The algorithm has achieved an overall accuracy rate of 91.0%.

*Reprinted with permission from "Automatic recognition of spurious surface in building exterior survey" by Y. Lu, D. Song, H. Li, and J. Liu, 2013. *IEEE International Conference on Automation Science and Engineering (CASE)*, pp. 1059-1064, Copyright © 2013 IEEE.

6.1 Related Work

PMDP is not a simple plane reconstruction problem using 3D vision. It relates to many areas including intelligence level tests in artificial intelligence community, planar catadioptric stereo (PCS) systems, construction of specular surfaces, and reflection invariant feature extractions.

In AI and animal behavior communities, researchers often assess intelligence levels based on the subject's ability of detecting a mirror or its own reflection [114, 115]. In the well known mirror and mark test, a subject has a mark that cannot be directly seen but is visible in the mirror. If the subject increases the exploration and self-direction actions towards the mark, it means that the subject recognizes the mirror image as self. Existing results show that chimpanzees [115], gorillas [116], dolphins [117], and magpies [118] have evident self-recognition in front of mirrors except monkeys [119]. We do not have mirror and mark tests for robots yet. It is clearly not a trivial problem. Initial related results focus on robot self recognition [120, 121] using motion and appearance, which is not as difficult as recognizing a mirror when a robot cannot see its own reflection. Such cases are not unusual because the robot cannot see itself when approaching a mirror from side. Our approach addresses this problem by exploring symmetricity in the scene.

Mirror detection is also related to PCS systems in computer vision. A PCS system usually consists of a static camera and one or more planar mirrors with the aim of achieving stereo or SFM [122–124]. Since detecting mirror pose is just a calibration problem in PCS systems, in-lab settings and calibration patterns (e.g. checkerboard) can be used here. However, this is not a viable approach when robots need to detect mirror surfaces *in situ*.

In a way, planar mirror detection can be viewed as a special case of specular surface construction. Existing approaches rely on active sensing by changing lighting [125–127] and polarity [128–131], or assuming curvature of the mirror [132]. These approaches have difficulty to be adapted for robots because natural lighting can easily overwhelm the setup. To avoid the issue, we use features from images.

SIFT [17] is well known for its invariance to image scaling and translation, and partial invariance to affine distortion. However, it is not reflection invariant and thus cannot be applied to our problem. As extensions of SIFT, descriptors invariant to mirror reflection have recently been designed by modifying the SIFT descriptor structure at the expense of distinctiveness, such as MI-SIFT [133] and FIND [134]. They still cannot fit our need because our feature correspondence involves not only a reflection difference but also a significant projective distortion induced by perspective changes. On the other hand, descriptors invariant to affine transforms can handle large perspective changes (e.g., [135, 136]). Among these affine invariant descriptors, ASIFT [137] shows promising performance and becomes our choice of feature transformation. Later we will show how to make ASIFT reflection-invariant.

In a previous work [138], our group has investigated the problem of estimating the orientation of a mirror plane using a single view. However, the depth information cannot be extracted from a single view and it limits the detection ability.

6.2 Problem Definition

To define our problem and focus on the most relevant issues, we have the following assumptions.

a.1 Each view captures a real scene and its mirror reflection, and the scene is feature-rich.

a.2 The intrinsic camera matrix is known to be K.

a.3 The baseline distance $|\mathbf{t}|$ between two views is known. The distance is usually short and can be measured by on-board sensors like IMU. If $|\mathbf{t}|$ is unknown, our method still applies but the depth result is measured in ratio instead of absolute value.

We also have the following conventions in notation. Let I and $\{I\}$ be the image and the image coordinate system (ICS) for the first view, respectively. I' and $\{I'\}$ are defined similarly for the second view. The camera coordinate system (CCS) is right-handed, with the origin $\mathbf{C}$ at the camera center, and Z-axis along the principal axis. With respect to the CCS of the first view, we define,

- $\pi_m = (\mathbf{n}_m^\mathsf{T}, d_m)^\mathsf{T}$ as the mirror plane where $\mathbf{n}_m$ is a 3×1 unit vector indicating the normal of π_m, and d_m is the plane depth (i.e., the distance from $\mathbf{C}$ to π_m),
- $\mathbf{X}_{\mathbf{r}i}$ as the i-th real 3D point and $\mathbf{X}_{\mathbf{v}i}$ as its mirror reflection (a virtual point),
- $\mathbf{x}_{\mathbf{r}i}$ and $\mathbf{x}_{\mathbf{v}i}$ as the projections of $\mathbf{X}_{\mathbf{r}i}$ and $\mathbf{X}_{\mathbf{v}i}$ in $\{I\}$, respectively, and
- $\mathbf{X}_{\mathbf{r}i} \leftrightarrow \mathbf{X}_{\mathbf{v}i}$ as a 3D real-virtual (R-V) pair and $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}$ as a 2D R-V pair.

In the CCS of the second view, notations differ from their counterparts in the CCS of the first view by adding a superscript $'$, e.g. $\mathbf{n}'_m$, $\mathbf{x}'_{\mathbf{r}i}$ and $\mathbf{x}'_{\mathbf{v}i}$. It is worth noting that there is a new type of correspondence between the 2D R-V pairs in both views, which is denoted in a quadruple format: $\mathbf{Q}_i = \{\mathbf{x}_{\mathbf{r}i}, \mathbf{x}_{\mathbf{v}i}, \mathbf{x}'_{\mathbf{r}i}, \mathbf{x}'_{\mathbf{v}i}\}$.

Also, all the above notations about points are represented in homogeneous coordinates while their inhomogeneous counterparts are denoted by adding a tilde on their top, e.g., $\widetilde{\mathbf{x}}_{\mathbf{r}i}$.

With assumptions and notations defined, our PMDP is,

Definition 4. *Given two views I and I', the camera calibration matrix K and the camera translation distance $|\mathbf{t}|$, determine if there is a mirror. If so, estimate π_m.*

6.3 Modeling

We begin with analyzing the geometric relationship between noise-free feature points. The geometric relationship will be used in a RANSAC framework to filter noisy inputs later. The noise-free feature inputs here are quadruples $\{\mathbf{Q}_i\}$. The geometric relationship is constraints on quadruples induced by 3D reflection and the imaging process. As the result, π_m will be derived as a function of quadruples in two stages: orientation and depth. First, we solve the mirror orientation using quadruples.

Lemma 6. *Given two quadruples $\mathbf{Q}_i$ and $\mathbf{Q}_j$, the mirror normal with respect to both CCSs can be obtained as follows,*

$$\begin{aligned}\mathbf{n}_m &= \mathrm{K}^{-1}(\mathbf{x}_{\mathbf{r}i} \times \mathbf{x}_{\mathbf{v}i}) \times (\mathbf{x}_{\mathbf{r}j} \times \mathbf{x}_{\mathbf{v}j}),\\ \mathbf{n}'_m &= \mathrm{K}^{-1}(\mathbf{x}'_{\mathbf{r}i} \times \mathbf{x}'_{\mathbf{v}i}) \times (\mathbf{x}'_{\mathbf{r}j} \times \mathbf{x}'_{\mathbf{v}j}),\end{aligned} \tag{6.1}$$

where symbol '$\times$' represents the cross product.

Proof. Consider the geometry relationship in Fig. 6.1. As a convention, we define $\overleftrightarrow{AB}$ as the line passing through points A and B. From the property of planar mirror reflection, we have $\overleftrightarrow{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}} \perp \pi_m$, $\overleftrightarrow{\mathbf{X}_{\mathbf{r}j}\mathbf{X}_{\mathbf{v}j}} \perp \pi_m$, and thus $\overleftrightarrow{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}} // \overleftrightarrow{\mathbf{X}_{\mathbf{r}j}\mathbf{X}_{\mathbf{v}j}}$. After a projective transformation, the projections of $\overleftrightarrow{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}}$ and $\overleftrightarrow{\mathbf{X}_{\mathbf{r}j}\mathbf{X}_{\mathbf{v}j}}$ in $\{I\}$ (or $\{I'\}$) would intersect at a vanishing point $\mathbf{v}$ (or $\mathbf{v}'$) in the corresponding ICS,

$$\begin{aligned}(\mathbf{x}_{\mathbf{r}i} \times \mathbf{x}_{\mathbf{v}i}) \times (\mathbf{x}_{\mathbf{r}j} \times \mathbf{x}_{\mathbf{v}j}) &= \mathbf{v}\\ (\mathbf{x}'_{\mathbf{r}i} \times \mathbf{x}'_{\mathbf{v}i}) \times (\mathbf{x}'_{\mathbf{r}j} \times \mathbf{x}'_{\mathbf{v}j}) &= \mathbf{v}'.\end{aligned} \tag{6.2}$$

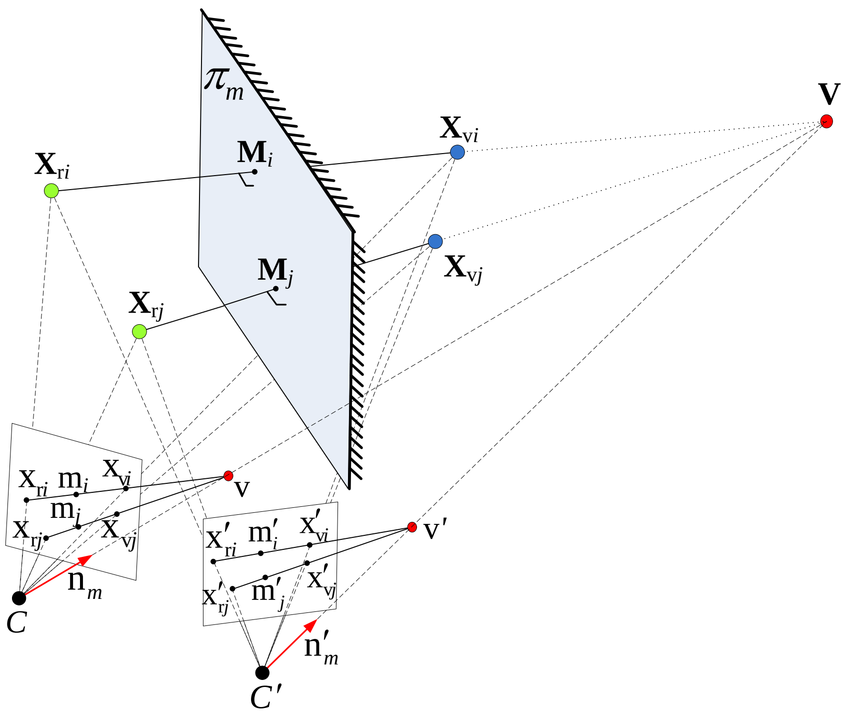


Figure 6.1: A perspective illustration of the geometry relationship between real-virtual pairs across two views.

On the other hand, $\mathbf{v}$ can be viewed as the projection of $\mathbf{n}_m$ in $\{I\}$

$$\mathbf{v} = \mathrm{K}\mathbf{n}_m, \text{ and similarly, } \mathbf{v}' = \mathrm{K}\mathbf{n}'_m. \tag{6.3}$$

Combining (6.2) and (6.3), we obtain (6.1). □

The second step is to derive mirror depth d_m. From epipolar geometry, we can obtain the camera rotation matrix R and translation vector $\mathbf{t}$ by decomposing the essential matrix [45]. A straightforward way of computing the equation of π_m is by reconstructing 3D points via triangulation. However, we will show a homography based method which avoids the triangulation process.

Our method involves the homography between the corresponding middle points of R-

V pairs in two views. Let $\overline{AB}$ denote the line segment defined by points A and B in the rest of this section. Denote the midpoint of $\overline{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}}$ by M_i, and its projection in $\{I\}$ by $\mathbf{m}_i$ (see Fig. 6.1 for examples). $\mathbf{m}_i$ can be obtained using a cross ratio which is detailed in the following lemma:

Lemma 7. *Given quadruple $\mathbf{Q}_i$, the projection $\mathbf{m}_i$ of the midpoint M_i of $\overline{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}}$ is determined as follows,*

$$\widetilde{\mathbf{m}}_i = (1-a)\widetilde{\mathbf{x}}_{\mathbf{r}i} + a\widetilde{\mathbf{x}}_{\mathbf{v}i}, \quad \textit{and} \ \ a = \frac{|\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{v}}|}{2|\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{v}}| - |\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{x}_{\mathbf{v}i}}|}, \tag{6.4}$$

where $|\cdot|$ denotes the length of the line segment.

Proof. Consider the projection from $\overleftrightarrow{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}}$ to $\overleftrightarrow{\mathbf{x}_{\mathbf{r}i}\mathbf{x}_{\mathbf{v}i}}$. A basic invariant in this projection is the cross ratio of the four collinear points $\mathbf{X}_{\mathbf{r}i}$, M_i, $\mathbf{X}_{\mathbf{v}i}$, and $\mathbf{V}$,

$$\frac{|\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{m}_i}||\overline{\mathbf{x}_{\mathbf{v}i}\mathbf{v}}|}{|\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{x}_{\mathbf{v}i}}||\overline{\mathbf{m}_i\mathbf{v}}|} = \frac{|\overline{\mathbf{X}_{\mathbf{r}i}\mathrm{M}_i}||\overline{\mathbf{X}_{\mathbf{v}i}\mathbf{V}}|}{|\overline{\mathbf{X}_{\mathbf{r}i}\mathbf{X}_{\mathbf{v}i}}||\overline{\mathrm{M}_i\mathbf{V}}|} = \frac{1}{2}. \tag{6.5}$$

Representing $\mathbf{m}_i$ as $\widetilde{\mathbf{m}}_i = (1-a)\widetilde{\mathbf{x}}_{\mathbf{r}i} + a\widetilde{\mathbf{x}}_{\mathbf{v}i}$, $0 \le a \le 1$, in the inhomogeneous coordinate, we have

$$\begin{aligned} |\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{m}_i}| &= a|\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{x}_{\mathbf{v}i}}|, \\ |\overline{\mathbf{m}_i\mathbf{v}}| &= |\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{v}}| - a|\overline{\mathbf{x}_{\mathbf{r}i}\mathbf{x}_{\mathbf{v}i}}|. \end{aligned} \tag{6.6}$$

Substituting (6.6) into (6.5) gives the final result in (6.4). □

We now can derive the mirror depth with $\mathbf{m}_i$.

Lemma 8. *Given quadruple $\mathbf{Q}_i$ and mirror normal $\mathbf{n}_m$, the mirror depth is*

$$d_m = ([\mathbf{m}'_i]_\times \mathrm{K}\mathrm{R}\mathrm{K}^{-1}\mathbf{m}_i)^\dagger [\mathbf{m}'_i]_\times \mathrm{K}\mathbf{t}\mathbf{n}_m^\mathsf{T}\mathrm{K}^{-1}\mathbf{m}_i \tag{6.7}$$

where $(\cdot)^\dagger$ *denotes the pseudoinverse operation, and* $[\mathbf{m}'_i]_\times$ *is a skew-symmetric matrix,*

$$\begin{bmatrix} 0 & -\mathbf{m}'_{i3} & \mathbf{m}'_{i2} \\ \mathbf{m}'_{i3} & 0 & -\mathbf{m}'_{i1} \\ -\mathbf{m}'_{i2} & \mathbf{m}'_{i1} & 0 \end{bmatrix}. \tag{6.8}$$

Proof. Observe that $\mathbf{M}_i$ lies on the plane π_m. Then $\mathbf{m}_i$ and $\mathbf{m}'_i$ must obey a homography $\mathbf{m}'_i = \mathrm{H}\mathbf{m}_i$ induced by π_m, where H can be expressed as [45]

$$\mathrm{H} = \mathrm{K}(\mathrm{R} - \frac{1}{d_m}\mathbf{t}\mathbf{n}_m^\mathsf{T})\mathrm{K}^{-1} \tag{6.9}$$

H has 1 degree of freedom (DOF) since only d_m is unknown.

$\mathbf{m}_i$ and $\mathbf{m}'_i$ can be computed from $\mathbf{Q}_i$ using (6.4). Since $\mathbf{m}'_i = \mathrm{H}\mathbf{m}_i = \mathrm{K}(\mathrm{R} - \frac{1}{d_m}\mathbf{t}\mathbf{n}_m^\mathsf{T})\mathrm{K}^{-1}\mathbf{m}_i$, we have

$$\begin{aligned} &\mathbf{m}'_i \times \mathrm{K}(\mathrm{R} - \frac{1}{d_m}\mathbf{t}\mathbf{n}_m^\mathsf{T})\mathrm{K}^{-1}\mathbf{m}_i \\ =&[\mathbf{m}'_i]_\times \mathrm{KRK}^{-1}\mathbf{m}_i - [\mathbf{m}'_i]_\times \mathrm{K}\frac{1}{d_m}\mathbf{t}\mathbf{n}_m^\mathsf{T}\mathrm{K}^{-1}\mathbf{m}_i = \mathbf{0} \end{aligned}$$

Then we have

$$[\mathbf{m}'_i]_\times \mathrm{KRK}^{-1}\mathbf{m}_i d_m = [\mathbf{m}'_i]_\times \mathrm{K}\mathbf{t}\mathbf{n}_m^\mathsf{T}\mathrm{K}^{-1}\mathbf{m}_i$$

The above system of equations is over-determined since the rank of $[\mathbf{m}'_i]_\times$ is 2. Thus, the least-square solution of d_m is given by (6.7), which is also an exact solution when the system is noise-free. □

6.4 Algorithm

Section 6.3 provides geometric relationship for noise-free quadruples. To complete the algorithm, we need to select correct feature transformation and verify the geometric relationship with respect to noisy features using the well-accepted RANSAC framework. First, let us detail the feature detection method selection in quadruple extraction.

6.4.1 Quadruple Extraction

To form a quadruple, we need two kinds of point correspondences: cross-view correspondence, e.g. $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}'_{\mathbf{r}i}$, and R-V pair correspondence, e.g. $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}$. The former can be handled by standard feature extraction methods, such as SIFT, as long as the perspective change is not significant. However, the latter is nontrivial because $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}$ involves an improper transformation in 3D (between $\mathbf{X}_{\mathbf{r}i}$ and $\mathbf{X}_{\mathbf{v}i}$).

Therefore, the key to this problem is how to find features and their correspondence under the improper transformation. We need to convert the reflection to a rigid body transformation such that existing feature extraction and matching algorithms can be employed. The intuition of our approach comes from a special scenario when a secondary mirror π_s is placed in the same plane as π_m but in the opposite orientation. Letting $\mathbf{X}_{\mathbf{s}i}$ be the result of $\mathbf{X}_{\mathbf{r}i}$ after a consecutive reflection about π_m and π_s, it is clear that $\mathbf{X}_{\mathbf{s}i}$ is exactly the same point as $\mathbf{X}_{\mathbf{r}i}$, which makes matching their projections in image a trivial problem.

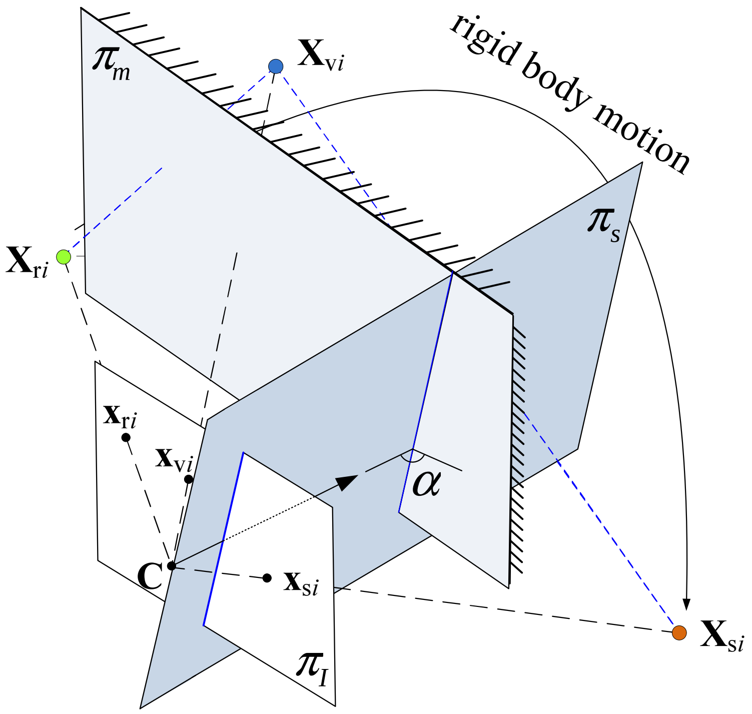
In fact, it is proved that two consecutive reflections lead to a rigid body transformation regardless of the mirror configuration [123]. Therefore, the position of π_s can be arbitrarily chosen. The remaining problem is that introducing the secondary mirror in 3D is difficult to implement because it requires the 3D positions of points to perform the secondary reflection, which is not viable. Fortunately, a special group of π_s allows the 3D reflection to be reduced to 2D image flipping about an arbitrary axis in the ICS, which is independent of point positions.

Lemma 9. *If π_s contains the camera principal axis, then for any $\mathbf{X}_{\mathbf{s}i}$, its projection $\mathbf{x}_{\mathbf{s}i}$ can be obtained by flipping $\mathbf{x}_{\mathbf{v}i}$ about an axis in the ICS.*

Proof. Denote π_I as the image plane as illustrated in Fig. 6.2.

Since it contains the principal axis (i.e. the Z-axis), π_s can be expressed as $(\mathbf{n}_s^{\mathrm{T}}, 0)^{\mathrm{T}}$ where $\mathbf{n}_s = (n_x, n_y, 0)^{\mathrm{T}}$. Since $\mathbf{X}_{\mathbf{v}i}$ and $\mathbf{X}_{\mathbf{s}i}$ are symmetrical about π_s, we have

$$\mathbf{X}_{\mathbf{s}i} = \mathrm{T}\mathbf{X}_{\mathbf{v}i}, \tag{6.10}$$



π_m
$\mathbf{X}_{\mathrm{v}i}$
rigid body motion
π_s
$\mathbf{X}_{\mathrm{r}i}$
$\mathbf{x}_{\mathrm{r}i}$
$\mathbf{x}_{\mathrm{v}i}$
α
$\mathbf{C}$
$\mathbf{x}_{\mathrm{s}i}$
$\mathbf{X}_{\mathrm{s}i}$
π_I

Figure 6.2: The configuration of π_I, π_s, and π_m. π_s is placed to contain the camera principal axis.

where

$$\mathrm{T} = \begin{bmatrix} \mathrm{I}_3 - 2\mathbf{n}_s\mathbf{n}_s^{\mathrm{T}} & 0 \\ 0 & 1 \end{bmatrix}.$$

Moreover, we have the projection relationship

$$\mathbf{x}_{\mathbf{v}i} = \mathrm{P}\mathbf{X}_{\mathbf{v}i}, \quad \mathbf{x}_{\mathbf{s}i} = \mathrm{P}\mathbf{X}_{\mathbf{s}i}, \tag{6.11}$$

where $\mathrm{P} = [\mathrm{K}|0]$ is the projection matrix.

Combining (6.10) and (6.11) gives

$$\begin{aligned} \mathbf{x}_{\mathbf{s}i} &= \mathrm{P}\mathrm{T}\mathrm{P}^{\dagger}\mathbf{x}_{\mathbf{v}i} \qquad (6.12) \\ &= \begin{bmatrix} \mathrm{I}_2 - 2\mathbf{n}_{s1:2}\mathbf{n}_{s1:2}^{\mathrm{T}} & 0 \\ 0 & 1 \end{bmatrix} \mathbf{x}_{\mathbf{v}i}, \end{aligned}$$

where $\mathrm{P}^{\dagger}$ is the pseudo-inverse of P and $\mathbf{n}_{s1:2} = (n_x, n_y)^{\mathrm{T}}$.

Eq. (6.12) implies that $\mathbf{x}_{\mathbf{v}i}$ and $\mathbf{x}_{\mathbf{s}i}$ are symmetrical about an axis with a normal $\mathbf{n}_{s1:2}$, which is actually the intersection of π_I and π_s. This completes the proof.

□

Lemma 9 allows us to find the correspondence between $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{s}i}$ instead of that of $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}$. Furthermore, the axis of flipping can be arbitrarily chosen because there is only a planar rotation difference between the resulting images with different flipping axes.

Although the image flipping process solves the improper transformation issue, it also introduces a new challenge for feature matching. The problem is that the rotation angle θ of the resulting rigid body motion (between $\mathbf{X}_{\mathbf{r}i}$ and $\mathbf{X}_{\mathbf{s}i}$) is as large as two times of the angle α between the principal axis and π_m [123] (see Fig. 6.2). Therefore, as the value of θ varies with α in different cases, it can easily lead to a significant perspective change which often fails the standard SIFT algorithm.

To handle this problem, we employ an affine invariant feature extraction algorithm ASIFT which has advantages over SIFT when dealing with large perspective changes. Once a correspondence $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{s}i}$ is identified, the R-V pair $\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}$ is readily established based on the known mapping between $\mathbf{x}_{\mathbf{v}i}$ and $\mathbf{x}_{\mathbf{s}i}$. Algorithm 6.1 summarizes how quadruples are constructed.

Algorithm 6.1: ASIFT-based Quadruple Extraction

```
   Input  : Two images $I$ and $I'$
   Output: A set of quadruples $\{\mathbf{Q}_k\}$
1  flip $I$ left-right (or up-down) to get $I_f$;
2  find matches $\{\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{s}i}\}$ between $I$ and $I_f$ using ASIFT;
3  map $\mathbf{x}_{\mathbf{s}i}$ in $I_f$ back to $\mathbf{x}_{\mathbf{v}i}$ in $I$ to establish R-V correspondences $\{\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}\}$;
4  apply steps 1-3 to $I'$ to obtain $\{\mathbf{x}'_{\mathbf{r}j} \leftrightarrow \mathbf{x}'_{\mathbf{v}j}\}$;
5  find cross-view matches $\{\mathbf{x}_{\mathbf{r}k} \leftrightarrow \mathbf{x}'_{\mathbf{r}k}\}$ from between $\{\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}\}$ and $\{\mathbf{x}'_{\mathbf{r}j} \leftrightarrow \mathbf{x}'_{\mathbf{v}j}\}$
   using putative matching of ASIFT;
6  construct quadruples $\{\mathbf{Q}_k\}$ from $\{\mathbf{x}_{\mathbf{r}i} \leftrightarrow \mathbf{x}_{\mathbf{v}i}\}$ and $\{\mathbf{x}'_{\mathbf{r}j} \leftrightarrow \mathbf{x}'_{\mathbf{v}j}\}$ according to
   $\{\mathbf{x}_{\mathbf{r}k} \leftrightarrow \mathbf{x}'_{\mathbf{r}k}\}$;
7  return $\{\mathbf{Q}_k\}$;
```

6.4.2 Maximum Likelihood Estimation

To apply RANSAC framework, we need to estimate $\mathbf{n}_m$, $\mathbf{n}'_m$ and d_m using the quadruples $\{\mathbf{Q}_i\}$ from the inlier set by minimizing a cost function. Assuming measurement errors are Gaussian, then the estimation is MLE if reprojection error is employed as the cost function. Let us derive this metric.

For $\mathbf{Q}_i$, let $\mathcal{X}_i = (\tilde{x}_{\mathbf{r}i}, \tilde{y}_{\mathbf{r}i}, \tilde{x}_{\mathbf{v}i}, \tilde{y}_{\mathbf{v}i}, \tilde{x}'_{\mathbf{r}i}, \tilde{y}'_{\mathbf{r}i}, \tilde{x}'_{\mathbf{v}i}, \tilde{y}'_{\mathbf{v}i})^\mathsf{T}$ be a 8-vector formed by concatenating the inhomogeneous coordinates of $\mathbf{x}_{\mathbf{r}i}, \mathbf{x}_{\mathbf{v}i}, \mathbf{x}'_{\mathbf{r}i}$ and $\mathbf{x}'_{\mathbf{v}i}$. Given points $\mathcal{X}_i$ in the measurement space $\mathbb{R}^8$, the task of estimating $\mathbf{n}_m$, $\mathbf{n}'_m$ and d_m becomes finding a variety that passes through the points $\mathcal{X}_i$ in $\mathbb{R}^8$. Because of noise, it is impossible to fit a variety exactly. In this case, let $\mathcal{V}$ be the variety corresponding to $\mathbf{n}_m$, $\mathbf{n}'_m$ and d_m, and let $\widehat{\mathcal{X}}_i$ be the closest point to $\mathcal{X}_i$ lying on $\mathcal{V}$.

Given $\mathbf{n}_m$, $\mathbf{n}'_m$ and d_m, define

$$\mathcal{C}_{\mathcal{V}}(\widehat{\mathcal{X}}_i) := \begin{bmatrix} (\widehat{\mathbf{x}}_{\mathbf{r}i} \times \widehat{\mathbf{x}}_{\mathbf{v}i})^{\mathsf{T}} \mathrm{K} \mathbf{n}_m \\ (\widehat{\mathbf{x}}'_{\mathbf{r}i} \times \widehat{\mathbf{x}}'_{\mathbf{v}i})^{\mathsf{T}} \mathrm{K} \mathbf{n}'_m \\ \widehat{\mathbf{m}}'_i \times \mathrm{H} \widehat{\mathbf{m}}_i \end{bmatrix},$$

where $\mathrm{H}, \widehat{\mathbf{m}}_i$ and $\widehat{\mathbf{m}}'_i$ are intermediate variables computed using (6.9) and (6.4), respectively, and $\widehat{\mathbf{x}}_{\mathbf{r}i} = (\widehat{x}_{\mathbf{r}i}, \widehat{y}_{\mathbf{r}i}, 1)^{\mathsf{T}}$, and similarly for $\widehat{\mathbf{x}}_{\mathbf{v}i}$,$\widehat{\mathbf{x}}'_{\mathbf{r}i}$ and $\widehat{\mathbf{x}}'_{\mathbf{v}i}$. Then the MLE method is to find $\mathbf{n}_m$, $\mathbf{n}'_m$, d_m and $\widehat{\mathcal{X}}_i$ that minimize the error function

$$\sum_i \|\mathcal{X}_i - \widehat{\mathcal{X}}_i\|^2_{\Sigma_i}, \tag{6.13}$$

subject to $\mathcal{C}_{\mathcal{V}}(\widehat{\mathcal{X}}_i) = \mathbf{0}, \forall i$, where Σ_i is the covariance of $\mathcal{X}_i$, and $\|\cdot\|_\Sigma$ represents the Mahalanobis distance.

Although minimizing the reprojection error is MLE, it involves solving a high-dimensional non-linear optimization problem, which is quite complex and time-consuming. To speed up the algorithm, we derive Sampson error approximation. Instead of finding the closest point $\widehat{\mathcal{X}}_i$ on the variety $\mathcal{V}$ to the measurement $\mathcal{X}_i$, the Sampson error function estimates a first-order approximation to $\widehat{\mathcal{X}}_i$. For given $\mathbf{n}_m$, $\mathbf{n}'_m$ and d_m, any point $\mathcal{X}_i$ lying on $\mathcal{V}$ will satisfy $\mathcal{C}_{\mathcal{V}}(\mathcal{X}_i) = \mathbf{0}$. Then the Sampson approximation to (6.13) is $\sum_i \epsilon_i^{\mathsf{T}} (\mathrm{J}_i \Sigma_i \mathrm{J}_i^{\mathsf{T}})^{-1} \epsilon_i$ where $\epsilon_i = \mathcal{C}_{\mathcal{V}}(\mathcal{X}_i)$ and $\mathrm{J}_i = \frac{\partial \mathcal{C}_{\mathcal{V}}}{\partial \mathcal{X}_i}$.

Algorithm 6.2: Robust Mirror Estimation using RANSAC

```
Input  : Two images $I$ and $I'$
Output: Mirror plane $\pi_m$ or no mirror
1  obtain a set $\mathcal{S}$ of quadruples using Algorithm 6.1;
2  $N = \infty$;
3  for $k \leftarrow 1$ to $N$ do
4      randomly sample 2 quadruples from $S$;
5      compute $\mathbf{n}_m^{(k)}$, $\mathbf{n}'^{(k)}_m$ and $d_m^{(k)}$ using (6.1) and (6.7);
6      $\mathscr{I}_k = \emptyset$ ;                         // initialize inlier set
7      for $\mathbf{Q}_i \in S$ do
8          $D_i = \sqrt{\epsilon_i^{\mathsf{T}}(\mathrm{J}_i\Sigma_i\mathrm{J}_i^{\mathsf{T}})^{-1}\epsilon_i}$;
9          if $|D_i| < \tau_d$ then
10             $\mathscr{I}_k = \mathscr{I}_k \cup \mathbf{Q}_i$;
11     update $N$ using (4.18) from [45] (Page 119);
12 $k^\star = \arg\max_k |\mathscr{I}_k|$;
13 $\mathscr{I}^\star = \mathscr{I}_{k^\star}$;
14 if $|\mathscr{I}^\star| < \tau_n$ then
15     return no mirror;
16 else
17     re-estimate $\mathbf{n}_m$, $\mathbf{n}'_m$ and $d_m$ with $\mathscr{I}^\star$ by minimizing Sampson error using the
       Levenberg-Marquardt algorithm;
18     (guided matching): find correspondence inliers consistent with the optimal
       estimation;
19     return $\pi_m$;
```

6.4.3 Applying RANSAC Framework

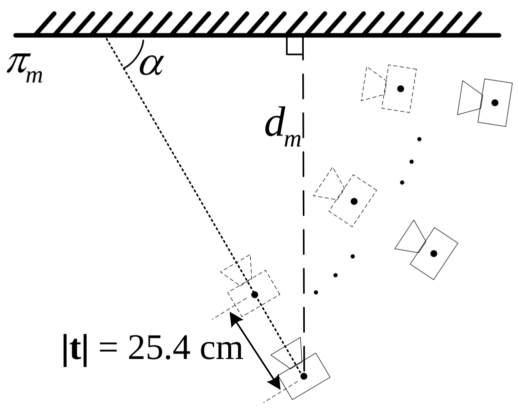
We are now ready to apply the RANSAC to the set $\mathcal{S}$ of quadruples to estimate π_m. The whole algorithm is summarized in Algorithm 6.2. There are two thresholds used: inlier-outlier threshold τ_d and mirror detection threshold τ_n. Threshold τ_d in step 9 is used to determine whether the quadruple belongs to the current inlier set. τ_d is chosen based on 8 DOFs of the decision variables. With a preset probability threshold of 0.95, $\tau_d = \sqrt{15.51\sigma^2}$ according to [45] where σ is the standard deviation of the measurement error for feature points. The algorithm returns "no mirror" when the size of maximum inlier set is smaller than τ_n (Step 15). τ_n will be determined experimentally through in-lab tests in Section 6.5.1. In step 11, the maximum sample iteration N is chosen adaptively (Page 119 of [45]). Steps 17 and 18 of Algorithm 6.2 can be iterated until the number of correspondence inliers is stable.

6.5 Experiments

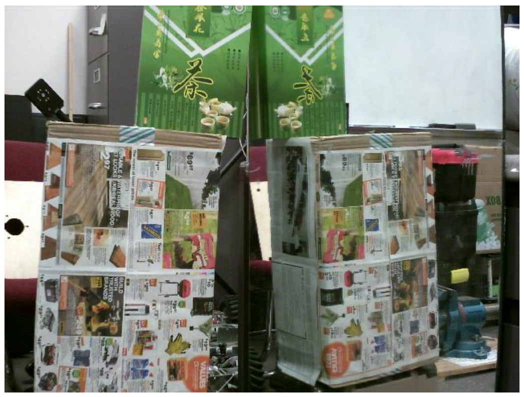
We have implemented the proposed algorithm using Matlab under a Windows 7 operating system. For the ASIFT algorithm, we use the open source implementation in [139]. Images are taken by a pre-calibrated Vivicam 7020 camera with a resolution of 640×480 pixels. We first test the algorithms in our lab to determine the algorithm accuracy under the controlled settings and to determine the proper threshold before the extensive field tests.

6.5.1 In-lab Tests

Fig. 6.3(a) illustrates the setup of in-lab tests. Define α as the angle between the camera optical axis and the mirror plane π_m. This is usually the robot approaching angle towards the mirror plane. It is important to know how α affects the estimation accuracy of π_m for the collision avoidance purpose. Data are collected in 6 different α values ranging from 5° to 60°. Scene structure is kept to be the same during the test (see Fig. 6.3(b)) with



(a)

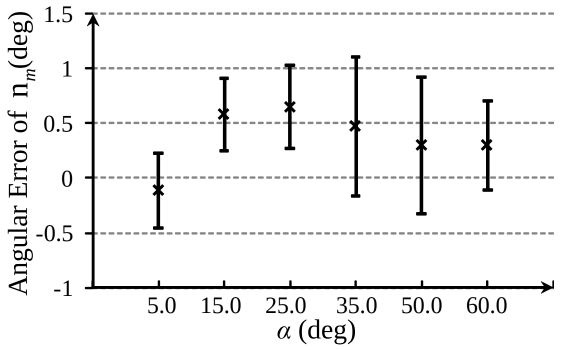

View 1 (α = 25.0°)

(b)

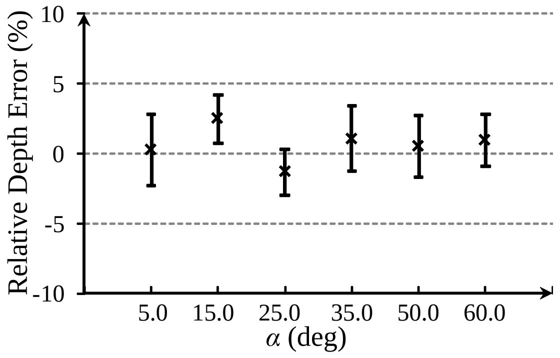
Figure 6.3: In-lab experiment setup. (a) Experiment configurations. (b) A sample view.

abundant features. The baseline distances between the first and second views are 25.4cm while maintaining the same optical axis. Ground truth data are obtained using physical measurements.

Fig. 6.4 illustrates that both angular errors of mirror plane normal and relative depth



(a)



(b)

Figure 6.4: The accuracy of estimated mirror plane with respect to α values. (a) Angular error of the mirror normal. (b) Relative depth error for the mirror plane. The vertical bar and the middle cross represent the one standard deviation range and sample mean, respectively.

errors are reasonably small under different α values. Note that 100 trials have been carried out for each α setting. The results are desirable because errors are not sensitive to α values. Note that we have not performed experiments for cases with large angle values (i.e. $\alpha > 60°$). At large angles, the camera/robot almost faces the mirror directly. Since a regular camera has a horizontal field of view larger than $55°$, the robot can see itself in the mirror. For such cases, the problem becomes trivial because it is reduced to self-appearance-based mirror detection, which is less challenging.

The second experiment is to explore the relationship between the quadruple inlier number and the estimation accuracy and hence determine threshold τ_n in Algorithm 6.2. We use the same data set from the first experiment. For every pair of images, the mirror parameters are computed each time as the number of quadruple inliers is changed through incrementally adjusting the ASIFT feature detection threshold. Then we group the estimation results according to their corresponding quadruple inlier numbers and compare the estimation error across groups. The results are shown in Fig. 6.5. As expected, the stan-

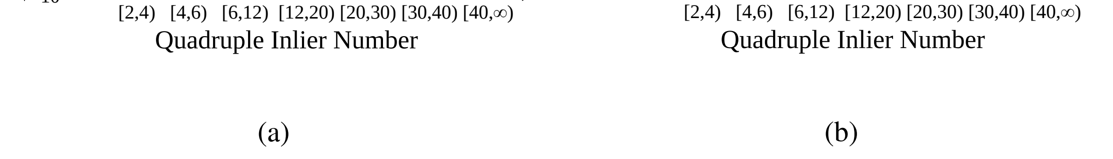


Figure 6.5: The mean and standard deviation plot of mirror plane parameters vs. number of quadruple inliers. (a) Angular error of the mirror normal. (b) Relative depth error for the mirror plane. The vertical bar represents one standard deviation range.

dard deviation of estimation generally decreases as the quadruple inlier number increases. When the quadruple inlier number drops below 6, the estimation accuracy becomes untrustable due to its large standard deviation. Hence we set $\tau_n = 6$ for our field tests.

6.5.2 Field Tests

We have tested our algorithm in the field. A data set of 100 pairs of images are collected from real world scenes with or without mirrored walls, such as gymnasiums, corridors, campus, and shopping malls (see Fig.6.6). In the data set, 50% of the image pairs contain mirrored walls such as wall mirrors, window glasses, and water surfaces.

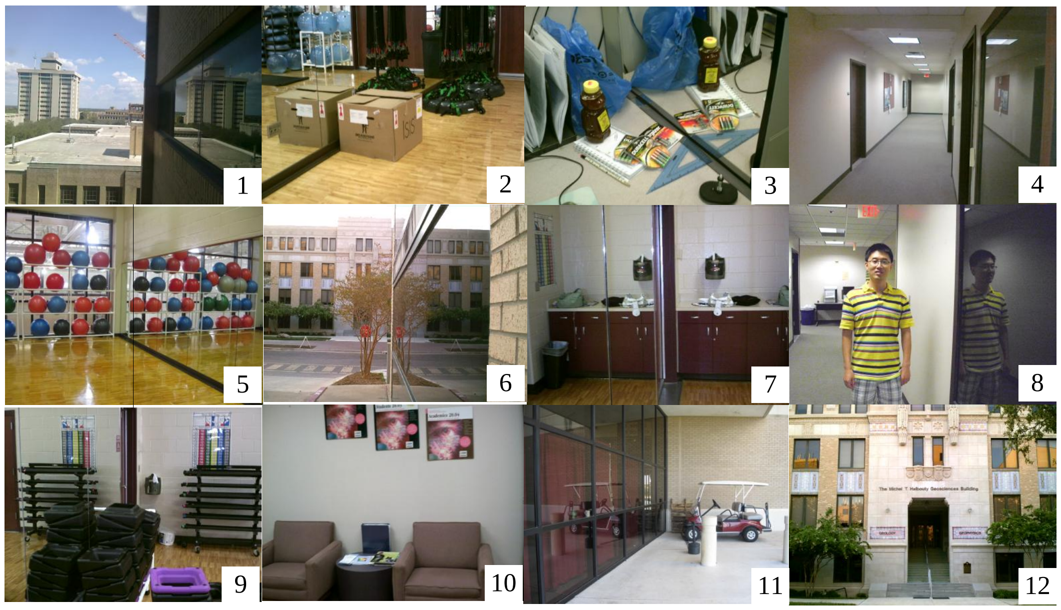


Figure 6.6: Sample images from the data set.

Table 6.1: FIELD TEST RESULTS

		Predicted	
		Positive	Negative
Actual	Positive	45	5
	Negative	4	46

The detection result is presented in a confusion matrix in Table 6.1, where “Positive” indicates the existence of mirrored walls. In the confusion matrix, true positive rate and true negative rate are both high, indicating desirable recognition ability. The false positive cases are typically caused by objects with strong symmetric appearances, e.g., sample image 12 in Fig. 6.6. The false negative cases are mainly due to lack of features in the scene. The overall detection accuracy is 91.0%.

6.6 Conclusions

We addressed PMDP using two views from an on-board camera. First, we derived geometric constraints for corresponding real-virtual features across two views. Based on the geometric constraints, we employed RANSAC framework and ASIFT to develop a robust mirror detection algorithm. We implemented the algorithm and tested it under both in-lab and field settings. The algorithm has achieved an overall accuracy of 91.0%.

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

Navigation is a fundamental problem in mobile robotics as a combination of localization, mapping, motion planning and obstacle avoidance. In particular, SLAM is the key to truly autonomous navigation. Visual navigation is an attractive alternative to traditional range sensor-based approaches because cameras are low-cost, lightweight and able to capture rich environmental information. In visual SLAM, the major challenges arise form the robustness against lighting variations and uneven feature distribution. Meanwhile, mapping with semantic landmarks is highly desirable but far from being solved. In this dissertation, we have focused on visual SLAM in urban and indoor environments, where we fuse multiple types of visual features to meet the challenges.

We proposed a new data structure MFG to organize heterogeneous landmarks, and analyzed the error propagation in a two view-based MFG construction algorithm. In the first work, our method constructed a sequence of two view-based MFGs from sequential input images, and produced facade pose estimation using EKF. The method has the advantage of being able to detect, track, and estimate 3D planes from sequential images in an on-line manner. The limitation is that MFG construction requires sufficient baselines between two views, which does not always hold.

In our second work, we devised a multiple view based MFG construction algorithm to overcome the aforementioned limitation. The new algorithm took a video stream as input, iteratively selected key frames and refined robot locations and 3D map using bundle adjustment. This advantages include 1) it does not require large baseline, and 2) the localization and mapping accuracy is improved due to the bundle adjustment framework. On the KITTI dataset our method reduced the translational error by 52.5% under urban scenarios, compared to state-of-the-art algorithms.

In the third work, we proposed a new approach for RGB-D SLAM to improve its robustness to lighting variation. The method extracted 3D points and lines from each RGB-D frame and analyzed their measurement uncertainties, which enabled maximum likelihood estimation of relative camera motion between two frames. The advantages include 1) the resulted motion has smaller estimation uncertainties and 2) the robustness against lighting change is superior. We demonstrated the advantages through both uncertainty analysis and physical experiments.

Besides visual SLAM, we also studied a special case of obstacle avoidance. Our method took two images as input and decided whether planar mirrors exist in the scene by checking geometric constraints between visual feature correspondences. The advantage of this method is that it can detect specular objects which easily fail range sensors. The limitation is that it produces false positive when symmetric objects exist.

To conclude, we have investigated feature fusion approaches to enhance visual SLAM accuracy and robustness. We designed novel data structure and algorithms to fuse heterogeneous features, using both regular and RGB-D cameras. The superiority of our feature fusion approaches was demonstrated on challenging urban and indoor datasets by comparing with state-of-the-art methods.

7.2 Future Work

Our work is just an initial step towards heterogeneous and high-level landmark based visual navigation. The following directions can be explored in the future.

- *Appearance information.* Image/video segmentation results contain rich scene structure information, which, if properly incorporated in MFG construction process, would help feature matching and plane boundary detection.

- *Acceleration.* To make MFG based SLAM run in real time, the construction algorithm should be parallelized. For example, the extraction of different feature types can be conducted in parallel.

- *Sensor fusion*. The current MFG is essentially a visual feature fusion framework. In extreme scenarios where visual features are absent, sensor fusion is necessary for robust navigation. For example, IMU can be used in combination with cameras.
- *Place recognition*. MFG can be applied to place recognition. This requires us to develop a matching algorithm that can match the heterogeneous features in MFG while taking their geometric constraints into account. The resulting matching should be more robust to perspective and lighting changes.
- *RGB-D SLAM*. Our current RGB-D SLAM algorithm only computes pairwise motion. In future, pose graph optimization or local bundle adjustment techniques can be added to our system for improving accuracy. Moreover, planes can also be fused with points and lines to generate a richer map.

REFERENCES

[1] A. Elfes, “Sonar-based real-world mapping and navigation,” *IEEE Journal of Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987.

[2] F. Bonin-Font, A. Ortiz, and G. Oliver, “Visual navigation for mobile robots: A survey,” *Journal of Intelligent & Robotic systems*, vol. 53, no. 3, pp. 263–296, 2008.

[3] H. Li, D. Song, Y. Lu, and J. Liu, “A two-view based multilayer feature graph for robot navigation,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012, pp. 3580–3587.

[4] Y. Lu, D. Song, Y. Xu, A. G. A. Perera, and S. Oh, “Automatic building exterior mapping using multilayer feature graphs,” in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2013, pp. 162–167.

[5] Y. Lu, D. Song, and J. Yi, “High level landmark-based visual navigation using unsupervised geometric constraints in local bundle adjustment,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1540–1545.

[6] Y. Lu and D. Song, “Visual navigation using heterogeneous landmarks and unsupervised geometric constraints,” *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 736–749, June 2015.

[7] Y. Lu, D. Song, H. Li, and J. Liu, “Automatic recognition of spurious surface in building exterior survey,” in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2013, pp. 1059–1064.

[8] J. J. Leonard and H. F. Durrant-Whyte, “Mobile robot localization by tracking geometric beacons,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, 1991.

[9] W. Burgard, D. Fox, D. Hennig, and T. Schmidt, “Estimating the absolute position of a mobile robot using position probability grids,” in *National Conference on Artificial Intelligence*, 1996, pp. 896–901.

[10] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, “Monte carlo localization: Efficient position estimation for mobile robots,” *AAAI/IAAI*, pp. 343–349, 1999.

[11] R. Smith, M. Self, and P. Cheeseman, “Estimating uncertain spatial relationships in robotics,” in *Autonomous Robot Vehicles*. Springer, 1990, pp. 167–193.

[12] M. Csorba and H. Durrant-Whyte, “A new approach to simultaneous localisation and map building,” in *SPIE Aerosense*, 1996.

[13] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “Fastslam: A factored solution to the simultaneous localization and mapping problem,” in *AAAI/IAAI*, 2002, pp. 593–598.

[14] S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte, “Simultaneous localization and mapping with sparse extended information filters,” *International Journal of Robotics Research*, vol. 23, no. 7-8, pp. 693–716, 2004.

[15] A. Davison, “Real-time simultaneous localisation and mapping with a single camera,” in *IEEE International Conference on Computer Vision (ICCV)*, October 2003, pp. 1403–1410.

[16] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, “Bundle adjustment – a modern synthesis,” in *Vision Algorithms: Theory and Practice*. Springer, 2000, pp. 298–372.

[17] D. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, no. 4, pp. 91–110, November 2004.

[18] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” in *European Conference on Computer Vision (ECCV)*. Springer, 2006, pp. 404–417.

[19] T. Lemaire and S. Lacroix, “Monocular-vision based SLAM using line segments,” in *IEEE International Conference on Robotics and Automation (ICRA)*, April 2007, pp. 2791–2796.

[20] E. Eade and T. Drummond, “Edge landmarks in monocular SLAM,” *Image and Vision Computing*, vol. 27, no. 5, pp. 588 – 596, 2009.

[21] J. Sola, T. Vidal-Calleja, and M. Devy, “Undelayed initialization of line segments in monocular SLAM,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 1553–1558.

[22] J. Zhang and D. Song, “Error aware monocular visual odometry using vertical line pairs for small robots in urban areas,” in *Special Track on Physically Grounded AI, AAAI*, Atlanta, Georgia, USA, July 2010.

[23] P. Smith, I. Reid, and A. Davison, “Real-time monocular SLAM with straight lines,” in *British Machine Vision Conference (BMVC)*, 2006, pp. 17–26.

[24] G. Zhang, D. H. Kang, and I. H. Suh, “Loop closure through vanishing points in a line-based monocular SLAM,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 4565–4570.

[25] A. Gee, D. Chekhlov, A. Calway, and W. Mayol-Cuevas, “Discovering higher level structure in visual SLAM,” *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 980–990, October 2008.

[26] J. Martinez-Carranza and A. Calway, “Unifying planar and point mapping in monocular SLAM,” in *British Machine Vision Conference (BMVC)*, 2010, pp. 43.1–43.11.

[27] A. Flint, C. Mei, I. Reid, and D. Murray, “Growing semantically meaningful models for visual SLAM,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 467–474.

[28] R. F. Salas-Moreno, B. Glocker, P. H. Kelly, and A. J. Davison, “Dense planar SLAM,” in *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2014, pp. 157–164.

[29] H. Strasdat, J. Montiel, and A. J. Davison, “Scale drift-aware large scale monocular SLAM,” in *Robotics: Science and Systems (RSS)*, vol. 1, no. 2, 2010, p. 4.

[30] P. Besl and N. D. McKay, “A method for registration of 3-D shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

[31] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, “KinectFusion: Real-time dense surface mapping and tracking,” in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011, pp. 127–136.

[32] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, “RGB-D mapping: Using kinect-style depth cameras for dense 3D modeling of indoor environments,” *International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.

[33] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, “An evaluation of the RGB-D SLAM system,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 1691–1696.

[34] J. Borenstein and Y. Koren, “Obstacle avoidance with ultrasonic sensors,” *IEEE Journal of Robotics and Automation*, vol. 4, no. 2, pp. 213–218, 1988.

[35] D. An and H. Wang, “VPH: A new laser radar based obstacle avoidance method for intelligent mobile robots,” in *World Congress on Intelligent Control and Automation*, vol. 5, 2004, pp. 4681–4685.

[36] A. Diosi and L. Kleeman, “Laser scan matching in polar coordinates with application to SLAM,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Alberta, Canada, August 2005, pp. 3317–3322.

[37] V. Nguyen, A. Harati, A. Martinelli, R. Siegwart, and N. Tomatis, “Orthogonal SLAM: A step toward lightweight indoor autonomous navigation,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2006, pp. 5007–5012.

[38] E. Eade and T. Drummond, “Scalable monocular SLAM,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2006, pp. 469–476.

[39] K. Konolige and M. Agrawal, “Frameslam: From bundle adjustment to real-time visual mapping,” *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, 2008.

[40] L. Chen, T. Teo, Y. Shao, Y. Lai, and J. Rau, “Fusion of LIDAR data and optical imagery for building modeling,” *International Archives of Photogrammetry and Remote Sensing*, vol. 35, no. B4, pp. 732–737, 2004.

[41] C. Rasmussen, “A hybrid vision + ladar rural road follower,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2006, pp. 156–161.

[42] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[43] G. Zhang and I. H. Sun, “Building a partial 3D line-based map using a monocular SLAM,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2011, pp. 1497 –1502.

[44] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, “LSD: A fast line segment detector with a false detection control,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, April 2010.

[45] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge Univ Pr, 2003.

[46] W. K. Newey and D. McFadden, “Large sample estimation and hypothesis testing,” in *Handbook of Econometrics*. Elsevier, 1994, vol. 4, pp. 2111–2245.

[47] P. Chen and D. Suter, “Simultaneously estimating the fundamental matrix and homographies,” *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1425–1431, 2009.

[48] C. Frueh, S. Jain, and A. Zakhor, “Data processing algorithms for generating textured 3D building facade meshes from laser scans and camera images,” *International Journal of Computer Vision*, vol. 61, no. 2, pp. 159–184, 2005.

[49] L. Zebedin, J. Bauer, K. Karner, and H. Bischof, “Fusion of feature-and area-based information for urban buildings modeling from aerial imagery,” in *European Conference on Computer Vision (ECCV)*. Springer, 2008, pp. 873–886.

[50] N. Cornelis, B. Leibe, K. Cornelis, and L. Van Gool, “3D urban scene modeling integrating recognition and reconstruction,” *International Journal of Computer Vision*, vol. 78, no. 2-3, pp. 121–141, 2008.

[51] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, “A comparison and evaluation of multi-view stereo reconstruction algorithms,” in *IEEE Conference on Computer vision and Pattern Recognition (CVPR)*, vol. 1, 2006, pp. 519–528.

[52] G. Vogiatzis, P. H. Torr, and R. Cipolla, “Multi-view stereo via volumetric graph-cuts,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2005, pp. 391–398.

[53] H. Jin, S. Soatto, and A. J. Yezzi, "Multi-view stereo reconstruction of dense shape and complex appearance," *International Journal of Computer Vision*, vol. 63, no. 3, pp. 175–189, 2005.

[54] H. Bay, V. Ferraris, and L. Van Gool, "Wide-baseline stereo matching with line segments," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2005, pp. 329–336.

[55] T. Yu, N. Xu, and N. Ahuja, "Shape and view independent reflectance map from multiple views," *International Journal of Computer Vision*, vol. 73, no. 2, pp. 123–138, 2007.

[56] D. Gallup, J.-M. Frahm, and M. Pollefeys, "Piecewise planar and non-planar stereo for urban scene reconstruction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1418–1425.

[57] J. A. Castellanos, R. Martinez-Cantin, J. D. Tardós, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 21–29, 2007.

[58] D. Hahnel, W. Burgard, D. Fox, and S. Thrun, "An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, 2003, pp. 206–211.

[59] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Generic and real-time structure from motion using local bundle adjustment," *Image and Vision Computing*, vol. 27, pp. 1178–1193, 2009.

[60] J. Civera, O. G. Grasa, A. J. Davison, and J. Montiel, "1-point RANSAC for extended Kalman filtering: Application to real-time structure from motion and visual

odometry," *Journal of Field Robotics*, vol. 27, no. 5, pp. 609–631, 2010.

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

[62] N. Carlevaris-Bianco and R. M. Eustice, "Learning visual feature descriptors for dynamic lighting conditions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 2769–2776.

[63] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?" in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 2657–2664.

[64] E. Royer, M. Lhuillier, M. Dhome, and J.-M. Lavest, "Monocular vision for mobile robot localization and autonomous navigation," *International Journal of Computer Vision*, vol. 74, no. 3, pp. 237–260, 2007.

[65] L. M. Paz, P. Piniés, J. D. Tardós, and J. Neira, "Large-scale 6-DOF SLAM with stereo-in-hand," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 946–957, 2008.

[66] G. Sibley, C. Mei, I. Reid, and P. Newman, "Vast-scale outdoor navigation using adaptive relative bundle adjustment," *International Journal of Robotics Research*, vol. 29, no. 8, pp. 958–980, 2010.

[67] J.-P. Tardif, Y. Pavlidis, and K. Daniilidis, "Monocular visual odometry in urban environments using an omnidirectional camera," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008, pp. 2531–2538.

[68] A. Pronobis, O. M. Mozos, B. Caputo, and P. Jensfelt, "Multi-modal semantic place classification," *International Journal of Robotics Research*, vol. 29, no. 2-3, pp. 298–320, 2010.

[69] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, “Automatic extrinsic calibration of vision and lidar by maximizing mutual information,” *Journal of Field Robotics*, 2014.

[70] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, “3-D mapping with an RGB-D camera,” *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 177–187, 2014.

[71] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007, pp. 225–234.

[72] S. Frintrop and P. Jensfelt, “Attentional landmarks and active gaze control for visual SLAM,” in *IEEE Transactions on Robotics, Special Issue on Visual SLAM*, vol. 24, no. 5, October 2008.

[73] J. Shi and C. Tomasi, “Good features to track,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1994, pp. 593–600.

[74] E. Tretyak, O. Barinova, P. Kohli, and V. Lempitsky, “Geometric image parsing in man-made environments,” *International Journal of Computer Vision*, vol. 97, no. 3, pp. 305–321, 2012.

[75] J.-Y. Bouguet, “Pyramidal implementation of the affine Lucas Kanade feature tracker: Description of the algorithm,” *Intel Corporation*, vol. 5, 2001.

[76] 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, 2004.

[77] B. Fan, F. Wu, and Z. Hu, “Line matching leveraged by point correspondences,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010, pp. 390–397.

[78] C. Schmid and A. Zisserman, "Automatic line matching across views," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 1997, pp. 666–671.

[79] R. Eustice, O. Pizarro, and H. Singh, "Visually augmented navigation in an unstructured environment using a delayed state history," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2004, pp. 25–32.

[80] V. Indelman, R. Roberts, C. Beall, and F. Dellaert, "Incremental light bundle adjustment." in *British Machine Vision Conference (BMVC)*, 2012, pp. 1–11.

[81] Y. Xu, S. Oh, and A. Hoogs, "A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 1376–1383.

[82] K. Ueda and N. Yamashita, "On a global complexity bound of the Levenberg-Marquardt method," *Journal of Optimization Theory and Applications*, vol. 147, no. 3, pp. 443–453, 2010.

[83] Y. Lu and D. Song, "Multilayer feature graph for visual navigation," http://telerobot.cs.tamu.edu/MFG, 2014.

[84] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos, "Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets," in *IROS Workshop on Benchmarks in Robotics Research*, 2006.

[85] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *International Journal of Computer Vision*, vol. 81, no. 2, pp. 155–166, 2009.

[86] J.-L. Blanco-Claraco, F.-Á. Moreno-Dueñas, and J. González-Jiménez, "The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario," *International Journal of Robotics Research*, vol. 33, no. 2, pp. 207–214, 2014.

[87] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3607–3613.

[88] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 3354–3361.

[89] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3D reconstruction in real-time," in *IEEE Intelligent Vehicles Symposium*, 2011, pp. 963–968.

[90] S. Song, M. Chandraker, and C. C. Guest, "Parallel, real-time monocular visual odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 4698–4705.

[91] C. Kerl, J. Sturm, and D. Cremers, "Dense visual SLAM for RGB-D cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 2100–2106.

[92] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2004, pp. 652–659.

[93] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual SLAM: Why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[94] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[95] F. Steinbrucker, J. Sturm, and D. Cremers, "Real-time visual odometry from dense RGB-D images," in *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2011, pp. 719–722.

[96] M. Meilland, A. Comport, and P. Rives, "Real-time dense visual tracking under large lighting variations," in *British Machine Vision Conference (BMVC)*, vol. 29, 2011.

[97] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2320–2327.

[98] S. Klose, P. Heise, and A. Knoll, "Efficient compositional approaches for real-time robust direct visual odometry from RGB-D data," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 1100–1106.

[99] T. Whelan, H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald, "Robust real-time visual odometry for dense RGB-D mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 5724–5731.

[100] A. Ranganathan, S. Matsumoto, and D. Ilstrup, "Towards illumination invariance for visual localization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 3791–3798.

[101] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, "Point-plane SLAM for hand-held 3D sensors," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 5182–5189.

[102] C. Raposo, M. Lourenço, J. P. Barreto, and M. Antunes, "Plane-based odometry using an RGB-D camera," in *British Machine Vision Conference (BMVC)*, 2013.

[103] T.-k. Lee, S. Lim, S. Lee, S. An, and S.-y. Oh, "Indoor mapping using planes extracted from noisy RGB-D sensors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 1727–1733.

[104] C. Choi, A. J. Trevor, and H. I. Christensen, "RGB-D edge detection and edge-based registration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 1568–1575.

[105] J. Smisek, M. Jancosek, and T. Pajdla, "3D with Kinect," in *IEEE International Conference on Computer Vision (ICCV) Workshops*, 2011, pp. 1154–1160.

[106] Z. Wang, F. Wu, and Z. Hu, "MSLD: A robust descriptor for line matching," *Pattern Recognition*, vol. 42, no. 5, pp. 941–953, 2009.

[107] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 5, pp. 698–700, 1987.

[108] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 4, pp. 376–380, 1991.

[109] Z. Zhang and O. D. Faugeras, "Determining motion from 3D line segment matches: A comparative study," *Image and Vision Computing*, vol. 9, no. 1, pp. 10–19, 1991.

[110] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[111] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 2564–2571.

[112] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of RGB-D SLAM systems,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 573–580.

[113] “RGBDSLAM trajectories,” https://svncvpr.in.tum.de/cvpr-ros-pkg/trunk/rgbd_benchmark/rgbd_benchmark_tools/data/rgbdslam/.

[114] W. G. Walter, “An imitation of life,” *Scientific American*, vol. 182, no. 2, pp. 42–45, 1950.

[115] G. Gallup, “Chimpanzees: Self-recognition,” *Science*, vol. 167, no. 3914, pp. 86–87, January 1970.

[116] F. Patterson and W. Gordon, “The case for the personhood of gorillas,” *The Great Ape Project: Equality Beyond Humanity. London: Fourth Estate*, pp. 58–77, 1993.

[117] D. Reiss and L. Marino, “Mirror self-recognition in the bottlenose dolphin: A case of cognitive convergence,” *PNAS*, vol. 98, no. 10, pp. 5937–5942, May 2001.

[118] H. Prior, A. Schwarz, and O. Gntrkn, “Mirror-induced behavior in the magpie (pica pica): Evidence of self-recognition,” *PLoS Biol*, vol. 6, no. 8, p. e202, 2008.

[119] M. Hauser, C. Miller, K. Liu, and R. Gupta, “Cotton-top tamarins (saguinus oedipus) fail to show mirror-guided self-exploration,” *American Journal of Primatology*, vol. 53, no. 3, pp. 131–137, 2001.

[120] P. Michel, K. Gold, and B. Scassellati, “Motion-based robotic self-recognition,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, September 2004, pp. 2763–2768.

[121] P. Haikonen, “Reflections of consciousness: The mirror test,” in *AAAI Fall Symposium on Consciousness and Artificial Intelligence*, Arlington, Virginia, USA, November 2007, pp. 67–71.

[122] J. Gluckman and S. Nayar, “Planar catadioptric stereo: Geometry and calibration,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, June 1999, pp. 22–28.

[123] ——, “Catadioptric stereo using planar mirrors,” *International Journal of Computer Vision*, vol. 44, no. 1, pp. 65–79, 2001.

[124] G. Mariottini, S. Scheggi, F. Morbidi, and D. Prattichizzo, “Planar catadioptric stereo: Single and multi-view geometry for calibration and localization,” in *IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009, pp. 1510–1515.

[125] K. Reiner and K. Donner, “Stereo vision on specular surfaces,” in *IASTED International Conference on Visualization, Imaging and Image processing*, Marbella, Spain, September 2004, pp. 335–339.

[126] K. N. Kutulakos and E. Steger, “A theory of refractive and specular 3D shape by light-path triangulation,” *International Journal of Computer Vision*, vol. 76, no. 1, pp. 13–29, January 2008.

[127] S. Rozenfeld, I. Shimshoni, and M. Lindenbaum, “Dense mirroring surface recovery from 1D homographies and sparse correspondences,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 325–337, February 2011.

[128] M. Ferraton, C. Stolz, and F. Meriaudeau, “Surface reconstruction of transparent objects by polarization imaging,” in *IEEE International Conference on Signal Image Technology and Internet Based Systems*, Bali, Indonesia, November 2008, pp. 474–479.

[129] D. Miyazaki, M. Kagesawa, and K. Ikeuchi, “Transparent surface modeling from a pair of polarization images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 1, pp. 73–82, January 2004.

[130] S. Rahmann and N. Canterakis, “Reconstruction of specular surfaces using polarization imaging,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Kauai, Hawaii, USA, December 2001, pp. 149–155.

[131] O. Morel, C. Stolz, F. Meriaudeau, and P. Gorria, “Active lighting applied to 3D reconstruction of specular metallic surfaces by polarization imaging,” *Applied Optics*, vol. 45, no. 17, pp. 4062–4068, January 2006.

[132] M. Oren and S. K. Nayar, “A theory of specular surface geometry,” *International Journal of Computer Vision*, vol. 24, no. 2, pp. 105–124, September 1997.

[133] R. Ma, J. Chen, and Z. Su, “MI-SIFT: Mirror and inversion invariant generalization for SIFT descriptor,” in *ACM International Conference on Image and Video Retrieval*, Xi’an, China, July 2010, pp. 228–235.

[134] X. Guo and X. Cao, “FIND: A neat flip invariant descriptor,” in *International Conference on Pattern Recognition*, Istanbul, Turkey, August 2010, pp. 515–518.

[135] J. Matas, O. Chum, M. Urban, and T. Pajdla, “Robust wide-baseline stereo from maximally stable extremal regions,” *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, September 2004.

[136] K. Mikolajczyk and C. Schmid, “Scale & affine invariant interest point detectors,” *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.

[137] J. Morel and G. Yu, “ASIFT: A new framework for fully affine invariant image comparison,” *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 438–469, April 2009.

[138] A. Agha-mohammadi and D. Song, "Robust recognition of planar mirrored walls using a single view," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 1186–1191.

[139] G. Yu and J.-M. Morel, "ASIFT: An algorithm for fully affine invariant comparison," http://www.ipol.im/pub/algo/my_affine_sift/, 2011.