Fakultät für Informatik
Technische Universität München

Globally Optimal Solutions for Unit-Norm Constrained Computer Vision Problems

Yinlong Liu

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender:
Prof. Dr. Martin Bichler

Prüfende der Dissertation:
1. Prof. Dr.-Ing. habil. Alois C. Knoll
2. Prof. Dr. Guang Chen, Tongji University

Die Dissertation wurde am 03.02.2022 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 13.07.2022 angenommen.

Abstract

In many safety-critical computer vision applications, it is mostly desired to seek the optimal solutions with provable guarantee in presence of noise and outliers, since local optimums may lead to serious failures. Therefore, in this thesis, globally optimal solutions for unit-norm constrained computer vision problems are investigated. Specifically, the globally optimal solutions are provided by the branch-and-bound algorithm, which is a deterministic global optimization algorithm. Moreover, to obtain global optimums for unit-norm constrained optimization problems, we explore the geometry of the unit-norm constraint and introduce a general inequality for n-sphere.

Based on the introduced general inequality and the branch-and-bound algorithm, three different unit-norm constrained computer vision tasks are studied to seek globally optimal solutions in this thesis. Specifically,

1. Globally optimal solution for estimating vertical direction from the Atlanta world. This work is about globally estimating the unique vertical direction in Atlanta world. Compared with the state-of-the-art methods, it has two advantages: (1) avoiding the curse of dimensionality in Atlanta world; (2) avoiding manual adjustment of the number of horizontal directions. Methodologically, the contributions are mainly as follows: (1) A novel global searching method for estimating vertical direction is proposed. It is different from conventional rotation search. Since the domain of the vertical directions is inherently in the unit sphere, the proposed searching method is more efficient in vertical direction estimation. (2) Three novel different bounds for branch-and-bound algorithm are derived. To the best of our knowledge, it is the first to propose such bounds in the unit sphere to the structural world frame estimation problem.

2. Globally optimal solution for camera orientation estimation from 2D-3D line feature correspondences. This work is concerned with the problem of estimating camera orientation from a set of 2D/3D line correspondences, which is a major part of the Perspective-n-Line (PnL) problem. The RANSAC algorithm is the de facto standard for solving outlier-contaminated PnL problems. However, RANSAC cannot produce a reasonable result with a provable guarantee. Therefore, a PnL algorithm that could obtain a certifiably optimal solution from outlier-contaminated data is highly needed. We take a big step towards this goal. Specifically, we first decouple camera orientation and position, then a globally optimal camera orientation estimation algorithm is investigated.

3. Globally optimal solution for camera relative pose estimation with known vertical direction. Recently, there has been a surge of interest in using prior gravity direction to solve traditional robot vision problems. However, most of them focus on solving outlier-free problems. To obtain a robust solution from outlier-contaminated inputs, we propose a globally optimal algorithm for relative pose estimation with known gravity direction. The proposed method employs the branch-and-bound algorithm to solve a consensus maximization problem, and thus it is able to obtain the global solution with a provable guarantee. The proposed algorithm has important potential to be used in some safety-demand applications.

Zusammenfassung

In vielen sicherheitskritischen Computer Vision Anwendungen ist es erwünscht nachweisbar optimale Lösungen mit einer Optimalitätsgarantie bei vorhandenem Rauschen und Ausreißern in den Daten zu finden, da lokale Optima zu gravierenden Ausfällen führen können. Deswegen, erforscht diese Dissertation global optimale Lösungen für Unit-norm Constrained Computer Vision Probleme. Genauer gesagt werden die global optimalen Lösungen mithilfe eines Branch-and-Bound Algorithmus, welcher ein deterministischer global optimaler Optimierungsalgorithmus ist, gefunden. Des Weiteren, untersuchen wir die Geometrie der Einheitsnormbeschränkung und stellen eine generelle Ungleichung für die n-Domaine vor, um globale Optima für Einheitsnorm-beschränkte Optimierungsprobleme zu erhalten.

Basierend auf der vorgestellten generellen Ungleichung und dem Branch-and-Bound Algorithmus, werden in dieser Dissertation drei verschiedene Einheitsnormbeschränkte Computer Vision Aufgaben bezüglich global optimaler Lösungen untersucht. Insbesondere:

1. Eine global optimale Lösung zu Abschätzung der vertikalen Richtung aus der Atlanta Welt. Diese Arbeit befasst sich mit der globalen Abschätzung der eindeutigen vertikalen Richtung in der Atlanta Welt. Verglichen mit anderen Methoden auf dem Stand der Technik gibt es zwei Vorteile: (1) Der Fluch der Dimensionalität in der Atlanta Welt und (2) manuelles Anpassen der Anzahl der horizontalen Richtungen werden vermieden. Die Arbeit leistet folgende Beiträge zur Forschung: (1) Eine neue globale Suchmethode zur Abschätzung der vertikalen Richtung wird vorgeschlagen. Diese Suchmethode unterscheidet sich von konventionellen Rotationssuchen. Da die Domäne der vertikalen Richtungen inhärent die Einheitskugel ist, ist die vorgestellte Suchmethode effizienter darin die vertikaler Richtung abzuschätzen. (2) Drei neue unterschiedliche Branch-and-Bound Algorithmen werden abgeleitet. Nach unserem besten Wissen ist dies der erste Vorschlag für derartige Schranken in der Einheitskugel für das structural world frame estimation Problem.

2. Eine global optimale Lösung für die Abschätzung der Kameraorientierung mithilfe von 2D-3D Linienkorrespondenzen. Diese Arbeit beschäftigt sich mit dem Problem der Abschätzung der Kameraorientierung mithilfe einer Menge von 2D-3D Linienkorrespondenzen. Dies ist ein wesentlicher Teil des Perspective-n-Line (PnL) Problems. Der RANSAC Algorithmus ist die de facto Standardlösung für PnL Probleme mit Ausreißern. Allerd-

ings liefert RANSAC keine sinnvollen Ergebnisse mit nachweisbarer Optimalitätsgarantie. Deswegen besteht Bedarf für einen PnL Algorithmus, welcher eine nachweisbar optimale Lösung für Daten mit Ausreißern erzielt. Wir gehen einen großen Schritt in Richtung dieses Ziels. Konkret entkoppeln wir erst die Kameraorientierung und -position und ermitteln dann einen Algorithmus zur global optimalen Abschätzung der Kameraorientierung.

3. Eine global optimale Lösung für relative Kamerapositionsabschätzung mit bekannter vertikaler Richtung. In letzter Zeit wuchs das Interesse daran Vorwissen über die Gravitationsrichtung zu verwenden um traditionelle Robot Vision Probleme zu lösen. Allerdings liegt der Fokus der meisten Methoden darauf ausreißerfreie Probleme zu lösen. Um robuste Lösungen von Inputdaten mit Ausreißern zu erhalten, stellen wir einen global optimalen Algorithmus für relative Kamerapositionsabschätzung mit bekannter Gravitationsrichtung vor. Die vorgeschlagene Methode verwendet den Branch-and-Bound Algorithmus um ein consensus maximization Problem zu lösen und ist deshalb in der Lage eine global optimale Lösung mit nachweisbarer Optimalitätsgarantie zu erzielen. Deshalb hat der vorgeschlagene Algorithmus Potential zur Verwendung in sicherheitskritischen Anwendungen.

Acknowledgment

First of all, I would like to express my deepest gratitude to my supervisor, Prof. Alois Knoll, for providing an opportunity to pursue the Ph.D. degree and for all his invaluable advice, support, and help during my Ph.D. study. Additionally, I would like to extend my sincere thanks to Prof. Guang Chen, for his continuous support, encouragement. Without their assistance in every step throughout my doctoral study, this thesis would have never been possible.

I would like to thank Yiru Wang and Treffler Judith. I have been fortunate enough to work with them. These collaborations have had direct and indirect impacts on making this thesis a reality.

I would like to thank all the members of the Chair of Robotics, Artificial Intelligence and Real-time Systems. It is their kind help and support that have made my study and life in Germany a wonderful time. Special thanks to Dr. Alexander Lenz, Amy Bücherl and Ute Lomp for their support in administrative tasks.

I would like to thank Yingbai Hu, Liguo Zhou, and Wenjun Liu. We shared an office room during most of my doctoral study period. Thanks for their company and support, especially during the COVID-19 outbreak. Besides, I learn a lot from them, especially in writing and reviewing papers.

I am grateful to my parents and numerous friends for their constant love and support during my doctoral study.

Last, I would like to thank Chinese Scholarship Council (CSC) for funding my Ph.D. study in Germany.

Contents

List of Figures

List of Tables

List of Symbols

In this part, the definitions and notations used in the thesis are introduced:

$\mathbb{R}$	real scalar number space
$\mathbb{R}^n$	n-dimensional real space
$\mathbb{S}^n$	the n-dimensional unit sphere in $\mathbb{R}^{n+1}$, a.k.a. n-sphere
$\mathbb{S}^{n+}$	the upper half n-sphere
$\mathbb{SO}(3)$	the special orthogonal group of order 3
$\mathbb{B}$	the branch space in the BnB algorithm
$a \in \mathbb{R}$	real scale number
$\boldsymbol{a} \in \mathbb{R}^n$	real $n \times 1$ vector
$\lvert a \rvert$	the absolute value of a scale number a
$\lVert \boldsymbol{a} \rVert$	L_2 norm of vector $\boldsymbol{a}$, $\lVert \boldsymbol{a} \rVert = \sqrt{\boldsymbol{a}^T \boldsymbol{a}}$
$\mathbf{R} \in \mathbb{SO}(3)$	3-dimensional rotation
$\boldsymbol{s} \in \mathbb{S}^3$	unit quaternion, $\boldsymbol{s} = a + b\boldsymbol{i} + c\boldsymbol{j} + d\boldsymbol{k}$ and $\lVert \boldsymbol{s} \rVert = 1$
$\angle(\boldsymbol{a}, \boldsymbol{b})$	the angle between vector $\boldsymbol{a}$ and $\boldsymbol{b}$, $\angle(\boldsymbol{a}, \boldsymbol{b})$=arccos $\left(\frac{\boldsymbol{a}^T \boldsymbol{b}}{\lVert \boldsymbol{a} \rVert \lVert \boldsymbol{b} \rVert} \right)$
$\angle(\mathbf{R}_1, \mathbf{R}_2)$	the angle distance between rotation $\mathbf{R}_1$ and $\mathbf{R}_2$
$d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2)$	the angle distance between two unit quaternions
$\mathbf{A}$	real matrix
$\mathbf{A}^T$	the transpose of matrix $\mathbf{A}$

$Tr(\mathbf{A})$	the trace of $\mathbf{A}$
$\mathcal{P}$	point set
$\mathcal{M}$	input measurement set
$\mathbb{I}(\cdot)$	indicator function
r	residual
ϵ	inlier threshold

List of Abbreviations

AI	Artificial Intelligence
SVD	Singular Value Decomposition
RANSAC	RANdom SAmple Consensus
EM	Expectation Maximization
IRLS	Iteratively Reweighted Least Squares
BnB	Branch-and-Bound
qBnB	Quasi Branch-and-Bound
SLAM	Simultaneous Localization And Mapping
SFM	Structure From Motion
PnL	Perspective-n-Line
PnP	Perspective-n-Point
PnPL	Perspective-n-Point-and-Line
IMU	Inertial Measurement Unit
GPS	Global Positioning System
SDP	Semidefinite Programming
MILP	Mixed-Integer Linear Programming
TLS	Truncated Least Squares
EGI	Extended Gaussian Image

List of Publications

This thesis is in part results of the work presented in the following papers:

- **Yinlong Liu**, Guang Chen, and Alois Knoll. Globally Optimal Vertical Direction Estimation in Atlanta World. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020).DOI: 10.1109/TPAMI.2020.3027047
- **Yinlong Liu**, Guang Chen, Rongqi Gu and Alois Knoll, Globally Optimal Consensus Maximization for Relative Pose Estimation with Known Gravity Direction. in IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 5905-5912, July 2021.
- **Yinlong Liu**, Guang Chen and Alois Knoll, Globally Optimal Camera Orientation Estimation from Line Correspondences by BnB algorithm. in IEEE Robotics and Automation Letters, vol. 6, no. 1, pp. 215-222, Jan. 2021.

Other publications during doctoral study:

- **Yinlong Liu**, Xuechen Li, et al. A Novel Method for the Absolute Pose Problem with Pairwise Constraints. Remote Sensing 11.24 (2019): 3007.
- **Yinlong Liu**, Yiru Wang, et al. Globally Optimal Linear Model Fitting with Unit-Norm Constraint. International Journal of Computer Vision (IJCV), 2022.
- Fan Lu, Guang Chen, **Yinlong Liu**, Zhongnan Qu and Alois Knoll. RSKDD-Net: Random Sample-based Keypoint Detector and Descriptor. Advances in Neural Information Processing Systems (NeurIPS) 2020.
- Dong, Jinhu, **Yinlong Liu**, Lixuan Tang, Guang Chen, and Alois Knoll. A Fast and Robust 2D LiDAR Alignment Method by Motion Decoupling. In 2020 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), pp. 21-26. IEEE, 2020.
- Guang Chen, **Yinlong Liu**, Jinhu Dong, Lijun Zhang, Haotian Liu, Bo Zhang, Alois Knoll, "Efficient and Robust Line-Based Registration Algorithm for Robot Perception under Large-Scale Structural Scenes. IEEE ICARM 2021.
- Fan Lu, Guang Chen, Sanqing Qu, Zhijun Li, **Yinlong Liu** and Alois Knoll. PointINet: Point Cloud Frame Interpolation Network. Association for the Advancement of Artificial Intelligence (AAAI) 2021.

Chapter 1

Introduction

1.1 Safety-Critical Systems and Computer Vision Algorithms

A safety-critical system(or life-critical system) [1–3] is a system whose failure or malfunction may lead to consequences that are determined to be unacceptable [1]. For example, self-driving system [4, 5] is a typical safety-critical system, because if we cannot provide the safety guarantee for the system, it might lead to very serious failures, even the death or serious injury to people [6, 7]. According to the news reports(see Fig. 1.1 [8]), as of Dec. 2021, there have been 10 verified fatalities involving Tesla's Autopilot, which is a famous self-driving system, and we can see the details from this webpage[1]. Therefore, to meet the safety demand, the safety-critical systems need extremely reliable solutions. Accordingly, the optimization algorithms in safety-critical systems should provide the provably optimal solutions with guarantees [9].

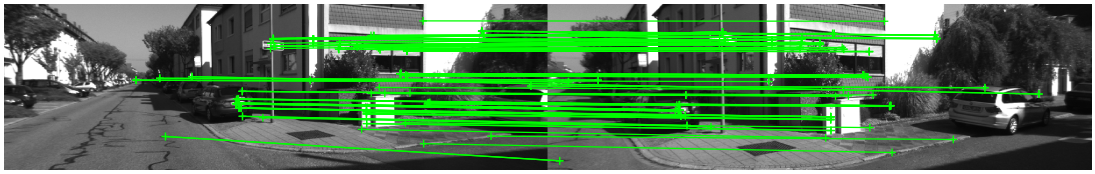
Figure 1.1: A Tesla electric car crashed into a highway barrier, California, on March 23, 2018 and investigators confirmed that Autopilot was partially to blame.

Computer vision, which is an important field of artificial intelligence (AI), seeks to develop techniques to help computer systems derive meaningful information from visual inputs [10, 11]. Nowadays, computer vision techniques play an increasingly important role in many modern tasks, such as intelligent traffic systems [12, 13], robot systems [14, 15], assisting in medical diagnosis and

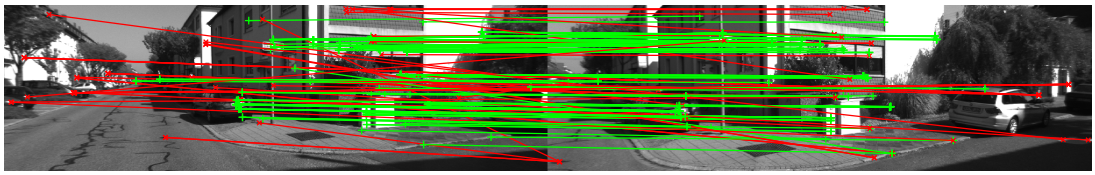
[1] `https://www.tesladeaths.com/`

treatment [16–18]. Naturally, the computer vision techniques are also widely applied in many safety-critical systems [5]. For example, there are many computer vision algorithms have been proposed to solve various visual perception problems in self-driving systems [4, 19], such as semantic segmentation [20], motion estimation [11, 21] and object detection [22, 23].

(a) Given two images of a scene, one important task is to estimate the relative camera pose.



(b) The relative pose can be solved from the established correspondences according to epipolar geometry [11]. '+' denotes the salient point feature. Green lines denote the feature correspondences.



(c) In real applications, mismatches are not able to be avoided and they may bias the estimation results significantly. '+' denotes the salient point feature. Green lines denote the true feature correspondences and red lines denote the mismatches.

Figure 1.2: Relative pose estimation in visual perception subsystem for self-driving car. In real applications, extremely reliable visual perception algorithms are highly needed to obtain the optimal solutions from the given outlier-contaminated input data [24].

In order to explain the role of computer vision techniques in safety-critical systems, we take the motion estimation in the self-driving system for example. Specifically, a calibrated camera is mounted on a moving car and it will record the visual images of environment surrounding the car. The task is to estimate the relative pose of the moving camera and thereby the self relative motion of the car from the images obtained by the calibrated camera. It is one of the core parts for visual perception subsystem of the self-driving system [25]. It should be mentioned that many high-level planning and decision in self-driving system highly rely the information provided by visual perception subsystem. If a bad motion estimation solution is given, the self-driving system may make a inappropriate decision, which may lead to very serious failures [9]. Therefore, to meet the safety demand, it is highly needed that the computer vision algorithms should provide correct solutions with provable guarantee.

Typically, to estimate the relative pose from two given images, the first step is to establish the correspondences of the salient features(e.g., keypoints) in the two images(see Fig. 1.2). Once the correspondences are established, the relative pose can be efficiently solved by classical computer vision algorithms, such as 5-point algorithm and 8-point algorithm [11, 26, 27]. However, in real applications, it is almost impossible to have the feature correspondences without mismatches [28](see Fig. 1.2). Theoretically, the mismatches can be considered as the infamous outliers in robust estimation problems [29–31]. It is well known that even one outlier can bias the estimation results significantly [32, 33]. To obtain the robust solution in the presence of outliers, robust estimation algorithms are needed to be explored. The de facto standard, which can return robust solution from outlier-contaminated inputs, is to embed outlier-free algorithms into RANSAC(*RANdom SAmple consensus*) framework [11, 34]. Unfortunately, RANSAC is a *non-deterministic* algorithm and it will provide a reasonable result only with *a certain probability* [35]. In other words, RANSAC may provide an unsatisfactory solution occasionally [29]. Consequently, the visual perception subsystem may return incorrect information, which thereby may be a risk for the self-driving system. Therefore, it is highly needed that the relative pose estimation algorithm can exact the extremely reliable solution from the outlier-contaminated correspondences. From the perspective of optimization, it is demanded to obtain the optimal relative pose with provable guarantee [29].

In summary, the computer vision technologies are widely applied in safety-critical systems, especially as the core parts in many visual perception subsystems, which certainly need to provide safety guarantee. However, the existence of outliers, which are almost unavoidable in real applications, will introduce the risks to safety-critical systems. Unfortunately, traditional computer vision algorithms usually cannot provide the correct solutions with provable guarantee from outliers-contaminated inputs [29]. Therefore, in this thesis, we explore globally optimal solutions to some computer vision problems for safety-critical applications. Specifically, the globally optimal algorithm is deterministic global optimization algorithm [36], which can provide the optimal solution with theoretical guarantees that the reported solution is indeed the global one (the best one).

1.2 Robust Objective Functions

In practical computer vision applications, it is rare that the input measurements are perfect. Noise and outliers are usually unavoidable and they are usually everywhere [29]. In addition, the outlier may introduce serious risks to visual perception system. A natural idea is to reject or remove all the outliers in advance and then to estimate parameters [29, 37]. However, it turns out the idea is mathematically intractable. The authors in [30] point that even for a simple linear instance of outlier rejection is inapproximable:

> *In the worstcase, there exist no quasi-polynomial algorithm that can compute (even an approximate) solution to the outlier rejection problem. [30]*

It is should be mentioned that this conclusion does not imply that we cannot remove all the outliers. It emphasizes that removing all the outliers is inefficient, i.e., it is time-consuming. Theoretically, removing outliers in advance is actually as difficult as solving the original outlier-robust estimation problem [29, 30].

(a) False point correspondences (mismatches) can be considered as outliers and usually cannot be avoidable.

(b) Epipolar geometry. $\mathbf{R} \in \mathbb{SO}(3)$ and $\boldsymbol{t}$ are to-be-solved relative pose.

Figure 1.3: Epipolar geometry in relative pose estimation.

Since rejecting outliers is inapproximable, to obtain robust solution, the outliers should be suppressed. Mathematically, the real input measurements $\{\boldsymbol{m}_i\}_{i=1}^{M}$ can be modeled as [38]

$$\begin{cases} \boldsymbol{m}_i = \boldsymbol{m}_i^{gt} + \boldsymbol{\kappa}_i, & \text{if } \boldsymbol{m}_i \text{ is an inlier} \\ \boldsymbol{m}_i = \boldsymbol{o}_i, & \text{if } \boldsymbol{m}_i \text{ is an outlier} \end{cases} \tag{1.1}$$

where $\boldsymbol{m}_i^{gt}$ is a *true* measurement and $\boldsymbol{\kappa}_i$ is the inlier measurement noise [38]; $\boldsymbol{o}_i$ is an outlier, which may be arbitrarily far from the true measurement. Note that $\boldsymbol{o}_i$, $\boldsymbol{\kappa}_i$ and $\boldsymbol{m}_i$ are the same data structures.

For example, in relative pose estimation, let $\{\boldsymbol{p}_i, \boldsymbol{q}_i\}_{i=1}^{M}$ denote the keypoint correspondences, which may include mismatches(see Fig. 1.3). $\mathbf{R} \in \mathbb{SO}(3)$ and $\boldsymbol{t}$ are to-be-solved relative pose. According to epipolar geometry [11], the residue for *absolutely true* correspondence is

$$r_i\left(\boldsymbol{p}_i, \boldsymbol{q}_i\right) = \left|\boldsymbol{p}_i^T [\boldsymbol{t}]_\times \mathbf{R} \boldsymbol{q}_i\right| = 0 \tag{1.2}$$

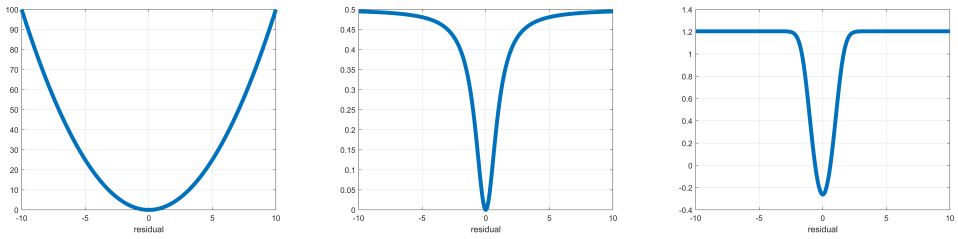
where $[\boldsymbol{t}]_\times$ is the matrix representation of the cross product with $\boldsymbol{t}$, and $|\cdot|$ is absolute value function. In practical applications, given a reasonable inlier threshold ϵ, we can define

$$\begin{cases} r_i = \left|\boldsymbol{p}_i^T [\boldsymbol{t}]_\times \mathbf{R} \boldsymbol{q}_i\right| \leq \epsilon & \text{, correspondence } \{\boldsymbol{p}_i, \boldsymbol{q}_i\} \text{ is inlier} \\ r_i = \left|\boldsymbol{p}_i^T [\boldsymbol{t}]_\times \mathbf{R} \boldsymbol{q}_i\right| > \epsilon & \text{, correspondence } \{\boldsymbol{p}_i, \boldsymbol{q}_i\} \text{ is outlier} \end{cases} \tag{1.3}$$

Notably, the noise is usually modeled as additive white Gaussian noise [39, 40]. M-estimation [41], which is generalizing maximum likelihood estimation, is usually applied to suppress Gaussian noise and estimate the parameters [42].

$$\min \sum_{i=1}^{M} \rho\left(r_i\left(\boldsymbol{p}_i, \boldsymbol{q}_i\right)\right) \tag{1.4}$$

where $\rho(\cdot)$ is an M-estimator function. Typical M-estimators are shown in Fig. 1.4 [43]. It is well known that, in Fig. 1.4(a), the least square estimator is not robust to outliers. In contrast, the redescending estimators [44] (b) and (c) are more robust to outliers.



(a) Least square **(b)** German-McClure **(c)** Blake-Zisserman

Figure 1.4: Typical M-estimators. (a) Least square loss function $\rho(x) = x^2$. (b) German-McClure loss function $\rho(x) = \dfrac{x^2/2}{1+x^2}$. (c) Blake-Zisserman loss function $\rho(x) = -\log\left(e^{-x^2} + \epsilon_0\right)$, where ϵ_0 is a to-be-tuned parameter.

Specifically, to suppress outliers, the robust M-estimators attempt to reduce the influence of outliers. Therefore, the minimum solution of Eq. (1.4) is the to-be-solved relative pose. In other words, relative pose estimation problem can be modeled as a optimization problem, whose minimum solution is the optimal relative pose. Unfortunately, the robust objective functions constructed by the redescending estimators are usually non-convex, which means there are many local optimums in solving robust objective functions [29]. The standard algorithm to find the maximum likelihood estimation is iteratively reweighted least squares (IRLS) [45, 46]. However, IRLS only guarantees to find a local optimum [29, 43], which may be unacceptable in real applications.

In fact, there are two sub problems here: (a) which measurements are inliers? (b) what are the to-be-estimated model parameters? If we can solve one sub problem, then we can solve the other one. However, both sub problems are unknown, which is a well-known chicken-and-egg problem [47]. Usually, expectation maximization(EM [48]) type algorithms are applied to solve chicken-and-egg problems [49]. Specifically, the key idea of EM-type methods is that it starts from an initial solution and solves the original problem by alternately solving the two sub problems. Notably, IRLS can be considered as one of EM type algorithms [50]. It starts from an initial model parameters (usually returned by ordinary least square method) to determine which observations are inliers with a

soft "weight", then it alternatively solve "model parameters" and "weight of each observations" problems until convergence. Theoretically, EM-type algorithms are local methods, which guarantee to return a local optimum, which maybe an unsatisfactory solution for real safety-critical applications [49, 51]

Consensus Maximization. In addition to robust M-Estimators, there are still many robust loss functions, which are able to suppress outliers. One of the most widely applied robust objective functions is inlier set maximization [52, 53], also known as consensus maximization [29, 54]. Mathematically, we can think that the consensus maximization applies the 0-1 loss function

$$\rho(x) = \begin{cases} 1, & |x| \leq \epsilon \\ 0, & others \end{cases} \tag{1.5}$$

For example, in relative pose estimation, the objective function (i.e., inlier maximization) can be formulated as

$$\max \sum_{i=1}^{M} \mathbb{I}\left(r_i\left(\boldsymbol{p}_i, \boldsymbol{q}_i\right) \leq \epsilon\right) \tag{1.6}$$

where $\mathbb{I}(\cdot)$ is a indicator function, which returns 1 if the inner condition is true and return 0 otherwise.

Compared to other robust objective functions that utilize other robust loss functions, consensus maximization is easier to use. Specifically, it can distinguish inliers and outliers by the inlier threshold ϵ and maximizes the count of inlier measurements. Therefore, consensus maximization has been very popular in many computer vision problems due to its simplicity [29]. In this thesis, to obtain the robust solution from outlier-contaminated inputs, we also use consensus maximization to construct the robust objective functions.

Usually, consensus maximization is quite easy to be formulated in various applications, however, it is non-smooth, which means it is difficult to obtain its optimum by traditional gradient based optimization algorithms. To solve consensus maximization, the de facto standard is RANSAC [29]. RANSAC is a heuristic global optimization method and it is able to avoid being trapped in local optimum. However, RANSAC can only provide a reasonable solution only with *a certain probability*. Specifically, RANSAC repeatedly samples minimal/non-minimal [55, 56] subset and solves the candidate solutions from the sampled subset. In the meantime, every candidate solution can be applied to calculate the inlier number. RANSAC takes the best candidate solution, which obtains maximum inlier number, as the final solution. Evidently, the randomized nature of RANSAC does not provide an absolute certainty whether the obtained result is a satisfactory solution [57].

1.3 Globally Optimal Solutions

For the computer vision algorithms in safety critical systems, the provable safety guarantees are usually desired [9, 58]. From the perspective of optimization,

to ensure the safety guarantee is to obtain the global optimums with provable guarantee. Being trapped in local optimums should be prohibited since the local optimums may be an incorrect solution, which is unacceptable [59]. For example, it is highly needed in self-driving cars to obtain the global optimal motion estimation [52], since the local optimum may lead to a serious failure [9]. From view of optimization, if the problem can be formulated as a convex problem, then the only-one local optimum must be the global optimum [60]. Many classical local optimization methods are sufficient to obtain the optimal solution. However, in real applications, noise and outliers are unavoidable. To suppress the corrupted data, the robust non-convex objective functions should be formulated. Consequently, there are usually many local optimums in the non-convex problems. If the initial start point is not properly set, local optimization methods might be trapped in the local optimums, which may lead to serious failures in some safety-critical applications. Therefore, globally optimal algorithms are highly needed to find the global solution to meet safety requirements in some applications. The whole mind flow chart is shown as Fig. 1.5.

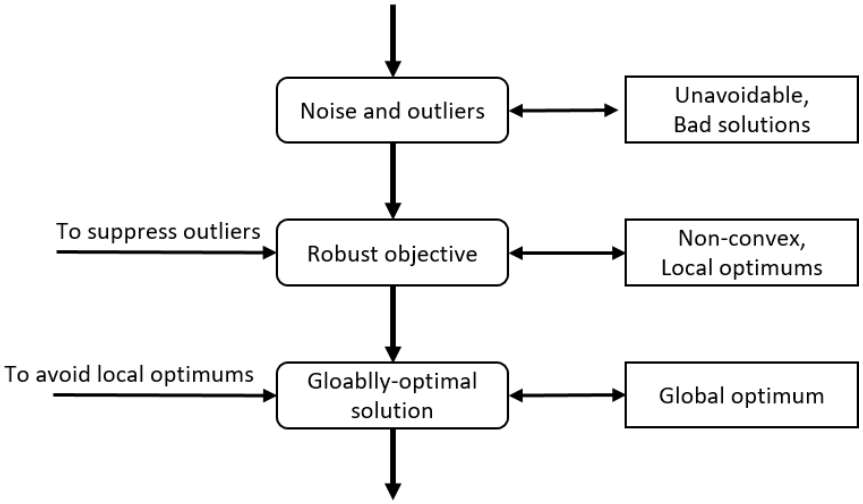


Figure 1.5: A flow chart for our motivation.

Notably, the globally optimal solutions in this thesis are usually meaning deterministic global optimization [61, 62], which is different from non-deterministic global optimization (e.g., multistart type methods [63] and genetic type algorithms [64]). Typically, non-deterministic global optimization algorithms (e.g., RANSAC) can avoid being trapped in local optimums. But they will *converge in probability* to the global optimum. Theoretically, only if the runtime is unlimited, which is unrealistic, the probability that finding the globally optimal optimum by non-deterministic global optimization algorithms can increase towards 100%. In contrast, deterministic global optimization algorithms have a theoretical guarantee of convergence to the globally optimal optimum.

In fact, due to the significance of global optimum in practical safety-critical applications, obtaining the globally optimal solution has become a hot research

topic in computer vision filed [29]. There have been many globally optimal solutions are proposed in many visual perception applications. For example, tracking objects [65], point set registration [66–68], simultaneous camera pose and feature correspondence [52, 69], camera calibration [70, 71] and event camera motion estimation [72, 73].

1.4 Unit-Norm Constrained Problems

Arguably, many computer vision algorithms are proposed to extract mathematical information from the input measurements [74], such as target point location in object tracking, rigid pose in target pose estimation, and weight parameters in deep neural networks [75–77]. Roughly speaking, the mathematical information that is expected to be estimated in various real applications should satisfy some physical constraints. For example, there will be a reasonable range for target location in object tracking. More strictly, there will be a famous epipolar constraint in estimating relative pose from two-view images [11]. More generally, if the mathematical information to be sought should satisfy some geometrical constraints, which arise from the physical properties of the scene, these computer vision problems are called geometric vision problems [58]. Typically, the geometric vision problems are focused on exploring the fundamental geometrical constraints, and they are a broad subclass of computer vision problems.

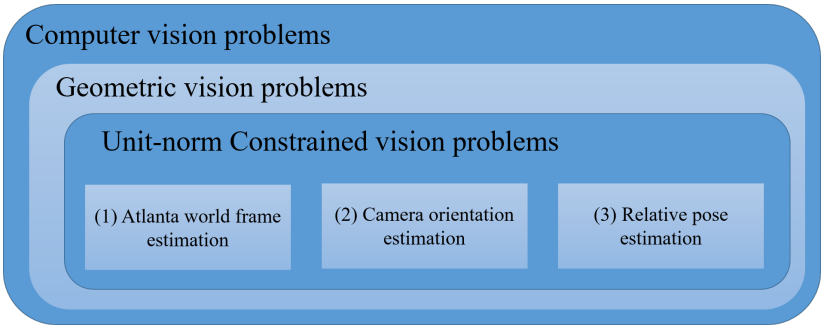
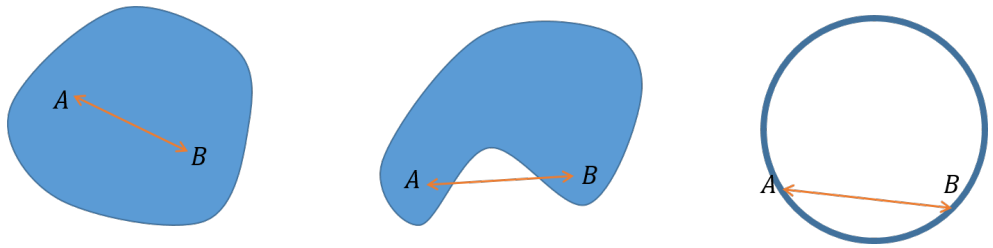


Figure 1.6: Unit-norm constrained computer vision problem is a special geometric vision problem. Three visual perception problems, which are investigated in this thesis, are all unit-norm constrained problems.

In this thesis, we mainly focus on unit-norm constrained computer vision problems, which are widely distributed in computer vision filed. Actually, we can also think that the unit-norm constraint is a special geometric constraint, since the unit-norm constraint can be considered as a unit sphere constraint. The relationship is illustrated as Fig. 1.6. Unfortunately, geometric vision problems are fulfilled with difficult optimization problems [58]. Notably, exactly solving some of geometric vision problems(i.e., obtaining the global optimum) are even inherently intractable [29]. For example, exactly solving consensus maximization for robust linear model fitting is NP-hard [78]. More pessimistically, it shows that solving consensus maximization for robust linear model fitting is impossible

to approximate, in other words, there are no polynomial time algorithm that computes an approximation solution efficiently [30, 78].

Although unit-norm constrained computer vision problems can be considered as special geometric problems, they have many special properties. In this thesis, we explore these special properties of the unit-norm constraint and propose globally optimal solutions for some typical unit-norm constrained computer vision problems. One thing should be mentioned is that the unit-norm constraint can be considered that the solution domain of the optimization problem is in the unit (hyper-)sphere, which is certainly non-convex (see Fig. 1.7). Therefore, an optimization problem with unit-norm constraint should be non-convex problem(no matter if the objective function is convex or not) [60]. Generally, solving a non-convex problem is NP-hard [79]. However, fortunately, solving a specific non-convex problem does not necessarily mean that it must be difficult [80].

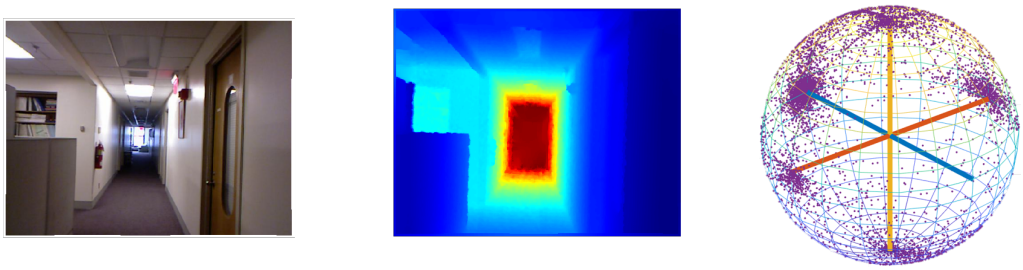


(a) convex region (b) non-convex region (c) non-convex region

Figure 1.7: Unit-norm constraint is non-convex. The region is convex if the segment between any two distinct points of the region is completely included in the region. (a) and (b) are typical convex and non-convex region, respectively. (c) Evidently, the unit-norm constraint(e.g., circle edge) is not convex.

Specifically, in this thesis, globally optimal solutions are explored for three applications:(1) vertical frame direction estimation in Atlanta world; (2) absolute camera orientation estimation from line correspondences; (3) relative pose estimation with known gravity direction. They are all unit-norm constrained computer vision problems, and they share many similar properties.

World frame estimation in Atlanta world [81]. In man-made environments, most of the objects and structures are usually organized in the form of orthogonal and parallel planes. Atlanta world makes an assumption that the man-made scene can be modeled by a horizontal plane (e.g., ground plane) and many vertical planes (e.g., buildings and walls). The normals of the planes, which are called world frames, can describe the scenes abstractly. In other words, one vertical frame and multiple horizontal frames could represent Atlanta world. It is a crucial step to estimate these vertical and horizontal frame directions in computer vision applications, which is named Atlanta frame estimation. It could be utilized as key modules for various high-level vision applications. Notably, there is an interesting property that all horizontal frames are in a plane and the vertical frame is parallel to the normal of the plane(see Fig. 1.8). In order to estimate the world frames in Atlanta world, the unique vertical direction can be



(a) Structural world **(b)** Depth image **(c)** Surface normal sphere

Figure 1.8: Atlanta frame estimation. According to Atlanta world assumption, one vertical frame and many horizontal frames can represent the world. We need to estimate the world frames from given inputs. For example, given the depth images, the surface normals can be exacted and they can be used to estimate the world frames.

estimated first. The vertical world frame direction is denoted as $\boldsymbol{v} = [v_1, v_2, v_3]^T$. Naturally, it should satisfy

$$\boldsymbol{v} \in \mathbb{S}^2 \tag{1.7}$$

where $\mathbb{S}^2$ means 2-sphere in three-dimensional space. Specifically, $\|\boldsymbol{v}\| = v_1^2 + v_2^2 + v_3^2 = 1$. Therefore, it is a typical unit-norm constrained problem. In fact, all Atlanta world frames are all in the 2-sphere. Therefore, Atlanta frame estimation is also a unit-norm constrained problem.

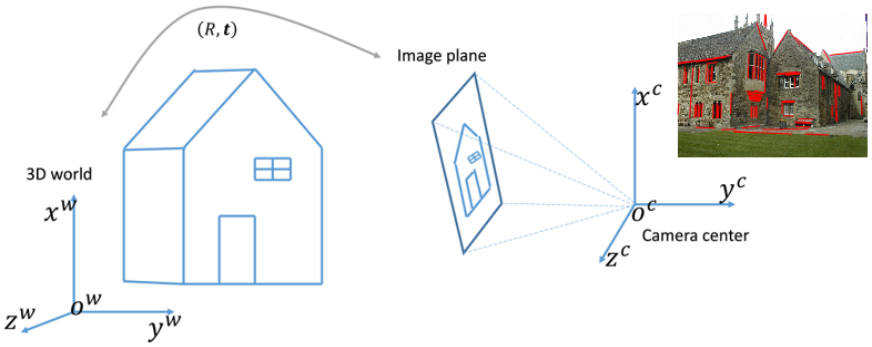


Figure 1.9: Absolute camera orientation estimation from line correspondences. It is a major part of the famous PnL problem [59], which is determining the relative position and orientation of a camera and an object from line correspondences.

Absolute camera orientation estimation from line correspondences [82]. This work is concerned with the problem of estimating camera orientation from a set of 2D/3D line correspondences, which is a major part of the Perspective-n-Line (PnL) problem(see Fig 1.9). Mathematically, camera rotation can be denoted by $\mathbf{R} \in \mathbb{SO}(3)$. Equivalently, we solve a rotation estimation problem [83].

At first glance, the constraint $\mathbf{R} \in \mathbb{SO}(3)$ seems nothing to do with unit-norm constraint. However, we notice that any rotation $\mathbf{R}$ can be represented not only

by a special orthogonal matrix but also by the unit quaternion numbers [84, 85]. Let $\boldsymbol{s} = [a, b, c, d]^T$ be a unit quaternion vector, then

$$\mathbf{R}_{\boldsymbol{s}} = \begin{bmatrix} a^2+b^2-c^2-d^2 & 2bc-2ad & 2bd+2ac \\ 2bc+2ad & a^2-b^2+c^2-d^2 & 2cd-2ab \\ 2bd-2ac & 2cd+2ab & a^2-b^2-c^2+d^2 \end{bmatrix}, \tag{1.8}$$

$$s.t. \quad a^2+b^2+c^2+d^2 = 1 \tag{1.9}$$

where $\mathbf{R}_{\boldsymbol{s}}$ is the rotation matrix represented by the unit quaternion $\boldsymbol{s}$. Consequently, camera orientation estimation can be formulated as a unit-norm constrained problem.

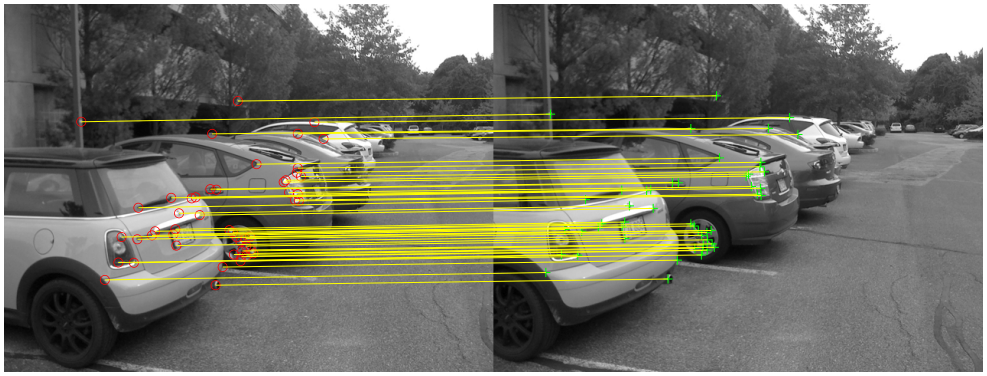


Figure 1.10: Relative pose estimation from keypoint correspondences.

Relative pose estimation with known gravity direction [86]. As we discussed, the task of relative pose estimation is estimating the relative camera pose from matching correspondences of two frames, also known as essential matrix estimation [11](see Fig. 1.10). It is well-known that we can obtain camera pose $\mathbf{R} \in \mathbb{SO}(3)$ and $\boldsymbol{t} \in \mathbb{S}^2$ (up to scale). Clearly it contains two unit-norm constraints and it is naturally a unit-norm constrained problem.

In this thesis, we focus on a quite common case in which the gravity direction is known in advance. If the gravity is known, then the epipolar geometry becomes

$$\boldsymbol{t}^T \left(\boldsymbol{a}_i + \sin(\theta)\boldsymbol{b}_i + \cos(\theta)\boldsymbol{c}_i\right) = 0 \tag{1.10}$$

where $\theta \in [-\pi, \pi]$ is to-be-solved rotation angle. $\boldsymbol{a}_i$,$\boldsymbol{b}_i$ and $\boldsymbol{c}_i$ are constructed by input correspondences and they are explained in detail in Section 5. Let $q_1 = \sin(\theta)$ and $q_2 = \cos(\theta)$, then Eq. (1.10) can be reformulated as

$$\boldsymbol{t}^T \left(\boldsymbol{a}_i + q_1\boldsymbol{b}_i + q_2\boldsymbol{c}_i\right) = 0, \quad s.t. \quad q_1^2 + q_2^2 = 1 \tag{1.11}$$

Therefore, relative pose estimation with known gravity direction is a unit-norm constrained problem.

Overall, unit-norm constrained problems are commonly distributed in computer vision filed. Mathematically, they can be considered as sub-parts of constrained

optimization problems, which are extensively studied [87,88]. Moreover, optimization methods on Riemannian manifolds are also well-studied [89,90]. However, in real applications, things are more complicated than the ideal mathematical models [29]. Specifically, the input measurements are usually imperfect, which may lead to incorrect solutions. Therefore, more robust objective functions instead of the sum of squared residual should be formulated to suppress the imperfect measurements. Generally, the robust objective functions are non-convex [43] and traditional algorithms may efficiently obtain a locally optimal solution, which may be an unacceptable solution for safety-critical applications. Unfortunately, it is non-trivial to obtain globally optimal solutions [91] for robust objective functions, especially, with unit-norm constraints.

1.5 Contributions

For unit-norm constrained computer vision problems, the unit-norm constraint is naturally non-convex. In addition, to suppress the noisy inputs, objectives are usually formulated as non-convex functions for practical computer vision problems. For such challenging tasks, in which both objective and constraint are non-convex, seeking the global optimum (i.e., extremely robust solution for safety-critical systems) is typically not easy. In this thesis, we take steps towards this goal. We focus on the globally optimal solutions for the unit-norm constrained computer vision problems.

The main contributions are as followings:

- Theoretically, a general inequality in n-sphere is introduced. Specifically, the unit-norm constraint is thoroughly explored, and a novel mapping way is applied to use compact parameters to represent the unit-norm constraint. Besides, by exploring the geometrical properties, a novel inequality is introduced. Solid proofs are provided in this thesis. Moreover, we find that the introduced inequality about n-sphere is a general formulation of one fundamental inequality in the famous rotation search theory [92]. It is worth noting that the inequality in rotation search theory is only applied in three-dimensional space i.e., $\mathbb{SO}(3)$. However, our introduced general inequality is extended to n-dimensional space.

- A novel globally optimal solution is proposed to estimate the vertical direction in Atlanta world [81]. In man-made environments, many objects and structures are organized in the form of orthogonal and parallel planes. These planes can be approximated by an Atlanta world assumption, in which the normals of planes can be represented by Atlanta frames. The Atlanta world assumption has one vertical frame and multiple horizontal frames. Given a set of inputs such as surface normals, the Atlanta frame estimation problem can be solved by a globally optimal algorithm [93]. However, the runtime will increase greatly when the dimensionality (i.e., the number of horizontal frames) increases. In contrast, we estimate only the vertical

direction, instead of all Atlanta frames at once. Accordingly, we propose a vertical direction estimation method by considering the relationship between the vertical frame and horizontal frames. More importantly, our approach is able to obtain the globally optimal solution for vertical direction without requiring prior knowledge of the number of Atlanta frames.

- A novel globally optimal solution is proposed to estimate the camera orientation in Perspective-n-Line (PnL) problem [82,94]. Estimating camera orientation from a set of 2D/3D correspondences, which is a major part of the PnL problem, is a basic computer vision problem. There are some cases that usually occur in real applications: the input point/line correspondences are corrupted by mismatches (a.k.a. outlier correspondences). RANSAC algorithm is the de facto standard for solving outlier-contaminated problems. However, RANSAC is a non-deterministic algorithm, which means that it produces a reasonable result only with a certain probability. To obtain a certifiably optimal solution from outlier-contaminated data is a matter of priority for some safety-critical applications, we propose a globally optimal camera orientation estimation algorithm. Specifically, it first decouple rotation and translation. After that, the optimal camera orientation can be obtained by rotation search theory, which is a deterministic global optimization method. Therefore, the estimated camera orientation is indeed the best optimum with provably guarantee.

- A novel globally optimal solution is proposed to estimate the relative pose from two matching images with a known vertical direction [86]. Relative pose estimation is a core task in computer vision, and it is the basis of many high-level applications (e.g., visual odometry). We focus on a quite common case in which the gravity direction is known in advance with the help of IMUs. Commonly, incorrect feature matches (a.k.a. outliers) are unavoidable, and they will impair the accuracy significantly. RANSAC is the de facto standard to suppress the outliers and obtain a robust solution. However, RANSAC is a non-deterministic algorithm, which means it produces a reasonable result only with a certain probability, and it cannot guarantee the global optimality to meet the safety demand in many life-critical applications. Therefore, we propose a globally optimal algorithm for relative pose estimation with known gravity direction. Specifically, the proposed method employs the branch-and-bound (BnB) algorithm to solve a consensus maximization problem, and thus it is able to obtain the global solution with a provable guarantee.

Overall, in this thesis, we first explore the fundamental geometric of the unit sphere in arbitrary dimension. Accordingly, a general inequality for n-sphere is introduced. After that, we combine the inequality and branch-and-bound algorithm to globally solve some typical unit-norm constrained computer vision problems to meet the demand in safety-critical applications.

1.6 Thesis Outline

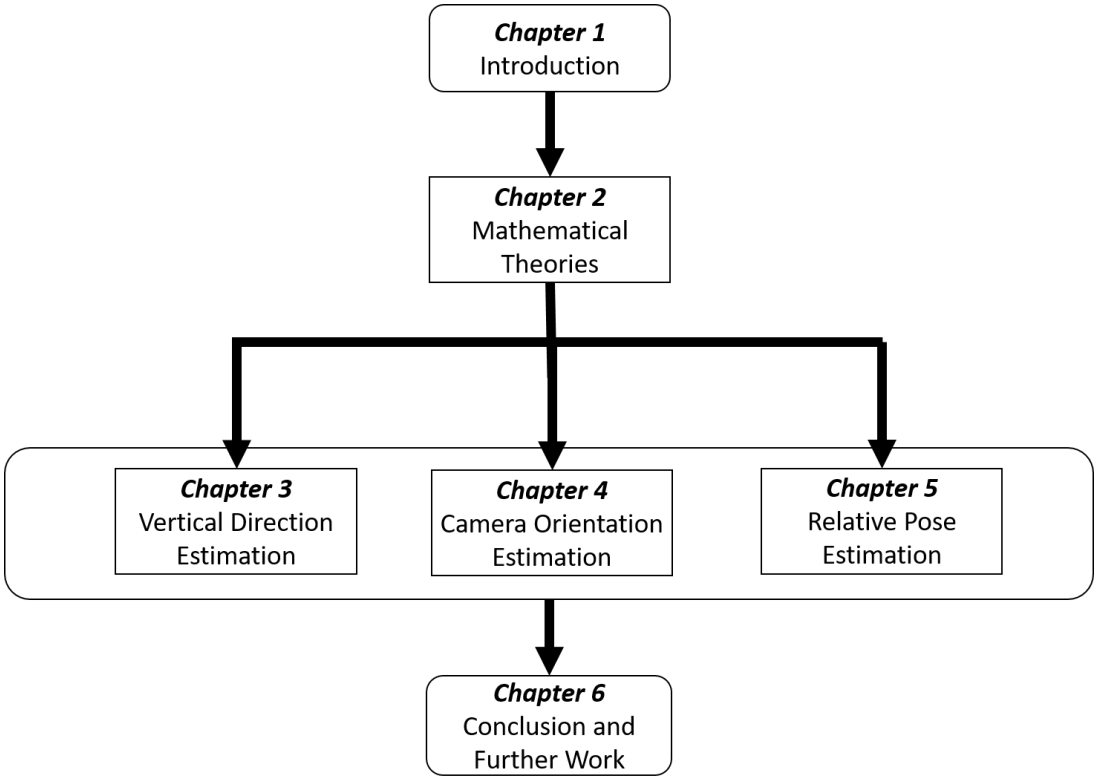


Figure 1.11: Thesis outline.

This thesis is structured as Fig. 1.11. Specifically, we first introduce the background of unit-norm constrained computer vision problems. Moreover, the motivation to obtain the globally optimal solutions are explained. Chapter 2 introduces the fundamental mathematical theories. It mainly includes the branch-and-bound algorithm, which is a deterministic optimization framework to obtain the globally optimal solutions, and a general inequality for n-sphere. Besides, the relationship between the general inequality and rotation search theory are explained in detail. After that, Chapter 3, 4, and 5 are the applications of the fundamental theories. Concretely, Chapter 3 presents a novel globally optimal solution for vertical direction estimation in Atlanta world. Chapter 4 introduces a novel globally optimal solution for camera orientation estimation from 2D and 3D line feature correspondences. Chapter 5 introduces a novel globally optimal solution for relative pose estimation with a known gravity direction. Chapter 6 provides conclusions and discussions on further work.

Chapter 2

Related Techniques

2.1 Branch and Bound

Branch-and-Bound (BnB) algorithm was first proposed in 1960s for discrete programming [95] and has become the most commonly used tool for solving many NP-hard optimization problems [96, 97]. Technically, the BnB algorithm will search the entire solution domain to seek the global optimum, therefore, it can avoid local optimums. However, different from naive brute-force search, the BnB algorithm consists of a systematically searching strategy, and therefore, it is usually more efficient than naive brute-force search [98].

Specifically, the BnB algorithm recursively splits the search space (i.e., solution domain) into smaller spaces, which is called ***branching***, and the smaller sub spaces are also called sub-branches. Given a sub-branch, it should be checked against upper and lower estimated bounds on the optimal solution, which is called ***bounding***. If it can be proved that it will not contain a better solution than the best one found so far by the algorithm in the given sub-branch, the sub-branch will be discarded, which is called ***pruning***. The BnB algorithm repeatedly conducts the process of ***branching***, ***bounding*** and ***pruning*** until convergence. As a consequence, some sub-branches can be efficiently discarded and the search space is gradually decreasing, which is significantly faster than brute-force enumeration of candidate solutions.

In this thesis, we introduce the BnB algorithm from the end-user standpoint. For more rigorous mathematical analysis about BnB algorithm, dear readers can refer to [99, 100]. From the perspective of usage, one thing is worth noting that the BnB algorithm highly depends on the efficient estimation of lower and upper bounds in the given sub-branch. The term of *efficiency* includes two points [83, 97]:

- Tightness. It should be expected that the lower and upper bounds of the optimal value are very tight. If we can tightly estimate the bounds, we can eliminate the branches that do not contain the optimal solution efficiently. The search space can be reduced rapidly and the BnB algorithm can terminate rapidly.

- Computing efficiently. Estimating the lower and upper bounds of the optimal value should be efficient. In each iteration, the bounds need be

estimated, therefore, computing the bounds efficiently contributes to the high efficiency of the BnB algorithm.

In addition, not any lower and upper bounds are sufficient to be nested into BnB algorithm. Two strict conditions for the convergence of BnB algorithms should be satisfied [100]:

1. Given a sub-branch, lower bound $\leq$ best optimal solution in the branch $\leq$ upper bound.

2. When the sub-branch collapses to a single point, the gap between the lower bound and upper bound should be zero.

Note that the second condition does not imply that when the BnB algorithm terminates, the gap between upper bound and lower bound must be zero. For smooth objective functions, when the gap between the upper and lower bounds is smaller than a certain threshold ϵ, the algorithm can stop and the found optimal solution is ϵ-suboptimal [100]. This can greatly reduce the computations burden and is usually used when the obtained solution is *good enough for practical purposes* [99]. Many famous computer vision algorithms, such as [66, 67, 101], rely on this strategy.

However, in this thesis, the robust objective functions are formulated by consensus maximization, which is non-smooth. Notably, the stop condition for the BnB algorithm is slightly stricter: when the gap between the upper and lower bounds becomes to zero, the algorithm terminates. In other words, the found solution by BnB algorithm is indeed the best optimal one with theoretical guarantee in this thesis.

To better explain how the BnB algorithm find the optimum, a simple example is given:

$$\text{Minimize} \quad f(x) = x^2 + 2x + 10\sin(2x), \quad x \in [-10, 10] \tag{2.1}$$

The true minimum is at $(-0.796, -10.956)$ (see Fig. 2.1).

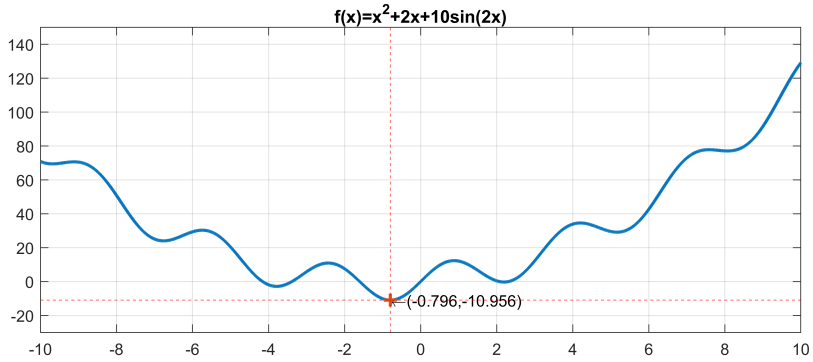


Figure 2.1: Graph of the example function.

From the figure of the objective function, we can find that there are many local optimums in the solution domain. If the initial point is not set properly, the classical gradient-based optimization methods may fall into a local optimum. In contrast, the BnB algorithm can obtain the global optimum. To estimate the lower and upper bound, we apply the interval analysis [102]. Specifically, given $x \in [a, b]$, $\min g_1 = x^2 + 2x \in [g_1^L, g_1^U]$ and $\min g_2 = 10\sin(2x) \in [g_2^L, g_2^U]$. Then $\min f = \min g_1 + \min g_2 \in [f^L, f^U]$, where $f^L = g_1^L + g_2^L$ and $f^U = g_1^U + g_2^U$. For example,

$$x \in [0, 1] \Rightarrow \begin{cases} \min g_1 \in [0, 1.250] \\ \min g_2 \in [0, 8.415] \end{cases} \Rightarrow \min f \in [0, 9.665] \tag{2.2}$$

$$x \in [-1, 0] \Rightarrow \begin{cases} \min g_1 \in [-1, -0.750] \\ \min g_2 \in [-10, -8.415] \end{cases} \Rightarrow \min f \in [-11, -9.165] \tag{2.3}$$

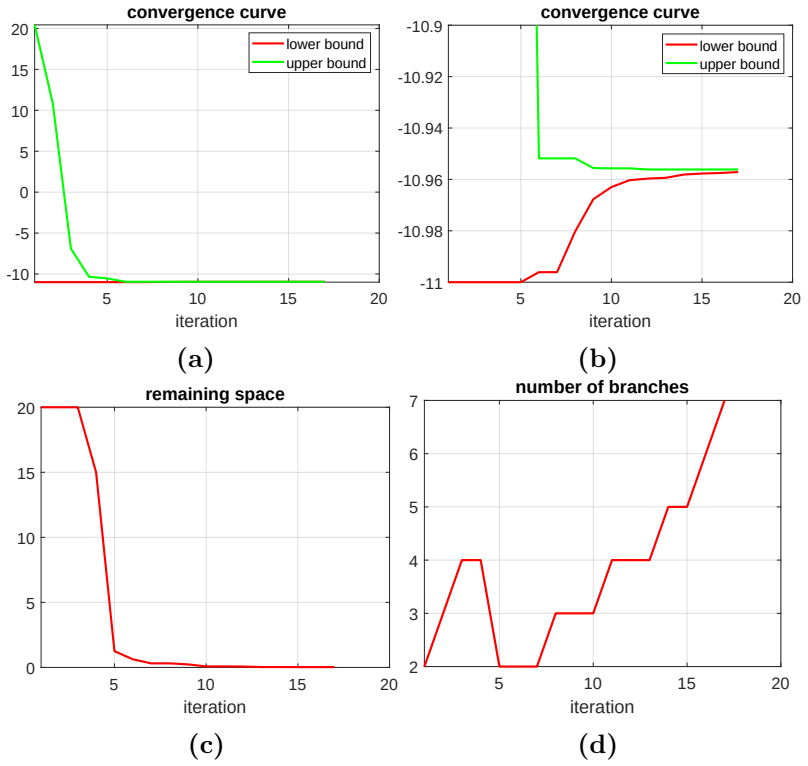


Figure 2.2: Convergence curve for the given example. (a) and (b) The gap between lower and upper bounds is becoming very small. (c) The remaining search space is gradually reduced to very small. (d) The number of branches represents the memory space occupied by the algorithm.

The termination condition for BnB algorithm is when the gap between lower and upper bounds is less than 0.001. The convergence curve is shown in Fig. 2.2. The concrete data in the convergence process is shown in Table 2.1. From the results, it is clearly shown that the search space is gradually decreasing and

the gap between the lower and upper bounds is gradually becoming small. The local optimums can be avoided and the algorithm can provide a globally optimal solution.

Table 2.1: Data in convergence process of BnB algorithm

iteration	x	lower	upper	branch num.	remaining
1	-5.0000	-11.0000	20.4402	2	20.0000
2	-2.5000	-11.0000	10.8392	3	20.0000
3	-1.2500	-11.0000	-6.9222	4	20.0000
4	-0.6250	-11.0000	-10.3492	4	15.0000
5	-0.9375	-11.0000	-10.5370	2	1.2500
6	-0.7813	-10.9961	-10.9518	2	0.6250
7	-0.7813	-10.9961	-10.9518	2	0.3125
8	-0.7813	-10.9802	-10.9518	3	0.3125
9	-0.8008	-10.9677	-10.9556	3	0.2344
10	-0.7910	-10.9630	-10.9557	3	0.0781
11	-0.7910	-10.9603	-10.9557	4	0.0781
12	-0.7959	-10.9597	-10.9561	4	0.0684
13	-0.7959	-10.9594	-10.9561	4	0.0293
14	-0.7959	-10.9581	-10.9561	5	0.0293
15	-0.7959	-10.9577	-10.9561	5	0.0244
16	-0.7959	-10.9575	-10.9561	6	0.0244
17	-0.7959	-10.9571	-10.9561	7	0.0244

2.2 Rotation Search Theory

In computer vision filed, rotation estimation is one of the most classical problems [83]. In this thesis, we consider rotation estimation problem as a special unit norm constrained problem. Typically, to estimate the rotation robustly, a robust non-convex objective function will be formulated. Consequently, classical gradient-based optimization algorithms may fall into the local optimum. Fortunately, the globally-optimal algorithms called rotation search theory have been developed to obtain the global optimum, and they have been applied in many applications [83, 103, 104].

Specifically, the rotation search theory applies the BnB algorithm to search the entire rotation space(i.e., $\mathbb{SO}(3)$) for the optimal solution. For different robust objective functions, different lower and upper bounds should be carefully designed. Nonetheless, to estimate the bounds, which involves the non-convex rotation constraint, is not an easy problem.

To estimate the lower and upper bounds, an method based on interval analysis is proposed in [103]. In fact, interval analysis defines a set of operations on intervals. For example,

- $[a,b]+[c,d] \Rightarrow [a+c, b+d]$
- $[a,b]-[c,d] \Rightarrow [a-d, b-c]$
- $[a,b]\times[c,d] \Rightarrow [\min(ac,ad,bc,bd), \max(ac,ad,bc,bd)]$
- $[a,b]\div[c,d] \Rightarrow \left[\min\left(\frac{a}{c},\frac{a}{d},\frac{b}{c},\frac{b}{d}\right), \max\left(\frac{a}{c},\frac{a}{d},\frac{b}{c},\frac{b}{d}\right)\right]$

This method is only applied to estimate the vanishing point in Manhattan world, which can be considered as a rotation estimation problem [103].

Notably, the most widely applied rotation search methods are based on two novel inequalities to estimate the lower and upper bounds, which are first introduced in [49,92] and they are discussed in the following parts. If not specifically pointed out, rotation search algorithms in the thesis only mean the methods that are based on the two novel inequalities by default. The rotation search theory has been applied in many applications to obtain the globally optimal solutions, such as 3D point set registration [66], hand-eye calibration [71],vanishing point estimation [103] and 2D-3D point/line registration [69].

2.2.1 Two Fundamental Inequalities

In this part, we introduce the two novel inequalities, which are the fundamental components in the most widely applied rotation search theory. First of all, we introduce the axis–angle representation of a rotation matrix. Mathematically, a rotation can be represented by a rotation angle θ and a rotation axis $\overline{\boldsymbol{r}}$, where $\overline{\boldsymbol{r}} \in \mathbb{R}^3$ and $\|\overline{\boldsymbol{r}}\| = 1$. The axis-angle representation can be defined as

$$\boldsymbol{r} = \theta\overline{\boldsymbol{r}} \tag{2.4}$$

Let $\mathbf{I}$ be identity matrix and $[\overline{\boldsymbol{r}}]_\times$ be cross product matrix of $\overline{\boldsymbol{r}}$.

$$[\overline{\boldsymbol{r}}]_\times = \begin{bmatrix} 0 & -\overline{r}_3 & \overline{r}_2 \\ \overline{r}_3 & 0 & -\overline{r}_1 \\ -\overline{r}_2 & \overline{r}_1 & 0 \end{bmatrix}, \quad [\overline{\boldsymbol{r}}]_\times^T = -[\overline{\boldsymbol{r}}]_\times, \quad [\overline{\boldsymbol{r}}]_\times^3 = -[\overline{\boldsymbol{r}}]_\times \tag{2.5}$$

The relationship of $\boldsymbol{r} \leftrightharpoons \mathbf{R}$ can be described by *Rodrigues' rotation formula* [11]:

$$\mathbf{R} = e^{\theta[\overline{\boldsymbol{r}}]_\times} = \mathbf{I} + \theta[\overline{\boldsymbol{r}}]_\times + \frac{1}{2!}(\theta[\overline{\boldsymbol{r}}]_\times)^2 + \frac{1}{3!}(\theta[\overline{\boldsymbol{r}}]_\times)^3 + \cdots \tag{2.6}$$

$$= \mathbf{I} + \left(\theta - \frac{1}{3!}\theta^3 + \cdots\right)[\overline{\boldsymbol{r}}]_\times + \left(\frac{1}{2!}\theta^2 - \frac{1}{4!}\theta^4 + \cdots\right)[\overline{\boldsymbol{r}}]_\times^2 \tag{2.7}$$

$$= \mathbf{I} + \sin(\theta)[\overline{\boldsymbol{r}}]_\times + (1-\cos(\theta))[\overline{\boldsymbol{r}}]_\times^2 \tag{2.8}$$

In addition [105],

$$\theta = \arccos\left(\frac{Tr(\mathbf{R})-1}{2}\right) \tag{2.9}$$

$$2\sin(\theta)[\overline{\boldsymbol{r}}]_\times = \mathbf{R} - \mathbf{R}^T \tag{2.10}$$

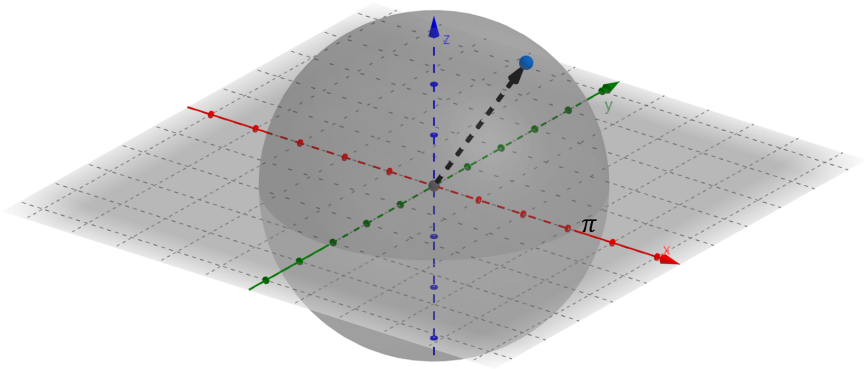


Figure 2.3: π-ball

where $Tr(\mathbf{R})$ is the trace of the rotation matrix.

One thing should be noted that without restricting θ, the axis-angle representation of a rotation matrix is not unique. For example, $\theta\overline{\boldsymbol{r}}$ and $(2\pi-\theta)(-\overline{\boldsymbol{r}})$ correspond to the same rotation. To reduce the redundancy, θ can be restricted in $[0,\pi]$. Geometrically, we have the famous π-ball [83], which is illustrated in Fig. 2.3. The $\mathbb{SO}(3)$ space can be represented by the π-ball. In other words, given a rotation matrix $\mathbf{R}$, there is at least a point $\boldsymbol{r}$, which corresponds to $\mathbf{R}$, in the solid ball.

After introducing the axis–angle representation of rotations, we then list the famous lemmas [92]:

Lemma 2.1. *For any vector* $\boldsymbol{v}\in\mathbb{R}^3$, $\mathbf{R}_1,\mathbf{R}_2\in\mathbb{SO}(3)$, *then [92]*

$$\angle(\mathbf{R}_1\boldsymbol{v},\mathbf{R}_2\boldsymbol{v})\leq\angle(\mathbf{R}_1,\mathbf{R}_2) \tag{2.11}$$

where $0\leq\angle(\mathbf{R}_1,\mathbf{R}_2)\leq\pi$ is the angle distance between $\mathbf{R}_1$ and $\mathbf{R}_2$. More specifically,

$$\angle(\mathbf{R}_1,\mathbf{R}_2)=\arccos\left(\frac{Tr(\mathbf{R}_1\mathbf{R}_2^T)-1}{2}\right)=\arccos\left(\frac{Tr(\mathbf{R}_1^T\mathbf{R}_2)-1}{2}\right) \tag{2.12}$$

The authors in [92] think that lemma 2.1 seems simple enough that they omit the proof. In this thesis, we show a picture in Fig. 2.4 to explain the results visually. Specifically, to prove lemma 2.1, it is sufficient to prove

$$\angle\left(\boldsymbol{v},\mathbf{R}_1^T\mathbf{R}_2\boldsymbol{v}\right)=\angle(\mathbf{R}_1\boldsymbol{v},\mathbf{R}_2\boldsymbol{v})\leq\angle(\mathbf{R}_1,\mathbf{R}_2) \tag{2.13}$$

Let $\mathbf{R}=\mathbf{R}_1^T\mathbf{R}_2$, then we need to prove

$$\angle(\boldsymbol{v},\mathbf{R}\boldsymbol{v})\leq\arccos\left(\frac{Tr(\mathbf{R}_1^T\mathbf{R}_2)-1}{2}\right)=\arccos\left(\frac{Tr(\mathbf{R})-1}{2}\right)=\theta \tag{2.14}$$

where θ is the rotation angle of $\mathbf{R}$.

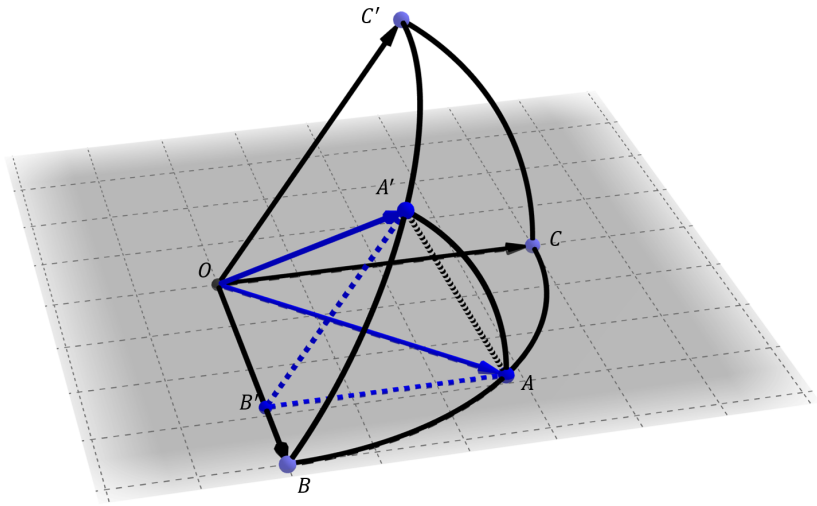


Figure 2.4: Illustration of lemma 2.1.

In Fig. 2.4, we define $\boldsymbol{v} = \overrightarrow{OA}$ and the unit direction of $\overrightarrow{OB}$ is the rotation axis $\overline{\boldsymbol{r}}$ of $\mathbf{R}$. $OC' \perp OB$, $OC \perp OB$, $|OA| = |OB| = |OC| = |OC'|$, and point O, A, B, C are in the same plane. We define $\angle C'OC = \theta$, therefore, $\mathbf{R}\overrightarrow{OC} = \overrightarrow{OC'}$, $\mathbf{R}\overrightarrow{OA} = \mathbf{R}\boldsymbol{v} = \overrightarrow{OA'}$ and points O, A', B, C' are in the same plane. In addition, $AB' \perp OB$ and $A'B' \perp OB$, then $\angle A'B'A = \angle C'OC = \theta$.

In $\triangle A'B'A$ and $\triangle A'OA$, we have

$$|AA'| = 2\sin\left(\frac{\theta}{2}\right)|AB'| = 2\sin\left(\frac{\angle A'OA}{2}\right)|OA| \tag{2.15}$$

In addition, $|OA||\sin(\angle AOB)| = |AB'|$, then

$$\sin\left(\frac{\theta}{2}\right)|\sin(\angle(\boldsymbol{v}, \overline{\boldsymbol{r}}))| = \sin\left(\frac{\angle(\boldsymbol{v}, \mathbf{R}\boldsymbol{v})}{2}\right) \tag{2.16}$$

$$\Rightarrow \angle(\boldsymbol{v}, \mathbf{R}\boldsymbol{v}) = 2\arcsin\left(\sin\left(\frac{\theta}{2}\right)|\sin(\angle(\boldsymbol{v}, \overline{\boldsymbol{r}}))|\right) \tag{2.17}$$

$$\leq 2\arcsin\left(\sin\left(\frac{\theta}{2}\right)\right) \tag{2.18}$$

$$= \theta \tag{2.19}$$

Only when the axis of rotation is perpendicular to the vector, i.e., $\angle(\boldsymbol{v}, \boldsymbol{r}) = \frac{\pi}{2}$, the equal sign can be established.

Lemma 2.2. *For any $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{SO}(3)$, $\boldsymbol{r}_1, \boldsymbol{r}_2$ in the π-ball are the corresponding axis–angle representations, then [92, 106]*

$$\angle(\mathbf{R}_1, \mathbf{R}_2) \leq \|\boldsymbol{r}_1 - \boldsymbol{r}_2\| \tag{2.20}$$

It is worth noting that this lemma describes the angle distance of two rotations is less than the Euclidean distance in their angle-axis representations.

To prove lemma 2.2, the authors in [92] are simply explaining *this relationship may be proved rigorously by computing an infinitessimal metric on the quaternion sphere* [92]. In addition, a more complete and strict proof is provided in [49], which is based on the basic theory in weakly compact lie groups [107]. However, this proof is not easy to be understood for some junior researchers and they have to understand many obscure mathematical concepts. Fortunately, a proof that is easier to understand is provided by Thomas Ruland in [108].

In this thesis, we introduce a simply explanation to understand this lemma more intuitively. First, we discuss unit quaternion representations of rotation more deeply [105,106]. Specifically, given rotation $\mathbf{R}$ and their axis-angle representations $\boldsymbol{r} = \theta\overline{\boldsymbol{r}}$, the unit quaternion representations should be

$$\boldsymbol{s} = \left[\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right)\overline{\boldsymbol{r}}\right] \triangleq [a, \boldsymbol{b}] \tag{2.21}$$

Notably, if we do not restrict the bound of θ, then

$$\mathbf{R} \leftrightharpoons \pm\boldsymbol{s} \tag{2.22}$$

Accordingly, we can define $d_{\angle}(\cdot,\cdot)$ to describe the rotation distance [105].

$$d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2) = \arccos\left(|\boldsymbol{s}_1^T \boldsymbol{s}_2|\right) \tag{2.23}$$

The *abs* function is to restrict

$$0 \leq d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2) \leq \pi/2 \tag{2.24}$$

Furthermore, the conjugate of the quaternion

$$\boldsymbol{s}^* = \left[\cos\left(\frac{\theta}{2}\right), -\sin\left(\frac{\theta}{2}\right)\overline{\boldsymbol{r}}\right] = [a, -\boldsymbol{b}] \tag{2.25}$$

In fact, conjugate of the quaternion is also a quaternion and $\boldsymbol{s}^*$ represents $\mathbf{R}^T$.

The multiplication of two quaternions [84]

$$\begin{aligned} \boldsymbol{s}_{\mathbf{R}_1} \circledast \boldsymbol{s}_{\mathbf{R}_2} &= [a_1, \boldsymbol{b}_1] \circledast [a_2, \boldsymbol{b}_2] && (2.26)\\ &= \left[a_1 a_2 - \boldsymbol{b}_1^T \boldsymbol{b}_2, a_1\boldsymbol{b}_2 + a_2\boldsymbol{b}_1 + \boldsymbol{b}_1 \times \boldsymbol{b}_2\right] && (2.27)\\ &= \boldsymbol{s}_{\mathbf{R}_1\mathbf{R}_2} && (2.28) \end{aligned}$$

Therefore, the multiplication of two quaternions is still a quaternion and in fact $\boldsymbol{s}_{\mathbf{R}_1\mathbf{R}_2}$ represents $\mathbf{R}_1\mathbf{R}_2$.

Lemma 2.3. *Given rotation $\mathbf{R}_1$, $\mathbf{R}_2$ and their unit quaternion representations $\boldsymbol{s}_1$, $\boldsymbol{s}_2$, then [92, 105, 106]*

$$\angle(\mathbf{R}_1, \mathbf{R}_2) = 2d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2) \tag{2.29}$$

Proof. To prove lemma 2.3, let $\mathbf{R} = \mathbf{R}_1^T \mathbf{R}_2$, then

$$\angle(\mathbf{R}_1, \mathbf{R}_2) = \angle\left(\mathbf{I}, \mathbf{R}_1^T \mathbf{R}_2\right) = \angle(\mathbf{I}, \mathbf{R}) = \theta \tag{2.30}$$

Observe the composition of the rotation $\mathbf{R}_1^T \mathbf{R}_2$ can be obtained by the product of the quaternions

$$\begin{aligned} \boldsymbol{s}_{\mathbf{R}_1^T \mathbf{R}_2} &= \boldsymbol{s}_{\mathbf{R}_1^T} \circledast \boldsymbol{s}_{\mathbf{R}_2} = \boldsymbol{s}_{\mathbf{R}_1}^* \circledast \boldsymbol{s}_{\mathbf{R}_2} && (2.31)\\ &= [a_1, -\boldsymbol{b}_1] \circledast [a_2, \boldsymbol{b}_2] && (2.32)\\ &= \left[a_1 a_2 + \boldsymbol{b}_1^T \boldsymbol{b}_2, a_1 \boldsymbol{b}_2 - a_2 \boldsymbol{b}_1 - \boldsymbol{b}_1 \times \boldsymbol{b}_2\right] && (2.33) \end{aligned}$$

and

$$\boldsymbol{s}_{\mathbf{R}} = \boldsymbol{s}_{\mathbf{R}_1^T \mathbf{R}_2} = \left[\cos\left(\frac{\theta}{2}\right), \sin\left(\frac{\theta}{2}\right) \overline{\boldsymbol{r}}\right] \tag{2.34}$$

Therefore,

$$a_1 a_2 + \boldsymbol{b}_1^T \boldsymbol{b}_2 = \cos\left(\frac{\theta}{2}\right) \tag{2.35}$$

Evidently,

$$\boldsymbol{s}_1^T \boldsymbol{s}_2 = a_1 a_2 + \boldsymbol{b}_1^T \boldsymbol{b}_2 = \cos\left(\frac{\theta}{2}\right) \tag{2.36}$$

Considering $0 \leq \theta \leq \pi$, then $\arccos\left(|\boldsymbol{s}_1^T \boldsymbol{s}_2|\right) = \frac{\theta}{2}$, therefore,

$$d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2) = \arccos\left(|\boldsymbol{s}_1^T \boldsymbol{s}_2|\right) = \frac{\theta}{2} \tag{2.37}$$

Consequently, $2d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2) = \angle(\mathbf{R}_1, \mathbf{R}_2)$. □

Lemma 2.4. *Given rotation $\mathbf{R}_1$, $\mathbf{R}_2$, their unit quaternion representations $\boldsymbol{s}_1$, $\boldsymbol{s}_2$, and corresponding axis–angle representations $\boldsymbol{r}_1, \boldsymbol{r}_2$, then [92, 108]*

$$2d_{\angle}(\boldsymbol{s}_1, \boldsymbol{s}_2) \leq \|\boldsymbol{r}_1 - \boldsymbol{r}_2\| \tag{2.38}$$

It is not easy to prove lemma 2.4. Nonetheless, a straight and detailed proof is given in [108]. Geometrically, lemma 2.4 shows a relationship between the 3-dimensional solid π-ball and unit quaternion sphere. In the following part, we provide a more general proof for n-sphere and its corresponding n-dimensional solid ball.

Overall, in this thesis, we separate lemma 2.2 into lemma 2.3 and lemma 2.4. This is also the way to explain lemma 2.2 in [92]. Specifically, the angle distance between two rotations is related to the angle distance between unit quaternions in lemma 2.3. After that the relationship between unit quaternions and solid π-ball are discussed and formulated in lemma 2.4.

Lemma 2.1 and 2.2 are two of the most important inequalities in rotation search theory. From lemma 2.1 and 2.2, it is straightforward to get the following result:

Lemma 2.5. *For any vector* $\boldsymbol{x} \in \mathbb{R}^3$, $\mathbf{R}_1, \mathbf{R}_2 \in \mathbb{SO}(3)$ *and* $\boldsymbol{r}_1, \boldsymbol{r}_2$ *in the* π*-ball are the corresponding axis–angle representations, then [92]*

$$\angle\left(\mathbf{R}_1\boldsymbol{x}, \mathbf{R}_2\boldsymbol{x}\right) \leq \|\boldsymbol{r}_1 - \boldsymbol{r}_2\| \tag{2.39}$$

This lemma is more widely applied in solving many complicated computer vision problems. Some following works just rely on this lemma without considering lemma 2.1 and lemma 2.2, for example [67, 93, 109].

2.3 An Inequality in n-sphere

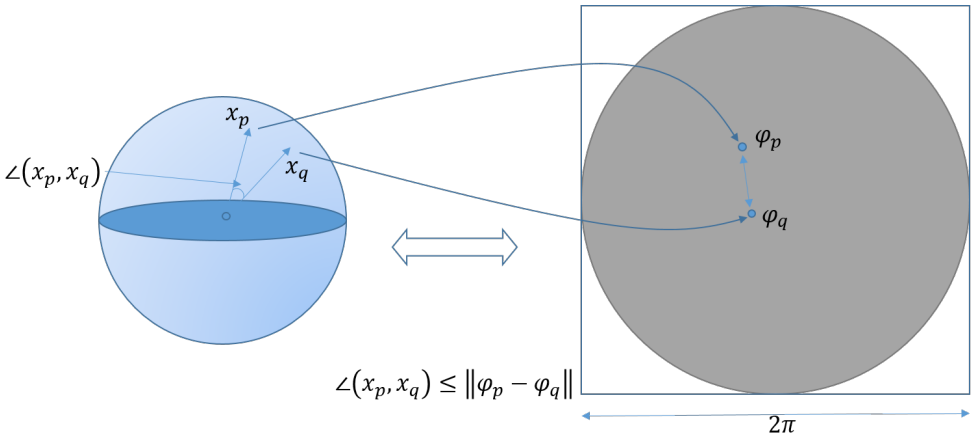


Figure 2.5: Illustration of exponential map

In this thesis, we focus on the general unit norm constrained problems and therefore we introduce a more general inequality in n-sphere ($\mathbb{S}^n$), which is a n-dimensional unit sphere in $\mathbb{R}^{n+1}$.

First, we introduce a novel parameterization for n-sphere. Generally, a point $\boldsymbol{x}$ in n-sphere can be

$$\boldsymbol{x} = [x_1, x_2, \cdots, x_n, x_{n+1}]^T \text{ and } \|\boldsymbol{x}\| = 1 \tag{2.40}$$

However, it is well known that the n-sphere is in fact n dimension, therefore, we need a compact parameterization to eliminate the unit norm constraint. Inspired from the relationship between unit quaternion sphere and solid π-ball, we apply the exponential map (Lie theory [110, 111]). Specifically, let $\boldsymbol{x} \in \mathbb{S}^d$, then

$$\boldsymbol{x}^T = \left[\cos(\theta), \sin(\theta)\boldsymbol{\omega}^T\right] \tag{2.41}$$

where $0 \leq \theta \leq \pi$; $\boldsymbol{\omega} = [\omega_1, \omega_2, \cdots, \omega_d]^T \in \mathbb{R}^d$ and $\|\boldsymbol{\omega}\| = 1$. From Eq. (2.41), we can find that $\boldsymbol{x}$ can be represented by θ and $\boldsymbol{\omega}$. Then, it can be defined as

$$\boldsymbol{\varphi} = \theta\boldsymbol{\omega} \tag{2.42}$$

Obviously, $\boldsymbol{\varphi} \in \mathbb{R}^d$. In addition, the domain of $\boldsymbol{\varphi}$ is similar to π-ball. Geometrically, the unit sphere is mapped into a solid ball. The relationship between the unit sphere and solid ball $\boldsymbol{\varphi} \leftrightharpoons \boldsymbol{x}$ is illustrated in Fig. 2.5.

Lemma 2.6. *$\boldsymbol{x}_p \in \mathbb{S}^d$, $\boldsymbol{x}_q \in \mathbb{S}^d$ are two vectors in the d-dimensional unit hyper-hemisphere, and $\boldsymbol{\varphi}_p \in \mathbb{R}^d$, $\boldsymbol{\varphi}_q \in \mathbb{R}^d$ are the two corresponding points in the d-dimensional solid ball.*

$$\angle(\boldsymbol{x}_p, \boldsymbol{x}_q) \leq \|\boldsymbol{\varphi}_p - \boldsymbol{\varphi}_q\| \tag{2.43}$$

Proof. $\boldsymbol{x}_p, \boldsymbol{x}_q, \boldsymbol{\varphi}_p, \boldsymbol{\varphi}_q$ can be formulated as:

$$\boldsymbol{\varphi}_p = \boldsymbol{\omega}_p \cdot \theta_p = [\omega_{p1}\theta_p, \omega_{p2}\theta_p, \cdots, \omega_{pd}\theta_p]^T \tag{2.44}$$
$$\boldsymbol{\varphi}_q = \boldsymbol{\omega}_q \cdot \theta_q = [\omega_{q1}\theta_q, \omega_{q2}\theta_q, \cdots, \omega_{qd}\theta_q]^T \tag{2.45}$$
$$\boldsymbol{x}_p = [\cos(\theta_p), \omega_{p1}\sin(\theta_p), \omega_{p2}\sin(\theta_p), \cdots, \omega_{pd}\sin(\theta_p)]^T \tag{2.46}$$
$$\boldsymbol{x}_q = [\cos(\theta_q), \omega_{q1}\sin(\theta_q), \omega_{q2}\sin(\theta_q), \cdots, \omega_{qd}\sin(\theta_q)]^T \tag{2.47}$$

Since $\|\boldsymbol{x}_p\| = \|\boldsymbol{x}_q\| = \|\boldsymbol{\omega}_p\| = \|\boldsymbol{\omega}_q\| = 1$, we can define

$$\gamma = \angle(\boldsymbol{\omega}_p, \boldsymbol{\omega}_q) = \arccos(\frac{\boldsymbol{\omega}_p^T \boldsymbol{\omega}_q}{\|\boldsymbol{\omega}_p\|\|\boldsymbol{\omega}_q\|}) = \arccos(\boldsymbol{\omega}_p^T \boldsymbol{\omega}_q) \tag{2.48}$$

On the left, we have

$$\rho = \angle(\boldsymbol{x}_p, \boldsymbol{x}_q) \tag{2.49}$$
$$= \arccos\big(\frac{\boldsymbol{x}_p^T \boldsymbol{x}_q}{\|\boldsymbol{x}_p\|\|\boldsymbol{x}_q\|}\big) = \arccos(\boldsymbol{x}_p^T \boldsymbol{x}_q) \tag{2.50}$$
$$= \arccos\big(\cos(\theta_p)\cos(\theta_q) + \omega_{p1}\omega_{q1}\sin(\theta_p)\sin(\theta_q) + \cdots + \omega_{pd}\omega_{qd}\sin(\theta_p)\sin(\theta_q)\big) \tag{2.51}$$
$$= \arccos\big(\cos(\theta_p)\cos(\theta_q) + (\boldsymbol{\omega}_p^T \boldsymbol{\omega}_q)\sin(\theta_p)\sin(\theta_q)\big) \tag{2.52}$$
$$= \arccos\big(\cos(\theta_p)\cos(\theta_q) + \cos(\gamma)\sin(\theta_p)\sin(\theta_q)\big) \tag{2.53}$$

Let $\lambda \in [0, 1]$, $\cos(\gamma) = 2\lambda - 1$, then we have

$$\rho^2 = \arccos^2\big(\cos(\theta_p)\cos(\theta_q) + (2\lambda - 1)\sin(\theta_p)\sin(\theta_q)\big) \tag{2.54}$$
$$= \arccos^2\big(\lambda\cos(\theta_p - \theta_q) + (1 - \lambda)\cos(\theta_p + \theta_q)\big) \tag{2.55}$$

On the right,

$$\xi = \|\boldsymbol{\varphi}_p - \boldsymbol{\varphi}_q\| = \|\boldsymbol{\omega}_p\theta_p - \boldsymbol{\omega}_q\theta_q\| \tag{2.56}$$
$$= \sqrt{(\omega_{p1}\theta_p - \omega_{q1}\theta_q)^2 + \cdots + (\omega_{pd}\theta_p - \omega_{qd}\theta_q)^2} \tag{2.57}$$
$$= \sqrt{\theta_p^2 + \theta_q^2 - 2\theta_p\theta_q\cos(\gamma)} \tag{2.58}$$

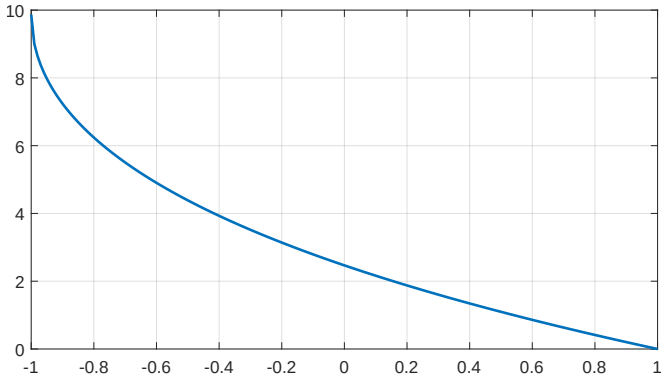


Figure 2.6: $f(\alpha) = \arccos^2(\alpha)$

Since $\cos(\gamma) = 2\lambda - 1$,

$$\begin{aligned} \xi^2 &= \theta_p^2 + \theta_q^2 - 2\theta_p\theta_q\cos(\gamma) && (2.59)\\ &= \theta_p^2 + \theta_q^2 - 2\theta_p\theta_q(2\lambda - 1) && (2.60)\\ &= \lambda(\theta_p - \theta_q)^2 + (1-\lambda)(\theta_p + \theta_q)^2 && (2.61) \end{aligned}$$

To compare ρ and ξ, we introduce a convex function $f(\alpha) = \arccos^2(\alpha)$, when $\alpha \in [-1, 1]$, see Fig. 2.6. Its second derivative is as follows

$$f''(\alpha) = \frac{2}{1-\alpha^2} - 2 \cdot \frac{\alpha \cdot \arccos(\alpha)}{(1-\alpha^2)^{3/2}} \tag{2.62}$$

It is easy to confirm $f''(\alpha)$ is non-negative, when $\alpha \in (-1, 1)$. According to Jensen's inequality, we then have

$$f\big(\lambda\alpha_1 + (1-\lambda)\alpha_2\big) \leq \lambda f(\alpha_1) + (1-\lambda)f(\alpha_2) \tag{2.63}$$

Let $\alpha_1 = \cos(\theta_p - \theta_q)$ and $\alpha_2 = \cos(\theta_p + \theta_q)$, then we have

$$\begin{aligned} \rho^2 &= \arccos^2(\lambda\cos(\theta_p - \theta_q) + (1-\lambda)\cos(\theta_p + \theta_q)) && (2.64)\\ &\leq \lambda\arccos^2(\cos(\theta_p - \theta_q)) + (1-\lambda)\arccos^2(\cos(\theta_p + \theta_q)) && (2.65) \end{aligned}$$

According to the bound of $\theta_p - \theta_q$ and $\theta_p + \theta_q$, two cases are discussed separately.

Case (**a**): If $\boldsymbol{x}_p, \boldsymbol{x}_q$ are on the *upper* hemisphere $\mathbb{S}^{d+}$, which means $x_{p1} \geq 0$ and $x_{q1} \geq 0$. Then $0 \leq \theta_p \leq \frac{\pi}{2}$, $0 \leq \theta_q \leq \frac{\pi}{2}$. Therefore, $-\pi \leq \theta_p - \theta_q \leq \pi$ and $-\pi \leq \theta_p + \theta_q \leq \pi$. Moreover,

$$\arccos^2(\cos(\theta)) = \theta^2, \quad \theta \in [-\pi, \pi] \tag{2.66}$$

Therefore,

$$\begin{aligned} \rho^2 &\leq \lambda\arccos^2(\cos(\theta_p - \theta_q)) + (1-\lambda)\arccos^2(\cos(\theta_p + \theta_q)) && (2.67)\\ &= \lambda(\theta_p - \theta_q)^2 + (1-\lambda)(\theta_p + \theta_q)^2 && (2.68)\\ &= \xi^2 && (2.69) \end{aligned}$$

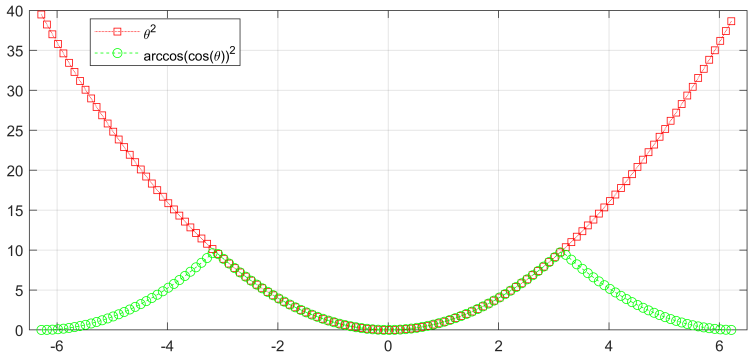


Figure 2.7: $\theta^2 \geq \arccos^2(\cos(\theta))$

Case (**b**): If $0 \leq \theta_p \leq \pi$, $0 \leq \theta_q \leq \pi$. Therefore, $-2\pi \leq \theta_p - \theta_q \leq 2\pi$ and $-2\pi \leq \theta_p + \theta_q \leq 2\pi$. The relationship between θ^2 and $\arccos^2(\cos(\theta))$ are shown in Fig. 2.7.

$$\arccos^2(\cos(\theta)) \leq \theta^2, \quad \theta \in [-2\pi, 2\pi] \tag{2.70}$$

Therefore,

$$\begin{aligned} \rho^2 &\leq \lambda \arccos^2(\cos(\theta_p - \theta_q)) + (1-\lambda)\arccos^2(\cos(\theta_p + \theta_q)) && (2.71)\\ &\leq \lambda(\theta_p - \theta_q)^2 + (1-\lambda)(\theta_p + \theta_q)^2 && (2.72)\\ &= \xi^2 && (2.73) \end{aligned}$$

Combing case (**a**) and (**b**), $\rho^2 \leq \xi^2$. Because of the non-negativeness of ρ and ξ, then it follows that $\angle(\boldsymbol{x}_p, \boldsymbol{x}_q) \leq \|\boldsymbol{\varphi}_p - \boldsymbol{\varphi}_q\|$. □

In fact, lemma 2.4 can be considered as a special case of lemma 2.6. Specifically, lemma 2.4 describes the relationship between between unit quaternions and solid π-ball in 3 dimension. However, lemma 2.6 describes the n-sphere and its corresponding n-dimensional solid ball. Since lemma 2.4 is core part in rotation search theory, which provides globally optimal solutions in 3-dimensional space, lemma 2.6 can be applied in the globally optimal solutions in n-sphere.

Chapter 3

Globally Optimal Direction Estimation in Atlanta World

This chapter is an application of lemma 2.6 in unit sphere ($\mathbb{S}^2$). The core idea is applying the BnB algorithm to solve vertical direction in Atlanta world estimation problem, which inherently includes a unit sphere constraint. This chapter is a modified version of

- Liu, Yinlong, Guang Chen, and Alois Knoll. *Globally optimal vertical direction estimation in Atlanta World*. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020).

It has been reproduced here with the permission of the copyright holder.

3.1 Background

In man-made environments, scenes usually have structural forms (e.g., the layout of buildings and many indoor objects such as furniture), which can be represented by a set of parallel and orthogonal planes [112]. Atlanta world makes an assumption that the man-made scene can be modeled by a horizontal plane (e.g., ground plane) and many vertical planes (e.g., buildings and walls), then the normals of the planes, which are called world frames, can describe the scenes abstractly. In other words, one vertical frame and multiple horizontal frames could represent Atlanta world [113, 114]. Therefore, it is a crucial step to estimate these vertical and horizontal frame directions in computer vision applications, which is named Atlanta frame estimation [114, 115]. More specifically, structural world frame estimation could be utilized as key modules for various high-level vision applications such as scene understanding [112, 116] and SLAM [117, 118].

Mathematically, an orientation in 3D Euclidean space corresponds to a point in the unit sphere (i.e., $\mathbb{S}^2$, $\mathbb{S}^2 = \{\boldsymbol{x} \in \mathbb{R}^3 : \|\boldsymbol{x}\| = 1\}$). This means that the Atlanta frame estimation which estimates multiple orientations is a multiple-clustering (also multi-model fitting) problem in $\mathbb{S}^2$. There have been many general multiple-clustering algorithms [119–121] and some of them have been applied in structural world frame estimation [122, 123]. However, Atlanta frame estimation is not exactly the same as the general multiple-clustering problem. It has some special constraints that all horizontal frames are in a plane and the vertical

frame is parallel to the normal of the plane. These special constraints reflect essential properties of the Atlanta world assumption. If these constraints are omitted, it will not only lead to a significant decrease in accuracy, but also increase the dimensionality of the problem. Furthermore, most of the adopted multiple-clustering algorithms cannot guarantee global optimality when there are many outliers in observations [103, 124]. Therefore, recent developments in structural world frame estimation highlight the imminent need for robust and globally optimal methods by considering the above special orthogonal constraints [93, 115].

Recently, Manhattan frame estimation [112], which is a special case of the Atlanta frame estimation, is solved efficiently by a branch-and-bound (BnB) method with the orthogonal constraints [93]. However, when the BnB method is extended to the Atlanta world [114, 115], two problems appear,

1. The algorithm requires the number of Atlanta frames to be specified, and it is rare for this number to be known in advance. Although an automatic method is proposed to estimate the number of horizontal directions in [115], if it is over- or under-estimated, the global optimum may not occur in the correct direction.

2. It will suffer the curse of dimensionality [125]. There are a considerable number of horizontal directions, whose relationships are unknown, which is different from the Manhattan world assumption. Consequently, the dimensionality of the problem will increase with the number of horizontal frames, and thus the runtime of the BnB algorithm will increase greatly.

We focus on estimating the unique vertical direction instead of all directions in Atlanta world at once. There are two advantages in comparison with the one-time solving-all-directions methods, as follows

1. More flexible. The vertical direction is unique in Atlanta frames, and we can estimate the vertical direction even though we do not know the total number of the horizontal directions. Additionally, we can also estimate the vertical direction from some irregular Atlanta world scenes (e.g., cylindrical buildings in Atlanta world, whose horizontal directions number $\rightarrow \infty$).

2. More efficient. Vertical direction estimation is solved in a closed two-dimensional space $\mathbb{S}^2$, which is a low-dimensional problem. In other words, only estimating vertical direction can significantly avoid the curse of dimensionality in Atlanta world.

Furthermore, estimating the vertical direction first is usually favorable for following operations in practical applications (e.g., scene classification [126], parsing indoor scenes [127] and point set registration [128]). Specially, it is also helpful for estimating other horizontal Atlanta frames, because given the vertical direction, all other horizontal directions will be in a plane, and estimating the other horizontal directions will be a one-dimensional clustering problem in angular space [115]. In other words, given the vertical direction in Atlanta world, it is

easy to estimate other horizontal directions with or without knowing the number of horizontal frames [115, 129].

3.2 Related Work

There is a large body of literature concerned with structural world frame estimation [93, 112, 115, 130]. Since it is a clustering problem in $\mathbb{S}^2$ with some orthogonal constraints, we first review the works that apply the classical clustering or fitting method. With the definition of Atlanta world, *Expectation Maximization* (EM) type algorithms, which are popular for solving the chicken-and-egg problems [49], are applied in direction estimation [113]. However, the EM-type algorithms are local methods and have no guarantee of global optimality. Therefore, there is an evident risk of local minima, and their performances rely heavily on good initialization [131]. Besides, the *RANdom SAmple Consensus* (RANSAC) [34] type multi-structure estimation algorithms (e.g., T-linkage [47] and J-linkage [132]) are applied in structural direction estimation [122, 123]. These RANSAC-type methods are fast, accurate and have the best performances in many cases, but their solution is sub-optimal due to their obvious heuristic nature [115]. More recently, Straub et al. [112] propose a real-time capable inference algorithm by considering the orthogonal constraints, which uses an adaptive Markov-Chain Monte-Carlo sampling algorithm.

However, when there are lots of outliers in the measurements, these above methods are sub-optimal [115]. To assure global optimality, J. Bazin et al. propose globally optimal methods [93, 103, 114, 115, 124] by applying a branch-and-bound algorithm to solve a consensus set maximization problem. The fundamental theory of these global methods is rotation search [83, 92]. Specifically, the problem is solved by combining *Interval Analysis* theory with BnB algorithm in [124]. By contrast, the method in [103] is a natural application of Hartley and Kahl's rotation search theory in $\mathbb{SO}(3)$ [92]. Furthermore, 2D-EGI (Extended Gaussian Image) and its integral image are applied in [93] to accelerate the calculation of the bounds in rotation search. Recently, rotation search theory is also extended to Atlanta frame estimation in [114, 115].

However, Atlanta world is more complicated than Manhattan world geometrically, since it has more than three frames. Consequently, the global searching method in [114] requires the number of horizontal directions to be hand-tuned according to the scene, which seems unrealistic in practical applications. Therefore, an automatic two-stage method (meta-BnB) is proposed in [115] to estimate the number of directions. Concretely, it first searches the vertical direction and the horizontal plane in $\mathbb{SO}(3)$, then it estimates the horizontal directions in one-dimensional angle space. It is worth noting that the meta-BnB is also based on rotation search theory in $\mathbb{SO}(3)$. However, searching vertical directions is inherently optimized in $\mathbb{S}^2$, whose dimensionality is less than that of $\mathbb{SO}(3)$.

Since the rotation search theory is closely related to our work, we then briefly review the rotation search theory. The rotation search theory has achieved great

success in geometric vision problems, for example, point set registration [52, 66], camera calibration [133, 134] and relative pose estimation [92, 135]. Because of the great success of rotation search, there have been several works focusing on improving the efficiency of the algorithm [52, 93, 136, 137].

More specifically, most of the rotation search methods rely lemma 2.1 and lemma 2.2 in chapter 2. For completeness, we still list these two lemmas here:

Lemma. *For* $\forall \boldsymbol{v} \in \mathbb{S}^2$, $\mathbf{R}_a, \mathbf{R}_b \in \mathbb{SO}(3)$, *then*

$$\angle(\mathbf{R}_a\boldsymbol{v}, \mathbf{R}_b\boldsymbol{v}) \leq \angle(\mathbf{R}_a, \mathbf{R}_b) \tag{3.1}$$

where $\angle(\mathbf{R}_a, \mathbf{R}_b)$ is the angle lying in the range $[0, \pi]$ of the rotation $\mathbf{R}_a\mathbf{R}_b^{-1}$ and $\angle(\cdot, \cdot)$ denotes the angular distance between vectors.

Lemma. *For* $\forall \mathbf{R}_a, \mathbf{R}_b \in \mathbb{SO}(3)$, *then*

$$\angle(\mathbf{R}_a, \mathbf{R}_b) \leq \|\boldsymbol{r}_a - \boldsymbol{r}_b\| \tag{3.2}$$

where $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$ are their corresponding angle-axis representations. These two lemmas are the basis for the success of the rotation search theory.

Additionally, it is also worth noting that the rotation search usually means optimization in $\mathbb{SO}(3)$, which is closely related to $\mathbb{S}^3$. Precisely, the homomorphism from a unit quaternion sphere (i.e., $\mathbb{S}^3$) to $\mathbb{SO}(3)$ is a two-to-one mapping, and then the search domain $\mathbb{SO}(3)$ may be represented by a hemisphere (including equator) of the unit quaternion sphere [106, 137]. However, the estimation of directions in three-dimensional Euclidean space (i.e., Manhattan or Atlanta frame) is inherently optimized in $\mathbb{S}^2$. Unfortunately, there is still a lack of rigid theories regarding globally optimal algorithms in $\mathbb{S}^2$. In order to estimate the vertical directions in Atlanta world, we originally propose some new and solid mathematical conclusions about searching in $\mathbb{S}^2$.

3.3 Contribution

To overcome the curse of the dimensionality and avoid the difficulty of requiring users to specify the number of Atlanta frames, we propose a novel method for vertical direction estimation in Atlanta world. The contributions of this chapter are mainly as follows:

- We propose a global searching method for estimating vertical direction, which is different from conventional rotation search in $\mathbb{SO}(3)$ [115]. Since the domain of the vertical directions is inherently in $\mathbb{S}^2$, our searching method is more efficient at vertical direction estimation.
- Four new different bounds for BnB algorithm are investigated. In contrast to rotation search theory in $\mathbb{SO}(3)$, more parametrizations for the hemisphere are considered, including exponential mapping, stereographic projection and the sphere coordinate system. To the best of our knowledge, we are the first to propose such bounds in $\mathbb{S}^2$ to solve the structural world frame estimation problem.

3.4 Methods

3.4.1 Problem Formulation

In this chapter, we estimate the vertical direction from the surface normals in Atlanta world. We denote the input normal set as $\mathcal{N} = \{\boldsymbol{n}_j\}_{j=1}^{N}$, where $\boldsymbol{n}_j \in \mathbb{S}^2$ is the j-th effective unit normal, and N is the number of input normals. In addition, the unknown-but-sought vertical direction is denoted as $\boldsymbol{v}$. It is in a hemisphere ($\mathbb{S}^{2+}$), which is defined as:

$$\mathbb{S}^{2+} = \{\boldsymbol{x} \in \mathbb{S}^2 | x_3 \geq 0\} \tag{3.3}$$

where $\boldsymbol{x} = [x_1, x_2, x_3]^T$ is a unit vector in $\mathbb{R}^3$.

To robustly estimate vertical direction, we then apply the inlier maximization approach to formulate the objective function as

$$\max_{\boldsymbol{v} \in \mathbb{S}^{2+}} \sum_{j=1}^{N} \mathbb{I}\Big(\mathcal{S}_j^{\perp} \vee \mathcal{S}_j^{\parallel^+} \vee \mathcal{S}_j^{\parallel^-}\Big) \tag{3.4a}$$

$$\mathcal{S}_j^{\parallel^+} = \mathbb{I}\big(\angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon\big) \tag{3.4b}$$

$$\mathcal{S}_j^{\parallel^-} = \mathbb{I}\big(\angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon\big) \tag{3.4c}$$

$$\mathcal{S}_j^{\perp} = \mathbb{I}\big(|\angle(\boldsymbol{v}, \boldsymbol{n}_j) - \frac{\pi}{2}| \leq \epsilon\big) \tag{3.4d}$$

where $\mathbb{I}(\cdot)$ is an indicator function which returns 1 if the condition $\cdot$ is true and 0 otherwise. $\vee$ is the logical *OR* operation. $|\cdot|$ is the *abs* function and $0 < \epsilon < \pi/2$ is the inlier threshold. Eq. (3.4b), (3.4c) and (3.4d) mean that only the surface normals, which are parallel or perpendicular to vertical direction, are inliers.

Additionally, because $\boldsymbol{n}_j^T \boldsymbol{v} = \cos(\angle(\boldsymbol{v}, \boldsymbol{n}_j))$, and when $x \in [0, \pi]$, $\cos(x)$ is a monotonically decreasing function, then an equivalent formulation can be given by

$$\max_{\boldsymbol{v} \in \mathbb{S}^{2+}} \sum_{j=1}^{N} \mathbb{I}\Big(Q_j^{\parallel} \vee Q_j^{\perp}\Big) \tag{3.5a}$$

$$Q_j^{\parallel} = \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}| \geq \cos(\epsilon)\big) \tag{3.5b}$$

$$Q_j^{\perp} = \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}| \leq \sin(\epsilon)\big) \tag{3.5c}$$

Note that rotation search based method [115] finds an optimal rotated motion $\mathbf{R}$ rather than the optimal direction vector $\boldsymbol{v}$ directly. Concretely, given an initial direction vector $\boldsymbol{v}_0 = [0, 0, 1]^T$ and because $\mathbf{R} \in \mathbb{SO}(3)$, then $\boldsymbol{v} = \mathbf{R}\boldsymbol{v}_0 \in \mathbb{S}^2$. For estimating vertical direction, it is sufficient to search the entire rotation domain and find the optimal $\mathbf{R}$ to satisfy that $\mathbf{R}\boldsymbol{v}_0$ is the optimal vertical direction.

3.4.2 Branch-and-Bound Algorithm

Finding the optimal $\boldsymbol{v} \in \mathbb{S}^{2+}$ to maximize the cardinality of the inlier set is by no means a trivial problem [138, 139]. Additionally, the outlier observations, which are unavoidable in the real applications, increase the *hardness* of the estimation problem, because it is well known that exactly distinguish inliers and outliers to a general outlier-contaminated robust estimation problem is NP-hard [78, 140, 141].

To obtain the robust optimal vertical direction, we then use the BnB algorithm. The BnB algorithm is one of the most commonly used tools for solving NP-hard optimization problems, and it is widely applied in many global optimization problems [142]. Briefly, the BnB algorithm recursively divides the search space into smaller spaces and estimates the upper bound and lower bound of the optimum in each subspace. Then, it removes the sub-spaces which cannot produce a better solution than the best one found so far by the algorithm. This process is repeated until the best optimum is found within the desired accuracy. The BnB algorithm for estimating vertical direction globally in Atlanta world is outlined in **Algorithm 1**. It is worth noting that the algorithm only needs the surface normals and the inlier threshold as the inputs. Specially, it needs no the prior knowledge of the number of horizontal frames.

Algorithm 1: Globally Searching Vertical Direction

Input: surface normal set $\{\boldsymbol{n}_j\}_{j=1}^{N}$, inlier threshold ϵ
Output: optimal vertical direction $\boldsymbol{v}^*$

Initialize the searching domain $\mathbb{D}$, upper bound $U \leftarrow N$, lower bound $L \leftarrow 0$, the best branch $\mathbb{B} \leftarrow \mathbb{D}$ and a queue $q \leftarrow \emptyset$;
while $|U - L| > 0$ **do**
    Divide the best branch $\mathbb{B}$ into sub-branches;
    Estimate the bounds for each sub-branch $\{L_i, U_i\}_{i=1}^{N}$;
    Add the sub-branches with their bounds into q;
    Update $L \leftarrow \max\{L_i\}$, $U \leftarrow \max\{U_i\}$ for all branches in q;
    Remove the branch that $U_i < L$ in q ;
    Update the best branch $\mathbb{B}$, which has the maximum upper bound in q;
    Remove the best branch from q;
end
$\boldsymbol{v}^* \leftarrow$ center point in best branch $\mathbb{B}$

The key of the BnB algorithm is estimating the upper and lower bounds of the optimum in each subspace tightly and efficiently. Accordingly, two general bounds are proposed as follows:

Proposition 3.1 (General bounds-1)**.** *Given a branch $\mathbb{B}$, if $\exists \boldsymbol{v}_c \in \mathbb{B}$, $\forall \boldsymbol{v} \in \mathbb{B}, \overline{\phi}_j \triangleq \max \angle(\boldsymbol{v}, \boldsymbol{n}_j)$, $\underline{\phi}_j \triangleq \min \angle(\boldsymbol{v}, \boldsymbol{n}_j)$, and $\phi_j = \angle(\boldsymbol{v}_c, \boldsymbol{n}_j)$, then the upper bound can be:*

$$U_s(\mathbb{B}) = \sum_{j=1}^{N} \mathbb{I}\Big(\overline{\mathcal{S}}_j^{\perp}\big(\underline{\phi}_j, \overline{\phi}_j\big) \vee \overline{\mathcal{S}}_j^{\|^+}\big(\underline{\phi}_j\big) \vee \overline{\mathcal{S}}_j^{\|^-}\big(\overline{\phi}_j\big)\Big) \tag{3.6a}$$

$$\overline{\mathcal{S}}_j^{\|^+}\big(\underline{\phi}_j\big) = \mathbb{I}\big(\underline{\phi}_j \leq \epsilon\big) \tag{3.6b}$$

$$\overline{\mathcal{S}}_j^{\|^-}\big(\overline{\phi}_j\big) = \mathbb{I}\big(\overline{\phi}_j \geq \pi - \epsilon\big) \tag{3.6c}$$

$$\overline{\mathcal{S}}_j^{\perp}\big(\underline{\phi}_j, \overline{\phi}_j\big) = \mathbb{I}\big(\underline{\phi}_j - \epsilon \leq \frac{\pi}{2} \leq \overline{\phi}_j + \epsilon\big) \tag{3.6d}$$

the lower bound can be:

$$L_s(\mathbb{B}) = \sum_{j=1}^{N} \mathbb{I}\Big(\underline{\mathcal{S}}_j^{\perp}\big(\phi_j\big) \vee \underline{\mathcal{S}}_j^{\|^+}\big(\phi_j\big) \vee \underline{\mathcal{S}}_j^{\|^-}\big(\phi_j\big)\Big) \tag{3.7a}$$

$$\underline{\mathcal{S}}_j^{\|^+}\big(\phi_j\big) = \mathbb{I}\big(\phi_j \leq \epsilon\big) \tag{3.7b}$$

$$\underline{\mathcal{S}}_j^{\|^-}\big(\phi_j\big) = \mathbb{I}\big(\phi_j \geq \pi - \epsilon\big) \tag{3.7c}$$

$$\underline{\mathcal{S}}_j^{\perp}\big(\phi_j\big) = \mathbb{I}\big(|\phi_j - \frac{\pi}{2}| \leq \epsilon\big) \tag{3.7d}$$

Proof. To prove general bounds-1 are effective bounds to be nested in the BnB algorithm, we need to prove Eq. (3.6) is an upper bound and Eq. (3.7) is a lower bound. Besides, we must prove that as $\mathbb{B}$ collapses to a single point, the gap of the upper and lower bounds converges to zero.

To prove Eq. (3.6) is an upper bound, it is sufficient to show that $\forall \boldsymbol{v} \in \mathbb{B}$, if $\boldsymbol{n}_j$ contribute 1 to the objective, it must contribute 1 to Eq. (3.6).

Observe

$$\begin{aligned}
\mathcal{S}_j^{\|^+} &= \mathbb{I}\big(\angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon\big) = 1 && (3.8)\\
&\Rightarrow \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon && (3.9)\\
&\Rightarrow \underline{\phi}_j \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon && (3.10)\\
&\Rightarrow \mathbb{I}\big(\underline{\phi}_j \leq \epsilon\big) = 1 && (3.11)\\
&\Rightarrow \overline{\mathcal{S}}_j^{\|^+} = 1 && (3.12)
\end{aligned}$$

Similarly,

$$\begin{aligned}
\mathcal{S}_j^{\|^-} &= \mathbb{I}\big(\angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon\big) = 1 && (3.13)\\
&\Rightarrow \angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon && (3.14)\\
&\Rightarrow \overline{\phi}_j \geq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon && (3.15)\\
&\Rightarrow \mathbb{I}\big(\overline{\phi}_j \geq \pi - \epsilon\big) = 1 && (3.16)\\
&\Rightarrow \overline{\mathcal{S}}_j^{\|^-} = 1 && (3.17)
\end{aligned}$$

And

$$\mathcal{S}_j^{\perp} = \mathbb{I}\big(|\angle(\boldsymbol{v}, \boldsymbol{n}_j) - \frac{\pi}{2}| \leq \epsilon\big) = 1 \tag{3.18}$$

$$\Rightarrow \angle(\boldsymbol{v}, \boldsymbol{n}_j) - \epsilon \leq \frac{\pi}{2} \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) + \epsilon \tag{3.19}$$

$$\Rightarrow \underline{\phi}_j - \epsilon \leq \frac{\pi}{2} \leq \overline{\phi}_j + \epsilon \tag{3.20}$$

$$\Rightarrow \mathbb{I}\big(\underline{\phi}_j - \epsilon \leq \frac{\pi}{2} \leq \overline{\phi}_j + \epsilon\big) = 1 \tag{3.21}$$

$$\Rightarrow \overline{\mathcal{S}}_j^{\perp} = 1 \tag{3.22}$$

Thus,

$$\mathbb{I}\Big(\mathcal{S}_j^{\perp} \vee \mathcal{S}_j^{\parallel^+} \vee \mathcal{S}_j^{\parallel^-}\Big) = 1 \Rightarrow \mathbb{I}\Big(\overline{\mathcal{S}}_j^{\perp} \vee \overline{\mathcal{S}}_j^{\parallel^+} \vee \overline{\mathcal{S}}_j^{\parallel^-}\Big) = 1 \tag{3.23}$$

Then,

$$\mathbb{I}\Big(\overline{\mathcal{S}}_j^{\perp} \vee \overline{\mathcal{S}}_j^{\parallel^+} \vee \overline{\mathcal{S}}_j^{\parallel^-}\Big) \geq \mathbb{I}\Big(\mathcal{S}_j^{\perp} \vee \mathcal{S}_j^{\parallel^+} \vee \mathcal{S}_j^{\parallel^-}\Big) \tag{3.24}$$

Therefore,

$$\sum_{j=1}^{N} \mathbb{I}\Big(\overline{\mathcal{S}}_j^{\perp} \vee \overline{\mathcal{S}}_j^{\parallel^+} \vee \overline{\mathcal{S}}_j^{\parallel^-}\Big) \geq \max_{\boldsymbol{v}\in\mathbb{B}} \sum_{j=1}^{N} \mathbb{I}\Big(\mathcal{S}_j^{\perp} \vee \mathcal{S}_j^{\parallel^+} \vee \mathcal{S}_j^{\parallel^-}\Big) \tag{3.25}$$

Thus, $U_s(\mathbb{B})$ is an upper bound.

For the lower bound, it is understandable that the maximum value in the subspace is no less than the function value at any specific point.

Lastly, when the branch $\mathbb{B}$ collapses to a single point $\boldsymbol{v}_b$, then $\boldsymbol{v}_c = \boldsymbol{v} = \boldsymbol{v}_b$ and $\overline{\phi}_j = \underline{\phi}_j = \phi_j = \angle(\boldsymbol{v}, \boldsymbol{n}_j) = \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) = \angle(\boldsymbol{v}_b, \boldsymbol{n}_j)$. Hence, $U_s(\mathbb{B}) = L_s(\mathbb{B})$. □

Proposition 3.2 (General bounds-2)**.** *Given a branch* $\mathbb{B}$*, if* $\exists \boldsymbol{v}_c \in \mathbb{B}$, $\forall \boldsymbol{v} \in \mathbb{B}, \angle(\boldsymbol{v}, \boldsymbol{v}_c) \leq \max \angle(\boldsymbol{v}, \boldsymbol{v}_c) \triangleq \psi$*, then the upper bound can be:*

$$U_q(\mathbb{B}) = \sum_{j=1}^{N} \mathbb{I}\Big(\overline{Q}_j^{\parallel}\big(\boldsymbol{v}_c, \psi\big) \vee \overline{Q}_j^{\perp}\big(\boldsymbol{v}_c, \psi\big)\Big) \tag{3.26a}$$

$$\overline{Q}_j^{\parallel}\big(\boldsymbol{v}_c, \psi\big) = \mathbb{I}\Big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \geq \cos\big(\min\{\epsilon + \psi, \pi/2\}\big)\Big) \tag{3.26b}$$

$$\overline{Q}_j^{\perp}\big(\boldsymbol{v}_c, \psi\big) = \mathbb{I}\Big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \leq \sin\big(\min\{\epsilon + \psi, \pi/2\}\big)\Big) \tag{3.26c}$$

the lower bound can be:

$$L_q(\mathbb{B}) = \sum_{j=1}^{N} \mathbb{I}\Big(\underline{Q}_j^{\parallel}\big(\boldsymbol{v}_c\big) \vee \underline{Q}_j^{\perp}\big(\boldsymbol{v}_c\big)\Big) \tag{3.27a}$$

$$\underline{Q}_j^{\parallel}\big(\boldsymbol{v}_c\big) = \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \geq \cos(\epsilon)\big) \tag{3.27b}$$

$$\underline{Q}_j^{\perp}\big(\boldsymbol{v}_c\big) = \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \leq \sin(\epsilon)\big) \tag{3.27c}$$

Proof. To prove general bounds-2 are effective bounds to be nested in the BnB algorithm, we first prove Eq. (3.26) is an upper bound and Eq. (3.27) is a lower bound. Then we prove that as $\mathbb{B}$ collapses to a single point, the gap of the upper and lower bounds converges to zero.

To prove Eq. (3.26) is an upper bound, it is sufficient to show that $\forall \boldsymbol{v} \in \mathbb{B}$, if $\boldsymbol{n}_j$ contribute 1 to the objective, it must contribute 1 to Eq. (3.26).

First, if $\epsilon + \psi \geq \pi/2$, then $\overline{Q}_j^{\parallel} = \overline{Q}_j^{\perp} = 1$. Obviously, $U_q(\mathbb{B})$ is an upper bound.

If $0 \leq \epsilon + \psi < \pi/2$, observe

$$Q_j^{\parallel} = \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}| \geq \cos(\epsilon)\big) = 1 \tag{3.28}$$

$$\Rightarrow \begin{cases} \mathbb{I}\big(\boldsymbol{n}_j^T \boldsymbol{v} \geq \cos(\epsilon)\big) = 1 & \text{(3.29a)} \\ \mathbb{I}\big(\boldsymbol{n}_j^T \boldsymbol{v} \leq \cos(\pi - \epsilon)\big) = 1 & \text{(3.29b)} \end{cases}$$

For Eq. (3.29a),

$$\begin{aligned} & \mathbb{I}\big(\boldsymbol{n}_j^T \boldsymbol{v} \geq \cos(\epsilon)\big) = 1 && (3.30)\\ \Rightarrow\ & \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon && (3.31)\\ \Rightarrow\ & \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \angle(\boldsymbol{v}_c, \boldsymbol{v}) \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon && (3.32)\\ \Rightarrow\ & \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \psi \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \epsilon && (3.33)\\ \Rightarrow\ & \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \leq \epsilon + \psi && (3.34)\\ \Rightarrow\ & \mathbb{I}\left(\boldsymbol{n}_j^T \boldsymbol{v}_c \geq \cos\left(\min\{\epsilon + \psi, \pi/2\}\right)\right) = 1 && (3.35)\\ \Rightarrow\ & \overline{Q}_j^{\parallel} = 1 && (3.36) \end{aligned}$$

For Eq. (3.29b),

$$\begin{aligned} & \mathbb{I}\big(\boldsymbol{n}_j^T \boldsymbol{v} \leq \cos(\pi - \epsilon)\big) = 1 && (3.37)\\ \Rightarrow\ & \angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon && (3.38)\\ \Rightarrow\ & \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \angle(\boldsymbol{v}_c, \boldsymbol{v}) \geq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon && (3.39)\\ \Rightarrow\ & \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \psi \geq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \geq \pi - \epsilon && (3.40)\\ \Rightarrow\ & \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \geq \pi - \epsilon - \psi && (3.41)\\ \Rightarrow\ & \mathbb{I}\left(\boldsymbol{n}_j^T \boldsymbol{v}_c \leq -\cos\left(\min\{\epsilon + \psi, \pi/2\}\right)\right) = 1 && (3.42)\\ \Rightarrow\ & \overline{Q}_j^{\parallel} = 1 && (3.43) \end{aligned}$$

Moreover,

$$Q_j^{\perp} = \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}| \leq \sin(\epsilon)\big) = 1 \tag{3.44}$$

$$\Rightarrow \pi/2 - \epsilon \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \pi/2 + \epsilon \tag{3.45}$$

$$\Rightarrow \begin{cases} \pi/2 - \epsilon \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \angle(\boldsymbol{v}_c, \boldsymbol{v}) \\ \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \angle(\boldsymbol{v}_c, \boldsymbol{v}) \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \leq \pi/2 + \epsilon \end{cases} \tag{3.46}$$

$$\Rightarrow \begin{cases} \pi/2 - \epsilon - \angle(\boldsymbol{v}_c, \boldsymbol{v}) \leq \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \\ \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \leq \pi/2 + \epsilon + \angle(\boldsymbol{v}_c, \boldsymbol{v}) \end{cases} \tag{3.47}$$

$$\Rightarrow \pi/2 - \epsilon - \psi \leq \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \leq \pi/2 + \epsilon + \psi \tag{3.48}$$

$$\Rightarrow -\sin(\epsilon + \psi) \leq \boldsymbol{n}_j^T \boldsymbol{v}_c \leq \sin(\epsilon + \psi) \tag{3.49}$$

$$\Rightarrow \mathbb{I}\left(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \leq \sin(\min\{\epsilon + \psi, \pi/2\})\right) = 1 \tag{3.50}$$

$$\Rightarrow \overline{Q}_j^{\perp} = 1 \tag{3.51}$$

Thus,

$$\mathbb{I}\Big(Q_j^{\parallel} \vee Q_j^{\perp}\Big) = 1 \Rightarrow \mathbb{I}\Big(\overline{Q}_j^{\parallel} \vee \overline{Q}_j^{\perp}\Big) = 1 \tag{3.52}$$

Therefore,

$$\mathbb{I}\Big(\overline{Q}_j^{\parallel} \vee \overline{Q}_j^{\perp}\Big) \geq \mathbb{I}\Big(Q_j^{\parallel} \vee Q_j^{\perp}\Big) \tag{3.53}$$

then,

$$\sum_{j=1}^{N} \mathbb{I}\Big(\overline{Q}_j^{\parallel} \vee \overline{Q}_j^{\perp}\Big) \geq \max_{\boldsymbol{v} \in \mathbb{B}} \sum_{j=1}^{N} \mathbb{I}\Big(Q_j^{\parallel} \vee Q_j^{\perp}\Big) \tag{3.54}$$

Thus, $U_q(\mathbb{B})$ is an upper bound.

For the lower bound, the proof is easy, which is that the function value at a specific point within the domain is less than or equal to the maximum.

Lastly, when the branch $\mathbb{B}$ collapses to a single point $\boldsymbol{v}_b$, then $\boldsymbol{v}_c = \boldsymbol{v} = \boldsymbol{v}_b$ and $\psi = 0$. Hence, $U_q(\mathbb{B}) = L_q(\mathbb{B})$. □

In fact, if they have the same $\boldsymbol{v}_c$ in both general bounds, then $L_s = L_q$. The main difference between general bounds-1 and general bounds-2 is the calculation of the upper bound. More specifically, given a subspace $\mathbb{B}$,

$$U_s(\mathbb{B}) \leq U_q(\mathbb{B}) \tag{3.55}$$

which means general bounds-1 is tighter than general bounds-2.

Proof. To prove $U_s(\mathbb{B}) \leq U_q(\mathbb{B})$, it is sufficient to show that given a branch $\mathbb{B}$, if $\boldsymbol{n}_j$ contribute 1 to $U_s(\mathbb{B})$, it must contribute 1 to $U_q(\mathbb{B})$.

First, $\boldsymbol{v}_c \in \mathbb{B}$, $\forall \boldsymbol{v} \in \mathbb{B}$,

$$\angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \angle(\boldsymbol{v}_c, \boldsymbol{v}) \geq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \tag{3.56}$$

$$\angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \angle(\boldsymbol{v}_c, \boldsymbol{v}) \leq \angle(\boldsymbol{v}, \boldsymbol{n}_j) \tag{3.57}$$

Then,

$$\angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \psi \geq \overline{\phi}_j \tag{3.58}$$
$$\angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \psi \leq \underline{\phi}_j \tag{3.59}$$

If $\epsilon + \psi \geq \pi/2$, then $\overline{Q}_j^{\parallel} = \overline{Q}_j^{\perp} = 1$. Obviously, $U_s(\mathbb{B}) \leq U_q(\mathbb{B})$.
If $0 \leq \epsilon + \psi < \pi/2$, observe

$$\overline{\mathcal{S}}_j^{\parallel^+} = \mathbb{I}\big(\underline{\phi}_j \leq \epsilon\big) = 1 \tag{3.60}$$
$$\Rightarrow \underline{\phi}_j \leq \epsilon \tag{3.61}$$
$$\Rightarrow \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \psi \leq \epsilon \tag{3.62}$$
$$\Rightarrow \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \leq \epsilon + \psi \tag{3.63}$$
$$\Rightarrow \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \geq \cos(\min\{\epsilon + \psi, \pi/2\})\big) = 1 \tag{3.64}$$
$$\Rightarrow \overline{Q}_j^{\parallel} = 1 \tag{3.65}$$

Similarly,

$$\overline{\mathcal{S}}_j^{\parallel^-} = \mathbb{I}\big(\overline{\phi}_j \geq \pi - \epsilon\big) = 1 \tag{3.66}$$
$$\Rightarrow \overline{\phi}_j \geq \pi - \epsilon \tag{3.67}$$
$$\Rightarrow \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \psi \geq \pi - \epsilon \tag{3.68}$$
$$\Rightarrow \mathbb{I}\big(\boldsymbol{n}_j^T \boldsymbol{v}_c \leq -\cos(\min\{\epsilon + \psi, \pi/2\})\big) = 1 \tag{3.69}$$
$$\Rightarrow \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \geq \cos(\min\{\epsilon + \psi, \pi/2\})\big) = 1 \tag{3.70}$$
$$\Rightarrow \overline{Q}_j^{\parallel} = 1 \tag{3.71}$$

And

$$\overline{\mathcal{S}}_j^{\perp} = \mathbb{I}\big(\underline{\phi}_j - \epsilon \leq \frac{\pi}{2} \leq \overline{\phi}_j + \epsilon\big) = 1 \tag{3.72}$$
$$\Rightarrow \underline{\phi}_j - \epsilon \leq \frac{\pi}{2} \leq \overline{\phi}_j + \epsilon \tag{3.73}$$
$$\Rightarrow \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) - \psi - \epsilon \leq \frac{\pi}{2} \leq \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) + \psi + \epsilon \tag{3.74}$$
$$\Rightarrow \pi/2 - \epsilon - \psi \leq \angle(\boldsymbol{v}_c, \boldsymbol{n}_j) \leq \pi/2 + \epsilon + \psi \tag{3.75}$$
$$\Rightarrow -\sin(\epsilon + \psi) \leq \boldsymbol{n}_j^T \boldsymbol{v}_c \leq \sin(\epsilon + \psi) \tag{3.76}$$
$$\Rightarrow \mathbb{I}\big(|\boldsymbol{n}_j^T \boldsymbol{v}_c| \leq \sin(\min\{\epsilon + \psi, \pi/2\})\big) = 1 \tag{3.77}$$
$$\Rightarrow \overline{Q}_j^{\perp} = 1 \tag{3.78}$$

Thus,

$$\mathbb{I}\Big(\overline{\mathcal{S}}_j^{\perp} \vee \overline{\mathcal{S}}_j^{\parallel^+} \vee \overline{\mathcal{S}}_j^{\parallel^-}\Big) = 1 \Rightarrow \mathbb{I}\Big(\overline{Q}_j^{\parallel} \vee \overline{Q}_j^{\perp}\Big) = 1 \tag{3.79}$$

Therefore,

$$\mathbb{I}\Big(\overline{\mathcal{S}}_j^{\perp} \vee \overline{\mathcal{S}}_j^{\parallel^+} \vee \overline{\mathcal{S}}_j^{\parallel^-}\Big) \leq \mathbb{I}\Big(\overline{Q}_j^{\parallel} \vee \overline{Q}_j^{\perp}\Big) \tag{3.80}$$

Finally, $U_s(\mathbb{B}) \leq U_q(\mathbb{B})$. □

Above discussions just provide theoretical bounds in the BnB algorithm. In the following sections, we introduce how to calculate the upper bounds in detail.

3.4.3 Parametrizing the Search Domain

Before estimating the bounds in BnB algorithm, we must first parametrize the solution space. In this section, we first recall the parametrization of $\mathbb{SO}(3)$ in the rotation search theory [92, 115], and introduce three different parametrizations of $\mathbb{S}^{2+}$. Furthermore, we analyze the similarities and differences in the parametrizations between $\mathbb{SO}(3)$ and $\mathbb{S}^{2+}$.

3.4.3.1 Parametrization of $\mathbb{SO}(3)$

It is well known that the rotation space $\mathbb{SO}(3)$ can be minimally parametrized with the angle-axis vector, whose norm is the angle of rotation, and whose direction is the axis of the rotation. Therefore, the space of all 3D rotations can be represented by a solid ball of radius π in $\mathbb{R}^3$ [66]. Furthermore, the π-ball is usually relaxed to a 3D cube for ease of manipulation in the BnB algorithm. Thus lemma 2.1 and 2.2 can be used to efficiently estimate the bounds [115].

3.4.3.2 Parametrization of $\mathbb{S}^{2+}$: Exponential Mapping

Geometrically, $\mathbb{S}^{2+}$ is a hemisphere in three-dimensional Euclidean space, and it is inherently a two-dimensional closed space. In order to parameterize $\mathbb{S}^{2+}$ minimally, we apply an exponential mapping method to map the hemisphere to a 2D disk. Concretely, let $\boldsymbol{v} = [v_1, v_2, v_3]^T \in \mathbb{S}^{2+}$, then it can be represented by its corresponding point $\boldsymbol{d} \in \mathbb{R}^2$ in the disk,

$$\boldsymbol{v}^T = \left[\sin(\theta)\hat{\boldsymbol{d}}^T, \cos(\theta)\right] \tag{3.81}$$

where $\theta \in [0, \pi/2]$, $\hat{\boldsymbol{d}}$ is a unit vector in $\mathbb{R}^2$ and $\boldsymbol{d} = \theta\hat{\boldsymbol{d}}$. Note that the domain of θ corresponds to $v_3 \geqslant 0$, and geometrically, θ is the radius of the disk. In the BnB algorithm, a square (side=π) circumscribing the mapped disk area is used as the vertical direction domain for ease of manipulation. The mapping $\boldsymbol{v} \rightleftharpoons \boldsymbol{d}$ is similar to the mapping from $\mathbb{SO}(3)$ to the 3D solid π-ball in rotation search theory [92]. More specifically, the exponential mapping is closely related to Lie theory [111, 143]. However, in this thesis we will not rely on any knowledge of the Lie groups theory without distracting readers' attention and focus on the direction estimation problem.

According to lemma 2.6 in Chapter 2, we have

Proposition 3.3. *For* $\forall \boldsymbol{v}_a, \boldsymbol{v}_b \in \mathbb{S}^{2+}$, *then*

$$\angle(\boldsymbol{v}_a, \boldsymbol{v}_b) \leq \|\boldsymbol{d}_a - \boldsymbol{d}_b\| \tag{3.82}$$

where $\boldsymbol{d}_a, \boldsymbol{d}_b$ are corresponding points of $\boldsymbol{v}_a, \boldsymbol{v}_b$ in the 2D disk(see Fig. 3.1).

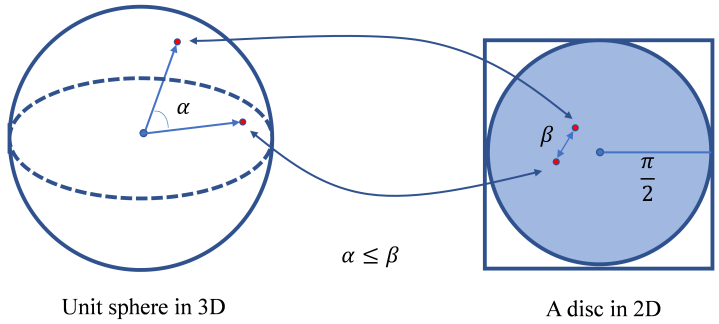


Figure 3.1: A visual interpretation of Proposition 3.3.

3.4.3.3 Parametrization of $\mathbb{S}^{2+}$: Stereographic Projection

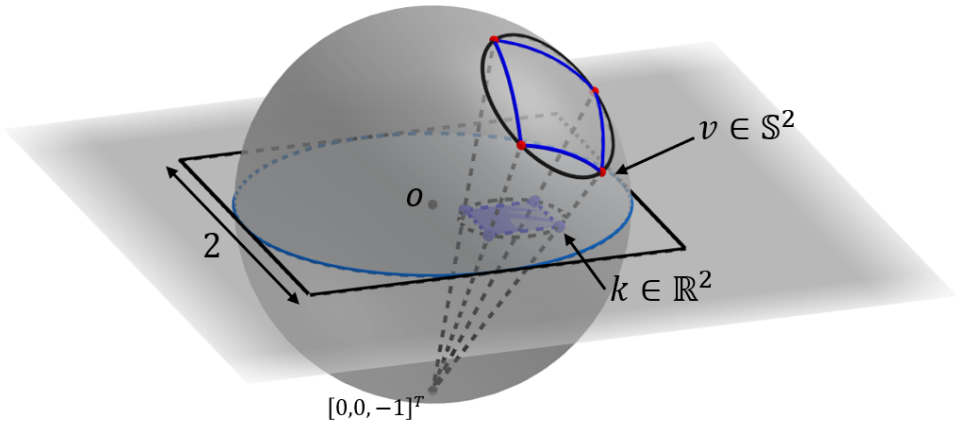


Figure 3.2: Visualization of stereographic projection. A point $\boldsymbol{v} \in \mathbb{S}^{2+}$ corresponds to a point $\boldsymbol{k} \in \mathbb{R}^2$. A square-shaped branch is relaxed to a circle, which corresponds to an umbrella-shaped region in $\mathbb{S}^{2+}$.

In geometry, stereographic projection is a particular mapping that can project a hemisphere to a disk in plane, which means we can also represent the $\mathbb{S}^{2+}$ minimally by applying stereographic projection.

The stereographic projection is visually described as Fig. 3.2. We denote a point $\boldsymbol{k} = [k_1, k_2]^T \in \mathbb{R}^2$ in the equatorial plane and its corresponding point $\boldsymbol{v} = [v_1, v_2, v_3]^T \in \mathbb{S}^{2+}$, and if the projection pole is at $[0, 0, -1]^T$ (south pole, see [144]), then we have:

$$\boldsymbol{k}^T = \left[\frac{v_1}{1+v_3}, \frac{v_2}{1+v_3}\right] \tag{3.83}$$

$$\boldsymbol{v}^T = \left[\frac{2k_1}{1+k_1^2+k_2^2}, \frac{2k_2}{1+k_1^2+k_2^2}, \frac{1-k_1^2-k_2^2}{1+k_1^2+k_2^2}\right] \tag{3.84}$$

Similarly, a square (side=2) circumscribing the mapped disk area is used as the vertical direction domain in the BnB algorithm. It is worth noting that the

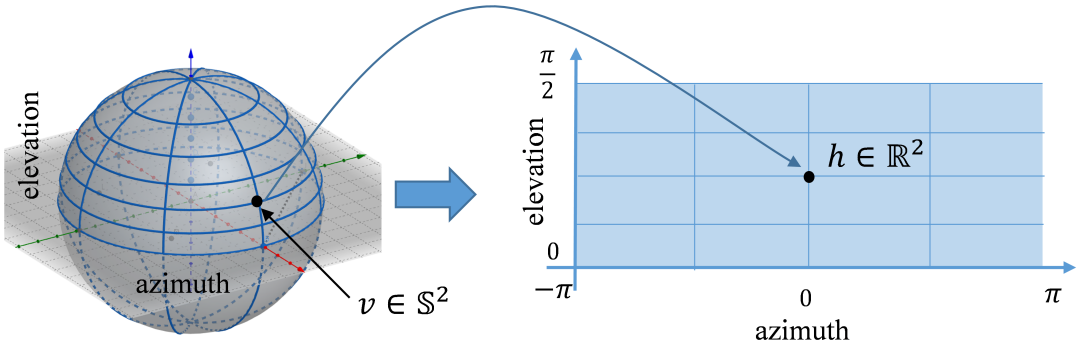


Figure 3.3: Visualization of the spherical coordinate system. The hemisphere is flattened to a rectangle, which leads to significant distortion, especially near the pole.

stereographic projection was also applied to accelerate the calculation in rotation search [136], which inspires our work.

3.4.3.4 Parametrization of $\mathbb{S}^{2+}$: Spherical Coordinate System

The $\mathbb{S}^{2+}$ can also be parameterized by the spherical coordinate system (Wikipedia: "spherical coordinate system"). Geometrically, the hemisphere is flattened to a rectangle (see Fig. 3.3). In the BnB algorithm, the rectangle region can be set as the initial search domain. For $\forall \boldsymbol{v} = [v_1, v_2.v_3]^T \in \mathbb{S}^{2+}$ and its corresponding point $\boldsymbol{h} = [h_1, h_2]^T$, the mapping from the three-dimensional Cartesian coordinate system to the spherical coordinate system is[1]

$$\boldsymbol{h}^T = \left[\arctan(v_2, v_1), \arctan\left(v_3, \sqrt{v_1^2 + v_2^2}\right)\right] \tag{3.85}$$

where $\arctan(\cdot,\cdot)$ is the four-quadrant inverse tangent function[2]. Conversely[3]

$$\boldsymbol{v}^T = [\cos(h_2)\cos(h_1), \cos(h_2)\sin(h_1), \sin(h_2)] \tag{3.86}$$

where $-\pi \leq h_1 \leq \pi$ and $0 \leq h_2 \leq \pi/2$ are azimuth angle and elevation angle, respectively.

In summary, the space of $\mathbb{SO}(3)$ is parameterized and relaxed to a 3D cube. By contrast, the parameterization of the hemisphere $\mathbb{S}^{2+}$ are a 2D disk or rectangle. Accordingly, in the BnB algorithm, the 3D cube is recursively subdivided into eight sub-cubes, however, the 2D disk or rectangle is recursively subdivide it into four smaller squares or rectangles. For ease of understanding, we call the point in the 2D solid disk or rectangle *image point*, meanwhile we call its corresponding point *pre-image point* in $\mathbb{S}^{2+}$.

[1] `https://www.mathworks.com/help/matlab/ref/cart2sph.html`
[2] `https://www.mathworks.com/help/matlab/ref/atan2.html`
[3] `https://www.mathworks.com/help/matlab/ref/sph2cart.html`

3.4.4 Estimating Bounds

In this section, we show how to calculate the bounds with different parameterizations in detail.

3.4.4.1 Bounds of Rotation Search

We first recall the bounds applied in rotation search. According to lemma 2.5,

$$\angle(\mathbf{R}_a\boldsymbol{x}, \mathbf{R}_b\boldsymbol{x}) \leq \|\boldsymbol{r}_a - \boldsymbol{r}_b\| \tag{3.87}$$

Then, we have the following Lemma.

Lemma 3.1 (rotation uncertainty angle bound)**.** *Given a cube-shaped rotation branch $\mathbb{B}^{rot}$, whose center is $\boldsymbol{c}^{rot}$, half-side is σ^{rot}. For $\forall \mathbf{R} \in \mathbb{B}^{rot}, \forall \boldsymbol{v}_0 \in \mathbb{S}^2$,*

$$\angle\big(\mathbf{R}\boldsymbol{v}_0, \mathbf{R}_c\boldsymbol{v}_0\big) \leq \sqrt{3}\sigma^{rot} \triangleq \psi^{rot} \tag{3.88}$$

where $\mathbf{R}_c$ is the matrix representation of $\boldsymbol{c}^{rot}$. Let initial vertical direction $\boldsymbol{v}_0 = [0, 0, 1]^T$, $\mathbf{R}\boldsymbol{v}_0 = \boldsymbol{v}^{rot}$ and $\mathbf{R}_c\boldsymbol{v}_0 \triangleq \boldsymbol{v}_c^{rot}$. Then,

$$\angle(\mathbf{R}\boldsymbol{v}_0, \mathbf{R}_c\boldsymbol{v}_0) \leq \psi^{rot} \Rightarrow \angle(\boldsymbol{v}^{rot}, \boldsymbol{v}_c^{rot}) \leq \psi^{rot} \tag{3.89}$$

Observe that it satisfies the conditions of Proposition 3.2: $\boldsymbol{v}^{rot} \to \boldsymbol{v}$, $\boldsymbol{v}_c^{rot} \to \boldsymbol{v}_c$, $\psi^{rot} \to \psi$ and $\{\mathbf{R}\boldsymbol{v}_0 | \mathbf{R} \in \mathbb{B}^{rot}\} \to \mathbb{B}$.

Then, given a divided cube-shaped rotation branch $\mathbb{B}^{rot}$, the bounds can be

$$U^{rot} = \sum_{j=1}^{N} \mathbb{I}\Big(\overline{Q}_j^{\parallel}\big(\boldsymbol{v}_c^{rot}, \psi^{rot}\big) \vee \overline{Q}_j^{\perp}\big(\boldsymbol{v}_c^{rot}, \psi^{rot}\big)\Big) \tag{3.90}$$

$$L^{rot} = \sum_{j=1}^{N} \mathbb{I}\Big(\underline{Q}_j^{\parallel}\big(\boldsymbol{v}_c^{rot}\big) \vee \underline{Q}_j^{\perp}\big(\boldsymbol{v}_c^{rot}\big)\Big) \tag{3.91}$$

Note that similar bounds are widely used in many geometrical vision problems [93, 115], which are not our original contributions. Besides, it is worth noting that there seems to be a tighter bound than Eq. (3.88) in [52], however, to calculate the bound efficiently, it is based on two unproven assumptions.

3.4.4.2 Bounds using Exponential Mapping

According to Proposition 3.3, we have,

Proposition 3.4. *Given a square-shaped branch $\mathbb{B}^{exp}$ in exponential mapping plane, whose center is $\boldsymbol{d}_c^{exp}$, and half-side is σ^{exp}. For $\forall \boldsymbol{d} \in \mathbb{B}^{exp}$,*

$$\angle(\boldsymbol{v}, \boldsymbol{v}_c^{exp}) \leq \sqrt{2}\sigma^{exp} \triangleq \psi^{exp} \tag{3.92}$$

where $\boldsymbol{v}_c^{exp}, \boldsymbol{v} \in \mathbb{S}^2$ are *pre-image points* of $\boldsymbol{d}_c^{exp}$ and $\boldsymbol{d}$.

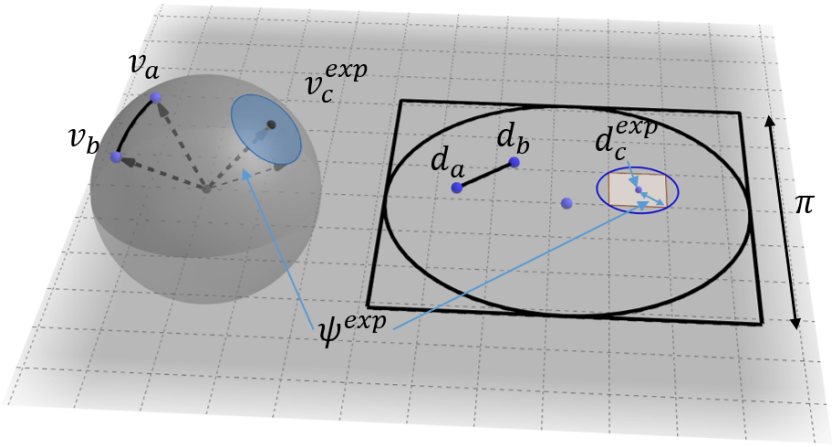


Figure 3.4: Visualization of exponential mapping. Two points $\boldsymbol{v}_a, \boldsymbol{v}_b \in \mathbb{S}^{2+}$ correspond to two points $\boldsymbol{d}_a, \boldsymbol{d}_b \in \mathbb{R}^2$ and $\angle(\boldsymbol{v}_a, \boldsymbol{v}_b) \leq \|\boldsymbol{d}_a - \boldsymbol{d}_b\|$. A square-shaped branch, whose center is $\boldsymbol{d}_c^{exp}$, is relaxed into a circle in $\mathbb{R}^2$. Then the pre-image of the circle is relaxed into a spherical patch, whose center is $\boldsymbol{v}_c^{exp}$, in $\mathbb{S}^2$. ψ^{exp} is the radius of the relaxed circle in the 2D plane.

Proof. This proposition can be derived as follows:

$$\angle(\boldsymbol{v}, \boldsymbol{v}_c^{exp}) \leq \|\boldsymbol{d} - \boldsymbol{d}_c^{exp}\| \leq \sqrt{2}\sigma^{exp} \tag{3.93}$$

which follows Proposition 3.3 (see Fig. 3.4). □

Proposition 3.4 and lemma 3.1 have similar formulations. However, to the best of our knowledge, it is the first time Proposition 3.4 has been explicitly introduced to the computer vision field. Obviously, given square-shaped branch $\mathbb{B}^{exp}$ in the exponential mapping plane, according to Proposition 3.2, the bounds can be:

$$U_q^{exp} = \sum_{j=1}^{N} \mathbb{I}\left(\overline{Q}_j^{\parallel}\left(\boldsymbol{v}_c^{exp}, \psi^{exp}\right) \vee \overline{Q}_j^{\perp}\left(\boldsymbol{v}_c^{exp}, \psi^{exp}\right)\right) \tag{3.94}$$

$$L_q^{exp} = \sum_{j=1}^{N} \mathbb{I}\left(\underline{Q}_j^{\parallel}\left(\boldsymbol{v}_c^{exp}\right) \vee \underline{Q}_j^{\perp}\left(\boldsymbol{v}_c^{exp}\right)\right) \tag{3.95}$$

3.4.4.3 Visual Interpretation of the Upper Bound using Exponential Mapping

In the proposed exponential mapping method, the hemisphere is flatten to a solid disk, which is then relaxed to its circumscribed square. In BnB algorithm, the upper bound for a sub-branch is constructed by geometrical relaxation. In this part, we explore the relaxation geometrically. Specifically, we draw relaxation region and the original sub-branch in the sphere and the solid disk, see Fig. 3.5. Evidently, in the exponential mapping, the square-shaped branch will correspond

to an irregular shape in the sphere (bounded by green line). The relaxed circle-region for sub-branch in the disk is also corresponding to an irregular shape in the sphere(bounded by red line). However, using lemma 2.6, the irregular shape can be bounded by a sphere patch (blue circle bounded umbrella-shaped patch).

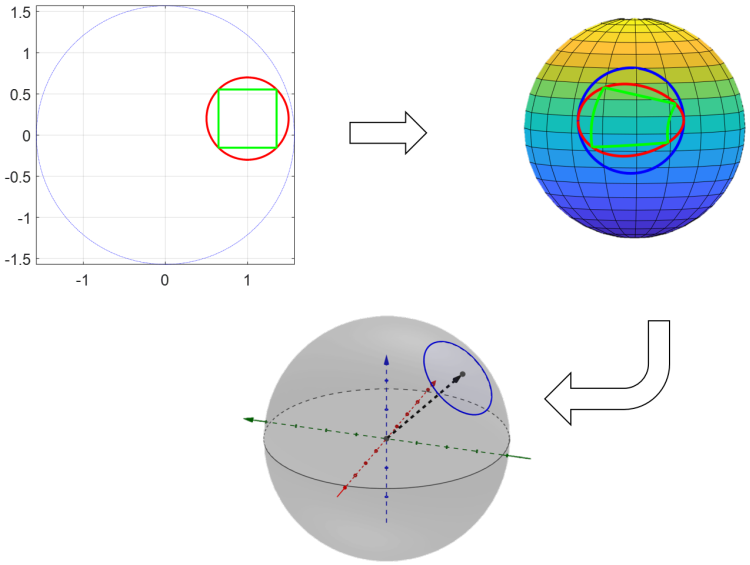


Figure 3.5: Geometric interpretation of the upper bound using exponential mapping

One may be curious about cases that the sub branch is across the boundary (equator), or the sub-branch is out of the hemisphere and is in the opposite hemisphere. We then draw the cases in Fig. 3.6.

From the figures, we can find that when the branch is far away from the original point, it will deform drastically, especially when the distance is greater than $\frac{\pi}{2}$ (out of hemisphere). It means the relaxation is becoming loose. Fortunately, hemisphere is enough for our proposed globally optimal solutions. Evidently, in a hemisphere, the relaxation is quit tight.

3.4.4.4 Bounds using Stereographic Projection

Stereographic projection has a crucial property: circles are projected as circles (circle preserving [136, 145]). We can use this property to calculate the bound based on stereographic projection.

Proposition 3.5. *Given a square-shaped branch $\mathbb{B}^{ste}$ in stereographic projection plane, and its circumscribed circle is $\mathbb{C}^{ste}_{2D}$. The pre-image of $\mathbb{C}^{ste}_{2D}$ is $\mathbb{C}^{ste}$ in $\mathbb{S}^2$, whose radius is σ^{ste} and the direction of its center point is $\boldsymbol{v}^{ste}_c$; $\forall \boldsymbol{k} \in \mathbb{B}^{ste}$, $\boldsymbol{v}$ is its pre-image point,*

$$\angle(\boldsymbol{v}, \boldsymbol{v}^{ste}_c) \leq \arcsin(\sigma^{ste}) \triangleq \psi^{ste} \tag{3.96}$$

Proof. Because $\boldsymbol{k} \in \mathbb{B}^{ste} \subset \mathbb{C}^{ste}_{2D}$, then its pre-image point $\boldsymbol{v} \in \mathbb{C}^{ste}$. The angle of $\boldsymbol{v}$ and $\boldsymbol{v}^{ste}_c$ must be no greater than the maximum angle ψ^{ste} (see Fig. 3.7). □

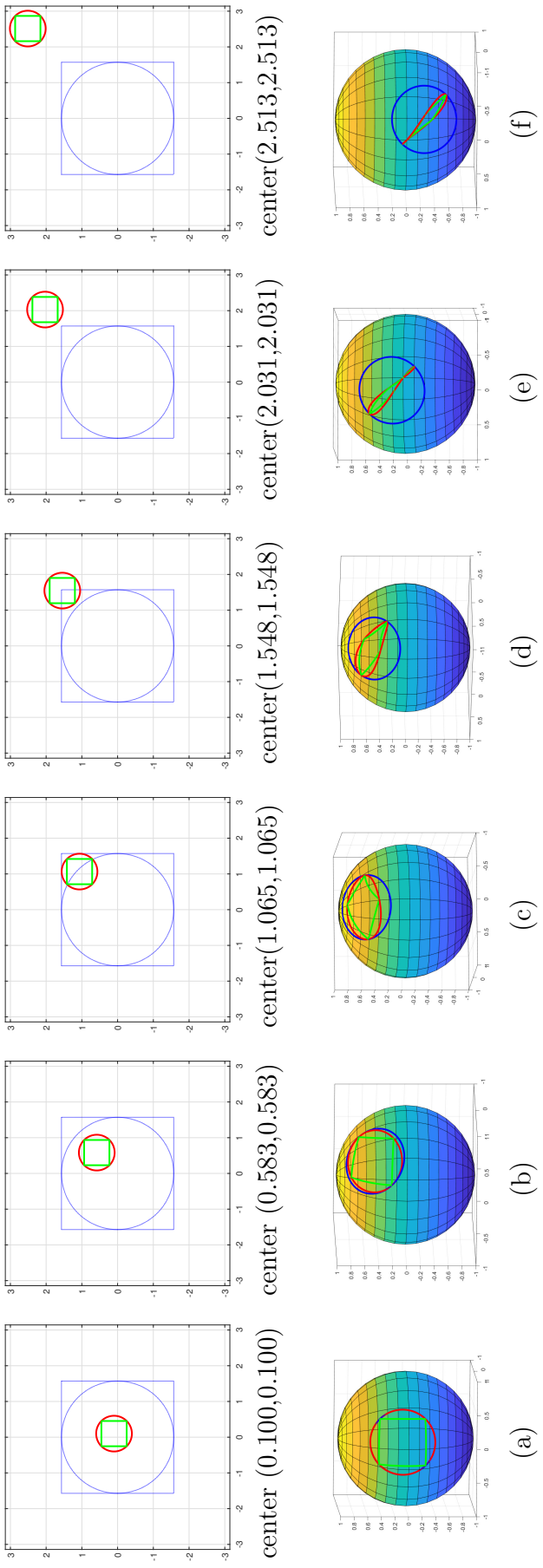


Figure 3.6: Different relaxations in different positions. In the first row, the blue circle bounded solid disk, which is mapped from the hemisphere and its circumscribed square is the initial searching domain. The green square means a sub-branch, which is relaxed to a region bounded by its circumscribed circle. Then according to our introduced lemma 2.6 in chapter 2, it is relaxed to a sphere patch in the sphere in the bottom row. The center coordinates of sub-branch are shown in middle row. Obviously, when the sub branch is far away the original point, the distortion is severe.

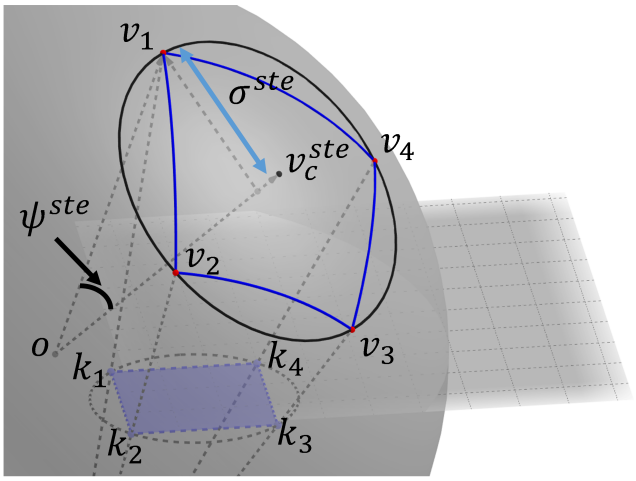


Figure 3.7: The geometry of a square-shaped branch in stereographic projection plane. A square-shaped branch $(\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3, \boldsymbol{k}_4)$ is projected to a domain $(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4)$ in $\mathbb{S}^2$. The radius of its circumscribed circle is σ^{ste}, and the direction of the center point is $\boldsymbol{v}_c^{ste}$, then $\psi^{ste} = \angle(\boldsymbol{v}_1, \boldsymbol{v}_c^{ste}) = \arcsin(\sigma^{ste})$.

We then explain how to calculate ψ^{ste} and $\boldsymbol{v}_c^{ste}$ in detail. Given a square-shaped branch $\mathbb{B}^{ste}$ in the 2D plane, its four vertexes $(\boldsymbol{k}_1, \boldsymbol{k}_2, \boldsymbol{k}_3, \boldsymbol{k}_4)$ must be in the circumscribed circle $\mathbb{C}_{2D}^{ste}$ (see Fig. 3.7). Then the pre-image points $(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4)$ of the vertexes must be in $\mathbb{C}^{ste}$. The direction of the center point $\boldsymbol{v}_c^{ste}$ is perpendicular to the plane crossing the circle $\mathbb{C}^{ste}$. Hence, $\boldsymbol{v}_c^{ste}$ is perpendicular to any vector in the circle-plane, which means $\boldsymbol{v}_c^{ste} \perp (\boldsymbol{v}_1 - \boldsymbol{v}_2)$ and $\boldsymbol{v}_c^{ste} \perp (\boldsymbol{v}_1 - \boldsymbol{v}_3)$. Let $\boldsymbol{v}_{cross} \triangleq (\boldsymbol{v}_1 - \boldsymbol{v}_2) \times (\boldsymbol{v}_1 - \boldsymbol{v}_3)$. Then, $\boldsymbol{v}_c^{ste} = \boldsymbol{v}_{cross}/\|\boldsymbol{v}_{cross}\|$ and $\psi^{ste} = \angle(\boldsymbol{v}_c^{ste}, \boldsymbol{v}_1) = \arcsin(\sigma^{ste})$.

Intuitively, Proposition 3.5 shows that a square-shaped branch in the stereographic projection plane is relaxed to a solid disk; meanwhile the corresponding domain in the 3D sphere is also relaxed to an umbrella-shaped patch surrounded by a circle of radius σ^{ste}.

Given a square-shaped branch $\mathbb{B}^{ste}$ in stereographic projection plane, according to Proposition 3.2, the bounds can be

$$U_q^{ste} = \sum_{j=1}^{N} \mathbb{I}\left(\overline{Q}_j^{\parallel}(\boldsymbol{v}_c^{ste}, \psi^{ste}) \vee \overline{Q}_j^{\perp}(\boldsymbol{v}_c^{ste}, \psi^{ste})\right) \tag{3.97}$$

$$L_q^{ste} = \sum_{j=1}^{N} \mathbb{I}\left(\underline{Q}_j^{\parallel}(\boldsymbol{v}_c^{ste}) \vee \underline{Q}_j^{\perp}(\boldsymbol{v}_c^{ste})\right) \tag{3.98}$$

The bounds (Eq. (3.97) and Eq. (3.98)) are called circle bounds using stereographic projection as the square-shaped branch is relaxed to its circumscribed circle.

3.4.4.5 Tighter Bounds using Stereographic Projection

For the stereographic projection, a tighter bound can be found without relaxing the divided square, and therefore, it does not apply the circle-preserving property.

Given a square-shaped branch $\mathbb{B}^{ste}$ in stereographic projection plane, the pre-image of its center is $\boldsymbol{v}_t^{ste}$. $\boldsymbol{v} \in \mathbb{S}^{2+}$ is the pre-image point of $\boldsymbol{k} \in \mathbb{B}^{ste}$.

$$\underline{\phi}_j^{ste} \triangleq \min \angle(\boldsymbol{v}, \boldsymbol{n}_j) \tag{3.99}$$

$$\overline{\phi}_j^{ste} \triangleq \max \angle(\boldsymbol{v}, \boldsymbol{n}_j) \tag{3.100}$$

$$\phi_j^{ste} = \angle(\boldsymbol{v}_t^{ste}, \boldsymbol{n}_j) \tag{3.101}$$

Considering the Proposition 3.1, the bounds can be

$$U_s^{ste} = \sum_{j=1}^{N} \mathbb{I}\left(\overline{\mathcal{S}}_j^{\perp}\left(\underline{\phi}_j^{ste}, \overline{\phi}_j^{ste}\right) \vee \overline{\mathcal{S}}_j^{\|^+}\left(\underline{\phi}_j^{ste}\right) \vee \overline{\mathcal{S}}_j^{\|^-}\left(\overline{\phi}_j^{ste}\right)\right) \tag{3.102}$$

$$L_s^{ste} = \sum_{j=1}^{N} \mathbb{I}\left(\underline{\mathcal{S}}_j^{\perp}\left(\phi_j^{ste}\right) \vee \underline{\mathcal{S}}_j^{\|^+}\left(\phi_j^{ste}\right) \vee \underline{\mathcal{S}}_j^{\|^-}\left(\phi_j^{ste}\right)\right) \tag{3.103}$$

We then explain the implementation details to calculate $\underline{\phi}_j^{ste}$ and $\overline{\phi}_j^{ste}$. Geometrically, the pre-image of a line in the stereographic projection plane is a circle passing through the projection pole in $\mathbb{S}^2$ [145]. Hence an edge of the square corresponds to an arc of the circle (see Fig.3.9). Therefore, given a square-shaped branch $\mathbb{B}^{ste}$ in stereographic projection plane, its pre-image in $\mathbb{S}^2$ is a region surrounded by four arcs. For the j-th surface normal $\boldsymbol{n}_j$, there are three cases (see Fig. 3.8):

1. $\pm\boldsymbol{n}_j$ are not in the closed pre-image of the square.
2. $\boldsymbol{n}_j$ is in the closed pre-image of the square-shaped branch.
3. $-\boldsymbol{n}_j$ is in the closed preimage of the square-shaped branch.

In case (1), it is obvious that $\underline{\phi}_j^{ste}$ and $\overline{\phi}_j^{ste}$ must occur on the edge of $\mathbb{B}^{ste}$. Therefore, it is sufficient to calculate the angle range between the $\boldsymbol{n}_j$ and the four arcs. We first calculate the angle between $\boldsymbol{n}_j$ and an arc. Intuitively, the angle between $\boldsymbol{n}_j$ and an arc may be in a part of a periodic function. Interestingly, we can get the maximum and minimum angle analytically with stereographic projection. Let $\boldsymbol{n}_j = [a_j, b_j, c_j]^T$ and $\boldsymbol{v}$ be a point in the arc, then

$$\phi_j = \arccos(\boldsymbol{v}^T \boldsymbol{n}_j) \tag{3.104}$$

where for $x \in [-1, 1]$, $\arccos(x)$ returns values in the interval $[0, \pi]$. To obtain the range of ϕ_j, it is sufficient to know the range of $\varpi \triangleq \boldsymbol{v}^T \boldsymbol{n}_j$.

$$\varpi = \frac{2k_1 \cdot a_j}{1 + k_1^2 + k_2^2} + \frac{2k_2 \cdot b_j}{1 + k_1^2 + k_2^2} + \frac{1 - k_1^2 - k_2^2}{1 + k_1^2 + k_2^2} \cdot c_j \tag{3.105}$$

$$= 2 \cdot \frac{k_1 \cdot a_j + k_2 \cdot b_j + c_j}{1 + k_1^2 + k_2^2} - c_j \tag{3.106}$$

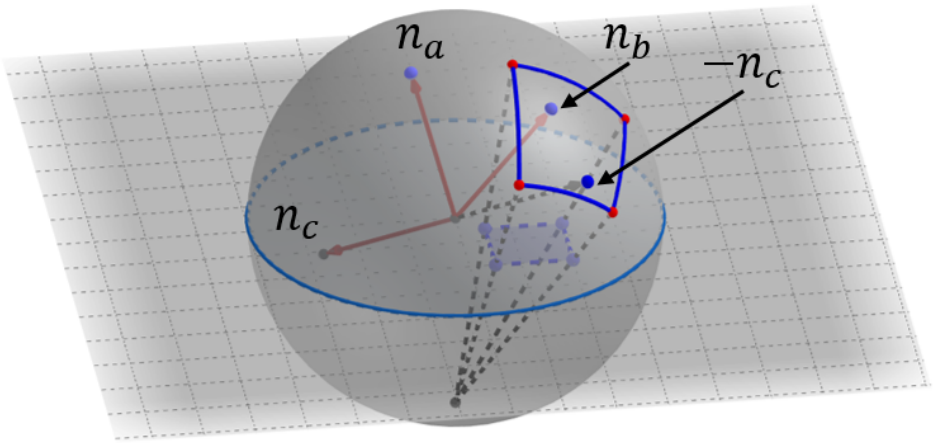


Figure 3.8: Three cases for estimating range $[\underline{\phi}_j^{ste}, \overline{\phi}_j^{ste}]$. (1) $\pm\boldsymbol{n}_a$ are not in the closed pre-image of the square-shaped branch, then $[\underline{\phi}_a^{ste}, \overline{\phi}_a^{ste}]$ depends on the boundary. (2) $\boldsymbol{n}_b$ is in the closed pre-image of the square-shaped branch, then $\underline{\phi}_b^{ste} = 0$. (3) $-\boldsymbol{n}_c$ is in the closed pre-image of the square-shaped branch, then $\overline{\phi}_c^{ste} = \pi$.

If we separate the branches along the coordinate axis in BnB, then it is a univariate function for each edge, and its derivatives are:

$$\varpi'_{k_1} = \frac{-2a_jk_1^2 - (4c_j + 4b_jk_2)k_1 + 2a_jk_2^2 + 2a_j}{(1 + k_1^2 + k_2^2)^2} \tag{3.107}$$

$$\varpi'_{k_2} = \frac{-2b_jk_2^2 - (4c_j + 4a_jk_1)k_2 + 2b_jk_1^2 + 2b_j}{(1 + k_1^2 + k_2^2)^2} \tag{3.108}$$

Let $\varpi'_{k_1} = 0$, we can get two optimal points, of which one corresponds to the maximum value and the other corresponds to the minimum value.

$$k_1^* = \frac{c_j + b_jk_2 \pm \sqrt{(c_j + b_jk_2)^2 + a_j(a_jk_2^2 + a_j)}}{-a_j} \tag{3.109}$$

Similarly, let $\varpi'_{k_2} = 0$, we can also get two optimal points:

$$k_2^* = \frac{c_j + a_jk_1 \pm \sqrt{(c_j + a_jk_1)^2 + b_j(b_jk_1^2 + b_j)}}{-b_j} \tag{3.110}$$

However, the optimal points may be outside of the arc because an edge of the square only corresponds to an arc in $\mathbb{S}^2$ not a whole circle. As a result, when the optimal points are outside the arc, the maximum and minimum value can be obtained at the endpoints of the arc. In other words, to calculate the angle range between $\boldsymbol{n}_j$ and arc, we need to calculate the optimal points in the whole

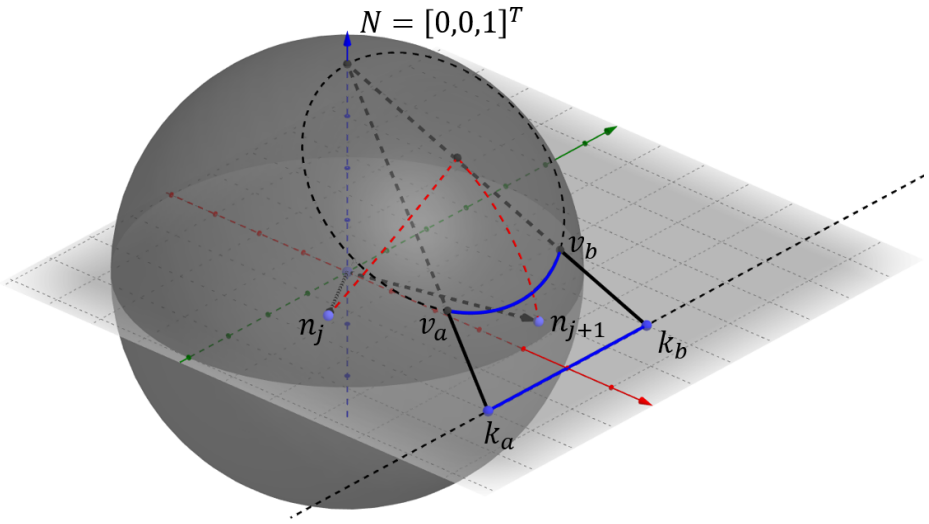


Figure 3.9: The geometry between an edge $(\boldsymbol{k}_a, \boldsymbol{k}_b)$ of a square-shaped branch and $\boldsymbol{n}_j, \boldsymbol{n}_{j+1}$ in stereographic projection. For better visualization, we set $N = [0, 0, 1]^T$ as the stereographic projection pole. For $\boldsymbol{n}_j$, the optimal points are not in the edge, then the optimal points are the endpoints ($\boldsymbol{k}_a$ and $\boldsymbol{k}_b$). In contrast, for $\boldsymbol{n}_{j+1}$, the optimal point that corresponds to the minimum angle is in the edge, but maximum angle occurs at one of the two endpoints ($\boldsymbol{k}_a$ or $\boldsymbol{k}_b$).

circle firstly. If the optimal points are not in the arc, then the angle range is determined by the endpoints of the arc. Otherwise, the angle range is depending on the optimal points.

For a square-shaped branch with four edges, $[\underline{\phi}_j^{ste}, \overline{\phi}_j^{ste}]$ can be obtained by calculating the angle range between $\boldsymbol{n}_j$ and four edges of the square one by one.

In case (2), since $\boldsymbol{n}_j$ is in the closed preimage of the divided square-shaped branch, then $\underline{\phi}_j^{ste} = 0$. However, $\overline{\phi}_j^{ste}$ still needs to be calculated in edges. Similarly, in case (3), $-\boldsymbol{n}_j$ is in the closed preimage of the square, then $\overline{\phi}_j^{ste} = \pi$, and $\underline{\phi}_j^{ste}$ is calculated in edges.

Note that the bounds are tighter than circle bounds using stereographic projection. The reason is obvious that a square-shaped branch is relaxed to a solid circle in the circle bounds but not relaxed in the tighter bounds. Additionally, because the bounds are based on the square, we call the tighter bounds square bounds using stereographic projection.

3.4.4.6 Bounds using the Sphere Coordinate System

In this part, we introduce the upper and lower bounds using the sphere coordinate system according to Proposition 3.1.

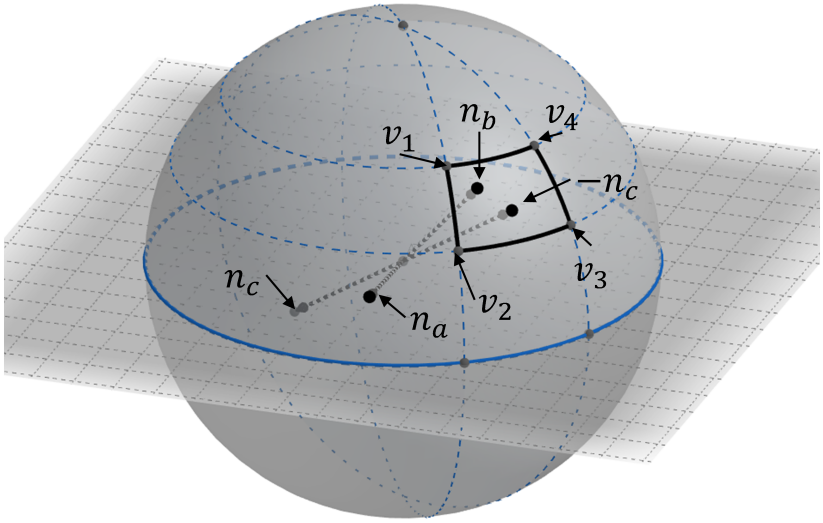


Figure 3.10: Estimating $[\underline{\phi}_j^{scs}, \overline{\phi}_j^{scs}]$ using the sphere coordinate system. (1) $\pm\boldsymbol{n}_a$ are not in the closed pre-image of the branch, then $[\underline{\phi}_a^{scs}, \overline{\phi}_a^{scs}]$ depends on the boundary. (2) $\boldsymbol{n}_b$ is in the closed pre-image of the branch, then $\underline{\phi}_b^{scs} = 0$. (3) $-\boldsymbol{n}_c$ is in the closed pre-image of the branch, then $\overline{\phi}_c^{scs} = \pi$.

Given a rectangle-shaped branch $\mathbb{B}^{scs}$ in the azimuth-elevation rectangle, the pre-image of its center is $\boldsymbol{v}_c^{scs}$. $\boldsymbol{v} \in \mathbb{S}^{2+}$ is the pre-image point of $\boldsymbol{h} \in \mathbb{B}^{scs}$.

$$\underline{\phi}_j^{scs} \triangleq \min \angle(\boldsymbol{v}, \boldsymbol{n}_j) \tag{3.111}$$

$$\overline{\phi}_j^{scs} \triangleq \max \angle(\boldsymbol{v}, \boldsymbol{n}_j) \tag{3.112}$$

$$\phi_j^{scs} = \angle(\boldsymbol{v}_c^{scs}, \boldsymbol{n}_j) \tag{3.113}$$

Then, the bounds can be

$$U_s^{scs} = \sum_{j=1}^{N} \mathbb{I}\left(\overline{\mathcal{S}}_j^{\perp}\left(\underline{\phi}_j^{scs}, \overline{\phi}_j^{scs}\right) \vee \overline{\mathcal{S}}_j^{\parallel^+}\left(\underline{\phi}_j^{scs}\right) \vee \overline{\mathcal{S}}_j^{\parallel^-}\left(\overline{\phi}_j^{scs}\right)\right) \tag{3.114}$$

$$L_s^{scs} = \sum_{j=1}^{N} \mathbb{I}\left(\underline{\mathcal{S}}_j^{\perp}\left(\phi_j^{scs}\right) \vee \underline{\mathcal{S}}_j^{\parallel^+}\left(\phi_j^{scs}\right) \vee \underline{\mathcal{S}}_j^{\parallel^-}\left(\phi_j^{scs}\right)\right) \tag{3.115}$$

where $\phi_j^{scs} = \angle(\boldsymbol{v}_c^{scs}, \boldsymbol{n}_j)$.

We then explain how to estimate the range $[\underline{\phi}_j^{scs}, \overline{\phi}_j^{scs}]$ in detail. Similarly, there are also three cases (see Fig. 3.10):

1. $\pm\boldsymbol{n}_j$ are not in the closed preimage of the divided rectangle-shaped branch.
2. $\boldsymbol{n}_j$ is in the closed preimage of the rectangle.
3. $-\boldsymbol{n}_j$ is in the closed preimage of the rectangle.

In case (1), it is understandable that $\underline{\phi}_j^{scs}$ and $\overline{\phi}_j^{scs}$ must occur on the edges. Given $\boldsymbol{n}_j = [a_j, b_j, c_j]^T$, and $\phi_j = \arccos(\boldsymbol{v}^T \boldsymbol{n}_j)$, then to estimate the range of ϕ_j, it is sufficient to estimate the range of $\varpi = \boldsymbol{v}^T \boldsymbol{n}_j$.

$$\varpi = \cos(h_2)\cos(h_1)a_j + \cos(h_2)\sin(h_1)b_j + \sin(h_2)c_j \tag{3.116}$$

If we separate the branches along the coordinate axis in BnB, then it is a univariate function for each edge, and its derivatives are:

$$\varpi'_{h_1} = \cos(h_2)\big(-a_j \sin(h_1) + b_j \cos(h_1)\big) \tag{3.117}$$

$$\varpi'_{h_2} = c_j \cos(h_2) - \sin(h_2)\big(a_j \cos(h_1) + b_j \sin(h_1)\big) \tag{3.118}$$

Let $\varpi'_{h_1} = \varpi'_{h_2} = 0$, we can obtain the optimal points (see Fig. 3.11).

$$h_1^* = \{\arctan(b_j, a_j), \arctan(-b_j, -a_j)\} \tag{3.119}$$

$$h_2^* = \arctan\big(c_j, a_j \cos(h_1) + b_j \sin(h_1)\big) \tag{3.120}$$

If the optimal points are inside the edge, then the maximum or minimum value can be obtained at the optimal points, otherwise, the optimal values are obtained at the endpoints of the edge. Finally, the range $[\underline{\phi}_j^{scs}, \overline{\phi}_j^{scs}]$ can be obtained by calculating the angle range between $\boldsymbol{n}_j$ and four edges of the square one by one.

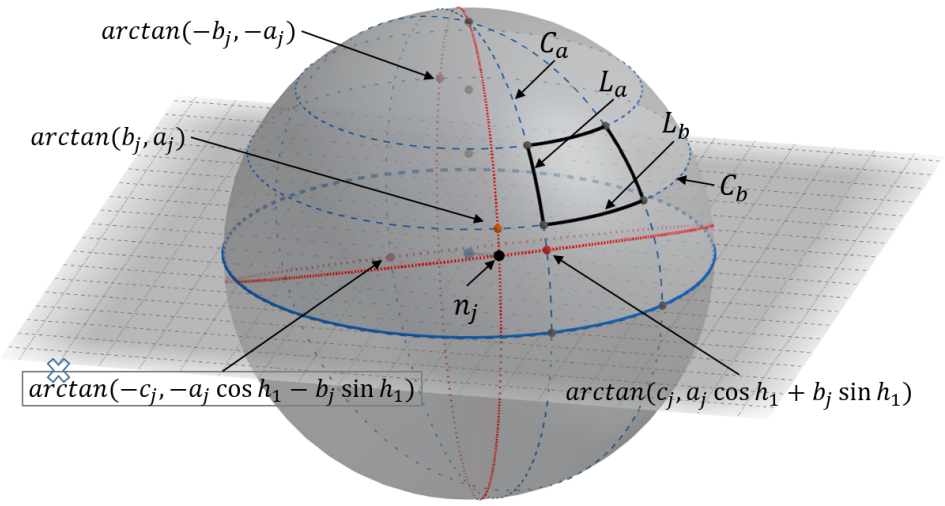


Figure 3.11: The geometry of calculating the optimal points in the sphere coordinate system. Given an edge L_a, which is a part of a circle C_a, all the points in C_a have the same azimuth h_1. Then for $\forall \boldsymbol{n}_j = [a_j, b_j, c_j]^T \in \mathbb{S}^{2+}$, the minimum angle between $\boldsymbol{n}_j$ and C_a must occur at $[h_1, h_2^*]^T$.Given an edge L_b, which is a part of a circle C_b, all the points in C_b have the same elevation h_2. Then for $\forall \boldsymbol{n}_j = [a_j, b_j, c_j]^T \in \mathbb{S}^{2+}$, the minimum and maximal angle between $\boldsymbol{n}_j$ and C_b must occur at $[\arctan(b_j, a_j), h_2]^T$ and $[\arctan(-b_j, -a_j), h_2]^T$, respectively.

In case (2), $\boldsymbol{n}_j$ is in the closed pre-image of the square-shaped branch, then $\underline{\phi}_j^{scs} = 0$. However, $\overline{\phi}_j^{scs}$ still needs to be calculated in edges. Similarly, in case (3),

Table 3.1: Different settings for different bounds in Algorithm 1.

Methods	Upper	Lower	Search domain
RS	U^{rot}	L^{rot}	3D cube (side=2π)
exp-BnB	U_q^{exp}	L_q^{exp}	2D square (side=π)
spc-BnB	U_q^{ste}	L_q^{ste}	2D square (side=2)
sps-BnB	U_s^{ste}	L_s^{ste}	2D square (side=2)
scs-BnB	U_s^{scs}	L_s^{scs}	2D rectangle ($2\pi \times \pi/2$)

$-\boldsymbol{n}_j$ is in the closed pre-image of the square, then $\overline{\phi}_j^{scs} = \pi$, and $\underline{\phi}_j^{scs}$ is calculated in edges.

3.4.4.7 Comparison of the Bounds

To show the relaxation and the tightness, in this section, we compare these bounds (Table 3.1) geometrically.

Bounds of rotation search. The search domain is parametrized as a 3D cube. In BnB, for each divided sub-cube, it is first relaxed to its circumscribed ball and then relaxed to a region in quaternion sphere (lemma 2.2). Lastly, it is relaxed to a spherical patch in $\mathbb{S}^2$ using Lemma 2.1.

Bounds using exponential mapping (*exp* bounds). The search domain is parametrized as a 2D square. The divided sub-square is first relaxed to its circumscribed circle and then relaxed to a spherical patch in $\mathbb{S}^2$ (Proposition 3.4). Therefore, it has a two-step geometrical relaxation.

Circle bounds using stereographic projection (*ste-cirlce* bounds). The search domain is parametrized as a 2D square. The divided sub-square is first relaxed to its circumscribed circle, which corresponds to a spherical patch in $\mathbb{S}^2$ (circle preserving). In geometric, it has only one-step relaxation.

Square bounds using stereographic projection (*ste-square* bounds). The search domain is the same as that of the ste-circle bounds, however, the ste-square bounds have no geometrical relaxations.

Bounds using the sphere coordinate system (*scs* bounds). The search domain is parametrized as a 2D azimuth-elevation rectangle, which leads to significant distortions. Nonetheless, they have no geometrical relaxations.

Note that what we say about geometrical relaxation is only for one specific input. There is another relaxation for the objective, which relaxes the connections among the inputs. In other words, for a large branch, it hardly obtains the upper bound simultaneously for all inputs.

Computational efficiency. The *exp* bounds and the *ste-circle* bounds are calculated more efficiently than the *ste-square* bounds and the *scs* bounds. This is because to estimate $\overline{\phi}_j$ and $\underline{\phi}_j$, it is needed to calculate the angle range between the $\boldsymbol{n}_j$ and four edges of the branch in the *ste-square* bounds and the *scs* bounds. However, given a branch $\mathbb{B}$, all $\{\boldsymbol{n}_j\}_{j=1}^N$ share the same ψ in the *exp* bounds and the *ste-circle* bounds.

Although there have different bounds, branching and pruning processes are repeated similarly until the difference between the upper and lower limits is less than 1, then the algorithm terminates.

3.5 Experiments

In this section, we verify the validity of the proposed method on challenging synthetic and real-world data. Firstly, we compared our proposed methods with RANSAC and the rotation search method to demonstrate robustness and efficiency. Then, full Atlanta frame estimation experiments were conducted to verify that estimating the vertical direction was helpful for estimating all Atlanta frames. Lastly, we tested the proposed methods in two real-world datasets to verify the practicality. All methods were implemented in Matlab 2019a and executed on an AMD Ryzen 7 2700X 3.7GHz CPU.

3.5.1 Experimental Setting

The settings of approaches/pipelines run on experiments were as follows:

- **RANSAC**: The number of minimal sample subsets was 2. It could get three directions from two inlier-inputs (two inlier directions and its cross product direction), and one of them might be the vertical direction. Besides, the confidence level $\zeta = 0.99$ was used for the stopping criterion [34]. The number of iterations was typically taken as

$$\left\lceil \frac{\log(1-\zeta)}{\log(1-(1-\rho)^2)} \right\rceil \tag{3.121}$$

  where ρ was the outlier proportion, $\lceil \cdot \rceil$ returned the nearest integer greater than or equal to the input.

- **RS**: Algorithm 1 with the rotation search bounds. Note that the bounds were also used in meta-BnB in [115]. We did not use the Extended Gaussian Image (EGI) and its integral image [93, 115], because we focused on the geometry and the validity of the proposed bounds. There may be more efficient bounds calculation methods for the proposed bounds but it is outside of the scope of this paper.

- **exp-BnB**: Algorithm 1 with the proposed bounds using the exponential mapping.

- **spc-BnB**: Algorithm 1 with the proposed circle bounds using the Stereographic Projection.

- **sps-BnB**: Algorithm 1 with the proposed square bounds using the Stereographic Projection.

- **scs-BnB**: Algorithm 1 with the proposed bounds using the Sphere Coordinate System.

In addition, to simulate the corrupted inputs in the synthetic experiments, noise and outliers were added. For noise, $\boldsymbol{e}_j \in \mathbb{R}^3$ was the j-th random vector, whose elements were randomly and uniformly distributed in the interval $[-1, 1]$. The noise was simulated by

$$\boldsymbol{n}_j \longleftarrow \frac{\boldsymbol{n}_j + \kappa \boldsymbol{e}_j}{\|\boldsymbol{n}_j + \kappa \boldsymbol{e}_j\|} \tag{3.122}$$

where κ was the amplitude of noise. For outliers, random orientations were added into the inputs. The total number of inputs was denoted N and the number of outlier inputs was denoted N_o, whereupon $\rho = N_o/N$ was the outlier proportion.

3.5.2 Synthetic Data Experiments

3.5.2.1 Synthetic Atlanta World

To simulate synthetic Atlanta world data, a random orientation was generated as the vertical direction ($\boldsymbol{v}_{gt}$). Except where otherwise specified, 20% inlier inputs were parallel to the vertical direction, and the other 80% inlier inputs were randomly generated to be perpendicular to the vertical direction and thus in the "horizontal plane". Note that the number of the horizontal frames were not specified. The inlier threshold was $\epsilon = \arctan(\kappa)$ according to the noise level in all the synthetic experiments. Once the vertical direction was estimated as $\boldsymbol{v}^*$, the error was calculated by

$$\arccos(abs(\boldsymbol{v}_{gt}^T \boldsymbol{v}^*)) \tag{3.123}$$

To evaluate the results of the experiment, the error and runtime were recorded. Additionally, the iterations of BnB algorithm with different bounds were also recorded. Moreover, to reduce the randomness, 500 trials were repeated in each setting.

Controlled experiments. We first tested all the methods with different outlier ratios $\rho = \{0.1, \cdots, 0.6\}$ and different noise levels $\kappa = \{0.005, 0.010, 0.020\}$. The number of input was set $N = 500$. The results are shown in Fig. 3.12. From the results, we can draw the following conclusions:

- All the four types of bounds in $\mathbb{S}^2$ and the bounds of rotation search could be nested into the BnB algorithm to estimate the vertical direction globally in Atlanta world.

- The BnB algorithms with different bounds in $\mathbb{S}^2$ had different efficiencies. Nonetheless, searching in $\mathbb{S}^2$ was more efficient than searching in $\mathbb{SO}(3)$.

- Broadly, the exp-BnB and the spc-BnB had similar efficiency. The sps-BnB and the scs-BnB had similar efficiency. More specifically, the first two were more efficient than the last two.

- Generally, the sps-BnB and the scs-BnB had fewer iterations than the exp-BnB and the spc-BnB. It revealed that the sps bounds and the scs bounds were tighter, which was consistent with the previous theoretical analysis.

There are three main reasons why the rotation search method is rather inefficient for vertical direction estimation.

1. Multiple solutions. Since $\mathbf{R}\boldsymbol{v}_0 = \boldsymbol{v}^*$, if the initial direction $\boldsymbol{v}_0$ and the optimal vertical direction $\boldsymbol{v}^*$ are fixed, there are numerous solutions for $\mathbf{R}$ [146]. Specifically, if $\mathbf{R}\boldsymbol{v}_0 = \boldsymbol{v}^*$ holds, then $\mathbf{R}_v\mathbf{R}\boldsymbol{v}_0 = \mathbf{R}_v\boldsymbol{v}^* = \boldsymbol{v}^*$ holds, where $\mathbf{R}_v$ is an arbitrary rotation about axis $\boldsymbol{v}^*$. Therefore, all possible $\mathbf{R}_v\mathbf{R}$ are solutions. For the BnB algorithm, if there are multiple solutions, there are many near-optimal branches, then the BnB algorithm must spend much time pruning the branches.

2. Higher dimensionality. Since the vertical direction is inherently in two dimension, searching in higher dimension leads to lower efficiency.

3. Conservative bound. Since rotation search bounds have a three-step geometrical relaxation, the bounds are relatively conservative.

Furthermore, why did exp-BnB and spc-BnB algorithms run faster despite having more iterations? We think this was because on one hand, tighter bounds would remove branches more aggressively and yield fewer iterations. However, on the other hand, using tighter bounds in BnB might be counter-productive if calculating the bound itself took significant time.

Challenging experiments. We conducted more experiments on challenging data. In this part, we only tested the bounds in $\mathbb{S}^2$, as the bounds of rotation search were obviously less efficient. The number of inputs was fixed at $N = 500$. First, all methods were tested at different high outlier ratios $\rho = \{0.65, \cdots, 0.9\}$ and different noise levels $\kappa = \{0.005, 0.010, 0.020\}$. The results are shown in Fig. 3.13. Second, all methods were tested at different large noise levels $\kappa = \{0.050, 0.100, 0.200\}$ and different outlier ratios $\rho = \{0.1, \cdots, 0.6\}$. The results are shown in Fig. 3.14. From all the results, we can draw the following conclusions:

- The exp-BnB had the highest efficiency among all BnB-based methods in such experimental settings. It is worth noting that in large outlier ratio cases ($\rho \geq 0.8$), the exp-BnB algorithm even had comparable efficiency with RANSAC.

- The sps-BnB had the least iterations among all BnB-based methods, which showed the bounds were very tight.

Theoretically, both ste-square bounds and *scs* bounds have no geometrical relaxations, then why did *scs* bounds need more iterations than *ste-square* bounds in the challenging experiments? We think this was due to the large distortion

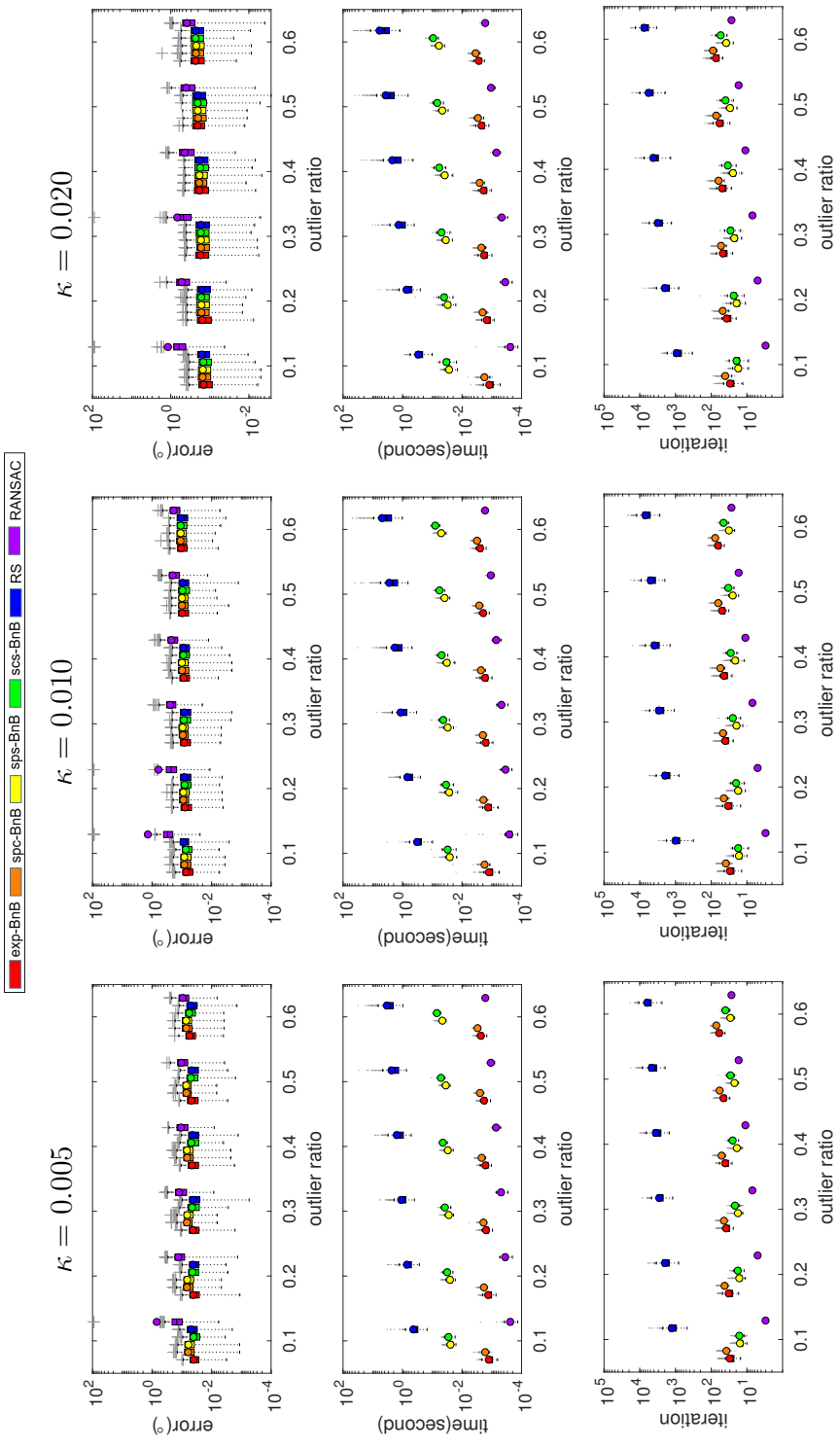


Figure 3.12: Controlled experiments. The first row shows the vertical direction error (°) at different noise levels. The second row shows the runtime (second) at different noise levels. The third row shows the iteration count at different noise levels.

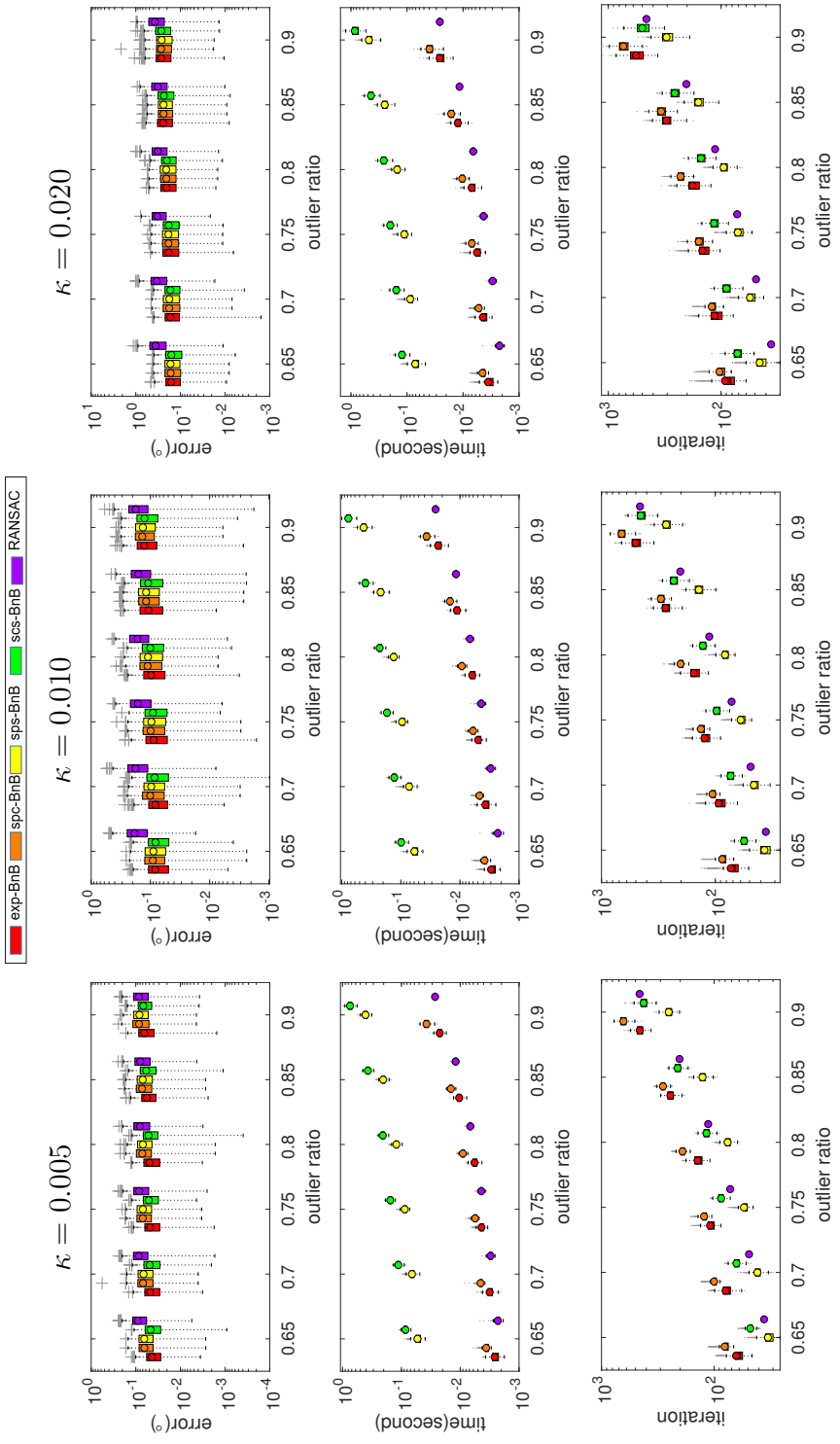


Figure 3.13: High outlier ratio experiments.The first row shows the vertical direction error (°) at different noise levels. The second row shows the runtime (second) at different noise levels. The third row shows the iteration count at different noise levels.

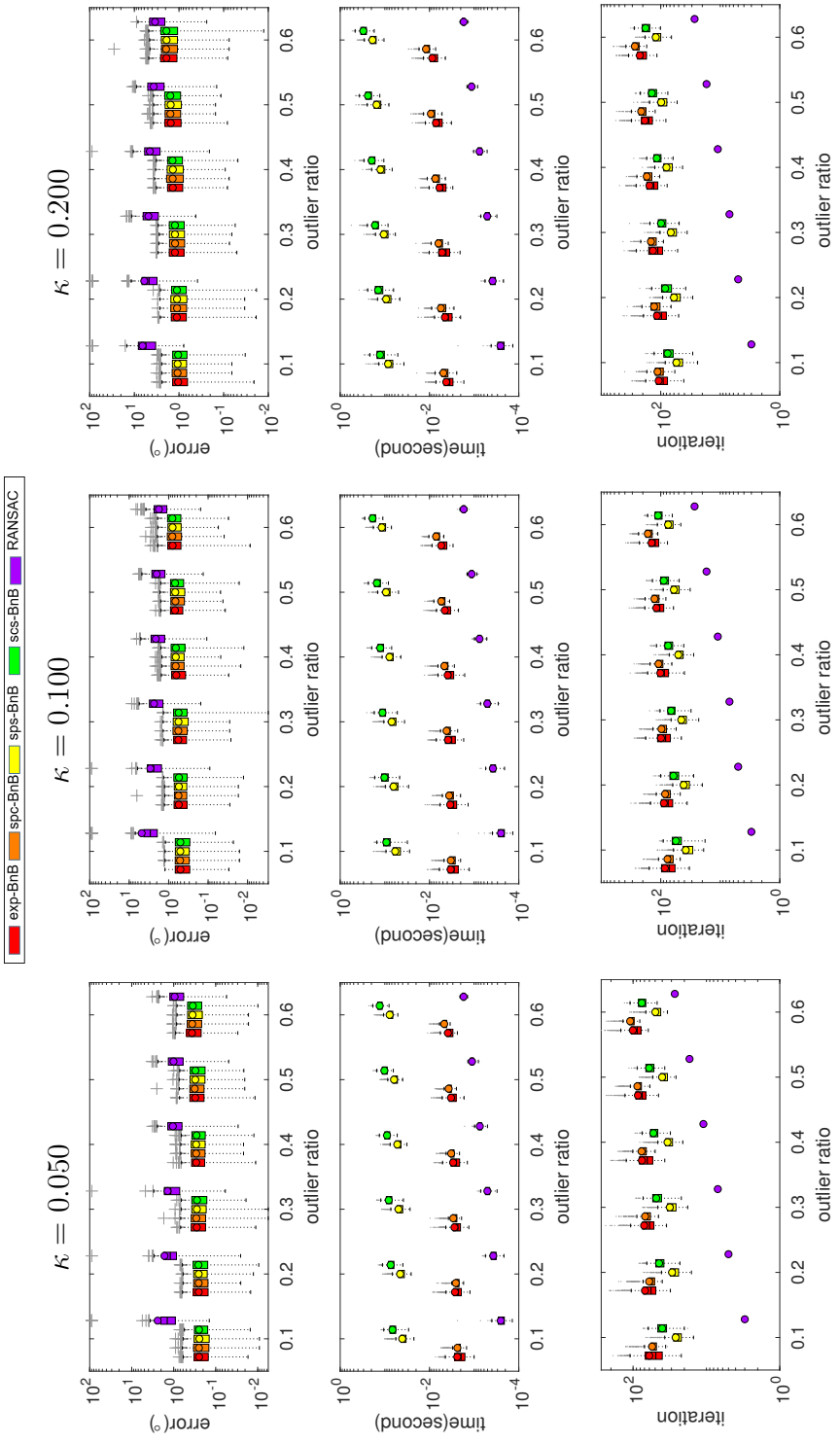


Figure 3.14: Large noise experiments. The first row shows the vertical direction error (°) at different noise levels. The second row shows the runtime (second) at different noise levels. The third row shows the iteration count at different noise levels.

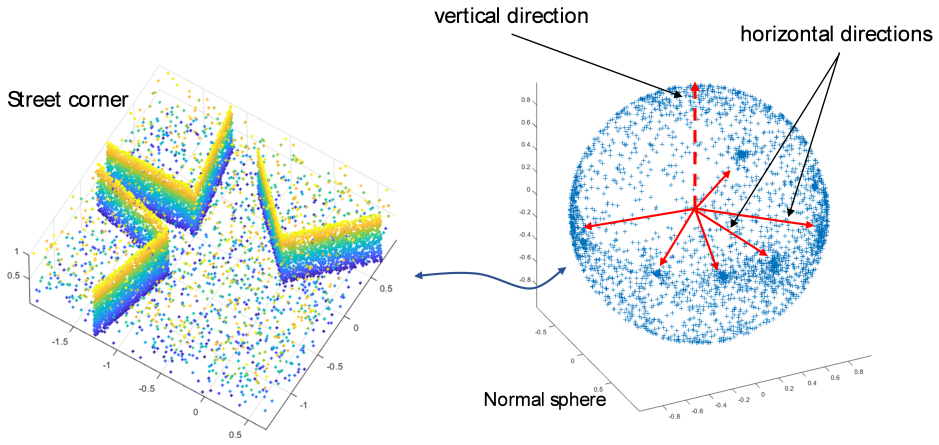


Figure 3.15: A synthetic street corner in Atlanta world, which contains six walls and no ground plane. The task is estimating the Atlanta frames from the surface normals.

of the search domain. For example, the region near the optimal direction in $\mathbb{S}^2$ might be expanded to a scale-up region in azimuth-elevation rectangle, therefore, the BnB algorithm needed more iterations to prune the near-optimal branches.

3.5.2.2 Full Atlanta Frame Estimation

In this part, we verified the performance of our proposed bounds in the full Atlanta frame estimation problem. For the sake of fairness, the experiments were conducted on the synthetic Manhattan world and the rotation search method was from [103] without EGI acceleration [93]. Our proposed methods first estimated the vertical frame direction, and then estimated the horizontal frames by a one-dimension clustering method, which we called sequential methods.

We then introduce how to sequentially estimate full Atlanta frame estimation (see Fig. 3.15). First, the vertical direction can be estimated by our proposed vertical estimation methods. After obtaining the vertical direction, the Atlanta world frame estimation problem becomes very easy, because all horizontal directions are all in the horizontal plane whose normal is parallel to the vertical direction. Therefore, estimating the horizontal directions with known vertical directions is a one-dimensional multi-model clustering or fitting problem. There have been lots of methods could solve the problem(e.g., Gaussian Mixture Models, k-Means Clustering, Hierarchical Clustering). Nevertheless, we introduce an easy-to-use method to estimate the horizontal directions in this part.

Firstly, according to the Atlanta world assumption, given the vertical directions ($\boldsymbol{v}_v$), the j-th horizontal frame ($\boldsymbol{v}_{h_j}$) must satisfy the following inequation:

$$|\angle(\boldsymbol{v}_v, \boldsymbol{v}_{h_j}) - \pi/2| \leq \epsilon \tag{3.124}$$

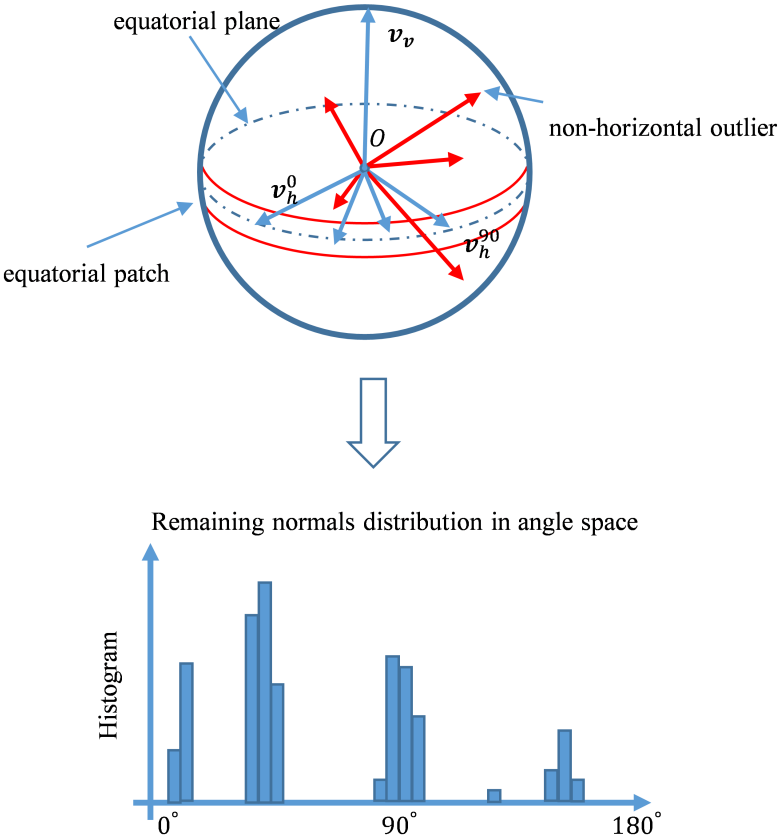


Figure 3.16: One-dimensional clustering method. After filtering the outliers, the remaining normals are parameterized by angles, then histogram is appplied to obtain the horizontal directions.

where ϵ is the inlier threshold. It gives a rule to reduce the outliers significantly and reduces the dimensionality of problem to one. Note that we cannot remove all the outliers in this step.

After filtering the non-horizontal outliers, the remaining normals all lie in equatorial patch of the normal-sphere as Fig. 3.16. We then use the angles to represent the normals. Given the vertical direction $\boldsymbol{v}_v$, we first define the 0-degree direction $\boldsymbol{v}_h^0$ and 90-degree direction $\boldsymbol{v}_h^{90}$. They are an orthonormal basis[4] for the null space of $\boldsymbol{v}_v^T$. By calculating the angle disctance between the j-th remaining normal $\boldsymbol{n}_j$ and $\{\boldsymbol{v}_h^0, \boldsymbol{v}_h^{90}\}$, we can represent $\boldsymbol{n}_j$ by an angle θ_j. Note that since $\boldsymbol{n}_j$ and $-\boldsymbol{n}_j$ may represent the same structral world frame, we then set $\theta_j \in [0°, 180°]$.

Next, the task is estimating the clustering centers of $\{\theta_j\}_{j=1}^M$, where M is the number of remaining normals. The clustering centers are corresponding to the directions of the horizontal frames. Specifically, we apply a Matlab built-in function *histcounts*[5] to obtain the angle-histogram bin counts. Then, the clustering centers can be estimated by finding the peaks of the histogram.

[4]https://ww2.mathworks.cn/help/matlab/ref/null.html
[5]https://ww2.mathworks.cn/help/matlab/ref/histcounts.html

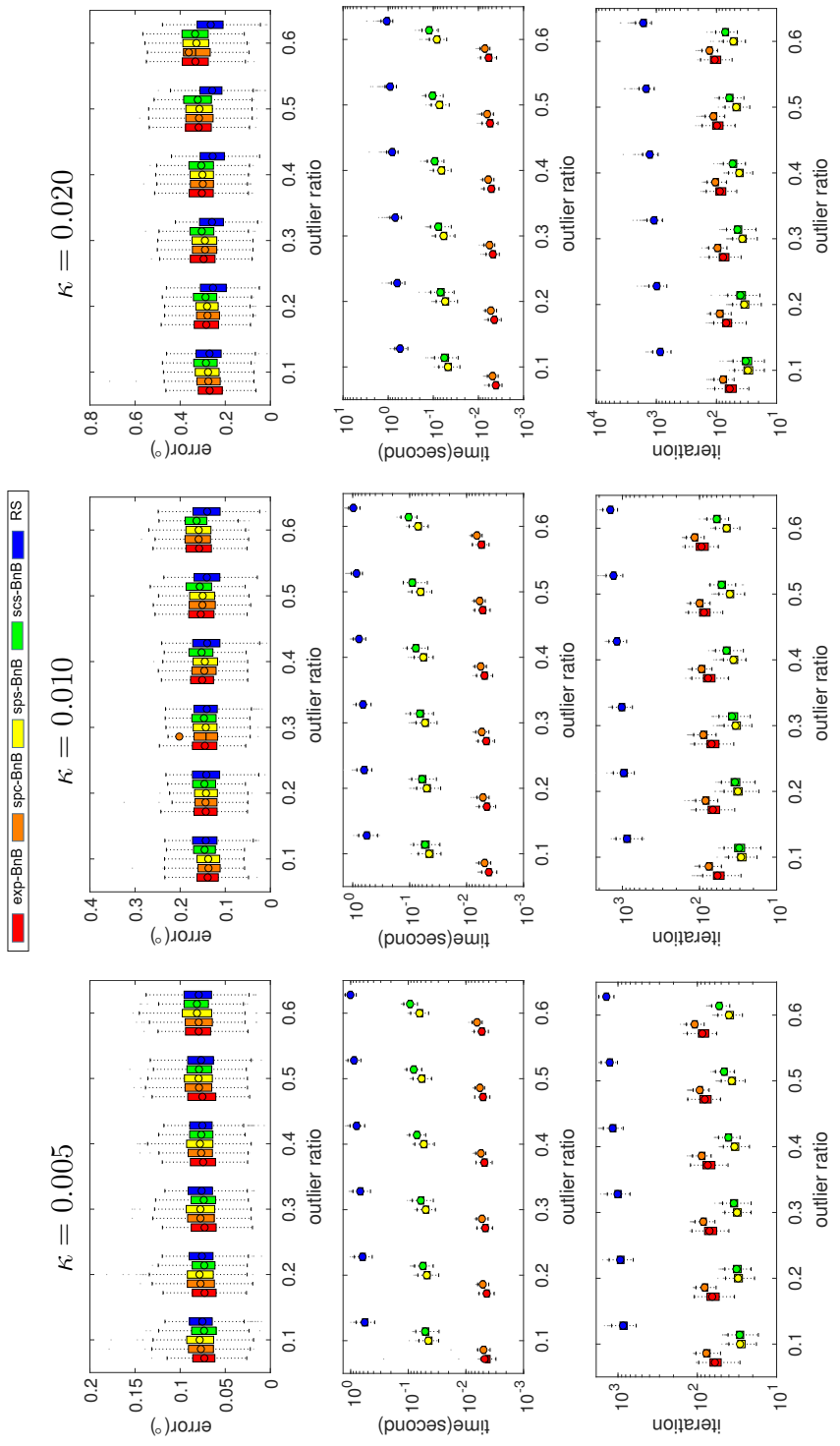


Figure 3.17: Full Atlanta frame estimation experiments. The first row shows the frame error (°) in different noise levels. The second row shows the runtime (second) in different noise levels. The third row shows the iteration count in different noise levels.

To generate the input normals in Manhattan world, we randomly selected a point $\mathbf{R}_{gt}$ in $\mathbb{SO}(3)$ as the Manhattan frames. In other words, each column of R_{gt} corresponded to a Manhattan frame. The experimental settings were $N = 500, \kappa = \{0.005, 0.010, 0.020\}$ and ρ was set from 0.1 to 0.6. Once the frame directions had been estimated as R^*, the estimation error was measured by

$$mean\big(\arccos\big(max(abs(\mathbf{R}_{gt}^T\mathbf{R}^*))\big)\big) \tag{3.125}$$

where $mean(\cdot)$ was the average function; $\arccos(\cdot)$ was the element-wise arccosine function; $max(\cdot)$ was the column-wise max function. It computed the average error of the three frames.

Note that the solution of rotation search method inherently satisfies the $\mathbb{SO}(3)$ constraint, while the solutions of our sequential methods were determined without this constraint, as they were formulated for general Atlanta frame estimation.

The results are given in Fig. 3.17. The accuracy of the rotation search method was slightly better than that of the sequential methods in large noise level. This was because the rotation search method considered the three orthogonal constraints. Nevertheless, the runtime of the rotation search was much greater than that of the sequential methods, due to the fact that the sequential methods had lower dimensionality, tighter bounds and fewer iterations in the BnB framework.

3.5.3 Real Data Experiments

3.5.3.1 NYUv2 Data

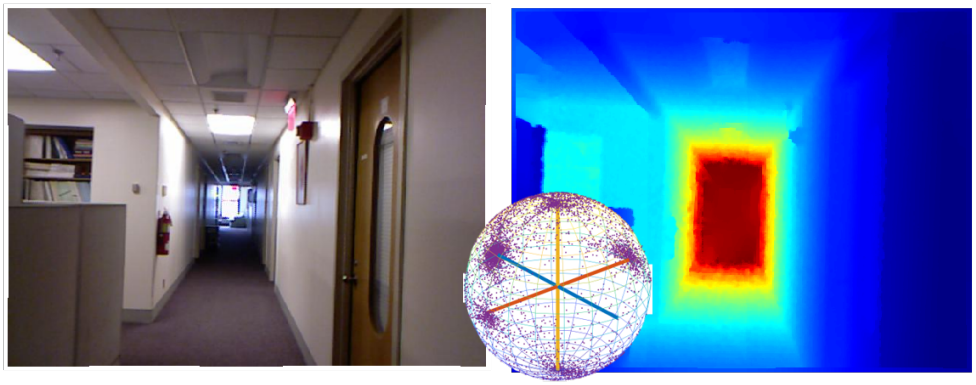


Figure 3.18: Selected image from NYUv2 data set.

We tested our method on the NYUv2 dataset [147], which contained 1449 RGB images, along with the corresponding depths, as well as camera information(see Fig. 3.18). The data involved a variety of indoor scenes that were considered to be a man-made structural world. In our experiments, we utilized the data to estimate the vertical direction of the scenes. Concretely, we generated the normals from the depth image by the Matlab built-in function *pcnormals*, and estimated the vertical

direction from the downsampled normal data ($N \approx 3000$) for all scenes. The threshold was set to 2° in all methods. For RANSAC, $\rho = \{0.65, 0.75, 0.85, 0.95\}$ were tested since the ground truth of the outlier ratio of each scene was unknown, and the sample iteration was determined by ρ (Eq.(3.121)).

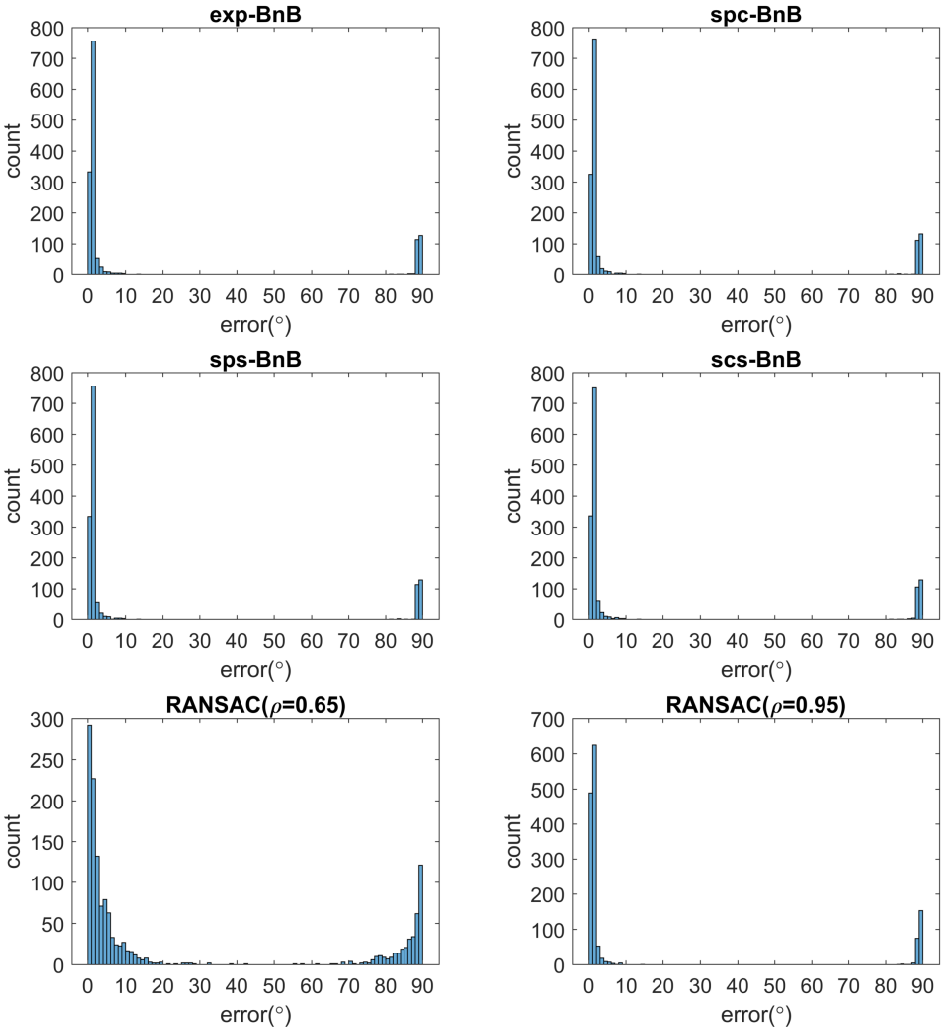


Figure 3.19: The distribution of error for different methods in NYUv2 data.

The distribution of error (Eq.(3.123)) is shown in Fig.3.19. The results revealed that the estimation errors of the BnB algorithms were all concentrated at 0° and 90°. This is because there were some degenerate scenes in the data set, which were degenerated into the Manhattan assumption, or even worse, only two main orthogonal frames (Fig.3.21). Estimating vertical direction in such degenerate scenes might return a frame direction in the horizontal plane. Consequently, some errors were concentrated at 90°. Furthermore, when the outlier proportion ρ was set low, the estimation errors of the RANSAC algorithm were not concentrated. When the outlier proportion ρ was set high, the errors were also concentrated. Moreover, to demonstrate the results quantitatively, the ϵ-recall

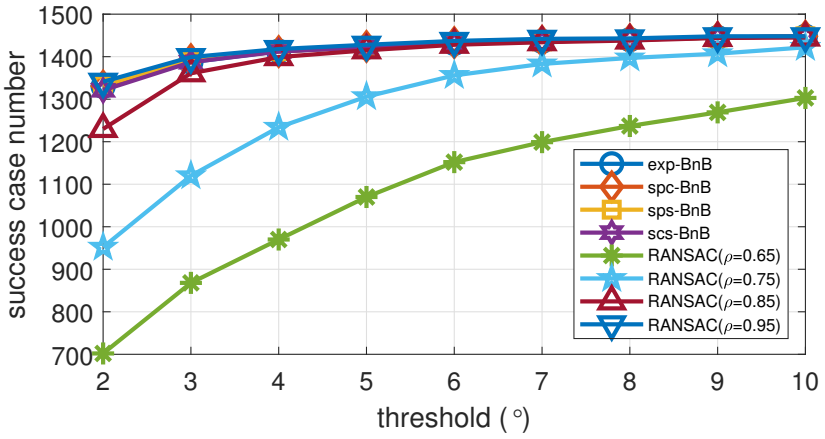
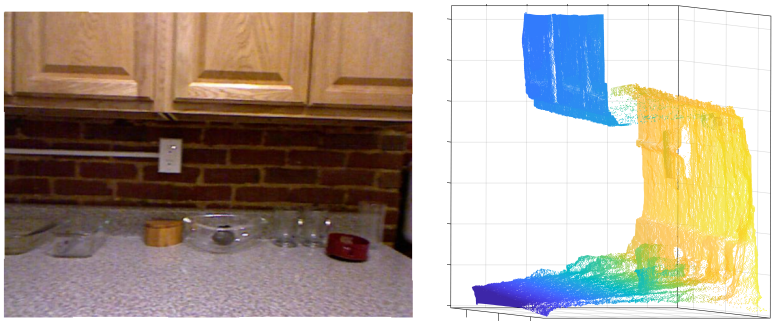


Figure 3.20: ϵ-recall curve in NYUv2 data. (Higher is better)

curve was presented in Fig. 3.20, where the success case was satisfied $error < \epsilon$ or $error > 90° - \epsilon$.



(a) (b)

Figure 3.21: Degenerate case in NYUv2 data. (a) a degenerate scene that has only two main frames. (b) the point cloud of the left scene, which is viewed from the right side.

Furthermore, the four bounds in $\mathbb{S}^2$ had different efficiencies. Specifically, the distribution of iteration and runtime in NYUv2 data are shown in Fig. 3.22. More specifically, the median runtime and iteration can be found in Table 3.2. Obviously, the exp-BnB algorithm was the most efficient. On the other hand, RANSAC ran very fast when the outlier proportion ρ was set low, although it may have returned incorrect results. If the outlier proportion ρ was set high ($\rho = 0.95$), its runtime was longer than that of the exp-BnB algorithm. Besides, to compare with rotation search, we directly quote the results from [93]. With rotation search bounds, it needs 117.06s on average to estimate Manhattan frames for each scene without input sampling. However, with an efficient bounds computation method like the one proposed in [93], it needs only 0.07s on average.

Lastly, it is worth noting that it is not easy to estimate the outlier proportion (i.e., ρ) appropriately in various real applications. If ρ is overestimated, RANSAC needs redundant random sampling, which leads to unnecessary running time. If

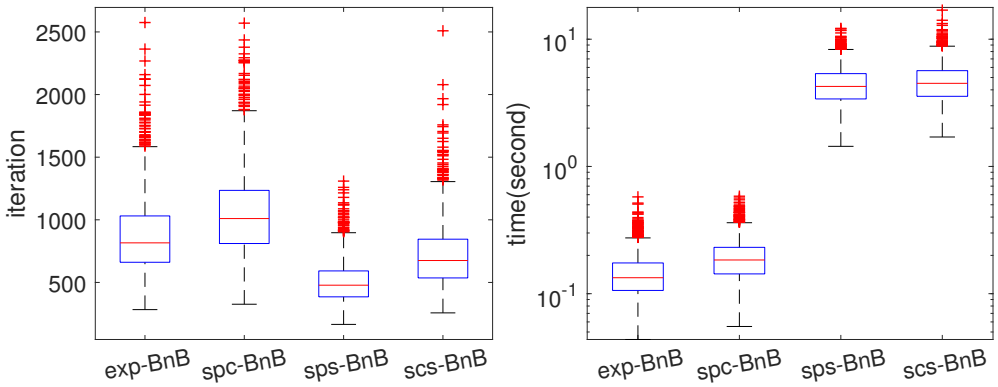


Figure 3.22: The distribution of iteration and runtime in NYUv2 data.

Table 3.2: Median runtime and iteration of different methods in NYUv2 data.

Methods	Median time(s)	Median iteration
exp-BnB	0.134	816
spc-BnB	0.184	1010
sps-BnB	4.256	478
scs-BnB	4.500	675
RANSAC($\rho = 0.65$)	0.007	36
RANSAC($\rho = 0.75$)	0.013	72
RANSAC($\rho = 0.85$)	0.038	203
RANSAC($\rho = 0.95$)	0.344	1840

ρ is underestimated, RANSAC may return incorrect results. In contrast, the proposed BnB-type methods only need the inlier threshold to be specified and they always return a globally-optimal solution, which is more adaptable.

3.5.3.2 Outdoor Data

In this part, we verified the validity of our methods with the outdoor scene. The data set[6] was recorded in the old town of Bremen, Germany (see Fig.(3.23)). It contained 13 3D scans, each with up to 22,500,000 points. Estimating the vertical direction first may be useful for registering the scenes [128]. For each scene, it was considered as an Atlanta world and $[0, 0, -1]^T$ was roughly set as the ground truth of vertical direction. We firstly down-sampled the inputs using the Matlab built-in function *pcdownsample*. More specifically, a box grid filter was used to reduce the inputs ($N \approx 400,000$). Then their normals were computed by *pcnormals*, and the vertical direction was estimated from the obtained normals. $\rho = \{0.8, 0.9\}$ were set in RANSAC. The inlier threshold was set $\epsilon = 1^\circ$ for all methods in these experiments. To emphasize the global optimality of our

[6] `http://kos.informatik.uni-osnabrueck.de/3Dscans/`

proposed algorithms, 200 trials were repeated in each scene. Average runtime and error were recorded.

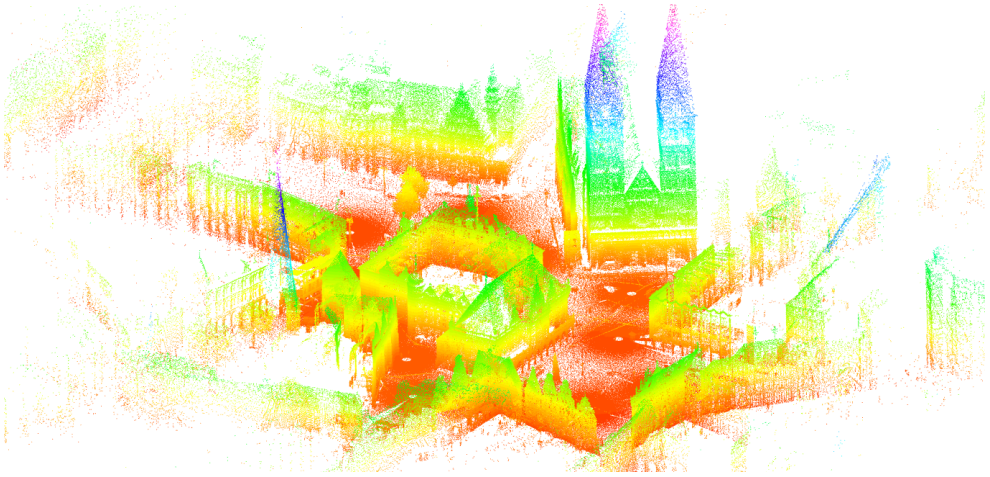


Figure 3.23: The whole scene of the Bremen city data, which are merged using markers as tie points.

The results can be found in Table 3.3 . Note that the ground truth for vertical direction was roughly set, and the errors were only indicating that the vertical direction estimation results were roughly correct. In these outdoor settings, all bounds in $\mathbb{S}^2$ can be nested into the BnB algorithm to globally estimate the vertical direction. Furthermore, the results showed that the exp-BnB and spc-BnB algorithm had a similar performance and were more efficient. Note that RANSAC returned incorrect solutions ($error > 10°$) occasionally due to its heuristic nature. That is why their average errors were slightly larger than that of globally optimal algorithms. Besides, the rotation search method could not terminate in 1800s (30min) in each scene. However, according to the results in [115], with the help of an accelerating method, it takes about 80s to estimate Atlanta frames in the whole scene.

Table 3.3: Vertical direction estimation results in outdoor data.

Methods	Average time(s)	Average error(°)
exp-BnB	1.233	1.229
spc-BnB	1.296	1.242
sps-BnB	85.014	1.240
scs-BnB	147.961	1.224
RANSAC($\rho = 0.8$)	0.928	1.421
RANSAC($\rho = 0.9$)	3.325	1.254

3.6 Conclusion

We propose a novel method for estimating the vertical direction in Atlanta world. It obtains the globally optimal solution by applying a BnB algorithm, without requiring any prior knowledge of the number of frames. Since estimating vertical direction is inherently a two-dimensional problem, we propose new bounds in $\mathbb{S}^2$ for BnB which are different from the conventional bounds in the rotation search.

The experimental results show that all the bounds (in $\mathbb{S}^2$ or $\mathbb{SO}(3)$) can be nested inside the BnB algorithm to obtain the global solution, and the bounds in $\mathbb{S}^2$ outperform the bounds in $\mathbb{SO}(3)$, which is the state-of-the-art technique, for estimating vertical direction globally. Furthermore, these four bounds in $\mathbb{S}^2$ exhibit varying performance. Generally, exp-BnB and spc-BnB have similar performance and are more efficient. Moreover, although sps-BnB and scs-BnB have tighter bounds, they are rather inefficient because of the heavy computational burden. In addition to the quality of the bounds, appropriate parametrization of the search domain is also an important factor of the efficiency of the BnB algorithm. We think this is why the sps-BnB is more efficient than the scs-BnB algorithm.

Lastly, since the sps-BnB has the least iterations, there may be a hope of accelerating the calculation of the *ste-square* bounds to obtain a faster BnB algorithm in further work. In addition, since the *ste-square* bounds are very tight in $\mathbb{S}^2$ according to the experimental results, similarly, there may be tighter provable bounds in the rotation search.

Chapter 4

Globally Optimal Camera Orientation Estimation from Line Correspondences

This chapter is regarding applications of rotation search theory in camera pose estimation. The core idea is to decouple rotation and translation from rigid pose. After that, we can sequentially solve the decoupled subproblems. Especially, the globally optimal solution for camera orientation can be obtained. This chapter is a modified version of

- Liu, Yinlong, Guang Chen, and Alois Knoll. *Globally Optimal Camera Orientation Estimation from Line Correspondences by BnB algorithm.* IEEE Robotics and Automation Letters 6.1 (2020): 215-222.

It has been reproduced here with the permission of the copyright holder.

4.1 Background

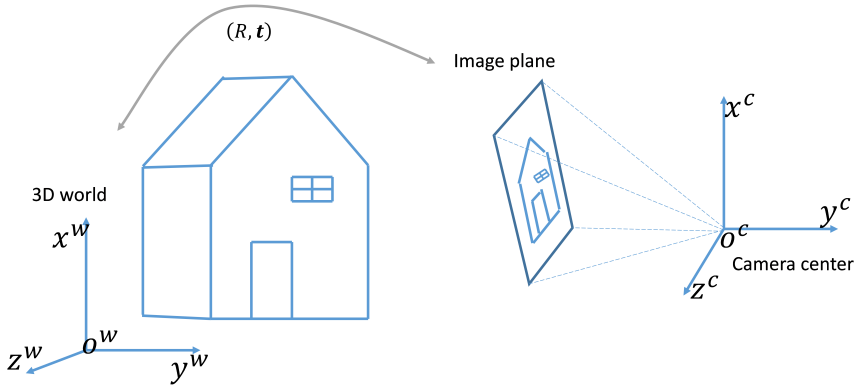


Figure 4.1: PnL problem: determining the relative position and orientation of a camera and an object from line correspondences.

Absolute camera pose estimation is determining the position and orientation of a camera in the scene, which is a core task in computer vision and robot navigation.

This task can be solved using 2D/3D line feature correspondences, which is also known as the Perspective-n-Line (PnL) problem [59](see Fig.4.1). It plays an import role in many computer vision applications, e.g., Simultaneous Localization and Mapping (SLAM) [148] and robot localization and navigation [149, 150]. It is worth noting that the PnL approach is inherently suitable for texture-less scenes, e.g., man-made structural environments [151, 152].

To address the PnL problem, one of the most important preconditions is knowing the correspondences between 3D line features in real world and their 2D counterparts in the image plane. Unfortunately, incorrect correspondences, which are also known as outliers, are usually unavoidable in the real applications. These mismatches seriously impair the camera pose estimation: even a small percentage of outliers might significantly decrease the accuracy of outlier-free algorithms [33]. To reduce the impact of corrupted data, the de facto standard mechanism is combining an outlier-free PnL algorithm with a *RANdom Sample Consensus* (RANSAC) scheme [34, 151]. Broadly speaking, RANSAC randomly samples minimal subsets of the inputs and applies the embedded outlier-free PnL algorithm to obtain candidate solutions. After repeating the sampling routine many times, RANSAC returns the best candidate with the largest inlier set as the final solution, which is also known as a hypothesize-and-verify framework. However, the randomized nature of RANSAC does provide no guarantee of the optimality of its solution. In other words, there is no absolute certainty that the obtained result is a satisfactory solution [57].

Nevertheless, there are some safety-critical applications (e.g., self-driving cars) which demand that such spatial perception algorithms can provide certifiably optimal solutions in the presence of noise and outliers [30, 153]. Conventional methods that combine outlier-free PnL algorithms and RANSAC fail to achieve this goal. In this thesis, we take a big step towards this objective: we propose a novel method for obtaining the globally optimal camera orientation from outlier-contaminated line correspondences. Furthermore, as the camera pose comprises position (i.e., translation $\boldsymbol{t} \in \mathbb{R}^3$) and orientation (i.e., rotation $\mathbf{R} \in \mathbb{SO}(3)$), obtaining the provably optimal camera orientation is usually favorable for addressing the problem of the certifiably optimal camera pose [154, 155].

In addition, many modern vision systems are equipped with Inertial Measurement Units (IMUs), which could provide a prior knowledge of the vertical direction [150, 156]. Accordingly, we first proposed a singularity-free and thus more accurate algorithm for the outlier-free known-vertical-direction PnL problem. Furthermore, for outlier-contaminated cases, a special camera orientation estimation algorithm with a guarantee of the optimality is proposed.

4.2 Related Work

Since camera pose estimation is a long-history and well-studied task in computer vision, it can be formulated as many different kinds of problems, e.g., Perspective-n-Point (PnP) problem [157, 158], Perspective-n-Line (PnL) problem [59], and

pose from points and lines(PnPL) problem [159,160]. These different formulations share many common techniques, nonetheless, we focus on PnL algorithms, which are the most related to our thesis.

Conventionally, combing outlier-free PnL algorithms with RANSAC is one of the most commonly used mechanisms for estimating the camera pose from outlier-contaminated line correspondences. For the RANSAC scheme [161], some recent advancements focus on special geometrical problems (e.g., pseudo-convex [162, 163]); other recent advancements still do not change its randomized nature and still lack absolute certainty that the obtained solution is optimal [164, 165]. In any case, they are not directly used to address PnL problems. Therefore, we review the most relevant PnL algorithms in detail.

4.2.1 PnL Algorithms

According to the optimizing techniques, we can divide these methods into three categories:

(a) Locally iterative PnL solutions [166–168]. Generally speaking, these local iterative algorithms formulate the pose estimation problem as a nonlinear least squares optimization problem. Unfortunately, the objectives usually are non-convex because of noise and outliers. Therefore, if the initialization is not carefully set, these locally iterative algorithms might be trapped in a local optimum, which might be far from the true camera pose [151].

(b) Algebraic solutions [169–171]. The algebraic algorithms estimate the camera pose by solving a polynomial system of equations. One of the most important advantages of these algebraic algorithms is that they can obtain the globally optimal solution from outlier-free inputs without careful initializing. However, to obtain a robust solution from outlier-contaminated inputs, they must be nested inside a non-deterministic RANSAC framework.

(c) Linearized PnL solutions [59, 151]. The linearized PnL methods formulate the line correspondences as a homogeneous system of linear equation by dropping some constraints that might compromise the accuracy [156,171]. The final camera pose can be extracted from the solution of the linear system by adding the dropped constraints.

It is worth noting that to suppress outlier inputs, these linearized PnL methods can incorporate with Algebraic Outlier Rejection (AOR [172]) [151]. AOR can estimate the robust solution of the linear objective very quickly. Therefore, the camera pose can be recovered from the robust linear solution. However, these methods usually have a *break-point*, which means that when the proportion of outliers reaches the *break-point*, these methods cannot obtain a satisfactory solution [151, 172].

4.2.2 Known-Vertical-Direction PnL Algorithms

With the help of relatively cheap IMUs, some recent work has focused on estimating camera pose from line correspondences with a known vertical direc-

tion [150, 156, 173]. In common, they first estimate the camera orientation and then solve translation. Specially, linearized algorithms are proposed for estimating camera rotation in [156] and [150]. The authors in [156] pointed out that the biggest disadvantage of a linear solution is ignoring orthogonality, which might lead to low accuracy.

To consider the orthogonality of camera rotation, a cubic polynomial solution, which belongs to the algebraic algorithm, is proposed for determining the camera orientation [173, 174]. However, it employs a singularity-affected rotation parameterization that will reduce the accuracy in some cases. In this paper, we apply a singularity-free rotation parameterization to improve accuracy.

For outlier-contaminated inputs, a novel RANSAC2 algorithm, whose inner routine requires only two random 2D-3D pairs of line correspondences, is proposed to estimate the rotation [150]. Actually, one line correspondence is sufficient to be nested inside RANSAC [175]. Nonetheless, RANSAC still cannot guarantee the optimality of the solution.

4.3 Rotation Estimation

4.3.1 Mathematical Formulation

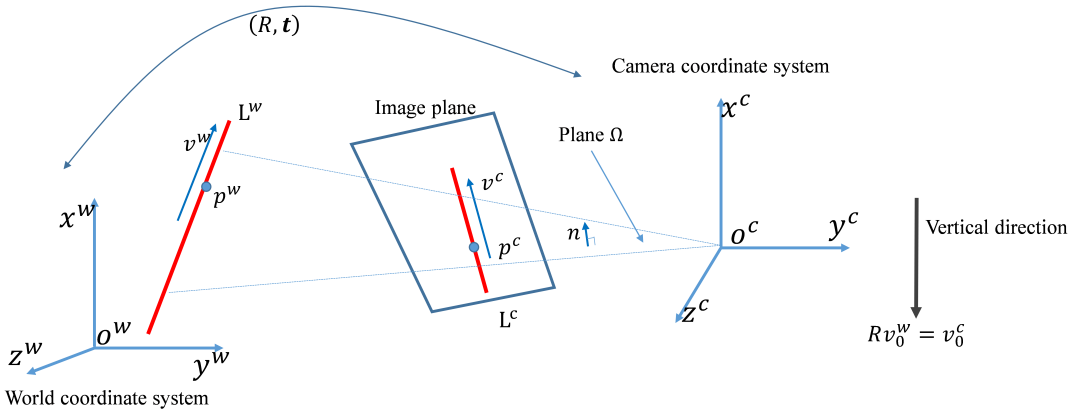


Figure 4.2: Line definition and the geometrical constraints in PnL problem.

Given a calibrated camera, the lines in the image plane can be denoted as L^c in the camera coordinate system $\{\boldsymbol{o}^c\boldsymbol{x}^c\boldsymbol{y}^c\boldsymbol{z}^c\}$ and the lines in the real world can be denoted as L^w in the world coordinate system $\{\boldsymbol{o}^w\boldsymbol{x}^w\boldsymbol{y}^w\boldsymbol{z}^w\}$ (see Fig. 4.2). Geometrically, a 3D line can be represented by a point $\boldsymbol{p} \in \mathbb{R}^3$ on the line and a unit direction $\boldsymbol{v} \in \mathbb{R}^3$ denoting the direction of the line, therefore L^c and L^w can be represented as:

$$L^c := \boldsymbol{p}^c + \lambda_c \boldsymbol{v}^c, \quad L^w := \boldsymbol{p}^w + \lambda_w \boldsymbol{v}^w \tag{4.1}$$

where λ_c and λ_w are parameters describing particular points on each line. Note that the point $\boldsymbol{p}^c$ is not necessarily the corresponding image point of the 3D point

$\boldsymbol{p}^w$. The objective of PnL is to estimate the camera pose (i.e., rotation/orientation $\mathbf{R}$ and translation/position $\boldsymbol{t}$) from a set of line correspondences $\{(L_i^w, L_i^c)\}_{i=1}^N$.

To obtain globally optimal rotation, we first formulate constraints about rotation $\mathbf{R}$ by constructing an auxiliary variable. For a line correspondence, there is a projection plane Ω passing through the camera center and both lines. The unit normal of the plane Ω is denoted by $\boldsymbol{n}$ in the camera coordinate system and it can be obtained from the line L^c in the image plane:

$$\boldsymbol{n} = \frac{\boldsymbol{v}^c \times \boldsymbol{p}^c}{\|\boldsymbol{v}^c \times \boldsymbol{p}^c\|} \tag{4.2}$$

For the i-th line pair, there is an important geometrical constraint [156, 176].

$$\boldsymbol{n}_i^T \mathbf{R} \boldsymbol{v}_i^w = 0 \tag{4.3}$$

Furthermore, there is another constraint for translation $\boldsymbol{t}$.

$$\boldsymbol{n}_i^T (\mathbf{R}\boldsymbol{p}_i^w + \boldsymbol{t}) = 0; \tag{4.4}$$

Conventionally, the outlier-free PnL algorithms formulate the objective as [171, 176].

$$\min_{\mathbf{R}} \sum_{i=1}^{N} \left(\boldsymbol{n}_i^T \mathbf{R} \boldsymbol{v}_i^w\right)^2 \tag{4.5a}$$

$$\min_{\boldsymbol{t}} \sum_{i=1}^{N} \left(\boldsymbol{n}_i^T (\mathbf{R}\boldsymbol{p}_i^w + \boldsymbol{t})\right)^2 \tag{4.5b}$$

Camera pose can be obtained by optimizing these two objectives sequentially. However, it is well known that the objectives are sensitive to outliers [59, 151]. In contrast to the outlier-free PnL algorithm, our robust objective is formulated by maximizing the cardinality of the inlier set [29].

$$\max_{R} \sum_{i=1}^{N} \mathbb{I}\big(|\boldsymbol{n}_i^T \mathbf{R} \boldsymbol{v}_i^w| \leq \epsilon\big) \tag{4.6a}$$

$$\max_{\boldsymbol{t}} \sum_{i=1}^{N} \mathbb{I}\big(|\boldsymbol{n}_i^T (\mathbf{R}\boldsymbol{p}_i^w + \boldsymbol{t})| \leq \epsilon\big) \tag{4.6b}$$

where ϵ is the inlier threshold and $\mathbb{I}(\cdot)$ is the indicator function. Specifically, $\mathbb{I}(\cdot)$ returns 1 if the condition $\cdot$ is true, and it returns 0 if the condition $\cdot$ is not true. The maximization cardinality formulation is inherently robust to outliers and is successfully applied in many applications of robust estimation [29].

It is worth noting that the formulation of Eq. (4.6) naturally decouples rotation and translation. Normally, to obtain the globally optimal solution of a rigid pose, nested BnB should be applied [52, 66]. However, the nested BnB algorithm needs to search a given 6-dimensional space (i.e., $\mathbb{SE}(3)$), which requires a lot of computing resources. By decoupling the optimization problem in 6-dimensional space into two optimization sub problems in 3-dimensional space will significantly reduce computing requirements.

4.3.2 Branch-and-Bound

We first focus on obtaining the globally optimal solution of camera orientation (i.e., solving Eq. (4.6a)). To obtain the globally optimal solution, we introduce the branch-and-bound algorithm (BnB) [177], which is a global optimization technique that is applied in many geometrical vision problems (e.g., 3D point cloud registration [66]).

Specifically, the BnB algorithm recursively divides the best possible branch (i.e., solution domain) into small branches (i.e., branching), then it calculates the upper and lower bounds of the optimum in each divided branches (i.e., bounding). By checking the upper and lower estimated bounds, it removes some divided branch which cannot produce a better solution than the best one found by the algorithm so far (i.e., pruning). The branching-bounding-pruning process is repeated until the desired accuracy is approached and then the optimal solution is found.

Algorithm 2: Camera orientation estimation by BnB

Input: Line correspondences $\{L_i^w, L_i^c\}_{i=1}^N$, inlier threshold ϵ
Output: Optimal camera oritiention $\mathbf{R}^*$

Initialize the searching domain $\mathbb{D}$ with $\mathbb{SO}(3)$;
Initialize the branches queue $\mathbb{Q} \leftarrow \emptyset$ and the optimal branch $\mathbb{B} \leftarrow \mathbb{D}$;
Initialize upper bound $E_U \leftarrow N$ and lower bound $E_L \leftarrow 0$;
while $|E_U - E_L| \geq 1$ **do**
    Divide $\mathbb{B}$ into eight cubes **(branching)**;
    Add all sub-branches into $\mathbb{Q}$ with their lower bounds e_L and upper bounds e_U **(bounding)**;
    Update $E_L \leftarrow \max\{e_L^i\}$, i for all branches in $\mathbb{Q}$;
    Update $E_U \leftarrow \max\{e_U^i\}$, i for all branches in $\mathbb{Q}$;
    Take out the branch that has the maxmum lower bound E_L from $\mathbb{Q}$ and use it to update optimal branch $\mathbb{B}$ and use its center to update R^*;
    Delete the branches whose $e_U^i < E_L$, i for all branches in $\mathbb{Q}$ **(pruning)**;
end

The algorithm for globally optimal rotation estimation using BnB is summarized in **Algorithm** 2. To globally search the optimal rotation, $\mathbb{SO}(3)$ needs to be parameterized properly. Accordingly, the minimally parameterized axis-angle parameterization is used to represent the rotation space [66]. Specifically, a 3D rotation $\mathbf{R} \in \mathbb{SO}(3)$ corresponds to a 3-vector $\boldsymbol{r} \in \mathbb{R}^3$ whose direction and norm specify the axis and angle of rotation. Their relationship follows Rodrigues' rotation formula:

$$\mathbf{R} = \exp([\boldsymbol{r}]_\times) = \mathbf{I} + \sin(\theta)[\overline{\boldsymbol{r}}]_\times + \big(1 - \cos(\theta)\big)[\overline{\boldsymbol{r}}]_\times^2 \tag{4.7}$$

where $\mathbf{I}$ is the 3×3 identity matrix; $\theta = ||\boldsymbol{r}||$ is the angle of the rotation; $\overline{\boldsymbol{r}} = \boldsymbol{r}/||\boldsymbol{r}||$ is the axis of the rotation; $[\bullet]_\times$ is the skew-symmetric matrix for a vector $\bullet$ and $\exp(\cdot)$ is the matrix exponential of the $\mathbb{SO}(3)$ algebra [178]. With the

help of axis-angle parameterization, all possible rotations are in a π-ball [66]. For ease of manipulation, a cube enclosing the π-ball is used as the initial searching domain. Therefore, the branching process in the BnB algorithm divides the larger cube into eight sub-cubes.

The key part of the BnB algorithm is estimating the upper and lower bounds efficiently and tightly in each divided branch. Given a divided cube-shape rotation branch, whose center is $\boldsymbol{r}_0$, the lower bound of the optimum can be set as

$$e_L = \sum_{i=1}^{N} \mathbb{I}\big(|\boldsymbol{n}_i^T \mathbf{R}_{r_0} \boldsymbol{v}_i^w| \leq \epsilon\big) \tag{4.8}$$

where $\mathbf{R}_{r_0}$ is the matrix form of rotation $\boldsymbol{r}_0$.

Proof. The function value at a specific point (i.e. $\mathbf{R}_{r_0}$) within the domain must be less than the maximum value. □

On the other hand, given a divided cube-shape rotation branch, whose center is $\boldsymbol{r}_0$ and whose side length is ϕ, the upper bound of the optimum can be set as

$$e_U = \sum_{i=1}^{N} \mathbb{I}\Big(|\boldsymbol{n}_i^T \mathbf{R}_{r_0} \boldsymbol{v}_i^w| \leq \cos\big(\lfloor \arccos(\epsilon) - \psi \rfloor\big)\Big) \tag{4.9}$$

where $\mathbf{R}_{r_0}$ is the matrix form of rotation $\boldsymbol{r}_0$; $\psi \triangleq \min\{\sqrt{3}\phi/2, \pi\}$; $\lfloor\cdot\rfloor$ is a non-negative function. Specifically,

$$\lfloor x \rfloor = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{4.10}$$

Proof. To derive the upper bound, we first introduce lemma 2.5 in Chapter 2:

Lemma. *For an arbitrary vector $\boldsymbol{v} \in \mathbb{R}^3$ and two rotations $\mathbf{R}_r$ and $\mathbf{R}_{r_0}$ in matrix form, $\boldsymbol{r}$ and $\boldsymbol{r}_0$ in angle-axis form, then*

$$\angle(\mathbf{R}_r \boldsymbol{v}, \mathbf{R}_{r_0} \boldsymbol{v}) \leq ||\boldsymbol{r} - \boldsymbol{r}_0|| \tag{4.11}$$

where $\angle(\cdot,\cdot)$ is the angle of two vectors. According to lemma 2.5, given a divided cube-shape rotation branch, whose center is $\boldsymbol{r}_0$ and side is ϕ, we have

$$\angle(\mathbf{R}_r \boldsymbol{v}_i^w, \mathbf{R}_{r_0} \boldsymbol{v}_i^w) \leq \min\{||\boldsymbol{r} - \boldsymbol{r}_0||, \pi\} \tag{4.12}$$

$$\leq \min\{\frac{\sqrt{3}}{2}\phi, \pi\} \triangleq \psi \tag{4.13}$$

where $\boldsymbol{r}$ is an arbitrary point in the cube, whose matrix form is $\mathbf{R}_r$. According to triangle inequality

$$\angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) + \angle\big(\mathbf{R}_r\boldsymbol{v}_i^w, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) \tag{4.14}$$
$$\le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) + \psi \tag{4.15}$$
$$\angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \ge \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) - \angle\big(\mathbf{R}_r\boldsymbol{v}_i^w, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) \tag{4.16}$$
$$\ge \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) - \psi \tag{4.17}$$

Hence,

$$\angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) - \psi \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) + \psi \tag{4.18}$$

Observe

$$\mathbb{I}\big(|\boldsymbol{n}_i^T\mathbf{R}_r\boldsymbol{v}_i^w| \le \epsilon\big) \tag{4.19}$$
$$\Leftrightarrow \mathbb{I}\big(-\epsilon \le \boldsymbol{n}_i^T\mathbf{R}_r\boldsymbol{v}_i^w \le \epsilon\big) \tag{4.20}$$
$$\Leftrightarrow \mathbb{I}\Big(-\epsilon \le \cos\Big(\angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big)\Big) \le \epsilon\Big) \tag{4.21}$$
$$\Leftrightarrow \mathbb{I}\big(\arccos(\epsilon) \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \pi - \arccos(\epsilon)\big) \tag{4.22}$$

Also, we note that

$$\arccos(\epsilon) \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \pi - \arccos(\epsilon) \tag{4.23}$$
$$\Rightarrow \begin{cases} \arccos(\epsilon) & \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) + \psi \\ \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) - \psi & \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \pi - \arccos(\epsilon) \end{cases} \tag{4.24}$$
$$\Rightarrow \begin{cases} \arccos(\epsilon) & \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) + \psi \\ \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) - \psi & \le \pi - \arccos(\epsilon) \end{cases} \tag{4.25}$$
$$\Rightarrow \ \arccos(\epsilon) - \psi \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) \le \pi - \arccos(\epsilon) + \psi \tag{4.26}$$
$$\Rightarrow \ \lfloor\arccos(\epsilon) - \psi\rfloor \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) \le \pi - \lfloor\arccos(\epsilon) - \psi\rfloor \tag{4.27}$$

$$\Rightarrow \ -\cos(\lfloor\arccos(\epsilon) - \psi\rfloor) \le \cos\Big(\angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big)\Big) \le \cos(\lfloor\arccos(\epsilon) - \psi\rfloor) \tag{4.28}$$
$$\Rightarrow \ -\cos(\lfloor\arccos(\epsilon) - \psi\rfloor) \le \boldsymbol{n}_i^T\mathbf{R}_{r_0}\boldsymbol{v}_i^w \le \cos(\lfloor\arccos(\epsilon) - \psi\rfloor) \tag{4.29}$$
$$\Rightarrow \ |\boldsymbol{n}_i^T\mathbf{R}_{r_0}\boldsymbol{v}_i^w| \le \cos\big(\lfloor\arccos(\epsilon) - \psi\rfloor\big) \tag{4.30}$$

where Eq. (4.27) follows $0 \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_{r_0}\boldsymbol{v}_i^w\big) \le \pi$.

Therefore,

$$|\boldsymbol{n}_i^T\mathbf{R}_r\boldsymbol{v}_i^w| \le \epsilon \tag{4.31}$$
$$\Leftrightarrow \arccos(\epsilon) \le \angle\big(\boldsymbol{n}_i, \mathbf{R}_r\boldsymbol{v}_i^w\big) \le \pi - \arccos(\epsilon) \tag{4.32}$$
$$\Rightarrow |\boldsymbol{n}_i^T\mathbf{R}_{r_0}\boldsymbol{v}_i^w| \le \cos\big(\lfloor\arccos(\epsilon) - \psi\rfloor\big) \tag{4.33}$$

Then

$$\mathbb{I}\big(|\boldsymbol{n}_i^T\mathbf{R}_r\boldsymbol{v}_i^w| \le \epsilon\big) = 1 \Rightarrow \mathbb{I}\left(|\boldsymbol{n}_i^T\mathbf{R}_{r_0}\boldsymbol{v}_i^w| \le \cos\left(\lfloor\arccos(\epsilon) - \psi\rfloor\right)\right) = 1 \tag{4.34}$$

Hence,

$$\mathbb{I}\big(|\boldsymbol{n}_i^T\mathbf{R}_r\boldsymbol{v}_i^w| \leq \epsilon\big) \leq \mathbb{I}\left(|\boldsymbol{n}_i^T\mathbf{R}_{r_0}\boldsymbol{v}_i^w| \leq \cos\left(\lfloor\arccos(\epsilon) - \psi\rfloor\right)\right) \tag{4.35}$$

And,

$$\max \sum_{i=1}^{N} \mathbb{I}\big(|\boldsymbol{n}_i^T\mathbf{R}\boldsymbol{v}_i^w| < \epsilon\big) \leq \sum_{i=1}^{N} \mathbb{I}\Big(|\boldsymbol{n}_i^T\mathbf{R}_{r_0}\boldsymbol{v}_i^w| \leq \cos\left(\lfloor\arccos(\epsilon) - \psi\rfloor\right)\Big) \tag{4.36}$$

Therefore, e_U is an upper bound. □

In addition, when the given sub branch collapses to a single point, obviously, $e_L = e_U$. Therefore, e_L and e_U are sufficient to be nested into the BnB algorithm.

4.4 Rotation Estimation with Known Vertical Direction

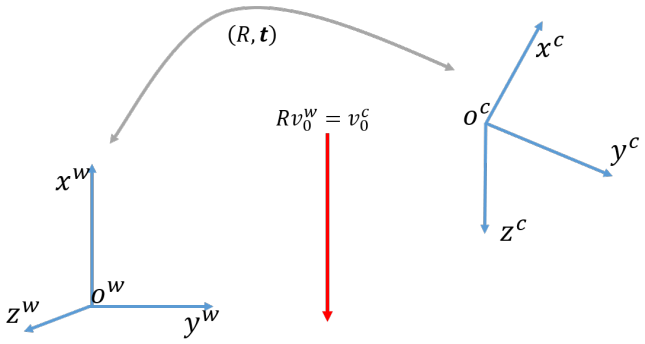


Figure 4.3: Geometrical constraints with a vertical direction

4.4.1 Mathematical Formulation

In this section, we consider the special case where the vertical direction is known. Geometrically, prior knowledge of the vertical direction is a constraint [156], see Fig. 4.3:

$$\mathbf{R}\boldsymbol{v}_0^w = \boldsymbol{v}_0^c \tag{4.37}$$

where $\boldsymbol{v}_0^w$ and $\boldsymbol{v}_0^c$ are the unit-norm vertical direction in the world coordinate system and camera coordinate system, respectively. The solution of Eq. (4.37) is

$$\mathbf{R} = \mathbf{R}_{\boldsymbol{v}_0^c} \cdot \mathbf{R}_w^c \tag{4.38}$$

where $\mathbf{R}_w^c$ is the rotation that aligns $\boldsymbol{v}_0^w$ to $\boldsymbol{v}_0^c$ with the minimum geodesic motion (see [146]); $\mathbf{R}_{\boldsymbol{v}_0^c} = \exp(\alpha[\boldsymbol{v}_0^c]_\times)$ is a rotation that rotates α about axis $\boldsymbol{v}_0^c$. Hence, the objective of our work is to determine α correctly.

For the i-th line pair constraints:

$$\begin{aligned}
&\boldsymbol{n}_i^T \mathbf{R} \boldsymbol{v}_i^w && (4.39)\\
=&\boldsymbol{n}_i^T \cdot \mathbf{R}_{\boldsymbol{v}_0^c} \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w && (4.40)\\
=&\boldsymbol{n}_i^T \cdot \Big(\mathbf{I} + \sin(\alpha)[\boldsymbol{v}_0^c]_\times + \big(1-\cos(\alpha)\big)[\boldsymbol{v}_0^c]_\times^2\Big) \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w && (4.41)\\
=&(-\boldsymbol{n}_i^T \cdot [\boldsymbol{v}_0^c]_\times^2 \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w) \cdot \cos(\alpha) + (\boldsymbol{n}_i^T \cdot [\boldsymbol{v}_0^c]_\times \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w) \cdot \sin(\alpha)\\
&+ \boldsymbol{n}_i^T \cdot \Big(\mathbf{I} + [\boldsymbol{v}_0^c]_\times^2\Big) \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w && (4.42)\\
=&a_i \cdot \cos(\alpha) + b_i \cdot \sin(\alpha) + c_i && (4.43)
\end{aligned}$$

where $a_i = -\boldsymbol{n}_i^T \cdot [\boldsymbol{v}_0^c]_\times^2 \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w$; $b_i = \boldsymbol{n}_i^T \cdot [\boldsymbol{v}_0^c]_\times \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w$; $c_i = \boldsymbol{n}_i^T \cdot \Big(\mathbf{I} + [\boldsymbol{v}_0^c]_\times^2\Big) \cdot \mathbf{R}_w^c \boldsymbol{v}_i^w$. Therefore, given the known vertical direction, the rotation constraint from the i-th line correspondence is:

$$\boldsymbol{n}_i^T \mathbf{R} \boldsymbol{v}_i^w = a_i \cdot \cos(\alpha) + b_i \cdot \sin(\alpha) + c_i = 0 \tag{4.44}$$

4.4.2 Unit Constraint Solution

To address the unknown α, a linear solution is proposed in [150] and [156], where $q_1 = \cos(\alpha)$, $q_2 = \sin(\alpha)$, then for the i-th line correspondence:

$$a_i \cdot q_1 + b_i \cdot q_2 + c_i = 0 \tag{4.45}$$

Hence, q_1 and q_2 can be determined from the linear system. However, they may not satisfy the trigonometric constraint $q_1^2 + q_2^2 = 1$ [150, 156]. In addition, the cubic polynomial solution is proposed in [156, 173, 174]. Specifically, let $q = \tan(\alpha/2)$, then,

$$\cos(\alpha) = \frac{1-q^2}{1+q^2}, \quad \sin(\alpha) = \frac{2q}{1+q^2} \tag{4.46}$$

Therefore, for the i-th line correspondence:

$$a_i \frac{1-q^2}{1+q^2} + b_i \frac{2q}{1+q^2} + c_i = 0 \tag{4.47}$$

The objective is

$$\min_q \sum_{i=1}^{N} \big((c_i - a_i)q^2 + 2b_i q + a_i + c_i\big)^2 \tag{4.48}$$

The optimal solution should occur at its first derivative equal zero

$$\sum_{i=1}^{N} \big(a_i' q^3 + b_i' q^2 + c_i' q + d_i'\big) = 0 \tag{4.49}$$

where $a'_i = 4a_i^2 + 4c_i^2 - 8a_ic_i$; $b'_i = 12b_ic_i - 12a_ic_i$; $c'_i = -4a_i^2 + 8b_i^2 + 4c_i^2$; $d'_i = 4a_ib_i + 4b_ic_i$. Therefore, q can be solved from the third order polynomial system. However, when $\alpha = \pi$, $q \to +\infty$ and when $\alpha = -\pi$, $q \to -\infty$. This is a singularity-affected parameterization [40], which might reduce the accuracy near the singularity angle (i.e., $\pm\pi$).

In this thesis, we apply a singularity-free parameterization to increase the accuracy. Specifically, the objective is formulated as:

$$\min \sum_{i=1}^{N} (a_i \cdot q_1 + b_i \cdot q_2 + c_i)^2, \qquad \text{s.t. } q_1^2 + q_2^2 = 1 \tag{4.50}$$

In contrast to the linear solution, it does not drop the unit-norm constraint. Additionally, in contrast to the cubic polynomial solution, it does not suffer from any degeneracy.

Clearly, Eq. (4.50) is a typical *Equality Constrained Optimization* problem [179]. The Lagrangian formulation is

$$f = \sum_{i=1}^{N} (a_i \cdot q_1 + b_i \cdot q_2 + c_i)^2 + \lambda(q_1^2 + q_2^2 - 1) \tag{4.51}$$

where λ is a Lagrange multiplier. The first-order optimality condition of the Lagrangian formulation is

$$\begin{cases} f'_{q_1} = 2\sum_{i=1}^{N} a_i^2 q_1 + 2\sum_{i=1}^{N} a_i b_i q_2 + 2\sum_{i=1}^{N} a_i c_i + 2\lambda q_1 = 0 \\ f'_{q_2} = 2\sum_{i=1}^{N} b_i^2 q_2 + 2\sum_{i=1}^{N} a_i b_i q_1 + 2\sum_{i=1}^{N} b_i c_i + 2\lambda q_2 = 0 \\ f'_{\lambda} = q_1^2 + q_2^2 - 1 = 0 \end{cases} \tag{4.52}$$

$$\Rightarrow \begin{cases} \widetilde{a} q_1^2 + \widetilde{b} q_2^2 + \widetilde{c} q_1 q_2 + \widetilde{d} q_1 + \widetilde{e} q_2 = 0 \\ q_1^2 + q_2^2 - 1 = 0 \end{cases} \tag{4.53}$$

where $\widetilde{a} = \sum_{i=1}^{N} a_i b_i$; $\widetilde{b} = -\sum_{i=1}^{N} a_i b_i$; $\widetilde{c} = \sum_{i=1}^{N} (b_i^2 - a_i^2)$; $\widetilde{d} = \sum_{i=1}^{N} b_i c_i$; $\widetilde{e} = -\sum_{i=1}^{N} a_i c_i$. The original optimal solution (q_1, q_2) can be obtained by identifying all solutions of Eq. (4.53).

Specifically, solving Eq. (4.53) can be considered as a case of solving the intersection of two conic curves, which is comprehensively discussed by Prof. Jürgen Richter-Gebert in [180]. The core idea is constructing a new degenerate conic, which can be the two possibly coincident lines. The points of intersection between the new degenerate conic and either one of the two original conics will be the desired points.

However, in computer vision, *hidden variable resultant* is more popular [181,182]. We then introduce solving Eq. (4.53) by hidden variable resultant. Specifically, from Eq. (4.53), we can get

$$\begin{cases} \widetilde{a} q_1^2 + \widetilde{b} q_2^2 + \widetilde{c} q_1 q_2 + \widetilde{d} q_1 + \widetilde{e} q_2 = 0 \\ q_1^2 + q_2^2 - 1 = 0 \\ \widetilde{a} q_1^3 + \widetilde{b} q_1 q_2^2 + \widetilde{c} q_1^2 q_2 + \widetilde{d} q_1^2 + \widetilde{e} q_1 q_2 = 0 \\ q_1^3 + q_1 q_2^2 - q_1 = 0 \end{cases} \tag{4.54}$$

Then, we rewrite the equations,

$$\begin{bmatrix} 0 & \widetilde{a} & \widetilde{c}q_2+\widetilde{d} & \widetilde{b}q_2^2+\widetilde{e}q_2 \\ 0 & 1 & 0 & q_2^2-1 \\ \widetilde{a} & \widetilde{c}q_2+\widetilde{d} & \widetilde{b}q_2^2+\widetilde{e}q_2 & 0 \\ 1 & 0 & q_2^2-1 & 0 \end{bmatrix} \begin{bmatrix} q_1^3 \\ q_1^2 \\ q_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{4.55}$$

To obtain the non-trivial solutions, the determinant of the coefficient matrix should be zero.

$$\begin{vmatrix} 0 & \widetilde{a} & \widetilde{c}q_2+\widetilde{d} & \widetilde{b}q_2^2+\widetilde{e}q_2 \\ 0 & 1 & 0 & q_2^2-1 \\ \widetilde{a} & \widetilde{c}q_2+\widetilde{d} & \widetilde{b}q_2^2+\widetilde{e}q_2 & 0 \\ 1 & 0 & q_2^2-1 & 0 \end{vmatrix} = 0 \tag{4.56}$$

Then, we can get an univariate polynomial of the hidden variable

$$m_4q_2^4+m_3q_2^3+m_2q_2^2+m_1q_2+m_0=0 \tag{4.57}$$

where $m_4=(\widetilde{a}-\widetilde{b})^2+\widetilde{c}^2$; $m_3=2\widetilde{c}\widetilde{d}-2\widetilde{e}(\widetilde{a}-\widetilde{b})$; $m_2=-\widetilde{c}^2+\widetilde{d}^2+\widetilde{e}^2-2\widetilde{a}(\widetilde{a}-\widetilde{b})$; $m_1=2\widetilde{a}\widetilde{e}-2\widetilde{c}\widetilde{d}$; $m_0=\widetilde{a}^2-\widetilde{d}^2$. By solving the univariate polynomial system, we can obtain q_2, and we the solving q_1 by $q_1=\pm\sqrt{1-q_2^2}$.

4.4.3 Outlier-Contaminated Cases

In the outlier-contaminated cases, just one line correspondence is sufficient to be a minimal-subset solver in RANSAC [175]. For the i-th line correspondence, if the unit norm constraint is not ignored, then we have

$$\begin{aligned} a_i\cdot q_1+b_i\cdot q_2+c_i&=0 \\ q_1^2+q_2^2&=1 \end{aligned} \Rightarrow\ (a_i^2+b_i^2)q_1^2+2a_ic_iq_1+c_i^2-b_i^2=0 \tag{4.58}$$

We can obtain two solutions for α from Eq. (4.58). Similarly, two α solutions can also be obtained from Eq. (4.47) (singularity-affected parameterization).

As discussed above, RANSAC-type algorithms cannot guarantee the optimality of the solution. To obtain the certifiably optimal solution, we apply the BnB algorithm to obtain the optimal α from outlier-contaminated inputs. According to Eq. (4.6a) and Eq. (4.44), the robust objective can be formulated as

$$\max_{\alpha}\sum_{i=1}^{N}\mathbb{I}\left(|a_i\cdot\cos(\alpha)+b_i\cdot\sin(\alpha)+c_i|\le\epsilon\right) \tag{4.59}$$

Equivalently,

$$\max_{\alpha}\sum_{i=1}^{N}\mathbb{I}\left(\left|\sqrt{a_i^2+b_i^2}\cdot\sin(\alpha+\varphi_i)+c_i\right|\le\epsilon\right) \tag{4.60}$$

where $\varphi_i = \arctan 2(a_i, b_i)$[1].

Given a branch $[\underline{\alpha}, \overline{\alpha}]$, the upper bound and lower bound can be

$$\widetilde{e}_L = \sum_{i=1}^{N} \mathbb{I}\left(\left|\sqrt{a_i^2 + b_i^2} \cdot \sin(\hat{\alpha} + \varphi_i) + c_i\right| \leq \epsilon\right) \tag{4.61}$$

$$\widetilde{e}_U = \sum_{i=1}^{N} \mathbb{I}\left(\left|\sqrt{a_i^2 + b_i^2} \cdot \sin(\alpha_i^* + \varphi_i) + c_i\right| \leq \epsilon\right) \tag{4.62}$$

where $\hat{\alpha} = 0.5(\underline{\alpha} + \overline{\alpha})$ and by *Interval Analysis* (see [103])

$$\alpha_i^* = \arg\max_{\alpha} \mathbb{I}\left(\left|\sqrt{a_i^2 + b_i^2} \cdot \sin(\alpha + \varphi_i) + c_i\right| \leq \epsilon\right) \tag{4.63}$$

In addition, when the branch is becoming a point, $\widetilde{e}_U = \widetilde{e}_L$. Therefore, the proposed bounds are sufficient to be nested into BnB algorithm.

4.5 Translation Estimation

After obtaining the optimal camera rotation $\mathbf{R}$, solving the translation then becomes a linear model fitting problem [29]. Theoretically, the globally optimal translation can also be obtained by solving Eq. (4.6b) using the BnB algorithm [35]. However, the translation domain, which is different from the closed $\mathbb{SO}(3)$ structure, is not easily estimated correctly for various applications. Moreover, sequentially solving rotation and translation by two separate BnB methods does not mean that the obtained optimal rotation and translation are necessarily globally optimal for the combined problem [155]. In this thesis, the RANSAC algorithm[2] is applied to estimate translation.

4.6 Experiments

To investigate the performance of our algorithms, we compared them with several state-of-the-art PnL methods on both synthetic and real-world data. All experiments were run on a personal computer with an AMD Ryzen 7 2700X CPU and 32GB RAM.

4.6.1 Experimental Setup

All comparison methods are listed:

[1] arctan2$(\cdot,\cdot)$ is the four-quadrant inverse tangent function: `https://www.mathworks.com/help/matlab/ref/atan2.html`

[2] It was implemented by a Matlab built-in function: `https://www.mathworks.com/help/vision/ref/ransac.html`

- Ansar+MLESAC4+RPnL: Ansar PnL algorithm [170] is nested into MLESAC [183] and the final solution is computed by RPnL [184].
- Mirzaei+MLESAC3: Mirzaei PnL algorithm [176] is nested inside MLESAC.
- RPnL+MLESAC4: RPnL algorithm [184] is nested inside MLESAC.
- P3L+RANSAC3: A P3L algorithm [59] is plugged into RANSAC.
- ASPnL+RANSAC4: ASPnLalgorithm [59] is plugged into RANSAC.
- DLT-Lines+AOR: DLT(Direct Linear Transformation)-Lines PnL algorithm [151].
- LPnL_Bar_LS+AOR: As proposed in [59], where the lines are parameterized with the barycentric coordinates.
- LPnL_Bar_ENull+AOR: As proposed by Xu et at. in [59].
- DLT-Pücker-Lines+AOR: It applies Pücker coordinates [151].
- DLT-Combined-Lines+AOR: It is a combination of DLT-Lines and DLT-Plücker-Lines [151].
- Ro_PnL: Our proposed Robust PnL algorithm, which first applies the BnB algorithm to obtain rotation then uses RANSAC to estimate translation.
- Cubic-solution: This is an outlier-free known-vertical-direction PnL algorithm in [156] and cubic-ransac is its outlier-robust version.
- Unit-solution: Our proposed singularity-free non-minimal PnL algorithm with a known vertical direction. Unit-ransac is its outlier-robust version, which is also proposed in [175].
- vBnB: This method first estimates rotation with a known vertical direction using our proposed non-RANSAC globally optimal algorithm, then estimates translation by RANSAC.

The number at the end of MLESAC/RANSAC denotes the number of line correspondences used to generate hypotheses and the maximal number of random trials is limited to 10,000. +AOR means the linear algorithm is combined with AOR [151, 172]. The inlier threshold ϵ was set to 1° in RANSAC-type methods and BnB-based methods. All the compared codes[3] and our codes were executed on Matlab2019a. To demonstrate the accuracy and robustness, the translation/position error is defined as: $e_{trans} = ||\boldsymbol{t}_{gt} - \boldsymbol{t}^*||$ where $\boldsymbol{t}_{gt}$ is ground

[3] `http://www.fit.vutbr.cz/~ipribyl/DLT-based-PnL/`

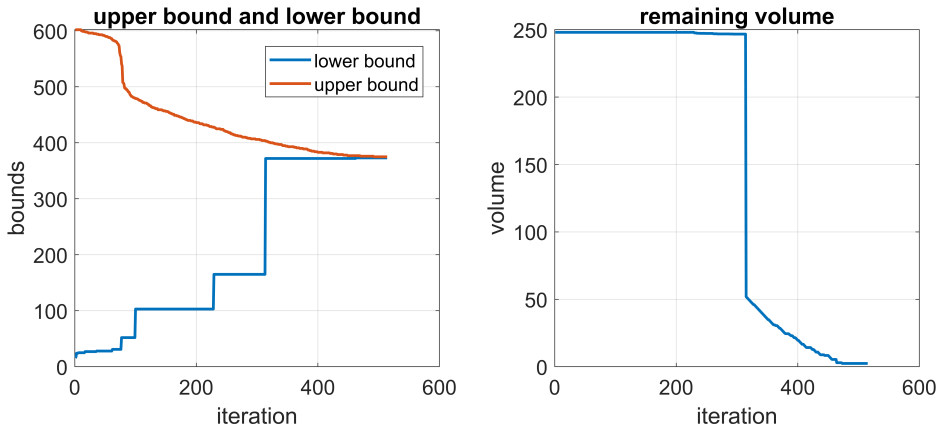


Figure 4.4: Convergence curves of our proposed globally optimal camera orientation estimation.

truth; $\boldsymbol{t}^*$ is the estimated translation. The rotation/orientation error is defined as the angle between ground truth $\mathbf{R}_{gt}$ and estimated rotation $\mathbf{R}^*$:

$$e_{rot} = \arccos\left(\frac{Tr(\mathbf{R}_{gt}^{-1}\mathbf{R}^*) - 1}{2}\right) \tag{4.64}$$

To compare the efficiency, the median runtime for 500 trials is recorded for each setting.

4.6.2 Synthetic Data Experiments

Data generation. N line segments were created randomly using $2N$ random endpoints, which were distributed in a cube $[100m \times 100m \times 100m]^3$. A virtual pinhole camera was randomly placed in the scene facing towards to the line segments. The camera is simulated using a 640×480 pixels image, 800 pixels focal length. To simulate noise, the 2D endpoints were perturbed with Gaussian noise with a standard deviation of $\sigma = 1$ pixel. The outliers were simulated by adding Gaussian noise with a very large standard deviation ($\sigma = 100$ pixels). We first showed the convergence process of our proposed method (see Fig. 4.4) to validate the global optimal solution. From the results, the proposed method can converge to the optimal solution in hundreds step with the zero-gap between lower and upper bounds.

Outlier-contaminated inputs. We then tested our proposed method on synthetic data with different outlier rates. $N = 200$ and the outlier rates were $\{0.1, \cdots, 0.7\}$. To demonstrate the guaranteed optimality of our method, we define the success rate: N^+/N, where N is the total number of inputs; N^+ is the number of successful cases that satisfy $e_{rot} < 5^\circ$ and $e_{trans} < 2m$. The results are shown in Fig. 4.5.

We can draw the following two conclusions from the results:

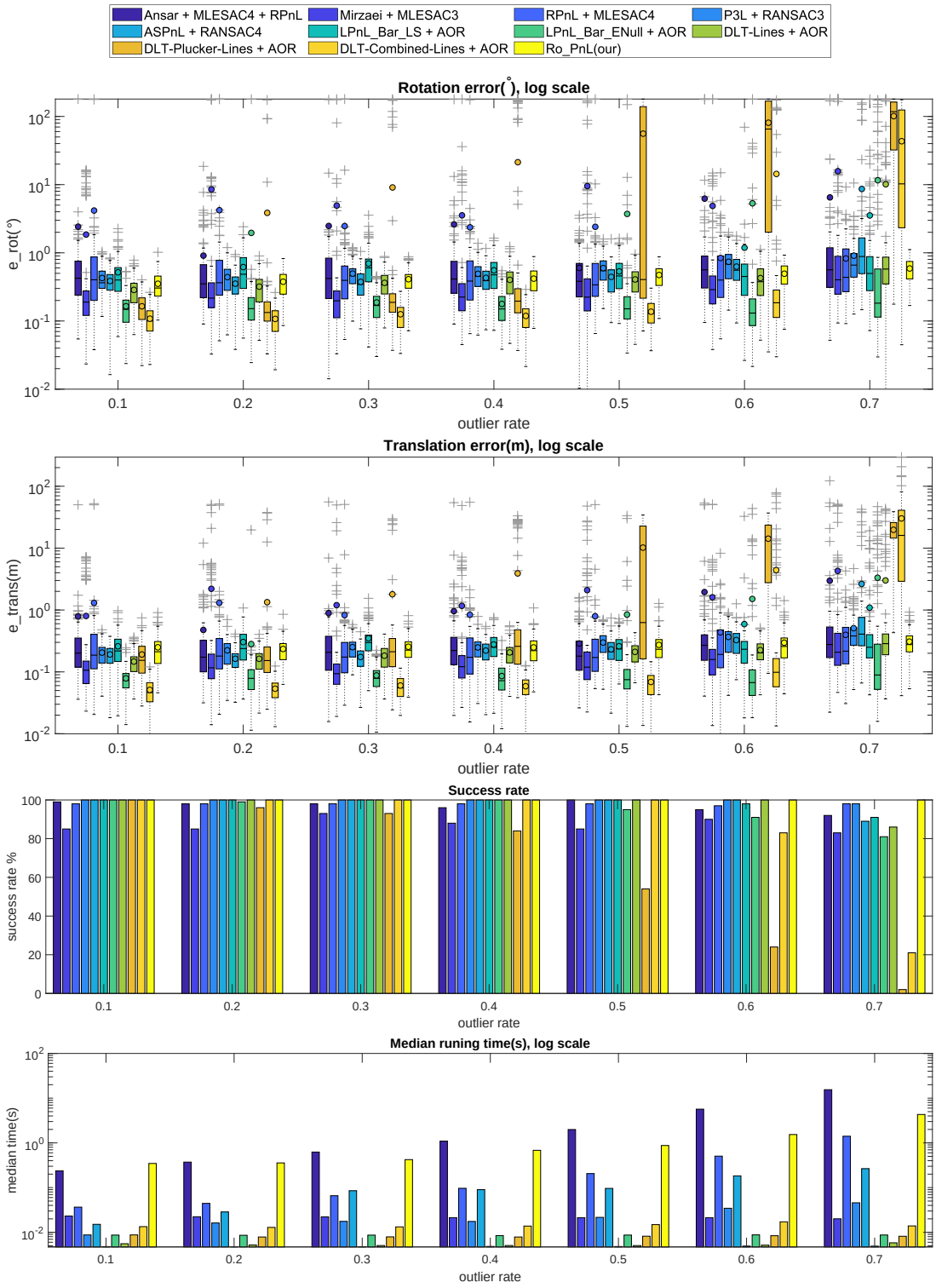


Figure 4.5: Controlled experiments on synthetic data with different outlier rates. Top subfigure: rotation error e_{rot}(deg). Second subfigure: translation error e_{trans}(m). Third subfigure: success rate. Bottom subfigure: median runtime (second).

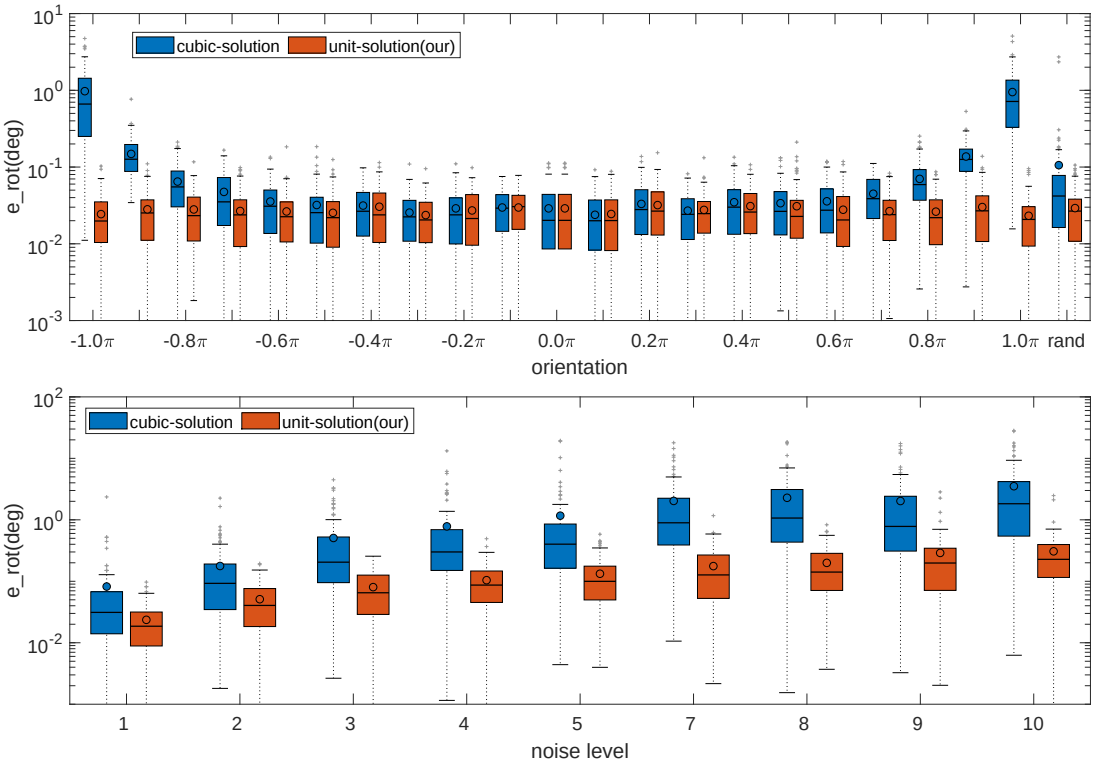


Figure 4.6: Comparison of experiments using cubic-solution and unit-solution. Top subfigure: noise level was set to 1 pixel and different camera orientations were tested. Bottom subfigure: random camera orientations were tested at different noise levels.

1. With the outlier rate increasing, the existing methods might return more unsatisfactory solutions. Conversely, our proposed methods (i.e., Ro_PnL) can obtain the globally optimal camera rotation, which was superior to other existing PnL methods. In addition, obtaining optimal rotation is helpful in estimating camera translation. Therefore, our method can always provide the maximum success rate.

2. Although our proposed method returns very robust solutions, it had a longer running time than most existing methods except for the Ansar algorithm [170], which has $O(N^2)$ computational complexity.

Outlier-free inputs with a known vertical direction. First we present the comparison between unit-solution and cubic-solution to confirm that our proposed singularity-free parameterization leads to superior accuracy. $N = 200$ and no outliers. Different camera orientations ($\alpha = \{-\pi, \cdots, \pi\}$) and random orientations were also tested with noise level $\delta = 1$. In addition, we also compared these two methods at different noise levels ($\{1, \cdots, 10\}$) with random camera orientations. The results are shown in Fig. 4.6.

From the results (top subfigure in Fig. 4.6), when the camera orientation is near to the singularity angle (i.e., $\pm\pi$), our proposed unit-solution was clearly

able to achieve better results, and therefore, our proposed method could obtain superior accuracy in random cases. In addition, the results (bottom subfigure in Fig. 4.6) show that the unit-solution is more accurate than the cubic-solution at different noise levels.

Outlier-contaminated inputs with a known vertical direction. We then compared our proposed vBnB method with RANSAC-type methods in outlier-corrupted data with different outlier rates ($\{0.1, \cdots, 0.8\}$). The results are shown in Fig. 4.8. Compared with RANSAC-type methods, our proposed BnB method can obtain a globally optimal camera rotation. Besides, unit-ransac obtained slightly better accuracy than cubic-solution. This is because cubic-ransac applied the singularity-affect parameterization. For efficiency, our proposed vBnB was slower than other RANSAC-type methods in low outlier rate cases, however, it was faster in high outlier rate cases, which was consistent with a conclusion that BnB-based method could be more efficient than RANSAC-type methods in high outlier rate cases [146].

Outlier-contaminated inputs with a biased vertical direction. To simulate the cases that the vertical direction was obtained inaccurately, we also tested our proposed vBnB method using synthetic data when the vertical direction was given with a bias. The outlier rate was $\{0.1, \cdots, 0.8\}$ and the vertical direction was biased by $\{0.1^\circ, 0.5^\circ, 1^\circ\}$. The results are illustrated in Fig. 4.9. It shows that due to the biased vertical direction, the final accuracy will also be biased, however, vBnB is still robust to outliers.

4.6.3 Real-Data Experiments

In this section, we investigated the performance on the real data from VGG Multiview Dataset[4] (see Fig. 4.7). The VGG dataset contained image sequences, line segments, true correspondences and a ground-truth projection matrix. Outliers were randomly added to the original data, and for each scene the outlier rate was 0.2. To emphasize the global optimality, maximum rotation error, maximum translation error and average runtime were recorded for each method.

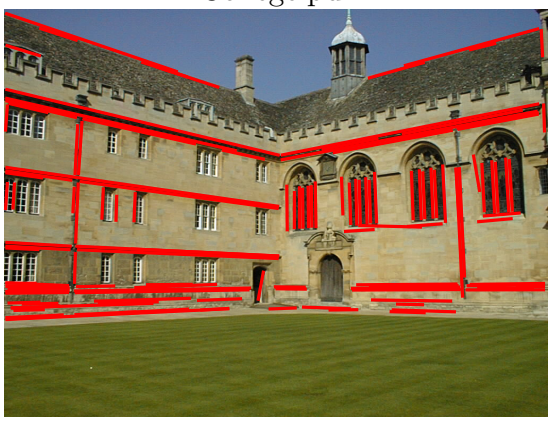
The results are listed in Table 4.1. Our proposed algorithm could obtain the right poses ($e_{rot} < 2^\circ$ and $e_{trans} < 1m$) in all scenes. Conversely, other PnL algorithms could not provide all satisfactory solutions. On the other hand, to obtain robust solutions, our algorithm had longer runtime than the other existing methods.

Next, we assumed that the vertical direction was known in each scene and compared our vBnB method and RANSAC-type methods. Random outliers were also added to the clean data and outlier rate was set to 0.2. We then recorded the maximum errors and average runtime in experiments. The results are present in Table. 4.2.

Our proposed vBnB method can obtain satisfactory solutions in all VGG data. However, RANSAC-type methods may return incorrect solutions occasionally.

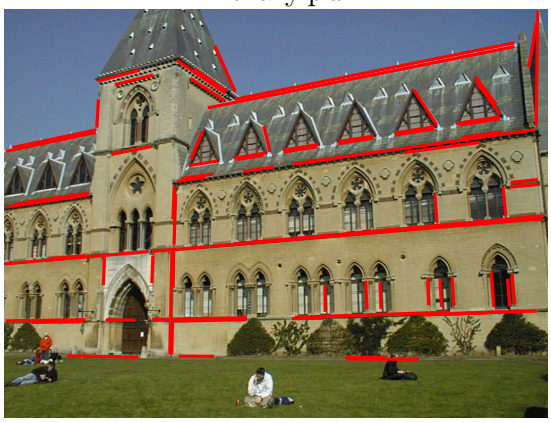
[4] `https://www.robots.ox.ac.uk/~vgg/data/mview/`

College.pdf



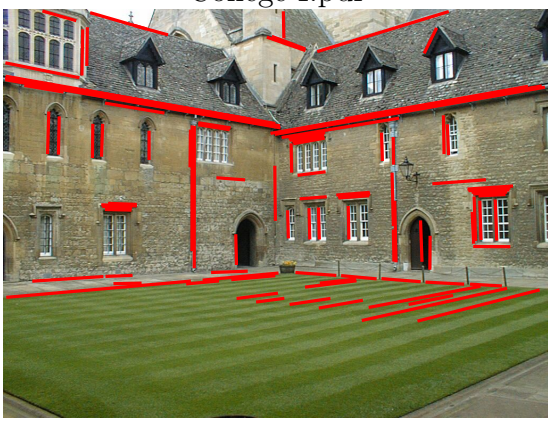
(a) Wadham College

Library.pdf



(b) University Library

College I.pdf



(c) Merton College I

College II.pdf

(d) Merton College II

College III.pdf

(e) Merton College III

(f) Corridor

Figure 4.7: Selected image data from VGG dataset. The red lines denote the line features exacting from the scenes.

Table 4.1: Experiments with real VGG data.

Methods	max $e_{rot}(^\circ)$	max $e_{trans}(m)$	avg. time (s)
Ansar + MLESAC4 + RPnL	179.12	46.90	0.719
Mirzaei + MLESAC3	178.85	46.84	0.025
RPnL + MLESAC4	177.95	32.97	0.067
P3L + RANSAC3	10.42	2.31	0.014
ASPnL + RANSAC4	2.25	1.11	0.024
LPnL_Bar_LS + AOR	178.21	32.43	0.010
LPnL_Bar_ENull + AOR	2.55	16.72	0.029
DLT-Lines + AOR	174.02	159.97	**0.008**
DLT-Plucker-Lines + AOR	179.40	122.61	0.012
DLT-Combined-Lines + AOR	178.11	2439.63	0.020
Ro_PnL (**Our**)	**1.56**	**0.84**	5.825

Moreover, the runtime of all methods in Table. 4.2 were usually faster than those in Table. 4.1. This is because with the help of the prior vertical direction, the rotation domain is reduced from three to one dimension, and thus the BnB-based algorithm is much faster [177].

4.7 Conclusion

To provide a certifiably optimal solution of PnL problem in some safety-critical applications, we propose globally optimal solutions to the camera orientation problem. The BnB algorithm is applied to search for the optimal rotation. In addition, if the vertical direction is known by other means, we first propose a novel non-minimal outlier-free PnL algorithm, which applies singular-free parameterization and thus achieves improved accuracy. Furthermore, for outlier-contaminated inputs, we propose a non-RANSAC and globally optimal algorithm to estimate camera orientation with a known vertical direction. Experiments on synthetic and real-world data all have demonstrated that our proposed methods are more robust than several existing PnL methods.

Table 4.2: Experiments with real VGG data with known vertical direction.

Methods	max $e_{rot}(^\circ)$	max $e_{trans}(m)$	avg. time (ms)
Cubic-ransac	57.61	43.61	3.03
Unit-ransac	11.61	3.79	**2.65**
vBnB (**Our**)	**1.51**	**0.53**	3.74

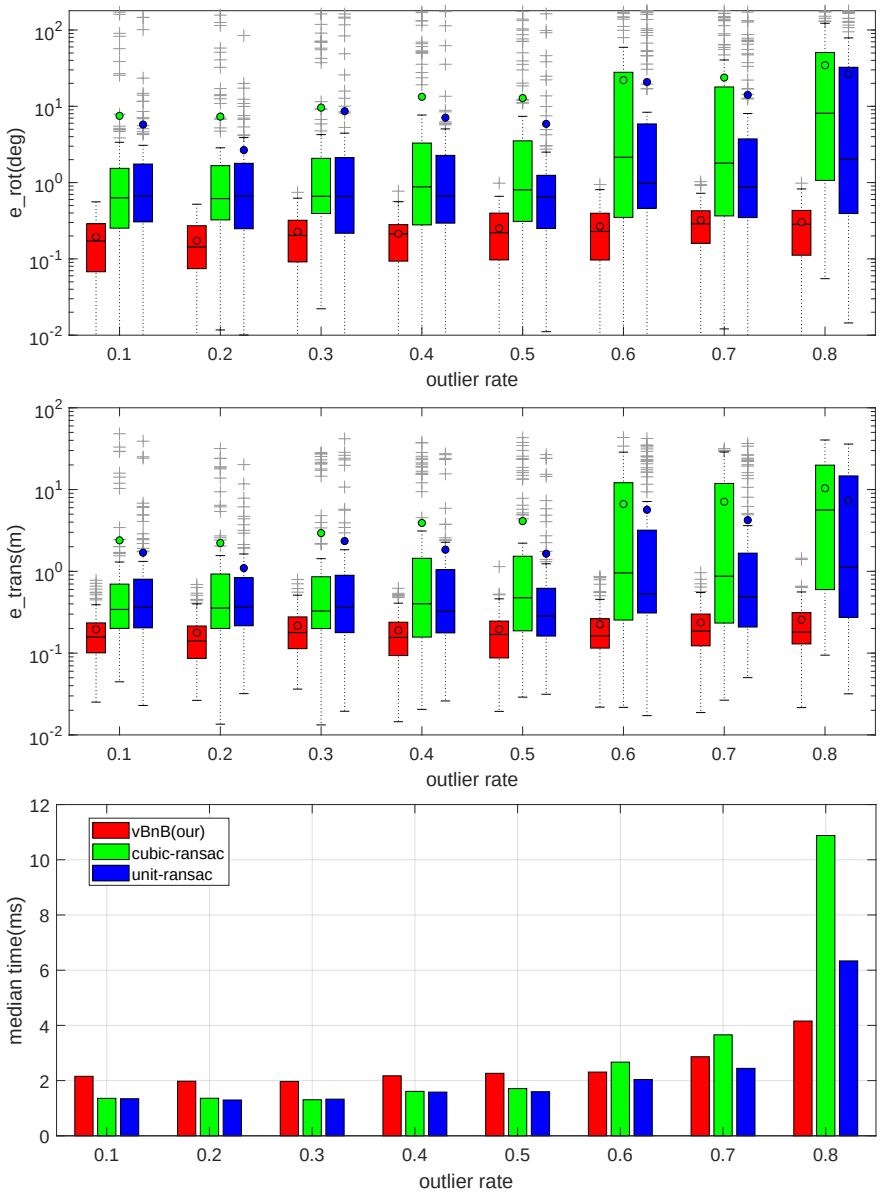


Figure 4.8: Comparison of RANSAC and vBnB algorithm experiments. Top subfigure shows the rotation error. Second subfigure shows the translation error. Bottom subfigure shows the median runtime.

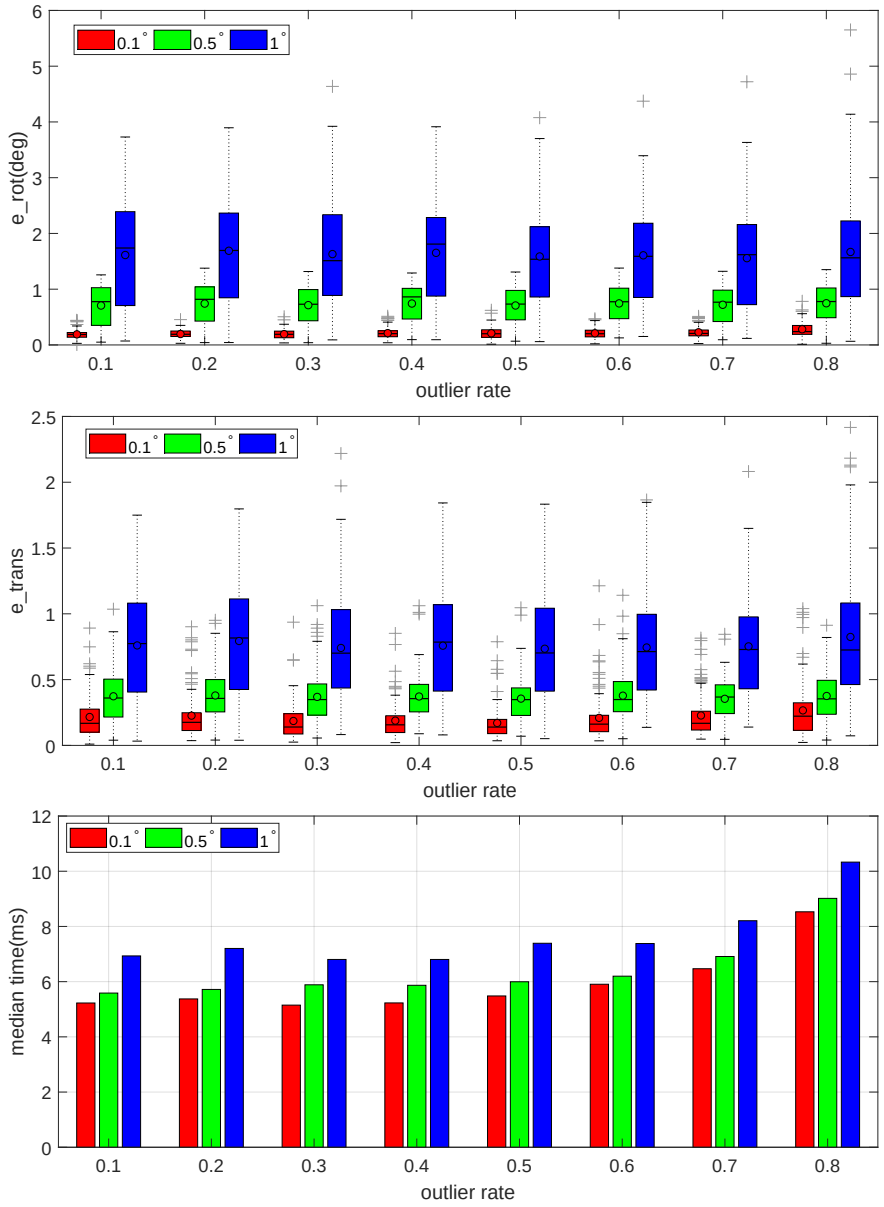


Figure 4.9: Controlled experiments with different vertical direction biases. Top subfigure shows the rotation error. Second subfigure shows the translation error. Bottom subfigure shows the median runtime

Chapter 5

Globally Optimal Solution for Relative Pose Estimation

This chapter explores the geometry of relative pose estimation with known vertical (gravity) direction and provide a globally optimal solution. The core idea is applying the inequality in unit sphere and using the nested BnB framework to obtain the global optimum. This chapter is a modified version of

- Liu, Yinlong, Guang Chen, Rongqi Gu, and Alois Knoll. *Globally Optimal Consensus Maximization for Relative Pose Estimation With Known Gravity Direction*, in IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 5905-5912, July 2021

5.1 Background

The task of relative pose estimation is estimating the relative camera pose from matching correspondences of two frames, which is also known as essential matrix estimation [27]. It is one of core tasks in computer vision [11], thereby it is the basis of many high-level applications (e.g., visual odometry, structure from motion and 3D reconstruction) [10, 185, 186].

In addition to cameras, modern applications, such as autonomous driving and robot navigation and localization, are usually equipped with many other sensors (e.g., GPS, inertial measurement units (IMUs)) [187, 188]. Therefore, we can obtain prior knowledge to help estimate the relative camera pose [189, 190].

In this chapter, we focus on estimating relative pose with the prior known gravity direction [191]. The gravity direction can be usually provided by IMUs, and the accuracy of this direction is usually reliable even by low cost IMUs (typically error $< 0.5°$) [174, 192]. Taking advantages of knowing gravity direction, the degrees-of-freedom of relative pose estimation problem reduce to three, which means three point correspondences instead of five [27] are sufficient to minimally estimate relative pose [191].

Unfortunately, the mismatches (a.k.a. outliers), which will impair the accuracy significantly [29], are unavoidable in real applications [189, 191]. To suppress the outliers, the defacto mechanism is *RANdom SAmple Consensus* (RANSAC) [34].

However, RANSAC is a non-deterministic algorithm, which means it cannot provide a correct solution with a provable guarantee [57]. More specifically, RANSAC produces a reasonable result only with a certain probability [29]. On the other hand, there are many safety-critical applications, which highly demand such algorithms that can return a extremely reliable solution in the presence of noise and outliers [30]. Obviously, RANSAC cannot meet this strict demand.

To provide a provably optimal solution for some safety-critical applications, we propose a globally-optimal solution, which applies a novel nested branch-and-bound (BnB) algorithm. The main contributions of this paper are as follows:

- In contrast to RANSAC, the proposed relative pose estimation method can obtain the globally optimal solution with a provable guarantee, which means it can meet the strict demand in many safety-critical applications.
- A nested BnB algorithm with novel geometric bounds is proposed. More specifically, a new geometric formulation and its essential geometric relationship are explored.

5.2 Related Work

We first review the outlier-free solutions for relative pose estimation with known gravity direction. In fact, the outlier-free solutions are well studied [189, 191, 193, 194]. They focus on finding solutions to algebraic systems. Specifically, closed-form solutions for sightly different formulations are explored in [189] and [193], respectively. Furthermore, Sweeney *et al.* point out that solving for relative pose with known direction is a quadratic eigenvalue problem [191], and therefore, they propose a simple and extremely efficient algorithm. More recently, given prior knowledge of gravity direction, Ding *et al.* go deeper to explore minimal solutions to relative pose estimation problem with unknown focal length in [195]. Moreover, a non-minimal ($N \geq 4$) solution for relative pose estimation with gravity prior is explored in [196].

If the inputs are contaminated by outliers, outlier-free algorithms should be nested into RANSAC scheme, which is the de facto mechanism [189]. However, their solutions are sub-optimal due to the obvious heuristic nature of RANSAC [29]. To assure the global optimality of the optimal solution, Yang *et al.* [135] propose a globally optimal solution to essential matrix estimation. Specifically, they adopt the consensus set maximization as the objective and the branch-and-bound algorithm to systematically search for the global optimum. However, due to rather high dimensionality of the solution space (i.e., five degrees of freedom), Yang's BnB algorithm tends to be very slow [197]. Furthermore, Fredriksson *et at.* propose a globally optimal method for a more difficult case in which the correspondences are unknown [198]. In detail, they apply an efficient branch and bound technique in combination with bipartite matching to solve the correspondence problem. However, their method becomes intractable in the cases where outlier rate is considerably high and the number of input is considerably large. Besides, with

the prior knowlege of full camera orientation, Fredriksson *et at.* explore globally-optimal methods for two-view translation estimation in [199] and [200]. Recently, under the plane-based Ackermann steering motion assumption, a globally optimal and correspondence-less solution is explored in [201].

In addition, the epipolar constraint can be linearized to enable the two-view relative pose to be estimated linearly [11]. Accordingly, many globally optimal algorithms for linear consensus maximization are proposed to solve the relative estimation problem. Li is one of pioneers to explore the globally-optimal solution for linearized relative pose estimation [35]. Furthermore, tree search method is proposed to find the global optimum in [202], and more recently tree search method is accelerated significantly in [162]. Nevertheless, because linearization will drop some constraints, the solution of linearized objective is not necessarily the same as the original solution.

It is worth mentioning that many recent studies focus on solving special pose estimation problems with the help of gravity direction. For example, homography-based minimal-case relative pose estimation with known gravity direction is thoroughly explored in [203]. Furthermore, minimal solutions for relative pose with a single affine correspondence and known vertical direction is studied in [204]. Besides, the prior knowledge of gravity direction is helpful for solving absolute pose estimation problem [188, 205] and PnL (Perspective-n-Line) problem [150, 206]. These gravity-known works also inspire our work.

5.3 Problem Formulation

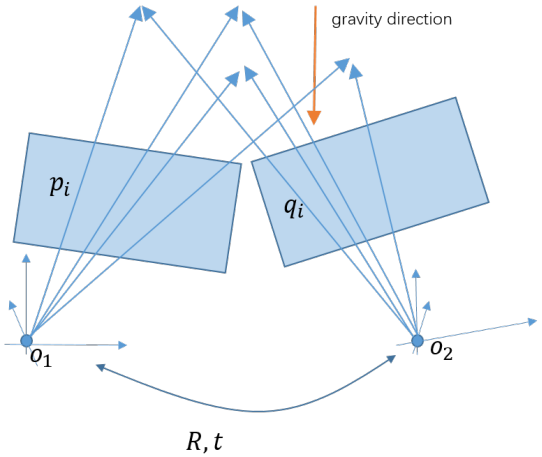


Figure 5.1: The geometric of two-view relative pose estimation with the known gravity direction.

Given N point correspondences $\{\boldsymbol{p}_i, \boldsymbol{q}_i\}_{i=1}^{N}$, which may be contaminated by outliers, the target is estimating the relative pose (i.e., rotation $\mathbf{R}$ and translation $\boldsymbol{t}$). We start with the epipolar geometry [11] (see Fig. 5.1).

$$\mathbf{R}\lambda_i^1 \boldsymbol{p}_i + \boldsymbol{t} = \lambda_i^2 \boldsymbol{q}_i, i = 1 \dots N \tag{5.1}$$

where λ_i^1 and λ_i^2 are two different projective scales. Furthermore, Eq. (5.1) can be reformulated [191] as

$$\boldsymbol{t}^T (\boldsymbol{q}_i \times \mathbf{R}\boldsymbol{p}_i) = 0, i = 1 \cdots N \tag{5.2}$$

where $\times$ is cross product. It should mention that the scaling of $\boldsymbol{t}$ cannot be determined by Eq. (5.2) and we set $\|\boldsymbol{t}\| = 1$.

In addition, given the known unit gravity direction $\boldsymbol{g}_1$ in left camera coordinate and $\boldsymbol{g}_2$ in right camera coordinate, we have

$$\boldsymbol{g}_2 = \mathbf{R}\boldsymbol{g}_1 \tag{5.3}$$

The solution of Eq. (5.3) is

$$\mathbf{R} = \mathbf{R}(\theta, \boldsymbol{g}_2) \cdot \mathbf{R}_{\boldsymbol{g}_1}^{\boldsymbol{g}_2} \tag{5.4}$$

where $\mathbf{R}_{\boldsymbol{g}_1}^{\boldsymbol{g}_2}$ is the rotation that maps $\boldsymbol{g}_1$ to $\boldsymbol{g}_2$ with the minimum geodesic motion. $\mathbf{R}(\theta, \boldsymbol{g}_2)$ is the rotation that rotates θ about the axis $\boldsymbol{g}_2$ and θ is the unknown-but-sought variable. Please refer to [207] for more details about the solution. Furthermore, according to Rodrigues' rotation formula [11],

$$\begin{aligned} \mathbf{R}(\theta, \boldsymbol{g}_2) &= \exp\left(\theta[\boldsymbol{g}_2]_\times\right) && (5.5)\\ &= \mathbf{I} + \sin(\theta)\,[\boldsymbol{g}_2]_\times + (1 - \cos(\theta))\,[\boldsymbol{g}_2]_\times^2 && (5.6) \end{aligned}$$

where $[\boldsymbol{g}_2]_\times$ is the cross-product matrix for $\boldsymbol{g}_2$.

For the i-th correspondence, with the help of gravity direction,

$$\begin{aligned} \boldsymbol{q}_i \times \mathbf{R}\boldsymbol{p}_i &= [\boldsymbol{q}_i]_\times \left(\mathbf{I} + \sin(\theta)\,[\boldsymbol{g}_2]_\times + (1 - \cos(\theta))\,[\boldsymbol{g}_2]_\times^2\right) \mathbf{R}_{\boldsymbol{g}_1}^{\boldsymbol{g}_2} \boldsymbol{p}_i && (5.7)\\ &= \boldsymbol{a}_i + \sin(\theta)\,\boldsymbol{b}_i + \cos(\theta)\,\boldsymbol{c}_i && (5.8) \end{aligned}$$

where $\boldsymbol{a}_i = [\boldsymbol{q}_i]_\times \left(\mathbf{I} + [\boldsymbol{g}_2]_\times^2\right) \mathbf{R}_{\boldsymbol{g}_1}^{\boldsymbol{g}_2} \boldsymbol{p}_i$, $\boldsymbol{b}_i = [\boldsymbol{q}_i]_\times [\boldsymbol{g}_2]_\times \mathbf{R}_{\boldsymbol{g}_1}^{\boldsymbol{g}_2} \boldsymbol{p}_i$ and $\boldsymbol{c}_i = -[\boldsymbol{q}_i]_\times [\boldsymbol{g}_2]_\times^2 \mathbf{R}_{\boldsymbol{g}_1}^{\boldsymbol{g}_2} \boldsymbol{p}_i$. Therefore, the epipolar constraint becomes

$$\begin{aligned} &\boldsymbol{t}^T (\boldsymbol{q}_i \times \mathbf{R}\boldsymbol{p}_i) = 0 && (5.9)\\ \Rightarrow &\boldsymbol{t}^T (\boldsymbol{a}_i + \sin(\theta)\,\boldsymbol{b}_i + \cos(\theta)\,\boldsymbol{c}_i) = 0 && (5.10) \end{aligned}$$

Traditionally, if we define $s = \tan\left(\frac{\theta}{2}\right)$, then

$$\begin{aligned} &\boldsymbol{t}^T \left(\boldsymbol{a}_i + \frac{2s}{1+s^2}\boldsymbol{b}_i + \frac{1-s^2}{1+s^2}\boldsymbol{c}_i\right) = 0 && (5.11)\\ \Rightarrow &\boldsymbol{t}^T \left(\left(1+s^2\right)\boldsymbol{a}_i + 2s\boldsymbol{b}_i + \left(1-s^2\right)\boldsymbol{c}_i\right) = 0 && (5.12)\\ \Rightarrow &\boldsymbol{t}^T \left(\left(\boldsymbol{a}_i - \boldsymbol{c}_i\right)s^2 + 2\boldsymbol{b}_i s + \boldsymbol{a}_i + \boldsymbol{c}_i\right) = 0 && (5.13) \end{aligned}$$

Consequently, the minimal outlier-free relative pose estimation problem can be converted to a quadratic eigenvalue problem [191]. It should be mentioned that

this formulation applies the singularity-affected parameterization, which means the accuracy will decease near the singularity angle (i.e., $\theta = \pm\pi$) [157, 195]. Nonetheless, the authors in [195] point that this singularity problem can almost be ignored in real applications.

However, in real application, it is almost impossible to estimate relative pose with clean and perfect point correspondences. The optimal solution should be obtained from the inlier observations, which should satisfy

$$\left|\boldsymbol{t}^{*T}\left(\boldsymbol{a}_i + \sin(\theta^*)\boldsymbol{b}_i + \cos(\theta^*)\boldsymbol{c}_i\right)\right| \leq \epsilon \tag{5.14}$$

where $\boldsymbol{t}^*$ and θ^* are the optimal solution; ϵ is the inlier threshold. Therefore, a robust objective, which can suppress the outliers, should be formulated by utilizing the idea of inlier consensus maximization [29].

$$\max_{\boldsymbol{t},\theta} \sum_{i=1}^{N} \mathbb{I}\left(\left|\boldsymbol{t}^{T}\left(\boldsymbol{a}_i + \sin(\theta)\boldsymbol{b}_i + \cos(\theta)\boldsymbol{c}_i\right)\right| \leq \epsilon\right) \tag{5.15}$$

where $\mathbb{I}(\cdot)$ is the indicator function which returns 1 if the condition $\cdot$ is true and returns 0 if condition $\cdot$ is not true.

To suppress the outliers, the robust objective is formulated. However, the objective is obviously non-smooth and non-concave which means some traditional optimizer (e.g., gradient descent) is infeasible and some (e.g., [163]) might be trapped in local optimum. Furthermore, many safety-critical applications need to obtain the globally optimal solution. To meet this demand, we apply the brand-and-bound algorithm, which is the most commonly used mechanism for solving NP-hard optimization problems [96], to seek the maximum value of the objective with provable guarantee.

5.4 Branch-and-bound

The BnB algorithm systematically explores the whole domain of the candidate solutions to find the optimal solution, thereby its solution is globally-optimal. More specifically, the BnB algorithm recursively branches the solution domain, and the sub-branches are checked against upper and lower estimated bounds on the optimal solution. If the branch cannot produce a better solution than the best one found so far by the algorithm, then it is discarded, and consequently, the solution domain is reduced. The iterative process of branching, bounding and cutting is terminated when the optimal solution is found.

5.4.1 Bounds estimation

Obviously, the key of the BnB algorithm is how to estimate the upper and lower bounds in a given sub-branch. Accordingly, the solution domain should be parameterized properly. We note that the solution domain is $\boldsymbol{t} \in \mathbb{S}^2$ and $\theta \in [-\pi, \pi]$. In addition, if $\boldsymbol{t}$ is the optimal solution then $-\boldsymbol{t}$ is also the optimal

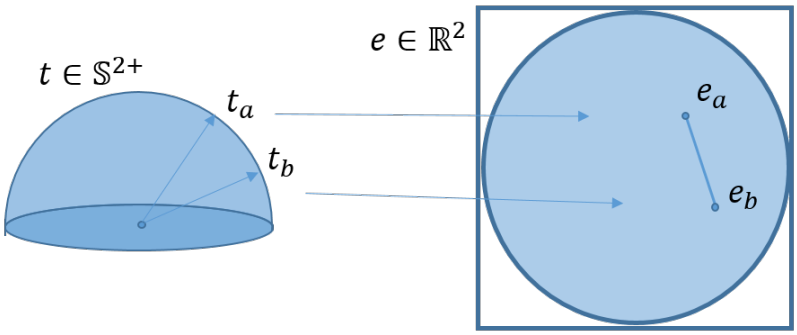


Figure 5.2: The exponential mapping. Geometrically, the hemisphere is flattened into a disk in a plane, and we have $\angle(\boldsymbol{t}_a, \boldsymbol{t}_b) \leq \|\boldsymbol{e}_a - \boldsymbol{e}_b\|$.

solution, therefore, we set the domain of $\boldsymbol{t}$ a hemisphere $\mathbb{S}^{2+}$ instead of a full sphere. $\mathbb{S}^{2+}$ is defined as

$$\mathbb{S}^{2+} = \{\boldsymbol{t} | t_3 \geq 0\} \tag{5.16}$$

where $\boldsymbol{t} = [t_1, t_2, t_3]^T$ is a unit vector in $\mathbb{R}^3$. To minimally parameterize the translation domain, we employ the exponential mapping method to map the hemisphere to a two dimensional disk [81]. More Specifically, given $\boldsymbol{t} \in \mathbb{S}^{2+}$, we can represent it by $\boldsymbol{e} \in \mathbb{R}^2$ (see Fig. 5.2).

$$\boldsymbol{t} = [\sin(\alpha)\cos(\beta), \sin(\alpha)\sin(\beta), \cos(\alpha)]^T \tag{5.17}$$

$$\boldsymbol{e} = \alpha[\cos(\beta), \sin(\beta)]^T \tag{5.18}$$

where $\alpha \in [0, \pi/2]$ and $\beta \in [-\pi, \pi]$. Consequently, we can apply the two dimensional disk to represent the solution domain of translation. Furthermore, for ease of manipulation, a circumscribed square of the solid disk is initialized as the translation solution domain in the BnB algorithm. Therefore, to find the optimal translation, the square-shaped branch will be subdivided into four sub-branches and we need to estimate the upper and lower bounds in these square-shaped sub-branches.

To derive the upper bound, according to the lemma 2.6 in chapter 2, then we have

Lemma. *Given $\boldsymbol{t}_a, \boldsymbol{t}_b \in \mathbb{S}^{2+}$, $\boldsymbol{e}_a$ and $\boldsymbol{e}_b$ are their corresponding points in the 2D disk. Then*

$$\angle(\boldsymbol{t}_a, \boldsymbol{t}_b) \leq \|\boldsymbol{e}_a - \boldsymbol{e}_b\| \tag{5.19}$$

In the BnB algorithm, let $\boldsymbol{e} \in \mathbb{B}_t$, where $\mathbb{B}_t$ is a translation branch. $\boldsymbol{e}_c$ is the center of the square-shaped branch and δ is the half-side length. Then we have

$$\angle(\boldsymbol{t}, \boldsymbol{t}_c) \leq \|\boldsymbol{e} - \boldsymbol{e}_c\| \leq \sqrt{2}\delta \tag{5.20}$$

where $\boldsymbol{t}$ and $\boldsymbol{t}_c$ correspond to $\boldsymbol{e}$ and $\boldsymbol{e}_c$, respectively.

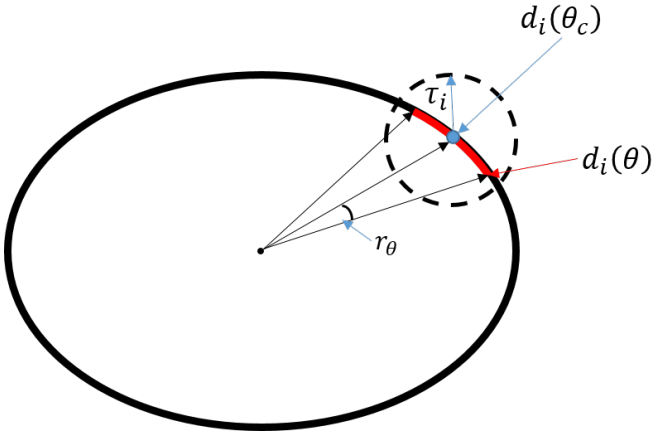


Figure 5.3: Illustration of the bound of $\boldsymbol{d}_i(\theta)$

Furthermore, we denote $\boldsymbol{d}_i(\theta) \triangleq \boldsymbol{a}_i + \sin(\theta)\,\boldsymbol{b}_i + \cos(\theta)\,\boldsymbol{c}_i$, then the objective function is

$$\max_{\boldsymbol{t},\theta} \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{t}^T \boldsymbol{d}_i(\theta)| \leq \epsilon\right) \tag{5.21}$$

Geometrically, $\boldsymbol{d}_i(\theta), \theta \in [-\pi, \pi]$ is an full ellipse. Let $\theta \in \mathbb{B}_\theta$ where $\mathbb{B}_\theta$ is an angle branch. θ_c is the center of the branch, and r_θ is the half length (radius) of the branch. Consequently, when $\theta \in \mathbb{B}_\theta$, $\boldsymbol{d}_i(\theta)$ is a curve of the full ellipse in 3D space, see Fig. 5.3.

Theorem (Mean value theorem for vector-valued functions)**.** *For a continuous vector-valued function $\boldsymbol{f} : [a, b] \to \mathbb{R}^n$, $\boldsymbol{f} : [a, b] \to \mathbb{R}^n$ differentiable on (a, b), there exists $x \in (a, b)$ such that [208–210]*

$$\|\boldsymbol{f}(b) - \boldsymbol{f}(a)\| \leq |b - a| \|\boldsymbol{f}'(x)\| \tag{5.22}$$

Inspired by *Lipschitz optimization* [49, 211] and *mean value inequality for vector-valued functions*, we have

$$\|\boldsymbol{d}_i(\theta) - \boldsymbol{d}_i(\theta_c)\| \leq r_\theta \cdot \max_{\theta \in \mathbb{B}_\theta} \|\boldsymbol{d}_i'(\theta)\| \tag{5.23}$$

where $\boldsymbol{d}_i'$ is the first derivative of $\boldsymbol{d}_i$. More specifically,

$$\boldsymbol{d}_i'(\theta) = \cos(\theta)\boldsymbol{b}_i - \sin(\theta)\boldsymbol{c}_i \tag{5.24}$$

Then we define

$$\tau_i = r_\theta \cdot \overline{\|\boldsymbol{d}_i'(\theta)\|} \geq r_\theta \cdot \max_{\theta \in \mathbb{B}_\theta} \|\boldsymbol{d}_i'(\theta)\| \tag{5.25}$$

where $\overline{\|\boldsymbol{d}_i'(\theta)\|}$ is the upper bound of $\|\boldsymbol{d}_i'(\theta)\|$ and it can be calculated by *Interval Analysis* [103, 212]. Therefore,

$$\|\boldsymbol{d}_i(\theta) - \boldsymbol{d}_i(\theta_c)\| \leq \tau_i \tag{5.26}$$

Geometrically, given $\theta \in \mathbb{B}_\theta$, $\boldsymbol{d}_i(\theta)$ is a curve that is contained by a relaxed ball, whose center is at $\boldsymbol{d}_i(\theta_c)$ and radius is τ_i. (see Fig. 5.3)

$$\boldsymbol{d}_i(\theta) = \boldsymbol{d}_i(\theta_c) + \eta_i \boldsymbol{n} \tag{5.27}$$

where $0 \leq \eta_i \leq \tau_i$; $\boldsymbol{n}$ is a unit direction. As a result,

$$\boldsymbol{t}^T \boldsymbol{d}_i(\theta_c) - \tau_i \leq \boldsymbol{t}^T \boldsymbol{d}_i(\theta) \leq \boldsymbol{t}^T \boldsymbol{d}_i(\theta_c) + \tau_i \tag{5.28}$$

We define $\xi_i = \angle(\boldsymbol{t}, \boldsymbol{d}_i(\theta_c))$. Since $\boldsymbol{t} \in \mathbb{S}^{2+}$, then

$$\|\boldsymbol{d}_i(\theta_c)\| \cos(\xi_i) - \tau_i \leq \boldsymbol{t}^T \boldsymbol{d}_i(\theta) \leq \|\boldsymbol{d}_i(\theta_c)\| \cos(\xi_i) + \tau_i \tag{5.29}$$

According to the triangle inequality in spherical geometry [52], we have

$$\angle(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c)) - \angle(\boldsymbol{t}, \boldsymbol{t}_c) \leq \xi_i \leq \angle(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c)) + \angle(\boldsymbol{t}, \boldsymbol{t}_c) \tag{5.30}$$

In addition, according to Eq. (5.20) we have

$$\angle(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c)) - \sqrt{2}\delta \leq \xi_i \leq \angle(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c)) + \sqrt{2}\delta \tag{5.31}$$

Consequently, we can sequentially derive the following bounds by *Interval Analysis*

$$\lfloor \xi_i \rceil \overset{(5.31)}{\Longrightarrow} \lfloor \cos(\xi_i) \rceil \overset{(5.29)}{\Longrightarrow} \lfloor \boldsymbol{t}^T \boldsymbol{d}_i(\theta) \rceil \Rightarrow \lfloor |\boldsymbol{t}^T \boldsymbol{d}_i(\theta)| \rceil \tag{5.32}$$

where$\lfloor \cdot \rceil$ means calculating the upper and lower bounds, and $\lfloor \cdot \rceil \Rightarrow \lfloor \cdot \rceil$ indicates that the bounds of the left-hand side leads to the bounds of the right-hand side.

For simplicity, we define the lower bound of $|\boldsymbol{t}^T \boldsymbol{d}_i(\theta)|$ is $\Delta_i(\mathbb{B}_t, \mathbb{B}_\theta)$. Then given $\boldsymbol{e} \in \mathbb{B}_t$ and $\theta \in \mathbb{B}_\theta$, the upper bound of the objective can be

$$U = \sum_{i=1}^{N} \mathbb{I}(\Delta_i(\mathbb{B}_t, \mathbb{B}_\theta) \leq \epsilon) \tag{5.33}$$

Proof. Since $\Delta_i(\mathbb{B}_t, \mathbb{B}_\theta) \leq |\boldsymbol{t}^T \boldsymbol{d}_i(\theta)|$, then

$$\mathbb{I}(\Delta_i(\mathbb{B}_t, \mathbb{B}_\theta) \leq \epsilon) \geq \mathbb{I}\left(|\boldsymbol{t}^T \boldsymbol{d}_i(\theta)| \leq \epsilon\right) \tag{5.34}$$

$$\Rightarrow \sum_{i=1}^{N} \mathbb{I}(\Delta_i(\mathbb{B}_t, \mathbb{B}_\theta) \leq \epsilon) \geq \max_{\mathbb{B}_t, \mathbb{B}_\theta} \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{t}^T \boldsymbol{d}_i(\theta)| \leq \epsilon\right) \tag{5.35}$$

Therefore, U is the upper bound. □

For the lower bound, since the maximum value in the branch should not be less than the value at a specific point, it can be set as

$$L = \sum_{i=1}^{N} \mathbb{I}(\Delta_i(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c)) \leq \epsilon) \tag{5.36}$$

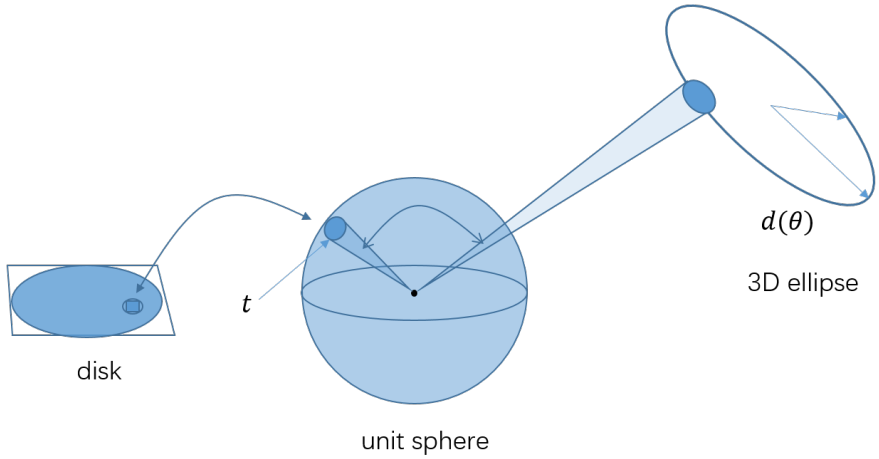


Figure 5.4: The geometric of the upper bound. Geometrically, the solution domain of $\boldsymbol{t}$ is a hemisphere, which can be mapped into a solid disk. Generally, $\boldsymbol{d}(\theta)$ is a 3D ellipse when $\theta \in [-\pi, \pi]$. Given the subdomain $\mathbb{B}_t$, and $\mathbb{B}_\theta$, $\boldsymbol{t}$ should be in a relaxed umbrella-shaped area centered at $\boldsymbol{t}_c$ and $\boldsymbol{d}(\theta)$ should be a curve which is in a relaxed ball centered at $\boldsymbol{d}(\theta_c)$.

where $\Delta_i(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c))$ is the function value at a specific point, which means $\tau_i = 0$ and $\delta = 0$.

$$\xi_i \overset{\delta=0}{\Longrightarrow} \cos(\xi_i) \overset{\tau_i=0}{\Longrightarrow} \boldsymbol{t}_c^T \boldsymbol{d}_i(\theta_c) \Rightarrow |\boldsymbol{t}_c^T \boldsymbol{d}_i(\theta_c)| = \Delta_i(\boldsymbol{t}_c, \boldsymbol{d}_i(\theta_c)) \tag{5.37}$$

In addition, when the branch collapses to a specific point $\{\boldsymbol{t}_s, \theta_s\}$, then $\delta = 0, \xi_i = \angle(\boldsymbol{t}_s, \boldsymbol{d}_i(\theta_s))$ and $\tau_i = 0$. As a result, $\lfloor |\boldsymbol{t}^T \boldsymbol{d}_i(\theta)| \rceil = |\boldsymbol{t}_s, \boldsymbol{d}_i(\theta_s)|$, then

$$L = U = \sum_{i=1}^{N} \mathbb{I}(|\boldsymbol{t}_s, \boldsymbol{d}_i(\theta_s)| \leq \epsilon) \tag{5.38}$$

In other words, the gap between upper and lower bound tends to be zero when the branch tends to a specific point. Thus the proposed upper and lower bounds are sufficient to be applied in BnB algorithm.

5.4.2 Nested BnB Algorithm

To avoid heavily computational burden, we leverage on the nested BnB idea [66], which has better memory and computational efficiency [52]. Specifically, two BnB algorithms (one is for θ and the other is for $\boldsymbol{t}$, see Fig. 5.5) are executed in a nested manner to obtain the optimal solution.

Concretely, given $\boldsymbol{t} \in \mathbb{B}_t$ and $\theta \in [-\pi, \pi]$, we can use the BnB algorithm to find the optimal θ, which is named inner BnB. Accordingly, in the inner BnB, given a branch $\mathbb{B}_\theta$, the upper bound is still

$$U_{in} = \sum_{i=1}^{N} \mathbb{I}(\Delta_i(\mathbb{B}_t, \mathbb{B}_\theta) \leq \epsilon) \tag{5.39}$$

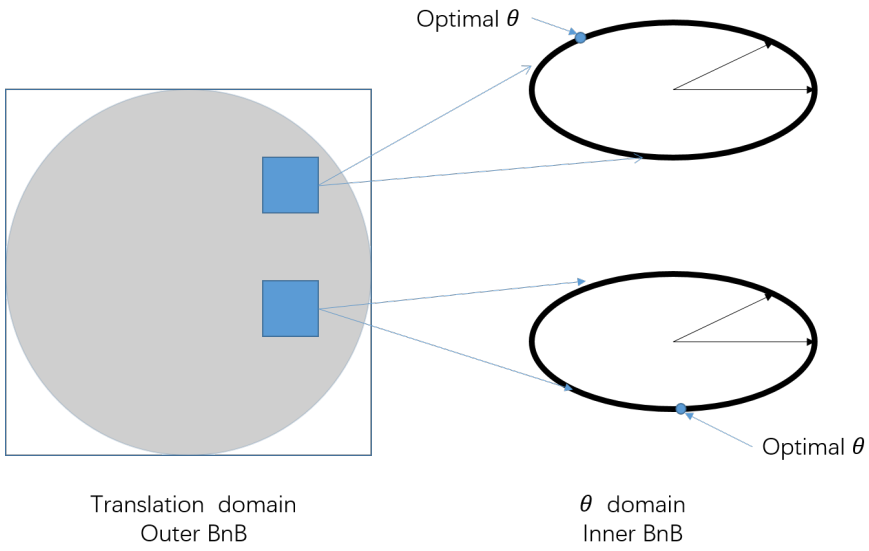


Figure 5.5: Illustration of nested BnB algorithm

But the lower bound is slightly changed

$$L_{in} = \sum_{i=1}^{N} \mathbb{I}\left(\Delta_i\left(\mathbb{B}_t, \boldsymbol{d}_i\left(\theta_c\right)\right) \leq \epsilon\right) \tag{5.40}$$

where $\Delta_i\left(\mathbb{B}_t, \boldsymbol{d}_i\left(\theta_c\right)\right)$ is the lower bound of $|\boldsymbol{t}^T \boldsymbol{d}_i(\theta_c)|$ at a specific angle θ_c, which means $\tau_i = 0$.

$$\lfloor \xi_i \rceil \Rightarrow \lfloor \cos(\xi_i) \rceil \overset{\tau_i=0}{\Longrightarrow} \lfloor \boldsymbol{t}^T \boldsymbol{d}_i(\theta_c) \rceil \Rightarrow \lfloor |\boldsymbol{t}^T \boldsymbol{d}_i(\theta_c)| \rceil \tag{5.41}$$

Furthermore, we can only consider branching the translation domain, which is called outer BnB. Accordingly, we denote the optimal θ^* which is obtained by the inner BnB in each branch $\mathbb{B}_t$. The upper and lower bound in the outer BnB are modified as

$$U_{out} = \sum_{i=1}^{N} \mathbb{I}\left(\Delta_i\left(\mathbb{B}_t, \boldsymbol{d}_i\left(\theta^*\right)\right) \leq \epsilon\right) \tag{5.42}$$

$$L_{out} = \sum_{i=1}^{N} \mathbb{I}\left(\Delta_i\left(\boldsymbol{t}_c, \boldsymbol{d}_i\left(\theta^*\right)\right) \leq \epsilon\right) \tag{5.43}$$

Note that L_{out} is the object value at a specific point $(\boldsymbol{t}_c, \theta^*)$.

The outline of the inner BnB and the whole nested BnB algorithms are summarized in Algorithm 3 and Algorithm 4.

5.5 Experiments

To verify the feasibility and the global optimality of our proposed method, we conduct experiments using both synthetic and real-world data. Besides, to

Algorithm 3: Inner BnB: Finding the optimal θ, given a translation branch $\mathbb{B}_t$.

Input: Point Matching $\{\boldsymbol{p}_i, \boldsymbol{q}_i\}_{i=1}^N$, inlier threshold ϵ, gravity direction $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ and a transalation branch $\mathbb{B}_t$.
Output: optimal θ^*

Initialize $\{\boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_i\}_{i=1}^N$;
Initialize a queue $\mathbb{Q}_\theta$ with a branch $[-\pi, \pi]$ and its initial upper and lower bounds are N and 0;
while $\mathbb{Q}_\theta$ *is non-empty* **do**
    Take out the best branch $\mathbb{B}_\theta$ which has the highest upper bound U_{in}^*;
    Branch $\mathbb{B}_\theta$ into sub-branches and estimate the upper and lower bounds;
    Update the optimal θ^* with the center of the branch which has the highest lower bound L_{in}^*;
    Remove the branches whose upper bound is less than the highest lower bound;
    if $L_{in}^* = U_{in}^*$ **then**
        Terminate
    end
end

Algorithm 4: Nested BnB: Finding the optimal relative pose $\mathbf{R}, \boldsymbol{t}$.

Input: Point Matching $\{\boldsymbol{p}_i, \boldsymbol{q}_i\}_{i=1}^N$, inlier threshold ϵ, gravity direction $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$.
Output: optimal relative pose $\mathbf{R}^*, \boldsymbol{t}^*$

Initialize $\{\boldsymbol{a}_i, \boldsymbol{b}_i, \boldsymbol{c}_i\}_{i=1}^N$;
Initialize a queue $\mathbb{Q}_t$ with the solid square and its upper and lower bound are N and 0 ;
while $\mathbb{Q}_t$ *is not empty* **do**
    Take out the best branch $\mathbb{B}_t$ which has the highest upper bound U_{out}^*;
    Branch $\mathbb{B}_t$ into sub-branches;
    Call Algorithm 3 to obtain the optimal θ^* in each sub-branch ;
    `/* ←Nested way */`
    Estimate the upper and lower bounds;
    Update the optimal $\boldsymbol{t}^*$ with the center of the branch which has the highest lower bound L_{out}^*;
    Remove the branches whose upper bound is less than the highest lower bound;
    if $L_{out}^* = U_{out}^*$ **then**
        Terminate
    end
end

demonstrate the performance, our proposed method and several state-of-the-art methods are compared.

5.5.1 Setup

All experiments are conducted in a computer with an AMD Ryzen 7 2700X CPU and 32G RAM. In all the experiments, if not specified, the inlier threshold ϵ in the objective (Eq. (5.15)) is set to 0.001. All compared state-of-the-art algorithms are listed:

- RANSAC+3pt: RANSAC framework with the sate-of-the-art gravity-known three points algorithm [191]. The confidence $\rho = 0.999$ for the stopping criterion in all the experiments.

- RANSAC+5pt: RANSAC framework with the famous five points algorithm [27]. The confidence $\rho = 0.999$ for the stopping criterion in all the experiments.

- 5d-BnB: the BnB method to solve the optimal essential matrix estimation problem without the prior knowledge of the gravity direction [135]. The code is from the authors' webpage[1].

- A*: A* tree search with Non-Adjacent Path Avoidance and Dimension-Insensitive Branch Pruning, which is named A*-NAPA-DIBP in original paper [162]. In addition, we set the maximum runtime to 60 seconds to avoid long time running. The code is from the authors' open-source code[2].

- gBnB: our proposed nested BnB method with known gravity direction.

Note that 5d-BnB is written in C++ and other algorithms are run in MATLAB2020A.

Data Generation. We randomly generate N points in 3D world. Specifically, the 3D points are inside a cube whose side is 1 meter and center is about 2 meters away from the camera. A simulated camera with 1000 pixel focal length is randomly moved but still faces the 3D points. We then have 100 matching point pairs in 2D image plane. The virtual gravity is recorded in the two different coordinates. To simulate the mismatches (i.e., outliers), we replace the correct point matches by incorrect random point matches. The outlier rate is $\eta = N_{outlier}/N$ where $N_{outlier}$ is the number of outliers and N is the total number of inputs. Besides, to simulate the noise of point localization, we add the Gaussian noise to the data, and standard deviation σ is taken as the noise level. Moreover,

[1] `http://jlyang.org`
[2] `https://github.com/ZhipengCai/MaxConTreeSearch`

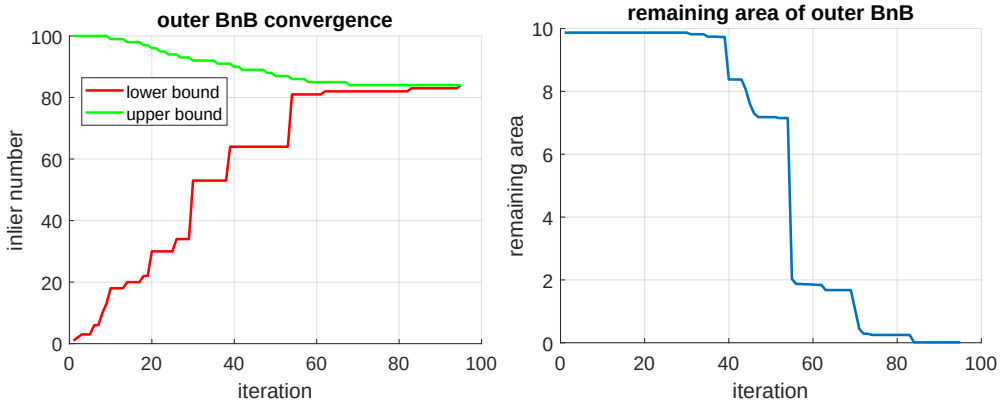


Figure 5.6: Convergence curves of the outer BnB

to calculate the accuracy, we define the error as

$$e_r = \arccos\left(\frac{Tr(\mathbf{R}_{gt}^T\mathbf{R}_{opt}) - 1}{2}\right) \tag{5.44}$$

$$e_t = \angle\left(\boldsymbol{t}_{gt}, \boldsymbol{t}_{opt}\right) \tag{5.45}$$

where $\mathbf{R}_{gt}$ and $\boldsymbol{t}_{gt}$ are motion ground truth; $\mathbf{R}_{opt}$ and $\boldsymbol{t}_{opt}$ are estimated solution; $Tr(\cdot)$ is the trace function of a square matrix.

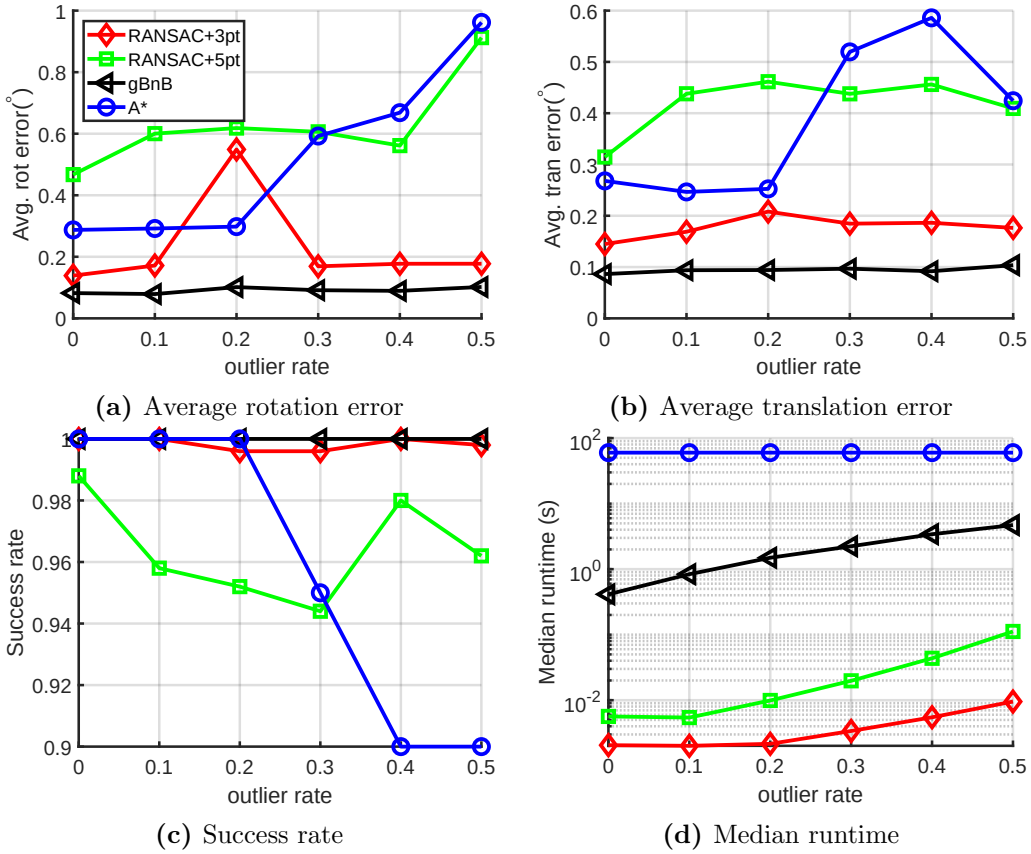
5.5.2 Convergence

In this part, we show the convergence curves to demonstrate the global optimality of the proposed method. Specifically, $N = 100$ point correspondences are generated. In the nested BnB algorithm, the inner BnB will be called in each iteration of the outer BnB. Therefore, we only draw the outlier BnB convergence curves to show the process. The curves are shown in Fig. 5.6. Evidently, the gap between the lower and upper bounds is becoming to zero. The remaining search domain reduces rapidly.

5.5.3 Controlled Experiments on Synthetic Data

In this section, to verify the global optimality of our proposed method, we conducted controlled experiments using synthetic data. First, we test our proposed method in different outlier rates, $\eta = \{0, \cdots, 0.5\}$. The noise level σ is set to 1. $N = 100$ point correspondences are generated. To observe the global optimality, the experiments are repeated 200 times for each experimental setting. We then calculate the success rate where a case that satisfies $err_{rot} \leq 2^\circ$ and $err_{tran} \leq 2^\circ$ is considered as a success case. In addition, the average error, median runtime and consensus number are recorded. The results are showed in Fig. 5.7 and Fig. 5.8.

Furthermore, we test our proposed method in different noise levels, $\sigma = \{0, \cdots, 2\}$. The outlier rate η is set to 0.2. $N = 100$ points are generated. 200



(a) Average rotation error

(b) Average translation error

(c) Success rate

(d) Median runtime

Figure 5.7: Controlled experiments on synthetic data with different outlier rates

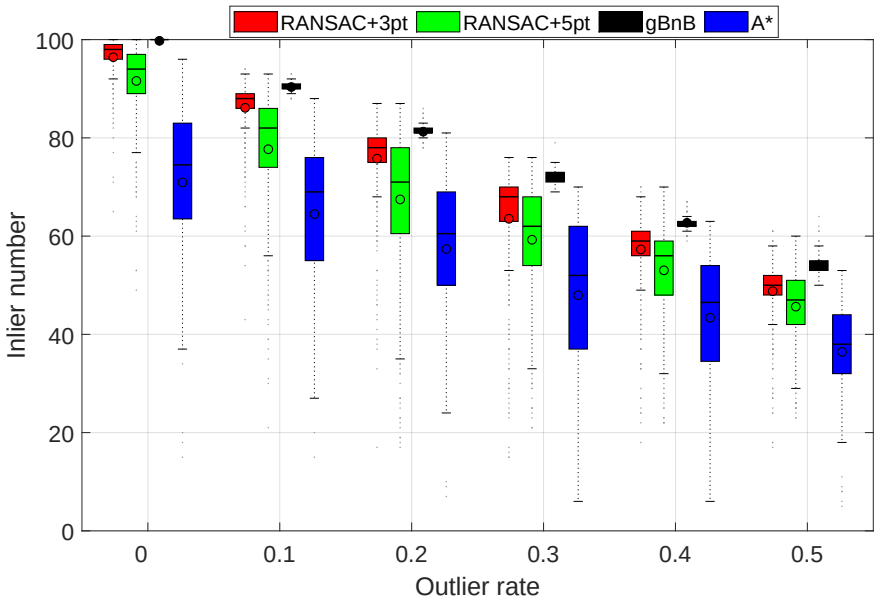
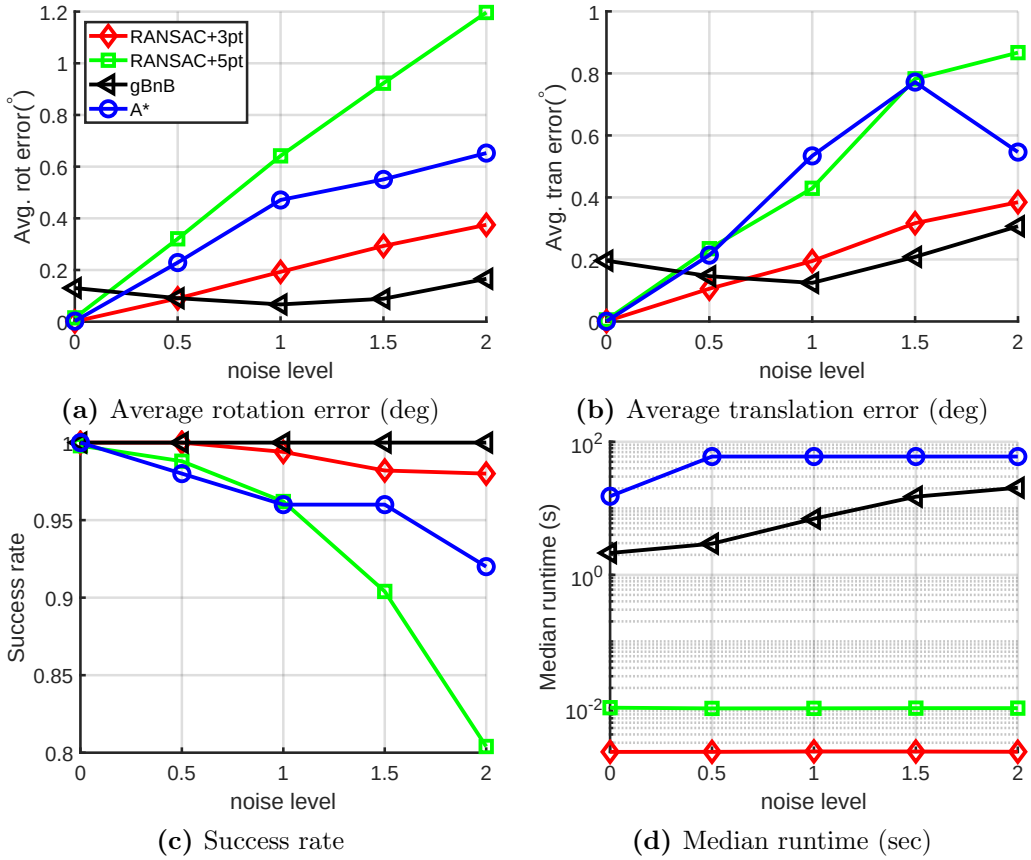


Figure 5.8: Consensus number comparison in controlled experiments on synthetic data with different outlier rates

times are repeated in each experimental setting to observe the global optimality. The success rate, average error and median runtime are shown in Fig. 5.9 and the consensus number are shown in Fig. 5.10.



(a) Average rotation error (deg)

(b) Average translation error (deg)

(c) Success rate

(d) Median runtime (sec)

Figure 5.9: Controlled experiments on synthetic data with different noise levels.

In addition, we compare the efficiency between our proposed method and the 5d-BnB method, which does not rely on prior gravity [135]. In this experimental setting, no outliers and noise are added. We only test the methods in different number of inputs, $N = \{20, \cdots, 140\}$. Since they are globally optimal methods, we only show the time duration in Table 5.1, which is median runtime on 50 trials.

Table 5.1: Comparison of median runtime (seconds) on synthetic data with different input numbers.

point number	20	50	80	110	140
gBnB	0.287	0.351	0.392	0.462	0.534
5d-BnB	21.666	29.963	32.884	37.405	58.828

From all the results, we can draw the following points:

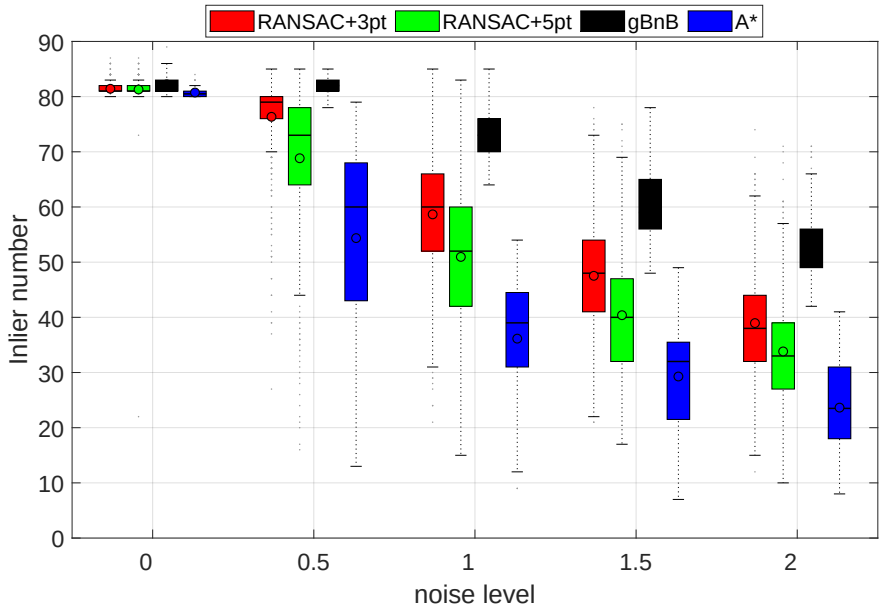


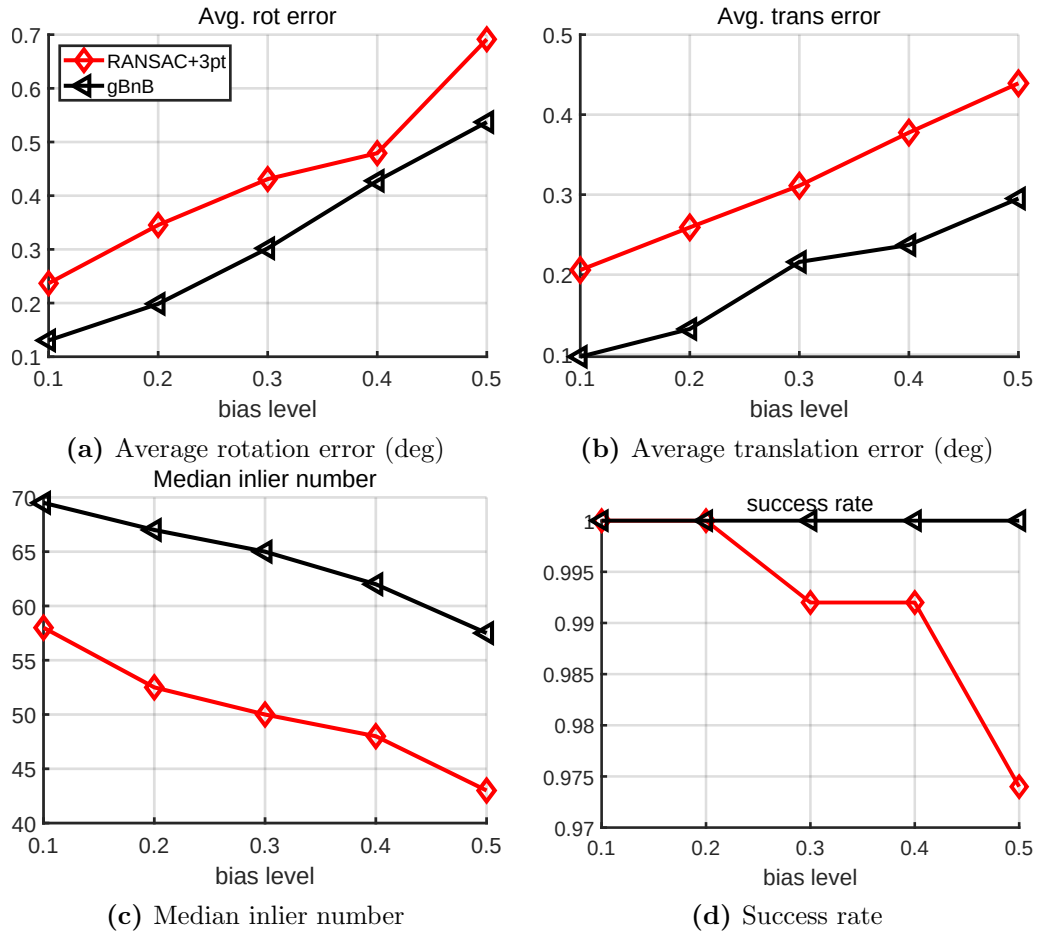
Figure 5.10: Consensus number comparison in controlled experiments on synthetic data with different noise levels

- Due to the random nature, RANSAC-based methods may return incorrect solutions while they run very fast. In contrast, our proposed BnB method can obtain the globally optimal solution from outlier-contaminated inputs in this experimental setting, while it needs more time than RANSAC-based methods.
- When outlier rate increases, tree search method consumes more runtime significantly. In addition, the tree search method cannot obtain all satisfactory solutions in our experiments. There are two reasons:
  1. The optimal solution for linearized objective is not necessarily the same as that of original objective.
  2. In large outlier rate, the tree search method reaches the time limitation (60 seconds) and terminates early, which mean it is not able to fully search all the solution candidates and just return the best-so-far solution.

  In contrast, our proposed method can obtain extremely robust solution and has the highest success rate using much less runtime.
- With the increase of noise level, the accuracy of all methods will decrease. However, when the noise level is large, our proposed method can obtain better accuracy than other methods, which reveals the robustness of our proposed method.
- With the help of gravity direction, our proposed BnB method is much faster than 5d-BnB method, which is one of the state-of-the-art globally-optimal methods. The reason is that with the help of gravity direction, the

dimensionality of solution domain reduces from five to three, which leads to high efficiency of the BnB framework.

5.5.4 Robustness to IMU Noise



(a) Average rotation error (deg)

(b) Average translation error (deg)

(c) Median inlier number

(d) Success rate

Figure 5.11: Controlled experiments under different bias gravity levels (deg).

In this part, given a biased gravity direction, which simulates the measurement bias of IMUs, we conduct experiments to verify the robustness of our proposed method. First, we set $N = 100$, $\eta = 0.2$, $\sigma = 1$ and the biased angle is set from $0.1°$ to $0.5°$. It is repeated 500 times under each experimental setting. In this case, we only compare our proposed method with RANSAC+3pt, since other methods are not sensitive to gravity direction. The median runtime, average error and success rate are recorded. Besides, we also record the inlier number, which is a common metric in consensus maximization solutions [57, 162]. In general, the maximum objective for consensus maximization always occurs at the optimal solution, therefore, we record the consensus set number. The results are showed in Fig. 5.11.

From the results, we can find that when the gravity bias level increases, the accuracy of our proposed method decreases. It is reasonable because the

given gravity direction is biased gradually. Nonetheless, our proposed method can still obtain satisfactory solution even with a biased gravity direction ($\leq 0.5°$). It is worth noting that due to the biased gravity direction, RANSAC+3pt method cannot obtain the maximum inlier number, and our proposed method is significantly more robust than the RANSAC+3pt method.

Table 5.2: Maximum error and average runtime(seconds) using selected images from KITTI dataset in 50 times.

Methods		RANSAC+3pt	RANSC+5pt	A*	gBnB
Frame	e_r	1.438	1.248	3.592	**0.685**
104-	e_t	3.397	7.701	24.982	**3.054**
108	T	**1.749**	21.162	60.028	23.706
Frame	e_r	0.389	0.672	2.295	**0.346**
198-	e_t	1.785	3.792	17.022	**1.599**
201	T	**1.703**	20.448	60.076	20.078
Frame	e_r	0.218	0.268	0.806	**0.176**
417-	e_t	3.360	2.491	7.635	**2.310**
420	T	**1.761**	20.333	18.106	5.566
Frame	e_r	0.276	0.375	1.452	**0.242**
579-	e_t	1.751	3.357	19.416	**1.668**
582	T	**1.757**	20.392	60.166	50.020
Frame	e_r	0.653	0.817	0.997	**0.597**
738-	e_t	7.642	10.502	10.505	**7.034**
742	T	**1.745**	20.215	54.302	12.632

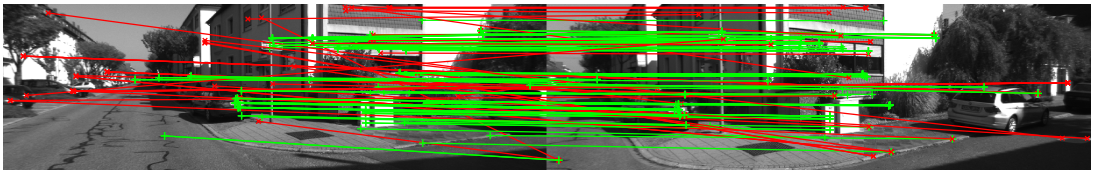
5.5.5 Real-World Data Experiments

In this part, we verify the feasibility and practicality using real-world data. We select 5 image pairs (the first 5 crossroads) from the sequence 00 of the KITTI Odometry dataset [24], see Fig. 5.12. The image pairs are obtained under ground motion. This experimental setting is similar to the setting in [162]. We utilize MATLAB built-in functions *detectMSERFeatures*[3] and *matchFeatures*[4] to obtain the input correspondences. Note that to emphasize the outlier-robustness, we adjust the parameters to obtain more-than-usual mismatches. Besides, the iteration number of RANSAC-based methods is set to 10,000 since we have no prior knowledge of outlier rate in each scene.

To show the robustness, we repeat 50 times in each image pair and we record the maximum error. The results are listed in Table 5.2. Note that 5d-BnB algorithm cannot terminate in 60 seconds in most of cases, we then do not list the results. From the results, we can find that RANSAC-based algorithms may return an unsatisfactory solution (error$> 10°$). In addition, A* method may not return

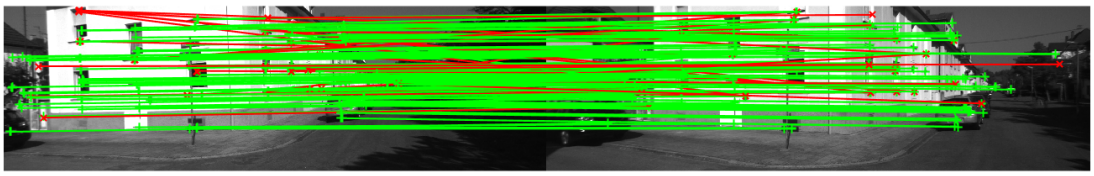
[3]https://www.mathworks.com/help/vision/ref/detectmserfeatures.html
[4]https://www.mathworks.com/help/vision/ref/matchfeatures.html

a correct solution due to time limitation and dropping non-linear constraints. In contrast, our proposed method usually obtain a satisfactory solution. It is worth mentioning that the proposed method needs more time than RANSAC-based method. However, the proposed method is more efficient than 5d-BnB and A* algorithm.

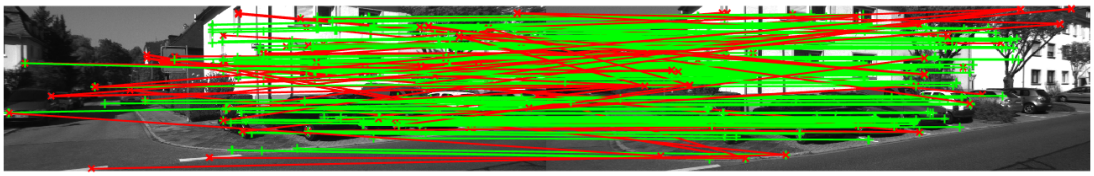


(a) Correspondences from Frame 104-108

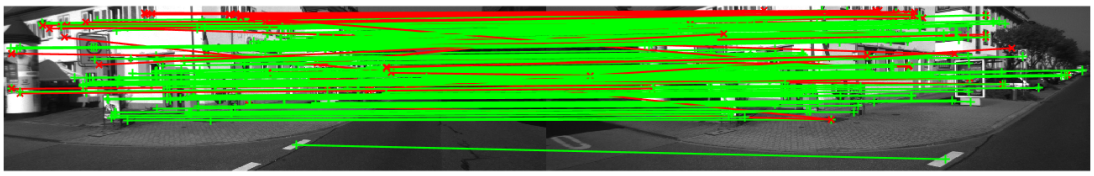
(b) Correspondences from Frame 198-201



(c) Correspondences from Frame 417-420



(d) Correspondences from Frame 579-582



(e) Correspondences from Frame 738-742

Figure 5.12: Input from KITTI dateset. Red lines denote the incorrect correspondences and green lines denote the correct inliers. The result is obtained by our proposed gBnB algorithm.

5.6 Conclusion

In this chapter, we focus on a special case in which given the gravity direction, solving the relative pose from outlier-contaminated inputs. Since traditional RANSAC-based methods fail to guarantee the global optimality of the solution,

we propose a novel nested BnB algorithm, which is able to obtain the globally-optimal solution. Even with a biased gravity direction, the proposed method can still obtain satisfactory solutions. We also verify the feasibility and practicality of our proposed method using both synthetic and real-world data. The experimental results reveal extreme robustness of our proposed method.

Chapter 6

Conclusion and Further Study

6.1 Conclusion

In this thesis, we focus on globally optimal solutions for unit-norm constrained computer vision problems, which can be considered as special cases of geometric computer vision problems and are frequently encountered in real applications. Mathematically, due to the non-convex nature of the unit-norm constraint, solving unit-norm constrained computer vision problems is naturally optimizing non-convex problems. In addition, to suppress the noise and outliers in real applications, robust objective functions, which normally are not convex, even not smooth, are formulated to distinguish inliers and outliers. The goal of our research is to find the optimal solution of modeled non-convex optimization problems, which may contains many local optimums. However, locally optimal algorithms and heuristic algorithms, which can return an optimum but do not ensure that the solution is indeed the best (global) optimum, may lead to serious failures. Therefore, to meet the demand in safety-critical applications, we study the globally optimal solutions for unit-norm constrained computer vision problems. The proposed globally optimal algorithms can provide theoretical guarantees: the returned optimal solution is indeed the global one,

Mathematically, the most of work in this thesis are applying the BnB framework to obtain the global optimums for the unit-norm constrained optimization problems. The main theoretical contributions of this thesis are introducing the general inequality (Lemma 2.6) in n-sphere ($\mathbb{S}^n$) and using it to estimate the upper and lower bounds in the BnB algorithm. Notably, the well-known rotation search theory thoroughly studies inequalities in $\mathbb{SO}(3)$ that is closely related to the quaternion sphere($\mathbb{S}^3$). Therefore, the general inequality in n-sphere is to some extent an extension of the core inequality (Lemma 2.2) in rotation search theory, see Fig. 6.1. In fact, to introduce the inequality in n-sphere, we provide solid proofs and explore the geometry of n-sphere. Furthermore, based on this inequality in n-sphere, several globally optimal solutions are proposed to solve classical unit-norm constrained computer vision problems.

Specifically, in this thesis, the following tasks are explored:

1. Globally optimal vertical world frame estimation in structural world. It is a typical robust estimation application in $\mathbb{S}^2$.

2. Globally optimal camera orientation estimation using 2D/3D feature correspondences. It is a rotation search problem in $\mathbb{SO}(3)$, which is closely related to $\mathbb{S}^3$.

3. Globally optimal relative pose estimation with known vertical direction. It uses the inequality in $\mathbb{S}^2$ and the nested BnB algorithm.

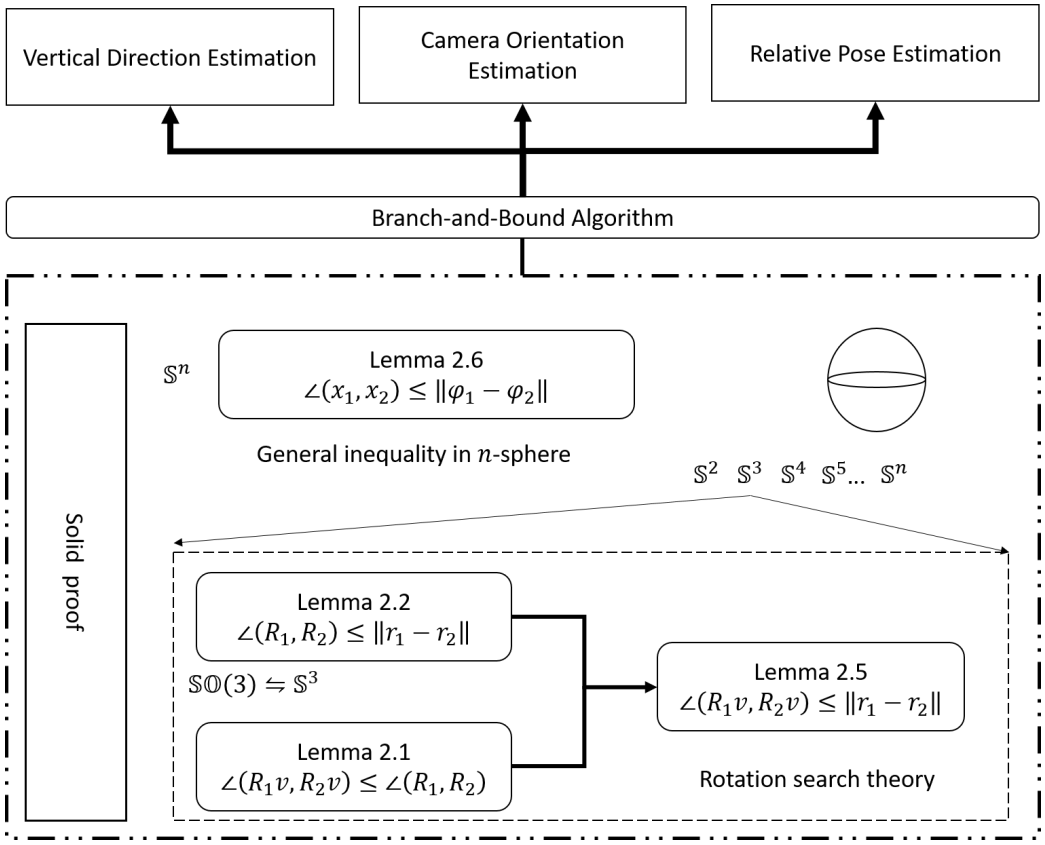


Figure 6.1: The theoretical overview of this thesis.

6.2 Limitation

In this thesis, three globally optimal solutions are proposed to meet the demand of the safety-critical systems. However, all methods rely on the BnB algorithm to obtain the global optimum. Although the BnB algorithm is more efficient than brute-force exhaustive search, it is still far from real time. Therefore, there are two main limitations introduced by BnB algorithm(see Fig. 6.2):

- The proposed methods are time-consuming. It is not easy to theoretically analyze the computational complexity of the BnB algorithm [213]. Practically, when the inputs are heavily contaminated by outliers, the BnB algorithms need a long time to execute.

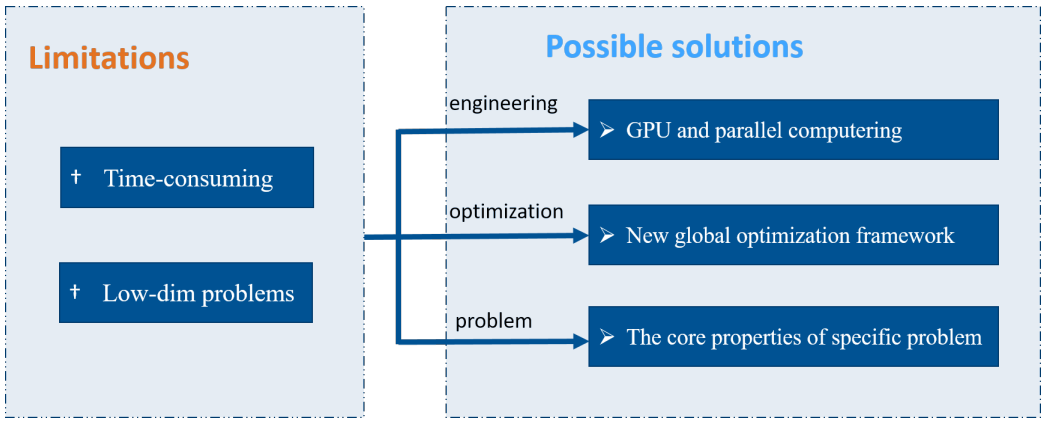


Figure 6.2: Main limitations of the proposed methods.

- Currently, the proposed methods can only solve low-dimensional problems. To ensure the global optimality, the BnB algorithm need to search the entire solution domain. Therefore, it will encounter the infamous curse of dimensionality when it comes to high-dimensional problems [62].

Accordingly, from different views, there are different possible solutions to alleviate limitations:

- From an engineering perspective, GPU and parallel computing may be a good way to accelerate the search process [52].
- From the perspective of optimization, a new optimization framework should be explored to avoid the curse of dimensionality.
- From the problem itself, if a good property of the to-be-solved problem is discovered, then it may accelerate the BnB algorithm.

6.3 Further Research Directions

There are many possible further research directions including theoretical side and practical side. We first list some possible theoretical directions:

- First Order Inequality [92]. Let us look back the lemma 2.5 in $\mathbb{SO}(3)$ space.

$$\angle\left(\mathbf{R}_1\boldsymbol{x}, \mathbf{R}_2\boldsymbol{x}\right) \leq \|\boldsymbol{r}_1 - \boldsymbol{r}_2\| \tag{6.1}$$

  The authors in [92] call it zero-th order approximation. To obtain a small gap, they propose a first order approximation to rotation. Specifically,

> Let $\mathbf{R}$ be a rotation and $\hat{\mathbf{R}}$ be its first order approximation about $\mathbf{R}_0$, where $\angle\left(\mathbf{R}, \mathbf{R}_0\right) < r_D < 0.76$. For $\boldsymbol{x} \in \mathbb{R}^3$ [92]
>
> $$\angle(\mathbf{R}\boldsymbol{x}, \hat{\mathbf{R}}\boldsymbol{x}) \leq \frac{1}{2} r_D^2 \tag{6.2}$$

They point out that the first order approximation gives a significantly better result in rotation search. Therefore, similarly, there may be a first order approximation in n-sphere. Specifically, since we already have lemma 2.6

$$\angle(\boldsymbol{x}_p, \boldsymbol{x}_q) \leq \|\boldsymbol{\varphi}_p - \boldsymbol{\varphi}_q\| \tag{6.3}$$

Possibly, with some conditions, there may be

$$\angle(\boldsymbol{x}, \dot{\boldsymbol{x}}) \leq K\|\boldsymbol{\varphi} - \boldsymbol{\varphi}_0\|^2 \tag{6.4}$$

where K is a scalar; $\dot{\boldsymbol{x}}$ is first order approximation about $\boldsymbol{x}_0$, which corresponds to $\boldsymbol{\varphi}_0$. As a consequence, the results may be improved significantly by using the first order approximation.

To explain it better, we abuse the definition of Lipschitz-Hölder continuous [214, 215],

$$d_Y(f(x_1), f(x_2)) \leq K d_X(x_1, x_2)^\alpha \tag{6.5}$$

where K is referred to as a constant scalar; $d_X(\cdot,\cdot)$ and $d_Y(\cdot,\cdot)$ are distance function in two metric spaces; α is order scalar. Here, α is very related to the **convergence order** in the BnB algorithm [216]. Theoretically, second order convergence (using Eq. (6.4)) performs faster than one order convergence (using Eq. (6.3)).

- Quasi Branch and Bound [216]. For the problem of minimizing a smooth function over a cube, the quasi branch and bound (qBnB) algorithm is recently proposed. In the classical branch-and-bound algorithm, given a sub domain, the upper and lower bounds of the optimum within this sub domain should be estimated. However, the basic idea of qBnB algorithm is to replace the lower bounds with quasi-lower bounds. Specifically, quasi-lower bounds are a generalization of lower bounds and the quasi-lower bounds are just required to be lower bounds only for sub-cubes containing a minimizer. Notably, lower bounds are quasi-lower bounds, but quasi-lower bounds are not necessarily lower bounds. The authors in [216, 217] point that qBnB compares favorably with alternative branch and bound algorithms. Therefore, in further study, we may replace BnB by qBnB to improve the performance.

- With other optimization frameworks. In this thesis, we applied BnB optimization framework to obtain the global optimum. However, the BnB algorithm is often slow in high dimension. To handle high dimensional problems, some other novel optimization frameworks should be explored.

In addition to theoretical innovations, the general inequality in n-sphere and BnB framework can be applied in many other unit-norm constrained applications to obtain the globally optimal solutions:

- Rigid pose estimation applications. Evidently, rotation estimation, which is an unit-norm constrained problem, is a vital part of rigid pose estimation and it can be solved by the proposed method. Therefore, our globally optimal solutions for unit-norm constrained problems can be the basic parts for obtaining globally optimal pose estimation. Many pose estimation problems have been explored to obtain their globally optimal solutions, such as [52, 66]. However, there are still many pose estimation application have not been explored. That is a good further research direction.

- Direction estimation applications. Direction estimation is a typical unit-norm constrained problems. In this thesis, we only explored the vertical direction in structural world. There are surely many other direction estimation applications. A further research direction is to provide the globally optimal solutions for these applications. An example is the densest hemisphere problem [139]. Specifically, given a set $\mathcal{K}$ of n points on the unit sphere $\mathbb{S}^d$, a hemisphere of $\mathbb{S}^d$ is densest if it contains a largest subset of $\mathcal{K}$. The problem of determining a densest hemisphere is a typical direction estimation problem.

- Other unit-norm constrained problems. In addition to computer vision, there are undoubtedly many unit-norm constrained problems in other scientific fields. Some of them may need to solve the problems globally, the methods developed in this thesis might be helpful.

6.4 Extension Example: Globally Optimal Solution for Line Fitting

This part shows how to apply the core idea in this thesis to other applications step by step, and a toy example is given. Specifically, we show a simple line fitting application and apply the method developed in this thesis to seek its global optimum.

Given many input points from image, estimating the line parameters from the noisy inputs is the called robust line fitting, see Fig. 6.3. If we apply the homogeneous representation, it is a typical unit-norm constrained problem.

$$ax_i + by_i + c = 0, \quad i = 1 \cdots N \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.6}$$

Since it is a unit-norm constrained problem, then the globally optimal solution can be obtained by the introduce inequality and BnB algorithm.

Notably, using homogeneous representation will make the inputs from two dimension $[x_i, y_i]^T$ to three dimension $[x_i, y_i, 1]^T$, see Fig. 6.4. Furthermore, $[a, b, c]^T$ lies in the unit sphere and it is the unit normal of a plane which is constructed by original point and $[x_i, y_i, 1]^T$.

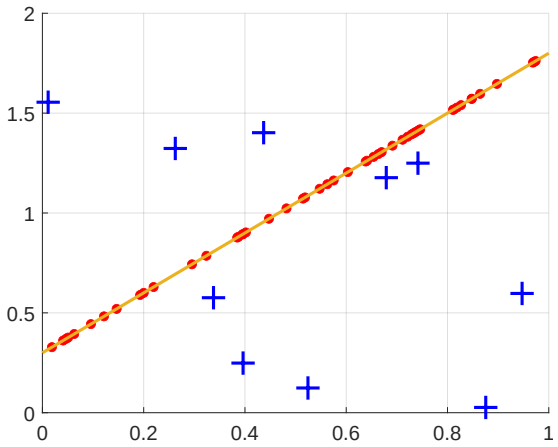


Figure 6.3: Line fitting example. Blue cross markers are the outliers and red dot markers are the inlier measurements.

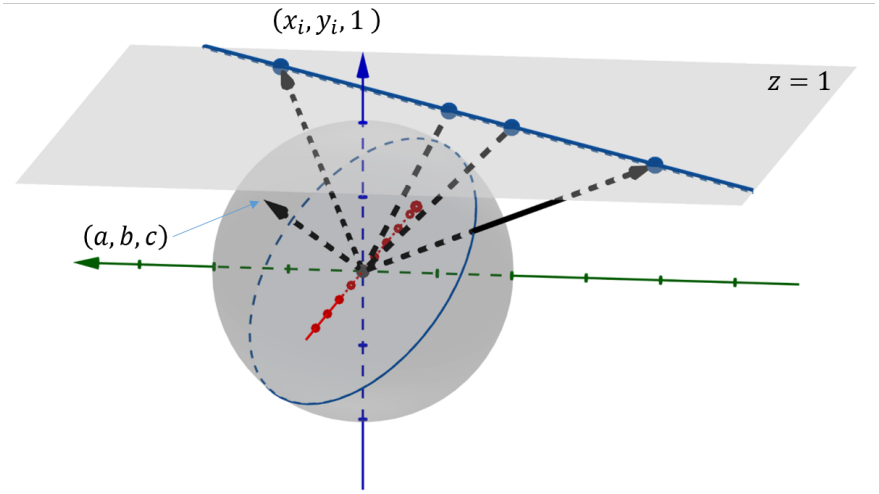


Figure 6.4: Geometric interpretation of homogeneous representation

First, to suppress the outliers, the robust objective function can be formulated by consensus maximization

$$\max \sum_{i=1}^{N} \mathbb{I}\left(|ax_i + by_i + c| \leq \epsilon\right) \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.7}$$

where $\mathbb{I}(\cdot)$ is indicator function and ϵ is inlier threshold.

If we initialize the inputs $t_i[x_i, y_i, 1] = [x_i', y_i', z_i']$, and $x_i'^2 + y_i'^2 + z_i'^2 = 1$, then

$$ax_i + by_i + c = 0 \leftrightharpoons ax_i' + by_i' + cz_i' = 0 \quad i = 1 \cdots N \tag{6.8}$$

Therefore, we reformulate the objective function

$$\max \sum_{i=1}^{N} \mathbb{I}\left(|ax_i' + by_i' + cz_i'| \leq \epsilon\right) \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.9}$$

Let $\boldsymbol{x} = [a, b, c]^T$ and $\boldsymbol{m}_i = [x_i', y_i', z_i']^T$. Equivalently,

$$\max \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}| \leq \epsilon\right) \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.10}$$

$$\Leftrightarrow \max \sum_{i=1}^{N} \mathbb{I}\left(|\cos\left(\angle\left(\boldsymbol{m}_i, \boldsymbol{x}\right)\right)| \leq \epsilon\right) \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.11}$$

$$\Leftrightarrow \max \sum_{i=1}^{N} \mathbb{I}\left(|\sin\left(\angle\left(\boldsymbol{m}_i, \boldsymbol{x}\right) - \frac{\pi}{2}\right)| \leq \epsilon\right) \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.12}$$

$$\Leftrightarrow \max \sum_{i=1}^{N} \mathbb{I}\left(|\angle\left(\boldsymbol{m}_i, \boldsymbol{x}\right) - \frac{\pi}{2}| \leq \arcsin\left(\epsilon\right)\right) \quad s.t. \quad a^2 + b^2 + c^2 = 1 \tag{6.13}$$

Note that the objective function is a special case of Chapter 3. Specifically, the objective in Chapter 3 needs to count the inliers that perpendicular or parallel the sought direction. However, in this part, the objective function only count the perpendicular inliers. Therefore, we can also consider exponential mapping and sightly change the upper bound in Chapter 3 to formulate the bounds for BnB algorithm. However, for the completeness, we still present the detailed derivation and mind flow.

Since $[a, b, c]^T$ and $[-a, -b, -c]^T$ represent a same line, we then set the optimal solution domain of $[a, b, c]^T$ is a hemisphere. Accordingly, to represent the hemisphere compactly, the hemisphere is flatten to a solid disk see Fig. 6.5. Then the circumscribed square of the disk is initialized as the solution domain for line parameters $\boldsymbol{x}$.

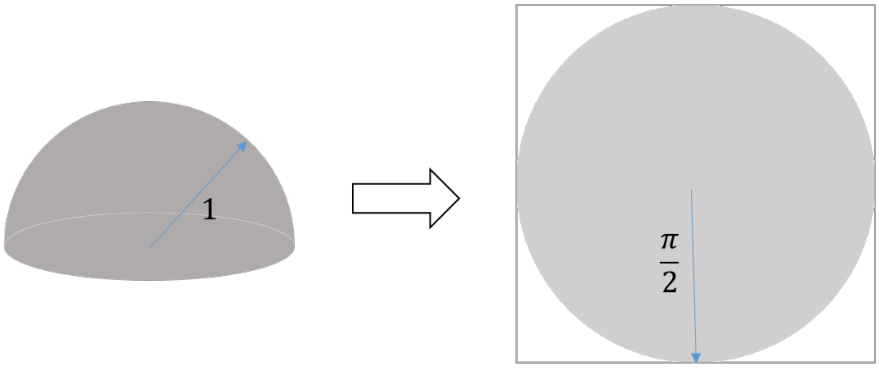


Figure 6.5: Illustration of exponential mapping of a hemisphere

To seek the global optimum, the BnB algorithm is applied and the square-shaped region is being divided into small regions (see Fig. 6.6). To estimate the bound, we then need to count the inlier number if $\boldsymbol{x}$ is in a given small branch. Specifically, given a square-shaped branch in the disk, the center is $\boldsymbol{x}_c$ and half side is σ, the upper bound can be

$$U = \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}_c| \leq \sin\left(\min\left\{\arcsin(\epsilon) + \sqrt{2}\sigma, \frac{\pi}{2}\right\}\right)\right) \tag{6.14}$$

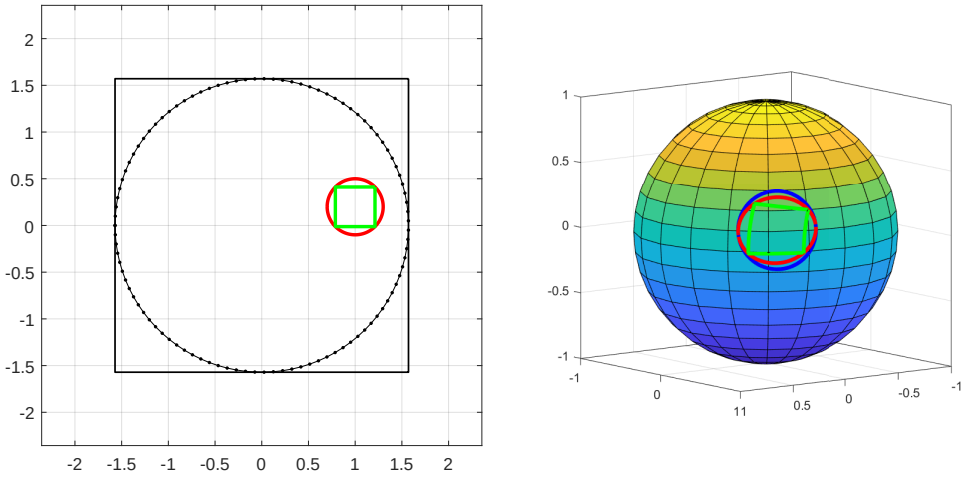


Figure 6.6: Geometrical illustration of the bound. The region bounded by green square in left sub-figure means the divided sub-branch. It corresponds to an irregular region bounded by green boundary in right sub-figure. Then it is relaxed to an umbrella-shaped patch in the sphere (region bounded by blue circle). The circle-region bounded by red line in the left sub-figure is the circumscribed circle of the small branch. The irregular region in the right sub-figure is corresponding to the circumscribed circle in the left.

The lower bound can be

$$L = \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}_c| \leq \epsilon\right) \tag{6.15}$$

Proof. 1. For the upper bound, observe

$$|\boldsymbol{m}_i^T \boldsymbol{x}| \leq \epsilon \Leftrightarrow |\angle(\boldsymbol{m}_i, \boldsymbol{x}) - \frac{\pi}{2}| \leq \arcsin(\epsilon) \tag{6.16}$$

$$\Rightarrow |\angle(\boldsymbol{m}_i, \boldsymbol{x}_c) - \frac{\pi}{2}| \leq \arcsin(\epsilon) + \sqrt{2}\delta \tag{6.17}$$

$$\Leftrightarrow \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}_c| \leq \sin\left(\min\left\{\arcsin(\epsilon) + \sqrt{2}\sigma, \frac{\pi}{2}\right\}\right)\right) \tag{6.18}$$

Then,

$$\mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}| \leq \epsilon\right) = 1 \Rightarrow \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}_c| \leq \sin\left(\min\left\{\arcsin(\epsilon) + \sqrt{2}\sigma, \frac{\pi}{2}\right\}\right)\right) = 1 \tag{6.19}$$

Therefore,

$$\max \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}| \leq \epsilon\right) \leq \sum_{i=1}^{N} \mathbb{I}\left(|\boldsymbol{m}_i^T \boldsymbol{x}_c| \leq \sin\left(\min\left\{\arcsin(\epsilon) + \sqrt{2}\sigma, \frac{\pi}{2}\right\}\right)\right) \tag{6.20}$$

2. For the lower bound, the maximum in the region should be no less than the objective value at any specific point in the region.

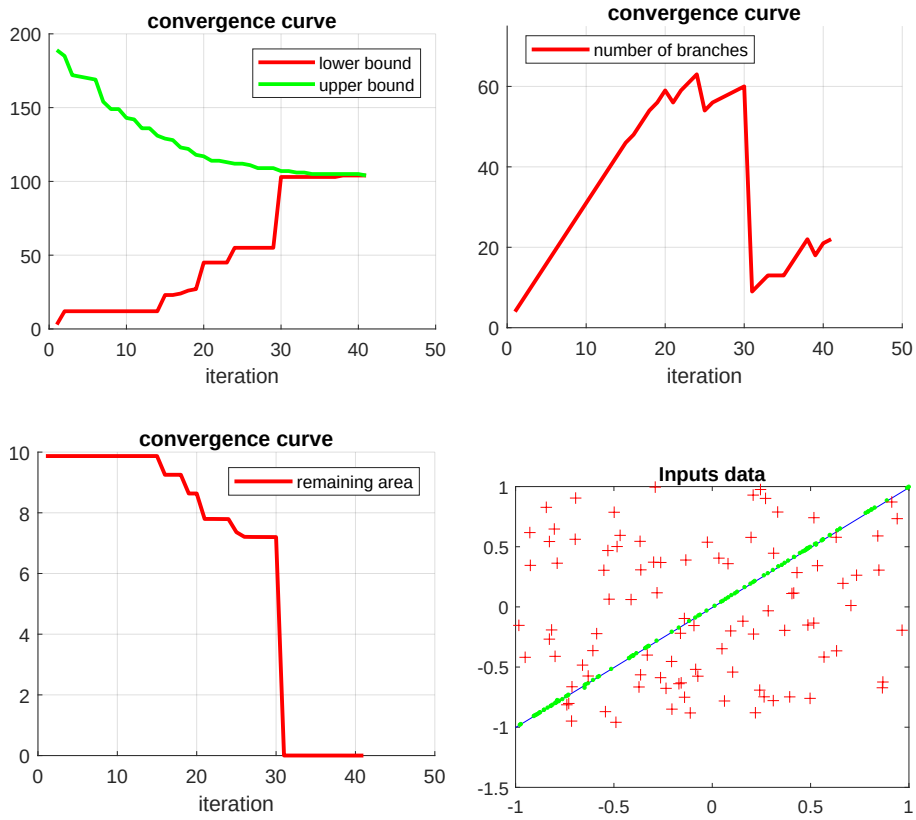


Figure 6.7: The convergence curves in BnB algorithm using two dimensional synthetic data. As the algorithm iterates, the gap between the lower and upper bounds is to zero and the remaining area is to be very small.

3. When the sub-branch is reduced to be a point, the gap between lower and upper bound is zero.

□

We verify the validation of the method using synthetic input data. The synthetic data are randomly generated N=200 with 50% outliers. The convergence curves are shown in Fig. 6.7. Evidently, the BnB algorithm can rapidly converge to the globally optimal solution.

In conclusion, we provide a simple globally optimal solution for robust line fitting from two dimensional outlier-contaminated measurements. Particularly, it demonstrates how to extend our proposed core idea in this thesis to other applications.

Appendix A

Triangle Inequality on the Hyper-sphere $\mathbb{S}^n$

In this thesis, we apply the triangle inequality on the hyper-sphere many times. We then introduce a simple proof in this part.

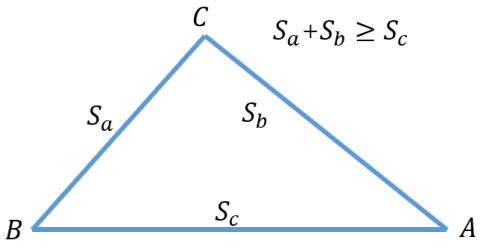


Figure A.1: An examples of the triangle inequality for triangles with sides of lengths S_a, S_b, S_c.

The triangle inequality [218], which states for any triangle in Euclidean space, the sum of the lengths of any two sides must be no less than the length of the remaining side, is well-known. Geometrically, it is shown in Fig A.1. Moreover, the triangle inequality not only stands in $\mathbb{R}^2$ but also holds in $\mathbb{R}^n$, where n is a positive integer which means the dimensionality of the space [219].

Formally, given each three points $\forall A, B, C \in \mathbb{R}^n$, there is

$$\|\overrightarrow{AB}\| \leq \|\overrightarrow{AC}\| + \|\overrightarrow{BC}\| \tag{A.1}$$

where $\|\cdot\|$ is the Euclidean norm of a vector. This results can be obtained elegantly by Cauchy–Schwarz inequality [220].

$$\begin{aligned}
\|\overrightarrow{AB}\|^2 &= \left(\|\overrightarrow{AC} + \overrightarrow{CB}\|\right)^2 && \text{(A.2)}\\
&= \|\overrightarrow{AC}\|^2 + \|\overrightarrow{BC}\|^2 + 2dot\left(\overrightarrow{AC}, \overrightarrow{CB}\right) && \text{(A.3)}\\
&\leq \|\overrightarrow{AC}\|^2 + \|\overrightarrow{BC}\|^2 + 2\|\overrightarrow{AC}\|\|\overrightarrow{BC}\| && \text{(A.4)}\\
&= \left(\|\overrightarrow{AC}\| + \|\overrightarrow{BC}\|\right)^2 && \text{(A.5)}
\end{aligned}$$

Without difficulty, $\|\overrightarrow{AB}\| \leq \|\overrightarrow{AC}\| + \|\overrightarrow{BC}\|$. Here $dot\,(\cdot,\cdot)$ means inner product of two vectors .

It should mention that there is a general formulation, which is well known as Minkowski inequality [221]:

$$\|\overrightarrow{AB}\|_p \leq \|\overrightarrow{AC}\|_p + \|\overrightarrow{BC}\|_p \tag{A.6}$$

where $1 \leq p < \infty$, and $\|\cdot\|_p$ means the L_p norm of a vector. More generally, the triangle inequality can be formulated as

$$dist(A, B) \leq dist(A, C) + dist(B, C) \tag{A.7}$$

where $dist(\cdot, \cdot)$ is a distance function in a metric space [222–224].

If we take the angle distance as $dist(\cdot, \cdot)$, we then have

$$\angle(\boldsymbol{v}_a, \boldsymbol{v}_b) \leq \angle(\boldsymbol{v}_a, \boldsymbol{v}_c) + \angle(\boldsymbol{v}_b, \boldsymbol{v}_c) \tag{A.8}$$

where $\boldsymbol{v}_a = \overrightarrow{OA}, \boldsymbol{v}_b = \overrightarrow{OB}, \boldsymbol{v}_c = \overrightarrow{OC}$ and $0 \leq \angle(\boldsymbol{v}_a, \boldsymbol{v}_b) \leq \pi$. Specifically, $\angle(\boldsymbol{v}_a, \boldsymbol{v}_b) = \arccos(\boldsymbol{v}_a^T \boldsymbol{v}_b) / (\|\boldsymbol{v}_a\| \|\boldsymbol{v}_b\|)$. Intuitively, Eq. (A.8) seems reasonable. However, we cannot assert it is correct without solid proof.

In [52], the authors point the triangle inequality in a sphere $\mathbb{S}^2$(see Fig. A.2). Here $\mathbb{S}^2$ means a unit sphere in $\mathbb{R}^3$, in other words, $\mathbb{S}^2 = \{\boldsymbol{x} | \boldsymbol{x} \in \mathbb{R}^3, \|\boldsymbol{x}\| = 1\}$. Formally, given three points $A, B, C \in \mathbb{S}^2$, there is

$$\angle AOB \leq \angle AOC + \angle BOC \tag{A.9}$$

$$\Longleftrightarrow \widetilde{AB} \leq \widetilde{AC} + \widetilde{BC} \tag{A.10}$$

where $0 \leq \angle AOB \leq \pi$ means the angle distance between $\overrightarrow{OA}$ and $\overrightarrow{OB}$; $\widetilde{AB}$ is the length of the shortest arc path from point A to point B on the sphere.

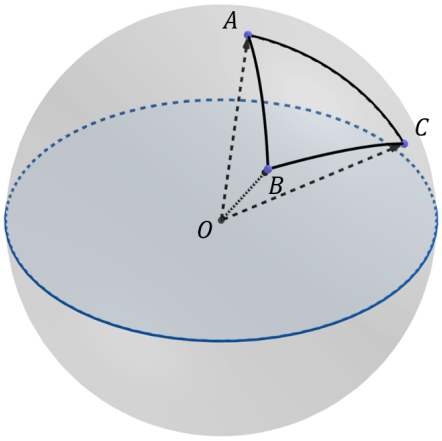


Figure A.2: The triangle inequality in $\mathbb{S}^2$.

Furthermore, this result can be extended to n-sphere, and the proof is then briefly given in this part. Formally, let $\mathbb{S}^n$ be the standard unit sphere embedded in Euclidean space as $\mathbb{S}^n = \{\boldsymbol{x} | \boldsymbol{x} \in \mathbb{R}^{n+1}, \|\boldsymbol{x}\| = 1\}$.

Given $A, B, C \in \mathbb{S}^n$ and $\boldsymbol{v}_a = \overrightarrow{OA}, \boldsymbol{v}_b = \overrightarrow{OB}, \boldsymbol{v}_c = \overrightarrow{OC}$. There is

$$\angle(\boldsymbol{v}_a, \boldsymbol{v}_b) \leq \angle(\boldsymbol{v}_a, \boldsymbol{v}_c) + \angle(\boldsymbol{v}_b, \boldsymbol{v}_c) \tag{A.11}$$

Proof. Let $\omega_{ab} = \arccos\left(\boldsymbol{v}_a^T \boldsymbol{v}_b\right)$, $\omega_{ac} = \arccos\left(\boldsymbol{v}_a^T \boldsymbol{v}_c\right)$ and $\omega_{bc} = \arccos\left(\boldsymbol{v}_b^T \boldsymbol{v}_c\right)$. Then

$$\boldsymbol{v}_a = \cos(\omega_{ac})\boldsymbol{v}_c + \sin(\omega_{ac})\boldsymbol{z}_{ac} \tag{A.12}$$
$$\boldsymbol{v}_b = \cos(\omega_{bc})\boldsymbol{v}_c + \sin(\omega_{bc})\boldsymbol{z}_{bc} \tag{A.13}$$

where $\boldsymbol{z}_{ac}, \boldsymbol{z}_{bc}$ are two unit vectors, which are orthogonal to $\boldsymbol{v}_c$. Note that $\boldsymbol{v}_a$,$\boldsymbol{v}_c$ and $\boldsymbol{z}_{ac}$ are in a plane; $\boldsymbol{v}_b$,$\boldsymbol{v}_c$,$\boldsymbol{z}_{bc}$ are in a plane (see Fig. A.3).

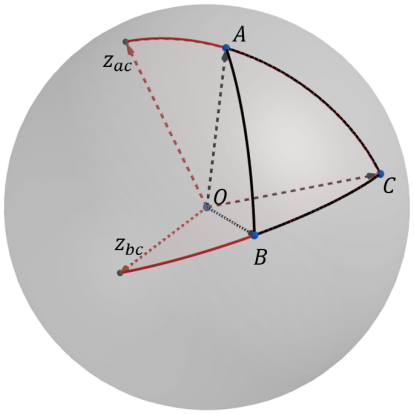


Figure A.3: The vector decomposition in $\mathbb{S}^2$.

Observe

$$\cos(\omega_{ab}) = \boldsymbol{v}_a^T \boldsymbol{v}_b \tag{A.14}$$
$$= \cos(\omega_{ac})\cos(\omega_{bc}) + \sin(\omega_{ac})\sin(\omega_{bc})\boldsymbol{z}_{ac}^T \boldsymbol{z}_{bc} \tag{A.15}$$
$$\geq \cos(\omega_{ac})\cos(\omega_{bc}) - \sin(\omega_{ac})\sin(\omega_{bc}) \tag{A.16}$$
$$= \cos(\omega_{ac} + \omega_{bc}) \tag{A.17}$$

Note that Eq. (A.15) is actually the spherical law of cosines [225].

After that, we have two cases:

1. If $\omega_{ac} + \omega_{bc} \geq \pi$, then $\omega_{ab} \leq \pi \leq \omega_{ac} + \omega_{bc}$.
2. If $\omega_{ac} + \omega_{bc} < \pi$, since $\cos(x)$ is a monotonically decreasing function when $x \in [0, \pi]$, therefore, $\omega_{ab} \leq \omega_{ac} + \omega_{bc}$.

Based on the above two cases, we then prove

$$\angle(\boldsymbol{v}_a, \boldsymbol{v}_b) \leq \angle(\boldsymbol{v}_a, \boldsymbol{v}_c) + \angle(\boldsymbol{v}_b, \boldsymbol{v}_c) \tag{A.18}$$

□

Appendix B

Convex Function $f(x) = \arccos(x)^2$

To prove lemma 2.6, we introduce a convex function $f(x) = \arccos(x)^2$. In this part, we show that the $f(x) = \arccos(x)^2$ is convex in detail.

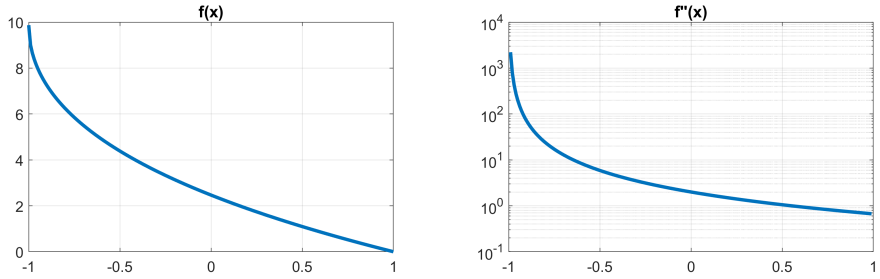


Figure B.1: Illustration of $f(x)$ and $f''(x)$

Observe,

$$f(x) = \arccos(x)^2 \tag{B.1}$$

$$f'(x) = -2\frac{\arccos(x)}{\sqrt{1-x^2}} \tag{B.2}$$

$$f''(x) = \frac{2\sqrt{1-x^2} - 2x\arccos(x)}{(1-x^2)^{3/2}} \tag{B.3}$$

To prove $f(x)$ is convex in $[-1, 1]$, it is sufficient to prove $f''(x) \geq 0$ in $(-1, 1)$. Therefore, we need to prove

$$\sqrt{1-x^2} - x\arccos(x) \geq 0, \quad x \in (-1, 1) \tag{B.4}$$

Let

$$g(x) = \sqrt{1-x^2} - x\arccos(x) \tag{B.5}$$

Then

$$g'(x) = -\arccos(x) \tag{B.6}$$

Observe, $g(1) = 0$ and

$$g'(x) \leq 0, \quad x \in [-1, 1] \tag{B.7}$$

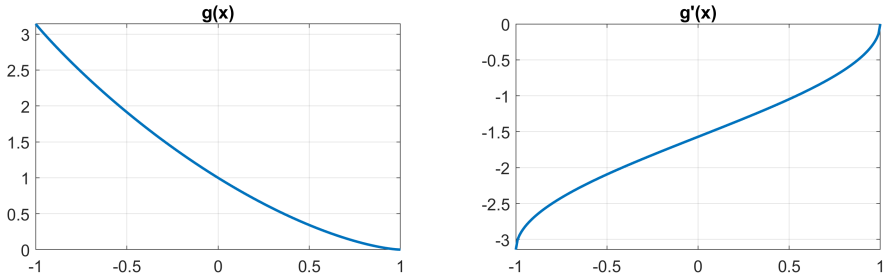


Figure B.2: Illustration of $g(x)$ and $g'(x)$

According to *mean value theorem*, $\exists c \in [a, b]$,

$$g'(c) = \frac{g(b) - g(a)}{b - a} \tag{B.8}$$

Therefore, for $x \in (-1, 1)$, there is

$$g'(c) = \frac{g(1) - g(x)}{1 - x} \leq 0 \tag{B.9}$$

Since $g(1) = 0$, and $1 - x \geq 0$, then

$$g(x) \geq 0 \quad x \in (-1, 1) \tag{B.10}$$

Equivalently,

$$\sqrt{1 - x^2} - x \arccos(x) \geq 0, \quad x \in (-1, 1) \tag{B.11}$$

Then

$$f''(x) = \frac{2\sqrt{1 - x^2} - 2x \arccos(x)}{(1 - x^2)^{3/2}} \geq 0, \quad x \in (-1, 1) \tag{B.12}$$

Therefore, $f(x) = \arccos(x)^2$ is a convex function in $[-1, 1]$.

Bibliography

[1] J. C. Knight. Safety critical systems: challenges and directions. In *Proceedings of the 24th international conference on software engineering*, pages 547–550, 2002.

[2] J. Athavale, A. Baldovin, R. Graefe, M. Paulitsch, and R. Rosales. Ai and reliability trends in safety-critical autonomous systems on ground and air. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pages 74–77. IEEE, 2020.

[3] M. Rabe, S. Milz, and P. Mader. Development methodologies for safety critical machine learning applications in the automotive domain: A survey. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 129–141, 2021.

[4] N. Agarwal, C.-W. Chiang, and A. Sharma. A study on computer vision techniques for self-driving cars. In *International Conference on Frontier Computing*, pages 629–634. Springer, 2018.

[5] X. Song, P. Wang, D. Zhou, R. Zhu, C. Guan, Y. Dai, H. Su, H. Li, and R. Yang. Apollocar3d: A large 3d car instance understanding benchmark for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5452–5462, 2019.

[6] P. Lin. Tesla autopilot crash: Why we should worry about a single death. *IEEE Spectrum: Technology, Engineering, and Science News*, 2016.

[7] 2 killed in driverless tesla car crash, officials say. URL: `https://www.nytimes.com/2021/04/18/business/tesla-fatal-crash-texas.html`.

[8] Tesla needs to fix its deadly autopilot problem. URL: `https://www.vox.com/recode/2020/2/26/21154502/tesla-autopilot-fatal-crashes`.

[9] T.-J. C. Luca Carlone, F. K. Anders Eriksson, and H. Yang. Global optimization for geometric understanding with provable guarantees. URL: `https://mit-spark.github.io/GlobalOptimization-ICCV2019/`.

[10] R. Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[11] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[12] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li. T-gcn: A temporal graph convolutional network for traffic prediction. *IEEE Transactions on Intelligent Transportation Systems*, 21(9):3848–3858, 2019.

[13] K. Muhammad, A. Ullah, J. Lloret, J. Del Ser, and V. H. C. de Albuquerque. Deep learning for safe autonomous driving: Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[14] A. Pugh. *Robot vision*. Springer Science & Business Media, 2013.

[15] V. Kyrki. Computer and robot vision. *IEEE robotics & automation magazine*, 18(2):121–122, 2011.

[16] C.-h. Chen. *Computer vision in medical imaging*, volume 2. World scientific, 2013.

[17] T. M. Ward, P. Mascagni, Y. Ban, G. Rosman, N. Padoy, O. Meireles, and D. A. Hashimoto. Computer vision in surgery. *Surgery*, 169(5):1253–1256, 2021.

[18] L. R. Kennedy-Metz, P. Mascagni, A. Torralba, R. D. Dias, P. Perona, J. A. Shah, N. Padoy, and M. A. Zenati. Computer vision in the operating room: Opportunities and caveats. *IEEE transactions on medical robotics and bionics*, 3(1):2–10, 2020.

[19] D. Dong, X. Li, and X. Sun. A vision-based method for improving the safety of self-driving. In *2018 12th International Conference on Reliability, Maintainability, and Safety (ICRMS)*, pages 167–171. IEEE, 2018.

[20] A. Sagar and R. Soundrapandiyan. Semantic segmentation with multi scale spatial attention for self driving cars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2650–2656, 2021.

[21] G. Hee Lee, F. Faundorfer, and M. Pollefeys. Motion estimation for self-driving cars with a generalized camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2746–2753, 2013.

[22] P. Kohli and A. Chadha. Enabling pedestrian safety using computer vision techniques: A case study of the 2018 uber inc. self-driving car crash. In *Future of Information and Communication Conference*, pages 261–279. Springer, 2019.

[23] R. Kulkarni, S. Dhavalikar, and S. Bangar. Traffic light detection and recognition for self driving cars using deep learning. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–4. IEEE, 2018.

[24] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[25] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.

[26] M. Garcia-Salguero, J. Briales, and J. Gonzalez-Jimenez. Certifiable relative pose estimation. *Image and Vision Computing*, 109:104142, 2021.

[27] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004.

[28] M. Hassaballah, A. A. Abdelmgeid, and H. A. Alshazly. Image features detection, description and matching. In *Image Feature Detectors and Descriptors*, pages 11–45. Springer, 2016.

[29] T.-J. Chin and D. Suter. The maximum consensus problem: recent algorithmic advances. *Synthesis Lectures on Computer Vision*, 7(2):1–194, 2017.

[30] V. Tzoumas, P. Antonante, and L. Carlone. Outlier-robust spatial perception: Hardness, general-purpose algorithms, and guarantees. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5383–5390. IEEE, 2019.

[31] C. C. Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer, 2017.

[32] J. Frost. *Introduction to Statistics: An Intuitive Guide for Analyzing Data and*. 2019.

[33] H. Zhou, T. Zhang, and J. Jagadeesan. Re-weighting and 1-point ransac-based p n n p solution to handle outliers. *IEEE transactions on pattern analysis and machine intelligence*, 41(12):3022–3033, 2018.

[34] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[35] H. Li. Consensus set maximization with guaranteed global optimality for robust geometry estimation. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1074–1080. IEEE, 2009.

[36] M.-H. Lin, J.-F. Tsai, and C.-S. Yu. A review of deterministic optimization methods in engineering and management. *Mathematical Problems in Engineering*, 2012, 2012.

[37] C. Olsson, A. Eriksson, and R. Hartley. Outlier removal using duality. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1450–1457. IEEE, 2010.

[38] H. Yang and L. Carlone. A quaternion-based certifiably optimal solution to the wahba problem with outliers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1665–1674, 2019.

[39] W. Liu and W. Lin. Additive white gaussian noise level estimation in svd domain for images. *IEEE Transactions on Image processing*, 22(3):872–883, 2012.

[40] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2344–2351, 2013.

[41] D. De Menezes, D. M. Prata, A. R. Secchi, and J. C. Pinto. A review on robust m-estimators for regression analysis. *Computers & Chemical Engineering*, page 107254, 2021.

[42] A. Ruckstuhl. Robust fitting of parametric models based on m-estimation. *Lecture notes*, page 40, 2014.

[43] K. Aftab and R. Hartley. Convergence of iteratively re-weighted least squares to robust m-estimators. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 480–487. IEEE, 2015.

[44] G. Shevlyakov, S. Morgenthaler, and A. Shurygin. Redescending m-estimators. *Journal of Statistical Planning and Inference*, 138(10):2906–2917, 2008.

[45] P. W. Holland and R. E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics-theory and Methods*, 6(9):813–827, 1977.

[46] R. Wolke and H. Schwetlick. Iteratively reweighted least squares: algorithms, convergence analysis, and numerical comparisons. *SIAM journal on scientific and statistical computing*, 9(5):907–921, 1988.

[47] L. Magri and A. Fusiello. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3954–3961, 2014.

[48] G. J. McLachlan and T. Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2007.

[49] H. Li and R. Hartley. The 3d-3d registration problem revisited. In *2007 IEEE 11th international conference on computer vision*, pages 1–8. IEEE, 2007.

[50] C. Ravazzi and E. Magli. Fast and robust em-based irls algorithm for sparse signal recovery from noisy measurements. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3841–3845. IEEE, 2015.

[51] C. Jin, Y. Zhang, S. Balakrishnan, M. J. Wainwright, and M. I. Jordan. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. *Advances in neural information processing systems*, 29:4116–4124, 2016.

[52] D. Campbell, L. Petersson, L. Kneip, and H. Li. Globally-optimal inlier set maximisation for camera pose and correspondence estimation. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):328–342, 2018.

[53] J.-C. Bazin, Y. Seo, R. Hartley, and M. Pollefeys. Globally optimal inlier set maximization with unknown rotation and focal length. In *European Conference on Computer Vision*, pages 803–817. Springer, 2014.

[54] T. Probst, D. P. Paudel, A. Chhatkuli, and L. V. Gool. Unsupervised learning of consensus maximization for 3d vision problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 929–938, 2019.

[55] V. Fragoso, C. Sweeney, P. Sen, and M. Turk. Ansac: Adaptive non-minimal sample and consensus. In *British Machine Vision Conference*, 2017.

[56] R. B. Tennakoon, A. Bab-Hadiashar, Z. Cao, R. Hoseinnezhad, and D. Suter. Robust model fitting using higher than minimal subset sampling. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):350–362, 2015.

[57] H. M. Le, T.-J. Chin, A. Eriksson, T.-T. Do, and D. Suter. Deterministic approximate methods for maximum consensus robust fitting. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[58] T.-J. Chin, A. Eriksson, and Y. Matsushita. Optimisation methods in geometric vision. 2019.

[59] C. Xu, L. Zhang, L. Cheng, and R. Koch. Pose estimation from line correspondences: A complete analysis and a series of solutions. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1209–1222, 2016.

[60] S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[61] C. A. Floudas. *Deterministic global optimization: theory, methods and applications*, volume 37. Springer Science & Business Media, 2013.

[62] D. Scholz. *Deterministic global optimization: geometric branch-and-bound methods and their applications*, volume 63. Springer Science & Business Media, 2011.

[63] K.-C. Hu, C.-W. Tsai, and M.-C. Chiang. A multiple-search multi-start framework for metaheuristics for clustering problems. *IEEE Access*, 8:96173–96183, 2020.

[64] O. Kramer. Genetic algorithms. In *Genetic algorithm essentials*, pages 11–19. Springer, 2017.

[65] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR 2011*, pages 1201–1208. IEEE, 2011.

[66] J. Yang, H. Li, D. Campbell, and Y. Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE transactions on pattern analysis and machine intelligence*, 38(11):2241–2254, 2015.

[67] D. Campbell and L. Petersson. Gogma: Globally-optimal gaussian mixture alignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5685–5694, 2016.

[68] C. Olsson, F. Kahl, and M. Oskarsson. Branch-and-bound methods for euclidean registration problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):783–794, 2008.

[69] M. Brown, D. Windridge, and J.-Y. Guillemaut. A family of globally optimal branch-and-bound algorithms for 2d–3d correspondence-free registration. *Pattern Recognition*, 93:36–54, 2019.

[70] M. Chandraker, S. Agarwal, D. Kriegman, and S. Belongie. Globally optimal algorithms for stratified autocalibration. *International journal of computer vision*, 90(2):236–254, 2010.

[71] J. Heller, M. Havlena, and T. Pajdla. Globally optimal hand-eye calibration using branch-and-bound. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(5):1027–1033, 2015.

[72] D. Liu, A. Parra, and T.-J. Chin. Globally optimal contrast maximisation for event-based motion estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6349–6358, 2020.

[73] X. Peng, L. Gao, Y. Wang, and L. Kneip. Globally-optimal contrast maximisation for event cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[74] A. Owen-Hill. Robot vision vs computer vision: What's the difference? URL: `https://blog.robotiq.com/robot-vision-vs-computer-vision-whats-the-difference`.

[75] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[76] E. Oja and L. Wang. Robust fitting by nonlinear neural units. *Neural networks*, 9(3):435–444, 1996.

[77] Y. Tong, L. Yu, S. Li, J. Liu, H. Qin, and W. Li. Polynomial fitting algorithm based on neural network. *ASP Transactions on Pattern Recognition and Intelligent Systems*, 1(1):32–39, 2021.

[78] C. Tat-Jun, C. Zhipeng, and F. Neumann. Robust fitting in computer vision: Easy or hard? *International Journal of Computer Vision*, 128(3):575–587, 2020.

[79] P. Jain and P. Kar. Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, 10(3-4):142–336, 2017.

[80] J. Sun. *When are nonconvex optimization problems not scary?* Columbia University, 2016.

[81] Y. Liu, G. Chen, and A. Knoll. Globally optimal vertical direction estimation in atlanta world. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[82] Y. Liu, G. Chen, and A. Knoll. Globally optimal camera orientation estimation from line correspondences by bnb algorithm. *IEEE Robotics and Automation Letters*, 6(1):215–222, 2020.

[83] Á. J. P. Bustos. *Robust Rotation Search in Computer Vision*. PhD thesis, University of Adelaide, School of Computer Science, 2016.

[84] Y.-B. Jia. Quaternions and rotations. *Com S*, 477(577):15, 2008.

[85] S. L. Altmann. *Rotations, quaternions, and double groups*. Courier Corporation, 2005.

[86] Y. Liu, G. Chen, R. Gu, and A. Knoll. Globally optimal consensus maximization for relative pose estimation with known gravity direction. *IEEE Robotics and Automation Letters*, 2021.

[87] D. Kalman. Leveling with lagrange: An alternate view of constrained optimization. *Mathematics Magazine*, 82(3):186–196, 2009.

[88] F. Rossi, P. Van Beek, and T. Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[89] C. Udriste. *Convex functions and optimization methods on Riemannian manifolds*, volume 297. Springer Science & Business Media, 2013.

[90] W. Ring and B. Wirth. Optimization methods on riemannian manifolds and their application to shape space. *SIAM Journal on Optimization*, 22(2):596–627, 2012.

[91] J. Brownlee. Local optimization versus global optimization. URL: `https://machinelearningmastery.com/local-optimization-versus-global-optimization/`.

[92] R. I. Hartley and F. Kahl. Global optimization through rotation space search. *International Journal of Computer Vision*, 82(1):64–79, 2009.

[93] K. Joo, T.-H. Oh, J. Kim, and I. S. Kweon. Robust and globally optimal manhattan frame estimation in near real time. *IEEE transactions on pattern analysis and machine intelligence*, 41(3):682–696, 2018.

[94] Y. Liu, X. Li, M. Wang, A. Knoll, G. Chen, and Z. Song. A novel method for the absolute pose problem with pairwise constraints. *Remote Sensing*, 11(24):3007, 2019.

[95] A. H. Land and A. G. Doig. An automatic method for solving discrete programming problems. In *50 Years of Integer Programming 1958-2008*, pages 105–132. Springer, 2010.

[96] J. Clausen. Branch and bound algorithms-principles and examples. *Department of Computer Science, University of Copenhagen*, pages 1–30, 1999.

[97] C. P. Tomazella and M. S. Nagano. A comprehensive review of branch-and-bound algorithms: Guidelines and directions for further research on the flowshop scheduling problem. *Expert Systems with Applications*, 158:113556, 2020.

[98] S. Violina. Analysis of brute force and branch & bound algorithms to solve the traveling salesperson problem (tsp). *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(8):1226–1229, 2021.

[99] L. Mitten. Branch-and-bound methods: General formulation and properties. *Operations Research*, 18(1):24–34, 1970.

[100] S. Boyd and J. Mattingley. Branch and bound methods. *Notes for EE364b, Stanford University*, pages 2006–07, 2007. URL: `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.178.2197&rep=rep1&type=pdf`.

[101] W. Lian, L. Zhang, and M.-H. Yang. An efficient globally optimal algorithm for asymmetric point matching. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1281–1293, 2016.

[102] R. E. Moore, R. B. Kearfott, and M. J. Cloud. *Introduction to interval analysis*. SIAM, 2009.

[103] J.-C. Bazin, Y. Seo, and M. Pollefeys. Globally optimal consensus set maximization through rotation search. In *Asian Conference on Computer Vision*, pages 539–551. Springer, 2012.

[104] Y. Liu, Y. Dong, Z. Song, and M. Wang. 2d-3d point set registration based on global rotation search. *IEEE Transactions on Image Processing*, 28(5):2599–2613, 2018.

[105] D. Q. Huynh. Metrics for 3d rotations: Comparison and analysis. *Journal of Mathematical Imaging and Vision*, 35(2):155–164, 2009.

[106] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *International journal of computer vision*, 103(3):267–305, 2013.

[107] C. J. Atkin. Weakly compact lie groups, 2004. URL: `https://homepages.ecs.vuw.ac.nz/~mathmeet/vic2004/slides/VIC2004_Chris_Atkin.pdf`.

[108] T. Ruland. On the relation between two rotation metrics. *CoRR*, abs/1512.04219, 2015. URL: `http://arxiv.org/abs/1512.04219`, `arXiv:1512.04219`.

[109] K. Joo, H. Li, T.-H. Oh, and I. S. Kweon. Robust and efficient estimation of relative pose for cameras on selfie sticks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[110] B. Hall. *Lie groups, Lie algebras, and representations: an elementary introduction*, volume 222. Springer, 2015.

[111] J. Deray and J. Solà. Manif: A micro lie theory library for state estimation in robotics applications. 2020.

[112] J. Straub, O. Freifeld, G. Rosman, J. J. Leonard, and J. W. Fisher. The manhattan frame model-manhattan world inference in the space of surface normals. *IEEE transactions on pattern analysis and machine intelligence*, 40(1):235–249, 2018.

[113] G. Schindler and F. Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I. IEEE, 2004.

[114] K. Joo, T.-H. Oh, I. So Kweon, and J.-C. Bazin. Globally optimal inlier set maximization for atlanta frame estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5726–5734, 2018.

[115] K. Joo, T.-H. Oh, I. S. Kweon, and J.-C. Bazin. Globally optimal inlier set maximization for atlanta world understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2656–2669, 2019.

[116] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *2009 IEEE 12th international conference on computer vision*, pages 1849–1856. IEEE, 2009.

[117] N. Sünderhauf and P. Protzel. Switchable constraints for robust pose graph slam. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1879–1884. IEEE, 2012.

[118] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu. Structslam: Visual slam with building structure lines. *IEEE Transactions on Vehicular Technology*, 64(4):1364–1375, 2015.

[119] L. Magri and A. Fusiello. Multiple model fitting as a set coverage problem. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3318–3326, 2016.

[120] D. Barath and J. Matas. Multi-class model fitting by energy minimization and mode-seeking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 221–236, 2018.

[121] P. Amayo, P. Piniés, L. M. Paz, and P. Newman. Geometric multi-model fitting with a convex relaxation algorithm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8138–8146, 2018.

[122] S. Kim and R. Manduchi. Multi-planar fitting in an indoor manhattanworld. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 11–19. IEEE, 2017.

[123] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1250–1257. IEEE, 2009.

[124] J.-C. Bazin, Y. Seo, C. Demonceaux, P. Vasseur, K. Ikeuchi, I. Kweon, and M. Pollefeys. Globally optimal line clustering and vanishing point estimation in manhattan world. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 638–645. IEEE, 2012.

[125] M. Köppen. The curse of dimensionality. In *5th Online World Conference on Soft Computing in Industrial Applications (WSC5)*, volume 1, pages 4–8, 2000.

[126] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 564–571, 2013.

[127] C. J. Taylor and A. Cowley. Parsing indoor scenes using rgb-d imagery. In *Robotics: Science and Systems*, volume 8, pages 401–408, 2013.

[128] Z. Cai, T.-J. Chin, A. P. Bustos, and K. Schindler. Practical optimal registration of terrestrial lidar scan pairs. *ISPRS journal of photogrammetry and remote sensing*, 147:118–131, 2019.

[129] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[130] G. H. Lee. Line association and vanishing point estimation with binary quadratic programming. In *2017 International Conference on 3D Vision (3DV)*, pages 584–592. IEEE, 2017.

[131] M. Antunes and J. P. Barreto. A global approach for the detection of vanishing points and mutually orthogonal vanishing directions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1336–1343, 2013.

[132] R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. In *European conference on computer vision*, pages 537–547. Springer, 2008.

[133] Y. Seo, Y.-J. Choi, and S. W. Lee. A branch-and-bound algorithm for globally optimal calibration of a camera-and-rotation-sensor system. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1173–1178. IEEE, 2009.

[134] J. Heller, M. Havlena, and T. Pajdla. Globally optimal hand-eye calibration using branch-and-bound. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(5):1027–1033, 2016.

[135] J. Yang, H. Li, and Y. Jia. Optimal essential matrix estimation via inlier-set maximization. In *European Conference on Computer Vision*, pages 111–126. Springer, 2014.

[136] Á. P. Bustos, T.-J. Chin, A. Eriksson, H. Li, and D. Suter. Fast rotation search with stereographic projections for 3d registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2227–2240, 2016.

[137] J. Straub, T. Campbell, J. P. How, and J. W. Fisher, III. Efficient global point cloud alignment using bayesian nonparametric mixtures. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[138] N. I. Fisher. *Statistical Analysis of Circular Data*. 1993.

[139] D. S. Johnson and F. P. Preparata. The densest hemisphere problem. *Theoretical Computer Science*, 6(1):93–107, 1977.

[140] Y. Feng, G.-G. Wang, S. Deb, M. Lu, and X.-J. Zhao. Solving 0–1 knapsack problem by a novel binary monarch butterfly optimization. *Neural computing and applications*, 28(7):1619–1634, 2017.

[141] G. Yuan and B. Ghanem. Binary optimization via mathematical programming with equilibrium constraints. *arXiv preprint arXiv:1608.04425*, 2016.

[142] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102, 2016.

[143] H. Abbaspour and M. Moskowitz. *Basic Lie Theory*. World Scientific Publishing Company, 2007.

[144] D. R. Wilkins. Möbius transformations and stereographic projection. URL: `https://www.maths.tcd.ie/~dwilkins/Courses/MA232A/MA232A_Mich2017/StWrapper.pdf`.

[145] T. Needham. *Visual complex analysis*. Oxford University Press, 1998.

[146] Á. P. Bustos and T.-J. Chin. Guaranteed outlier removal for point cloud registration with correspondences. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2868–2882, 2017.

[147] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[148] T. Lemaire and S. Lacroix. Monocular-vision based slam using line segments. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 2791–2796. IEEE, 2007.

[149] N. Andreff, B. Espiau, and R. Horaud. Visual servoing from lines. *The International Journal of Robotics Research*, 21(8):679–699, 2002.

[150] L. Lecrosnier, R. Boutteau, P. Vasseur, X. Savatier, and F. Fraundorfer. Camera pose estimation based on pnl with a known vertical direction. *IEEE Robotics and Automation Letters*, 4(4):3852–3859, 2019.

[151] B. Přibyl, P. Zemčík, and M. Čadík. Absolute pose estimation from line correspondences using direct linear transformation. *Computer Vision and Image Understanding*, 161:130–144, 2017.

[152] H. Yu, W. Zhen, W. Yang, and S. A. Scherer. Line-based camera pose estimation in point cloud of structured environments. *CoRR*, abs/1912.05013, 2019. URL: `http://arxiv.org/abs/1912.05013`, `arXiv:1912.05013`.

[153] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard. Se-sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125, 2019.

[154] J. Fredriksson, V. Larsson, C. Olsson, O. Enqvist, and F. Kahl. Efficient algorithms for robust estimation of relative translation. *Image and Vision Computing*, 52:114–124, 2016.

[155] Y. Liu, X. Li, M. Wang, A. Knoll, G. Chen, and Z. Song. A novel method for the absolute pose problem with pairwise constraints. *Remote Sensing*, 11(24):3007, 2019.

[156] N. Horanyi and Z. Kato. Multiview absolute pose using 3d-2d perspective line correspondences and vertical direction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2472–2480, 2017.

[157] L. Kneip, H. Li, and Y. Seo. Upnp: An optimal o (n) solution to the absolute pose problem with universal applicability. In *European Conference on Computer Vision*, pages 127–142. Springer, 2014.

[158] S. Hadfield, K. Lebeda, and R. Bowden. Hard-pnp: Pnp optimization using a hybrid approximate representation. *IEEE transactions on pattern analysis and machine intelligence*, 41(3):768–774, 2018.

[159] A. Vakhitov, J. Funke, and F. Moreno-Noguer. Accurate and linear time pose estimation from points and lines. In *European Conference on Computer Vision*, pages 583–599. Springer, 2016.

[160] P. Miraldo, T. Dias, and S. Ramalingam. A minimal closed-form solution for multi-perspective pose estimation using points and lines. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 474–490, 2018.

[161] S. Choi, T. Kim, and W. Yu. Performance evaluation of ransac family. In *Proceedings of the British Machine Vision Conference*, pages 81.1–81.12. BMVA Press, 2009. doi:10.5244/C.23.81.

[162] Z. Cai, T.-J. Chin, and V. Koltun. Consensus maximization tree search revisited. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1637–1645, 2019.

[163] Z. Cai, T.-J. Chin, H. Le, and D. Suter. Deterministic consensus maximization with biconvex programming. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 685–700, 2018.

[164] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2017.

[165] E. Brachmann and C. Rother. Neural-guided ransac: Learning where to sample model hypotheses. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4322–4331, 2019.

[166] R. Kumar and A. R. Hanson. Robust methods for estimating pose and a sensitivity analysis. *CVGIP: Image understanding*, 60(3):313–342, 1994.

[167] Y. Liu, T. S. Huang, and O. D. Faugeras. Determination of camera location from 2-d to 3-d line and point correspondences. *IEEE Transactions on pattern analysis and machine intelligence*, 12(1):28–37, 1990.

[168] Y. Zhang, X. Li, H. Liu, and Y. Shang. Probabilistic approach for maximum likelihood estimation of pose using lines. *IET Computer Vision*, 10(6):475–482, 2016.

[169] H. Abdellali, R. Frohlich, and Z. Kato. A direct least-squares solution to multi-view absolute and relative pose from 2d-3d perspective line pairs. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[170] A. Ansar and K. Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003.

[171] P. Wang, G. Xu, Y. Cheng, and Q. Yu. Camera pose estimation from lines: a fast, robust and general method. *Machine Vision and Applications*, pages 1–12, 2019.

[172] L. Ferraz, X. Binefa, and F. Moreno-Noguer. Very fast solution to the pnp problem with algebraic outlier rejection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 501–508, 2014.

[173] H. Abdellali and Z. Kato. Absolute and relative pose estimation of a multi-view camera system using 2d-3d line pairs and vertical direction. In *2018*

Digital Image Computing: Techniques and Applications (DICTA), pages 1–8. IEEE, 2018.

[174] N. Horanyi and Z. Kato. Generalized pose estimation from line correspondences with known vertical direction. In *2017 International Conference on 3D Vision (3DV)*, pages 244–253. IEEE, 2017.

[175] L. Lecrosnier, R. Boutteau, P. Vasseur, X. Savatier, and F. Fraundorfer. Vision based vehicle relocalization in 3d line-feature map using perspective-n-line with a known vertical direction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1263–1269. IEEE, 2019.

[176] F. M. Mirzaei and S. I. Roumeliotis. Globally optimal pose estimation from line correspondences. In *2011 IEEE International Conference on Robotics and Automation*, pages 5581–5588. IEEE, 2011.

[177] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102, 2016.

[178] W. A. De Graaf. *Lie algebras: theory and algorithms*. Elsevier, 2000.

[179] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. 1982. URL: `https://academic.microsoft.com/paper/1669104078`.

[180] J. Richter-Gebert. *Perspectives on projective geometry: a guided tour through real and complex geometry*. Springer Science & Business Media, 2011.

[181] H. Li and R. Hartley. Five-point motion estimation made easy. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 1, pages 630–633. IEEE, 2006.

[182] R. Hartley and H. Li. An efficient hidden variable approach to minimal-case camera motion estimation. *IEEE transactions on pattern analysis and machine intelligence*, 34(12):2303–2314, 2012.

[183] P. H. Torr and A. Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer vision and image understanding*, 78(1):138–156, 2000.

[184] L. Zhang, C. Xu, K.-M. Lee, and R. Koch. Robust and efficient pose estimation from line correspondences. In *Asian Conference on Computer Vision*, pages 217–230. Springer, 2012.

[185] T. Moons, L. Van Gool, and M. Vergauwen. *3D Reconstruction from Multiple Images: Principles*. Now Publishers Inc, 2009.

[186] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[187] O. Saurer, P. Vasseur, R. Boutteau, C. Demonceaux, M. Pollefeys, and F. Fraundorfer. Homography based egomotion estimation with a common direction. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):327–341, 2016.

[188] L. Svärm, O. Enqvist, F. Kahl, and M. Oskarsson. City-scale localization for cameras with known vertical direction. *IEEE transactions on pattern analysis and machine intelligence*, 39(7):1455–1461, 2016.

[189] O. Naroditsky, X. S. Zhou, J. Gallier, S. I. Roumeliotis, and K. Daniilidis. Two efficient solutions for visual odometry using directional correspondence. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):818–824, 2011.

[190] M. Kalantari, A. Hashemi, F. Jung, and J.-P. Guédon. A new solution to the relative orientation problem using only 3 points and the vertical direction. *Journal of Mathematical Imaging and Vision*, 39(3):259–268, 2011.

[191] C. Sweeney, J. Flynn, and M. Turk. Solving for relative pose with a partially known rotation is a quadratic eigenvalue problem. In *2014 2nd International Conference on 3D Vision*, volume 1, pages 483–490. IEEE, 2014.

[192] Z. Kukelova, M. Bujnak, and T. Pajdla. Closed-form solutions to minimal absolute pose problems with known vertical direction. In *Asian Conference on Computer Vision*, pages 216–229. Springer, 2010.

[193] F. Fraundorfer, P. Tanskanen, and M. Pollefeys. A minimal case solution to the calibrated relative pose problem for the case of two known orientation angles. In *European Conference on Computer Vision*, pages 269–282. Springer, 2010.

[194] Y. Ding, J. Yang, and H. Kong. An efficient solution to the relative pose estimation with a common direction. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11053–11059. IEEE, 2020.

[195] Y. Ding, J. Yang, J. Ponce, and H. Kong. Minimal solutions to relative pose estimation from two views sharing a common direction with unknown focal length. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[196] Y. Ding, D. Barath, J. Yang, H. Kong, and Z. Kukelova. Globally optimal relative pose estimation with gravity prior, 2020. `arXiv:2012.00458`.

[197] K. Joo, H. Li, T.-H. Oh, Y. Bok, and I. S. Kweon. Globally optimal relative pose estimation for camera on a selfie stick. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4983–4989. IEEE, 2020.

[198] J. Fredriksson, V. Larsson, C. Olsson, and F. Kahl. Optimal relative pose with unknown correspondences. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1728–1736, 2016.

[199] J. Fredriksson, O. Enqvist, and F. Kahl. Fast and reliable two-view translation estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[200] J. Fredriksson, V. Larsson, and C. Olsson. Practical robust two-view translation estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2684–2690, 2015.

[201] L. Gao, J. Su, J. Cui, X. Zeng, X. Peng, and L. Kneip. Efficient globally-optimal correspondence-less visual odometry for planar ground vehicles. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2696–2702, 2020. doi:10.1109/ICRA40945.2020.9196595.

[202] T.-J. Chin, P. Purkait, A. Eriksson, and D. Suter. Efficient globally optimal consensus maximisation with tree search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):758–772, 2017.

[203] Y. Ding, J. Yang, J. Ponce, and H. Kong. Homography-based minimal-case relative pose estimation with known gravity direction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[204] B. Guan, J. Zhao, Z. Li, F. Sun, and F. Fraundorfer. Minimal solutions for relative pose with a single affine correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[205] C. Albl, Z. Kukelova, and T. Pajdla. Rolling shutter absolute pose problem with known vertical direction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[206] H. Abdellali, R. Frohlich, and Z. Kato. Robust absolute and relative pose estimation of a central camera system from 2d-3d line correspondences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[207] Á. Parra Bustos and T.-J. Chin. Guaranteed outlier removal for point cloud registration with correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2868–2882, 2018.

[208] Mean value theorem. URL: `https://en.wikipedia.org/wiki/Mean_value_theorem`.

[209] J. Matkowski. Mean-value theorem for vector-valued functions. *Mathematica Bohemica*, 137(4):415–423, 2012.

[210] W. S. Hall and M. L. Newell. The mean value theorem for vector valued functions: a simple proof. *Mathematics Magazine*, 52(3):157–158, 1979.

[211] P. Hansen and B. Jaumard. Lipschitz optimization. In *Handbook of global optimization*, pages 407–493. Springer, 1995.

[212] E. Hansen and G. W. Walster. *Global optimization using interval analysis: revised and expanded*, volume 264. CRC Press, 2003.

[213] W. Zhang. Branch-and-bound search algorithms and their computational complexity. Technical report, UNIVERSITY OF SOUTHERN CALIFORNIA MARINA DEL REY INFORMATION SCIENCES INST, 1996.

[214] K. Eriksson, D. Estep, and C. Johnson. Lipschitz continuity. In *Applied Mathematics: Body and Soul*, pages 149–164. Springer, 2004.

[215] G. J. Minty. On the extension of lipschitz, lipschitz-hölder continuous, and monotone functions. *Bulletin of the American Mathematical Society*, 76(2):334–339, 1970.

[216] N. Dym. Quasi branch and bound for smooth global optimization. *arXiv preprint arXiv:2005.13728*, 2020.

[217] N. Dym and S. Z. Kovalsky. Linearly converging quasi branch and bound algorithms for global rigid registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1628–1636, 2019.

[218] S. Saitoh. Generalizations of the triangle inequality. *J. Inequal. Pure Appl. Math*, 4(3):5, 2003.

[219] M. A. Khamsi and W. A. Kirk. *An introduction to metric spaces and fixed point theory*, volume 53. John Wiley & Sons, 2011.

[220] J. M. Steele. *The Cauchy-Schwarz master class: an introduction to the art of mathematical inequalities*. Cambridge University Press, 2004.

[221] R. Kantrowitz and M. M. Neumann. Yet another proof of minkowski's inequality. *The American Mathematical Monthly*, 115(5):445–447, 2008.

[222] V. Bryant. *Metric spaces: iteration and application*. Cambridge University Press, 1985.

[223] D. Burago, I. D. Burago, Y. Burago, S. Ivanov, S. V. Ivanov, and S. A. Ivanov. *A course in metric geometry*, volume 33. American Mathematical Soc., 2001.

[224] S. Drury. Honours analysis 3, math 354, metric spaces. calculus in several variables and to some extent on banach spaces, 2019. URL: `https://www.math.mcgill.ca/drury/teaching/teaching.php`.

[225] R. Johnson. Spherical trigonometry. *West Hills Institute of Mathematics*.