Robust Graph-Based Localization and Mapping

Pratik Agarwal

Technische Fakultät
Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Dissertation zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften

Betreuer: Prof. Dr. Wolfram Burgard

Robust Graph-Based Localization and Mapping

Pratik Agarwal

Dissertation zur Erlangung des akademischen Grades Doktor der Naturwissenschaften
Technische Fakultät, Albert-Ludwigs-Universität Freiburg

Dekan	Prof. Dr. Georg Lausen
Erstgutachter	Prof. Dr. Wolfram Burgard
Zweitgutachter	Prof. Dr. Cyrill Stachniss
Tag der Disputation	April 10, 2015

Abstract

Localization and mapping are fundamental requirements to enable mobile robots to operate robustly in their environments. The solution of these two problems will allow robots to assist humans in important tasks such as autonomous driving or hazardous search-and-rescue missions. These complex navigation tasks require a robot to build a map while simultaneously localizing itself in it. It is relatively straightforward to build a map with perfect sensors. The challenge arises when a mobile robot needs to perform both mapping and localization with noisy sensors. This requires a robot to reliably recognize a previously visited place, while being robust to noise in sensor measurements and perceptual aliasing due to repetitive structures in the environment.

In this thesis, we have proposed novel simultaneous localization and mapping (SLAM) algorithms which are more robust to place recognition and sensor errors. Our methods can recover from errors resulting from visual aliasing due to ambiguity in the environment. Additionally, our methods are more robust to gross non-Gaussian errors in sensor measurements. As map building can be cast as an optimization problem, mapping algorithms should be robust to poor initialization. Furthermore, the methods we have developed enable robots to mitigate the effects of poor initializations. In contrast to other methods that first find a good initialization to seed the optimization or that remove outliers before optimization, our method is more elegant as it does not require an additional pre-processing step.

We also survey several geodetic mapping methods with an aim at providing a geodetic perspective of SLAM, as both fields share similarities when it comes to building large maps. We show that many state-of-the-art robot mapping algorithms have a direct relation to geodetic mapping techniques developed many decades earlier. Our survey will possibly inspire new solutions to large-scale, autonomous robotic SLAM methods.

Finally, we believe that robots should be able to navigate with already available maps even if they are built for humans or for other purposes. This will save significant computational and sensor requirements on the robot, reduce the setup time, and allow them to operate in environments without previously visiting them. In this regard, we outline a novel approach to metric localization that does not require the construction of a new map when one already exists. The central idea is to leverage geotagged imagery on Google Street View as an accurate source for global positioning and localize a robot using only a monocular camera with odometric estimates.

We have released open source implementations of our methods and tested them extensively on simulated as well as real-world datasets collected on mobile robots. Our claims are supported by comparisons to existing state-of-the-art methods and verified by other researchers. We believe that our methods will bring us closer to a future with mobile robots navigating autonomously and assisting humans in their daily tasks.

Zusammenfassung

Die Fähigkeit autonom zu navigieren ist eine wesentliche Voraussetzung für mobile Roboter, um robust in unbekannten Umgebungen zu agieren. Sie versetzt Roboter in die Lage, Menschen bei vielen Aufgaben zu unterstützen: Hierzu gehören beispielsweise Transportaufgaben, autonomes Fahren oder gefährliche Such- und Rettungseinsätze. Solch komplexe Navigationsaufgaben erfordern, dass Roboter eine Karte der Umgebung erstellen und sich gleichzeitig darin lokalisieren können. Für einen Roboter ist es relativ einfach, eine Karte zu erstellen, solange er sich - beispielsweise mit fehlerfreien Sensoren - exakt lokalisieren kann. Da Sensoren in der Realität allerdings nicht perfekt sind, wird dies jedoch zu einer Herausforderung, sobald ein mobiler Roboter beide Aufgaben, d.h. die Umgebungskartierung und die Lokalisierung, gleichzeitig lösen muss. Um beide Probleme zeitgleich zu lösen, muss ein Roboter einen zuvor besuchten Ort zuverlässig wiedererkennen und dabei gleichzeitig mit Sensorrauschen und Mehrdeutigkeiten in der Wahrnehmung aufgrund sich wiederholender Umweltstrukturen umgehen können.

In dieser Arbeit haben wir Algorithmen zur gleichzeitigen Lokalisierung und Kartierung (Simultaneous Localization and Mapping, SLAM) entwickelt, die robust gegenüber Ortserkennungs- und Sensorfehlern sind. Unsere Methoden können mit groben Fehlern umgehen, die aus visuellem “Aliasing” aufgrund von Mehrdeutigkeiten in der Umgebung entstanden sind. Darüber hinaus sind unsere Methoden robust gegen extreme nicht-normalverteilte Fehler in den Sensormessungen. Da Lokalisierung und Kartierung als Optimierungsproblem verstanden werden kann, sollten die zu entwickelnden Algorithmen robust gegen schlechte Initialisierungen sein. Die von uns entwickelten Methoden sind unter anderem in der Lage, die Auswirkungen einer schlechten Initialisierung abzufedern. Im Gegensatz zu den meisten anderen Methoden, welche zunächst eine gute Initialisierung mit externen Mitteln finden oder die Ausreißer vor der Optimierung entfernen müssen, ist unsere Methode zudem eleganter, da sie keine zusätzlichen Vorverarbeitungsschritte erfordert.

Des Weiteren führen wir in dieser Arbeit auch eine geodätische Betrachtung des SLAM Problems ein, da die Struktur des SLAM Problems viele Gemeinsamkeiten mit mehreren geodätischen Kartierungsmethoden in Bezug auf die Erstellung größerer Karten aufweist. Wir zeigen, dass viele hochmoderne Kartierungsalgorithmen für Roboter einen direkten Bezug zu geodätischen Verfahren besitzen. Darüber hinaus sind wir der Meinung, dass Roboter in der Lage sein sollten, mithilfe von Karten zu navigieren, welche ursprünglich

für Menschen erstellt wurden. Dies würde die Anforderung an Rechenleistung und Sensorik erheblich reduzieren und Robotern ermöglichen, in vorher nicht-besuchten Umgebungen flexibel und selbstständig zu agieren. In diesem Zusammenhang beschreiben wir einen neuen Ansatz zur metrischen Lokalisierung, der weder die Erstellung einer konsistenten Karte noch einen vorherigen Besuch der Umgebung erfordert. Die zentrale Idee ist, georeferenzierte Bilder aus Google Street View als zentrale Quelle zur globaler Ortsbestimmung zu nutzen und einen Roboter, der nur mit einer monokularen Kamera und Odometrieschätzungen ausgestatte ist, robust und genau zu lokalisieren.

Neben der Konzeption und der Entwicklung neuer Verfahren haben wir auch Open-Source-Implementierungen unserer Methoden veröffentlicht und sie ausführlich mit simulierten sowie mit Datensätzen, die von mobilen Robotern aufgezeichnet wurden, getestet und evaluiert. Unsere Ansätze wurden mit dem aktuellen Stand der Technik verglichen und des Weiteren auch von anderen Forschern auf deren Robotern überprüft. Wir sind der Ansicht, dass unsere Methoden uns näher an eine Zukunft bringen, in der mobile Roboter autonom navigieren und Menschen bei ihrer täglichen Arbeit unterstützen.

Acknowledgements

I would like to take this opportunity to thank all the people without whom this thesis could not have been possible. Firstly, a huge shout out and thanks to my advisor Wolfram Burgard for giving me the opportunity to pursue a PhD in robotics in his lab. He has always encouraged me to pursue hard problems and provided me the freedom to perform open-ended research. I have learned a lot from him, specially on focusing the "why" question.

I would like to take this opportunity to also thank Cyrill Stachniss for being a fabulous mentor. I have learnt a lot from him, both professionally and personally, and I am sure this will be very helpful in the future.

A huge thank you to Luciano Spinello for guiding and collaborating with me on both wonderful and not-so wonderful projects. I have really enjoyed brainstorming and implementing interesting ideas with him.

I thank Nichola Abdo, Mladen Mazuran and Diego Tipaldi for listening to all my rants about the big picture and "what is the point" and for many fruitful discussions. I would also like to thank Andreas Wachaja for collaborating with me and taking the lead on iView and AAL projects to prototype a smart walker to aid visually and physically handicapped people.

Settling in Germany and managing to survive all these years without speaking German would not have been possible without the support of my awesome colleagues. Special thanks goes to Christoph Sprunk in this regard for helping me with every piece of official and non-official German letter and for resurrecting my existence in the official government records. I also thank all members of AIS specially, Markus Kuderer, Henrik Kretzschmar, Andreas Wachaja, Nichola Abdo, Barbara Frank, Bastian Steder, Michael Ruhnke and Diego Tiapaldi in this regard. Additionally, I thank Jose Mota, Christoph Sprunk, Barbara Frank and Tim Caselitz for proofreading the Zusammenfassung. I thank Michael Keser and Susanne Bourjaillat for taking care of all the administrative and unglamorous work for me.

I thank all the developers of g2o, Rainer Küemmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige and Wolfram Burgard. My research has been heavily influenced by the availability of their state-of-the-art graph optimization toolkit. I also thank Niko Sünderhauf, Yasir Latif, John Wang, Michael Kaess, Frank Dellaert, Giorgio Grisetti and Edwin Olson for their SLAM datasets and many valuable discussions.

I thank Edwin Olson for introducing the early work on Geodetic mapping to me and our librarian Susanne Hauser for locating some of the old geodetic journals and manuscripts.

I sincerely believe that this PhD thesis would not have been possible without the valuable lessons learned at the April Robotics Lab in Michigan. I thank Edwin Olson for introducing me to the field of robust SLAM and giving me the opportunity to learn from him and his students Andrew Richardson, Pradeep Ranganathan, Johannes Storm, Ryan Morton and Rob Goeddel. I also thank Madhava Krishna for allowing me to be a part of the IIIT robotics lab and get exposed to robotics research through Gururaj, Jyotika, Rahul and Abhijeet.

A huge round of thank you to Dmitri Dolgov, Jiajun Zhu and Ian Mahon at Google's self driving car team for providing me the opportunity to contribute to Chauffeur.

I would also like to extend my gratitude to my friends in Michigan Komal and Deval for supporting and motivating me through out the "ups and downs". I would also like to thank Harsh, Utsav, Soumitro, Kunal, David, Rini and Reedhima for their constant support. I thank Chit, Michael, Baharam, Lukas and Arbaaz for giving me the opportunity to supervise them on various projects. It was a huge part of my learning experience.

Above all, I would like to thank my parents and my sister for their constant support.

This work has been partially supported by BMBF under contract number 13EZ1129B-iView, by the MKW for the project ZAFL-AAL and by the EC under contract numbers ERG-AG-PE7-267686-LifeNav.

Contents

Chapter 1

Introduction

Localization and mapping are fundamental requirements for mobile robots. Robots must be able to avoid obstacles by planning safe paths to reach the desired destination in order to operate autonomously. Planning an efficient and safe path typically requires a map. Furthermore, following the path safely requires the robot to know its precise location on the map at any point in time, i.e., it must be localized. Hence, mapping new environments robustly while being accurately localized is essential for mobile robots. For this reason, simultaneous localization and mapping (SLAM) is highly relevant for uninterrupted robot operation.

Robots like the Roomba vacuum cleaner, which work in controlled environments, only need to sense the surrounding environment to avoid obstacles. This is limiting if the task at hand requires advanced planning and obstacle avoidance as in the case of autonomous robotic cars. In complex robot navigation applications, a larger detailed map is required to generate an accurate model of the environment. The detailed map will not only help to plan paths, but also help for example to predict the behavior of other agents in the world. With a detailed map, the autonomous car can anticipate intersections and crossings to reason about where to expect pedestrian, cyclists or other cars.

Building maps in itself is not new. For example, the field of photogrammetry and geodetics have researched on building large maps, spanning across continents for many decades. Autonomous mapping on the other hand contains several additional challenges. A robot needs to reliably recognize a previously visited place while being robust to perceptual aliasing due to repetitive structures in the environment. Place recognition is critical to accurate map building as a mobile robot will accumulate position errors due to noisy measurements. It is relatively straight forward to perform mapping given perfect localization and vice versa. The challenge arises when a mobile robot needs to perform both mapping and localization simultaneously when it visits a new unknown environment.

A central challenge in SLAM is to identify a previously visited place. It is difficult to distinguish two similar looking but different places. Researchers have approached this place recognition problem by developing sensor specific algorithms. These include robust features for image matching and laser scan matching algorithms. Even though

these methods are efficient, guaranteeing a *zero* place-recognition error rate is unrealistic. This is because these methods lack the semantic human understanding of a place. For example, it is difficult for a robot to identify that it is revisiting an office room if the furniture arrangement has changed. Similarly, it is challenging for a robot to differentiate between two hospital rooms which look exactly the same. Sometimes, it is even hard for humans to perform accurate place recognition. For example, differentiating between the Eiffel tower in Paris and its replica in Las Vegas might be impossible based on just their visual appearance. Any SLAM method which relies on place recognition systems to be always correct is doomed to fail eventually. Thus, we need methods which are robust to place recognition errors in order to make mobile robots fully autonomous.

SLAM algorithms must be robust to noise and errors in sensor measurements. Most approaches assume that the measurements are affected by sensor noise and model its Gaussian uncertainty, but ignore the fact that outliers (false positives) may be present. Despite improvements in sensor technology this is not true. Global Positioning Systems (GPS) provide good positional measurements outdoors, but their performance significantly deteriorates in the surrounding of tall buildings. They can report large errors due to multi-path effects which behave in a highly non Gaussian manner. Inertial measurement units (IMUs) rely on gyroscopes to estimate egomotion, but accumulate significant amount of drift with time. LIDARs used for estimating the distance to nearby objects are susceptible to reflections from shiny objects. Camera images from the same place may not resemble each other under different lighting conditions. SLAM algorithms must be able to use richer models to explain errors in sensor measurements.

Sometimes, sensors and place recognition systems can accurately narrow down the ambiguities in terms of multiple hypothesis, but cannot reliably identify the correct one amongst them. This is slightly different from the previously mentioned errors in place-recognition and sensor measurements. In these scenarios, the system may be able to determine a space of solutions - multiple hypothesis, but it cannot reliably say which one is correct. For example, GPS based triangulation in urban canyons with tall buildings leads to multiple echoes of a single signal. These multiple echoes may be used individually to enumerate all possible triangulation results. We can then conclude that the correct solution must be one of the many hypotheses generated which are mutually exclusive. Similarly, consider a scenario where the robot's wheels slip while the encoders record wheel rotations. The two hypotheses – the robot either moved by a certain distance or the robot was stuck and the wheels were rotating freely are mutually exclusive and multi-modal. The standard approach of treating all such observations as a unimodal Gaussian is limiting in such scenarios. An intelligent mapping and localization system should be able to exploit such disjoint multiple hypothesis scenarios.

One way to make SLAM algorithms robust to uncertain place recognition and errors in sensor measurements is to place visually distinct artificial features all over the en-

vironment. This will allow the robot to robustly identify its location by comparing its perception to the true positions of the artificial markers known beforehand. Such an approach has been applied to great effect by Kiva Systems. They completely control the robot environment by placing unique visual fiducials on the factory floor. The robot can localize precisely by comparing the observed fiducials to their known positions. Unfortunately, this is not practical for many real world applications, where the use of autonomous robots is highly desirable. It is not feasible to fix markers, deep under the ice-sheets in Antarctica or on the Great Barrier Reef where unmanned underwater vehicles study marine life, ocean floors and coral structures. Additionally, it may not be economically viable to place artificial markers all along our road networks to facilitate autonomous driving. Currently, autonomous robots which rely only on onboard sensors, are the only way to explore other planets and asteroids. Thus, the ability of robots to simultaneously localize and map their environments without relying on artificial features is highly relevant.

Modern SLAM algorithms iteratively linearize the non-linear problem and use optimization methods such as Gauss-Newton, Levenberg-Marquardt, or Dog-Leg to find the best configuration of the map. However, due to the non-convexity of the error surface, they cannot guarantee to find the global minimum. In practice, the initial guess has a strong impact on the quality of the computed solution. Finding the right solution can be challenging and sometimes even impossible if the initial guess is far away from the correct solution. This means that a *good* initialization, i.e., an initialization close to the correct solution is required to reach the actual solution. Thus, we need algorithms which are robust to poor initialization.

By solving the above challenges, robots will be able to autonomously navigate in unknown environments, but currently extensive maps exist only for humans navigation. Can robots leverage such maps built for humans navigation? This will save significant resources and allow robots to operate in environments without previously visiting it. For example, Google has a huge database of maps in the form of Street View, which is available for indoor as well as outdoor environments. Street View's main purpose is to provide its human users with a photo-realistic turn by turn navigation guidance. But, it is also a source of billions of geotagged images for a large portion of our planet, sometimes available across multiple seasons. If the existing maps built for humans can be re-used for robot localization, we can reduce computing and sensor requirements on the robot.

In summary, the key problems in the context of autonomous mapping and localization are

- How to handle place recognition ambiguities?
- How to deal with noise and errors in sensor measurements?
- How to exploit multi-modal sensor observations?

- How to build map optimization algorithms robust to poor initialization?
- How to localize robots, re-using existing maps built for human navigation?

In this thesis, we address these questions in the context of robust robot localization and mapping and present methods to solve the problems.

1.1 Contributions

The contribution of this thesis are novel methods for improving SLAM and making them resilient to place-recognition and sensor errors. Additionally, our methods are robust to poor initialization. We first outline max-mixtures, which allows efficient inference of mixture models. The max-mixture approach allows to explicitly model the probability of an outlier or error in data association. Additionally it allows to model multi-modal observations. Effectively, the method acts as a selector for choosing the component from the mixture with the highest probability. Our second contribution, dynamic covariance scaling (DCS), complements max-mixtures. Max-mixtures can exploit well modeled error uncertainties even when they are multi-modal. DCS, on the other hand, is more robust to measurements with erroneous uncertainties to begin with. Unlike max-mixtures, DCS models outliers by scaling the covariance of measurements with large errors. This has the effect of reducing the influence of measurements with large errors in the optimization process. This allows dynamic covariance scaling to be robust to large place recognition failures as well as poor initialization far away from the correct solution. The third contribution of this thesis is a novel robot localization technique which leverages maps built for humans. If maps do not exist, we can use max-mixtures and DCS to build new maps, but if maps exist and a robot can re-use them, it will allow them to navigate without exploring the full environment first. We develop techniques which allows localization using back-ground knowledge in the form of existing maps built for human navigation. Further, we provide a survey of geodetic mapping methods related to graph SLAM in order to point out similarities and differences between robotics and traditional mapping technologies. This survey is highly relevant to both fields as they share many challenges.

Key contributions of this thesis include:

- A survey of geodetic mapping approaches and its relationship to robotic mapping
- A method which handles multi-modal distributions
- Several methods robust to outliers and poor initialization
- A method to localize robots using maps built for humans navigation

This thesis is organized as follows. First, we provide a review of preliminary concepts of SLAM explaining recent work in front-ends and back-ends mainly using the graph-based SLAM paradigm. In Chapter 3, we provide a survey of geodetic mapping methods and its relation to robotic mapping techniques. Our survey goes back to map building methods of the late nineteenth century. The geodetic mapping community has addressed large-scale map building for centuries, computing maps that span across continents. Such techniques are relevant to robotic mapping methods and might enable scientific collaborations between both of these well-established fields. It may inspire novel solutions to large-scale autonomous robotic mapping methods.

In Chapter 4, we introduce max-mixtures, which is our first robust SLAM method. Max-mixtures is robust to data-association outliers by allowing richer error models that allow the probability of a failure to be explicitly modeled as a Gaussian with a large covariance. Our use of a mixture model, also allows multi-modal distributions to be encoded in the map building process without exponential memory complexity. This is contrast to other multi-hypothesis approaches. The ability to directly represent multi-modal distributions is a feature of our approach.

In Chapter 5, we introduce dynamic covariance scaling (DCS), which is also a robust SLAM algorithm. It is robust to large front-end outliers while at the same time, it mitigates the effects of poor initializations. Unlike max-mixtures, it does not model outliers with fixed large covariance. Instead, the covariance is dynamically scaled during optimization to best explain the observation. DCS complements max-mixtures as it is robust to erroneous distributions, while max-mixtures is able to exploit richer error distributions even when they are multi-modal. Chapter 5 provides a theoretical proof and experimental validation of robustness against outliers. In Chapter 6, we evaluate the robustness of DCS against poor initialization on many challenging datasets. Our method is more elegant than other approaches that first aim at finding a good initial guess to seed the optimization, as it does not require an additional preprocessing step.

In Chapter 7, we outline a monocular camera based metrical localization algorithm that exploits maps built for human navigation. The novelty of the method is that it does not require additional large scale map building when they already exist for localization. It is the first concrete step of moving away from building large scale maps on a robot before it can navigate or plan paths in an environment. Our method leverages Google's Street View imagery to localize monocular images without building large scale maps. The localization method developed in Chapter 7 compliments the previous robust SLAM techniques. The previous techniques, developed in Chapters 4-6, can be employed when a robot is operating in unknown environments where maps do not exist. The method developed in Chapter 7 leverages existing maps, when they exist, even if they were not built for robots.

1.2 Publications

This following peer-reviewed publications were used towards this thesis:

- E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 313–320, 2012.

- E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. *International Journal of Robotics Research (IJRR)*, 32(7):826–840, July 2013.

- P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Dynamic covariance scaling for robust robotic mapping. In *Workshop on robust and Multimodal Inference in Factor Graphs, (ICRA)*, 2013a.

- P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Robust map optimization using dynamic covariance scaling. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 62–69, 2013b. **Best Student Paper Award Finalist.**

- P. Agarwal, W. Burgard, and C. Stachniss. Helmert's and Bowie's Geodetic Mapping Methods and Their Relationship to Graph-Based SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3619–3625, 2014a.

- P. Agarwal, W. Burgard, and C. Stachniss. A Survey of Geodetic Approaches to Mapping and the Relationship to Graph-Based SLAM. *Robotics and Automation Magazine*, 21(3):63–80, September 2014b.

- P. Agarwal, G. Grisetti, G. D. Tipaldi, L. Spinello, W. Burgard, and C. Stachniss. Experimental analysis of dynamic covariance scaling for robust map optimization under bad initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3626–3631, 2014c.

Chapter 2

Preliminaries

SLAM has been extensively researched in the last two decades, leading to efficient algorithms based on the Extended Kalman Filter (EKF) (Smith et al., 1988), the Extended Information Filter (EIF) (Schumitsch et al., 2006), the Sparse Extended Information Filter (SEIF) (Thrun et al., 2004; Walter et al., 2005, 2007) and particle filters (Montemerlo, 2003; Grisetti et al., 2005; Stachniss et al., 2005; Kwak et al., 2007). The computation and memory requirements for EKF-based methods grow quadratically with respect to the number of mapped features. EIF-and SEIF-based methods are algorithmically more efficient than EKF-based methods, but do not allow relinearization of observations. This makes them susceptible to measurements with high sensor noise. Particle-filters ameliorate issues with non-linearities and sensor noise but lead to the particle-depletion problem (Stachniss et al., 2005). Some of these shortcomings are handled by graph-based SLAM as it allows to re-linearize measurements. As the methods developed in this thesis uses the graph-based SLAM paradigm, we provide a brief overview of the topic in the next section.

2.1 Graph-Based SLAM

In the graph-based SLAM paradigm (Lu and Milios, 1997; Folkesson and Christensen, 2004; Thrun and Montemerlo, 2006; Dellaert and Kaess, 2006; Grisetti et al., 2010a), each pose of the robot is represented as a node in a graph. In general, the nodes may also represent features in the world, such as interest points extracted from images or laser point clouds (Ruhnke et al., 2011). It may also represent physical landmarks such as trees or structures in the world. A factor between two nodes is represented by an edge in the graph. The factor may represent rigid body transformations for pose-to-pose constraints or bearing measurements for features. The first part of the overall problem is to create a graph (factor-graph), by identifying nodes and factors between the nodes based on sensor data. Such a system is often referred to as the *front-end*. The second part entails finding the configuration of the nodes that best explains the factors. This step corresponds to computing the maximum-likelihood map and a system solving it is typically referred

to as a *back-end*. In the next two sections we outline recent advances in front-ends and back-ends for graph-based SLAM.

2.1.1 Front-End

The primary function of a graph-based SLAM front-end is to identify the geometrical relationships between multiple nodes in the graph, by interpreting sensor data from perception sensors. Consider a robot that drives around a large city block. When the robot comes back to the starting position, its dead reckoning, based on inertial and odometry sensors will accumulate error. This is due to various reasons. Inertial measurements rely on double-integrating acceleration measurements from gyroscopes, resulting in drift over time. Odometry sensors record wheel rotations which are subject to discretization and wheel slippage. If the trajectory is plotted just based on dead-reckoning sensors, in the long run we can expect the robot position to be significantly different than the correct position. The job of the front-end system is to process sensor data to identify that the robot is back to the starting point or a previously visited place. This problem is commonly known as identifying *loop closures*.

The earliest loop-closure systems were based on shape registration of laser scans (Cox, 1991; Lu, 1995; Gutmann and Schlegel, 1996). Cox (1991) matched laser scans by aligning points to line segments extracted from an a-priori environment representation. Gutmann and Schlegel (1996) extended this method by extracting line segments from reference scans to make the approach applicable for a scan-to-scan alignment. Lu (1995) improved the point-to-line matching of Cox (1991) with a point-to-point matching based on the iterative closest point algorithm (ICP) (Besl and McKay, 1992) and thus making scan matching model free. The first cross correlation based laser scan matching was performed by Weiß and Puttkamer (1995). Each laser scan is represented with a histogram and the matching is found by maximizing the cross correlation between them. Since then, significant improvements have been made in front-ends based on laser scan matching (Biber and Straßer, 2004; Olson, 2009a; Li and Olson, 2010; Tipaldi et al., 2013; Segal et al., 2009; Censi, 2008; Mazuran and Amigoni, 2014; Diosi and Kleeman, 2007). Biber and Straßer (2004) transform each scan with a normal distribution. In this method small regions in laser scans are represented with Gaussians and the correct transformation between scans is found by maximizing the overlap of points with the computed Gaussians. This is related to the correlative scan matching method (Olson, 2009a), where, instead of approximating local points with a Gaussian as in (Biber and Straßer, 2004), the author builds a rasterized cost table based on the log probability of observing a new point. The cost table is then used to evaluate different alignments providing a solution independent of initialization. Other authors extract local descriptors from laser scans and match these descriptors to perform alignment (Li and Olson, 2010; Tipaldi et al., 2013). Over the years there have been significant improvements in scan alignment based on ICP and its

variants. For example, many authors have extended scan matching to 3D (Borrmann et al., 2008; Nieto et al., 2007; Bosse and Zlot, 2009) as well as modeled the registration errors accurately for better map estimation (Censi, 2008; Segal et al., 2009).

In addition to laser based scan matching, many camera based front-ends have also been proposed. Most common camera based front-end systems operate by extracting low-level image descriptors. If multiple image descriptors extracted from two images can be matched with each other, a loop closure may be proposed. A plethora of image descriptors exist (Mikolajczyk and Schmid, 2005), each having its advantage over the others. For example, SIFT (Lowe, 1999) and SURF (Bay et al., 2008) are scale and rotational invariant. BRIEF (Calonder et al., 2010) and BRISK (Leutenegger et al., 2011) are computationally efficient to match given they are represented as binary masks. Other authors have proposed to use image patches instead of explicit descriptors. Such an approach, commonly known as bag-of-words has been successfully applied for image matching (Fei-Fei and Perona, 2005; Nister and Stewenius, 2006; Cummins and Newman, 2009). Other researchers have come up with algorithms which can encode whole images as a descriptor (Liu and Zhang, 2012). These whole image representations can then be used for matching other images for place recognition.

The result of the above laser based scan-matching and camera based place recognition algorithms can be further improved by performing compatibility test (Neira and Tardos, 2001; Bosse, 2004; Olson et al., 2005). Bosse (2004) proposed a cycle verification step for identifying correct loop closures. The intuition behind cycle verification is that if the cumulative transformation using multiple loop closures around a cycle is close to identity, then all of the loop closures in that cycle are consistent and hence correct. Joint Compatibility Branch and Bound (JCBB) (Neira and Tardos, 2001) rejects false feature correspondences by measuring their joint compatibility from a set of initial correspondence pairs. It is computationally efficient at handling the combinatorial explosion of pairing by incrementally building an "interpretation tree". Related, Bailey (2002) outlined Combined Constraint Data Association (CCDA) to remove spurious loop closures. CCDA is able to identify wrong loop closures by finding maximum clique size in complete subgraphs. Single Cluster Graph Partitioning (SCGP) (Olson et al., 2005) also identifies false loop closures by computing a subset of measurements that are all compatible which each other but unlike CCDA, it is not limited to discrete Boolean quantities. These front-end verification methods reduce the number of false positives, but as the initialization of the poses deteriorates there performance also degrades. Later, in Chapters 4-6, we will outline SLAM back-ends which further improve the robustness against outliers and wrong loop-closures. Our methods are complementary to the above mentioned front-end verification techniques. The presence of a front-end verification technique will help improve the performance of our methods, but additionally we will show that in the extreme case, our methods are robust to loop-closure errors even in the

absence of any front-end compatibility tests.

2.1.2 Back-End

Graph SLAM front-ends as explained in the previous section generate a graph with factors between nodes. Each factor for example may represent rigid body transformation between the robot positions represented by the nodes. Back-ends aim at finding the configuration of the nodes that minimize the error induced by factors from the front-end. Let $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_n)^T$ be a state vector where $\mathbf{x}_i$ describes the pose of node i, which maybe the position of the robot or landmark. This pose $\mathbf{x}_i$, is typically three-dimensional for a robot living in the 2D plane and consist of two Cartesian coordinates x, y and angular coordinate θ. For robots like unmanned aerial vehicles which operate in full 6D, pose $\mathbf{x}_i$ may be six-dimensional for robots and three-dimensional for point features. $\mathbf{x}_i$ may also be generalized to other positions such as the x and y positions for a planar landmark or x, y and z positions for a three-dimensional landmark.

We can describe the error function $\mathbf{e}_{ij}(\mathbf{x})$ for a single factor between the nodes i and j as the difference between the observed measurement $\mathbf{z}_{ij}$ and the expected measurement $\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j)$ given the current state

$$\mathbf{e}_{ij}(\mathbf{x}) \quad = \quad \hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}. \tag{2.1}$$

As the measurements $\mathbf{z}$ and the poses $\mathbf{x}$ do not form an Euclidean space, it is advantageous to not subtract the expected and obtained measurement in Equation (2.1) but to perform the operations in a non-Euclidean manifold space (Grisetti et al., 2010b; Hertzberg, 2008). We can define an operator $\boxminus$ that computes the difference while accounting for different domains of the involved variables,

$$\tilde{\mathbf{e}}_{ij}(\mathbf{x}) \quad = \quad \hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) \boxminus \mathbf{z}_{ij}. \tag{2.2}$$

The realization of the measurement function $\hat{\mathbf{z}}$ depends on the sensor setup. For pose to pose factors, one typically uses the transformation between the poses. For pose to landmark factors, we minimize the re-projection error of the observed landmark into the frame of the observing pose. The error minimization can be written as

$$\mathbf{x}^* \quad = \quad \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{ij} \mathbf{e}_{ij}(\mathbf{x})^T \mathbf{\Sigma}_{ij}^{-1} \mathbf{e}_{ij}(\mathbf{x}), \tag{2.3}$$

where $\mathbf{\Sigma}_{ij}$ is the covariance matrix and $\mathbf{\Sigma}_{ij}^{-1}$ is the information matrix associated to a factors between $\mathbf{x}_i$ and $\mathbf{x}_j$, and $\mathbf{x}^*$ is the optimal configuration of nodes with minimum sum of error induced by the factors. The solution to Equation (2.3) is based on successive linearization of the original non-linear cost, typically around the current estimate. The

non-linear error function is linearized using Taylor series expansion around an initial guess $\breve{\mathbf{x}}$:

$$\mathbf{e}_{ij}(\breve{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x}) \simeq \mathbf{e}_{ij}(\breve{\mathbf{x}}) + \mathbf{J}_{ij}\boldsymbol{\Delta}\mathbf{x}, \tag{2.4}$$

with the Jacobian

$$\mathbf{J}_{ij} = \left.\frac{\partial \mathbf{e}_{ij}(\breve{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x})}{\partial \boldsymbol{\Delta}\mathbf{x}}\right|_{\boldsymbol{\Delta}\mathbf{x}=0}. \tag{2.5}$$

This leads to an approximate cost function this is on a quadratic form. In the end, the Jacobians from all factors can be stacked into a matrix and the minimum of the quadratic form can be found by solving the linear system

$$\mathbf{H}\boldsymbol{\Delta}\mathbf{x}^* = -\mathbf{b}, \tag{2.6}$$

where $\mathbf{H} = \sum_{ij} \mathbf{J}_{ij}^T \boldsymbol{\Sigma}^{-1}{}_{ij} \mathbf{J}_{ij}$ and $\mathbf{b} = \sum_{ij} \mathbf{J}_{ij}^T \boldsymbol{\Sigma}^{-1}{}_{ij} \mathbf{e}_{ij}$ are the elements of the quadratic form that results from the linearized error terms. $\boldsymbol{\Delta}\mathbf{x}^*$ is the increment to the nodes in the graph configuration that minimizes the error in the current iteration at the current linearization point:

$$\mathbf{x}^* = \breve{\mathbf{x}} + \boldsymbol{\Delta}\mathbf{x}^*. \tag{2.7}$$

For operations performed on a manifold representation, the operator $\boxplus$ is used to update the increment $\boldsymbol{\Delta}\mathbf{x}^*$ with a non-linear function. A more detailed analysis of operations on the manifold representation can be found in (Grisetti et al., 2010b; Hertzberg, 2008; Kümmerle, 2013).

Equation (2.6) may be solved directly using sparse matrix decompositions such as Cholesky or QR decomposition. Alternatively, given that the system of equations originally comes from a non-linear problem, Levenberg-Marquardt, Powell's Dog Leg, Stochastic Gradient descent and it variants may also be used for computational or convergence improvements. It is important to note that all these methods should ideally converge to the same global minimum in the case of a strictly convex problem. Optimization problems resulting from SLAM are highly non-convex due to non-linear measurements and hence offer no such guarantees.

In their seminal paper, Lu and Milios (1997) compute Equation (2.6) by inverting the quadratic matrix $\mathbf{H}$. This is reported to be the most computationally expensive operation as it scales cubicly with the number of nodes. Recent graph-based SLAM research focuses on efficiently solving Equation (2.6) using domain knowledge and sparse linear algebra methods. Gutmann and Konolige (1999) propose a method that could incrementally build a map and required the expensive matrix inversions only after certain pre-determined steps. Konolige (2004) provided a computationally efficient algorithm

with a complexity of $O(n)$ for single loop closures and $O(n \log n)$ for multiple looped maps. He identified the sparse connectivity of the information matrix resulting from SLAM graphs and used Conjugate Gradient preconditioned with an incomplete Cholesky factor to reduce computational requirements. Folkesson and Christensen (2004) formulate the least squares problem in Equation (2.3) as an energy-minimization problem. They also incorporate data association within the energy function, which implicitly performs a χ^2 test. They furthermore reduce the complexity and size of the graph by collapsing subgraphs into a single node, which they call star nodes.

Dellaert and Kaess (2006) provide an interpretation of the graph-based formulation as a factor graph using smoothing and mapping. Their method, namely $\sqrt{\mathrm{SAM}}$, uses matrix factorization methods such as QR, LU, and Cholesky decomposition to solve Equation (2.6). Besides the off-line approach, Kaess et al. (2007a) also proposed an efficient incremental optimization method called iSAM that uses Givens rotation to directly update the square root of the information matrix. The computational requirements were further improved in iSAM2 (Kaess et al., 2012) by using an efficient data structure called the Bayes Tree.

Other authors minimize Equation (2.3) using relaxation techniques, where parts of graphs are incrementally optimized (Howard et al., 2001; Duckett et al., 2000; Frese et al., 2005). Olson et al. (2006) solve the least squares problem using stochastic gradient descent. In stochastic methods, the error induced by each factor is reduced, one at a time, by moving the nodes accordingly. TORO (Grisetti et al., 2007b) proposed a re-parametrization of the problem to improve convergence. Unlike (Olson et al., 2006), which uses the odometry chain as the spanning tree for error distribution, TORO created a spanning tree by initializing each node with its oldest neighbor. Stochastic methods are typically more robust to poor initial estimates and can be run incrementally by tuning the learning rate (Olson et al., 2007; Grisetti et al., 2008). Also hierarchical and submapping approaches have shown to efficiently compute solutions by decomposing the optimization problem at different levels of granularity (Bosse et al., 2004; Estrada et al., 2005; Grisetti et al., 2010b, 2012; Ni et al., 2007; Ni and Dellaert, 2010; Paz et al., 2008).

A lot of the above mentioned state-of-the-art back-ends are closely related to several decade old geodetic mapping methods that were used to build massive survey maps, some even spanning across continents. In principle, these continental scale maps are built in a similar way to the front-end/back-end approach used in SLAM. In the next chapter we provide a summary of such geodetic mapping methods and explain how they relate to graph-based SLAM back-ends.

Chapter 3

A Survey of Geodetic Mapping Methods

The ability to simultaneously localize a robot and build a map of the environment is central to most robotic applications and the problem is often referred to as SLAM. Robotics researchers have proposed a large variety of solutions allowing robots to build maps and use them for navigation. Also the geodetic community addressed large-scale map building for centuries, computing maps which span across continents. These large-scale mapping processes had to deal with several challenges that are similar to those of the robotics community. In this chapter, we explain key geodetic map building methods that we believe are relevant for robot mapping. We also aim at providing a geodetic perspective on current state-of-the-art SLAM methods and identifying similarities both in terms of challenges faced as well as in the solutions proposed by both communities. The goal of this chapter is to connect both fields and to enable future synergies between them.

3.1 Introduction

As we have explained in the first chapter of this thesis, autonomous map building is essential for mobile robots to operate in unknown environments. A mobile robot needs a map of its environment to plan appropriate paths towards a goal location. Given that the geodetic mapping community built massive survey maps, some even spanning across continents in a similar way to the SLAM routines used in robotics, we are interested in how do these two fields – geodetic mapping and robot mapping relate to each other.

Geodetics is the science that studies the earth at various global and local scales. It measures large scale changes in the earth's crustal movements, tidal waves, magnetic and gravitational fields, etc. The aspects of geodesy, which we are most interesting in the context of SLAM, are related to geodetic mapping. In principle, geodetic maps are built in a similar way to approach used in graph-based SLAM. Constraints are acquired through observations between physical observation towers. These towers correspond to positions of the robot as well as the landmarks in the context of SLAM. Once the

constraints between observation towers are obtained, the goal is to optimize the resulting system of equations to get the best possible locations of the towers on the surface of the earth.

The aim of this chapter is to survey key approaches of the geodetic mapping community and put them in relation to recent SLAM research. This mainly relates to the back-ends of graph-based SLAM systems. We believe that this step will enable further collaborations and exchanges among both fields. As we will illustrate during this survey, both communities have come up with sophisticated approximations to the full least squares approach for computing maps with the aim of reducing memory and computational resources.

This chapter is organized as follows. First, we identify key similarities between the challenges faced in geodetic and robotic mapping. Second, we explain geodetic mapping and introduce commonly used terminologies. Third, we explain the geodetic mapping back-ends, including the motivation and insights central to the developed solutions. Finally, we highlight the relationships between individual graph-based SLAM approaches and geodetic mapping methods.

3.2 Common Problems between SLAM and Geodetic Mapping

SLAM and geodetic mapping have several common problems. The first common challenge between them is the large size of the maps. The map size in the underlying estimation problems is represented by the number of unknowns. In geodetic mapping, the unknowns are the positions of the observation towers, while for robotics, the unknowns corresponds to robot positions and observed landmarks. For example, the system of equations for the North American Datum of 1927 (NAD 27) required solving for 25,000 positions and the North American Datum of 1983 (NAD 83) required solving for 270,000 positions (Kolata, 1978). The largest real-world SLAM datasets have up to 21,000 poses (Johannsson et al., 2013), while simulated datasets with 100,000 poses (Konolige et al., 2010) and 200,000 poses (Grisetti et al., 2010a) have been used for evaluation. At the time of NAD 27 and 83, the map building problems could not be solved by standard least squares methods as no computer was capable to handle such large problems. Even nowadays, computational constraints are challenging in robotics. SLAM algorithms often need to run on real mobile robots, including battery-powered wheeled platforms, small flying robots, and low-powered underwater vehicles. For autonomous operation, the memory and computational requirements are often constrained so that approximate but online algorithms are often preferred over more precise but offline ones.

The second challenge results from outliers or spurious measurements. The front-ends for both, robotics and geodetic mapping, are affected by outliers and noisy measurements.

In robotics, the front-end is often unable to distinguish between similar looking places, which leads to perceptual aliasing. A single outlier observation generated by the front-end can lead to a wrong solution, which in turn results in a map that is not suitable to perform navigation tasks. To deal with this problem, there recently has been a more intense research for reducing the effect of outliers on the resulting map by using extensions to the least squares approach (Olson and Agarwal, 2012; Sünderhauf and Protzel, 2012; Latif et al., 2012b; Agarwal et al., 2013b). Some of these methods are covered in the later chapters. For geodetic mapping, the front-end consisted of humans carefully and meticulously acquiring measurements. However, even this process was prone to making mistakes (Schwarz, 1989).

The third challenge comes from the non-linearity of constraints, which is frequently the case in SLAM as well as geodetic mapping. A commonly used approximation is to linearize the problem around an initial guess. However, this approximation to the non-linear problem may lead to a suboptimal solution if the initial guess is not in the basin of convergence. There are various methods, which enable finding a better initial guess, but this still remains a challenge (Carlone et al., 2011; Grisetti et al., 2012). The importance of the initial guess in SLAM has also motivated the study of the convergence properties of the corresponding nonlinear optimization problem (Wang et al., 2012; Carlone, 2013; Hu et al., 2013; Agarwal et al., 2014c).

Fourth, both geodetic mapping and SLAM ideally require an incremental and online optimization algorithm. In robotics, a robot is constantly building and using the map. It is advantageous if the system is capable of optimizing the map incrementally (Kaess et al., 2007b; Olson et al., 2007; Grisetti et al., 2008; Kaess et al., 2012). In geodetic mapping, new survey towers are build and new constraints are obtained as and when required. It is not feasible to optimize the full network from scratch when new areas or constraints are added. Thus, also geodetic methods must be able to incorporate new information into existing solutions with minimum computational demands.

Given these similarities, we believe that studying the achievements of the geodetic scholars is likely to inspire novel solutions to large-scale, autonomous robotic SLAM. In the next section we provide an overview of Geodetic mapping methods. We first describe how the massive geodetic surveys, which typically result in triangle nets, are created. We then explain various geodetic back-ends and compare them to current graph-based SLAM methods for obtaining a minimal error configuration.

3.3 Geodetic Mapping

The basic principle behind geodetic surveying is triangulation. Various observation towers, called Bibly towers, typically 20-30 meters in height, are built in line-of-sight with neighboring towers (Burkard, 1983). An example of a Bibly tower with surveyors

Figure 3.1: Observing towers called Bibly towers built for triangulating other towers. Figure courtesy of (Natural Resources Canada, 2009).

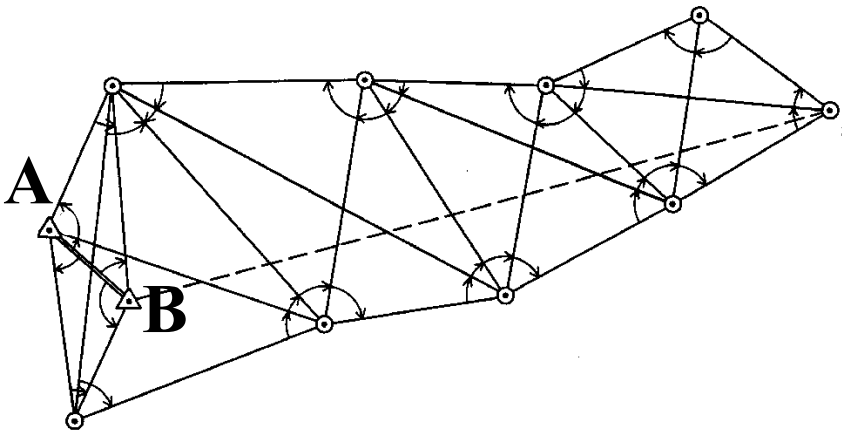


Figure 3.2: A simple triangle net. The distance between towers A and B is physically measured. Angular constraints to all other towers are measured and the lengths of all other segments are computed with respect to the baseline AB with the measured angles. Figure courtesy of (Burkard, 1983).

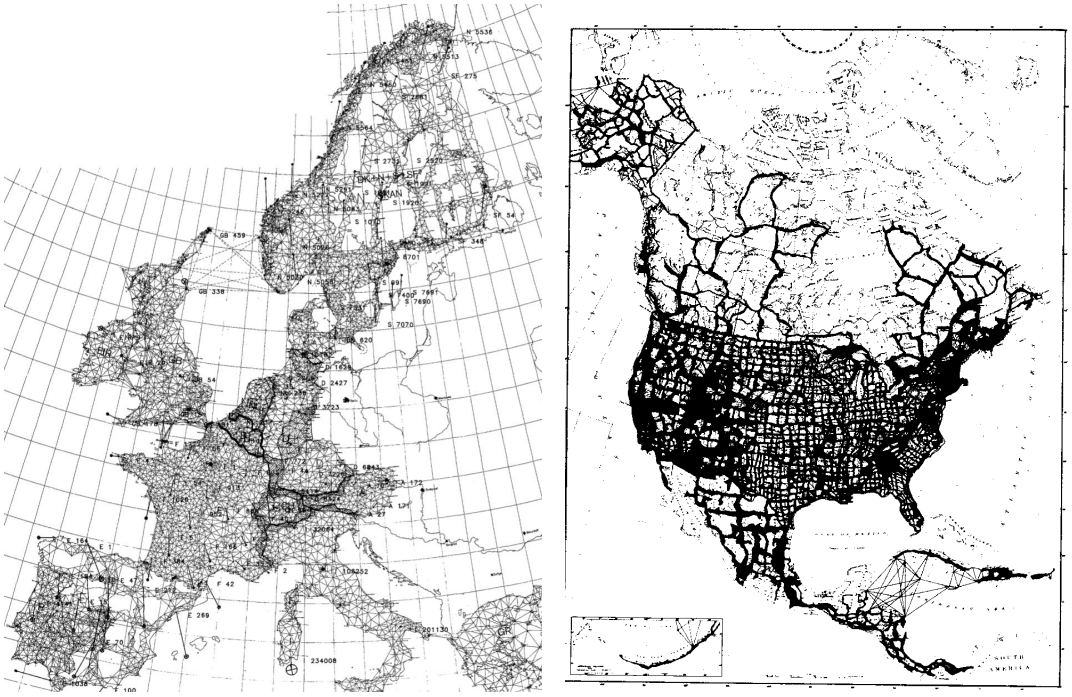


Figure 3.3: The left image illustrates the triangulation networks spanning across Europe for the construction of the European Datum of 1987. Figure courtesy of (Ehrnsperger, 1991). The right image shows the network of constraints existing in 1981 for mapping the North American continent. Figure courtesy of (Schwarz, 1989).

on top is shown in Figure 3.1. Geodetic surveyors built large interconnected mesh of observation towers by triangulating measurements between them. The resulting mesh of towers and constraints is commonly called a triangle net. A simple example of a triangle net is shown in Figure 3.2. Each line segment in Figure 3.2 is a constraint between two observation towers. Some of these constraints are directly measured while others are computed using trigonometrical relationships.

The method of obtaining constraints between the observing towers has evolved over time. Initially, constraints were distance only or angle only measurements. The distance measurements were obtained using tapes and wires of Invar, which has a low coefficient of thermal expansion. Later, more sophisticated instruments, such as parallax range-finders, stadimeter, tellurometer, and electronic distance measuring instruments were used. Angular measurements were obtained using theodolites. Measuring distance using tapes and wires is more cumbersome compared to computing angles with theodolites. Hence, only a few distance measurements called baselines are computed and the other distance measurements are deduced based on angular measurements and trigonometrical relationships. In Figure 3.2, the towers A and B are the baseline and only angular measurements to other towers are measured. The distances between all other towers are

deduced using the measured baseline AB, angles, and trigonometrical relations.

Moreover, measurement constraints used in geodetic surveys can be differentiated as absolute and relative measurements. Absolute measurements involve directly measuring the position of a tower on the surface of the earth. These include measuring latitudes by observing stars at precise times and then comparing them to the existing catalogs. Stars were also used as fixed landmarks for bearing only measurements from different base towers at precisely the same time. Later, more sophisticated techniques such as very long baseline interferometry (VLBI) and GPS measurements, which lead to an improved measurement accuracy, were introduced. Angular measurements obtained with theodolites for large scale geodetic mapping were abandoned after the introduction of VLBI and the triangulation techniques have been replaced by trilateration around 1960. In the US, High Accuracy Reference Network (HARN) was built using VLBI and GPS only (Milbert) by 1990. The Continuously Operational Reference Station (CORS) introduced in 1995 uses only GPS measurements and is regularly updated. National Spatial Reference System of 2007 known as NAD 83(NSRS2007) is the latest readjustment, containing solely of static GPS measurements (Pursell and Potterfield, 2008).

Examples of large scale triangle nets are shown in Figure 3.3. Figure 3.3(a) displays the geodetic triangle net used for mapping Central Europe in 1987 whereas Figure 3.3(b) shows the triangulation network in existence for mapping North-America in 1983 (NAD 83). The thick lines in NAD 83 are long sections of triangulation nets comprising of thousands of Bibly towers. Connections between multiple sections are small triangle networks and are called junctions.

The geodetic mapping community typically chose the earth's center of gravity as the sensor origin. This eased the process of acquiring measurements as it is a quick and easy to standardize calibration technique. The primary form of triangulation was done using theodolites, all of which contain a level, which allows aligning the instruments with respect to the earth's center of gravity (E_{cg}). All other instruments use some form of a plumb-line as a reference, which aligned them with E_{cg}. In fact, choosing E_{cg} is also preferred when using satellites for acquiring measurements as they orbit the earth's E_{cg}. The exact shape of the earth is not a sphere but a geoid, which can be approximated as an ellipsoid. However, the center of this approximate ellipsoid does not coincide with E_{cg}. This is because earth's mass is not uniformly distributed. Hence, although choosing the sensor origin as E_{cg} is practically a good choice, it is mathematically inconvenient. It requires additional mathematical computations for mapping measurements to the center of the ellipsoid. This problem is commonly called computing the "deflection of the vertical".

The geodetic mapping problem can be broken down into two major sub problems, the first being the "adjustment of the vertical", the second being "adjustment of the net". The problem of "adjustment of the net" is finding the least squares system of the planar

triangulation net whereas "adjustment of the vertical" involves finding parameters to wrap this mesh network on a geoid representing earth (Wolf, 1950).

3.4 Adjustment of the Net – Geodetic Back-Ends

The problem of adjustment of the net is quite similar to the graph-based SLAM formulation. In the SLAM notation, the geodetic mesh network consists of various observation towers constrained by non-linear measurements. These physical towers are similar to the unknown state vector $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T$ in the SLAM problem explained in Section 2.1.2. In SLAM, 2D and 3D robot headings (roll, pitch and yaw) are estimated in addition to the Cartesian coordinates (x, y, z). For geodesy, the surveyors were mainly interested in Cartesian coordinates of the towers on the surface of the earth. Angular components were used until the 1960ies in geodesy before GPS and VLBI systems were available.

The task of the back-end optimizer in geodesy is to find the best possible configuration of the towers on the surface of the earth to minimize the sum of errors from all constraints. All non-linear constraints can be linearized using Taylor series expansion and stacked into a matrix (Kolata, 1978). The least squares solution can be computed directly by solving the least square system in Equation (2.6). Both, SLAM and geodetic problems inherently contain non-linear equations constraining a set of positional nodes. The process of linearizing the constraints and solving the linearized system is repeated to improve the solution.

The biggest challenge faced by the geodetic community over centuries was limited computing power. Even with the most efficient storage and sparse matrix techniques, there was no computer available which could solve the full system of equations as a single problem. For example, the NAD 83 mesh network, illustrated in Figure 3.3(b), requires solving 900,000 unknowns with 1.8 million equations (Bossler, 1987). A dense matrix of size 900,000 times 900,000 would require more than 3,000 GB just to store it. Even when using sparse matrix data-structures such as compressed sparse rows or columns, NAD 83 would still require roughly 30 GB of memory to store the matrix of normal equations (considering that only 0.5% of the matrix elements are non-zeros). In the next section, we explain some of the geodetic back-end optimizers used for mapping large continental networks such as that of North America and Europe.

3.4.1 Helmert Blocking

When geodetics started to develop the large scale triangulations in the mid-1800s, the only way of solving large geodetic networks was by dividing the problem so that multiple people could work on solving each sub-problem. Helmert proposed the first method for solving the large optimization problem arising from the triangulation network in

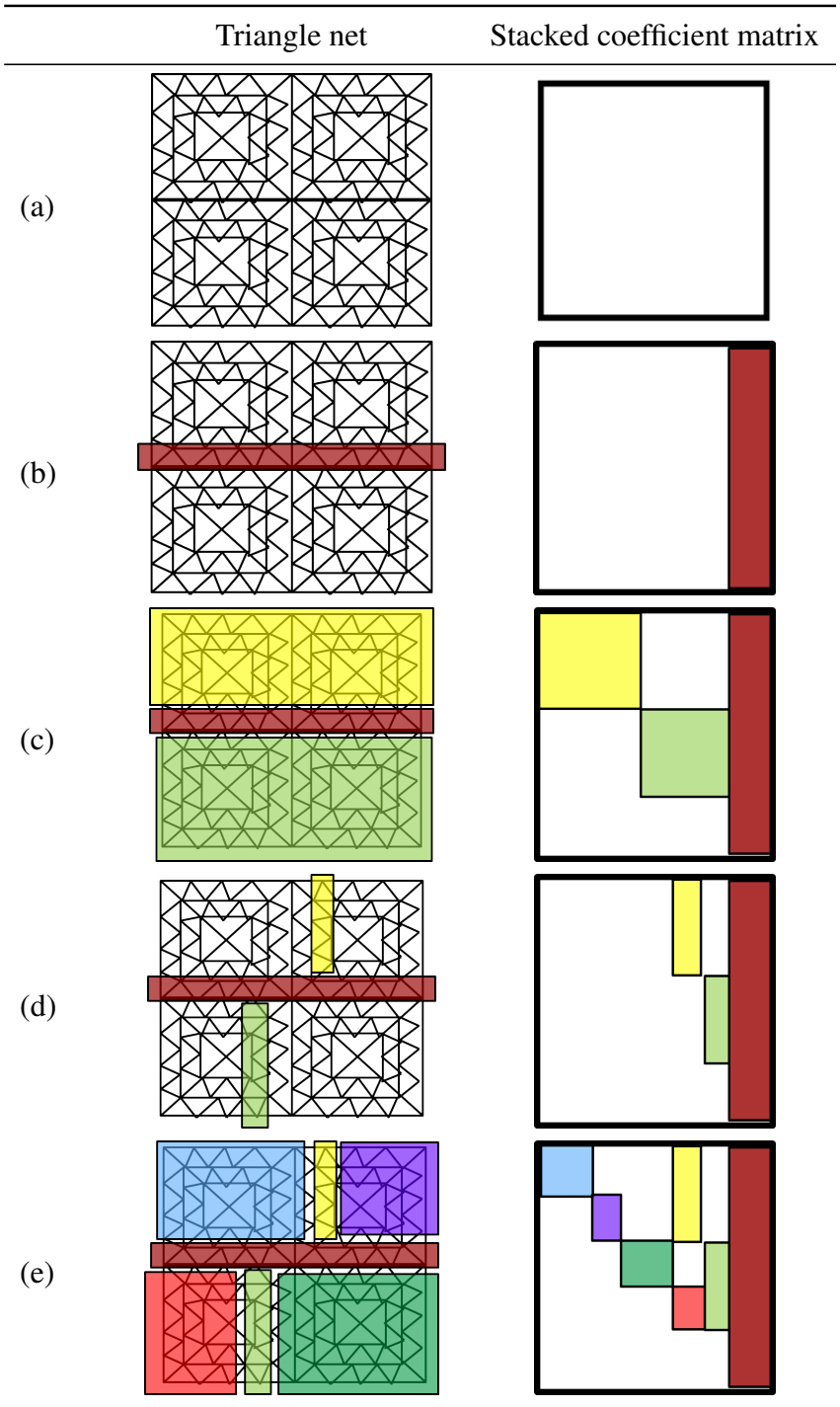


Figure 3.4: A toy example illustrating the Helmert blocking method. The left column shows a triangle net and the partitions. The right column shows the corresponding stacked coefficient matrices for each triangle net.

parallel. His strategy, which was proposed in 1880, is possibly the oldest technique to partition the set of equations into smaller problems (Helmert, 1880; Aeronautical Chart and Information Center, 1964). Although graph partitioning and submapping has been frequently used in robotics, to the best of our knowledge, Helmert's method has not been referenced in the robot mapping community—only Triggs et al. (2000) mention it as an optimization procedure in their bundle adjustment survey.

Helmert observed that by partitioning the triangle net in a particular way, one can solve the overall system of equations in parallel. He outlined that the whole triangle net can be broken into multiple smaller subnets. All nodes that have constraints only within the same subnet are called inner nodes and can be eliminated. All separator-nodes, i.e., those that connect multiple subnets, are then optimized. The previously-eliminated inner nodes can be computed independently given the values of the separator nodes. Most importantly, the so-formed subnets can be solved in parallel. Helmert's blocking method is outlined in Algorithm 1 and is explained more precisely as a mathematical algorithm by Wolf (1978).

Algorithm 1 Helmert Blocking

1: Given a partitioning scheme, establish the normal equations for each partial subnet separately
2: Eliminate the unknowns for all nodes which do not have constraints with neighboring partial subnets
3: Establish the main system after eliminating all the inner nodes containing only intra-subnet measurements
4: Solve the main system of equations containing only separator nodes
5: Solve for inner nodes given the value of separator nodes

Consider a simple triangle net shown in Figure 3.4. In Figure 3.4(a), each line segment is a constraint and the end of segments represent a physical observation tower. Helmert observed that if he divides the triangle net into two halves, for example as illustrated in Figure 3.4(b), the top half of the towers will be independent of the bottom half given the values of the separators as shown in Figure 3.4(c). Such a system can be solved using reduced normal equations (Wolf, 1978; Schwarz, 1989).

Let us represent the whole system of equations from the triangle net in Figure 3.4(a), as:

$$\mathbf{A}\mathbf{x} \;=\; \mathbf{b}. \tag{3.1}$$

This equation can be subdivided into 3 parts in the following manner:

$$\begin{bmatrix}\mathbf{A}_s & \mathbf{A}_1 & \mathbf{A}_2\end{bmatrix} \begin{bmatrix}\mathbf{x}_s \\ \mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} \;=\; \mathbf{b}. \tag{3.2}$$

Here, $\mathbf{A}_s$ and $\mathbf{x}_s$ represent the coefficients and unknowns respectively, arising from the central separator. $\mathbf{A}_1$ and $\mathbf{A}_2$ are coefficients of the top and bottom subnets. The coefficient matrix $\begin{bmatrix}\mathbf{A}_s & \mathbf{A}_1 & \mathbf{A}_2\end{bmatrix}$ in Equation (3.2) is shown on the right-hand side of Figure 3.4(c). The corresponding system of normal equations is:

$$\begin{bmatrix}\mathbf{N}_s & \mathbf{N}_1 & \mathbf{N}_2 \\ \mathbf{N}_1^T & \mathbf{N}_{11} & 0 \\ \mathbf{N}_2^T & 0 & \mathbf{N}_{22}\end{bmatrix}\begin{bmatrix}\mathbf{x}_s \\ \mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\mathbf{b}_s \\ \mathbf{b}_1 \\ \mathbf{b}_2\end{bmatrix}. \tag{3.3}$$

The towers in $\mathbf{x}_1$ do not share any constraints with towers in $\mathbf{x}_2$. Both, $\mathbf{x}_1$ and $\mathbf{x}_2$ share constraints with $\mathbf{x}_s$ but not with each other. The key element in Equation (3.3) is the block structure of $\mathbf{N}_{11}$ and $\mathbf{N}_{22}$. The system of equation in Equation (3.3) can be reduced such that:

$$\bar{\mathbf{N}}_s\mathbf{x}_s = \bar{\mathbf{b}}_s, \tag{3.4}$$

where $\bar{\mathbf{N}}_s$ is computed as:

$$\begin{aligned}\bar{\mathbf{N}}_s &= \mathbf{N}_s - \begin{bmatrix}\mathbf{N}_1 & \mathbf{N}_2\end{bmatrix}\begin{bmatrix}\mathbf{N}_{11}^{-1} & 0 \\ 0 & \mathbf{N}_{22}^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{N}_1^T \\ \mathbf{N}_2^T\end{bmatrix} \\ &= \mathbf{N}_s - \sum_{i=1,2}\mathbf{N}_i\mathbf{N}_{ii}^{-1}\mathbf{N}_i^T, \mathbf{b}\end{aligned} \tag{3.5}$$

and $\bar{\mathbf{b}}_s$ is computed as:

$$\bar{\mathbf{b}}_s = \mathbf{b}_s - \sum_{i=1,2}\mathbf{N}_i\mathbf{N}_{ii}^{-1}\mathbf{b}_i^T. \tag{3.6}$$

Here, $\bar{\mathbf{N}}_s$ is called the reduced normal equations. Once $\mathbf{N}_s$ has been solved, $\mathbf{x}_1$ and $\mathbf{x}_2$ can be computed by solving:

$$\mathbf{N}_{11}\mathbf{x}_1 = \mathbf{b}_1 - \mathbf{N}_s^T\mathbf{x}_s \tag{3.7}$$

$$\mathbf{N}_{22}\mathbf{x}_2 = \mathbf{b}_2 - \mathbf{N}_s^T\mathbf{x}_s. \tag{3.8}$$

Moreover, Wolf (1978) states that matrices should never be inverted for computing Equations (3.4, 3.7, and 3.8). Instead, Cholesky decomposition or Doolittle-based LU decomposition should be employed. The steps outlined for computing the reduced normal forms in Equations (3.5 and 3.6) represent the Schur complement. The inverse computation in this step is trivial if the subnets result in block diagonal matrices. If not, both Wolf and Schwarz mention that each of the subnets can themselves be sparse and can be further sub divided as illustrated in Figure 3.4(d) (Wolf, 1978; Schwarz, 1989). Part IV of (Pursell and Potterfield, 2008) also contains a detailed methodology for applying

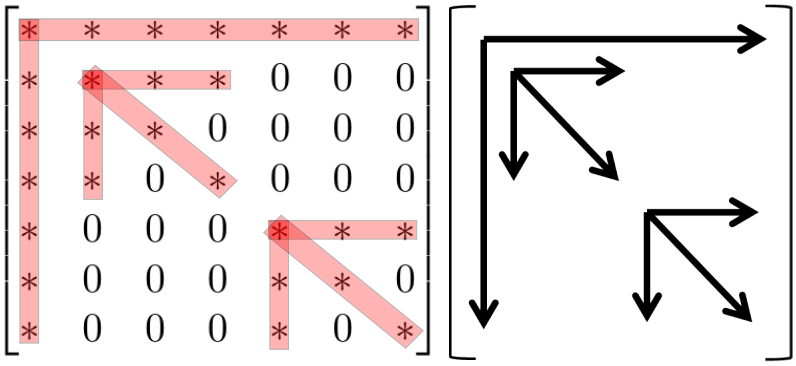


Figure 3.5: The structure of the coefficient matrix arising from two levels of Helmert blocks. The arrows in the right figure show non-zero cells.

Helmert Blocking for optimizing geodetic networks used for National Spatial Reference System of 2007.

The structure of the normal matrix represented in Figure 3.4(d) looks like the matrix represented in Figure 3.5. In the example explained in Figure 3.5, we have two levels of Helmert blocks. The number of levels can be arbitrary and each subnet can have a different number of recursive sub-blocks. Elimination takes place in a bottom-up manner. Then, the solution at the highest level is used to recursively propagate back to smaller subnets. Helmert also outlines the strategy to select separators at each level. This is discussed later in Section 3.4.7.

The Helmert blocking method was used for creating the North American Datum of 1983 (NAD 83). Ten levels of Helmert blocks were created with a total of 161 first level blocks. This is illustrated in Figure 3.6. The colored lines show the partitions of the graph at different levels. On average each of these 161 subnets contained 1,000 nodes, but roughly 80% of these nodes contained only internal measurements and can be eliminated (Schwarz, 1989). Three non-linear iterations of the whole system were performed, which took roughly four months each (Schwarz, 1989).

The Helmert blocking method is inherently multi-core and parallel but it still requires solving a large number of equations when solving for the reduced normal equation for a large separator set. The complexity of the exact method can be reduced if networks are built in sections as shown in Figure 3.7(b). Whenever the triangular network was not built using long sections, the net was approximated using polygon filling methods, which are explained below.

3.4.2 Polygon Filling

Polygon filling methods are used to convert a dense net into sections. An example for a dense network is shown in Figure 3.7(a) and the corresponding sparse mesh of

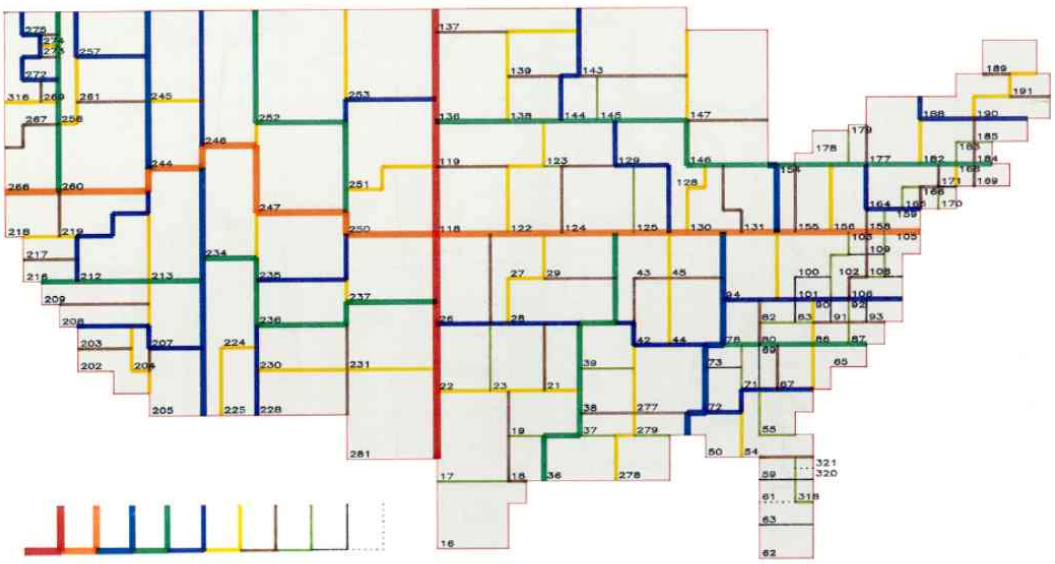
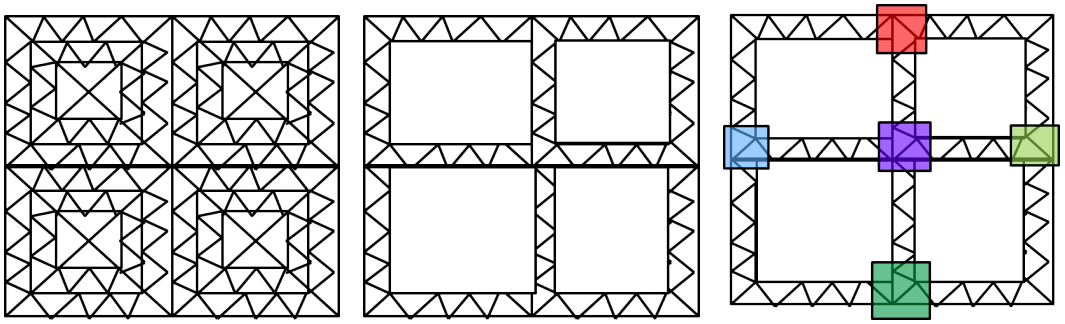
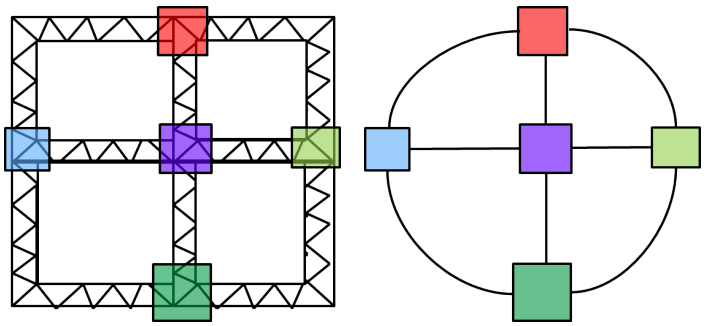


Figure 3.6: Helmert blocking applied to North America with ten levels. The legend in the bottom left corner shows progressive levels of partitions. The first partition, cuts the continent into east-west blocks, whereas the second cut partitions each half into north-south blocks. Hence, each geographical block is partitioned into four regions and this method is recursively performed on each sub-block. Figure courtesy of (Schwarz, 1989).



(a) Original triangle net. (b) Polygon filled net. (c) Helmert's method applied to polygon filled net.

Figure 3.7: The original triangle net (a) and the corresponding net after applying the polygon filling method (b), which leads to a smaller and more tractable problem at the expense of reduced accuracy. Once the sparse solution is computed, the positions of the interior stations can be calculated by keeping the outer sections fixed. Helmert blocking can also be applied here leading to the separator nodes illustrated by the colored squares (c).



(a) (b)

Figure 3.8: Bowie's approximate triangle net. Size of separators are much smaller once the polygon filling method is used. The separators are shown in colored squares (left). The Bowie method approximates this problem by abstracting the sections into a single constraint and all separator nodes as a single node (right).

sections in Figure 3.7(b). Inner regions in Figure 3.7(b) are completely removed from the initial optimization. Only the sections as shown in Figure 3.7(b) are optimized. In a subsequent optimization, the structure of the grids is fixed and the interior dense nets are optimized. This is an approximation as the optimization of the sections does not contain any measurements from the interior nets. Any error in the sections is distributed over the interior, shaded subnets (Hough, 1948).

Helmert's method can also be exploited in meshes comprising of sections to reduce the number of separator nodes. The colored small squares in Figure 3.7(c) are the separators after applying Helmert's method on the section and their size is much smaller compared to those depicted in Figure 3.4. The new separators are formed at the intersections of the sections rather than the long partitions in the original method. This difference can be observed by comparing Figure 3.7(c) and Figure 3.4(e).

The polygon filling method is also the preferred technique for incremental optimization. As triangulation networks evolve over time, large sections are built first and are subsequently triangulated densely—if and when required. This means that the outer sections are fixed and are never updated with measurements from inner nodes. Thus, any error in the initial optimization of the outer section has to be distributed over the inner regions. Even when adding new sections from new surveys covering unmapped area, the initially obtained sections are often kept fixed. Although this procedure is an approximation, it allows for increasing the size of a triangulation net without having to start the optimization from scratch. Thus, it is an efficient incremental map building method.

3.4.3 The Bowie Method

The use of networks of sections and the polygon filling method reduces the size of the optimization problem significantly but it is still computationally demanding considering the resources of the early 1900s. Bowie approximated the above methods further to create the North American Datum in 1927. Bowie's main insight was that he can approximate the net comprising of sections by collapsing intersections into a single node and the sections into a single virtual constraint. This is illustrated in Figure 3.8. This much smaller least squares system is solved to recover the positions of the intersections, which can then be used to compute the sections independently.

The core steps of the Bowie method consist of separating the sets of unknowns into segments and junctions. The junctions act as a small set of separator nodes. The junction nodes are shown as colored squares in Figure 3.8. These are not a single node but a small subnet of towers, which separates the sections. A generic junction is shown in Figure 3.9. All nodes in one junction are optimized together using least squares adjustment but ignore any inter-junction measurements. After this optimization, the structure of the junction does not change, i.e., each node in a junction is fixed with respect to other nodes in that particular junction.

As a next step, new latitudinal and longitudinal constraints are created between junctions. This is done by approximating each section with a *single* constraint. Each of such single constraints is a two-dimensional longitude and latitude constraint. As a result, each junction turns into a single node and each section into a single constraint. This leads to a much smaller but approximate problem, which is optimized using the full least-squares approach.

The above described steps of the Bowie method are summarized in Algorithm 2, see also Adams (1930). The full least squares problem is not solved by matrix inversion but by a variant of Gaussian elimination called Doolittle decomposition (see Wolf (1952)). The Doolittle algorithm is a LU matrix decomposition method, which decomposes a matrix column-wise into a lower triangular and upper triangular matrix. The solution can be computed using forward and backward substitution steps as with other matrix decomposition methods as well.

In essence, the Bowie method generates new, virtual constraints from sections. The uncertainty of such a virtual constraint is not equal to a the uncertainty of an individual measurement in the section. Thus, Bowie introduces a weight for each virtual measurement. These weights are chosen as the ratio of the length of the section with respect to the sum of all the section lengths. Hence, the weights are proportional to the length of the sections so that the larger proportion of the error is distributed over long sections compared to shorter ones.

Another computational trick to lower the efforts is to use diagonal covariances for the two-dimensional, virtual, latitude/longitude constraints. This enables to separate the

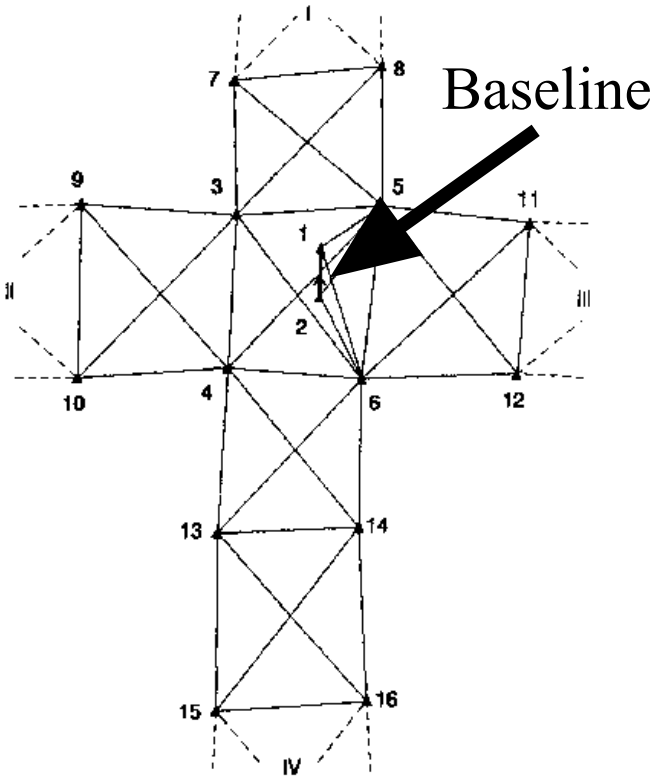


Figure 3.9: Example of a typical junction connecting segments in the four directions. The baseline and azimuth of stations 1 and 2 were measured directly. All other measurements were relative to other stations. Figure courtesy of (Schwarz, 1989).

system of equations for longitude and latitude. This yields two least squares problems with half of the original size.

The partitioning of the triangular net into junctions and sections is done manually. Each junction has to contain at least one measured baseline and one measured azimuth direction. This is sometimes referred to as astronomical stations. Occasionally, the size of junctions were enlarged to include an azimuth measurement because azimuth measurement have not been taken at all towers. Figure 3.10 illustrates the original triangle net and Bowie's approximated net into segments and junctions used for NAD 27. In Figure 3.10, the small circles represent junctions and all lines connecting junctions are sections of triangle nets. These sections and junctions individually represent a subset of the constraints connecting stations.

In sum, the Bowie method is an approximation of Helmert Blocking. The approximation uses single level subnets and an approximate optimization of the junction nodes. The optimization of the highest level in Helmert Blocking consists of junction nodes and is computed via reduced normal equations. In contrast to that, the Bowie method uses

virtual constraints at the highest level. The main difference is that the system of equations created by the virtual constraints is much smaller and sparser and hence easier to optimize than the full set of reduced normals as in the exact Helmert blocking method.

To the best of our knowledge, the Bowie method is the first implementation of a large scale approximate least squares system. It was effectively used in creating the NAD 27 as the triangulation nets were build in sections forming large loops. The Bowie methods exploits this structure. With time, more triangulation nets have been created creating many new loops. In this process, new triangulation nets were not optimized as a whole. They were integrated into the existing system by keeping the previous positions fixed. This led to inaccuracies but was better tractable than optimizing the system as a whole.

Algorithm 2 Bowie method

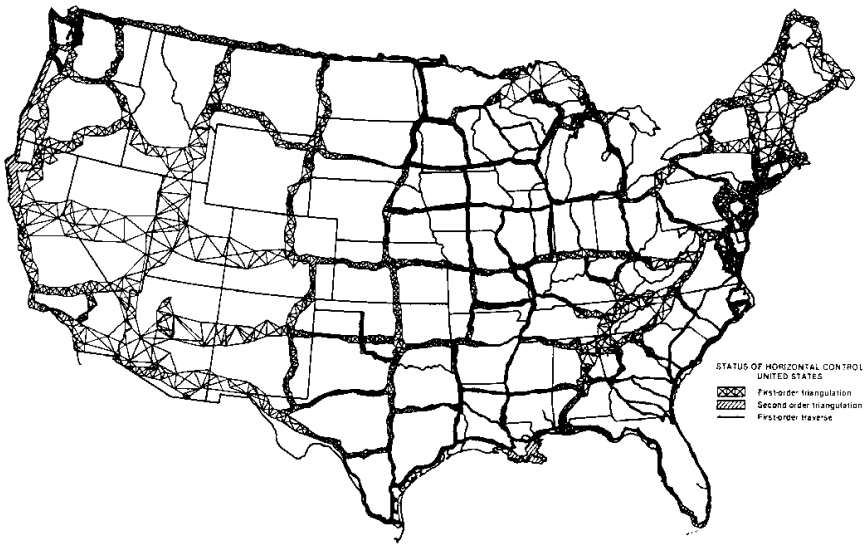
1: Separate triangle net into junctions and segments.
2: Optimize each junction separately.
3: Create new virtual equations between junctions treated as a single node
4: Solve the abstract system of equations comprising of each junction as a single node and each section as a single constraint
5: Update the resulting positions of stations in the segments using the new junction values

3.4.4 Modified Bowie Method for Central European Triangulation

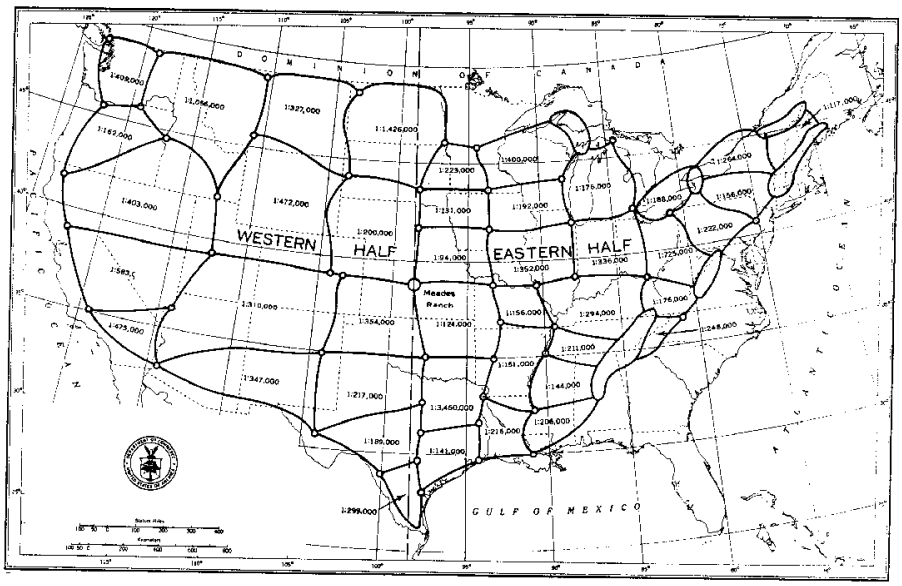
The approximation introduced by the Bowie method yielded suboptimal results for the European Datum of 1950 (Hough, 1948). For example, the virtual latitudinal and longitudinal constraints are artificially generated and not directly measured constraints but this fact is not fully considered in the optimization. Furthermore, cross correlations between latitudes and longitudes are ignored by using diagonal covariances and the junctions are fixed as a whole. Another issue in the Bowie method results from the assumption that the size of sections are much longer than junctions, which was the case for NAD 27, but the triangulation nets in the European Datum of 1950 (ED 50) are more dense. This results in the amplification of the errors introduced by the approximations (Hough, 1948).

The European geodetics thus proposed two modifications to the Bowie method to cope with the above mentioned problems for optimizing the ED 50 (Hough, 1948). The first one addresses the virtual measurements. Line 3 of Algorithm 2 sets up as many equations as sections. Instead, one virtual equation for every loop was created, enforcing a zero error around the loop.

The second modification addresses the way the linear system is solved efficiently. Instead of using the Doolittle method, they used the Boltz method, as explained below,



(a) The triangle net used for creating NAD 27.



(b) The Bowie method as applied for solving NAD 27.

Figure 3.10: Bowie method as used for NAD 27 triangle net (top). The figure below illustrates Bowie's approximation. Each small circle represent a junction and the junctions are connected by sections. Given the values of the junctions, the segments are independent of the rest. The western half contains 26 junctions and 42 sections creating 16 loops. The eastern half contains 29 junctions and 55 sections forming 26 loops. All measurements were computed with respect to Meades Ranch located in Kansas shown by the big circle in the center. The numbers inside the regions depict the error in a loop after the optimization is completed. Figure courtesy of (Schwarz, 1989).

which allowed for computing the matrix inversion in one shot.

3.4.5 Boltz Method

The Boltz method is an alternative to the Gaussian elimination with LU decomposition for solving large set of equations in *one shot* (Boltz, 1923, 1938). An explanation of the Boltz method was provided by Wolf, which basically says that Boltz was tabulating matrices and their corresponding inverted solution (Wolf, 1952). Boltz was able to simply *look up* the solutions for sub-problems from a table instead of recomputing the inverse.

The central question here is how to setup the linear system so that the matrix of the normal equations, which needs to be inverted, reappears during the calculations. In general, for this method to work infinitely many matrices would need to be tabulated. To keep the number of tabulated matrices tractable, Boltz proposed to divide the equations into two groups:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{w} \end{bmatrix}. \tag{3.9}$$

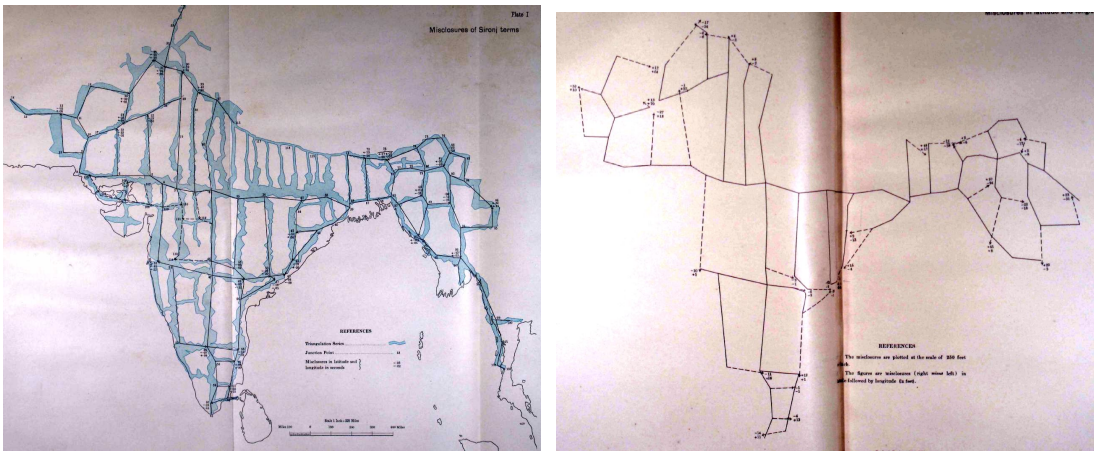
Here, $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$ are coefficients of the normal matrix and $\mathbf{x}$ and $\mathbf{y}$ are unknowns. Boltz proposed to cache $\mathbf{A}^{-1}$ and solve the reduced system of equation via

$$\left(\mathbf{C} - \left(\mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}\right)\right)\mathbf{y} = \left(\mathbf{w} - \mathbf{B}^T\mathbf{A}^{-1}\mathbf{v}\right). \tag{3.10}$$

The key idea is to separate angle-only measurements from all others. Let $\mathbf{x}$ be the unknowns that arise from angle-only measurements, while $\mathbf{y}$ arise from all other measurements (see also (Hough, 1948)). The constraints in $\mathbf{x}$ are simple given the triangular structure of the net: the sum of the angles in a triangle equals 180°, the sum of angles around a point must add up to 360°. Hence, the coefficients of the matrix $\mathbf{A}$ in Equations (3.9 and 3.10) can be written so that it contains only 1's, 0's and -1's. This, in combination with domain knowledge about the structure of the triangular nets, allows for efficiently caching $\mathbf{A}^{-1}$. Boltz method is designed for triangle nets in which the structure repeats itself. This is often the case in geodetic mapping.

The look up was done by humans but we could not find details about the procedure for physically storing and retrieving the inverted matrices. One might even spot similarities between Boltz' proposal of caching matrix inversions and techniques like lifted probabilistic inference (Kersting, 2012). Caching can result in a substantial performance gain if majority of operations are done manually as was the case for the ED 50. Boltz' method was successfully used for eliminating up to 80 by 80-sized matrices when creating ED 50 (Hough, 1948).

Note that reduction in Equation (3.10) is the Schur compliment but the motivation behind using the Boltz method is the fact that $\mathbf{A}^{-1}$ is tabulated and looking up the inverse



(a) (b)

Figure 3.11: The Indian triangulation net (left) and the section network with errors in circuit points (right). The blue region shows the area covered by sections of nets and the solid line is the represented single section. The dotted lines are sections, which complete loops. The point of intersection between a solid and a dotted line is a circuit point, which has a latitude and longitudinal error induced by the dotted line. Figures courtesy of (Bomford, 1939).

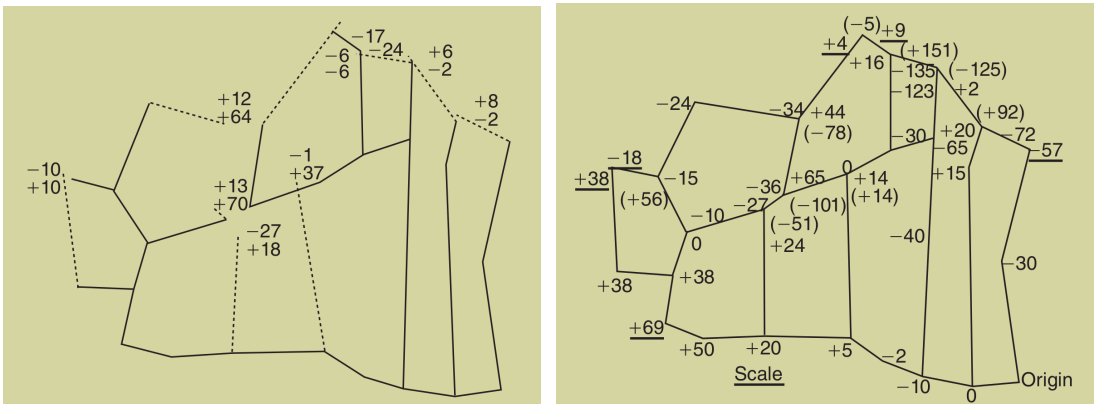
is substantially faster than computing it (by hand). The geodetic researchers prior to 1950 mention that the Boltz method allowed solving least squares without biasing the solution towards any particular unknown. They mention that the Boltz method treated all unknown equally unlike the Gaussian elimination which caused the last variable to contain larger errors.

3.4.6 The Indian Method of 1938

Bomford proposed a different approach than the above mentioned methods for solving the triangulation net for the Indian subcontinent in 1938 (Bomford, 1939). He mentions that the Bowie method was infeasible for optimizing the network of towers in India, as the number of astronomical stations was too small and each junction in the Bowie method requires to have an astronomical station with independent latitude and longitude observations.

The starting point of the Indian method is similar to the Bowie method. Junctions and

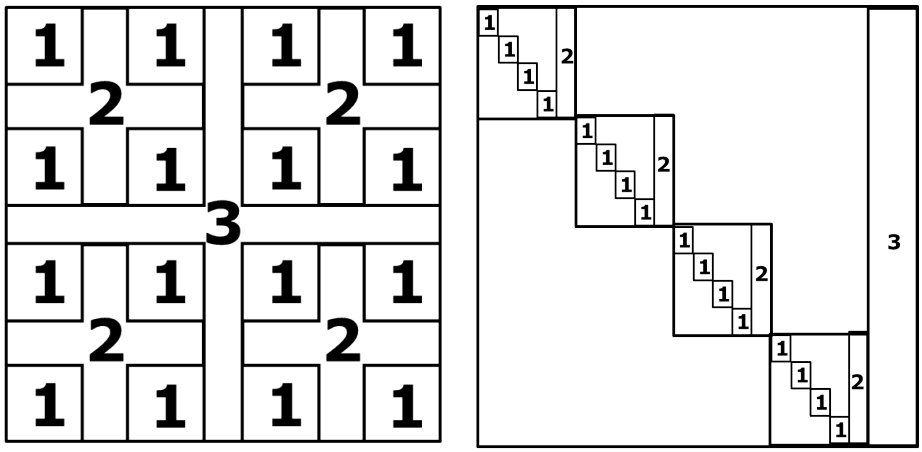
From American Congress on Surveying et al. (1994): "A method devised by Boltz for solving the normal equations occurring in the adjustment of a triangulation network; it allows a large set of equations to be solved in one straight operation by the method of least squares without biasing the solution towards any particular unknown. Initially, the normal equations were solved using the Gaussian method of successive elimination. This method, however, causes the last determined value to contain larger errors than the first determined value. Boltz's method treats all unknowns equally and is particularly suitable for solving large systems of equations."



(a) (b)

Figure 3.12: A zoomed in view on the North-East section of the Indian subcontinent with the circuit errors. It shows the initial and final configuration of the triangle net used in 1939 for surveying India. The solid lines represent the initial spanning tree. The dotted lines are sections which induce circuit points and thus a residual error. Each circuit point contains the latitudinal residual error (left) and longitudinal residual error (right).

sections are created from the triangle nets. These junctions are points of intersection of sections and do not require any astronomical constraints. This is illustrated in Figure 3.11(a). A spanning tree consisting of sections and junctions is chosen from the mesh network of the triangle nets. The initial spanning tree does not have errors as there are no loops and it is depicted by the solid lines in Figure 3.11(b). All other sections that are not part of the spanning tree will introduce a residual error (see dotted lines). Bomford refers to the points where multiple sections meet and have a residual error as circuits. Circuit points are similar to nodes in a SLAM pose graph with loop closures. These circuit points have a residual in latitude and longitude. A zoomed-in portion of the North-east part of the map is shown in Figure 3.12. Figure 3.12(a) illustrates the errors in latitude and longitude at each circuit point. Bomford proposed that the error in circuit points could be solved by distributing it around the loop. He had no automated system to distribute errors around the loop. He manually performed trial and error methods to reduce the total error induced in the circuit nodes. For example, he states that he first reduced the longitudinal error in the North West regions, shown in Figure 3.12(a), by adjusting longitudinal values of the lower section. He incorporates the “stiffness” of sections, which he approximated by the length and uncertainty of the sections (Bomford, 1939). This method does not appear to be as rigorous as the Bowie or Helmert methods but still resulted in accurate maps of the Indian subcontinent.



(a) A planar mesh partitioned according to nested dissection.

(b) The corresponding matrix picture for this net.

Figure 3.13: Nested dissection and the corresponding matrix arrangement. The four higher block with nodes numbered 1 and 2 are independent given the separator 3. All sub-blocks numbered 1 are independent given the sub-block numbered 2.

3.4.7 Variable Ordering

In Section 3.4.1 we outlined the Helmert Blocking method but did not mention how the partitioning was performed. Helmert himself provided simple instructions for creating subnets and for partitioning the blocks, which is critical for his method. He first instructed to pick a latitude such that it partitions all the towers roughly into halves. Next, a longitude is chosen for each upper and lower half to partition the Northern and Southern regions into Eastern and Western partitions. Each obtained geographical rectangular block is recursively partitioned into four further blocks. This strategy is illustrated in Figure 3.4(d) and Figure 3.4(e). The proposed method is simple but effective, since the triangulation network is built roughly as a planar graph and the density of the net was approximately similar across different locations. Helmert's approach to partition the triangle nets also shares similarities to the nested dissection variable reordering strategies (George, 1973) used to efficiently factorize sparse matrices. The use of variable re-ordering significantly improves the computation and memory requirements for matrix decomposition methods (Davis, 2006b; Agarwal and Olson, 2012).

The nested dissection variable reordering scheme was initially proposed for solving a system of equations for n×n grid problems arising from finite-element discretization of a mesh (George, 1973). It partitions the graph with "+" shapes, as illustrated in Figure 3.13. Each resulting block is then recursively partitioned with a "+" shape. The number on each partition corresponds to the entry in the coefficient matrix. Helmert's proposal to divide a triangulation net recursively along latitudes and longitudes was actually used by

Avila and Tomlin for solving large least squares using the ILLIAC IV parallel processor for optimizing geodetic networks (Avila and Tomlin, 1979). Later, Golub and Plemmons (1980) used the Helmert blocking variable ordering strategy for solving large system of equations using orthogonal decomposition techniques such as QR decomposition for the system of equations arising from the geodetic network.

Given that both, Helmert's proposal and the nested dissection algorithm, are so similar, researchers performed a study to understand the best way to create the separators and to join the blocks given a four-way partitioning for NAD 83 (Schwarz, 1989). Figure 3.14 shows four ways of joining a block partitioned into four sub-blocks. The partitioning can be done using either Helmert's strategy or nested dissection. The key design choice is whether to use a deep tree or a broad tree strategy, see Figure 3.15. The deep tree strategy creates smaller and denser final blocks while broad trees result in larger and sparser blocks. This can be seen from Figure 3.15 (left) and Figure 3.15 (right) for a simple example. The large sparse matrix in the broad tree strategy implies further variable reordering to minimize the matrix fill-in. In the deep tree strategy, parallelism can be exploited better, but it requires more matrix allocations. In the simple example shown in Figure 3.15, seven matrices are allocated for the deep tree compared to five allocations for the broad tree. Hence, the deep tree strategy was preferred for NAD 83 to enable more parallelism and as the created, dense sub-blocks do not require further reordering.

3.4.8 Removing Outliers for Geodetic Mapping

The task of traditional geodetic mapping involves hundreds of surveyors working in parallel to obtain measurements. This also results in many faulty constraints (Schwarz, 1989). The typical source of faulty constraints are human errors, errors in instruments, and sometimes errors when transferring entries from physical journals into the database management systems. For NAD 83, erroneous constraints are detected and removed through a block validation process. The block validation process validates constraints and towers in small geographical regions, one block at a time. The blocks are created by a geographical partitioning of stations as shown in Figure 3.16. These also represent the lowest level of Helmert blocks.

The block validation process consists of two iterating steps. First, by identifying under-constrained nodes and second, by checking for outliers given small blocks of nodes. If any node in a block is under-constrained, either additional constraints are added from neighboring blocks or the node is removed from the block. This avoids any singularities within the least squares solution. After this issue is resolved, the block is optimized and all constraints having a weighted residual error of greater than three ($\chi^2 > 3$) are evaluated. These are regarded as possible erroneous constraints. A constraint with a large residual error ($\chi^2 > 3$) is deleted if there is another constraint between the same nodes having a similar sensor modality but a smaller residual error. A constraint with

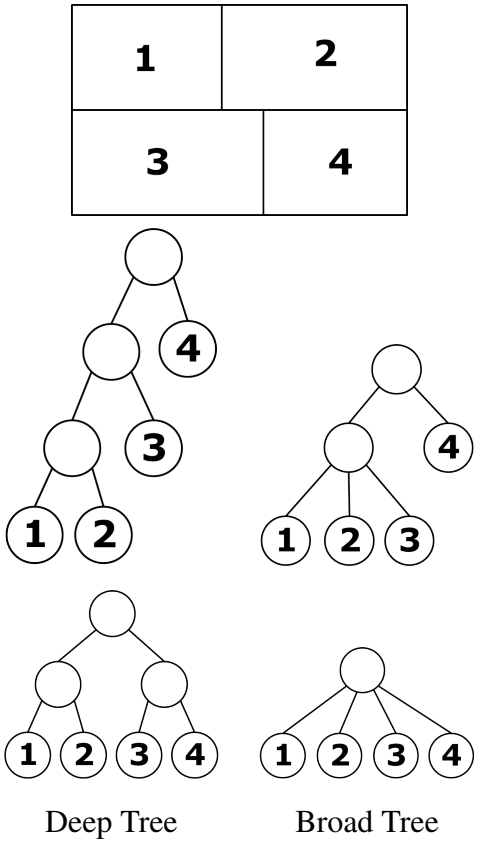


Figure 3.14: Possible different ways for joining four Helmert blocks according to (Schwarz, 1989). The figure illustrates a rectangular area partitioned into four blocks, which can be joined in four ways. Out of these four, deep tree and broad tree were the most interesting ones.

high residual error is also deleted if there are additional constraints between other nodes capable of constraining the nodes under consideration. If no additional constraints are found, the standard deviation of the possible outlier constrained is doubled until $\chi^2 < 3$. In other words, the assumed sensor noise for this constraint is increased, which is a similar principle behind M-estimators in robust statistics (Hartley and Zisserman, 2004; Zhang, 1997).

The whole process of optimizing a sub-block and evaluating constraints with large error is repeated till the block is free of outliers. For NAD 83, a total of 843 blocks were evaluated using block validation methods, as illustrated in Figure 3.16. Each block consisted of 300 to 500 nodes and required roughly one person month to validate (Schwarz, 1989). This process for outlier detection was also used as recently as 2007 for NAD83(NSRS2007), where seven trial solutions were carried out to account for outliers, singularities and handle weakly determined areas (Pursell and Potterfield, 2008).

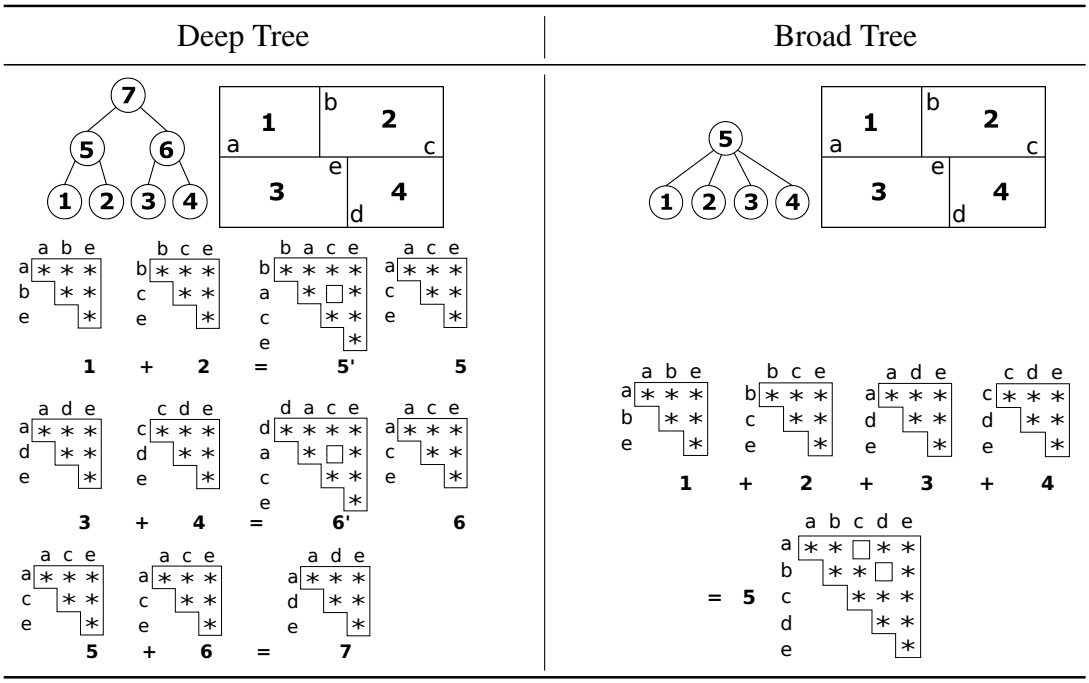


Figure 3.15: Comparison between deep tree (left) and broad tree (right) ordering according to (Schwarz, 1989). The area is partitioned into four blocks labeled 1, 2, 3 and 4. *a* to *e* represents separator variables. The variable set *e* are separators with constraints in all 4 blocks, *a* are variables with constraints only between blocks 1 and 3, *b* are variables constraining between blocks 1 and 2 only, etc. In this example, the deep tree approach would merge two subgraphs at a time, while the broad tree approach merges four subgraphs at a time. In both methods all low level subgraphs labeled 1 to 4 are processed in parallel. The deep tree method requires two more matrix allocation compared to the broad tree approach but the broad tree approach results in a larger matrix with zero blocks, which require additional variable re-ordering to reduce computation.

3.5 Relationship between Geodetic Mapping Methods and Graph-Based SLAM

In the previous sections we outlined various map optimization techniques used by the geodetic mapping researchers. In this section we highlight some of the similarities between the back-end methods developed by both the geodetic and robotic communities.

3.5.1 Hierarchical and Sub-Map-Based SLAM Methods

Grisetti et al. (2010b) propose an efficient hierarchical multi-level method for optimizing pose-graphs. The higher the level in the hierarchy, the smaller the pose-graph. At each level, a sub-graph from a lower level is represented by a single node on the higher level. The optimization is primarily carried out at the higher levels and only propagated down

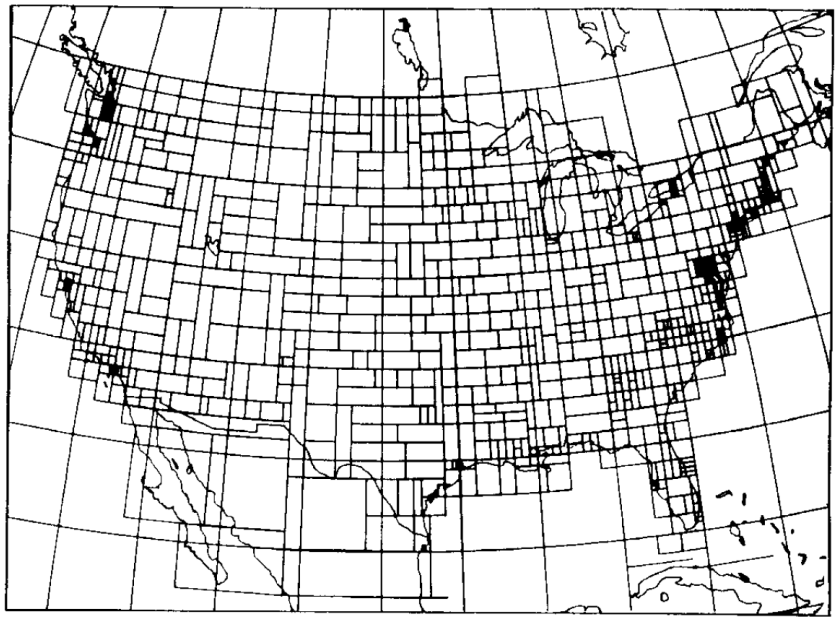


Figure 3.16: Boundaries of blocks of data used in the block validation process. Figure courtesy of (Schwarz, 1989).

to the lower level if there is a significant update to the pose of a node. Each edge in the higher level is computed via a new deduced virtual measurement, which is created after a local optimization. The gain in speed by this method results from the fact that computationally expensive optimization is only performed when the pose of a node in the coarse representation gets affected more than a certain threshold (which was chosen as 5 cm or 0.05° by the authors). There exists also an extension to general SLAM graphs and bundle adjustment problems (Grisetti et al., 2012).

The virtual measurements created by Grisetti et al. are similar in idea to those in the Bowie method explained in Section 3.4.3. The Bowie method creates a two-level hierarchy instead of the multiple levels as in (Grisetti et al., 2010b). Both methods use a single node from a dense sub-graph in the higher, coarser level and add a virtual deduced measurement between the smaller new problem instances. A difference is that given a geodetic network consisting of sections, each edge in the Bowie method represents a sub-graph whereas in the hierarchical approach of Grisetti et al., only nodes represent sub-graphs. Furthermore, Grisetti et al. compute the uncertainty of a virtual constraint explicitly.

Also Ni et al. (2007) propose back-ends, which divide the original problem into multiple smaller sub problems. These smaller problems are then optimized in parallel. The first approach, (Ni et al., 2007), partitions the problem into a single level of sub-maps, while (Ni and Dellaert, 2010) partitions each map recursively multiple times. The speed-up is mainly due to caching the linearization result of each sub-graph. In both methods,

all nodes in a sub-graph are expressed with respect to a single base node. This allows for efficiently re-using the linearization of the constraints within each sub-graph. This insight results in a reduction of the total computation time. Boundary nodes and constraints connecting multiple subgraphs need to be recomputed at each iteration, while results for nodes within a sub-graph can be reused. The methods of Ni et al. would result in a batch solution if each non-linear constraint within a sub-graph is re-linearized at every iteration, but they show experimentally that by not doing so, a high speed-up in optimization time is achieved at the expense of small errors.

The sub-maps methods proposed by Ni et al. show many similarities to Helmert's approach of partitioning the triangle nets into subnets. The idea of anchoring sub-maps in (Ni and Dellaert, 2010) has a similar motivation as Bowie's idea for anchoring junctions and moving them as a whole by shifting the anchor nodes and not re-optimizing each sub-graph. Krauthausen et al. (2006) prove that approximately planar SLAM graphs can be optimized in $O(n^{1.5})$ by using the nested dissection reordering. The nested-dissection algorithm, which is at the heart of (Ni and Dellaert, 2010) and (Krauthausen et al., 2006), is similar to Helmert's strategy of recursively partitioning a planar triangulation net (Golub and Plemmons, 1980). Furthermore, the out-of-core parallel Cholesky decomposition-based non-linear solver for geodetic mapping proposed by Avila and Tomlin (1979) uses Helmert blocking and is thus strongly connected to (Ni and Dellaert, 2010). It should, however, be noted that the geodetic community was generating the partitioning manually with domain knowledge and that the triangle-nets have a simpler and more planar structure than typical SLAM graphs.

3.5.2 Stochastic Methods

Olson et al. (2006) propose stochastic gradient descent for solving pose-graphs with a bad initialization. They re-parametrize the constraints from a global pose to a relative pose representation. In the global pose, all positional nodes are represented with global coordinates in the world frame, whereas in the relative pose representation, each pose is represented with respect to its previous pose in the odometry chain. This allows the authors to distribute the errors of each constraint within the loop induced by it. Grisetti et al. (2007b, 2009) improve this approach by using a tree parameterization based on a spanning tree instead of the odometry chain. The basic idea of both approaches is to stochastically select a constraint and move nodes based on the parameterization to reduce the error induced by the constraint. A learning rate controls the step size and is gradually decreased to prevent oscillations. Both approaches also assume spherical covariances and intelligent data-structures to efficiently distribute the errors. These methods are also capable of online and incremental approaches by increasing the learning rate for active areas (Olson et al., 2007; Grisetti et al., 2007a, 2008).

A major intuition of using stochastic methods is equating error around a loop to zero.

This is somewhat similar to the motivation for the modified Bowie method, where one virtual equation is set-up for each loop in the sparse mesh and the error around every loop is equated to zero. Examples of distributing errors around loops has also been explored in the "zero-sum property" of (Carlone et al., 2011) and "trajectory bending" of (Dubbelman et al., 2012).

The Indian method of 1938 (Bomford, 1939) described in Section 3.4.6 has a similar motivation to the stochastic methods described above. In the Indian method, an initial spanning tree is manually chosen, which is a similar initialization strategy than the one of Grisetti et al. The Indian method as a whole, however, is rather informal and distributes the errors manually in a trial and error fashion. In contrast to that, Olson et al. and Grisetti et al. minimize the error using stochastic gradient descent and provide a more formal treatment of the approach.

3.5.3 Robust Methods for Outlier Rejection

For quite a while, SLAM back-ends suffered from data association failures that result in wrong constraints between nodes in the graph. Recently, a set of different approaches have been proposed that are robust even if a substantial number of constraints are outliers.

Latif et al. (2012b) propose RRR, which is a robust SLAM back-end, capable of rejecting false constraints. RRR first clusters mutually consistent and topological related constraints together. Each cluster is checked for intra-cluster consistency by comparing the residual of each constraint with a theoretical bound. Constraints that do not satisfy the bound are removed. This approach is similar to the block validation technique used for NAD 83. The individual blocks in block validation and the clusters in RRR conceptually represent similar sub-graphs. The blocks are geographical partitions but are actually a set of nodes and constraints similar to what clusters represent in RRR. Again, it should be noted that the triangular networks of the geodetic community have a simpler structure than SLAM graphs and thus the verification step is easier to conduct.

Dynamic covariance scaling (DCS) is a robust back-end that is able to reject outliers by scaling the covariance of the outlier constraint. DCS can be formulated as a generalization of switchable constraints (Sünderhauf and Protzel, 2012), another state-of-the-art back-end for robust operation in the presence of outliers. In DCS, the covariance matrix of constraints with large residuals is scaled such that the error stays within a certain bound. This is related to the "doubling of standard deviation of each constraint" strategy used in (Schwarz, 1989). It is, however, not clear if and how the scaling in NAD 83 was modified between different iterations. DCS is covered in details in Chapters 5 and 6.

3.5.4 Linear SLAM

A good initial guess is critical for iterative non-linear methods to converge to the correct solution. For providing a good initialization, Carlone et al. (2011) propose a linear approximation for planar pose-graphs. Their approach yields an approximate solution to the non-linear problem without any initial guess and thus can be used as a high quality initial guess for state-of-the-art back-ends. Carlone et al. (2011) formulate the SLAM problem in terms of graph-embedding and suggest to partition the system of equations into:

$$\mathbf{A}_2^T \boldsymbol{\rho} = \mathbf{R}(\boldsymbol{\theta})\boldsymbol{\Delta}^l \tag{3.11}$$

$$\mathbf{A}_1^T \boldsymbol{\theta} = \boldsymbol{\delta}. \tag{3.12}$$

Here, $\boldsymbol{\theta}$ contains angular unknowns and $\boldsymbol{\rho}$ represents the positional unknowns. The terms $\mathbf{A}_1$ and $\mathbf{A}_2$ are their respective coefficient matrices, $\boldsymbol{\Delta}^l$ and $\boldsymbol{\delta}$ are the corresponding constraint residuals, and $\mathbf{R}(\boldsymbol{\theta})$ is the stacked rotation matrix. Equation (3.12) is solved first and the computed value of $\boldsymbol{\theta}$ is used for solving Equation (3.11). The authors provide both, a theoretical proof and real-world examples for the algorithm in (Carlone et al., 2011).

Related to Linear SLAM, Boltz was reordering the equations in Equation (3.9) and eliminating angular unknowns first. His motivation was to cache matrix inversions for a faster human look-up, but geodetics used it not only for gains in speed but also because it "did not bias the solution towards any particular unknown like Gaussian elimination" (American Congress on Surveying et al., 1994). In our geodetic survey, we did not find any proof for this statement, but as both approaches share similarities, we suspect that this statement is related to the properties of Linear SLAM. The theoretical justification of (Carlone et al., 2011) also holds in the case of Boltz' reordering of eliminating angular constraints first.

3.5.5 Dense Sub-Blocks

The manuscript (Schwarz, 1989) describing NAD 83 offers several interesting aspects ranging from database management to memory and cache-friendly algorithms. During the project, engineers used punch card machines—rather inconvenient and cumbersome tools for solving large matrix problems compared to modern computers. Physical file management system and efficient card storage are reminiscent of sophisticated techniques used in modern software. We have decided to not discuss all topics in detail here but in essence, the NAD 83 engineers were arranging data similar to the block matrix representations. The punch cards describing the data comprising station positions and constraints were also physically stored as low-level Helmert Blocks. All punch cards corresponding to a single block were stored together for faster retrieval.

Such dense sub-blocks are central elements in some of the fastest SLAM back-ends. Konolige et al. (2010) describes a 2D SLAM implementation where the computation time is significantly reduced by storing each Jacobian as a 3×3 block matrix. Kümmerle et al. (2011) generalize the block storage and indexing strategy for 6×6 and other types of feature nodes. Finally, most modern graph-based SLAM implementations use the Cholesky decomposition algorithm from the Suite-Sparse library by Davis (2006a). The fastest super-nodal Cholesky decomposition routine, CHOLMOD, also exploits dense sub-blocks (Chen et al., 2008).

3.6 Further Remarks

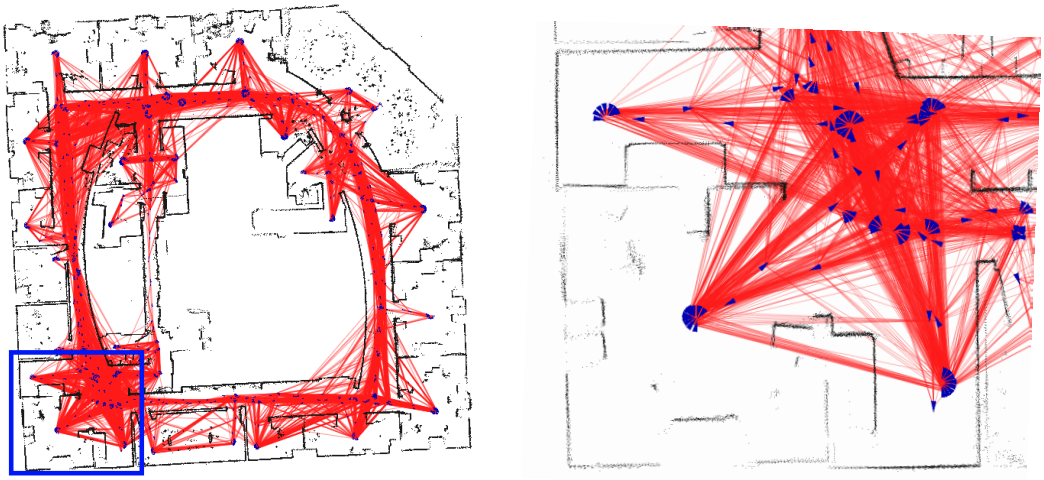
The use of sparse linear algebra and matrix decomposition methods have been introduced in robotics only recently (Dellaert and Kaess, 2006; Kaess et al., 2007b; Kümmerle et al., 2011; Kaess et al., 2012). In contrast to that, the geodetic scholars were using LU, QR and Cholesky decomposition for a long time—Cholesky decomposition was developed in the early 1900s by André-Louis Cholesky for geodesy and map building while he was in the French Geodetic section. The χ^2 distribution was also published in Helmert (1876). This is further elaborated in (Kruskal, 1946) and details with respect to Pearson's report can be found in Sec. 7.3 of (Sheynin, 1995).

Although there are similarities between the problems of both communities, it is also important to highlight the additional challenges that autonomous robots, which rely on working SLAM solutions, face compared to geodetic mapping.

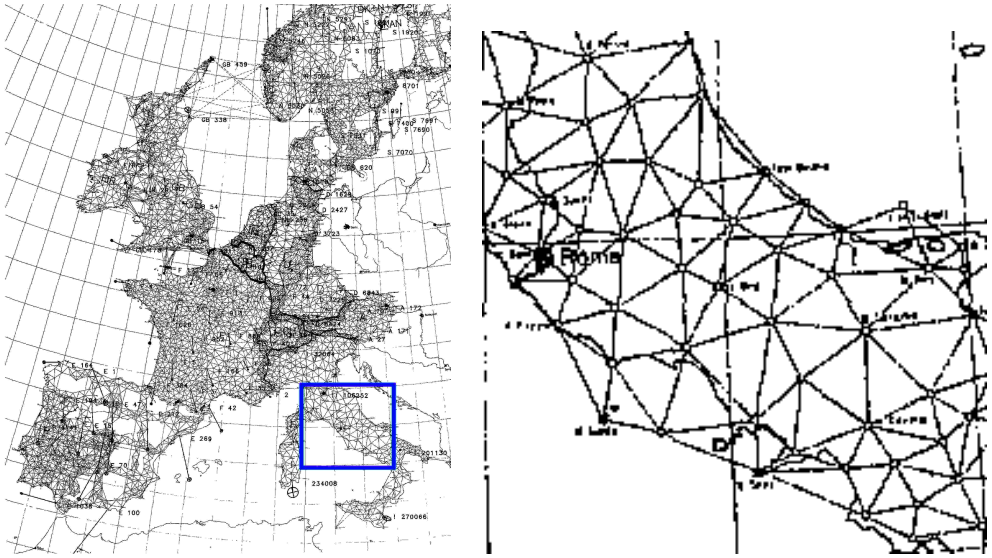
First, SLAM systems are autonomous while geodetic mapping inherently involves humans at all levels of the process. It is difficult for automated front-end data-association methods to distinguish between visually similar but physically different places and this is likely to occur, for example, in large man-made buildings. Perceptual aliasing creates false constraints, which often introduces errors in the localization and map building process.

Second, the quality of the initial guess is often different. The initial guess that is available for geodetic triangle networks are typically better than the pose initializations of typical wheeled robots using odometry as well as flying or walking robots. A good initial guess simplifies and even enables the use of polygon filling and other types of approximations.

Third, the geodetic triangle networks are almost planar, while most SLAM graphs are not. This can be intuitively seen from Figure 3.17. Additionally Eiffel-tower type landmarks which connect all poses, create highly non-planar SLAM graphs (Dellaert et al., 2010). Helmert's simple partitioning scheme of segmenting along latitudes and longitudes works for Geodetic networks because the graphs are almost planar. In (Krauthausen et al., 2006), the authors prove that planar or approximately planar SLAM graphs can



(a) Intel dataset with 875 robot positions and 15675 constraints (left) and a zoomed in section (right). The robot positions are shown as blue triangles and constraints in red. The zoomed in section of the bottom left portion intuitively shows the non-planarity of a SLAM graph.



(b) Triangle net for ED87 (left) and a zoomed-in section for Italy (right). The constraints covering Italy intuitively show that triangle nets used for geodesy were almost planar. Figure courtesy of (Ehrnsperger, 1991).

Figure 3.17: Intuitive illustration of planarity of geodetic triangle net compared to a SLAM graph.

be optimized in $O(n^{1.5})$ by using the nested dissection reordering. Comparable results can be expected for Helmert's Blocking strategy as both are rather similar (Avila and Tomlin, 1979). The way most modern SLAM methods work, however, leads to a highly non-planar graph with a high crossing number (Hliněnỳ and Salazar, 2007).

3.7 Summary

This chapter surveys geodetic mapping methods and aims at providing a geodetic perspective on SLAM. We showed that both fields share similarities when it comes to the error minimization task: maps are large, computational resources are limited and incremental methods are required, non-linear factors require procedures which iteratively re-linearize and data associations may be erroneous. There are, however, also differences: geodetic triangular nets have a simpler structure, which can be exploited in the optimization, methods for robotics must be autonomous while the geodetic surveys always have humans in the loop, and often the geodetic community had a better initial configuration to begin with.

Besides the elaborated similarities and differences between geodetic mapping and SLAM, we surveyed several core techniques developed by the geodetic community and related them to state-of-the-art SLAM methods. The central motivation for this chapter is to connect both fields and to enable future synergies among them. While surveying the geodetic methods, we experienced strong respect towards the geodetic scholars. Their achievements, especially given their lack of computational resources, are outstanding.

SLAM researchers have often gone back to the graph theory and sparse linear algebra community for efficient algorithms. It is probably worth to also look into the geodetic mapping literature given that they addressed large-scale error minimization and developed highly innovative solutions to solve them. Actually, several research activities by linear algebra researchers have been motivated by the large problem instances of geodetic mapping.

There might still be more methods in geodetic mapping that are unknown outside their community but could inspire other fields. Interested readers should begin with the detailed document by Schwarz (1989) on the history of North American Datum of 1983.

Chapter 4

Max-Mixtures

The central challenge in robotic mapping is obtaining reliable data associations. State-of-the-art inference algorithms can fail catastrophically even if one erroneous loop closure is incorporated into the map. Consequently, much work has been done to push error rates closer to zero. However, a long-lived or multi-robot system will still encounter errors, leading to system failure. In this chapter, we propose a fundamentally different approach: allow richer error models that allow the probability of a failure to be explicitly modeled. Our approach leads to an fully-integrated Bayesian framework for dealing with error-prone data and multi-modal distributions. Unlike earlier multiple-hypothesis approaches, our approach avoids exponential memory complexity and is fast enough for real-time performance. We show that the proposed method not only allows loop closing errors to be automatically identified, but also that in extreme cases, loop-validation systems can be unnecessary. We demonstrate our system both on standard benchmarks and on the real-world datasets that motivated this work.

4.1 Introduction

As explained in Chapter 1, a central challenge in SLAM is to recognize a previously visited place without making errors. This is an essential requirement for deploying uninterrupted autonomous mobile robots. The conventional strategy is to build better front-end systems. Indeed, much effort has been devoted to creating better front-end systems (Neira and Tardos, 2001; Bailey, 2002; Olson, 2009b), and these approaches have succeeded in vastly reducing the rate of errors. But for systems that accumulate many robot-hours of operation, or robots operating in particularly challenging environments, even an extremely low error rate still results in errors. These errors lead to divergence of the map and failure of the system.

Additionally, relying on algorithms which can only handle uni-modal Gaussian distributions is limiting. Gaussian error models are convenient as the maximum likelihood

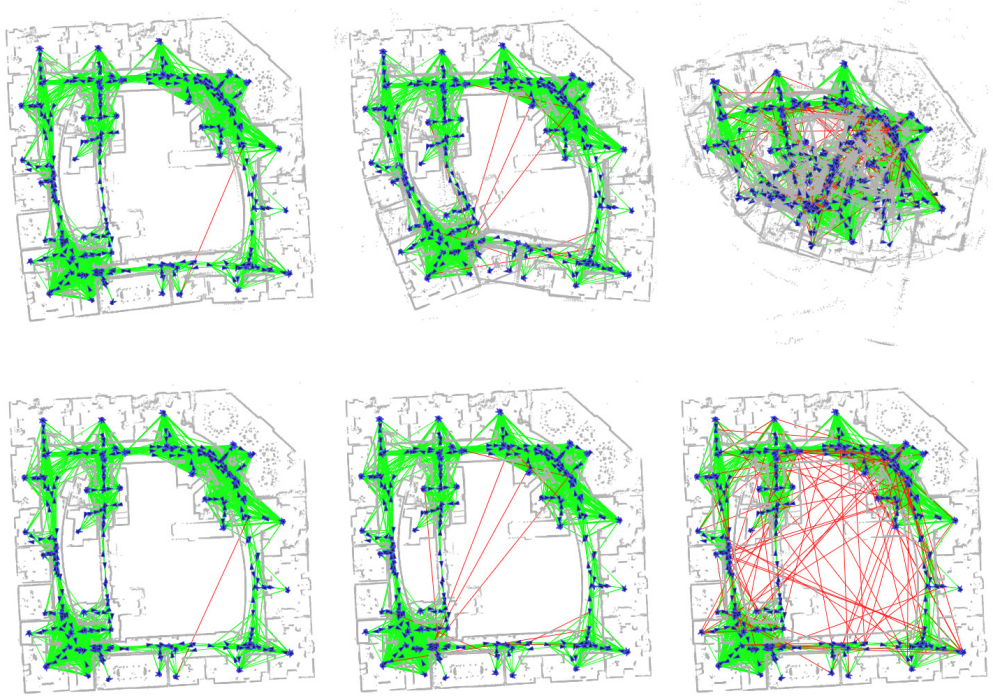


Figure 4.1: Recovering a map in the presence of erroneous loop closures. We evaluated the robustness of our method by adding erroneous loop closures to the Intel data set (Howard and Roy, 2003). The top row reflects the posterior map as computed by a state-of-the-art sparse Cholesky factorization method with 1, 10, and 100 bad loop closures. The bottom row shows the posterior map for the same data set using our proposed max-mixture method. While earlier methods produce maps with increasing global map deformation, our proposed method is essentially unaffected by the presence of the incorrect loop closures.

estimate leads directly to the least square problem which can be solved efficiently using sparse linear algebra methods (see Chapter 2). The uni-modal Gaussian error models though efficient, cannot handle many hard SLAM problems. These include perceptual aliasing due to repeated structure in the environment or bi-modal distributions resulting from the occasional wheel slippage on a robot. Similar multi-modal disjoint hypothesis occur with sensors such as radar, sonar, and GPS.

In this chapter, we propose a novel approach that allows efficient maximum-likelihood inference on SLAM graphs that contain arbitrarily complex probability distributions. This is in contrast to state-of-the-art graph-based SLAM methods, which are limited to uni-modal Gaussian distributions, and which suffer from the real-world problems described earlier. Specifically, we propose a new type of mixture model, a *max*-mixture, which provides similar expressivity as a sum-mixture, but avoids the associated computational costs. With such a mixture, the bi-modal wheel slippage problems can be modeled with two Gaussians, one modeling the wheel slipped and the other modeling the wheel gripped.

Additionally, loop closures can be accompanied by a "null" hypothesis to explain an error in the front-end. In essence, the back-end optimization system serves as a part of the front-end— playing an important role in validating loop closures and preventing divergence of the map.

We will demonstrate our system on real data, showing that it can easily handle the error rates of current front-end data validation systems, allowing robust operation even when these systems produce poor output. We will also illustrate that, in extreme cases, *no* front-end loop validation is required at all: all candidate loop closures can simply be added to the SLAM graph, and our approach simultaneously produces a maximum likelihood map while identifying the set of edges that are correct. This is an interesting development, since it provides a fully integrated Bayesian treatment of both mapping and data association, tasks which are usually decoupled.

It has previously been shown that exact inference on even poly-trees of mixtures is NP-hard (Lerner and Parr, 2001). Our method avoids exponential complexity at the expense of guaranteed convergence to the maximum likelihood solution. We formulate a new mixture model that provides significant computational advantages over the more traditional sum-of-Gaussians mixtures, while retaining similar expressive power. Our algorithm allows for fast maximum-likelihood inference on networks containing these max-mixtures demonstrating how robot mapping systems can use these methods to robustly handle errors in odometry and loop-closing systems.

The remainder of this chapter is organized as follows. In the next section we outline prior work in the area of back-ends robust to outliers. Additionally, we cover methods which can handle non-Gaussian error models. In Section 4.3, we introduce our method, both from a least squares perspective as well as from a probabilistic Bayesian reasoning. We provide an extensive evaluation of our method in Section 4.4. Our first set of experiments evaluate the robustness of our method in the presence of simulated outliers in 2D and 3D pose-graphs. Next, we evaluate our method on multi-modal factors which occur due to wheel slippage or ambiguity in the appearance of the environment. This ability is a feature of our approach. Finally, we characterize the robustness of our method to local minima, identifying factors such as error rate and overall graph degree and their impact. We show that the basin of convergence is large for a variety of benchmark 2D and 3D datasets over a range of parameter values.

4.2 Related Work

We are not the first to consider estimation in the presence of non-Gaussian noise. Two well-known methods allow more complex error models to be used: particle filter methods and multiple hypothesis tracking (MHT) approaches.

Particle filters, perhaps best exemplified by FastSLAM (Montemerlo, 2003), approxi-

mate arbitrary probability distributions through a finite number of samples. Particle filters attempt to explicitly (and non-parametrically) describe the posterior distribution. Unfortunately, the posterior grows in complexity over time, requiring an ever-increasing number of particles to maintain the quality of the posterior approximation. This growth quickly becomes untenable, forcing practical implementations to employ particle resampling techniques (Hähnel et al., 2003a; Kwak et al., 2007; Stachniss et al., 2005). Unavoidably, resampling leads to a loss of information, since areas with low probability density are effectively truncated to zero. This loss of information can make it difficult to recover the correct solution, particularly after a protracted period of high uncertainty (Bailey et al., 2006; Grisetti et al., 2005).

Multiple Hypothesis Tracking approaches (Durrant-Whyte et al., 2003; Blackman, 2004) provide an alternative approach more closely related to mixture models. These explicitly represent the posterior using an ensemble of Gaussians that collectively encode a mixture. However, the size of the ensemble also grows rapidly: the posterior distribution arising from N observations each with c components is a mixture with c^N components. As with particle filters, this exponential blow-up quickly becomes intractable, forcing approximations that cause information loss and ultimately lead to errors.

In the special case where errors are modeled as uni-modal Gaussians, the maximum likelihood solution of the graph-based SLAM network can be found using non-linear least squares. Beginning with the observation that the information matrix is sparse (Thrun and Liu, 2003; Walter et al., 2005; Eustice et al., 2006), efforts to exploit that sparsity resulted in rapid improvements to map inference by leveraging sparse factorization and good variable-ordering heuristics (Dellaert and Kaess, 2006; Kaess et al., 2008; Konolige et al., 2010; Agarwal and Olson, 2012). While the fastest of these methods generally provide only maximum-likelihood inference (a shortcoming shared by our proposed method), approximate marginal estimation methods are fast and easy to implement (Bosse et al., 2004; Olson, 2008). It is highly desirable for new methods to be able to leverage the same insights that made these approaches so effective.

Pfingsthorn and Birk (2012) have recently utilized Gaussian sum-mixtures for map optimization. The mixtures are converted into uni-modal Gaussians via a "pre-filtering" step, yielding a problem that can be approximately solved using standard sparse methods. Another approach for increasing robustness is to use the χ^2 of individual measurements in order to identify clusters of mutually-consistent loop closures (Latif et al., 2012b). This mutual consistency can be re-evaluated as new information arrives (Latif et al., 2012a). The "max-mixture" approach described in this chapter differs from these approaches in that the challenging process of approximating a sum-mixture is avoided, and that the set of activated modes is intrinsically re-evaluated at every iteration. In our approach, the selection of the single Gaussian component is re-evaluated after each optimization step. This allows initially unlikely components, to be later selected in the presence of more

consistent observations.

One method similar to our own explicitly introduces switching variables whose value determines whether or not a loop closure is accepted (Sünderhauf and Protzel, 2012). This work namely, Switchable Constraints, is notable for being the first to propose a practical way of dealing with front-end errors. In comparison to our approach, they penalize the activation/deactivation of a loop closure through a separate cost function (as opposed to being integrated into a mixture model), and must assign initial values to these switching variables (as opposed to our implicit inference over the latent variables). Our approach does not introduce switching variables, instead explaining poor quality data in the form of a non-Gaussian probability density function which can be arbitrarily complex (including having multiple maxima). Later in Chapter 5, we will outline a robust SLAM algorithm, namely dynamic covariance scaling which generalizes Switchable Constraints (Sünderhauf and Protzel, 2012) and has better properties.

Robust cost functions (Hartley and Zisserman, 2004) provide resilience to errors by reducing the cost associated with outliers, and have been widely used in the vision community (Strasdat et al., 2010; Sibley et al., 2009). The proposed method can approximate arbitrary probability distributions, including those arising from robust cost functions such as corrupted Gaussian.

Our proposed method avoids the exponential growth in memory requirements of particle filter and MHT approaches by avoiding an explicit representation of the posterior density. Instead, like other methods based on sparse factorization, our method extracts a maximum likelihood estimate. Critically, while the memory cost of representing the posterior distribution grows exponentially, the cost of storing the underlying graph-based SLAM network (which implicitly encodes the posterior) grows only linearly with the size of the network. In other words, our method (which only stores the graph) can recover solutions that would have been culled by particle and MHT approaches. In addition, our approach benefits from the same sparsity and variable-ordering insights that have recently benefited uni-modal approaches.

4.3 Approach

Our goal is to infer the posterior distribution of the state variables $\mathbf{x}$, which can be written in terms of the factor potentials in the SLAM graph. The probability is conditioned on sensor observations $\mathbf{z}$; with an application of Bayes' rule and by assuming an uninformative prior $p(\mathbf{x})$, we obtain:

$$p(\mathbf{x} \mid \mathbf{z}) \propto \prod_i p(\mathbf{z}_i \mid \mathbf{x}) \tag{4.1}$$

It is generally assumed that the factor potentials $p(\mathbf{z}_i \mid \mathbf{x})$ can be written as Gaussians:

$$p(\mathbf{z}_i \mid \mathbf{x}) = \frac{1}{(2\pi)^{n/2}|\mathbf{\Sigma}_i|^{1/2}} e^{-\frac{1}{2}(\hat{\mathbf{z}}_i(\mathbf{x})-\mathbf{z}_i)^T \mathbf{\Sigma}_i^{-1}(\hat{\mathbf{z}}_i(\mathbf{x})-\mathbf{z}_i)} \tag{4.2}$$

$\hat{\mathbf{z}}_i(\mathbf{x})$, the predicted measurement, is typically non-linear and can be approximated using a first-order Taylor expansion as explained in Equation (2.4).

The posterior maximum likelihood value can be easily solved in such cases by taking the logarithm of Equation (4.1), differentiating with respect to $\mathbf{x}$, then solving for $\mathbf{x}$. This classic least-squares approach leads to Equation (2.3) and hence a linear system of the form $\mathbf{A}\mathbf{x} = \mathbf{b}$ already explained in Chapter 2. Critically, this is possible because the logarithm operator can be "pushed" inside the product in Equation (4.1), reducing the product of N terms into a sum of N simple quadratic terms. No logarithms or exponentials remain, making the resulting expression easy to solve.

We might now consider a more complicated function $p_i(\mathbf{x} \mid \mathbf{z})$, such as a sum-mixture of Gaussians:

$$p(\mathbf{z}_i \mid \mathbf{x}) = \sum_j w_j \mathcal{N}(\mathbf{z}_i; \boldsymbol{\mu}_j, \mathbf{\Sigma}_j) \tag{4.3}$$

In this case, each $\mathcal{N}(\mathbf{z}_i; \boldsymbol{\mu}_j, \mathbf{\Sigma}_j)$ represents a different Gaussian probability distribution function. Such a sum-mixture allows great flexibility in specifying the distribution $p(\mathbf{z}_i \mid \mathbf{x})$. For example, we can encode a robust cost function by using two components with the same mean, but with different variances. More complicated distributions, including those with multiple maxima, can also be represented.

The problem with a sum-mixture is that the maximum likelihood solution is no longer simple: the logarithm can no longer be pushed all the way into the individual Gaussian components: the summation in Equation (4.3) prevents it. As a result, the introduction of a sum-mixture means that it is no longer possible to derive a simple solution for $\mathbf{x}$.

4.3.1 Max-Mixtures

Our solution to this problem is a new mixture model type, one based on a max operator rather than a sum:

$$p(\mathbf{z}_i \mid \mathbf{x}) = \max_j w_j \mathcal{N}(\mathbf{z}_i; \boldsymbol{\mu}_j, \mathbf{\Sigma}_j) \tag{4.4}$$

While the change is relatively minor, the implications to optimization are profound. The logarithm *can* be pushed inside a max mixture: the max operator acts as a selector, returning a single well-behaved Gaussian component.

A max mixture has much of the same character as a sum mixture and retains a similar expressivity: multi-modal distributions and robust distributions such as Corrupted

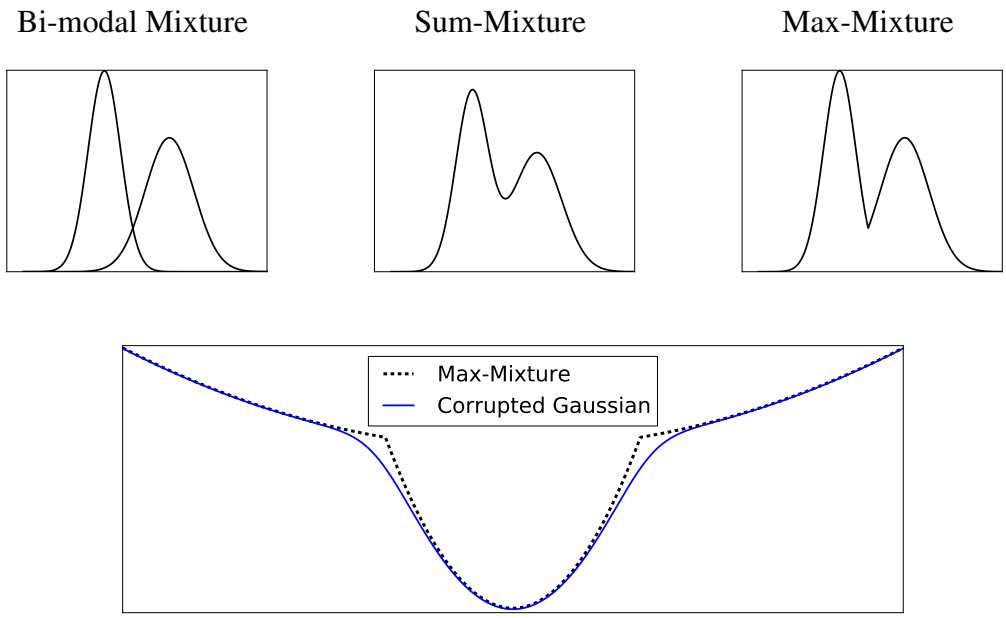


Figure 4.2: Mixture Overview. Given two mixture components (top left), the max- and sum-mixtures produce different distributions. In both cases, arbitrary distributions can be approximated. A robust cost function such as a Corrupted Gaussian (bottom), can be constructed from two Gaussian components with equal means but different variances.

Gaussian (Hartley and Zisserman, 2004) can be similarly represented (see Figure 4.2). Note, however, that when fitting a mixture to a desired probability distribution, different components will result for sum- and max- mixtures. Assuring that the distributions integrate to one is also handled differently: for a sum mixture, $\sum w_j = 1$ is a necessary and sufficient condition; for a max mixture, proper normalization is generally more difficult to guarantee. Usefully, for maximum likelihood inference, it is inconsequential whether the distribution integrates to 1. Specifically, suppose that some normalization factor γ is required in order to ensure that a max mixture integrates to one. Since γ is used to scale the distribution, the log of the resulting max mixture is simply the log of the un-normalized distribution plus a constant. The addition of such a constant does not change the solution of a maximum-likelihood problem, and thus it is unnecessary for our purposes to compute γ.

4.3.2 Cholesky-MM

We now show how max mixture distributions can be incorporated into existing graph optimization frameworks. The principle step in such a framework is to compute the Jacobian, residual, and information matrix for each factor potential. As we described previously, these are trivial to compute for a uni-modal Gaussian distribution.

For the max-mixture case, it might seem that computing the needed derivatives for the Jacobian is difficult: the max-mixture is not actually differentiable at the points where the maximum-likelihood component changes. While this makes it difficult to write a closed-form expression for the derivatives, they are none-the-less easy to compute.

The key observation is that the max operator serves as a selector: once the mixture component with the maximum likelihood is known, the behavior of the other mixture components is irrelevant. In other words, the solver simply iterates over each of the components, identifies the most probable, then returns the Jacobian, residual, and information matrix for that component scaled according to the weight w_j of that component. If the likelihood of two components are tied—an event which corresponds to evaluating the Jacobian at a non-differentiable point—we pick one of the components arbitrarily. However, these boundary regions comprise an area with zero probability mass.

The resulting Jacobians, residuals, and information matrices are combined into a large least-squares problem which we subsequently solve with a minimum-degree variable ordering heuristic followed by sparse Cholesky factorization using Gauss-Newton steps, in a manner similar to that described by (Dellaert, 2005; Grisetti et al., 2010a). With our modifications to handle max-mixtures, we call our system Cholesky-MM.

It is often necessary to iterate the full least-squares problem several times. Each time, the best component in each max-mixture is re-evaluated: in essence, as the optimization proceeds, we dynamically select the best mixture component as an integral part of the optimization process.

Even in the non-mixture case, this sort of non-linear optimization cannot guarantee convergence to the global optimal solution. It is useful to think of a given inference problem as having a "basin of convergence"— a region that contains all the initial values of x that would ultimately converge to the global optimal solution. For most well-behaved problems with simple Gaussian distributions, the basin of convergence is large. Divergence occurs when the linearization error is so severe that the gradient points in the wrong direction.

The posterior distribution for a network with N mixtures, each with c components, is a mixture with as many as c^N components. In the worst-case, these could be non-overlapping, resulting in c^N local minima. The global optimal solution still has a basin of convergence: if our initial solution is "close" to the optimal solution, our algorithm will converge. But if the basin of convergence is extremely small, then the practical utility of our algorithm will be limited.

In other words, the key question to be answered about our approach is whether the basin of convergence is usefully large. Naturally, the size of this basin is strongly affected by the properties of the problem and the robustness of the algorithm used to search for a solution. One of the main results of this chapter is to show that our approach yields a large basin of convergence for a wide range of useful robotics problems.

4.4 Applications and Evaluation

In this section, we show how our approach can be applied to several real-world problems. We include quantitative evaluations of the performance of our algorithm, as well as characterize its robustness and runtime performance.

4.4.1 Uncertain Loop Closures

We first consider the case where we have a front-end that produces loop closures with a relatively low, but non-zero, error rate. For each uncertain loop closure, we introduce a max-mixture consisting of two components: 1) the front-end's loop closure and 2) a null hypothesis. The null hypothesis, representing the case when the loop closure is wrong, is implemented using a mixture component with a large covariance. In our experiments, we set the mean of the null-hypothesis component equal to that of the other component. Weights and variances are assigned to these two components in accordance with the error rate of the front-end.

In practice, the behavior of the algorithm is not particularly sensitive to the weights associated with the null hypotheses. The main benefit of our approach arises from having a larger probability associated with incorrect loop closures, as opposed to the exponentially-fast decreasing probability specified by the loop closer's Gaussian. Even if the null hypothesis has a very low weight (for example 10^{-5}), it will provide a sufficiently plausible explanation of the data to prevent a radical distortion of the graph. Second, once the null hypothesis becomes dominant, its large variance results in a weak gradient for the edge. As a result, the edge plays virtually no role in subsequent optimization. We set the mean of the null hypothesis equal to that of the front-end's hypothesis so that the small amount of gradient that remains produces a slight bias back towards the front-end's hypothesis. If subsequent observations re-affirm the front-end's hypothesis, it can still become active in the future. Unlike particle filter or MHT methods which must eventually truncate unlikely events, no information is lost.

A two-component mixture model in which both components have identical means but different variances can be viewed as a robustified cost function. In particular, parameters can be chosen so that a two-component max mixture closely approximates a corrupted Gaussian (Hartley and Zisserman, 2004) (see Figure 4.2).

To evaluate the performance of our approach, we added randomly-generated loop closures to two standard benchmark datasets: the 3500 node Manhattan set (Olson, 2008) and the Intel data set (Howard and Roy, 2003). These were processed in an online fashion, adding one pose at a time and potentially one or more loop closures (both correct and incorrect). This mimics real-world operation better than a batch approach, and is more challenging due to the fact that it is easier to become caught in a local minimum since fewer edges are available to guide the optimization towards the global optimum.

Manhattan Data Set

True Edges	False Edges	True Pos.	False Pos.	Avg. FP Err.	MSE (Our method)	MSE (Non-mixture)
2099	0	2099	0	NaN	0.6726	0.6726
2099	10	2099	0	NaN	0.6713	525.27
2099	100	2099	1	0.0208	0.6850	839.39
2099	200	2099	2	0.5001	0.6861	874.67
2099	500	2099	3	0.6477	0.6997	888.82
2099	1000	2099	10	0.7155	0.7195	893.98
2099	2000	2099	22	0.5947	0.7151	892.54
2099	3000	2099	36	0.5821	0.7316	896.01
2099	4000	2099	51	0.6155	0.8317	896.05

Intel Data Set

True Edges	False Edges	True Pos.	False Pos.	Avg. FP Err.	MSE (Our method)	MSE (Non-mixture)
14731	0	14731	0	NaN	$7.122\text{x}10^{-10}$	$1.55\text{x}10^{-9}$
14731	10	14731	0	NaN	$7.123\text{x}10^{-10}$	0.044
14731	100	14731	2	0.1769	$4.431\text{x}10^{-6}$	2.919
14731	200	14731	9	0.1960	$5.583\text{x}10^{-6}$	8.810
14731	500	14731	19	0.1676	$1.256\text{x}10^{-5}$	34.49
14731	1000	14731	29	0.1851	$5.840\text{x}10^{-5}$	71.86
14731	2000	14731	64	0.1937	$2.543\text{x}10^{-4}$	86.37
14731	3000	14731	103	0.1896	$3.307\text{x}10^{-4}$	91.04
14731	4000	14217	146	0.1699	0.014	95.36

Figure 4.3: Null-hypothesis robustness. We evaluate the robustness of our method and a standard Gaussian method to the presence of randomly-generated edges. As the number of randomly-generated edges increases, the mean squared error (MSE) of standard approaches rapidly degenerates; our proposed method produces good maps even when the number of randomly-generated edges is large in comparison to the number of true edges. Our approach does accept some randomly-generated edges (labeled "false positives" above), however the error of these accepted edges is comparable to that of the true positives. In each case, the initial state estimate is that from the open-loop odometry.

For a given number of randomly-generated edges, we compute the posterior map generated by our method and a standard non-mixture method, using a laser-odometry solution as the initial state estimate. The mean-squared error of this map is compared to the global optimal solution (Olson, 2011b), and listed in Figure 4.3.

Our proposed method achieves dramatically lower mean squared errors (MSE) than standard non-mixture versions. While the true positive rate is nearly perfect in both

We report MSE based on translational error, i.e. MSE_{xy} for 3dof and MSE_{xyz} for 6dof problems.

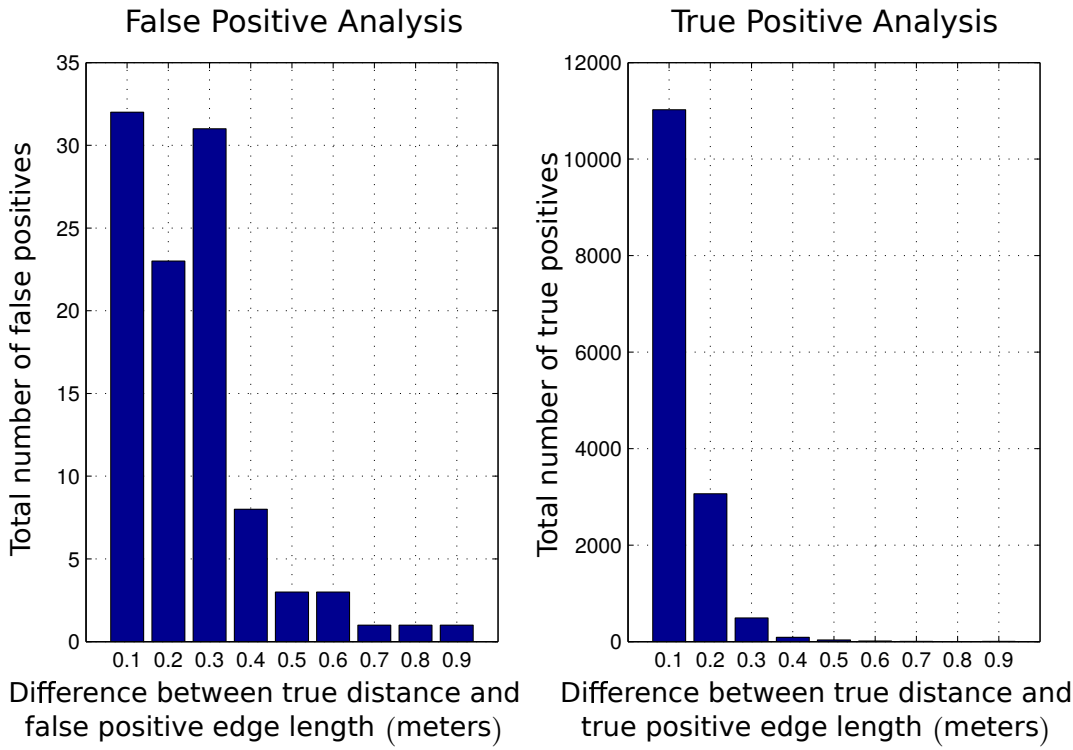


Figure 4.4: Error distribution for true and false positives. Our method accepts some randomly-generated "false positives", but an analysis of the error of those edges indicates that they (left) are only slightly worse than the error of true edges (right).

experiments, some randomly-generated edges (labeled false positives) *are* accepted by our system. However, since the false positives are randomly generated, some of them (by chance) are actually close to the truth. Such "accidentally correct" edges *should* be accepted by our algorithm.

We can evaluate the quality of the accepted edges by comparing the error distribution of the true positives and false positives (see Figure 4.4). As the histogram indicates, the error distribution is similar, though the error distribution for the false positives is slightly worse than for the true positives. Still, no extreme outliers (the type that cause divergence of the map) are accepted by our method.

4.4.1.1 Extension to 6DOF

While many important domains can be described in terms of planar motion (with three-dimensional factor potentials reflecting translation in x, translation in y, and rotation), there is increasing interest in 6 degree-of-freedom problems. Rotation is a major source

We favor generating "false positives" in a purely random way, even though it leads to "accidentally" correct edges. Any filtering operation to reject these low-error edges would introduce a free parameter (the error threshold) whose value could be tuned to favor the algorithm.

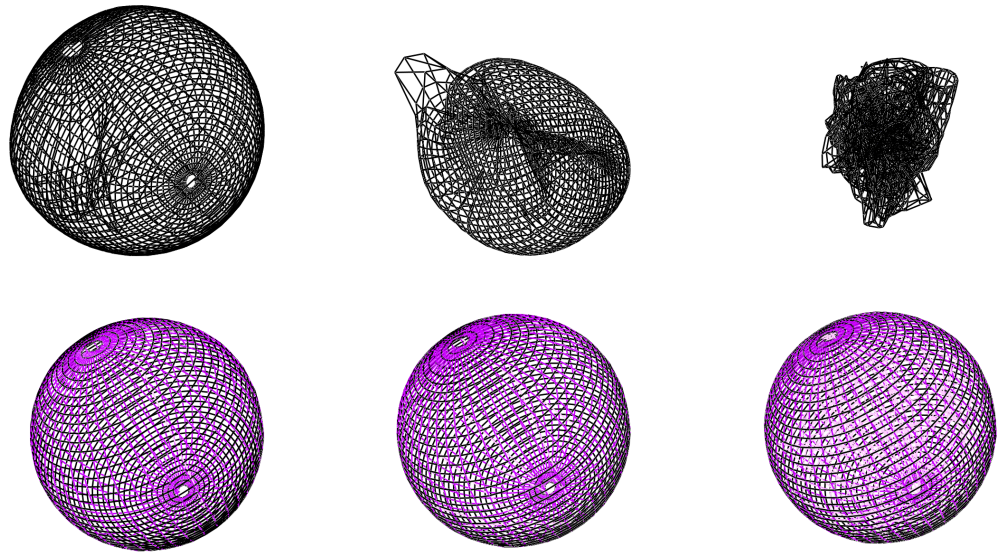


Figure 4.5: Recovering a map in the presence of outliers. We evaluated the robustness of our method by adding erroneous loop closure edges to the Sphere2500 dataset, a dataset with a full 6 degrees-of-freedom. The top row reflects the posterior of the map with a standard least square Cholesky solver with 1, 10, and 100 wrong edges. The bottom row shows the corresponding map for the same dataset using max mixtures method.

of non-linearity in SLAM problems, and full six degree-of-freedom problems can be particularly challenging.

To evaluate the performance of our method on a six degree-of-freedom problem, we used the benchmark Sphere2500 dataset (Grisetti et al., 2007b). This dataset does not contain incorrect loop closures, and so we added additional erroneous loop closures. In Figure 4.5, we show the results of a standard Cholesky solver and our max mixture approach applied to corrupted Sphere2500 dataset with an additional 1, 10, and 100 erroneous edges. As in previous examples, the maps produced by a standard method quickly deteriorate. In contrast, the proposed method produces posterior maps that are essentially unaffected by the errors. In this experiment, each loop closure edge in the graph (both correct and false) was modeled as a two-component max mixture in which the second component had a large variance (10^7 times larger than the hypothesis itself) and a small weight (10^{-7}). The method is relatively insensitive to the particular values used: the critical factor is ensuring that, if the hypothesis is incorrect, the null hypothesis provides a higher probability explanation than the putative (incorrect) hypothesis, and that the null hypothesis is sufficiently weak so as to not distort the final solution. The impact of the relative strength of the null hypothesis on the basin of convergence is explored experimentally in Section 4.4.5.

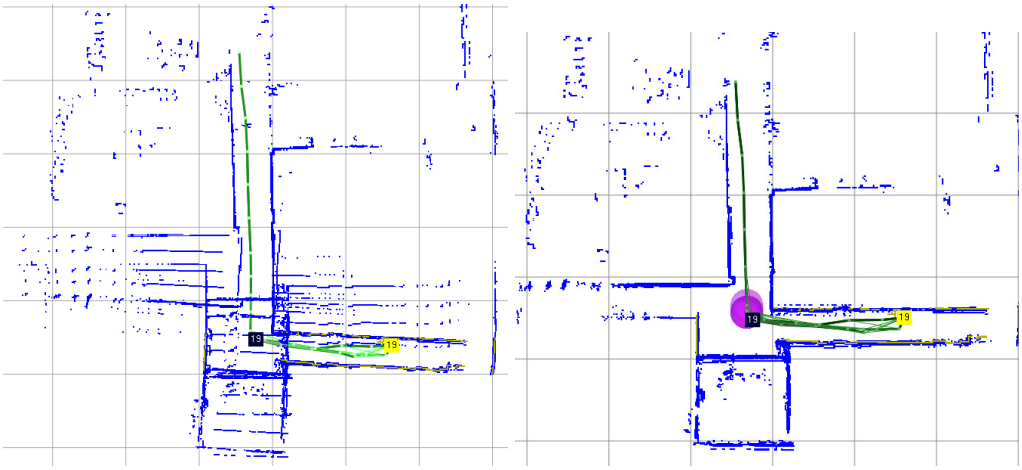


Figure 4.6: Slip or Grip Example. We evaluate the ability of our algorithm to recover a good map in the presence of catastrophically slipping wheels. In this case, the robot is obtaining loop closures using a conventional laser scanner front-end. These loop closures are of high quality, but the odometry edges still cause significant map distortions when using standard methods (left). When a small probability is added to account for slippage, our mixture approach recovers a much improved map (right).

4.4.2 Multi-Modal Distributions

In the previous sections, we demonstrated that our method can be used to encode null hypotheses, or equivalently, implement robustified cost functions—a capability similar to earlier work (Sünderhauf and Protzel, 2012). In that case, the probability distributions being approximated by each mixture have only a single maximum. Our use of a mixture model, however, also allows multi-modal distributions to be encoded. The ability to directly represent multi-modal distributions is a feature of our approach.

4.4.2.1 Slip or Grip problem

One of the original motivating problems of this work was dealing with the "slip or grip" problem: the case where a robot's wheels occasionally slip catastrophically, resulting in near zero motion. With a typical odometry noise model of 10-20%, such an outcome would wreak havoc on the posterior map.

Our approach to the "slip or grip" problem is to use a two-component mixture model: one component (with a large weight) corresponds to the usual 15% noise model, while the second component (with a low weight) is centered around zero. No changes to our optimization algorithm are required to handle such a case. However, since the distribution now has multiple local maxima, it poses a greater challenge in terms of robustness.

Of course, without some independent source of information that contradicts the odometry data, there is no way to determine that the wheels were slipping. To provide this

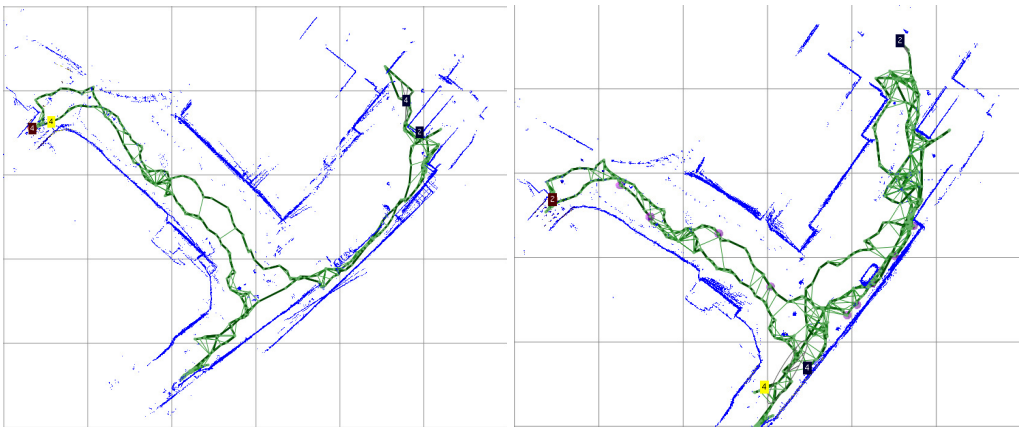


Figure 4.7: Online results using odometry mixture model. The left figure shows a map of a 30m $\times$ 25m area in which our multi-robot urban mapping team produced a poor map due to wheel slippage and the ensuing inability to find loop-closures. With our odometry mixture model (right), the wheel slippage is (implicitly) detected, and we find additional loop closures. The result is a significantly improved map.

independent information, we used a state-of-the-art scan matching system (Olson, 2009a) to generate loop closures. We manually induced wheel slippage by pulling on the robot. Despite the good loop closures, standard methods are unable to recover the correct map. In contrast, our method determines that "slip" mode is more likely than the "grip" mode, and recovers the correct map (see Figure 4.6).

As part of an earlier multi-robot mapping work (Ranganathan et al., 2010; Olson et al., 2012), a team of 14 robots were employed to explore a large urban environment. Wheel slippage contributed to a poor map in two ways: 1) the erroneous factor potentials themselves, and 2) the inability to identify good loop closures due to a low quality motion estimate. By using a better odometry model, our online system produced a significantly improved map (see Figure 4.7).

4.4.2.2 Simplifying the Front End with "one-of-k" Formulation

In current approaches, front-end systems are typically responsible for validating loop closures prior to adding them to the SLAM graph network. However, if the back-end can recover from errors, is it possible to omit the filtering entirely?

In certain cases, our inference method can eliminate the need for loop validation by the front-end. This is desirable from a conceptual standpoint: in principle, a better map should result from handling loop closing and mapping from within an integrated Bayesian framework. The conventional decoupling of mapping into a front-end and back-end, while practical, prevents a fully Bayesian solution.

We evaluated this possibility using the Intel data set. At every pose, a laser scan

matcher attempts a match to *every* previous pose. The top k matches (as measured by overlap of the two scans) are formed into a mixture containing $k + 1$ components. (The extra component remains a null hypothesis to handle the case where all k matches are incorrect.) To push the experiment as far as possible, no position information was used to prune the set of k matches. Larger values of k provide robustness against perceptual aliasing, since it increases the likelihood that the correct match is present somewhere within the set of k components.

An example of one mixture with $k = 4$ putative matches is shown in Figure 4.8. The

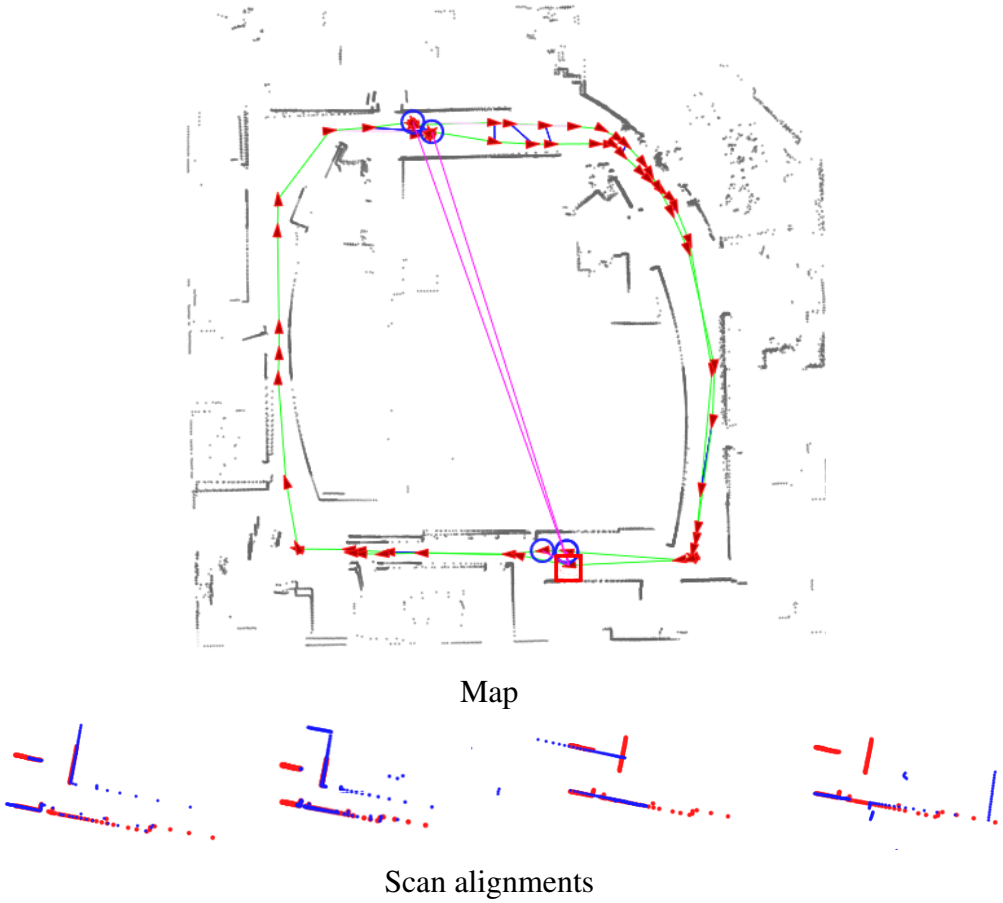


Figure 4.8: Data-association as a mixture. Given a query pose (red square at bottom of map), we perform a brute-force scan matching operation to all previous poses. The best 4 scan match results, based on overlap, are added to a max-mixture model that also includes a null hypothesis. The position of the best matches are shown as blue circles, and the corresponding scan matches shown at the bottom. The similarity in appearance between the blue poses represents a significant degree of perceptual aliasing. The scan matcher finds two correct matches and two incorrect matches. The two correct matches are the two blue circles at the bottom of the map and the first two scan alignments.

Figure 4.9: Intel without front-end loop validation. Our system can identify correct loop closures and compute a posterior map from within a single integrated Bayesian framework (right); the typical front-end loop validation has been replaced with a $k+1$ mixture component containing the k best laser scan matches (based purely on overlap) plus a null hypothesis. In this experiment, we used $k = 5$. For reference, the open-loop trajectory of the robot is given on the left.

weight of the components is set in proportion to the score of the scan matcher.

Running our system in an online fashion, we obtain the final map illustrated in Figure 4.9. Online operation is more difficult than batch operation, since there is less information available early on to correct erroneous edges. Our system recovers a consistent global map despite the lack of any front-end loop validation.

The quality of the open-loop trajectory estimate plays an important role in determining whether the initial state estimate is within the basin of convergence. In this case, our open-loop trajectory estimate is fairly good, and our method is able to infer the correct mode for each mixture despite the lack of any front-end loop validation.

The robustness of our method is amplified by better front-end systems: with better quality loop closures, the basin of convergence is enlarged, allowing good maps to be computed even when the open-loop trajectory is poor.

4.4.2.3 Performance Impact of Uncertainty Modeling

In the previous section, uncertain data associations were modeled as "one-of-k" mixtures, in which multiple candidate loop closures were grouped together in a single edge. Alternatively, each candidate loop closure could be encoded as a two-component mixture in a "null-hypothesis" style mixture; this approach is well-suited to the case where little is known about alternatives to a putative loop closure, while still allowing for the possibility that it is incorrect. (It is also possible that the mixture components have no obvious

semantic meaning: the mixture model could simply be approximating a more complex distribution. For example, a max mixture could be fit to an empirically derived cost function from a correlation-based scan matcher (Olson, 2009a)).

In this section, we explore the performance impact of "one-of-k" mixtures versus "null-hypothesis" mixtures. Consider a "one-of-k" mixture consisting of three candidate loop closures plus a null hypothesis: $\{L_1, L_2, L_3, \text{null}\}$. This can be transformed into three "null-hypothesis" mixtures: $\{L_1, \text{null}\}$, $\{L_2, \text{null}\}$, and $\{L_3, \text{null}\}$. These two formulations are not exactly equivalent: the "one-of-k" encodes mutual-exclusion between the hypotheses, whereas the k separate "null-hypotheses" would permit solutions in which more than one of the loop closures was accepted. In many practical situations, however, the semantic difference is relatively minor. In this section, we show that the performance impact of this choice can be dramatic.

In Table 4.1, we show results from an experiment in which both formulations were used. We consider the case where loop hypotheses are generated in pairs and in triples; this leads to "one-of-k" mixtures with three and four components respectively once a null hypothesis is added. For the "one-of-k" formulation, the null hypotheses has a mean chosen randomly from one of the k constraints and a large variance roughly the size of the whole map. An alternative graph, constructed from "null-hypothesis" mixtures is constructed from the same sets of loop closure hypotheses; naturally, each of these has two components.

An obvious difference between the two formulations is the number of edges in the graph: the "null-hypothesis" approach creates many more edges. That alone can be expected to increase computational time versus a "one-of-k" encoding. However, a more critical scaling issue becomes apparent: the "null-hypothesis" formulation leads to dramatically higher fill-in due to the fact that more nodes are connected to factor potentials. In contrast, a "one-of-k" edge does not contribute the same fill-in, since only one of the components in the mixture has any effect during a single Cholesky iteration. In other words, the $\max$ operator in the max mixture formulation effectively severs edges corresponding to sub-dominant mixture components, improving the sparsity of the information matrix.

The difference in fill-in leads to significant increases in runtime: on the Manhattan-3500 dataset with groups of three candidate hypotheses, moving from a "one-of-k" to a "null-hypothesis" formulation causes an increase in non-zero entries from 0.17% to 4.3%, with a corresponding increase in computation time from 0.13 s to 1.2 s.

Table 4.1 also reports runtimes for the Switchable Constraints approach (Sünderhauf and Protzel, 2012), which adds an additional "switching" variable for every edge. In this way, it is semantically comparable to the "null-hypothesis" approach, though the formulation is somewhat different. The runtime of the switchable constraints approach, 1.5s, is somewhat worse than "null-hypothesis" approach and much worse than the

Dataset		Switchable Constraints	bi-modal Max-Mixture	k-modal Max-Mixture
Manhattan	iter time (s)	0.90 s	0.74 s	0.13 s
with $k = 2$	fill-in (%)	1.50 %	2.89 %	0.17 %
outliers $= 2099$	#loop edges	4198	4198	2099
	#components	-	2	3
Manhattan	iter time (s)	1.5 s	1.2 s	0.13 s
with $k = 3$	fill-in (%)	1.70 %	4.30 %	0.17 %
outliers$= 4198$	#loop edges	6277	6277	2099
	#components	-	2	4

Table 4.1: Runtime comparison between switchable constraints, "null-hypothesis", "one-of-k" formulations. Groups of related hypotheses were generated and either grouped as a single set of mutually-exclusive edges (one-of-k), individually associated with a null hypothesis, or individually associated with a switching variable (Sünderhauf and Protzel, 2012). Using the one-of-k formulation reduces the effective connectivity in the graph, reducing fill-in, and resulting in faster computation time.

"one-of-k" approach. (Note, for this comparison, all methods were implemented in the g2o (Kümmerle et al., 2011) framework using *CHOLMOD* with a *COLAMD* variable ordering.)

These results suggest that, when semantically reasonable to do so, it is preferable to use "one-of-k" mixtures rather than either "null-hypothesis" mixtures or switchable constraints.

4.4.3 Robustness

We have identified two basic factors that have a significant influence on the success of our method: the number of incorrect loop closures and the node degree of the graph. The node degree is an important factor because it determines how over-determined the system is: it determines the degree to which correct edges can "overpower" incorrect edges.

To illustrate the relationship between these factors and the resulting quality of the map (measured in terms of mean squared error), we considered a range of loop-closing error rates (ranging from 0% to 100%) for graphs with an average node degree of 4, 8, and 12. Note that an error rate of 80% means that incorrect loop closures outnumber correct loop closures by a ratio of 4:1. In each case, the vehicle's noisy odometry is also provided. For each condition, we evaluate the performance of our method on 100,000 randomly-generated Manhattan-world graphs (see Figure 4.10). Our method produces good maps even when the error rate is high, and the performance improves further with increasing node degree. In contrast, a standard non-mixture approach diverges almost

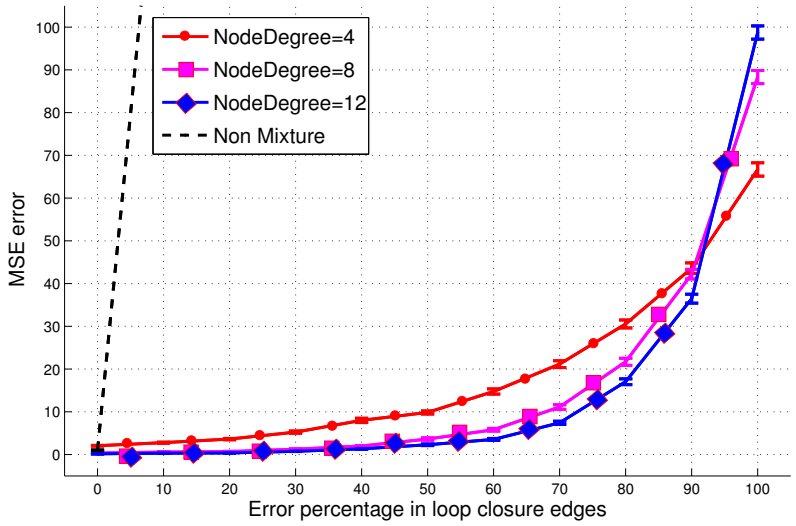


Figure 4.10: Effect of error rate and node degree on robustness. We evaluate the quality of posterior maps (in terms of mean squared error) as a function of the percentage of bad edges and the node degree of the graph. Each data point represents the average of 3,000 random trials; the standard error is also plotted showing that the results are significant. The quality of the posterior graph is robust to even high levels of error, and is improved further by problems with a high node degree. Our methods, regardless of settings, dramatically out-perform non-mixture methods (black dotted line).

immediately.

4.4.4 Runtime Performance

The performance of our method is comparable to existing state-of-the-art sparse factorization methods (see Figure 4.11). It takes additional time to identify the maximum likelihood mode for each mixture, but this cost is minor in comparison to the cost of solving the resulting linear system.

4.4.5 Basin of Convergence

A key issue in non-linear optimization methods is whether the globally optimal solution will be found, or whether the optimization process will get stuck in a local minimum. This is a function of the initial solution as well as the parameters of the problem. In this section, we describe the effects of these parameters on the robustness of our method, as well as an experiment to empirically evaluate the magnitude of these effects.

The effect of the initial solution is relatively straight-forward: some initial solutions provide a better path for the optimization system to follow. In high-noise cases, some initial solutions may be far from the desired solution and in a different basin of convergence,

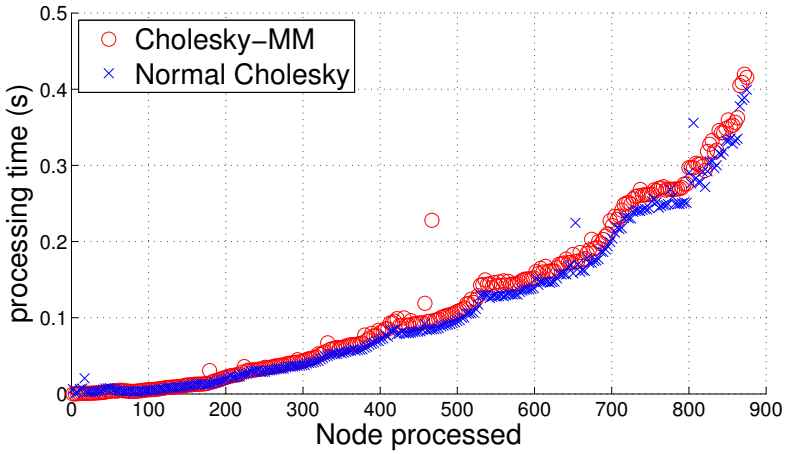


Figure 4.11: Runtime performance. Using the Intel dataset, we plot the time required to compute a posterior map after every pose, using a batch solver. Our Intel dataset contains 875 nodes and 15605 edges, and each edge is modeled as a two-component max-mixture with a null hypothesis. The additional cost of handling mixtures is quite small in comparison to the total computation time. Run times were computed using the Java-based april.graph library, which is slower than g2o, but exhibits the same scaling behavior as other methods.

leading to a poor solution. We consider two initializations: 1) open-loop odometry (well-suited to online optimization) and 2) an approach which initializes each node relative to its oldest neighbor (a heuristic used by TORO (Grisetti et al., 2007b) and implemented within g2o (Kümmerle et al., 2011), most useful in batch processing).

The type of errors that occur also affect the robustness of the method. In this analysis, we consider four types of erroneous loop closures based on the Vertigo package (Sünderhauf, 2012): 1) random errors, 2) locally clustered (but not mutually consistent) errors, 3) randomly grouped errors in which the groups are mutually consistent (group size = 10), and 4) locally grouped errors (group size = 10).

The relative "strength" of the null hypothesis in comparison to the putative hypothesis also has an effect on the optimization. This strength can be described in terms of two parameters. The weight parameter (w) is the mixing parameter associated with the null-hypothesis component. Larger values of w increase the likelihood of the null hypothesis and cause the system to reject more of the putative hypotheses. If w is too small, and the system will accept incorrect hypotheses.

The second parameter, s, is the scale factor used in generating the information matrix associated with the null hypothesis from the information matrix associated with the putative hypothesis. When $s = 1$, both mixture components are identical and thus no robustness from the method can be expected. Smaller values (closer to zero) of s yield null hypotheses that are less certain than the putative hypothesis. This is equivalent to increasing its covariance, which pushes more probability mass away from the mean.

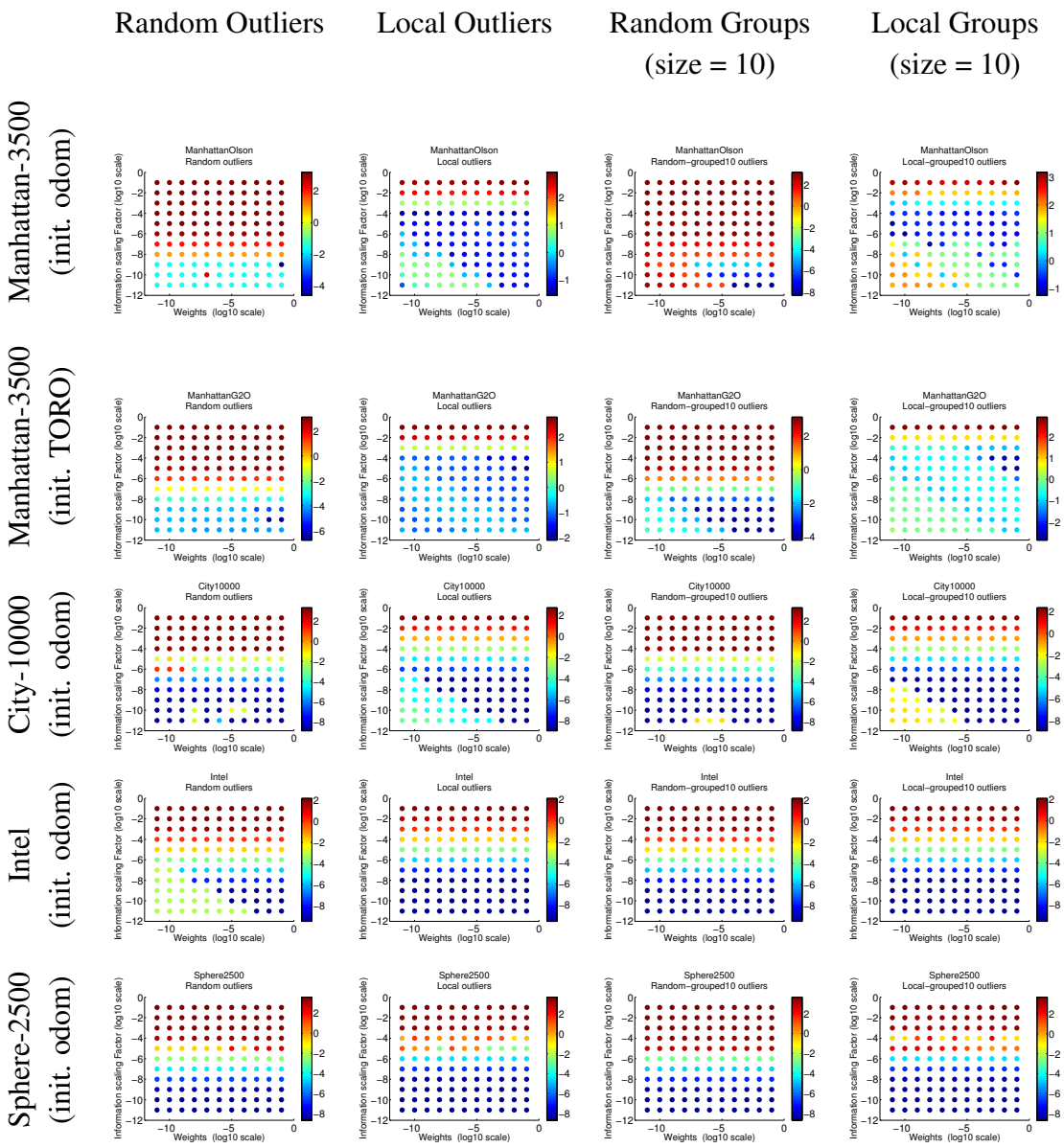


Figure 4.12: Robustness of max-mixtures evaluated over a range of parameters. We consider five different data sets (rows) and four data association error generation methods (columns). The Manhattan3500 dataset is evaluated with both the odometry as well as the spanning tree initialization. The simulated data-association errors are generated using four strategies, namely "random", "local", "random grouped" and "local grouped". A parameter sweep over w and s is illustrated within each grid cell. Colors correspond to the log of the mean squared error; maps less than -0.1 are relatively good and maps less than -1 are excellent. These plots show that the basin of convergence for the max mixture method is quite large. A total 1000 outliers of each error type was added for each dataset. For the grouped errors it resulted in 100 groups, each with 10 mutually consistent outliers.

This not only allows the null-hypothesis to produce a higher probability explanation of observed data, but also results in less curvature in the cost function. As s gets smaller, the cost function becomes increasingly flat, decreasing any influence of the mixture on the posterior solution.

We explore the effect of these parameters in Figure 4.12. Columns of the table represent the four different outlier generation strategies and rows represent different data sets and initializations (not all combinations are presented for space reasons). Within each cell, the parameters w and s are swept resulting in a two-dimensional grid of map scores. At each data point, a graph was constructed and solved using the max mixture method and the log of the mean squared error (evaluated with respect to ground truth) is plotted according to color.

The data in Figure 4.12 shows graphically how to tune the free parameters w and s to maximize the quality of the resulting map. Across virtually all of the experiments, the best performance is generally found in the lower right corner of the parameter sweep. This area corresponds to null-hypotheses with relatively large weights and low-information (equivalently large covariances). However, it is also evident that the region of good performance (which we subjectively appraise to be mean squared errors less than about about -1) is quite large in almost all cases. From these results, we conclude that the method is robust across many orders of magnitude of w and s, and that in general, w should be made relatively close to 1 and s relatively close to zero.

4.5 Summary

We have described a method for performing inference on networks of mixtures, describing an application of our method to robot mapping. Our method consists of a novel mixture model based on a "max" operator that makes the computation of the Jacobian and residual fast, and we show how existing sparse factorization methods can be extended to incorporate these mixtures. We believe that such an approach is necessary for long-lived systems, since any system that relies on a *zero* error rate will fail.

We demonstrate how the mixture model allows null hypotheses and robust cost functions such as corrupted Gaussian, to be incorporated into a maximum likelihood inference system. We show that our system is robust to a large error rates far in excess of what can be achieved with existing front-end loop validation methods. We further demonstrate a multi-modal formulation, addressing the wheel slippage problem and showing that our system can make loop validation unnecessary in some cases.

Our algorithm cannot guarantee convergence to the global optimum, but we characterized the basin of convergence, demonstrating the relationship between error rate, node degree, and convergence to a good solution.

Finally, we demonstrate that the runtime performance of our algorithm is similar to that

of existing state-of-the-art maximum likelihood systems. In comparison to other robust formulations, including those based on switching constraints, the ability of our method to encode "one-of-k" mixtures provides a significant performance advantage. Further, while we have explored the case of batch solvers, our method can be equally-well adapted to incremental systems (Kaess et al., 2008, 2012). An open source implementation can be downloaded from (Agarwal et al., 2012).

Chapter 5

Dynamic Covariance Scaling

Developing the perfect SLAM front-end that produces graphs which are free of outliers is generally impossible due to perceptual aliasing. Therefore, optimization back-ends need to be able to deal with outliers resulting from an imperfect front-end. In the previous chapter we introduced one such method, namely max-mixtures, which encodes outliers with a null-hypothesis and then selects the hypothesis with the highest probability during each iteration of the optimization. In this chapter, we introduce dynamic covariance scaling (DCS), a novel approach for effective optimization of factor graphs under the presence of outliers. Unlike max-mixtures, DCS does not require a null-hypothesis or a hypothesis selector. The key idea is to use a robust function that generalizes classical gating and dynamically rejects outliers without compromising convergence speed.

5.1 Introduction

Most SLAM approaches assume that the constraints are affected by noise but no outliers (false positives) are present, i.e., there are no constraints that identify actually different places as being the same one. This corresponds to the assumption of having a perfect SLAM front-end. In traditional methods, a single error in the data association often leads to inconsistent maps. Generating outlier-free graphs in the front-end, however, is challenging, especially in environments showing self-similar structures (Olson, 2009b; Cummins and Newman, 2008). Thus, having the capability to identify and to reject wrong data associations is essential for robustly building large scale maps without user intervention.

In the previous chapter we outlined max-mixtures, a robust back-end that mitigates data association errors and allows to optimize over multi-modal hypothesis. Max-mixtures handle data-association errors in the back-end by adding a null hypothesis for each constraint. At every optimization step, the max operator acts a selector and decides whether or not the null-hypothesis should be used in the optimization process. The contribution of this chapter, is an approach which avoids this decision by smoothly and dynamically

scaling the covariance. This allows Dynamic Covariance Scaling (DCS) to be robust to a huge number of outliers while at the same time being robust to poor initialization. Additionally, it is computationally efficient, avoiding an increase in execution time.

Compared to other state-of-the-art robust back-ends such as Switchable Constraints (SC) (Sünderhauf and Protzel, 2012) and RRR (Latif et al., 2012b), our strategy has a reduced computational overhead and typically results in better convergence. It uses a robust function that generalizes classical gating by dynamically scaling the covariance. The proposed function shares common grounds with existing robust M-estimators. Furthermore, it can be integrated easily into existing graph-based SLAM systems and does not require significant changes in the code.

In the remainder of this chapter we first provide a brief introduction to switchable constraints (SC) (Sünderhauf and Protzel, 2012) as DCS was formulated after a careful analysis of SC. In Section 5.3, we derive dynamic covariance scaling from SC. The relation to M-estimators is explained in Section 5.4 and we show how DCS is a special instance of the generalized Geman-McClure M-estimator. Finally, in Section 5.5 we provide experimental evaluation on both simulated and real world datasets.

5.2 Analysis of the Switchable Constraint's Error Function

Following from Equation (2.3), the graph-based approach to solve the SLAM problem may be further expanded such that:

$$\begin{aligned} \mathbf{x}^* &= \operatorname*{argmin}_{\mathbf{x}} \sum_i \| \, \hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_{i+1}) - \mathbf{z}_{i,i+1} \, \|^2_{\mathbf{\Sigma}_i} \\ &\quad + \sum_{ij} \underbrace{\| \, (\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}) \, \|^2_{\mathbf{\Lambda}_{ij}}}_{\chi^2_{l_{ij}}}, \end{aligned} \tag{5.1}$$

where $\mathbf{x}_i$ represents a pose of the robot, $\mathbf{z}_{ij}$ is the observed measurement between two poses obtained by odometry or front-ends by processing sensor data and $\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j)$ is the predicted measurement between $\mathbf{x}_i$ and $\mathbf{x}_j$. The covariances of the odometry and sensor measurements are $\mathbf{\Sigma}_i$ and $\mathbf{\Lambda}_{ij}$ respectively. Thus, whereas the first term in Equation (5.1) represents the incremental constraints that result from odometry constraints, the second term refers to loop closing constraints.

Sünderhauf and Protzel (2012) published an effective method for optimizing graphs in the presence of outliers. Their basic idea is to introduce switching variables $s_{ij} \in [0, 1]$

that can disable potential outlier constraints. They minimize:

$$\begin{aligned}\mathbf{x}^*, \mathbf{s}^* &= \underset{\mathbf{x},\mathbf{s}}{\operatorname{argmin}} \sum_i \| \hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_{i+1}) - \mathbf{z}_{i,i+1} \|^2_{\mathbf{\Sigma}_i} \\ &\quad + \sum_{ij} \| \Psi(s_{ij})(\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}) \|^2_{\mathbf{\Lambda}_{ij}} \\ &\quad + \sum_{ij} \| 1 - s_{ij} \|^2_{\Xi_{ij}} \end{aligned} \tag{5.2}$$

which can be interpreted as three sums over different χ^2 errors, i.e., the one of the incremental constraints, the loop closing constraints and the switch priors. Here, $\Psi(s_{ij}) \in [0, 1]$ is a scaling function that determines the weight of a constraint given s_{ij} and a switching prior Ξ_{ij}. The function $\Psi(\cdot)$ can be interpreted as a scaling factor in the information matrix associated to a constraint, since

$$\begin{aligned} &\| \Psi(s_{ij})(\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}) \|^2_{\mathbf{\Lambda}_{ij}} \\ &= \Psi(s_{ij})(\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij})^\top \mathbf{\Lambda}_{ij}^{-1} \Psi(s_{ij})(\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}) \qquad (5.3) \\ &= (\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij})^\top \mathbf{\Lambda}_{ij}^{-1} \Psi(s_{ij})^2 (\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}) \qquad (5.4) \\ &= \| \hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij} \|^2_{\Psi(s_{ij})^{-2}\mathbf{\Lambda}_{ij}}. \qquad (5.5) \end{aligned}$$

Sünderhauf and Protzel (2012) suggest to set $\Psi(s_{ij}) = s_{ij}$ within the interval $[0, 1]$ to obtain the best results.

5.3 Dynamically Scaled Covariance Kernel

Sünderhauf and Protzel (2012) propose a joint optimization of $\mathbf{x}$ and $\mathbf{s}$ that results in an additional switch variable s_{ij} for each loop closing constraint. The addition of the switch variables increase computation cost of each iteration and in this case increases the problem complexity, thus potentially decreasing the convergence speed. In the remainder of this chapter, we will show how to circumvent the use of the switching variables inside the optimizer with DCS. This leads to a significantly faster convergence, meanwhile obtaining comparable robustness. We next introduce the main contribution of this work, namely a technique that provides an analytical solution to the scaling factors. This does not only simplify the optimization problem by greatly reducing the number of variables, but it also creates an easier to optimize cost surface as shown empirically in Section 5.5.

We first analyze the behavior of the error function defined in Equation (5.2). In particular, we investigate how the switching variables influence the χ^2 error at the local minima. Without the loss of generality, let us consider the edge between two nodes k and l. We can split the error function into two blocks, the first one considers all edges except

those between k and l and the second block with all edges between nodes k and l:

$$\begin{aligned}
\mathbf{x}^*, \mathbf{s}^* &= \underset{\mathbf{x},\mathbf{s}}{\operatorname{argmin}} \sum_i \| \hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_{i+1}) - \mathbf{z}_{i,i+1} \|^2_{\mathbf{\Sigma}_i} \\
&\quad + \sum_{ij \neq kl} \| s_{ij}(\hat{\mathbf{z}}(\mathbf{x}_i, \mathbf{x}_j) - \mathbf{z}_{ij}) \|^2_{\mathbf{\Lambda}_{ij}} \\
&\quad + \sum_{ij \neq kl} \| 1 - s_{ij} \|^2_{\Xi_{ij}} \\
&\quad + \| s_{kl}(\hat{\mathbf{z}}(\mathbf{x}_k, \mathbf{x}_l) - \mathbf{z}_{kl}) \|^2_{\mathbf{\Lambda}_{kl}} + \| 1 - s_{kl} \|^2_{\Xi_{kl}} \\
&= \underset{\mathbf{x},\mathbf{s}}{\operatorname{argmin}} \underbrace{\mathbf{G}(\mathbf{x}, s_{ij \neq kl}) + \underbrace{s_{kl}^2 \chi_{l_{kl}}^2 + (1 - s_{kl})^2 \Phi}_{\mathbf{H}(s_{kl}, \chi_l^2)}}_{b}
\end{aligned} \tag{5.6}$$

with $\Phi := \Xi_{ij}^{-1}$. In Equation (5.6), the term $\mathbf{G}(\cdot)$ represents the error for all edges, including odometry, except of the edge between nodes k and l.

A necessary condition for optimality is that the partial derivatives with respect to *all* entries in $\mathbf{x}$ and $\mathbf{s}$ are zero. Hence the derivative with respect to s_{kl} must be 0. Taking the partial derivative of Equation (5.6) with respect to s_{kl}, we obtain for a generic constraint (indices are omitted for notational simplicity):

$$\nabla b = \begin{bmatrix} \vdots \\ \frac{\partial b}{\partial s} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ 2s\chi_l^2 - 2(1-s)\Phi \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ 0 \\ \vdots \end{bmatrix} \tag{5.7}$$

Solving for s, in terms of Φ and χ_l^2, leads to

$$\begin{aligned}
2s\chi_l^2 - 2(1-s)\Phi &= 0 \\
s\left(\chi_l^2 + \Phi\right) &= \Phi \\
s &= \frac{\Phi}{\chi_l^2 + \Phi}.
\end{aligned} \tag{5.8}$$

Substituting s from Equation (5.8) in $\mathbf{H}(\cdot)$, we obtain:

$$\bar{\mathbf{H}} = \frac{\Phi^2 \chi_l^2}{\left(\chi_l^2 + \Phi\right)^2} + \Phi - \frac{2\Phi^2}{\chi_l^2 + \Phi} + \frac{\Phi^3}{\left(\chi_l^2 + \Phi\right)^2} \tag{5.9}$$

The function $\bar{\mathbf{H}}(\cdot)$ represents the projection of $\mathbf{H}(\cdot)$ on the manifold where the gradient with respect to $\mathbf{s}$ is 0. Finding the maxima of this function is equivalent to obtaining an upper bound on χ_l^2 for all possible solutions computed by the optimizer. Let us analyze the derivative of Equation (5.9):

$$\frac{d\bar{\mathbf{H}}}{d\chi_l} = \frac{2\chi_l\Phi^2}{\left(\Phi+\chi_l^2\right)^2} \tag{5.10}$$

As can be seen from Equation (5.10), the derivative is 0 when $\chi_l = 0$. Thus, we evaluate the function $\bar{\mathbf{H}}$ at $\pm\infty$ and 0:

$$\lim_{\chi_l\to\pm\infty} \bar{\mathbf{H}} = 0+\Phi-0+0=\Phi \tag{5.11}$$

$$\chi_l = 0 \Rightarrow \bar{\mathbf{H}} = 0+\Phi-2\Phi+\Phi=0 \tag{5.12}$$

Thus, $\mathbf{H}(\cdot) \leq \Phi$ for every solution computed by the optimizer. This can be generalized to all switch variables and hence $\mathbf{H}(\cdot) \leq \Phi$ for every constraint. By using Φ as an upper bound for all robust edges, we obtain:

$$\begin{aligned}
(1-s)^2\Phi + s^2\chi_l^2 &\leq \Phi \\
\Phi + s^2\Phi - 2s\Phi + s^2\chi_l^2 &\leq \Phi \\
s^2(\Phi+\chi_l^2) + s(-2\Phi) + (\Phi-\Phi) &\leq 0 \\
s(s(\Phi+\chi_l^2) - 2\Phi) \leq 0.&
\end{aligned} \tag{5.13}$$

The solution to this inequality is given as

$$0 \leq s \leq \frac{2\Phi}{\Phi+\chi_l^2}. \tag{5.14}$$

In theory, one could choose any value for s within that interval. We choose the value that minimize $\mathbf{H}(\cdot)$ within that interval, while not exceeding 1. It is given by

$$s = \min\left(1, \frac{2\Phi}{\Phi+\chi_l^2}\right). \tag{5.15}$$

In sum, we have a closed-form solution for computing the scaling factor s individually for each loop closing constraint. It depends on χ_l^2, which is the original error term for each loop closure constraint. This formulation dynamically scales the information matrix of each non-incremental edge by s^2 given in Equation (5.15) and thus by a factor that considers the magnitude of the current error. A gradient always exists in the direction of an edge and gradually increases in the presence of more mutually consistent factors. The gradient is scaled according to s, which depends on the current error χ_l^2 and Φ.

It should be noted that this result can be integrated in basically any graph-based SLAM system with minimal efforts. Once s_{ij} is computed based on the error of the corresponding

constraint, the residuals can be scaled with s_{ij} or alternatively the information matrix can be scaled with s_{ij}^2, depending on implementation of the SLAM back-end.

The analytical solution of s^2 derived above shows that DCS is related to iteratively reweighted least squares (IRLS) and robust M-estimation. In the next section we outline how DCS is related to IRLS.

5.4 Relation to Iteratively Reweighted Least Squares

In IRLS, the influence of outliers is reduced by replacing the squared error in Equation (2.3) with another function (Zhang, 1997):

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \sum_{ij} \rho\Big(\mathbf{e}_{ij}(\mathbf{x})^T \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij}(\mathbf{x})\Big). \tag{5.16}$$

If the function ρ is symmetric and positive-definite with a minimum at zero, the solution to Equation (5.16) can be obtained by solving a sequence of problems of the form:

$$\mathbf{x}_k^* = \operatorname*{argmin}_{\mathbf{x}} \sum_{i} w\left(r_i\left(\mathbf{x}^{(k-1)}\right)\right) \hat{r}_i^2(\mathbf{x}), \tag{5.17}$$

where $r_i(\mathbf{x})$ denotes the i-th residual with respect to the optimization vector $\mathbf{x}$, $\mathbf{x}^{(k-1)}$ is the optimization vector computed at iteration $k-1$, $\hat{r}_i(\mathbf{x})$ denotes the linear approximation of $r_i(\mathbf{x})$ around $\mathbf{x}^k$ and $w(\cdot)$ is a weighting function associated to $\rho(\cdot)$. Equation (5.17) states that the weighting function using the residual error of the previous iteration can be used to update the current fixed point iteration. This is similar to what we do in DCS. We calculate the value of s^2 based on the residual of the previous iteration and scale the covariance to modify the residuals iteratively.

According to Zhang (1997), every IRLS problem with weighting function $w(x)$ is equivalent to an M-estimator with kernel function $\rho(x)$, and the two are related by the functional equation:

$$w(x) = \frac{1}{x}\frac{\mathrm{d}}{\mathrm{d}x}\rho(x). \tag{5.18}$$

In fact, the Geman-McClure M-estimator (Zhang, 1997; Geman and McClure, 1987) has a similar weighting function to DCS. In the next sections, we first outline DCS as an IRLS and then explain the relationship of the Geman-McClure M-estimator to DCS.

5.4.1 Dynamic Covariance Scaling as an Iteratively Reweighted Least Squares

DCS can be formulated as an IRLS problem, i.e. DCS aims to find a fixed point of the recurrence in Equation (5.17). The weighting function for DCS is

$$\begin{aligned} w(x) &= \left[\min\left\{1, \frac{2\Phi}{x^2+\Phi}\right\}\right]^2 \\ &= \min\left\{1, \frac{4\Phi^2}{\left(x^2+\Phi\right)^2}\right\}. \end{aligned} \tag{5.19}$$

The M-estimator equivalent to DCS can be obtained from Equation (5.18) by integration, specifically:

$$\begin{aligned} \rho(x) &= \int_0^x z\, w(z)\, \mathrm{d}z \\ &= \begin{cases} \displaystyle\int_0^x z\, \mathrm{d}z & \text{if } x^2 \leq \Phi \\ \displaystyle\int_0^{\sqrt{\Phi}} z\, \mathrm{d}z + \int_{\sqrt{\Phi}}^x \frac{4\Phi^2 z}{\left(z^2+\Phi\right)^2}\, \mathrm{d}z & \text{if } x^2 > \Phi \end{cases} \\ &= \begin{cases} \dfrac{x^2}{2} & \text{if } x^2 \leq \Phi \\ \dfrac{\Phi\left(3x^2-\Phi\right)}{2\left(x^2+\Phi\right)} & \text{if } x^2 > \Phi \end{cases} \end{aligned} \tag{5.20}$$

Equation (5.20) provides the cost function for DCS derived from the weighting function. It means that DCS behaves like a standard least-squares problem when the error is less than the tuning parameter Φ.

5.4.2 Geman-McClure M-estimator

Despite having been popularized as a parameterless robust kernel in (Zhang, 1997), the Geman-McClure kernel in the original notation introduced by Geman and McClure (1987) is the following function:

$$\phi(x) = -\frac{1}{1+\left(\frac{x}{\delta}\right)^2}, \tag{5.21}$$

which is further multiplied by a scalar β; effectively obtaining a two-parameter kernel with vertical and horizontal scale.

The original kernel defined in Equation (5.21) is strictly negative, while a positive definite function is required for robust non-linear least squares. This can be achieved without changing the overall minimum by summing the constant β to the kernel, hence obtaining the kernel:

$$\begin{aligned}\rho(x) &= \beta\phi(x) + \beta \\ &= \beta\frac{\left(\frac{x}{\delta}\right)^2}{1\left(\frac{x}{\delta}\right)^2} \\ &= \frac{\beta x^2}{\delta^2 + x^2}\end{aligned} \tag{5.22}$$

In SLAM problems we often formulate the optimization problems by mixing different kernel functions. For example, we may choose to not use M-estimators on odometric factors. Hence, a further regularity constraint is that $\rho(x) \sim \frac{x^2}{2}$ as $x \to 0$. This will ensure that in the absence of outliers the M-estimator will closely approximate the standard least squares problem. The unique β satisfying this property is $\beta = \frac{\delta^2}{2}$. We will henceforth refer to the resulting kernel:

$$\rho(x) = \frac{1}{2}\frac{\delta^2 x^2}{\delta^2 + x^2}, \tag{5.23}$$

as the *generalized Geman-McClure* kernel (generalized with respect to (Zhang, 1997)).

5.4.3 Relation to Geman-McClure

As shown earlier in Section 5.3, each switchable constraint (Sünderhauf and Protzel, 2012) on a factor connecting nodes i and j in a SLAM graph contributes the following value to the chi-squared cost of the full problem:

$$s_{ij}^2\chi_{ij}^2 + (1 - s_{ij})^2\Phi, \tag{5.24}$$

where χ_{ij}^2 denotes the chi-squared value of the factor connecting i to j and Φ is the usual scale parameter for DCS and SC. It was proven in Equation (5.8) that the necessary condition for optimality of a switchable constraint s_{ij} is:

$$s_{ij} = \frac{\Phi}{\chi_{ij}^2 + \Phi}. \tag{5.25}$$

By substituting Equation (5.25) into Equation (5.24) we find the following:

$$\begin{aligned}&\left(\frac{\Phi}{\chi_{ij}^2 + \Phi}\right)^2 \chi_{ij}^2 + \left(1 - \frac{\Phi}{\chi_{ij}^2 + \Phi}\right)^2 \Phi \\ =\ &\frac{\left(\Phi + \chi_{ij}^2\right)\Phi\chi_{ij}^2}{\left(\chi_{ij}^2 + \Phi\right)^2} \\ =\ &\frac{\Phi\chi_{ij}^2}{\Phi + \chi_{ij}^2}.\end{aligned} \tag{5.26}$$

We thus conclude that, from an optimization standpoint, removing the dependence on the switching variable s_{ij} is equivalent to optimizing the original problem with the kernel function:

$$\rho(x) = \frac{\Phi x^2}{\Phi + x^2}. \tag{5.27}$$

It is important to note that SC (Sünderhauf and Protzel, 2012) and the kernel function in Equation (5.27) are equivalent, in the sense that all the minima of one method are also minima of the other, and vice versa. The major difference between the two lays in efficiency and convergence, which will be shown later in the experimental section.

Notice that Equation (5.27) is an instance of the generalized Geman-McClure kernel, with $\delta = \sqrt{\Phi}$. The factor of one half is implicit in the sense that it is neglected when computing the negative log-likelihood of the original probabilistic model.

Interestingly, even the DCS formulation in Equation (5.20) is a scaled and translated generalized Geman-McClure for $x^2 > \Phi$, with $\delta = \sqrt{\Phi}$, in fact:

$$\frac{\Phi\left(3x^2 - \Phi\right)}{2\left(x^2 + \Phi\right)} = 4\rho(x) - \frac{\Phi}{2}. \tag{5.28}$$

Qualitatively, DCS and the generalized Geman-McClure behave similarly. The two main differences are:

- DCS has a more defined inlier-outlier distinction. As a matter of fact, for $x^2 < \Phi$, DCS is *exactly* a non-robust least squares problem with a squared kernel, while the generalized Geman-McClure is only asymptotic to it.
- DCS and generalized Geman-McClure saturate at different values, i.e.:

$$\lim_{x \to \pm\infty} \frac{\Phi\left(3x^2 - \Phi\right)}{2\left(x^2 + \Phi\right)} = \frac{3}{2}\Phi, \tag{5.29}$$

$$\lim_{x \to \pm\infty} \frac{1}{2}\frac{\Phi x^2}{\Phi + x^2} = \frac{\Phi}{2}. \tag{5.30}$$

  This qualitatively means that DCS is in general more accepting of outliers or inliers with wrong uncertainties than the generalized Geman-McClure kernel.

5.5 Experimental Evaluation

We implemented DCS, which was derived in Section 5.3, and conducted a large set of evaluations comparing it to switching constraints (SC) (Sünderhauf and Protzel, 2012). We have used the g2o framework with Gauss-Newton optimization steps for all our experiments (Kümmerle et al., 2011).

Table 5.1: Datasets used in our experiments.

Dataset	# Poses & Landmarks	# Correct Loop Constraints	# Outliers (max.)
ManhattanOlson	3,500	2,099	5,000
ManhattanG2O	3,500	2,099	5,000
City10000	10,000	10,688	5,000
Intel Research	943	894	5,000
Sphere2500	2,500	2,450	5,000
CityTrees10000	10,100	4,443	1,000
Victoria Park	7,120	3,640	1,000
Bicocca	43,116	767	516
Lincoln Labs	6,357	2,334	3,754

5.5.1 Datasets and Outliers

To support comparisons, we used publicly available datasets, namely the Manhattan3500, Intel Research Lab, City10000, Sphere2500, CityTrees10000, and Victoria Park datasets. For Manhattan3500, we considered the two different initialization procedures provided by Olson et al. (2006) and g2o (Kümmerle et al., 2011). The Intel Research Lab dataset is available in the g2o package (Kümmerle et al., 2011) and the City10000, CityTrees10000, Sphere2500 datasets as well as the Victoria Park dataset were released with the iSAM package (Kaess et al., 2007a). We also evaluated additional large-scale datasets such as the 36 loops of the Lincoln Lab and the five loops of the Bicocca multi-session experiment initially evaluated with RRR (Latif et al., 2012a). Table 5.1 lists the properties of the different datasets as well as the maximum number of outliers present in the corrupted versions. For landmark datasets, the loop constraints are pose-landmark edges.

The corrupted versions of the data sets contain both real and simulated outliers. For simulated outliers, we used four different approaches to generate them, namely "random", "local", "random grouped", and "local grouped" as described in (Sünderhauf and Protzel, 2012). Random outliers connect any two randomly sampled nodes in the graph. Local outliers connect random nodes that are in the vicinity of each other. For the grouped outliers, we create clusters of 10 mutually consistent outliers. We believe that randomly grouped outliers are the most realistic form of outliers as such constraints are similar to systematic errors generated due to perceptual aliasing by a front-end. The outliers are generated using the script provided in the Vertigo package (Sünderhauf, 2012). For landmark datasets such as Victoria Park and CityTrees10000, we added wrong loop closures between random pairs of nodes and landmarks.

For the Bicocca and Lincoln multi-session datasets, we used the processed datasets

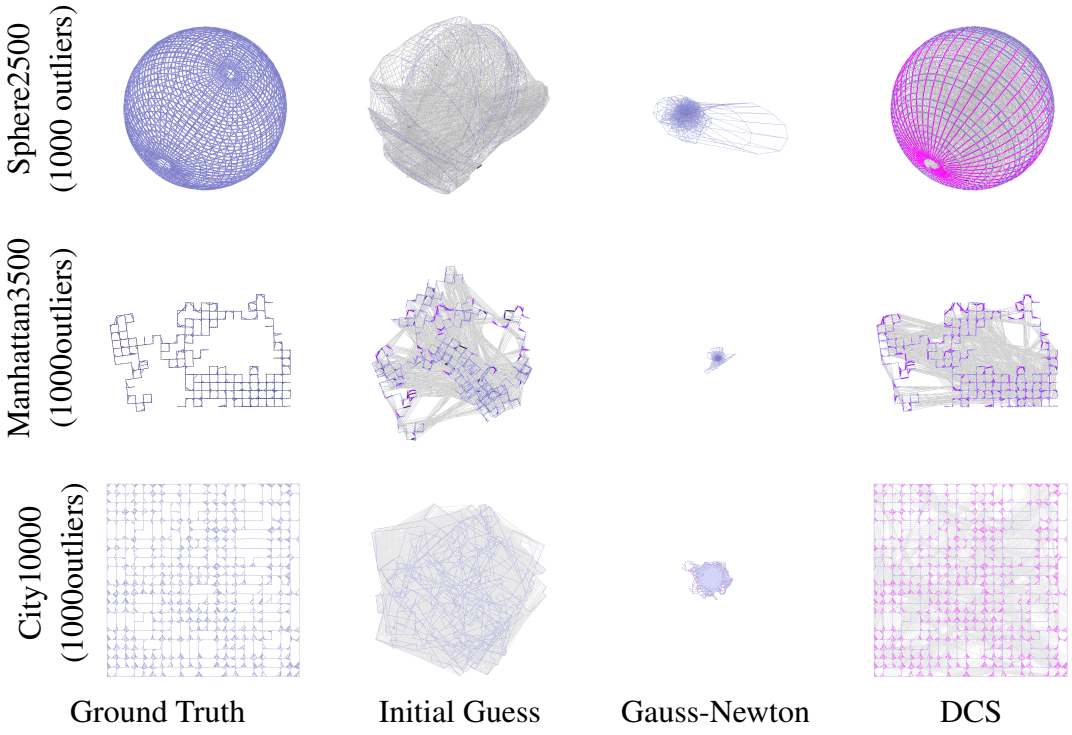


Figure 5.1: The figure illustrates the performance of Dynamic Covariance Scaling (DCS) in the presence of outliers. DCS converges to the correct solution in the presence of 1000 outliers for the sphere2500, Manhattan3500 and City10000 datasets. The third column shows that the standard Gauss-Newton with a squared kernel fails to reach the optimum solution.

provided by Latif et al. (2012a) in which loop closures are generated using a place recognition system subject to perceptual aliasing. The Bicocca dataset uses a bag-of-words based front-end while the Lincoln Lab dataset was created with a GIST-based front-end (Oliva and Torralba, 2001; Sünderhauf and Protzel, 2011). For all evaluations, unless otherwise stated, $\Phi = 1$, since this is suggested value according to Sünderhauf and Protzel (2012).

5.5.2 Robustness Against Simulated and Real Outliers

To show the robustness against outliers we evaluated DCS on both simulated and real outliers. First, we evaluated DCS on datasets with up to 5,000 simulated outliers. In total, we evaluated 400 graphs per dataset—100 for each of the four outlier generation strategy. Scatter plots of the resulting re-projection error (RPE) (Kümmerle et al., 2009) after convergence are shown in Figure 5.2. As can be seen, for the ManhattanG2O, Intel and Sphere2500 datasets, DCS always converges to the correct solution. For ManhattanOlson and City10000, DCS converges in all the local outlier cases but is sensitive to the non-

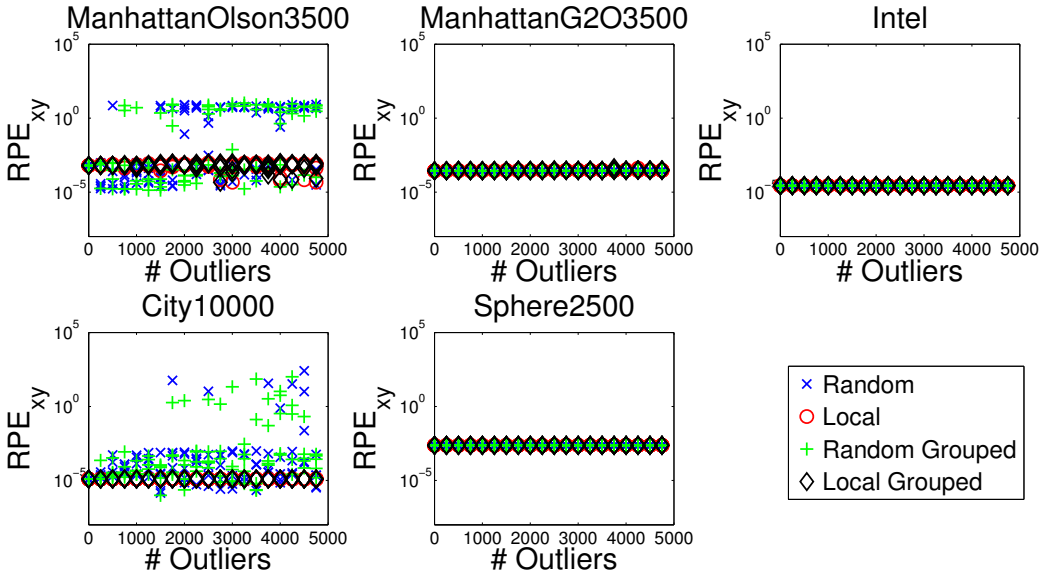
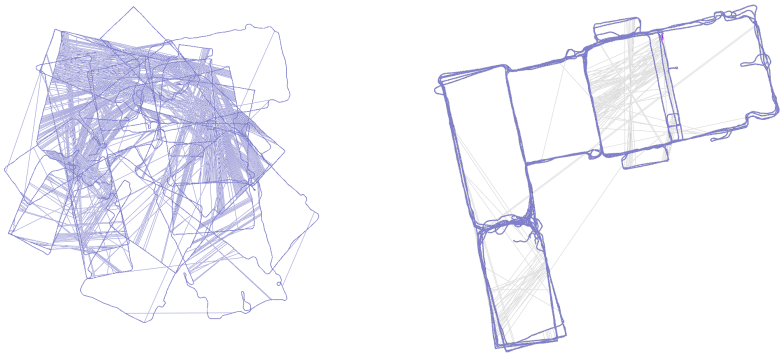


Figure 5.2: Scatter plots showing the error depending on the number and type of outliers for DCS. ManhattanG2O, Intel, and Sphere2500 converge to the correct solution even with 5,000 outliers while City10000 and ManhattanOlson always convergence in the case of local outliers. City10000 converges to the correct solution for up to 1,500 outliers which are not local. ManhattanOlson is more sensitive to non-local outliers.

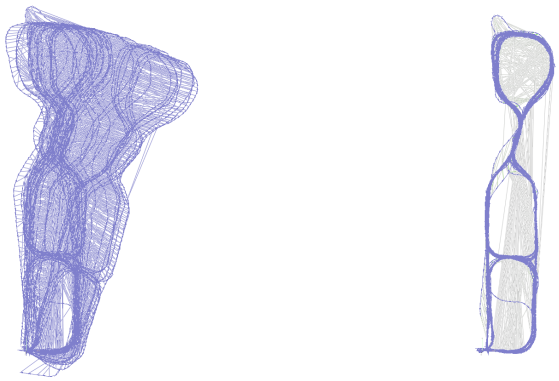
local outliers. City10000 fails to converge to the correct solution in some non-local cases with more than 1500 outliers. Even when ManhattanOlson does not converge, the RPE is always less than 10 and appears somewhat constant. This case is further analyzed in Section 5.5.8.

Compared to the other datasets evaluated in this chapter, the next two datasets contain outliers created from a front-end due to place recognition errors. The goal of these experiments is to evaluate the performance of DCS with an error prone front-end. We use the data-sets evaluated by Latif et al. (2012a) and thus also provide an informal comparison to RRR. Figure 5.3 depicts the optimization results of DCS on the Bicocca and Lincoln Lab datasets.

DCS takes 0.79 s (3 iterations) to optimize the Lincoln Lab dataset and 1.56 s (16 iterations) to optimize the Bicocca dataset. For the Bicocca dataset, we achieved the best result with $\Phi = 5$. By visual inspection, we can see that our solution is close to the reported correct structure in (Latif et al., 2012a). Compared to RRR, which reports a timing of 314 s for the Bicocca dataset, DCS takes only 1.56 s and thus is around two orders of magnitude faster. SC does not find the correct solution in the standard settings and requires an additional Huber robust kernel which takes 5.24 s to find the solution (Latif et al., 2012a). We also evaluated DCS on the peninsula graph which was originally evaluated by Wang and Olson (2014). DCS is able to converge to the correct



Bicocca initialization Result after optimization with DCS



Lincoln Labs initialization Result after optimization with DCS

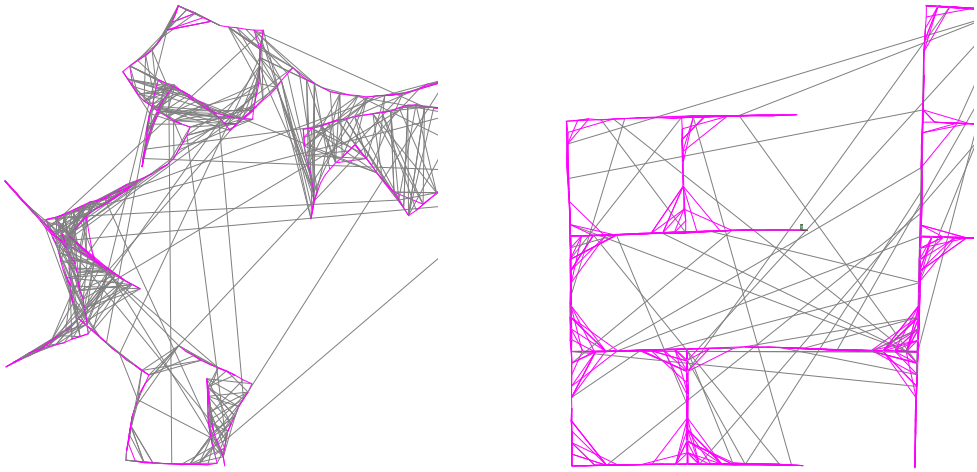
Figure 5.3: Qualitative evaluation on 5 sessions of Bicocca (top) and 36 loops of Lincoln Labs (bottom) datasets that contain outliers generated by the vision system. Latif et al. (2012a) report that RRR solves Bicocca in 314 s whereas DCS requires only 1.56 s to obtain the solution.

minimum, illustrated in Figure 5.4. This dataset is particularly challenging as it contains two lobes connected by a single odometry chain. We achieved the best result with $\Phi = 5$.

5.5.3 Timing analysis

The next set of experiments are conducted in order to analyze the timing performance of DCS and to support the claim that our method converges faster than SC. We first show in Figure 5.6 that the optimization time required by DCS depends only on the number of constraints and the outlier generation criteria. Importantly, DCS does not increase the optimization time significantly compared to the standard least squares. The required optimization time shows a larger variance for random outliers in ManhattanOlson compared to ManhattanG2O as the former starts with a worse initialization.

Figure 5.5 illustrates the advantage over SC in terms of convergence speed. SC takes many more iterations compared to DCS. Table 5.2 compares the time required by DCS



Intialization Optimized result with DCS

Figure 5.4: Analysis of DCS on the peninsula graph that was previously evaluated with SGD-MM (Wang and Olson, 2014). DCS is able to converge to the correct solution even for this challenging simulated example.

and SC to converge in presence of outliers. This table compares the total time taken for both these algorithms to reach the optimal solution. As can be seen from the table, DCS is faster than SC in all cases. The increase in convergence speed is most noticeable in City10000 dataset. The optimization process for both methods were stopped when the change in χ^2 was less than the set threshold. In the next section we show reduction of χ^2 and RPE is significantly faster and smoother for DCS compared to SC.

5.5.4 Convergence Properties

The experiments in this section analyze the convergence behavior of DCS and SC in the presence of 1000 randomly grouped errors, as this is the most difficult and realistic scenario. Figure 5.7 plots the evolution of the RPE (top row) and the χ^2 error (bottom row) during optimization for SC (blue) and DCS (green). As can be seen from these plots, within at most 6 iterations, DCS converges while SC typically needs between 15 and 20 iterations to converge. The shapes of the plots for SC reveal a frequent increase of the RPE as well as χ^2 error. We believe this may be indicative of the fact that the Gauss-Newton quadratic approximation of the cost functions for the new optimization problem with additional switch variables in SC is not completely accurate in the neighborhood of

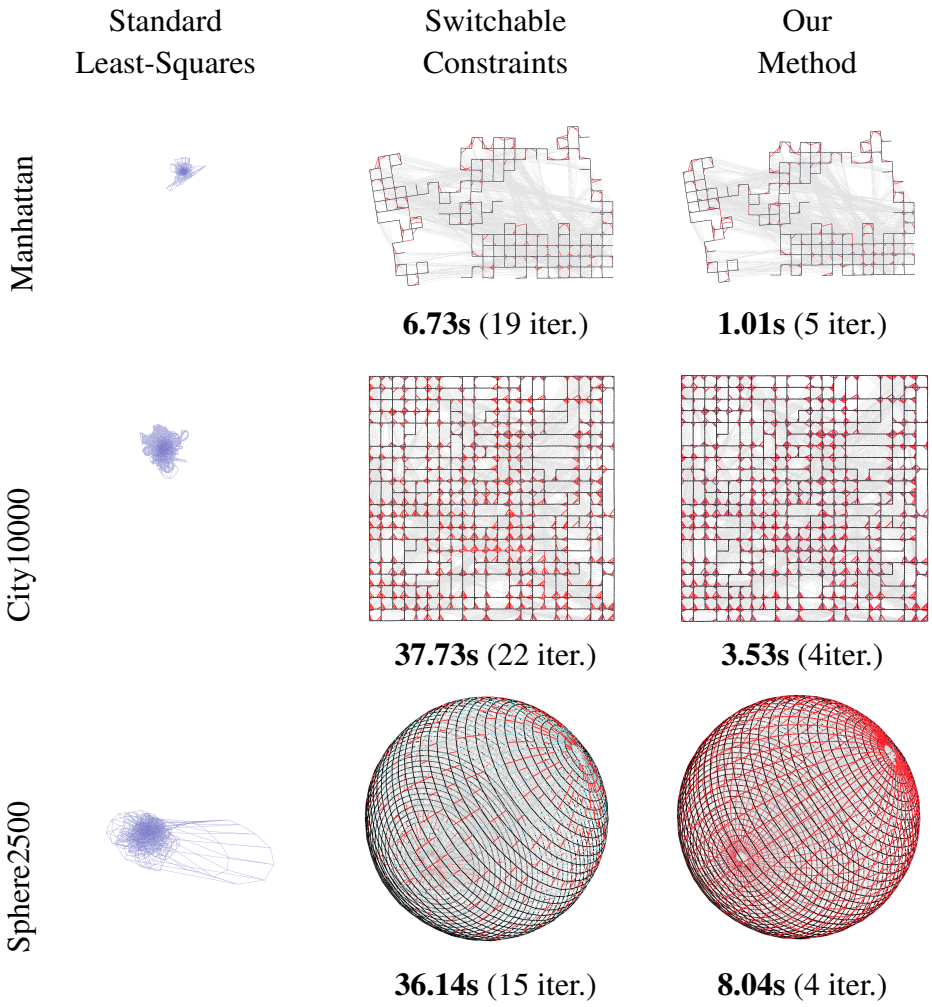


Figure 5.5: Graph optimization on standard datasets with 1,000 outliers. Standard least squares (left) fails while the switchable constraints method (center) as well as our approach (right) provide comparable results. Our method shows substantially faster convergence of up to a factor of 10. Colored lines depict accepted loop closures.

Table 5.2: Optimization time needed by SC and DCS in the presence of 1000 to 5000 outliers with random(R), local(L), random grouped(RG) and local grouped(LG) outlier generation strategies.

Dataset	1000 R, L, RG, LG	2000 R, L, RG, LG	3000 R, L, RG, LG	4000 R, L, RG, LG	5000 R, L, RG, LG
ManG2O					
SC	4.70, 1.91, 3.11, 1.55	8.17, 2.93, 4.46, 2.85	10.11, 3.45, 11.89, 5.11	11.21, 2.80, 11.17, 3.32	24.53, 3.14, 15.33, 4.67
DCS	**2.09, 0.86, 1.41, 0.88**	**3.83, 1.07, 2.80, 1.00**	**5.47, 1.25, 4.24, 1.17**	**7.62, 1.44, 6.27, 1.38**	**9.29, 1.69, 8.42, 1.59**
ManOlson					
SC	14.53, 2.21, 10.65, 2.21	18.96, 2.80, 15.45, 2.71	39.34, 3.41, 39.94, 3.27	53.29, 4.69, 36.71, 4.54	67.44, 5.33, 61.16, 5.09
DCS	**4.62, 1.08, 3.40, 1.07**	**6.57, 1.35, 3.23, 1.27**	**26.21, 1.57, 20.21, 1.46**	**29.24, 1.84, 26.46, 1.71**	**16.80, 2.03, 14.00, 1.93**
Intel					
SC	0.54, 0.42, 0.51, 0.39	1.20, 0.94, 1.18, 0.94	1.60, 1.22, 1.61, 1.20	2.00, 1.52, 2.01, 1.50	2.37, 1.78, 2.44, 1.74
DCS	**0.34, 0.22, 0.31, 0.21**	**0.52, 0.31, 0.52, 0.34**	**0.69, 0.45, 0.71, 0.42**	**0.85, 0.53, 0.85, 0.50**	**1.00, 0.58, 1.08, 0.58**
City10000					
SC	47.61, 30.06, 41.11, 29.86	108.2, 33.84, 79.50, 33.52	212.8, 41.14, 134.9, 39.04	285.7, 43.82, 207.1, 40.70	389.9, 49.98, 446.5, 49.92
DCS	**10.09, 3.98, 7.88, 3.93**	**36.94, 4.80, 15.74, 4.53**	**51.60, 5.95, 34.02, 5.65**	**218.8, 6.92, 50.09, 6.44**	**262.9, 8.04, 393.2, 7.37**
Sphere2500					
SC	53.83, 11.09, 48.26, 10.62	115.5, 14.88, 108.9, 16.03	240.1, 24.10, 170.3, 18.55	218.7, 30.78, 230.2, 57.22	310.7, 67.53, 281.8, 63.37
DCS	**19.52, 7.83, 16.84, 7.51**	**42.52, 9.22, 38.39, 9.02**	**50.58, 10.40, 50.32, 9.94**	**66.51, 11.31, 69.39, 11.36**	**90.12, 12.35, 97.07, 11.97**

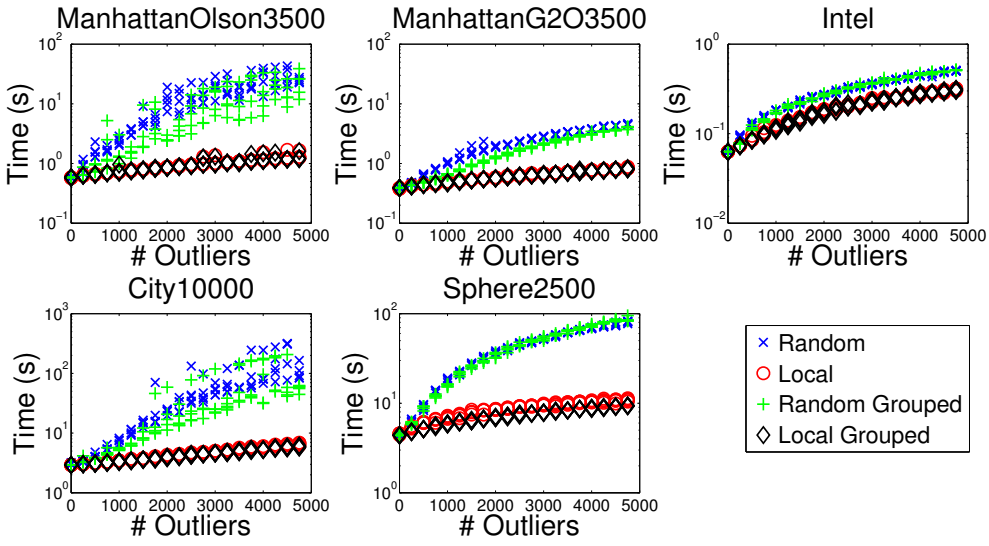


Figure 5.6: Scatter plots showing the runtime depending on the number and type of outliers for DCS. DCS does not add significant overheads to the sparse matrix factorization algorithms compared to standard least squares. Local outliers create less fill-in inside and hence requires less time.

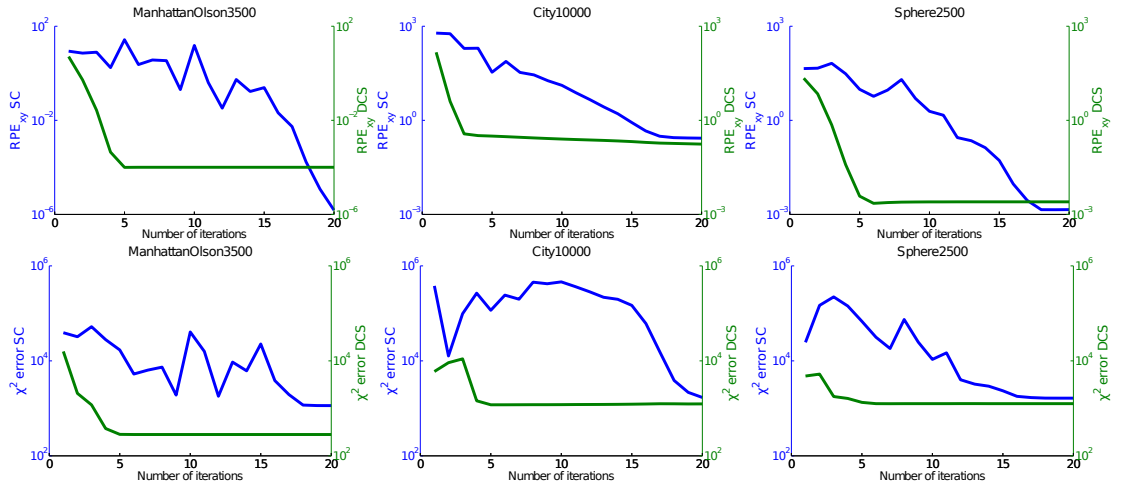


Figure 5.7: The figure plots RPE (top row) and χ^2 error (bottom row) for 20 iterations for SC and DCS. While DCS converges within 6 iterations or less, SC needs between 15 and 20 iterations to converge. The shapes of the plots for SC reveal a frequent increase of RPE and χ^2 error which tend to indicate that there are more local minima in the SC formulation compared to DCS.

evaluation.

For our method, the evolution of χ^2 and RPE is smooth and almost monotonous. The plots illustrate that DCS requires a smaller number of iterations and offers a faster convergence while at the same time being robust to outliers. This is also apparent from the following video http://www.informatik.uni-freiburg.de/%7Eagarwal/videos/icra13/DCS.mp4. Note that the absolute χ^2 values for SC and DCS have to be interpreted differently since SC introduces extra switch prior constraints contributing to the overall error.

5.5.5 Parameter Sensitivity

To analyze the sensitivity of DCS and SC with respect to the parameter Φ, we analyzed several values of Φ in the range of 0.01 and 10. Both methods are evaluated on the standard datasets adding 1,000 outliers using random, local random, grouped, and local grouped outliers. Once chosen, the outliers are not changed for different values of Φ. Figure 5.8 shows the RPE for varying values of Φ. In general, we found that the sensitivity of DCS and SC is similar for ManhattanG2O, Intel and City10000 datasets. Small values of $\Phi < 0.1$ lead to larger RPE. The RPE however is small, around 10^{-5}, for a wide range of values $0.1 < \Phi < 10$.

For the Sphere2500 dataset, both DCS and SC do not converge for $\Phi < 0.1$. The convergence improves with increasing Φ but DCS fails to converge for $\Phi > 5$ in the presence of local grouped outliers. For the ManhattanOlson dataset, DCS and SC converge for values $0.1 < \Phi \leq 1$ in all cases while both approaches appear to be sensitive to non-local errors. This may be explained by the structure of this dataset since it can be

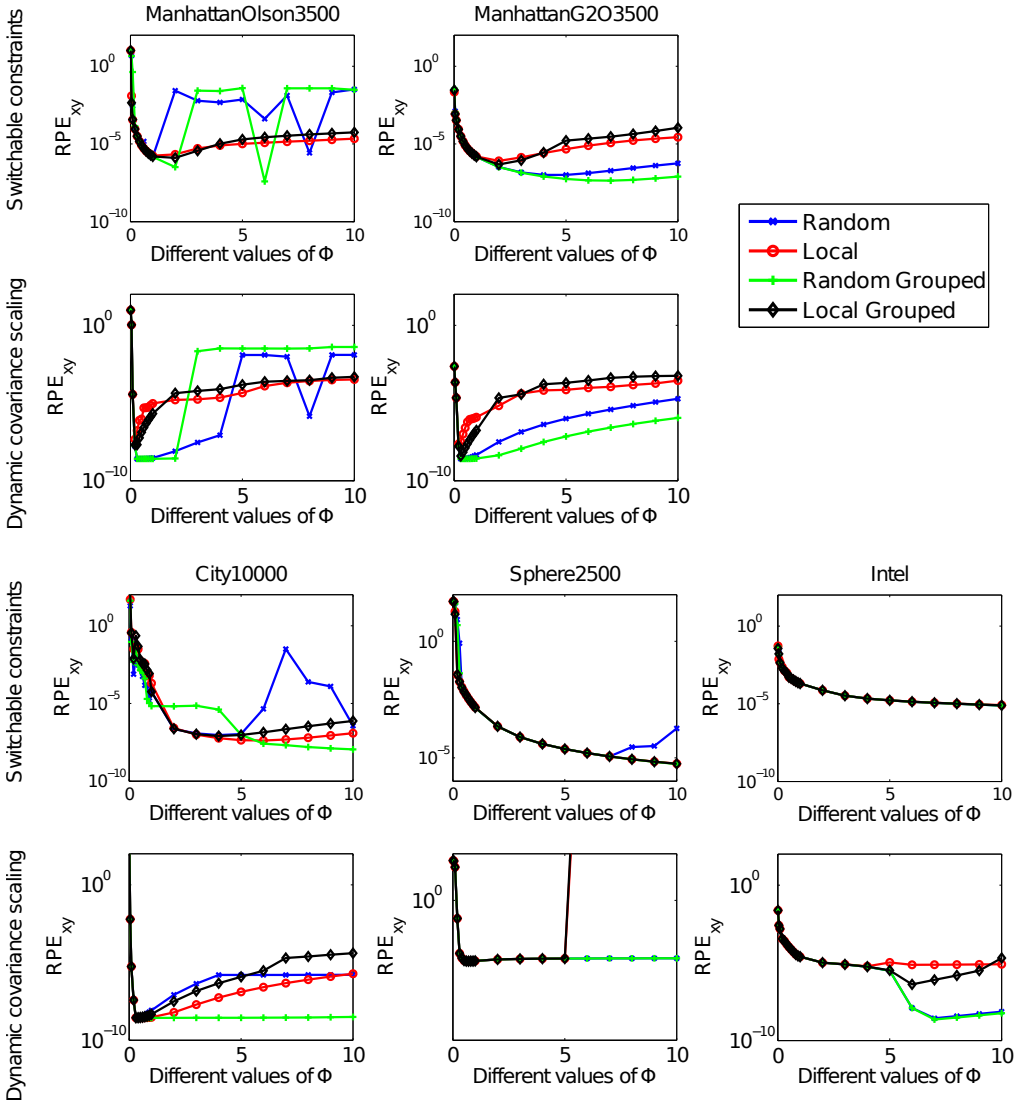
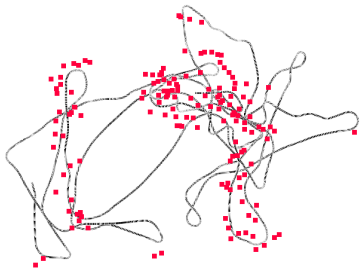
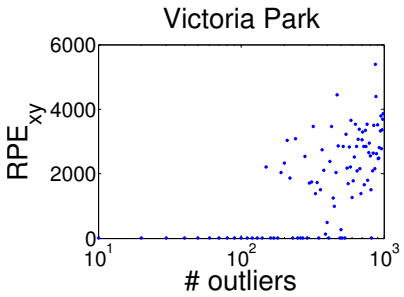


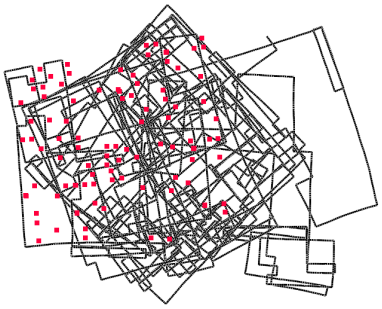
Figure 5.8: Robustness of SC and DCS with respect to the parameter $\Phi \in [0.01, 10]$ in presence of 1,000 outliers.



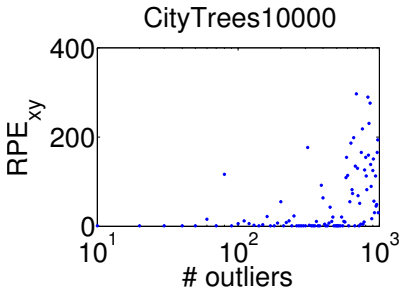
(a) Victoria Park initialization



(b) RPE error vs Num of outliers



(c) CityTrees10000 initialization



(d) RPE error vs Num of outliers

Figure 5.9: Resulting RPE for Victoria Park and cityTrees10000 dataset in the presence of a varying number of outliers. Although the initialization is far from the global minimum, DCS is able to converge to the correct solution for small number of outliers.

divided into three parts which are connected by a few constraints only (see left part of ground truth configuration in Figure 5.11 and the colored parts). In summary, DCS appears to be more sensitive to the value of Φ in the case of the Sphere2500 dataset but for all other datasets the sensitivity on Φ is comparable for both approaches. Importantly, DCS and SC show a wide range of values for Φ and we thus fixed $\Phi = 1$ as also suggested by Sünderhauf and Protzel (2012).

5.5.6 Analysis on Landmark-Based Graphs

So far, we evaluated our method only on datasets that contain pose-to-pose constraints, i.e., that directly relate pairs of poses of the robot with each other. Our method also works for landmark-based SLAM. Most landmark-based SLAM systems provide pose-feature range-bearing constraints and pose-to-pose constraints only for odometry. Operating on pose-feature constraints is more challenging for outlier rejection since there is no reliable constraints such as odometry between the feature nodes. In the previous evaluated pose graphs, every node is constrained by two odometry edges which are not subjected to being an outlier. For landmark datasets all constraints to a feature node are potential outliers

and hence create large number of local minimas.

For landmark datasets we corrupt the outlier free Victoria Park and the CityTrees10000 dataset with up to 1,000 random outlier constrains. The outliers are random measurements from a robot pose to a landmark. Figure 5.9 shows the initialization for these two datasets and the RPE with an increasing number of outliers. As can be seen from the plots, in these two datasets, DCS is robust up to around 100 outliers and the robustness decreases as the outliers are increased thereafter. The fact that DCS is still able to optimize the Victoria Park dataset from the initialization shown for 100 random outliers is strong evidence that the method can be used in synergy with existing front-ends validation techniques for landmark based system to improve robustness.

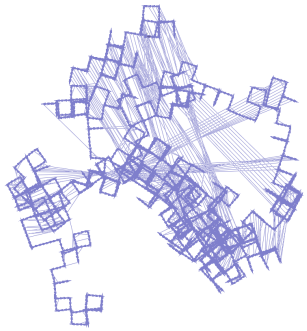
5.5.7 Comparison to other M-Estimators

Huber and Cauchy are commonly used M-estimators to reject outliers during the optimization phase. For this reason we compared Huber and Cauchy with DCS on the Manhattan3500 dataset with 100 outliers generated in groups of 10. One instance of the graph is shown in the top row of Figure 5.10. DCS outperforms both Huber and Cauchy as illustrated in Figure 5.10. Moreover, it converges to the correct solution for a wide range of the tuning parameter Φ.

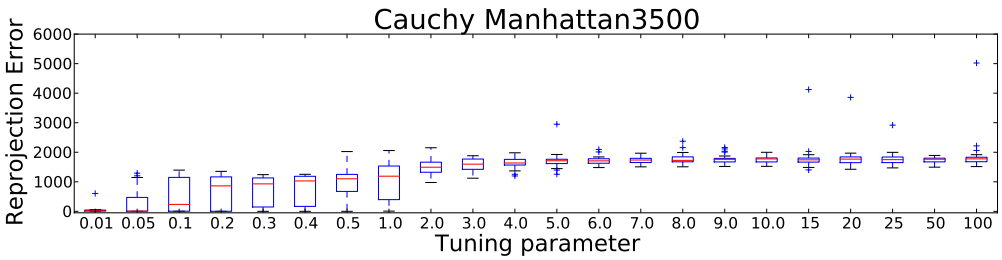
5.5.8 Degenerate Cases

Investigations of the failure cases in ManhattanOlson found is Section 5.5.2 reveal an interesting behavior. We analyze two specific failure cases, one with 501 and one with 4,751 random outliers. After converging, both solutions appear to have similar configuration, even though the second case is subjected to roughly ten times more outliers, shown in Figure 5.11. They are locally consistent and appear to have converged to a similar local minima. The scaling values of each false positive edge is shown in the plots in Figure 5.11. The problem here is that three parts of the graph are only sparsely connected (see Figure 5.11-top). By adding non-local and mutual consistent outliers, there exists configurations in which the system cannot determine all outliers correctly. SC shows a similar issue with ManhattanOlson, which the authors solved by introducing an additional robust Huber kernel at the expense of an even slower convergence (Sünderhauf and Protzel, 2012).

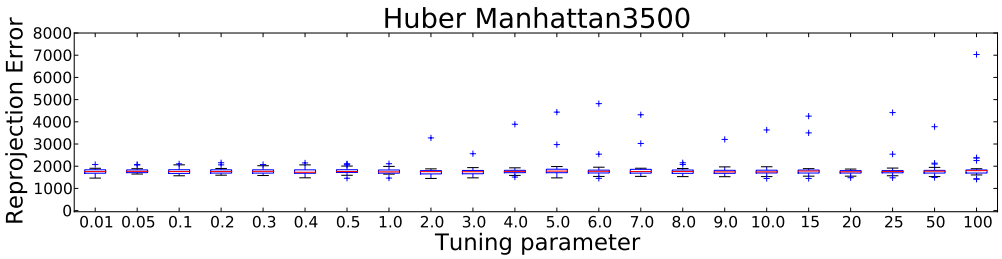
The parking garage dataset is a difficult real world dataset compared to all the previous datasets. This is mainly because of the sparse nature of loop closures. Each deck of the parking garage is connected by two odometry chains. SC had reported degenerate behavior with this dataset (Sünderhauf and Protzel, 2012). It argued that since only a small number of constraints connect the decks robust methods were not able to outperform non-robust methods.



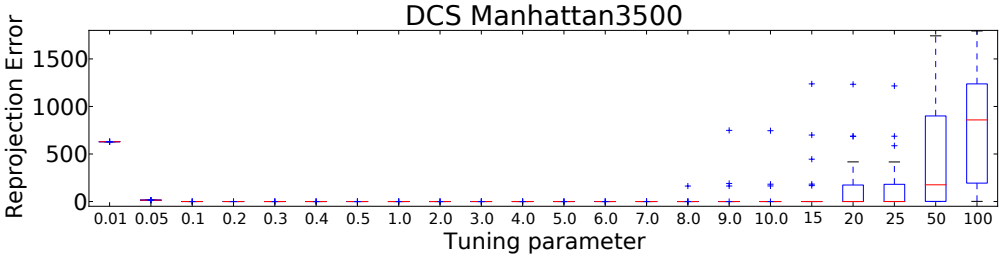
An example of one of the 25 instances used for this evaluation.



The use of Cauchy kernel reaches local minimum.



The use of Huber kernel reaches local minimum.



DCS reaches the correct solution for a wide range of tuning parameters.

Figure 5.10: DCS outperforms Cauchy and Huber M-estimators on the Manhattan3500 dataset containing 100 outliers mutually consistent in groups of 10. Each tuning parameter is evaluated on 25 simulated graphs.

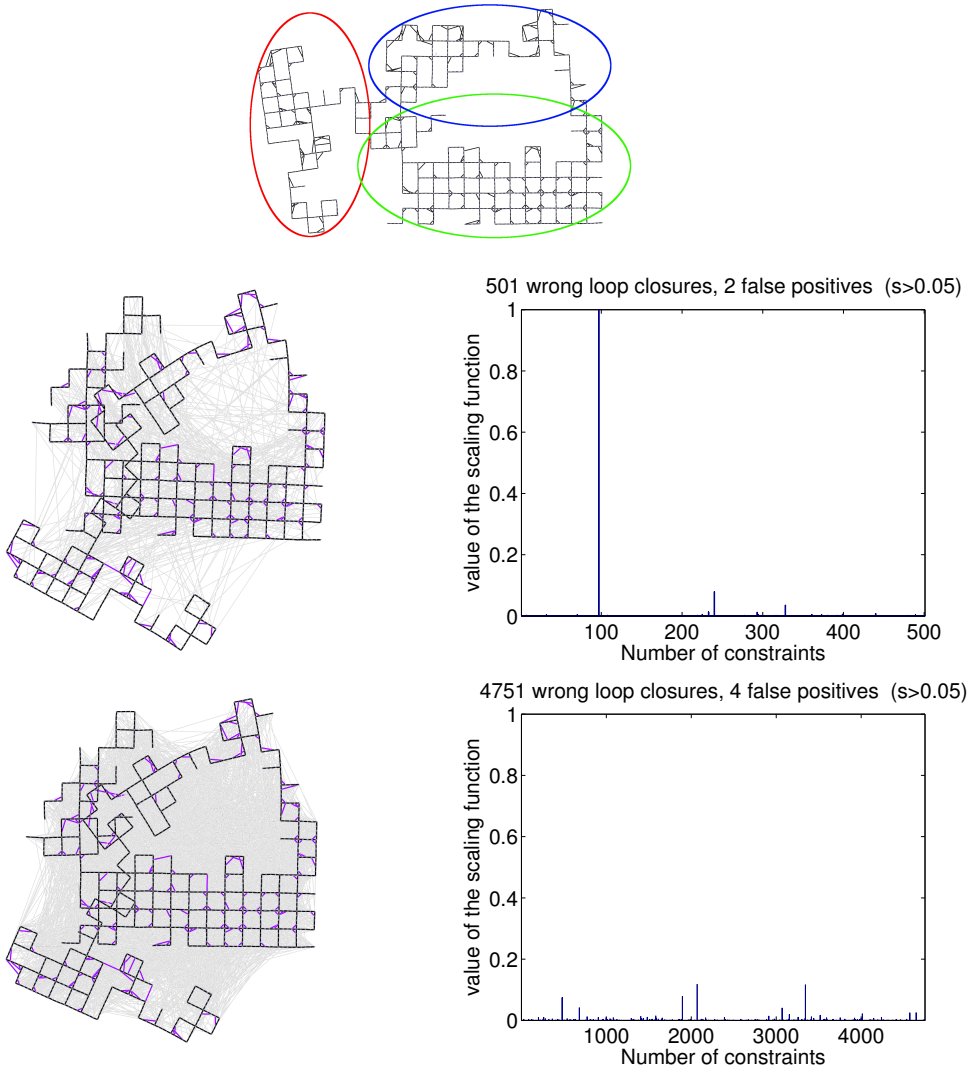


Figure 5.11: Top: Ground truth configuration for Manhattan3500. The dataset reveals three sparsely connected region illustrated by the colored ellipses. The other four images are designed to illustrate the two failure cases, obtained for 501 and 4,751 random outliers, in the ManhattanOlson dataset. The images show the local minima maps in both situations together with the scaling values for the false positive constraints. The plots show that even if our method fails to converge to the optimal solution, the number of false positives accepted by the system is small, evident by a small scaling factor. With 501 outliers only two constraints have a scale value of more than 0.05 and with 4,751 outliers only four outliers have a scale value more than 0.05.

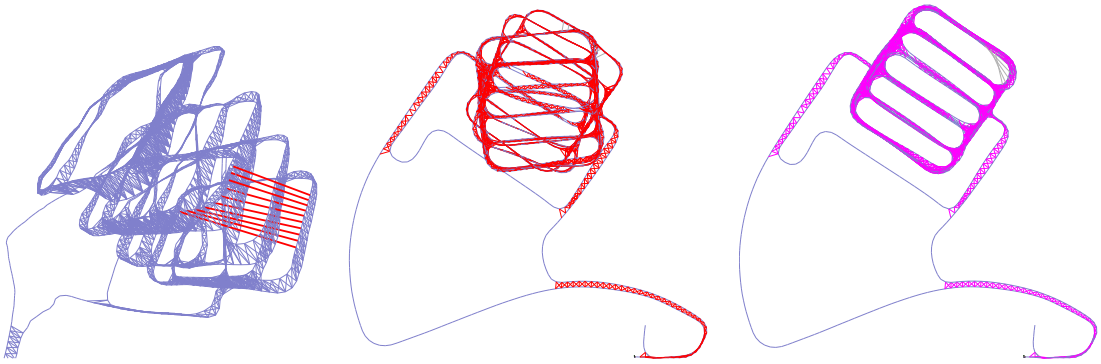


Figure 5.12: Parking garage dataset with sparse connection. (Left) The original datasets with wrong loop closures connecting different decks in red. Note: the z-axis is scaled up to show the wrong edges. (Center) SC returns the wrong solution while DCS rejects the outliers(right). This figure shows DCS being able to reject outliers even in the challenging case of datasets with minimal graph connectivity.

DCS is able to reject outliers even in this dataset. We also added mutually consistent constraints between decks at multiple levels and compared both methods with standard parameters as shown in Figure 5.12. We believe DCS is able to reject outliers as the gradients of odometry edges and correct loop edges outweigh those provided by the outliers.

The findings reported here have been confirmed by other authors (Sünderhauf and Protzel, 2013; Latif et al., 2014). Furthermore, DCS has been integrated into the popular g2o optimization framework (Kümmerle et al., 2011) and is used by several research groups worldwide. For example, Carlevaris-Bianco (2015) rely on DCS for improving the robustness of long-term visual SLAM in dynamic environments. Additionally, Ozog et al. (2015) use it for autonomous ship hull inspection using an unmanned underwater rover and Lee and Myung (2014) rely on it for building large maps with RGB-D cameras.

5.6 Summary

In this chapter we introduced dynamic covariance scaling (DCS), an elegant and principled method to cope with outliers in graph-based SLAM systems. We showed that DCS generalizes the switchable constraint method of Sünderhauf and Protzel (2012), while introducing a substantially lower computational overhead. Additionally, we relate DCS to robust M-estimators and show that DCS is a special instance of Geman-McClure. This is achieved by analyzing the behavior of the error function and deriving an analytical solution for computing the weighting factors. We thoroughly evaluated our approach and supported our claims with extensive experiments and comparisons to state-of-the-art methods on publicly available datasets. The results show DCS is robust to outliers with a

convergence rate that is substantially faster than existing state-of-the-art methods. In the next chapter, we will illustrate that DCS, in addition to being robust to front-end outliers, is also robust to poor variable initialization.

Chapter 6

Robust Map Optimization Under Poor Initial Estimates

As we have seen in the previous chapters, non-linear error minimization methods became widespread approaches for solving the simultaneous localization and mapping problem (SLAM). Converging to the correct solution is challenging if outliers are present or the initial guess is far away from the global minimum. In the previous chapter we introduced dynamic covariance scaling (DCS) and evaluated its robustness against outliers. This chapter presents an experimental analysis of DCS, in the context of a poor initialization. Our evaluation shows that DCS is able to mitigate the effects of poor initializations far away from the true solution. In contrast to other methods that first aim at finding a good initial guess to seed the optimization, our method is more elegant because it does not require an additional method for initialization.

6.1 Introduction

State estimation and environment modeling are core capabilities of modern robots and in many such problems, non-linear optimization plays a major role. This is, for example, the case in SLAM and bundle adjustment. Approaches to non-linear optimization such as Gauss-Newton, Levenberg-Marquardt, or Dog-Leg seek to find a minimum of the given error function. However, due to the non-convexity of the error surface, they cannot guarantee to find the global minimum. In practice, the initial guess has a strong impact on the quality of the computed solution. A good initialization, close to the correct solution, often guides the optimization to the global minimum, while a bad initialization could result in the optimization converging to a local one. Finding the right solution is often challenging and sometimes even impossible if the initial guess is far away from the correct solution.

The initial configuration of the graph to be optimized often has a strong impact on the final result as the error minimization procedure may get stuck in local minima as

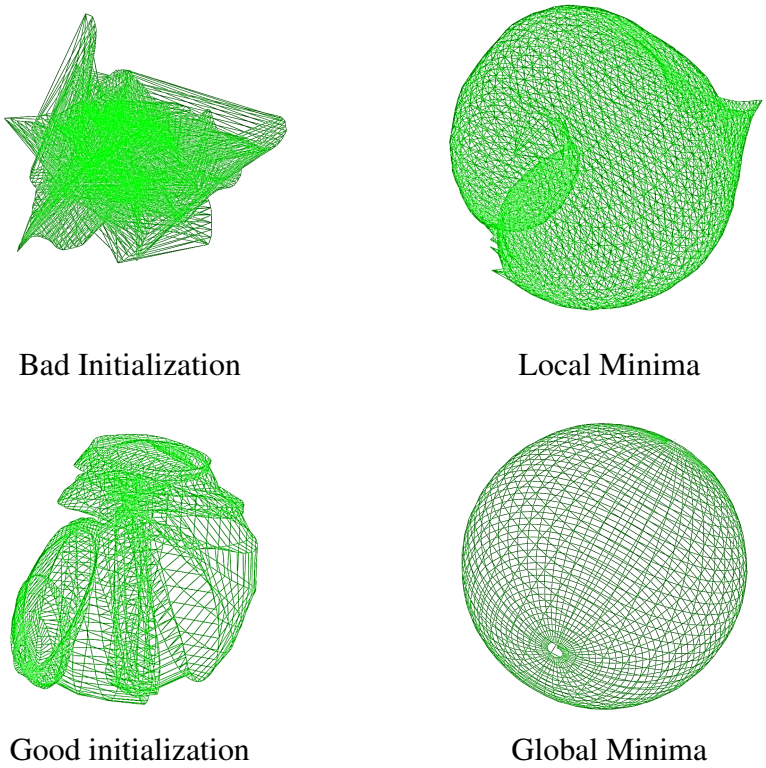


Figure 6.1: The figure illustrates that a poor initialization may result in a suboptimal solution: one that converges to a local minima, while a better initialization reaches the global minimum.

illustrated in Figure 6.1. This holds for pose-graph SLAM as well as for graphs that contain robot poses and features. The sensor properties and the choice of the observation function has a strong impact on the convergence properties. As illustrated by Grisetti et al. (2012), bad initializations quickly lead to divergence especially in the context of non-linear models. They propose to employ an approximation of the original problem that partitions the factor graph with a divide-and-conquer approach to exploit local estimates. Their method offers a larger convergence basin than Levenberg-Marquardt and yields convergence to the true solution in real world and simulated scenes where other state-of-the-art methods fail. Hu et al. (2013) use the Cauchy M-estimator as a bootstrapper for optimizing datasets with high-noise but no outliers. For pose-graphs, Carlone et al. (2011) proposed a solution for finding a linear approximations. Additionally, Carlone and Censi (2012) outline a method to first compute the rotational unknowns which help in improving the convergence. These results, however, do not generalize to graphs with features or with constraints in 3D.

The problem of good initialization is implicitly addressed by the submapping and hierarchical techniques proposed over the last years in the context of EKF SLAM and graph-based techniques, for example (Bosse et al., 2004; Frese, 2006; Grisetti et al.,

2010b; Ni and Dellaert, 2010). Although the motivation for most submapping techniques was bounding the computational complexity or online optimization, these techniques often also increase the robustness to initialization of the system. Computing local solutions and combining them to a global solution can be seen as computing an improved initial alignment for parts of the problem. As a result, standard approaches often perform well when combining the partial solutions into a global one. Incremental optimization approaches (Grisetti et al., 2010b; Kaess et al., 2008; Olson et al., 2007) that optimize the graph at each step can have a similar effect.

The contribution of this chapter is an analysis of dynamic covariance scaling (DCS), proposed in the previous chapter, under poor initial configurations that can occur in the context of SLAM. We believe this positive impact is caused by the reweighting of factors as this reduces the influence of erroneous Jacobians computed far away from the correct configuration. The experimental results presented in this chapter show that DCS solves the optimization problems that previously required the computation of condensed measurements (CM) (Grisetti et al., 2012). The advantage of DCS consists in not relying on any sophisticated initialization method and at the same time in naturally handling outliers. All other initialization methods assume that all constraints are correct.

6.2 Intuition

In practice, DCS has the effect of down-weighting constraints with large errors. Close to the zero-error configuration, DCS behaves like a normal squared kernel without any scaling. As the error increases, DCS scales the information matrix gradually.

With non-linear problems such as those involving orientations, the linear approximation in Equation (2.4) is poor if the initial estimate is far away from the correct solution. DCS mitigates the impact of a poor initial guess as it optimizes the problem while down-weighting constraints with large errors. The down-weighted constraints are those whose estimates are far away from the predicted measurements.

6.3 Experimental Evaluation

We have evaluated DCS on both real and simulated datasets, which were originally evaluated with CM (Grisetti et al., 2012). These include Victoria Park with range and bearing measurements and simulated 2D and 3D Manhattan world datasets with point features. In all experiments, we used the odometry as the initial guess for the

The datasets used in this chapter are publicly available and can be downloaded from http://www.informatik.uni-freiburg.de/~agarwal/resources/datasets-icra14-dcs.tar.gz.

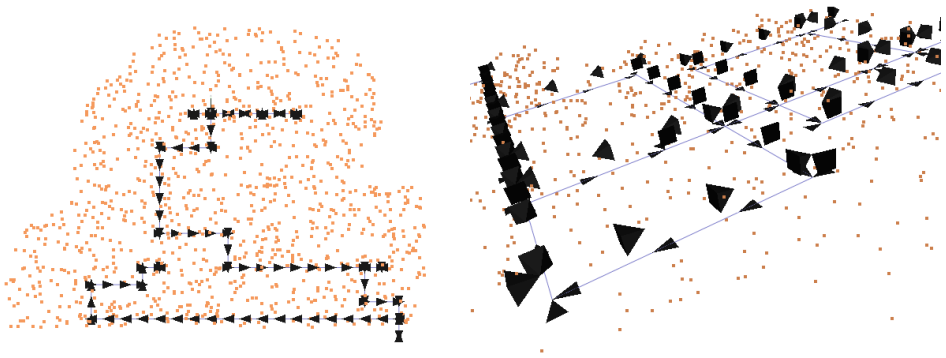


Figure 6.2: Simulated 2D (left) and 3D (right) datasets used by us and Grisetti et al. (2012) for evaluating condensed measurements. The robot is driven around in a Manhattan world and measurements to point features are recorded. The point features are shown in orange.

optimization. Figure 6.3(b) shows the initialization for a simulated dataset. To obtain a baseline comparison, we used ground truth initialization followed by the method under investigation. All errors reported with DCS were computed without the scaling function. Otherwise, DCS would report errors less than the global minimum after convergence.

DCS has one free parameter Φ, which influences the scaling variable s, In all experiments, we set $\Phi = 10$ unless otherwise stated, but the optimization works on a wide range of values for Φ. This experimental evaluation is designed to show the positive effect that DCS has on the computed solution in case of bad initial estimates. We show both, quantitative and qualitative benefits. We also illustrate the effect of the parameter Φ on the optimization process.

6.3.1 Victoria Park Dataset

The original Victoria Park dataset contains range-bearing observations of trees, which are used as point landmarks. It contains a total of 151 landmarks observed from 6,969 poses. This high pose to landmark ratio makes the problem challenging to converge for batch methods as illustrated in Figure 6.4. The batch method with Gauss-Newton without DCS seems to converge to the correct solution as shown in Figure 6.4(b), but a more detailed analysis reveals that this is not the case. Figures 6.4(d) and 6.4(f) show enlarged parts for the solution obtained by batch methods. Non-existing loops appears in the odometry chain, which corresponds to a local minima in the optimization process. Figures 6.4(e) and 6.4(g) show the correct results obtained with DCS. This correct result without the small loops can also be verified by incrementally optimizing the graph which typically comes at an increased overall computational cost.

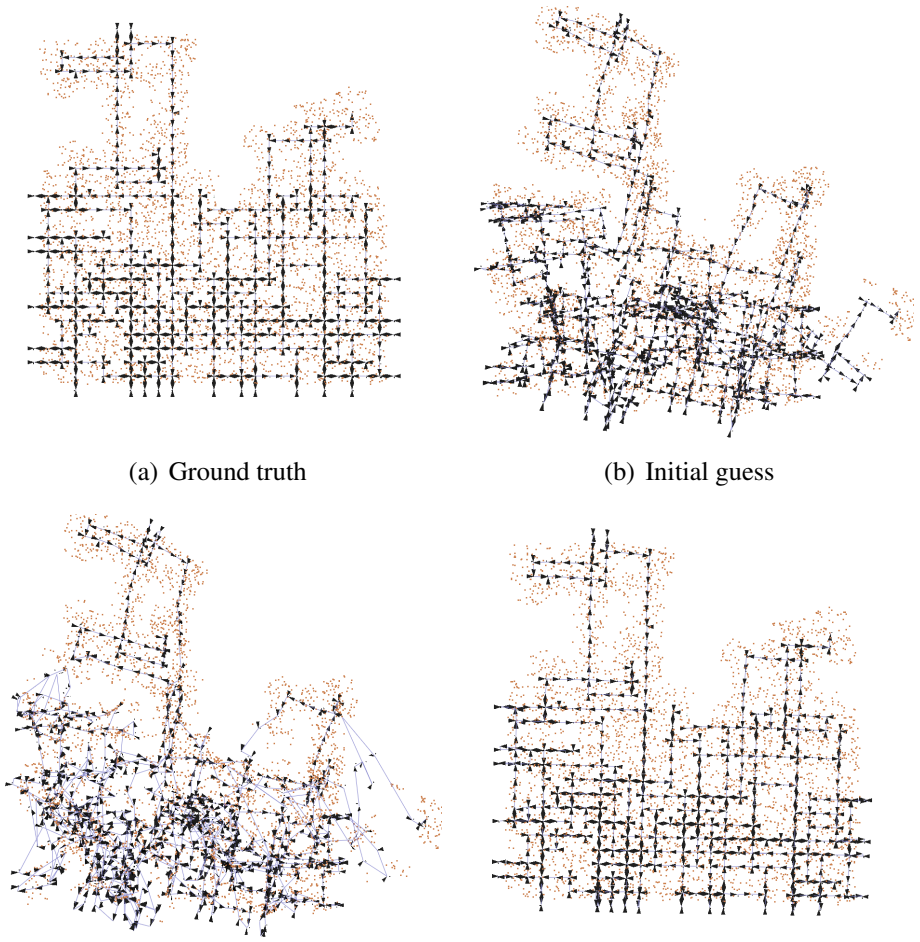
As depicted in Table 6.1, The total error of the solution with DCS is 389.78 compared to 30,607.16 with Gauss-Newton, 13,319.25 with Dog-Leg and 87,147.58 with LM. The

Table 6.1: The results obtained for the Victoria Park dataset using different optimization methods (batch mode). DCS and CM converge to the correct global minimum while all other methods converge to a local minimum.

Method	Resulting Error
CM	389.00
DCS	389.78
Gauss-Newton	30,607.16
Gauss-Newton with Dog-Leg	13,319.25
Levenberg-Marquardt	87,147.58

Table 6.2: DCS converges to the correct solution for a range of values for the parameter $\Phi \in [0.1, 20]$. $F_{\text{robust-DCS}}$ is the χ^2 error computed with the robust kernel. $F_{\text{robust-DCS}}$ represents the robust χ^2 error and hence cannot be directly compared to F_{CM} as they use different error function. We run a few iterations of DCS setting all $s_{ij} = 1$ which is equivalent to running gauss-newton iterations with a squared kernel after the optimization has converged and the resulting χ^2 error is reported as F_{DCS}.

Φ	F_{DCS}	$F_{\text{robust-DCS}}$
0.1	389.78	37.01
1.0	389.78	79.97
2.0	389.78	86.97
3.0	389.78	94.10
4.0	389.78	128.17
5.0	389.78	135.34
6.0	1,581.72	513.13
7.0	1,009.15	145.09
8.0	1,009.15	148.74
9.0	1,009.15	152.87
10.0	1,009.15	157.51
20.0	1,009.15	804.95



(a) Ground truth

(b) Initial guess

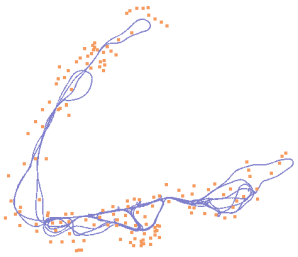
(c) Solution with Levenberg-Marquardt after 100 iterations.

(d) Solution with dynamic covariance scaling after 15 iterations.

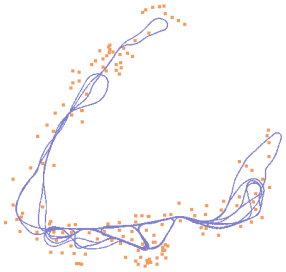
Figure 6.3: A simulated robot (black triangles) equipped with a stereo camera moves in a grid world (odometry edges in blue) and observes features (orange squares). The top row shows the ground truth and the initialization. Levenberg-Marquardt fails to compute the optimal solution even after 100 iterations, while dynamic covariance scaling is able to obtain a close-to-optimal solution within 15 iterations.

solution obtained with DCS is similar in quality and final error compared to the CM approach (Grisetti et al., 2012). Table 6.2 shows that DCS converges to the correct solution for a wide variety of Φ. $\mathrm{F}_{\text{robust-DCS}}$ is the χ^2 error computed with the robust kernel. Note that the $\mathrm{F}_{\text{robust-DCS}}$ cannot be directly compared to F_{CM} as they use different error function. Thus, we run a few iterations of DCS setting all $s_{ij} = 1$. This results in using the identical squared error function as none of the constraints are scaled. It can also be interpreted as running GN with the initialization computed by DCS. In addition to being robust to initialization, in Chapter 5 we also showed that DCS could reject a significant

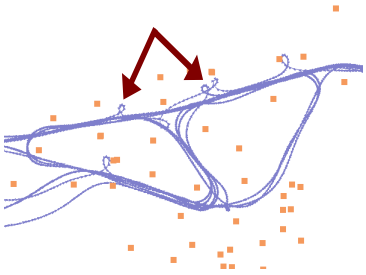
(a) Initialization of Victoria Park dataset with odometry



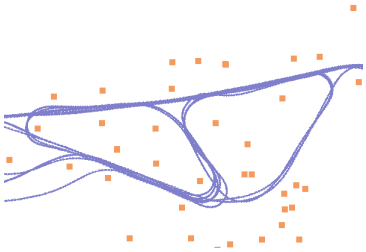
(b) GN (batch)



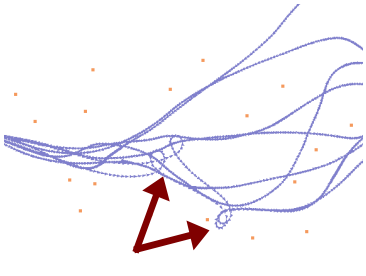
(c) DCS (batch)



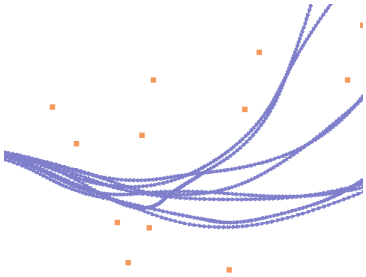
(d) GN (batch): zoom A



(e) DCS (batch): zoom A



(f) GN (batch): zoom B



(g) DCS (batch): zoom B

Figure 6.4: Optimization of the Victoria Park dataset with range-bearing measurements. The batch solution without DCS converges to the wrong solution. The errors in the robot poses are indicated by small loops in the odometry chain. These are not present when we used DCS. The batch solutions have a total error of 30,607.16 compared to an error of 389.78 with DCS for Φ between 0.1 and 5.

Table 6.3: Summary of the comparison experiments between LM, CM, and DCS.

Dataset	#constraints	Sensor model	F_{init}	F_{ideal}	F_{LM}	F_{CM}	F_{DCS}	#DCS-iters
A (2D)	1229	Cartesian	25137.90	1706.69	1706.69	1709.69	1706.76	5
B (2D)	10223	Cartesian	366551.00	18079.25	18079.25	18079.25	18079.39	5
C (2D)	105399	Cartesian	1.26742e+09	205207.54	205207.54	205206.32	205206.35	7
D (2D)	534688	Cartesian	1.79237e+10	1056677.58	1056677.58	1056677.58	1056677.58	10
E (3D)	226	depth	4706.70	116.91	116.91	116.91	116.91	6
E (3D)	226	disparity	6300.35	115.77	115.77	115.77	115.77	6
F (3D)	1809	depth	4.22496e+06	2988.96	2988.96	2988.96	3275.75	20
F (3D)	1809	disparity	1.40376e+07	2936.47	8038.63	2936.47	4309.50	20
G (3D)	19267	depth	1.72095e+11	43531.55	**16418112.01**	43531.54	43628.92	10
G (3D)	19267	disparity	4.53128e+11	43499.34	**10181039.20**	43499.35	43968.83	15
H (3D)	96659	depth	3.67085e+13	260937.23	**4547959956.76**	260937.23	261210.85	17
H (3D)	96659	disparity	2.42777e+12	261054.82	**1051509415.61**	261008.57	3172216.34	39

number of data association outliers in the Victoria Park dataset.

6.3.2 Simulation Results without Outliers

These experiments are designed to show that our approach is more robust with respect to the inital guess compared to Levenberg-Marquardt (LM) while performing similarly to the condensed measurement approach. Note, that the behavior of LM depends on the initial Lambda value. We use the default Lambda values provided by g2o for all experiments with LM. The 2D simulated datasets contain planar range-bearing measurements. The 3D simulated datasets contain depth, disparity, and range-bearing sensor modalities. These were simulated using the `g2o_simulator2d` and `g2o_simulator3d` applications.

Table 6.3 summarizes the result of experiments on the simulated datasets showing the type of dataset (2D or 3D), its size in terms of robot poses, number of landmarks, number of constraints, as well as the measurement model and the final χ^2 errors. The measurement model "Cartesian" describes a sensor that is capable of measuring the $(\Delta X, \Delta Y)$-position of a landmark in the reference frame of the observer. "Depth" refers to a sensor that measures the depth of points in an image plane. Finally, "disparity" refers to a stereo camera model. F_{init} represents the total χ^2 of the initialization, which is performed by composing the odometry measurements. The landmarks are then initialized using the first pose-landmark constraint. F_{ideal} is obtained by running Levenberg-Marquardt (LM) starting with the ground truth solution as the initial guess, i.e., running LM on the perfect initialization. F_{ideal} will form our baseline comparison for correctness of the solution. F_{LM} is the final error after running LM on the odometry based initialization. F_{CM} is the result obtained by the method of Grisetti et al. (2012) followed by LM. F_{DCS} represents the final error of the DCS solution. The last column displays the total number of iterations required by DCS.

Table 6.3 shows that by using DCS the optimization always converges to the correct solution. LM can solve the smaller 3D datasets but as the problem size increases, it is unable to reach the correct solution. The significant examples are those, where LM fails to converge to the correct solution but DCS does. These include the larger 3D depth and

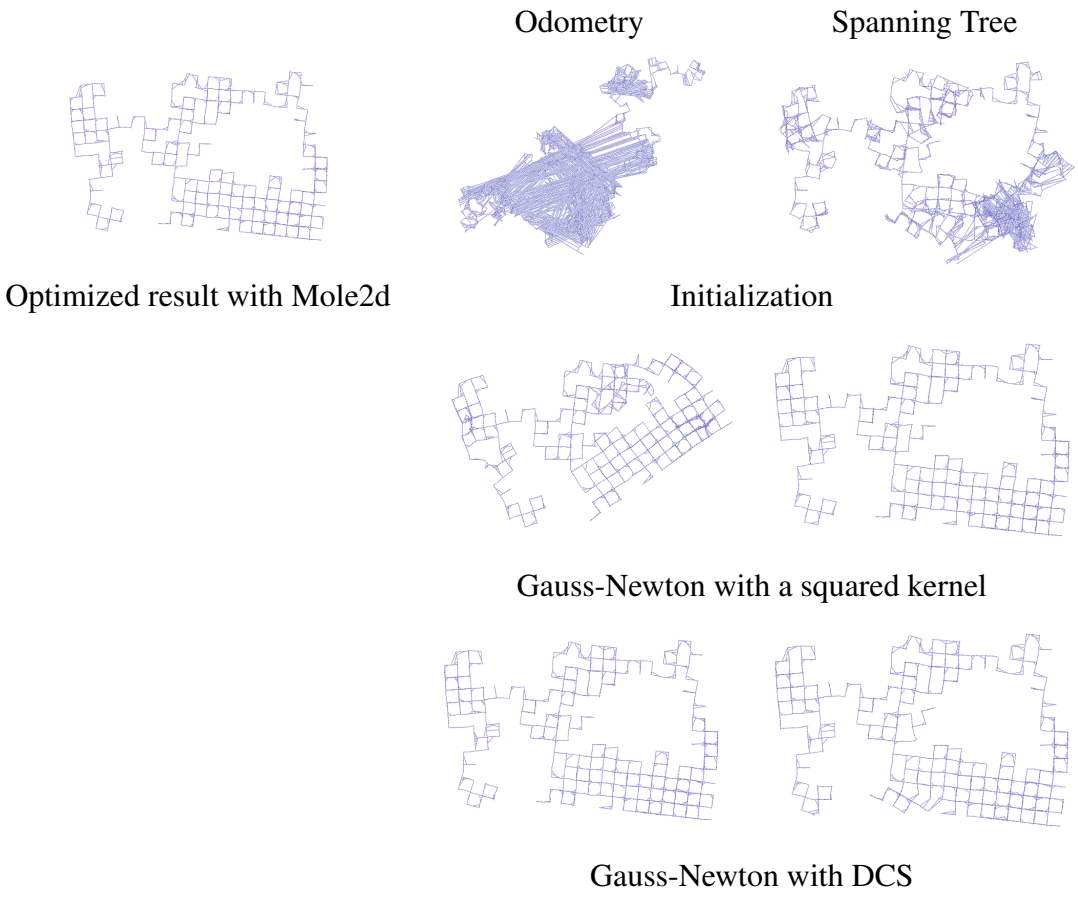


Figure 6.5: Analysis of dataset m3500a. DCS converges to the correct global minimum, similar to Mole2d, without the need of a separate initialization method while the use of a squared kernel reaches a local minimum.

disparity-based datasets. Note that the CM approach of Grisetti et al. (2012) followed by LM always converges to the correct solution as DCS does. DCS, however, has the advantage over CM to not require an initialization technique that is different from the optimization method. In addition, DCS can also deal with data association outliers while CM cannot.

We further evaluated the robustness of DCS on the increased noise Manhattan3500 datasets released with Mole2d (Carlone and Censi, 2012). These datasets were generated by Carlone and Censi (2012) using the same Manhattan3500 pose-graph released by Olson et al. (2006) but by re-sampling the edges in the pose-graph with an increase in the noise. This results in a harder optimization problem. As these graphs do not contain outliers, the position of the nodes may be initialized using either the odometry constraints or a spanning tree. The spanning tree initialization typically results in a better initialization but relies on the pose-graph to be free of outliers. The ground truth and resulting initializations based on odometry and spanning tree for m3500a, m3500b and

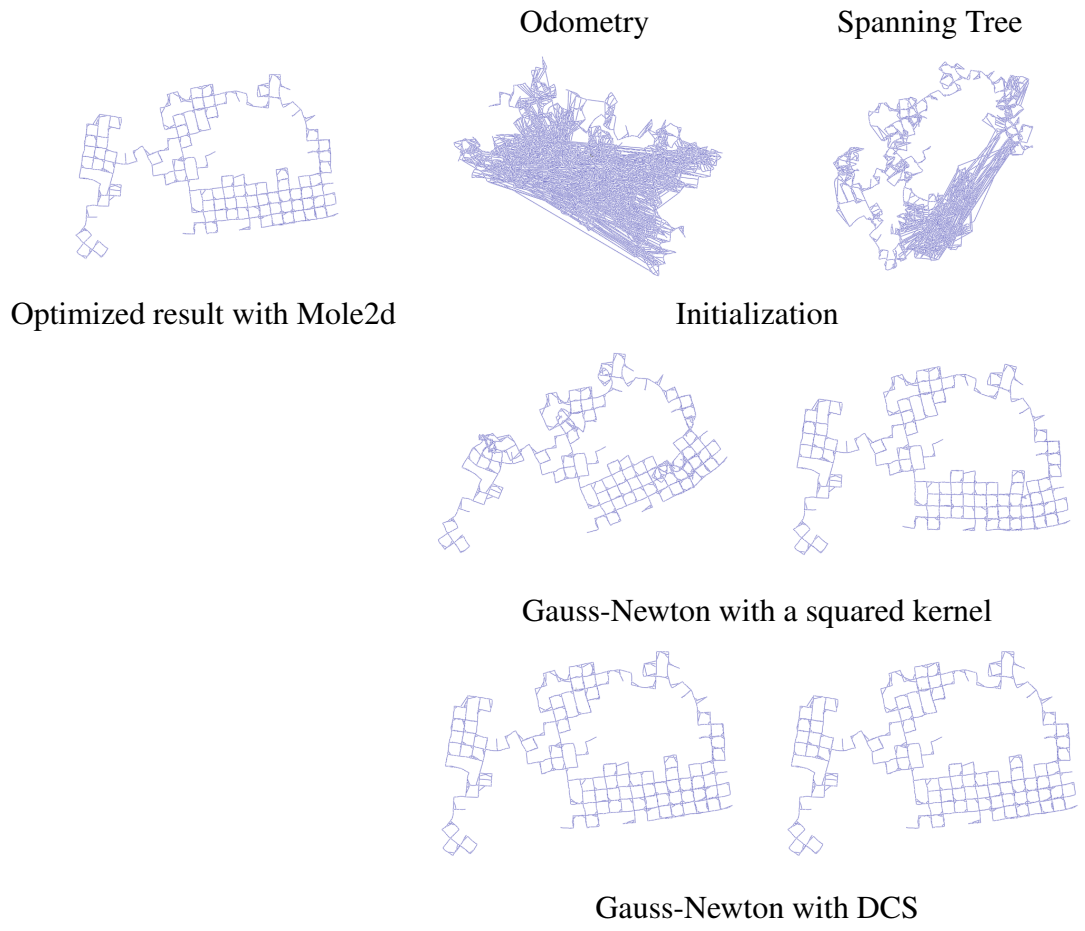


Figure 6.6: Analysis of dataset m3500b. DCS converges to the correct global minimum, similar to Mole2d, without the need of a separate initialization method while the use of a squared kernel reaches a local minimum.

m3500c are illustrated in the top rows of Figures 6.5, 6.6 and 6.7 respectively.

DCS is able to converge to the same global minima as `MOLE2D` for both m3500a and m3500c. This is illustrated in the last rows of Figure 6.5, 6.6 and 6.7, while the use of standard Gauss-Newton with a squared kernel reaches only a local minimum. These datasets are interesting as they clearly show that DCS improves the optimization process when the initialization is poor or the front-end produces measurements with high noise.

6.3.3 Influence of Φ on the Optimization

The next experiment is designed to illustrate the effect of the parameter Φ on the optimization process. In all experiments before, we set Φ to 10. It turns out that DCS converges for a large range of values for Φ, but it has an impact in the number of iterations needed. Figure 6.8 illustrates this behavior. The number of iterations required to reach

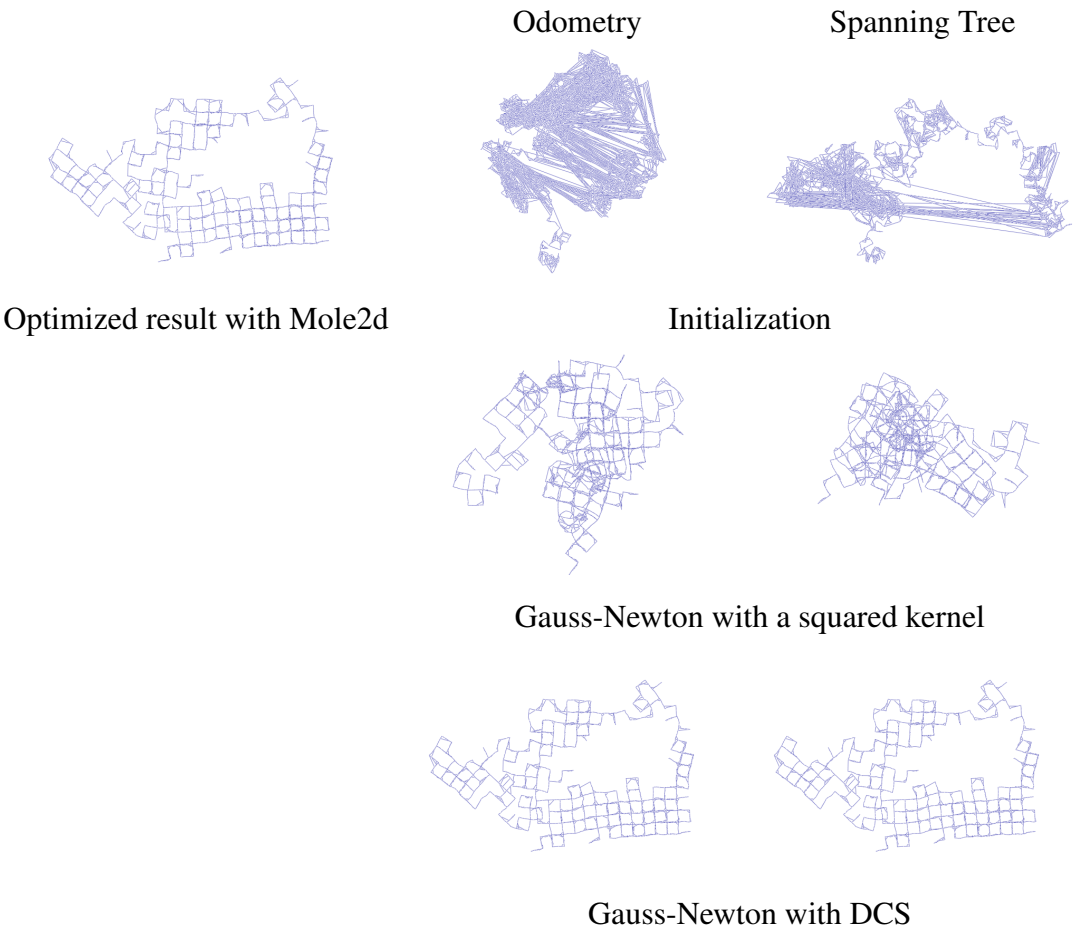


Figure 6.7: Analysis of dataset m3500c. DCS converges to the correct global minimum, similar to Mole2d, without the need of a separate initialization method while the use of a squared kernel reaches a local minimum.

the global minimum decreases with an increase in Φ. This does not scale arbitrarily since as $\Phi \to \infty$, DCS behaves like the original squared kernel.

The computation time per iteration of all optimization methods used here is dominated by the sparse Cholesky factorization. The only overhead that DCS creates over LM is computing the scaling coefficient in each error function. This does not lead to any measurable increase in runtime.

6.3.4 Simulation Results in the Presence of Outliers

With the final experiment we want to show that DCS can deal with a substantial number of outliers, even for the more difficult optimization problems. Although the focus of this chapter is not about robustness with respect to outliers, we evaluated how DCS is able to reject outliers in cases where LM can not optimize the problem, even with zero outliers.

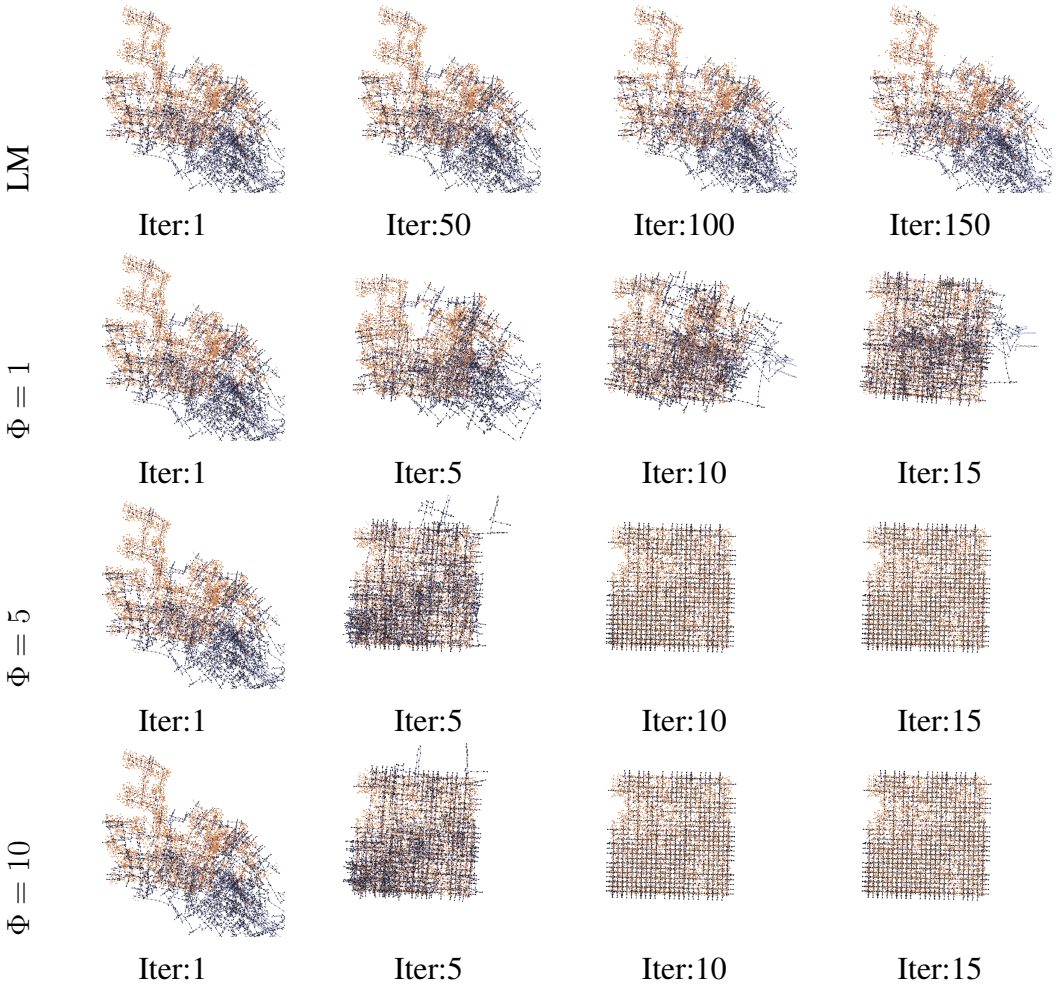


Figure 6.8: Effect of Φ on the optimization process. All values of $\Phi \in 1, 5, 10$ are capable of optimizing the pose graph. By increasing the values of Φ from 1 to 10, lesser number of optimization steps are required. The above pose-graph could not be solved using standard Levenberg-Marquardt. The time required for each iteration of LM is similar to that of DCS as the time is dominated by the sparse matrix factorization.

Table 6.4: DCS performance with a different numbers of outliers (percentage of outliers w.r.t. the total number of observations). Y=right solution; N=wrong minimum.

Dataset	Sensor model	#Constraints	5%	10%	25%	30%
C (2D)	Cartesian	105399	Y	Y	N	N
D (2D)	Cartesian	534688	Y	Y	N	N
G (3D)	depth	19267	Y	Y	Y	N
G (3D)	disparity	19267	Y	Y	Y	N

Table 6.4 summarizes our results when adding up to 30% outliers to the simulated datasets. We created the outliers by adding wrong constrains randomly between robot and landmark positions. The last four columns represent error percentage of false constraints added. "Y" represents a success in converging to the correct solution and "N" represents failure. Note than LM was unable to converge to the correct solutions for the depth and disparity 3D-datasets G, even without outliers. For these challenging datasets, DCS succeeds even with 25% outlier constraints.

6.4 Summary

The initial guess can have a substantial impact on the solution found by non-linear error minimization methods such as Gauss-Newton or Levenberg-Marquardt. In this chapter, we evaluated the dynamic covariance scaling method on SLAM-graphs with poor initial estimates. Our experiments suggest that dynamic covariance scaling is more resilient and robust to bad initial configurations compared to the standard use of Levenberg-Marquardt and Gauss-Newton for optimization. We also showed that compared to other M-estimators such as Cauchy and Huber, it is more robust to poor initialization and outliers. DCS can solve complex non-linear problems without the need of additional initialization mechanisms and without increasing the computational cost per iteration. In contrast to other methods that first aim at finding a good initialization to seed the optimization or that remove outliers before optimizing, our method is more elegant as it does not require an additional pre-processing step.

Chapter 7

Localization using Google Street View

Accurate metrical localization is one of the central challenges in mobile robotics. Many existing navigation methods perform localization as a pose estimation problem after building a map with the robot. Hence, in the previous chapters we outlined multiple robust robot mapping techniques. In this chapter, we present a novel approach to metric localization that does not require the construction of a new map when one already exists. It is able to leverage maps built for human navigation. The central idea is to use geotagged imagery on Google Street View as a source of global positioning. The only input of this approach is a stream of monocular camera images and odometry estimates. We quantified the accuracy of the method by running the approach on a robotic platform in a parking lot by using visual fiducials as ground truth. Additionally, we applied the approach in the context of personal localization in a real urban scenario by using data from a Google Tango tablet.

7.1 Introduction

So far in Chapters 4-6, we outlined different SLAM techniques that focus on map building while being robust to errors in place recognition and errors in sensor measurements. Such robust mapping and localization algorithms are of utmost importance for robot navigation tasks in completely unknown environments. The map allows the robot to plan paths, while the localization algorithm keeps the robot on the path to ensure that the robot reaches its destination. In the previous chapters, we show that our methods improve the general problem of simultaneously localizing a robot while building a map of the environment. In this chapter, we want to explore techniques which will allow to perform localization using back ground knowledge in the form of existing maps whenever possible. Specifically, we want to re-use maps for localization even if they were not designed for robots. This is because map building is a computationally expensive and time consuming process. If maps do not exist, we can use the algorithms developed in previous chapters, but if maps exist and a robot can re-use them, it will allow them to navigate without needing to

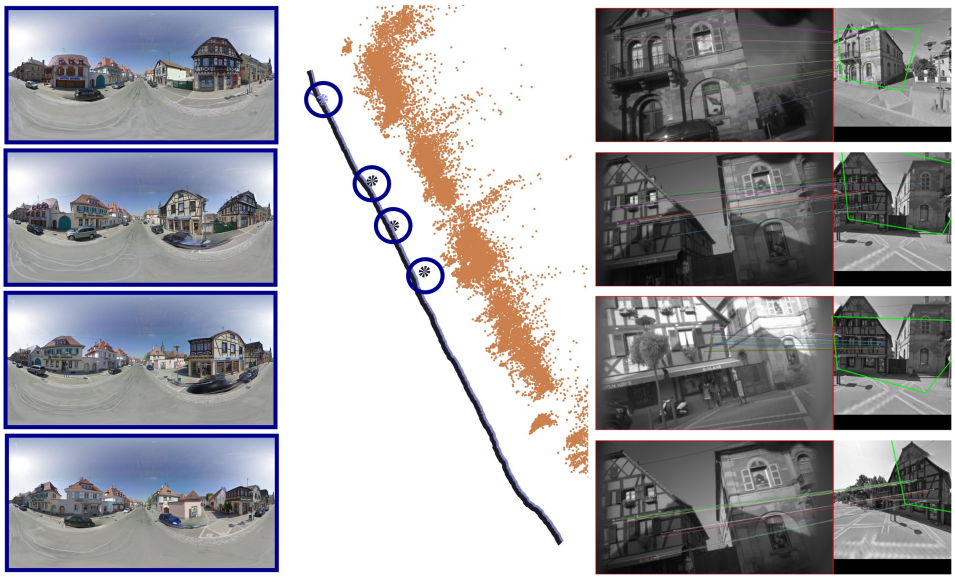


Figure 7.1: Localization of a moving camera from Street View panoramas. The location of the panoramas are 27 Rue du Mal Foch, Marckolsheim, France. Four panoramas shown in left are localized with respect to the camera trajectory (black) and estimated 3D points (orange). The images on the right show feature matching between Tango images and rectilinear views of the panorama.

explore the full environment first.

Our central idea is to leverage Google Street View as an abundant source of accurate geotagged imagery. In particular, our key contribution is to formulate localization as the problem of estimating the position of Street View's panoramic imagery relative to monocular image sequences obtained from a moving camera. With our approach, we can leverage Google's global panoramic image database, with data collected each 5-10m and continuously updated across five continents (Google Inc., 2012; Anguelov et al., 2010). To make this approach as general as possible, we only make use of a monocular camera and a metric odometry estimate, such as the one computed from IMUs or wheel encoders.

We formulate our approach as a non-linear least squares problem of two objectives. In the first, we estimate the 3D position of the points in the environment from a short monocular camera trajectory. The short trajectories are motivated by limiting the computation time, restricting the estimation problem and the presence of abundant panoramic imagery. In the second, we find panoramas that match the images and compute their 6DOF transformation with respect to the camera trajectory and the estimated 3D points. As the GPS coordinates of the panoramic images are known, we obtain estimates of the camera positions relative to the global GPS coordinates. Our aim is not to accurately model the environment or to compute loop closures for improving reconstruction. Our approach can be considered as a complement of GPS systems, which computes accurate

positioning from Street View panoramas. For this reason, we tested our method on a Google Tango smartphone in two kinds of urban environments, a suburban neighborhood in Germany and a main road of a village in France. Additionally, we quantify the accuracy of our technique by running experiments in a large parking lot with ground truth computed from visual fiducials. In the experiments, we show that with our technique we are able to obtain sub-meter accuracy and localize in urban environments.

The rest of this chapter is organized as follows. We fist provide an over view of related work in Section 7.2. Next in Section 7.3, we outline our method for localizing street view panoramas against monocular image stream. Finally, in Section 7.4 we evaluate our algorithm on a robot as well as on Google's Tango device.

7.2 Related work

There exist previous literature about using Street View imagery in the context of robotics and computer vision. Majdik et al. (2013) use Street View images to localize a Micro Aerial Vehicle by matching images acquired from air to Street View images. Their key contribution is matching images with strong view point changes by generating virtual affine views. Their method only solves a place recognition problem. We, on the other hand, compute a full 6DOF metrical localization on the panoramic images. In (Majdik et al., 2014), they extended that work by adding 3D models of buildings as input to improve localization. Other researchers have matched Street View panoramas by matching descriptors computed directly on it (Torii et al., 2011). They learn a distinctive bag-of-word model and use multiple panoramas to match the queried image. Those methods provide only topological localization via image matching. Related to this work is the topic of visual localization, which has a long history in computer vision and robotics, see (Fuentes-Pacheco et al., 2012) for a recent survey. Various approaches have been proposed to localize moving cameras or robots using visual inputs (Cummins and Newman, 2009; Davison et al., 2007; Agrawal and Konolige, 2008; Klingner et al., 2013; Torii et al., 2009). Our work is complimentary to such place recognition algorithms as these may serve as a starting point for our method. Topological localization or place recognition serves as a pre-processing step in our pipeline. We use a naive bag-of-words based approach, which we found to be sufficient for place recognition. Any of the above-mentioned methods can be used instead to make the place recognition more robust.

Authors have also looked into localizing images in large scale metrical maps built from structure-from-motion. Irschara et al. (2009) build accurate point clouds using structure from motion and then compute the camera coordinates of the query image. In addition, they generate synthetic views from the dense point cloud to improve image registration. Zhang and Kosecka (2006) triangulate the position of the query image by matching features with two or more geotagged images from a large database. The accuracy of their

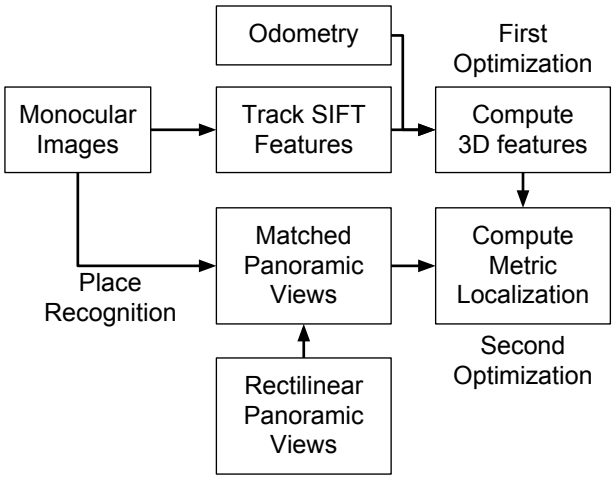


Figure 7.2: Flowchart illustrating various modules in our pipeline.

method depends on the density of the tagged database images. Sattler et al. (2012b) also localize query images in a 3D point cloud. Instead of using all the matching descriptors, they use a voting mechanism to detect robust matches. The voting scheme enables them to select 3D points which have support from many database images. This approach is further improved in (Sattler et al., 2012a) by performing a search around matched 2D image features to 3D map features and vice versa. Zamir and Shah (2010) build a dense map from 100,000 Google street view images and then localize query images by a GPS-tag-based pruning method. They provide a reliability score of the results by evaluating the kurtosis of the voting based matching function. In addition to localizing single images, they can also localize a non-sequential group of images.

Unlike others, our approach does not rely on accurate maps built with a large amount of overlapping geotagged images. As demonstrated by the experiments, our approach requires only a few panoramas for reliable metric localization with sub-meter accuracy.

7.3 Method

In this section we outline the technical details for using Google's Street View geotagged imagery as our map source for robot localization. Our goal is not to built large scale accurate maps. Instead, we want to approximately estimate the 3D position of the features relative to the camera positions and then compute the rigid body transformation between the Street View panoramas and the estimated points. This allows us to compute the GPS positions of the camera position in global GPS coordinates. Our current implementation works offline. The flowchart shown in Figure 7.2 illustrates the workflow between the various modules.

7.3.1 Tracking Features in an Image Stream

The input of our method is an image stream acquired from a monocular camera. We define $\mathcal{S} = (s_1, \ldots, s_S)$ as a sequence of S images. A sequence is implemented as a short queue that consists only of the last few hundreds frames acquired by the camera. An image s_i is a 2D projection of the visible 3D world, through the lens, on a camera's CCD sensor. For estimating the 3D position of the image points, we need to collect bearing observations from several positions as the camera moves.

We take a sparse features approach for tracking features in the stream of camera images. For each image s_i, we extract a set of keypoints computed by using state-of-the-art robust feature detectors, such as SIFT (Lowe, 2004). A description $d \in \mathcal{D}_i$ is computed from the image patch around each keypoint. $\mathcal{F}_i$ is the set of keypoints and descriptors and is denoted as the feature set.

Each time a new image arrives, we find feature correspondences between s_i and s_{i-1}. We compute neighbor matches using FLANN (Muja and Lowe, 2012) between all elements of $\mathcal{D}_i$ and $\mathcal{D}_{i-1}$. We consider a match valid if the distance to the best match is 0.7 times closer than the second best (Lowe, 2004). These correspondences only consider closeness in descriptor space. We employ a homography constraint to also consider the keypoint arrangement between two images. We use the keypoints of the matched descriptors for a RANSAC procedure that computes the inlier set for the perspective transformation between the two images. We call a track $\mathcal{T}_j$, the collection of all the matched keypoints relative to the same descriptor over the consecutive image frames $\mathcal{S}$. A track is terminated as soon as the feature cannot be matched in the current image. For an image stream $\mathcal{S}$, we collect the set of tracks $\mathcal{T}_\mathcal{S}$ consisting of the features $\mathcal{F}_\mathcal{S}$.

Note that tracks have different length. Some keypoints are seen from many views, while others are seen from few. Intuitively, long tracks are good candidates for accurate 3D point estimation as they have longer baselines. We only perform feature matching across sequential image frames. No effort is spent on matching images which are not sequential: this work does not make any assumption on the motion of the robot, on the visibility of the environment or on the existence of possible loops.

7.3.2 Non-Linear Least Squares Optimization for 3D Point Estimation

The next step is to compute 3D points from the tracks $\mathcal{T}_\mathcal{S}$. In our system, we have rigid body odometric constraints between consecutive camera poses $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$, associated to frame s_i and s_{i+1}. Our method is agnostic to the kind of odometry: it can be computed by integrating IMUs, wheel encoders, or by employing an IMU-assisted visual odometry. In our problem formulation, we consider the monocular camera calibrated and all the intrinsic parameters known.

Each keypoint in track $\mathcal{T}_j$ corresponds to a 3D point $\mathbf{y}_j$ observed in one of the images with pixel coordinates u, v. If we consider a pinhole camera model, the camera matrix $\mathbf{C}$ projects a point $\mathbf{y}_j$ into the camera frame:

$$\mathbf{C} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{7.1}$$

The direction vector

$$\mathbf{d} = \mathbf{C}^{-1}[u, v, 1]^T, \tag{7.2}$$

can be interpreted as the direction of $\mathbf{y}_j$ with respect to the camera center. Then, we compute the elevation and bearing angles:

$$\theta = \arccos\left(\frac{d_z}{\sqrt{d_x^2 + d_y^2 + d_z^2}}\right) \tag{7.3}$$

$$\varphi = \arctan\left(\frac{d_y}{d_x}\right) \tag{7.4}$$

A least squares minimization problem can be described by the following equation:

$$\begin{aligned} \mathbf{F}(\mathbf{x}, \mathbf{y}) &= \sum_{ij} \mathbf{e}_{ij}(\mathbf{x}, \mathbf{y})^T \mathbf{\Lambda}_{ij}^{-1} \mathbf{e}_{ij}(\mathbf{x}, \mathbf{y}) \\ &+ \sum_{k} \mathbf{e}_{k,k+1}(\mathbf{x})^T \mathbf{\Sigma}_{k}^{-1} \mathbf{e}_{k,k+1}(\mathbf{x}) \end{aligned} \tag{7.5}$$

Here

- $\mathbf{x} = (\mathbf{x}_1^T, \ \dots \ , \mathbf{x}_n^T)^T$ is a vector of monocular camera poses, where each $\mathbf{x}_i$ represents a 6DOF pose.
- $\mathbf{y} = (\mathbf{y}_1^T, \ \dots \ , \mathbf{y}_m^T)^T$ is a vector of 3D points in the world associated to the tracked features.
- $\mathbf{e}_{ij}(\mathbf{x}, \mathbf{y})$ is a vector error function that computes the distance between a measurement prediction $\hat{\mathbf{z}}_{ij}(\mathbf{x}, \mathbf{y})$ and a real measurement $\mathbf{z}_{ij} = [\theta_{ij}, \phi_{ij}]$. The error is $\mathbf{0}$ if $\mathbf{z}_{ij} = \hat{\mathbf{z}}_{ij}$, that is when the measurement predicted via $\hat{\mathbf{z}}_{ij}(\mathbf{x}, \mathbf{y})$ from the states $\mathbf{x}_i$ and $\mathbf{y}_j$ is equal to the real measurement.
- $\hat{\mathbf{z}}_{ij}(\mathbf{x}, \mathbf{y})$ computes the bearing and azimuthal angles from camera pose $\mathbf{x}_i$ to feature $\mathbf{y}_j$ in the camera frame.
- $\mathbf{\Lambda}_{ij}^{-1}$ represents the information matrix of a measurement that depends on the state variables in $\mathbf{x}$.

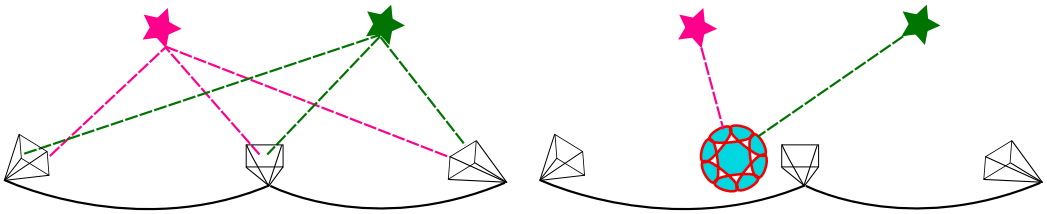


Figure 7.3: Optimization problem for estimating the position of the features, $\mathbf{y}$ shown as stars, and the camera positions $\mathbf{x}$ shown as frustums. The dotted lines represent bearing constraints while the solid black line represents the odometry constraint. The right image shows the optimization problem for computing the position of the panorama $\mathbf{p}$ from the computed 3D points.

- $\mathbf{e}_{k,k+1}(\mathbf{x})$ is a vector error from the predicted odometry measurements.
- $\mathbf{\Sigma}_k^{-1}$ represents the information matrix of the odometry.

We initialize the camera position $\mathbf{x}$ with odometry and the feature positions $\mathbf{y}$ by triangulation. We employ the optimization framework g2o (Kümmerle et al., 2011) as our non-linear least squares solver. First, we solve Equation (7.5) by keeping $\mathbf{x}$ fixed:

$$\mathbf{y}^* = \underset{\mathbf{y}}{\operatorname{argmin}} \mathbf{F}(\mathbf{x}, \mathbf{y}) \tag{7.6}$$

This results in an improved estimation of $\mathbf{y}$. Second, we perform a full joint optimization of all the estimated 3D points $\mathbf{y}$ and camera poses $\mathbf{x}$.

$$(\mathbf{x}^*, \mathbf{y}^*) = \underset{\mathbf{x},\mathbf{y}}{\operatorname{argmin}} \mathbf{F}(\mathbf{x}, \mathbf{y}) \tag{7.7}$$

The use of RANSAC helps improve the feature correspondences but does not guarantee an absence of outliers. Therefore, the robust methods developed in the previous chapters are used to improve the robustness against such errors. We use Dynamic Covariance Scaling kernel outlined in Chapters 5 and 6 to improve convergence and to handle wrong data associations.

Note that we are not aiming at an accurate reconstruction of the environment. In our approach, we only perform data association between sequential images as we do not compute loop closures or perform large baseline feature triangulation. There may be situations where a track is broken due to occlusions or changes in the viewpoint. We do not try to merge tracks in such scenarios. This is avoided for the process to be less computationally demanding. Doing a full bundle-adjustment will definitely help in a better reconstruction of the environment but that is not the goal of our work.

7.3.3 Matching of Street View Panoramas with Camera Images

Google Street View can be considered as an online browsable dataset consisting of billions of street-level panoramic images acquired all around the world (Google Inc., 2012). It is of key importance that each image is geotagged with a GPS position. This position is highly accurate and it is the result of a careful optimization at global-scale by Google (Klingner et al., 2013). In particular, Street View images are acquired by vehicles with a special apparatus consisting of cameras mounted around a spherical mounting. All camera images are stitched together to form a spherical panoramic image represented via a *plate carrée* projection. This results in a high quality image often exceeding 20M pixels resolution for each panorama.

Google provides public APIs for requesting virtual camera views of a given panorama. These views are rectilinear projection of the spherical panorama with a user selected field-of-view, orientation and elevation angle. Rectilinear views can be considered as undistorted images from a pinhole camera, free of distortion. A panorama can be selected via its GPS position or its ID. An example of a panorama acquired from Wall Street, New York, is illustrated in Figure 7.4. For robustness, we extract rectilinear horizontal overlapping images. The overlapping region aids in matching at image boundaries. We do not use the top and the bottom image as it often contains only sky and floor.

In order to match panoramas with monocular camera trajectories we first need a candidate set of panoramas. In our approach we rely on an inaccurate GPS sensor to download all panoramic images in a certain large radius of approximately 1 km. The motivation behind this approach is that a robot will roughly know which neighborhood or city it is operating in. First, we collect all the rectilinear views from the panoramic images $\mathcal{P}$ and build a bag-of-words image retrieval system (Fei-Fei and Perona, 2005). We compute SIFT keypoints and descriptors $\mathcal{F}_{\mathcal{P}}$ for all rectilinear panoramic views in $\mathcal{P}$ and group them with k-means clustering to generate a visual codebook. Once the clusters are computed and we describe each image as histograms of visual words, we implement a TF-IDF histogram reweighing. For each camera image, we compute the top K images from the panoramic retrieval system, which have the highest cosine similarity. This match can be further improved by restricting the search within a small radius around the current GPS location or from the approximate location received from cellular network towers. Second, we run a homography-based feature matching, similar to the one used for feature tracking in Section 7.3.1 to select the matching images from K. These matched images are used as the final candidate panoramic images used to compute the global metric localization explained in the next section.

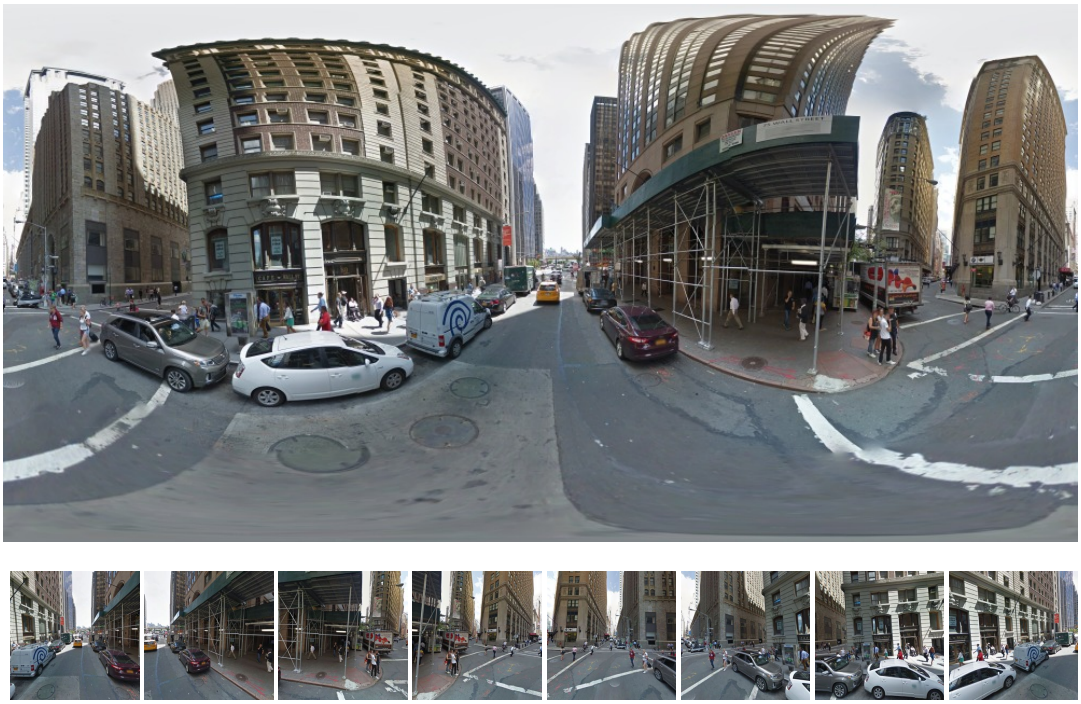


Figure 7.4: A panorama downloaded from Street View (top). Extracted rectilinear views (bottom). Each image has a 90° field-of-view. These are considered pinhole cameras, free of distortion and they overlap horizontally to aid matching across image boundaries.

7.3.4 Computing Global Metric Localization

To localize in world reference frame, we compute the rigid body transformation between the moving camera imagery and the geotagged rectilinear panoramic views. We look for the subset of features $\mathcal{F}^* = \{\mathcal{F}_\mathcal{P} \cap \mathcal{F}_\mathcal{S}\}$ that are common between the monocular images $\mathcal{S}$ and the top K panoramic views. The 3D positions of $\mathcal{F}$ have been estimated using the methods in Section 7.3.2. We consider the rectilinear views as perfect pinhole cameras: the focal length f_x, f_y are computed from the known field-of-view; c_x, c_y is assumed to be the image center. We follow the same procedure of Section 7.3.2 for computing the azimuthal and bearing angles for each element of $\mathcal{F}^*$ using Equation (7.3) and Equation (7.4).

To localize the positions of the K panoramas from the feature positions $\mathbf{y}$, we formulate another non-linear least squares problem similar to Equation (7.5):

$$\mathbf{G}_1(\mathbf{p}, \mathbf{y}) \quad = \quad \sum_{ij} \mathbf{e}_{ij}(\mathbf{p}, \mathbf{y})^T \mathbf{\Lambda}_{ij}^{-1} \mathbf{e}_{ij}(\mathbf{p}, \mathbf{y}) \tag{7.8}$$

where

- $\mathbf{p} = (\mathbf{p}_1^T, \ \ldots \ , \mathbf{p}_{8\times m}^T)^T$ is a 6DOF vector of poses associated to the rectilinear views taken from m panorama images.

- $\mathbf{y}$ is the vector of the estimated 3D points.
- $\mathbf{e}_{ij}(\mathbf{p}, \mathbf{y})$ is the same error function defined for the optimization Equation (7.5). This is computed for all $\mathcal{F}^*$ between panorama view $\mathbf{p}_i$ and 3D points $\mathbf{y}_j$.
- $\mathbf{\Lambda}_{ij}^{-1}$ represents the information matrix of the measurement.

For robustness, we connect multiple views from the same panorama that are constrained to have the same position but a relative yaw offset of $90°$. The optimization problem becomes

$$\begin{aligned} \mathbf{G}_2(\mathbf{p}, \mathbf{y}) &= \mathbf{G}_1(\mathbf{p}, \mathbf{y}) \\ &+ \sum_k \mathbf{e}_{k,k+1}(\mathbf{p})^T \mathbf{\Sigma}_k^{-1} \mathbf{e}_{k,k+1}(\mathbf{p}) \end{aligned} \tag{7.9}$$

where $\mathbf{e}_{k,k+1}(\mathbf{p})$ is the error between two rectilinear views computed from the same panorama. The optimal value for $\mathbf{p}*$ can be found by solving:

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}}\, \mathbf{G}_1(\mathbf{p}, \mathbf{y}) \tag{7.10}$$

or alternatively by solving:

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}}\, \mathbf{G}_2(\mathbf{p}, \mathbf{y}) \tag{7.11}$$

After optimization, the panoramic views are in the frame of reference of the monocular camera trajectory $\mathbf{x}$. Now, it is trivial to compute the relative offset between the map and the panorama, hence computing precise global GPS coordinates of the camera images.

7.4 Experimental Evaluation

We evaluated our method in two different scenarios. In the first, we considered an outdoor parking lot area and placed visual fiducials for accurate ground truth. In the second, we used a Google Tango device in two different urban scenarios. The first scenario is in Freiburg, Germany where we personally uploaded panoramas acquired with mobile devices. This is required as Street View is only partially available in Germany. For the second scenario, we tested our technique on panoramas from Street View collected by Google in Marckolsheim, France. All of the panoramas used in these experiments are publicly available.

In our experiments, some of the panoramas were manually acquired with a cell phone and hence the panorama rig is not fixed. By optimizing the additional rig parameters we are more robust to small errors in the panorama building process. Additionally, we do not have any constraints between different panoramas collected from different places. Each panorama is independently optimized.

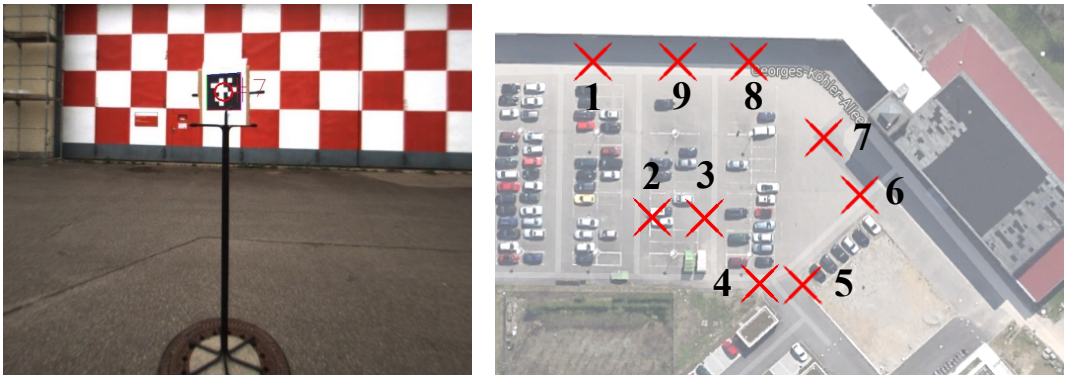


Figure 7.5: The left figure shows an example of an AprilTag placed above a manhole from where a panoramic image was acquired. The right figure illustrates an aerial view of the parking lot for the parking lot experiment. Red crosses highlighting the positions of the panoramas. The numbers represent the AprilTag ID.

7.4.1 Metric Accuracy Quantification

The parking lot experiment is designed to evaluate the accuracy of our method. It is full of dynamic objects and visual aliasing. Additionally, most of the structures and buildings are only on the far-away perimeter of the parking lot.

Using GPS as ground truth is not sufficient as our method aims at providing accurate estimations, potentially better than GPS accuracy. For reference, we collected spherical panoramas by using a smartphone, on visually distinct landmarks such as manholes. Then, as a ground truth we placed visual fiducials, namely AprilTags (Olson, 2011a) above the manholes from where the panoramas were acquired. AprilTags come with a robust detector and allow for precise 3D positioning. We use the tag family 36h11 and the open source implementation available from (Kaess, 2013). Figure 7.5 shows one such tag from the view of the camera with the tag detection and detected id superimposed on the image. Figure 7.5 also illustrates the aerial view of parking lot with the position from where the panoramas were generated (red crosses). The numbers represent IDs for each April Tag. To have a fine estimate of the panoramic image pose, we use non-linear least squares to optimize for the full 6D tags positions from the computed camera poses as illustrated in Figure 7.6.

For these experiments, we used a robot equipped with an odometry estimation system and a monocular 100° wide field-of-view camera. We performed 4 runs around all 9 AprilTags and 1 shorter run. In total we performed a total of 5 different runs in the parking lot. The position of panoramas and AprilTags are computed with respect to the camera positions. Tables 7.1 and 7.2 report the error between the computed pose of the panorama and the associated tag for the 5 runs. Figure 7.7 shows examples of feature matches between three panorama views and camera images. Each of the three images in Figure 7.7

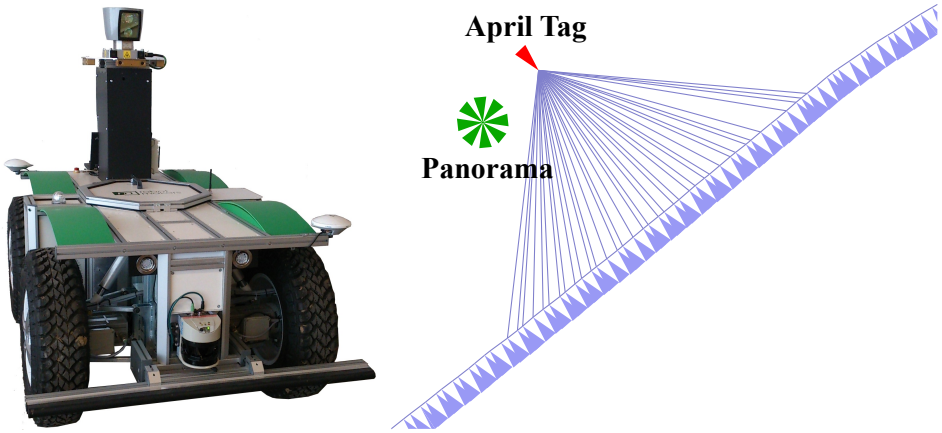
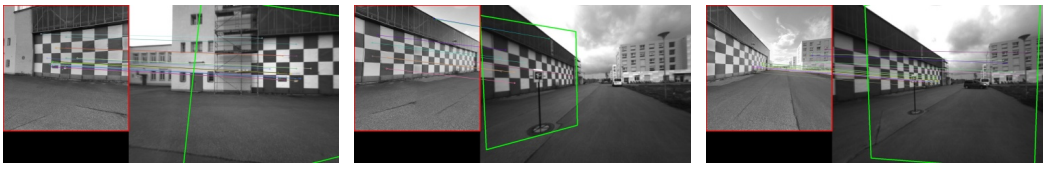


Figure 7.6: The left figure shows the robot used to conduct the parking lot experiments and the right figure illustrates the final monocular camera positions with the estimated position of the panorama and April tag in the parking lot.



(a) 135 (b) 180 (c) 225

Figure 7.7: Matching 3 views of the same panorama to monocular images. 90° field-of-view rectilinear projections and the corresponding feature matches for each view can be seen. Homography projection of the panoramas on the monocular image is shown in green.

show the matched features and homography of the rectified panorama projected on to the image acquired from the monocular camera.

The errors reported in Table 7.1 correspond to optimizing the individual views of the panoramas without any constraints among them. This corresponds to the optimization in Equation (7.10). That is, if two views of a panorama match at a certain place, we optimize them independent of each other. Table 7.2 reports errors when all the views of the panoramas are connected together. This corresponds to the optimization in Equation (7.11). Connecting views from the same panorama together improves the accuracy as can be seen from Figure 7.8. The system does not report localization results if the matching rectilinear views are estimated too far with respect to the current pose.

The panorama acquired from the position of tag id 5 and 7 is localized with least accuracy as most of the estimated 3D features are far way ($> 50m$). The panoramas acquired from the position of tags 10 and 8 are localized with the highest accuracy as the tracked features are relatively closer ($15m - 20m$).

1	2	3	4	5	6	7	8	9
6.00	3.14	0.99	1.65	5.36	2.94	1.22	0.53	1.29
0.72	1.07	-	9.62	4.91	2.00	5.53	0.70	1.28
-	-	-	-	-	3.35	0.74	1.07	4.03
-	-	0.37	1.03	3.92	5.34	2.07	0.39	1.07
-	0.47	0.58	0.37	12.38	5.89	1.53	0.55	2.59

Table 7.1: Error (in meters) between estimated pose of each individual panoramic view compared to the ground truth tag.

1	2	3	4	5	6	7	8	9
10.11	3.14	1.09	1.65	3.44	3.61	1.22	0.36	1.29
0.72	1.07	-	5.80	3.27	0.80	1.51	0.68	1.28
-	-	-	-	-	3.35	0.70	1.07	0.90
-	-	0.37	1.03	2.84	5.34	2.07	0.60	1.07
-	0.47	0.59	0.42	3.75	5.88	1.53	0.55	3.39

Table 7.2: Error (in meters) between estimated pose of the connected panoramic views compared to the ground truth tag.

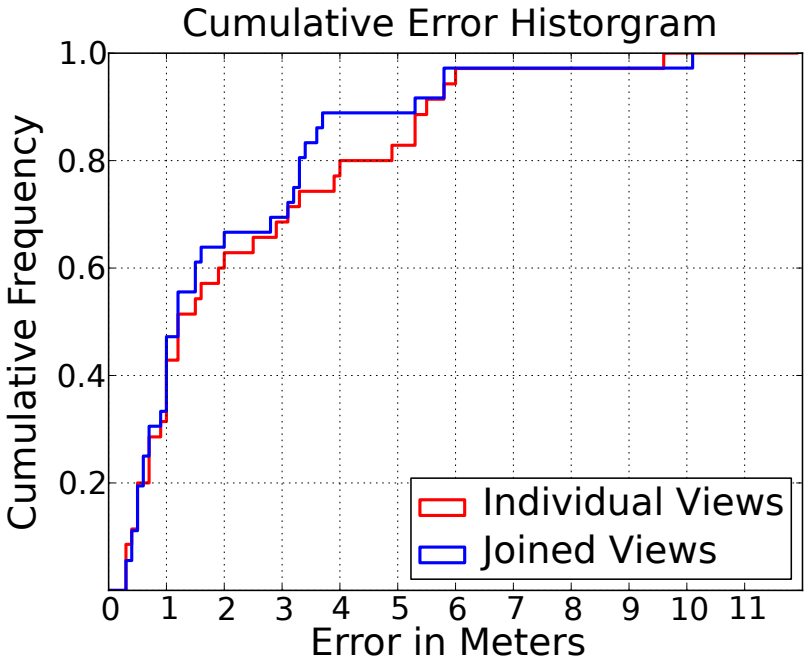


Figure 7.8: Cumulative error histogram for the parking lot experiment. The optimization of the connected panoramic views (blue) improves the performance.

As expected, the localization accuracy decreases with increase in the distance to tracked features. Points that are far away from the camera show small motion. In these cases, small errors in the odometry estimate and in the keypoint position in the image cause considerable errors in the estimated feature distances in 3D. Nevertheless, about 40% of the runs we are within a $1m$ accuracy, 60 % within $1.5m$. This is significantly lower than the accuracy on mobile devices (5 to 8.5 m) which use cellular network and GPS (Zandbergen and Barbeau, 2011).

Despite our efforts in providing accurate ground truth, this is not free of errors. Especially because the exact center of the panorama is unknown. Manually acquired panoramas are difficult to generate and often the camera center moves. The individual images which are stitched together often do not share the same exact camera center.

7.4.2 Urban Localization with a Google Tango Device

In order to show the flexibility of our approach, we evaluated our algorithm with a Google Tango device in two urban environments. We used the integrated visual inertial odometry estimated on the Tango device for our method. Tango has two cameras: one that has a high resolution but a narrow field-of-view, and another one, that has a lower resolution but a wider field-of-view. The narrow field-of-view camera has a frame rate of 5Hz, the other streams at 30Hz. We use the higher resolution camera as the monocular image source for our framework, meanwhile the wide angle camera is used internally by the device for the odometry estimates. Throughout our experiments, we found the odometry from Tango to be significantly more accurate indoor than outdoor. This is probably due to a relatively weak feature stability outdoors and the presence of only small baselines when navigating in wide areas. To alleviate this problem, we mounted a mirrored $45 - 90 - 45$ prism on the narrow-field-of-view camera and pointed the wide field-of-view to the floor. In this way, the Tango device reliably tracks features on the asphalt and computes accurate odometry estimates, meanwhile the other camera points at the side. Figure 7.11 shows the prism mounted Tango device.

The first urban scenario has been run on roads around the University campus in Freiburg, Germany. In particular, around the area with GPS coordinates $48.0125518, 7.8322567$. The panoramas used in the Tango experiments are public and can be viewed on Street View. In the experiment, we crossed a railway line by using an underpass where GPS connection is lost, see Figure 7.9. Our method is able to estimate 3D points from the images acquired from Tango and then match them to the nearby panoramic images, see Figure 7.9. Then, we moved into the suburban road with houses on both sides. This location is challenging due to the fact that all houses look similar. Also in this case, our approach is able to correctly estimate the 3D points of the track and localize the nearest panorama, see Figure 7.10. In the figure, the black points are the estimated 3D points while the circles in the center of the image are the positions of the panorama views. The

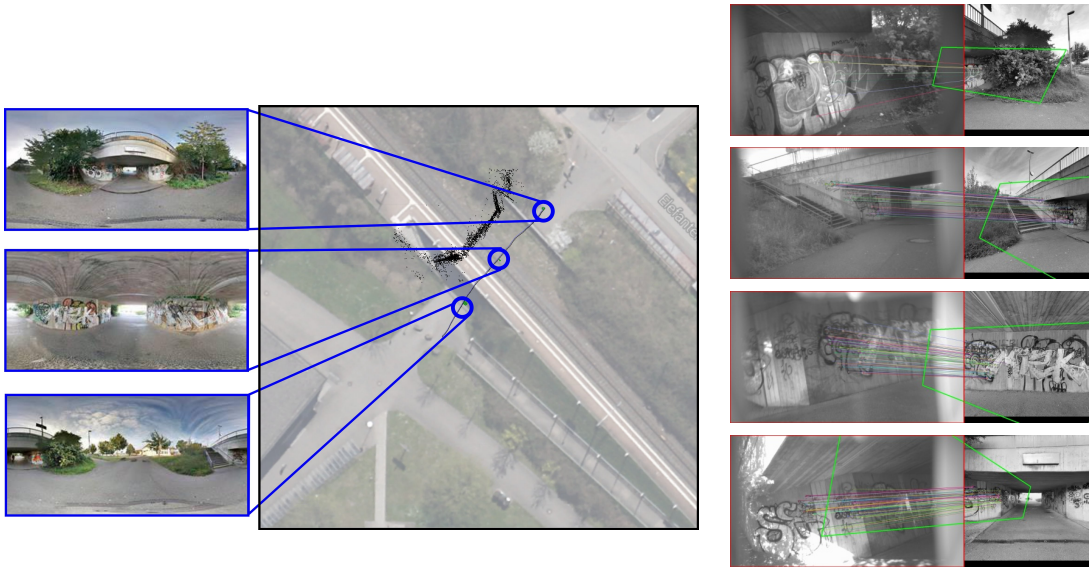


Figure 7.9: Matches between monocular images and Street View panoramas for a railway underpass used for the Tango experiments. The middle image shows the aerial imagery of the location, superimposed with localization results of the panoramic view with respect to the camera trajectory. The second panorama is acquired under the bridge while the other two are outside. The images on the right show example matches that were found between the monocular images and extracted rectilinear panoramic views.

Figure 7.10: Optimized 3D points with the estimated panorama position overlayed on Google maps (top). An example of matching between panorama views and Google Tango images. As both cameras are pointing in different directions, the features used internally for visual inertial odometry are different from the features used for localizing Street View panoramas.

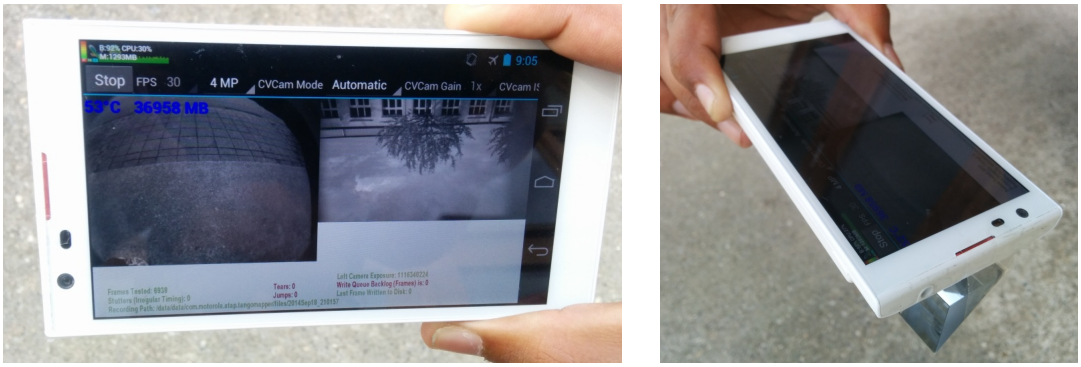


Figure 7.11: Google Tango with the prism attached to the narrow field of view camera. The screen shows the camera used for visual odometry pointing downwards, while the narrow field of view camera points sidewards.

pose of the Tango device is overlayed on the street.

To test our technique in a Street View panorama acquired by Google, we ran another experiment on the main road of the village of Marckolsheim, France. Despite being a busy road, our technique correctly estimated 3D points from the Tango image stream and successfully estimated the panorama positions, see Figure 7.1.

7.5 Discussion

Our method can use any kind of odometric input. In the case of Tango, the odometry is based on the work of Mourikis and Roumeliotis (2007) that makes use of visual odometry and IMUs to generate accurate visual-inertial odometry (VIO). This system is offered by the Google's Tango device libraries. When implemented on Tango, our method uses the two onboard cameras. One is the wide angle camera, that is used exclusively for VIO and the other is the narrow FOV camera, that is used for matching against Street View imagery. It is important to note that they point in different directions and do not share views. For this reason, the resulting features for VIO and 3D localization are not directly correlated. Note also that our two step optimization can in principle be done in one step. Our choice to do it in two steps resulted from a practical perspective: the first is used to compute a good initial solution for the second optimization. For the scope of this paper, we are not interested in using the panoramas to build an accurate large model of the environment: we aim at localizing without building new large scale maps where Street View exists.

7.6 Summary

In this chapter, we present a novel approach to metric localization by matching Google's Street View imagery to a moving monocular camera. Our method is able to metrically localize without requiring a robot to pre-visit locations to build a map where Street View exists. This is in contrast to the general problem of SLAM for which we developed robust methods in earlier chapters. If maps do not exist, we can use the robust SLAM algorithms developed in the previous chapters, but if maps exist and a robot can re-use them, it will allow them to navigate without needing to explore the full environment first.

We model the problem of localizing a robot with Street View imagery as a non-linear least squares estimation in two phases. The first estimates the 3D position of tracked feature points from short monocular camera streams, while the second computes the rigid body transformation between the points and the panoramic image. The sensor requirements of our technique are a monocular image stream and odometry estimates. This makes the algorithm easy to deploy and affordable to use. In our experiments, we evaluated the metric accuracy of our technique by using fiducial markers in a wide outdoor area. The results demonstrate high accuracy in different environments. Additionally, to show the flexibility and the potential application of this work to personal localization, we also ran experiments using images acquired with Google Tango smartphone in two different urban scenarios. We believe that this technique paves the way towards a new cheap and widely useable outdoor localization approach.

Chapter 8

Conclusion

Robust localization and mapping are fundamental requirements for mobile robots to be deployed for autonomous tasks. Together, the solution to these two problems will enable autonomous agents to reason about their surroundings for tasks such as obstacle avoidance and path planning. Modern graph-based simultaneous localization and mapping (SLAM) algorithms are computationally efficient at solving large problems, but the resulting solution significantly deteriorates when a robot cannot identify a previously visited place correctly. Recognizing a previously visited place is fundamentally hard and is prone to error making due to many reasons. For example, the appearance of a place may change over time and many different places may look similar. Additionally, sensors are inherently noisy. Measurements from Radars, LIDARs and GPS can all report erroneous measurements. A localization and mapping algorithm that relies on an error-free place recognition system or perfect sensors will eventually fail. We believe that localization and mapping algorithms should be robust to such failures. In this thesis, we have presented several methods which are robust to place recognition and sensor errors, while at the same time our methods have a large basin of convergence, i.e., they are robust to poor initialization. In addition, we have surveyed geodetic map building methods and presented them in relation to SLAM.

Researchers from geodesy and photogrammetry have been building maps for centuries. However, until now, no article has provided a systematic overview of their relation to robot mapping. The first contribution of this thesis is an extensive survey of geodetic mapping methods and the relation to graph-based SLAM. The geodetic methods surveyed in this thesis are relevant as robotic and geodetic map building methods share many similarities. Both require solving large optimization problems generated from observations between multiple positions. Additionally, maps require incremental updating over time with observations subjected to gross errors. Since both fields share similar challenges, this survey is highly relevant. We believe that this survey will inspire new robot map building techniques.

The second contribution of this thesis is max-mixtures, which is a technique to build maps by optimizing over a network of Gaussian mixtures. Conventionally, all constraints

are considered to be unimodal Gaussians. This is limiting in multiple scenarios. The max-mixture method allows to express constraints as Gaussian mixtures such that loop closures can be robustly represented with a mixture of Gaussians. A null hypothesis encoded by an additional uniform distribution accounts for the small probability that the constraint is an outlier. A traditional sum mixture cannot be converted directly into a least squares formulation. The key insight of the max-mixture approach is to replace the sum with a max operator. This seemingly trivial change allows to convert the Gaussian mixture constraint into an optimizable least squares problem while retaining the multi-modal expressivity with linear memory requirements. The resulting error of this approximation is small if the modes are far apart which is often the case in SLAM. Max-mixtures is robust to front-end errors and also allows to express multi-modal constraints. We have evaluated max-mixtures on many real world SLAM problems and have shown its robustness against front-end outliers, perceptual aliasing and robot wheel slippage.

Our third contribution is Dynamic Covariance Scaling (DCS), which is a SLAM algorithm robust to significant front-end outliers as well as poor graph initialization. It complements max-mixtures as it does not require accurate uncertainty distributions. Another characteristic of this method is that is generalizes switchable constraints (Sünderhauf and Protzel, 2012) and explains the robustness of switchable constraints from a theoretical perspective. DCS is related to iterative reweighted least squares and is a special instance of the Geman-McClure M-estimator. It is the current state-of-the-art method for robust SLAM. Its application includes robust localization and map optimization problems with 2D or 3D landmarks, or bundle adjustment problems. Additionally, it has been successfully deployed on various robots and tested by other researchers.

The fourth contribution is a monocular camera based metrical localization method which leverages maps built for humans. Most of the existing robot navigation methods present robot localization as a pose estimation problem in a previously computed map by the robot. Instead, our approach does not require the construction of a large scale consistent map by the robot. Specifically, it uses panoramic images on Google's Street View as an accurate source of global positioning. It can localize by formulating the problem as a non-linear least squares estimation in two phases. The first estimates the 3D position of tracked feature points from short camera sequences. The second computes the rigid body transformation between the panoramic Street View images and the estimated points. This research has significant implications as it will allow to accurately estimate the position of a robot equipped only with a monocular camera without building new large-scale maps.

The techniques developed in this thesis have improved localization and mapping algorithms and made them more robust to gross errors in sensor measurements and motion model and robust to failures of place recognition systems. Our techniques were thoroughly tested by us on both simulated datasets and those collected with a robot. Our algorithms

have subsequently been re-implemented and evaluated by other research groups for improving the robustness of long-term visual SLAM in dynamic environments (Carlevaris-Bianco, 2015) for ship hull inspection using an unmanned underwater rover (Ozog et al., 2015) and for building large maps with RGB-D cameras (Lee and Myung, 2014). In general, before the development of the techniques in this thesis, SLAM back-ends used in robotics relied on front-ends to identify wrong loop-closures and errors in sensor measurements. A single outlier that passed through the front-end verification could result in the SLAM process to completely fail. The robust techniques developed in this thesis have relaxed the reliance on perfect front-ends and are helping current robotic systems to build better maps. Today, our methods also serve as a state-of-art benchmark for robust SLAM evaluations (Sünderhauf and Protzel, 2013; Latif et al., 2014).

Future work

Although the methods described in this thesis have improved the robustness of SLAM algorithms against significant sensor and place recognition errors, there exists several possibilities for future extensions.

In the future, we would like to improve the robustness of SLAM back-ends even further. It will be interesting to combine the multi-modal benefits of max-mixtures with the robustness of DCS. Currently, each mode of max-mixtures is a Gaussian. For robustness against outliers, max-mixtures uses a mixture of two components. The first represents the front-end observation and the second is a null-hypothesis encoded with a large variance Gaussian. This can be easily subsumed by DCS. More interestingly, the k-modal or multi-modal constraints in max-mixture for k constraints are modeled as a Gaussian mixtures with $k + 1$ modes, with the extra mode accounting for the null-hypothesis that represents all the constraints are wrong. This could be replaced with k modes, each with a DCS kernel. This will be an interesting topic to study, even though, it would lead to loosing the probabilistic interpretation of max-mixtures.

One limitation of the tuning parameter Φ in DCS is that it is fixed, typically chosen at the beginning of the optimization process. Intuitively, a larger value of Φ will allow more factors with large errors. Mazuran et al. (2014) have explored methods to find the best parameter by searching over the space of parameters. This works in practice, but it does not allow to formulate richer models of different Φ values for different regions of the map. This could be relevant to an incremental mapping problem where newly explored regions are more susceptible to data association errors compared to relatively, well explored parts of the map. One alternative to a fixed Φ value is to use a learning rate which is re-weighted as the optimization converges, similar to the learning rate used by Olson et al. (2007).

Additionally, the use of robust methods such as DCS and max-mixtures affects the

sparsity structure of the information matrix. The outliers they reject typically increase computational requirements during the matrix factorization phase. This is because SLAM back-ends typically rely on sparse matrix factorization methods, whose efficiency goes down with the reduction in sparsity of the information matrix. With robust back-ends, the resulting entries for outliers in the information matrix are *almost* zero. This typically results in extra fill-in, which cannot be exploited by various variable ordering algorithms. In the future, we would like to explore heuristics which maintain the sparsity while still being robust to outliers.

Currently, all the methods explored in this thesis converge to a single solution. For a large number of non-linear SLAM problems, verifying that this single solution is correct is challenging. One alternative solution would be to evaluate multiple solutions that are seeded from different initializations and then evaluate the maps they converge to.

In the future, we would like to use our camera based localization method that leverages Google's Street View for end-to-end robot navigation tasks. Currently, our approach only localizes a stream of monocular images, but Street View can be also used to first plan paths to the desired destination and then navigate a robot on this path while keeping the robot localized using the approach outlined in Chapter 7. Additionally, researchers are focusing on localization algorithms which are robust to seasonal changes (Churchill and Newman, 2012; Naseer et al., 2014; Milford and Wyeth, 2012; Neubert et al., 2013). A large part of Google's Street View image database contains images across multiple seasons. Our localization algorithm based on Street View could in principle be directly applied to localize a robot across multiple seasons at locations where Street View images across seasons exist.

Finally, most SLAM algorithms typically assume a static world, i.e., the dynamic objects in the world are not part of the localization and map optimization process. Some researchers have addressed this problem by first identifying the dynamic part in each individual sensor data frame (camera image or LIDAR point cloud) and then removing it before building the map (Hähnel et al., 2003b; Pomerleau et al., 2014; Carlevaris-Bianco, 2015). In the real world, autonomous robots are always interacting with other moving agents. For example, self-driving cars are constantly tracking other moving cars, cyclists and pedestrians. We believe that the traditional approach of removing dynamic parts of the scene and tracking each object in the environment with filtering approaches while using non-linear approaches for robot SLAM is limiting. In the future, we would like to integrate object tracking and SLAM in a single robust non-linear optimization method capable of tracking dynamic objects and performing SLAM jointly. Conceptually, this is not different from multi-robot graph-based SLAM algorithms (Cunningham et al., 2010; Olson et al., 2012; Kim et al., 2010). Instead of building one map, the system builds multiple models, one of each tracked object and one of the environment.

Appendices

Appendix A

Robust M-estimators

M-estimators reduce the influence of outliers by replacing the squared error:

$$\min \sum_i r_i^2, \tag{A.1}$$

with another function such that:

$$\min \sum_i \rho(r_i), \tag{A.2}$$

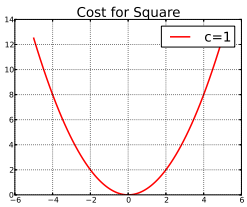
Different characteristics of $\rho(x)$ provides different behavior during optimization. In this section we provide an overview of the following common M-estimators:

- Cauchy
- DCS
- Fair
- Geman-McClure
- Huber
- Psuedo-Huber
- Saturated (also called Blake-Zisserman)
- Tukey
- Welsch

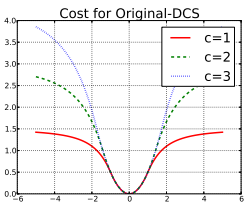
Table A.1 describes the robust error function, influence and the weight of each of the above mentioned M-estimators. Figure A.1 illustrates all the robust cost surface for different tuning parameters $c \in \{1, 2, 3\}$. An M-estimator is redescending if it satisfies:

$$\lim_{z \to \pm\infty} \rho'(z) \quad = \quad 0 \tag{A.3}$$

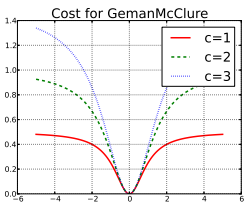
Non-redescending M-estimators such as L1 and Huber have a breakdown point of $\frac{1}{N}$, i.e. they can be biased just by 1 outlier (Müller, 2004).



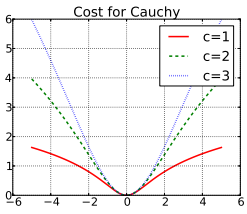
(a) Squared



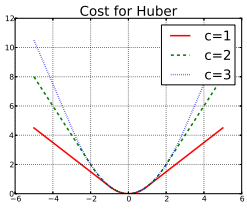
(b) DCS



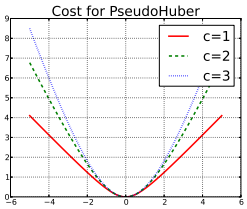
(c) GemanMcClure



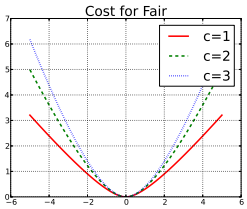
(d) Cauchy



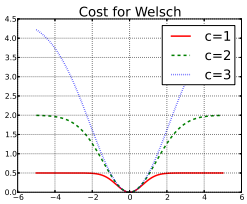
(e) Huber



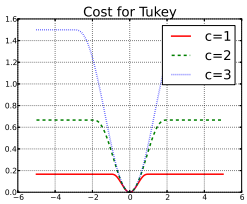
(f) PseudoHuber



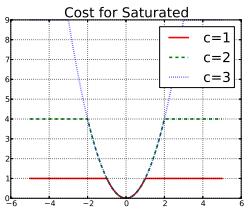
(g) Fair



(h) Welsch



(i) Tukey



(j) Saturated

Figure A.1: Common M-estimators and their robust cost functions $\rho(e)$ for different tuning parameters $c \in \{1, 2, 3\}$.

Table A.1: Common M-estimators and their cost, influence and weight terms

M-estimator	$\rho(x)$	$\psi(x)$	$w(x)$	Redescending or Non-Redescending
Cauchy	$\frac{c^2}{2}\log\left(1+\frac{x^2}{c^2}\right)$	$\frac{x}{1+\frac{x^2}{c^2}}$	$\frac{1}{1+\frac{x^2}{c^2}}$	Redescending
DCS $\begin{cases}\text{if } x^2 \le \Phi \\ \text{if } x^2 > \Phi\end{cases}$	$\begin{cases}\frac{x^2}{2} \\ \frac{\Phi\left(3x^2-\Phi\right)}{2\left(x^2+\Phi\right)}\end{cases}$	$\begin{cases}x \\ \frac{4\Phi^2 x}{(x^2+\Phi)^2}\end{cases}$	$\begin{cases}1 \\ \frac{4\Phi^2}{(x^2+\Phi)^2}\end{cases}$	Redescending
Geman-McClure (generalized)	$\frac{1}{2}\frac{\delta^2 x^2}{\delta^2+x^2}$	$\frac{\delta^4 x}{\left(\delta^2+x^2\right)^2}$	$\frac{\delta^4}{\left(\delta^2+x^2\right)^2}$	Redescending
Fair	$c^2\left[\frac{\lvert x\rvert}{c}-\log\left(1+\frac{\lvert x\rvert}{c}\right)\right]$	$\frac{x}{1+\frac{\lvert x\rvert}{c}}$	$\frac{1}{1+\frac{\lvert x\rvert}{c}}$	Non-Redescending
Huber $\begin{cases}\text{if } \lvert x\rvert \le k \\ \text{if } \lvert x\rvert > k\end{cases}$	$\begin{cases}\frac{x^2}{2} \\ k\left(\lvert x\rvert-\frac{k}{2}\right)\end{cases}$	$\begin{cases}x \\ k\,\mathrm{sign}(x)\end{cases}$	$\begin{cases}1 \\ \frac{k}{\lvert x\rvert}\end{cases}$	Non-Redescending
Psuedo-Huber	$k^2\left(\sqrt{1+\frac{x^2}{k^2}}-1\right)$	$\frac{x}{\sqrt{\frac{x^2}{k^2}+1}}$	$\frac{1}{\sqrt{\frac{x^2}{k^2}+1}}$	Non-Redescending
Saturated $\begin{cases}\text{if } x^2 \le c^2 \\ \text{if } x^2 > c^2\end{cases}$	$\begin{cases}\frac{x^2}{2} \\ \frac{c^2}{2}\end{cases}$	$\begin{cases}x \\ 0\end{cases}$	$\begin{cases}1 \\ 0\end{cases}$	Redescending
Tukey $\begin{cases}\text{if } \lvert x\rvert \le c \\ \text{if } \lvert x\rvert > c\end{cases}$	$\begin{cases}\frac{c^2}{6}\left[1-\left(1-\frac{x^2}{c^2}\right)^3\right] \\ \frac{c^2}{6}\end{cases}$	$\begin{cases}x\left(1-\frac{x^2}{c^2}\right)^2 \\ 0\end{cases}$	$\begin{cases}\left(1-\frac{x^2}{c^2}\right)^2 \\ 0\end{cases}$	Redescending
Welsch	$\frac{c^2}{2}\left[1-\exp\left(-\frac{x^2}{c^2}\right)\right]$	$x\exp\left(1-\frac{x^2}{c^2}\right)$	$\exp\left(1-\frac{x^2}{c^2}\right)$	Redescending

Bibliography

O. S. Adams. *The Bowie Method of Triangulation Adjustment, as applied to the first-order net in the Western part of the United States*. US Government Printing Office, 1930.

Aeronautical Chart and Information Center. *Preface and Part I: The Mathematical Theories*. St. Louis 18, MO., 1964.

P. Agarwal and E. Olson. Variable reordering strategies for SLAM. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3844–3850, 2012.

P. Agarwal, E. Olson, and W. Burgard. Max-mixture - open source implementation with g2o. http://openslam.org/maxmixture.html/, 2012.

P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Dynamic covariance scaling for robust robotic mapping. In *Workshop on robust and Multimodal Inference in Factor Graphs, (ICRA)*, 2013a.

P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Robust map optimization using dynamic covariance scaling. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 62–69, 2013b.

P. Agarwal, W. Burgard, and C. Stachniss. Helmert's and Bowie's Geodetic Mapping Methods and Their Relationship to Graph-Based SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3619–3625, 2014a.

P. Agarwal, W. Burgard, and C. Stachniss. A Survey of Geodetic Approaches to Mapping and the Relationship to Graph-Based SLAM. *Robotics and Automation Magazine*, 21 (3):63–80, September 2014b.

P. Agarwal, G. Grisetti, G. D. Tipaldi, L. Spinello, W. Burgard, and C. Stachniss. Experimental analysis of dynamic covariance scaling for robust map optimization under bad initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3626–3631, 2014c.

M. Agrawal and K. Konolige. FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008.

American Congress on Surveying, American Society for Photogrammetry and Remote Sensing, and American Society of Civil Engineers. The glossary of the mapping sciences, 1994.

D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver. Google street view: Capturing the world at street level. *Computer*, (6): 32–38, 2010.

J. Avila and J. Tomlin. Solution of very large least squares problems by nested dissection on a parallel processor. In *Proceedings of the 12th Symposium on the Interface*, pages 9–14, 1979.

T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, Australian Centre for Field Robotics, University of Sydney, August 2002.

T. Bailey, J. Nieto, and E. Nebot. Consistency of the FastSLAM algorithm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 424–429, 2006.

H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Computer vision and image understanding*, 110(3):346–359, 2008.

P. Besl and N. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2):239–256, 1992.

P. Biber and W. Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2743–2748, 2004.

S. S. Blackman. Multiple hypothesis tracking for multiple target tracking. *IEEE Aerospace and Electronic Systems Magazine*, 19(1):5–18, 2004.

H. Boltz. *Entwickelungs-Verfahren zum Ausgleichen Geodätischer Netze nach der Methode der kleinsten Quadrate*. Number 90. Veröffendlichungen des Geodätischen Instituts Potsdam, 1923.

H. Boltz. *Substitutions-Verfahren zum Ausgleichen grosser Dreiecksnetze in einem Guss nach der Methode der kleinsten Quadrate*. Number 108. Veröffendlichungen des Geodätischen Instituts Potsdam, 1938.

G. Bomford. The readjustment of the Indian triangulation. *Professional paper No. 28, Survey of India, Dehra Dun*, 1939.

D. Borrmann, J. Elseberg, K. Lingemann, A. Nüchter, and J. Hertzberg. Globally consistent 3d mapping with scan matching. *Robotics and Autonomous Systems*, 56(2): 130–142, 2008.

M. Bosse and R. Zlot. Continuous 3d scan-matching with a spinning 2d laser. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4312–4319, 2009.

M. Bosse, P. Newman, J. J. Leonard, and S. Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *International Journal of Robotics Research (IJRR)*, 23(12):1113–1139, December 2004.

M. C. Bosse. *ATLAS: A Framework for Large Scale Automated Mapping and Localization*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, February 2004.

J. D. Bossler. Geodesy solves 900,000 equations simultaneously. *Eos, Transactions American Geophysical Union*, 68(23):569–569, 1987.

R. K. Burkard. *Geodesy for the Layman*. US Department of commerce, National Oceanic and Atmospheric Administration, St. Louis, Missouri, 1983.

M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 778–792. Springer, 2010.

N. Carlevaris-Bianco. *Long-term simultaneous localization and mapping in dynamic environments*. PhD thesis, Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, USA, January 2015.

L. Carlone. A convergence analysis for pose graph optimization via gauss-newton methods. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 965–972, 2013.

L. Carlone and A. Censi. From angular manifolds to the integer lattice: Guaranteed orientation estimation with application to pose graph optimization. *arXiv preprint arXiv:1211.3063*, 2012.

L. Carlone, R. Aragues, J. Castellanos, and B. Bona. A linear approximation for graph-based simultaneous localization and mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 41–48, 2011.

A. Censi. An ICP variant using a point-to-line metric. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 19–25, 2008.

Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software (TOMS)*, 35:22:1–22:14, 2008.

W. Churchill and P. Newman. Practice makes perfect? managing and leveraging visual experiences for lifelong navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4525–4532, 2012.

I. J. Cox. Blanche-an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics (TRO)*, 7(2):193–204, 1991.

M. Cummins and P. Newman. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

M. Cummins and P. Newman. Highly scalable appearance-only SLAM - FAB-MAP 2.0. In *Proceedings of Robotics: Science and Systems (RSS)*, 2009.

A. Cunningham, B. Paluri, and F. Dellaert. Ddf-sam: Fully distributed slam using constrained factor graphs. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

T. A. Davis. Suitesparse, 2006a. URL http://www.cise.ufl.edu/research/sparse/SuiteSparse/.

T. A. Davis. *Direct methods for sparse linear systems*, volume 2. Society for Industrial and Applied Mathematics, 2006b.

A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6):1052–1067, 2007.

F. Dellaert. Square root SAM. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 177–184, 2005.

F. Dellaert and M. Kaess. Square root SAM: Simultaneous localization and mapping via square root information smoothing. *International Journal of Robotics Research (IJRR)*, 25(12):1181–1203, 2006.

F. Dellaert, J. Carlson, V. Ila, K. Ni, and C. E. Thorpe. Subgraph-preconditioned conjugate gradients for large scale slam. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2566–2571, 2010.

A. Diosi and L. Kleeman. Fast Laser Scan Matching using Polar Coordinates. *International Journal of Robotics Research (IJRR)*, 26(10):1125–1153, 2007.

G. Dubbelman, P. Hansen, B. Browning, and M. B. Dias. Orientation only loop-closing with closed-form trajectory bending. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 815–821, 2012.

T. Duckett, S. Marsland, and J. Shapiro. Learning globally consistent maps by relaxation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3841–3846, 2000.

H. Durrant-Whyte, S. Majumder, S. Thrun, M. de Battista, and S. Scheding. A bayesian algorithm for simultaneous localisation and map building. In R. Jarvis and A. Zelinsky, editors, *Robotics Research*, volume 6 of *Springer Tracts in Advanced Robotics*, pages 49–60. Springer Berlin / Heidelberg, 2003.

W. Ehrnsperger. The ED87 Adjustment. *Journal of Geodesy*, 65(1):28–43, 1991.

C. Estrada, J. Neira, and J. D. Tardós. Hierachical SLAM: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics (TRO)*, 21(4):588–596, 2005.

R. Eustice, H. Singh, J. Leonard, and M. Walter. Visually mapping the RMS Titanic: Conservative covariance estimates for SLAM information filters. *International Journal of Robotics Research (IJRR)*, 25(12):1223–1242, December 2006.

L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 524–531. IEEE, 2005.

J. Folkesson and H. I. Christensen. Graphical SLAM - a self-correcting map. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 383–390, 2004.

U. Frese. Treemap: An $O(\log n)$ algorithm for indoor simultaneous localization and mapping. *Autonomous Robotics*, 21(2):103–122, 2006.

U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics*, 21(2):196–207, April 2005.

J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, pages 1–27, 2012.

S. Geman and D. E. McClure. Statistical methods for tomographic image reconstruction. *Bulletin of the International Statistical Institute*, 52:5–21, 1987.

A. George. Nested dissection of a regular finite element mesh. *SIAM Journal on Numerical Analysis*, 10(2):345–363, 1973.

G. H. Golub and R. J. Plemmons. Large-scale geodetic least-squares adjustment by dissection and orthogonal decomposition. *Linear Algebra and Its Applications*, 34: 3–28, 1980.

Google Inc. The never-ending quest for the perfect map. http://googleblog.blogspot.de/2012/06/never-ending-quest-for-perfect-map.html/, 2012.

G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2432–2437, Barcelona, April 2005.

G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Transactions on Intelligent Transportation Systems Magazine*, 2:32–43, 2010a.

G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, and C. Hertzberg. Hierarchical optimization on manifolds for online 2D and 3D mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 273–278, 2010b.

G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. Efficient estimation of accurate maximum likelihood maps in 3D. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3472–3478, 2007a.

G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 65–72, 2007b.

G. Grisetti, D. L. Rizzini, C. Stachniss, E. Olson, and W. Burgard. Online constraint network optimization for efficient maximum likelihood map learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1880–1885, 2008.

G. Grisetti, C. Stachniss, and W. Burgard. Nonlinear constraint network optimization for efficient map learning. *Intelligent Transportation Systems, IEEE Transactions on*, 10 (3):428–439, 2009.

G. Grisetti, R. Kummerle, and K. Ni. Robust optimization of factor graphs by using condensed measurements. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 581–588, 2012.

J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence (CIRA)*, pages 318–325, 1999.

J.-S. Gutmann and C. Schlegel. Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *Advanced Mobile Robot, 1996., Proceedings of the First Euromicro Workshop on*, pages 61–67. IEEE, 1996.

D. Hähnel, W. Burgard, D. Fox, and S. Thrun. A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 206–211, 2003a.

D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003b.

R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

F. Helmert. Über die Wahrscheinlichkeit der Potenzsummen der Beobachtungsfehler und über einige damit im Zusammenhang stehende Fragen. *Zeitschrift für Mathematik und Physik*, 21:192–218, 1876.

F. Helmert. *Die mathematischen und physikalischen Theorien der hoheren Geodasie-Einleitung*. B.G. Teubner, Leipzig, Germany, 1880.

C. Hertzberg. A framework for sparse, non-linear least squares problems on manifolds. Master's thesis, Universität Bremen, department of Computer Science, 2008.

P. Hliněnỳ and G. Salazar. On the crossing number of almost planar graphs. In *Graph Drawing*, pages 162–173. Springer, 2007.

F. W. Hough. The adjustment of the central european triangulation network. *Bulletin Géodésique*, 7(1):64–93, 1948.

A. Howard and N. Roy. The Robotics Data Set Repository (Radish), 2003. URL http://radish.sourceforge.net/.

A. Howard, M. J. Mataric, and G. Sukhatme. Relaxation on a mesh: a formalism for generalized localization. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 1055–1060, 2001.

G. Hu, K. Khosoussi, and S. Huang. Towards a reliable SLAM back-end. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 37–43, 2013.

A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2599–2606, 2009.

H. Johannsson, M. Kaess, M. Fallon, and J. J. Leonard. Temporally scalable visual slam using a reduced pose graph. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 54–61, 2013.

M. Kaess. AprilTags C++ Library, 2013. URL http://people.csail.mit.edu/kaess/apriltags.

M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Fast incremental smoothing and mapping with efficient data association. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1670–1677, 2007a.

M. Kaess, A. Ranganathan, and F. Dellaert. Fast incremental square root information smoothing. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2129–2134, 2007b.

M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Transactions on Robotics (TRO)*, 24(6):1365–1378, December 2008.

M. Kaess, H. Johannsson, R. Roberts, V. . Ila, J. J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research (IJRR)*, 31(2):216–235, 2012.

K. Kersting. Lifted probabilistic inference. In *Proceedings of 20th European Conference on Artificial Intelligence (ECAI–2012)*, pages 33–38, 2012.

B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. Multiple relative pose graphs for robust cooperative mapping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3185–3192, May 2010.

B. Klingner, D. Martin, and J. Roseborough. Street view motion-from-structure-from-motion. In *IEEE International Conference on Computer Vision (ICCV)*, pages 953–960, 2013.

G. B. Kolata. Geodesy: dealing with an enormous computer task. *Science*, 200(4340): 421–466, 1978.

K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent. Efficient sparse pose adjustment for 2d mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 22–29, 2010.

K. Konolige. Large-scale map-making. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 457–463, 2004.

P. Krauthausen, F. Dellaert, and A. Kipp. Exploiting locality by nested dissection for square root smoothing and mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 73–80, 2006.

W. Kruskal. Helmert's distribution. *The American Mathematical Monthly*, 53(8):435–438, 1946.

R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, and A. Kleiner. On Measuring the Accuracy of SLAM Algorithms. *Journal of Autonomous Robots*, 27(4):387–407, 2009.

R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3607–3613, 2011.

R. Kümmerle. *State Estimation and Optimization for Mobile Robot Navigation*. PhD thesis, Albert-Ludwigs-University of Freiburg, Department of Computer Science, April 2013.

N. Kwak, I.-K. Kim, H.-C. Lee, and B. H. Lee. Analysis of resampling process for the particle depletion problem in fastslam. In *IEEE International Symposium on Robots and Human Interactive Communications (RO-MAN)*, pages 200–205, 2007.

Y. Latif, C. Cadena, and J. Neira. Realizing, reversing, recovering: Incremental robust loop closing over time using the irrr algorithm. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4211–4217, 2012a.

Y. Latif, C. C. Lerma, and J. Neira. Robust loop closing over time. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 233–240, 2012b.

Y. Latif, C. Cadena, and J. Neira. Robust Graph SLAM Back-ends: A Comparative Analysis. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.

D. Lee and H. Myung. Solution to the SLAM problem in low dynamic environments using a pose graph and an RGB-D sensor. *Sensors*, 14(7):12467–12496, 2014.

U. Lerner and R. Parr. Inference in hybrid networks: Theoretical limits and practical algorithms. In *Proceedings of the Proceedings of the Seventeenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 310–331. Morgan Kaufmann, 2001.

S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2548–2555, 2011.

Y. Li and E. Olson. Extracting general-purpose features from LIDAR data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1388–1393, May 2010.

Y. Liu and H. Zhang. Visual loop closure detection with a compact image descriptor. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1051–1056. IEEE, 2012.

D. G. Lowe. Object Recognition from Local Scale-Invariant Features. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1150–1157, 1999.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 2004.

F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4(4):333–349, 1997.

F. Lu. *Shape registration using optimization for mobile robot navigation*. PhD thesis, University of Toronto, 1995.

A. L. Majdik, Y. Albers-Schoenberg, and D. Scaramuzza. MAV urban localization from Google street view data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3979–3986, 2013.

A. L. Majdik, D. Verda, Y. Albers-Schoenberg, and D. Scaramuzza. Micro air vehicle localization and position tracking from textured 3d cadastral models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 920–927, 2014.

M. Mazuran and F. Amigoni. Matching line segment scans with mutual compatibility constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4298–4303, 2014.

M. Mazuran, G. D. Tipaldi, L. Spinello, W. Burgard, and C. Stachniss. A statistical measure for map consistency in SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3650–3655, 2014.

K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, pages 1615–1630, 2005.

K. O. Milbert. An evaluation of the High Accuracy Reference Network relative to the Continuously Operating Reference Stations. URL http://www.ngs.noaa.gov/PUBS_LIB/HARN_CORS_COMP/eval_harn_to_cors.html.

M. J. Milford and G. F. Wyeth. SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1643–1649, 2012.

M. Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2003.

A. I. Mourikis and S. I. Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3565–3572, 2007.

M. Muja and D. G. Lowe. Fast matching of binary features. In *Computer and Robot Vision (CRV)*, pages 404–410, 2012.

C. Müller. Redescending M-estimators in regression analysis, cluster analysis and image analysis. *Discussiones Mathematicae-Probability and Statistics*, 24:59–75, 2004.

T. Naseer, L. Spinello, W. Burgard, and C. Stachniss. Robust visual robot localization across seasons using network flows. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2014.

Natural Resources Canada. 100 years of geodetic surveys in canada. http://www.geod.nrcan.gc.ca/timeline/timeline_e.php/, 2009.

J. Neira and J. D. Tardos. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6):890–897, December 2001.

P. Neubert, N. Sünderhauf, and P. Protzel. Appearance change prediction for long-term navigation across seasons. In *European Conference on Mobile Robots (ECMR)*, pages 198–203, 2013.

K. Ni and F. Dellaert. Multi-Level submap based SLAM using nested dissection. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2558–2565, 2010.

K. Ni, D. Steedly, and F. Dellaert. Tectonic SAM: Exact; out-of-core; submap-based slam. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1678–1685, 2007.

J. Nieto, T. Bailey, and E. Nebot. Recursive scan-matching slam. *Robotics and Autonomous Systems*, 55(1):39–49, 2007.

D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2161–2168. IEEE, 2006.

A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

E. Olson. *Robust and Efficient Robotic Mapping*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, June 2008.

E. Olson. Real-time correlative scan matching. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4387–4393, 2009a.

E. Olson. Recognizing places using spectrally clustered local matches. *Robotics and Autonomous Systems*, 2009b.

E. Olson. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011a.

E. Olson. Evaluating back-ends: Metrics. In *Automated SLAM Evaluation Workshop, Robotics Science and Systems*, Los Angeles, USA, 2011b.

E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 313–320, 2012.

E. Olson and P. Agarwal. Inference on networks of mixtures for robust robot mapping. *International Journal of Robotics Research (IJRR)*, 32(7):826–840, July 2013.

E. Olson, M. Walter, J. Leonard, and S. Teller. Single cluster graph partitioning for robotics applications. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 265–272, 2005.

E. Olson, J. Leonard, and S. Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2262–2269, 2006.

E. Olson, J. Leonard, and S. Teller. Spatially-adaptive learning rates for online incremental SLAM. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 73–80, 2007.

E. Olson, J. Strom, R. Morton, A. Richardson, P. Ranganathan, R. Goeddel, M. Bulic, J. Crossman, and B. Marinier. Progress towards multi-robot reconnaissance and the MAGIC 2010 competition. *Journal of Field Robotics (JFR)*, 29(5):762–792, September 2012.

P. Ozog, N. Carlevaris-Bianco, A. Kim, and R. M. Eustice. Long-term mapping techniques for ship hull inspection and surveillance using an autonomous underwater vehicle. *Journal of Field Robotics, Special Issue on Safety, Security and Rescue Robotics*, 2015. In Press.

L. Paz, J. Neira, and J. D. Tardós. Divide and Conquer: EKF SLAM in O(n). *IEEE Transactions on Robotics (TRO)*, 24(5):1107–1120, 2008.

M. Pfingsthorn and A. Birk. Simultaneous Localization and Mapping (SLAM) with Multimodal Probability Distributions. *International Journal of Robotics Research (IJRR)*, 32(2):143–171, 2012.

F. Pomerleau, P. Krusi, F. Colas, P. Furgale, and R. Siegwart. Long-term 3d map maintenance in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3712–3719. IEEE, 2014.

D. G. Pursell and M. Potterfield. NAD 83(NSRS2007) National Readjustment Final Report. Technical Report NOS NGS 60, National Oceanic and Atmospheric Administration, National Ocean Service, Silver Spring, MD, 2008.

P. Ranganathan, R. Morton, A. Richardson, J. Strom, R. Goeddel, M. Bulic, and E. Olson. Coordinating a team of robots for urban reconnaisance. In *Proceedings of the Land Warfare Conference (LWC)*, November 2010.

M. Ruhnke, R. Kümmerle, G. Grisetti, and W. Burgard. Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2812–2817, 2011.

T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 752–765, 2012a.

T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image retrieval for image-based localization revisited. In *British Machine Vision Conference (BMVC)*, page 7, 2012b.

B. Schumitsch, S. Thrun, G. Bradski, and K. Olukotun. The information-form data association filter. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1193–1200, Cambridge, MA, 2006. MIT Press.

C. R. Schwarz. North American datum of 1983. Technical Report NOAA Professional Paper NOS 2, US Department of Commerce, National Oceanic and Atmospheric Administration (NOAA), Rockville, MD, 1989.

A. Segal, D. Haehnel, and S. Thrun. Generalized-icp. In *Proceedings of Robotics: Science and Systems (RSS)*, Seattle, USA, June 2009.

O. Sheynin. Helmert's work in the theory of errors. *Archive for history of exact sciences*, 49(1):73–104, 1995.

G. Sibley, C. Mei, I. Reid, and P. Newman. Adaptive relative bundle adjustment. In *Proceedings of Robotics: Science and Systems (RSS)*, Seattle, USA, June 2009.

R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In O. Faugeras and G. Giralt, editors, *Proceedings of the International Symposium of Robotics Research (ISRR)*, pages 467–474, 1988.

C. Stachniss, G. Grisetti, and W. Burgard. Recovering particle diversity in a Rao-Blackwellized particle filter for SLAM after actively closing loops. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 667–672, Barcelona, Spain, 2005.

H. Strasdat, J. Montiel, and A. Davison. Real-time monocular SLAM: Why filter? In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2657–2664, 2010.

N. Sünderhauf. Vertigo: Versatile extensions for robust inference using graphical models, 2012. URL http://openslam.org/vertigo.html.

N. Sünderhauf and P. Protzel. Towards a robust back-end for pose graph slam. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1254–1261, 2012.

N. Sünderhauf and P. Protzel. Brief-gist-closing the loop by simple means. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1234–1241. IEEE, 2011.

N. Sünderhauf and P. Protzel. Switchable constraints vs. max-mixture models vs. rrr – a comparison of three approaches to robust pose graph slam. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

S. Thrun and Y. Liu. Multi-robot SLAM with sparse extended information filters. In *Proceedings of the International Symposium of Robotics Research (ISRR)*, pages 254–266, 2003.

S. Thrun and M. Montemerlo. The GraphSLAM algorithm with applications to large-scale mapping of urban structures. *International Journal of Robotics Research (IJRR)*, 25(5-6):403–429, 2006.

S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *International Journal of Robotics Research (IJRR)*, 23(7-8):693–716, July-August 2004.

G. D. Tipaldi, L. Spinello, and W. Burgard. Geometrical FLIRT phrases for large scale place recognition in 2d range data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2693–2698, 2013.

A. Torii, M. Havlena, and T. Pajdla. From google street view to 3d city models. In *Computer Vision Workshops (ICCV Workshops)*, 2009.

A. Torii, J. Sivic, and T. Pajdla. Visual localization by linear combination of image descriptors. In *Computer Vision Workshops (ICCV Workshops)*, 2011.

B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision algorithms: theory and practice*, pages 298–372. Springer, 2000.

M. Walter, R. Eustice, and J. Leonard. A Provably Consistent Method for Imposing Exact Sparsity in Feature-based SLAM Information Filters. In *Proceedings of the International Symposium of Robotics Research (ISRR)*, pages 214–234. Springer, 2005.

M. R. Walter, R. M. Eustice, and J. J. Leonard. Exactly Sparse Extended Information Filters for Feature-based SLAM. *International Journal of Robotics Research (IJRR)*, 26(4):335–359, 2007.

H. Wang, G. Hu, S. Huang, and G. Dissanayake. On the structure of nonlinearities in pose graph slam. In *Proceedings of Robotics: Science and Systems (RSS)*, pages 425–432, 2012.

J. Wang and E. Olson. Robust pose graph optimization using stochastic gradient descent. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, June 2014.

G. Weiß and E. v. Puttkamer. A map based on laserscans without geometric interpretation. In *Intelligent Autonomous Systems*, volume 4, pages 403–407, 1995.

H. Wolf. Gaußscher Algorithmus (Doolittle-Methode) und Boltzsches Entwicklungsverfahren. *Bulletin Géodésique*, 26(1):445–452, 1952.

H. Wolf. The helmert block method, its origin and development. In *Proceedings of the Second International Symposium on Problems Related to the Redefinition of North American Geodetic Networks*, pages 319–326, 1978.

H. Wolf. Triangulation adjustment general discussion and new procedure. *Bulletin Géodésique*, 16(1):87–104, 1950.

A. R. Zamir and M. Shah. Accurate image localization based on google maps street view. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 255–268, 2010.

P. A. Zandbergen and S. J. Barbeau. Positional accuracy of assisted gps data from high-sensitivity gps-enabled mobile phones. *Journal of Navigation*, 64(03):381–399, 2011.

W. Zhang and J. Kosecka. Image based localization in urban environments. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 33–40. IEEE, 2006.

Z. Zhang. Parameter Estimation Techniques: A Tutorial with Application to Conic Fitting. *Image and Vision Computing Journal*, 15:59–76, 1997.