Why do we optimize what we optimize in multiple view geometry?

Seong Hun Lee

Advisor: Javier Civera Sancho

Departmento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

This dissertation is submitted for the degree of
Doctor of Philosophy.

November 2022

Universidad Zaragoza

To Sinitta and Liana,
my two angels

Acknowledgements

First, I want to thank everyone in the defense committee: Professor José María Martínez Montiel, Professor Laurent Kneip and Professor Federica Arrigoni. I admire your research and contributions to the field, and I am honored that you are taking part in the final phase of my PhD journey. Thank you for the time and effort you put into reviewing this thesis.

Above all, I want to express my special thanks to my supervisor, Javier. It all started in early 2017 when I wrote my first email to Javier. I asked him if I could be his PhD student and do research on visual SLAM. At that time, I was still working on my Master's thesis at TU Delft in the Netherlands and had never met any academic researchers working on SLAM. I only knew of Javier from my prior internship at Wikitude in Salzburg, during which I read a lot of SLAM papers, including those of Alejo (Javier's PhD student at that time) and Javier. At Wikitude, I heard good things about Jaiver from the colleagues who had talked to him. So, knowing his previous works and that my ex-colleagues seemed to like him, I did what any sane PhD applicant should do: I sent an email to his PhD student, Alejo, asking about his experience with Javier. He was kind enough to send me a detailed reply telling me that Javier is an excellent person and researcher and that he really enjoys working with him. This made me feel certain that I want to work with Javier, and for that, I am very grateful to Alejo.

After I sent my first email to Javier (which ended up being the only PhD application I have ever sent, because I didn't consider anyone other than him), I had my first online meeting with him. The meeting lasted for more than three hours, and it was such an exhilarating experience. We talked about literally everything we were both interested in, sharing papers and ideas and imagining all the cool things we could do together. It was the first time I felt that my passion and curiosities were understood as a whole and mutually shared. In the end, I got the position, and in September 2017, my PhD journey started.

Now, looking back at all those years, I definitely feel that it was a much crazier roller-coaster ride than I had initially anticipated. When I think about the lows, I am glad that I had not known them beforehand, because I don't know if I would have still had the courage to take the first step if I had known the difficulties and challenges waiting ahead of me. In this sense, I believe that ignorance is bliss, although it may not be the most befitting thing to say in a PhD thesis. During this crazy roller-coaster ride, Javier was my safety bar (a well

functioning one, of course). No offense to the Spaniards, but when it comes to administrations, the Spanish system can give you one hell of time, especially if you are a foreigner who does not speak Spanish. Whenever I needed help, Javier took matters into his hands and took care of so many things for me, including the residence, university contract/administrations, scholarship applications, bank problems, etc. Without his countless rescue operations, I would have been still stuck in an infinite loop of administration hell to this day.

Also, Javier has given me a tremendous amount of trust and encouragement whenever I was having a hard time doing research. I still cannot believe that he allowed me to work on two-view triangulation after we had successfully published our first work on SLAM. Any other reasonable supervisor would have said that two-view triangulation was solved long time ago and it is not really interesting to anyone anymore. Certainly, if I were in his position, I would have said that to my student. But he didn't. Instead, he told me to take my time and give it enough thought until I am satisfied and ready to pursue a new direction. In fact, changing the research direction from visual SLAM to two-view triangulation was a risky move because I might have ended up spending a lot of time without finding anything new or useful. But Javier understood me and supported me when I told him that I wanted to work on a very small problem so that I can understand every little details and come up with something on my own from the scratch. In the end, this decision led to two publications on triangulation at top vision conferences, and possibly, a journal paper (which is still a work in progress). It also had a pivotal impact on my mindset, which helped me pursue new research directions without feeling like an imposter.

Javier was an amazing supervisor during not only the hard times, but also the good times. Undoubtedly, the best time we had together was the ICCV 2019 in Seoul. In Spain, I often felt like I was a tourist and Javier was my tour guide. In Korea, the roles were reversed and I finally got the chance to take the lead and introduce him to Korean culture and food. We went to so many places in Seoul and it was so fun! I am especially fond of the memory of us going to Itaewon on Halloween night, enjoying great South-African food, and dancing in a club together. How many PhD students get to do that with his/her supervisor? Honestly, I wouldn't exchange this experience with anything else, not even with a best paper award (although Javier might think otherwise).

(Okay, I promise this is the last paragraph about Javier.) There are so many things I have learned from Javier, especially from his attitude towards his students. During our weekly meetings, Javier was always attentive and supportive. Even when I didn't have any good results to show, I would often feel optimistic by the end of the meeting. He always gives me detailed and constructive feedback, and his comments are insightful in many ways, helping me think and question from new perspectives. When he expresses his concerns in rare occasions, he is always careful with his words, not to hurt my feeling. He is so humble and

often gives credit to others when there are achievements. From years of working with him, I really see that before he is a good supervisor, he is a good person. He genuinely cares about his students, and I can sincerely feel that. I am so lucky and grateful to have you as my supervisor, Javier. Thank you for being such an awesome mentor!

I also want to thank my former lab mates who have given me lots of help whenever I had problems in my daily life in Zaragoza. Thanks to them, I got to experience the fun part of living in Spain. I really enjoyed our time together and I feel sorry that I have not been able to fully reciprocate the favors I have received for so long. Thank you Chema, Berta, Carlos, Richard, Jose and Iñigo. You guys are amazing and I wish all of you all the best in your life and career! I also want to thank Leo for the interesting discussion we had on my triangulation method. Lastly, I want to mention two friends who made my life in Zaragoza much more enjoyable: Javier Dominguez, with whom I had a lot of fun talking about many interesting things and hanging out together, and my fellow Korean friend, Incheol, who connected with me on a personal level and gave me lots of warm memories in Zaragoza. I am glad that I have met you guys and that we had a great time together while we could.

I would also like to thank my old friends who have been providing me with constant support, even though they are living far away from me: Taehwan in Korea, Timmy and Wooyeong in the Netherlands, and Jongseok in Germany. To visit me, Taehwan, Timmy and Wooyeong travelled all the way to Belgium, and Jongseok to Zaragoza. Thank you guys for being such great friends and let us stay in touch no matter where we are. I am already looking forward to seeing you guys again!

I saved the most important people for last – my dear family. First, I want to thank my parents, my brother, and my grandparents.

사랑하는 엄마 아빠 그리고 성호야. 끝이 안 보이던 기나긴 박사과정에 드디어 마침표를 찍게 됐어. 그동안 항상 걱정해주고 응원해주고 같이 있을 때마다 잘 챙겨줘서 고마워. 우리가 어디에 있든 무엇을 하든 항상 서로를 사랑하고 응원하고 있다는 걸 기억하자. 무엇보다 건강이 최고니까 앞으로도 우리 가족 계속 건강하고 행복하자. 사랑해! 사랑하는 할머니, 할아버지. 큰손자 열심히 공부해서 이제 박사 됐어요! 항상 저와 가족을 위해 기도해주시는 할머니 덕분이고 멀리서도 응원해주시는 할아버지 덕분이에요. 두 분 모두 항상 건강하시고 오래오래 행복하게 사세요. 사랑해요!

Also, I want to thank my dear family in Lille: Kris, Andy, and my awesome siblings-in-law, Kyle and Shanesia. Especially, Kris has done so much for me since Liana was born. She not only takes care of Liana for several days a week, but also makes delicious food for me time after time, so that I can focus on my work without worrying about food. She is always so caring, and at the same time, fun to talk to. Without her constant care, love and support, my life would have been very hard for the past couple of years. I sincerely appreciate every-

thing you have been doing, Kris. Thank you for always being there for us when we needed you the most. Let's create a lot of happy memories together with everyone for years to come! Lots of love to you, from your best son-in-law!

Most importantly, there is one person, without whom nothing would have been possible in the first place. My dear Sinitta, my soulmate. Soon, it will be 10 years since we have been together. It is hard to believe that during half of that time, I have been doing a PhD. Five years ago when I left for Spain, I did not know how hard it would be to be away from you. I remember our countless video chats, in the lab and in the apartment. Hearing your voice and seeing your face was always a joy and my only source of energy. But I still feel so bad when I think of the times you told me in tears that you miss me and that you want me at your side. You would often say *'Take the plane and come here right now'* in a joking way, and I would reply *'There is no plane at this hour, so I will start walking to Belgium now'*. Then, we would just laugh together. Sinitta, I am so grateful that you let me pursue my ambition and that you supported my selfish decision all the way until the very end.

Sinitta. So many things happened during the pandemic: We started living together, got married, and now we have a beautiful baby. During this eventful period of time, you have always cared for me and cheered me up. Whenever I said I don't think I can make it, you would tell me *'Seong. Don't be such a *****. Stop whining and suck it up like a man'*. Weirdly, you know exactly the right time and the right way to say this magic sentence, and it always makes me laugh and gives me energy.

Sinitta. I know that your daily life is extremely tough. You started a new job that requires a long commute and you are also taking care of a 7-month-old clingy baby who wakes you up in the middle of the night. You are doing so much housework and you are also arranging everything regarding the house, bank, notaris, etc. You always shoulder the burden to give me enough time and space to work on my thesis, even when you are tired or ill (or both). Without your love and sacrifice, I wouldn't have been able to make it this far. Thank you so much. I love you more than anyone.

Finally, a message to Liana, mijn baby'tje: By the time you are able to read this message on your own, I guess you are no longer a baby anymore (although you will always be my baby'tje no matter how old you are). One thing I know for sure is that I will always miss this time, because right now I can lift your tiny body and hug you anytime I want. Then you would look me in the eye and show me the most beautiful smile in the world. Your existence alone is the greatest gift God has given me, and it makes everything I do meaningful. Every morning, you wake up smiling at me like the sunshine, and that is all I need to get through the day. Thank you for being a happy healthy baby. I cherish every little moment I have with you and I love you so much. My Liana, my little angel.

Abstract

In order for a computer to "see" the 3D geometry of the world, it needs to be able to derive the geometric relations between the 2D images and the 3D world. Multiple view geometry is the field of research that studies this problem. Many existing methods attempt to solve a small part of this big problem by minimizing a certain objective function. This function is typically made of algebraic or geometric errors representing the deviations of the observation from the model. In short, we often try to recover the 3D structure of the world and the camera poses by finding the model that minimizes the discrepancy with respect to the observations.

The focus of this thesis is placed mainly on two aspects of multiview reconstruction problems: the error criteria and the robustness. First, we study the error criteria used in several geometric problems and ask ourselves *'Why do we optimize what we optimize?'* Specifically, we analyze their pros and cons and come up with novel methods that either combine the existing criteria or adopt a better alternative. Second, we try to achieve state-of-the-art robustness against outliers and challenging scenarios that are often encountered in practice. To this end, we propose multiple novel ideas that can be incorporated in the optimization-based methods.

Specifically, we study the following problems: monocular SLAM, two-view and multi-view triangulation, single and multiple rotation averaging, rotation-only bundle adjustment, robust averaging of numbers, and quantitative trajectory evaluation.

For monocular SLAM, we propose a novel hybrid approach that combines the strengths of direct and feature-based methods. Direct methods minimize the photometric errors between corresponding pixels in images, whereas feature-based methods minimize the reprojection errors. Our method loosely couples direct odometry and feature-based SLAM, and it is shown to improve the robustness in challenging scenarios, as well as the accuracy when the camera motion is mostly loopy.

For two-view triangulation, we propose optimal methods that minimize the angular reprojection errors in closed form. Since the angular error is rotationally invariant, these methods can be used for perspective, fisheye or omnidirectional cameras. Also, they are much

faster than the existing optimal methods. Another two-view triangulation method we propose takes a completely different approach: We modify the classical midpoint method slightly and show that it provides a superior balance of 2D and 3D accuracy even though it is non-optimal.

For multiview triangulation, we propose a robust and efficient method using two-view RANSAC. We introduce several early termination criteria for two-view RANSAC using the midpoint method, and show that it improves the efficiency when the outlier ratio is high. Additionally, we show that the uncertainty of a triangulated point can be modeled as a function of three factors: the number of cameras, the mean reprojection error and the maximum parallax angle. By learning this model, the uncertainty can be interpolated at test time.

For single rotation averaging, we propose a robust method based on the Weiszfeld algorithm. The main idea is to start with a robust initialization and perform an implicit outlier rejection scheme inside the Weiszfeld algorithm to further boost the robustness. Also, we use an approximation of the chordal median on $SO(3)$ which provides a significant speedup.

For multiple rotation averaging, we propose HARA, a novel approach that incrementally initializes the rotation graph based on a hierarchy of triplet support. Essentially, we build a spanning tree by prioritizing the edges with many strong triplet supports and gradually adding those with weaker and fewer supports. As a result, we reduce the risk of adding outliers in the initial solution, which then allows us to filter outliers prior to nonlinear optimization. Furthermore, we show that we can improve the results by using the smoothed L_{0^+} function in the local refinement step.

Next, we propose rotation-only bundle adjustment, a novel method for estimating the absolute rotations of multiple views independently of the translations and the scene structure. The key is to minimize a specially designed cost function based on the normalized epipolar error, which is closely related to the L_1-optimal angular reprojection error among other geometric quantities. Our approach provides multiple benefits such as complete immunity to inaccurate translations and structure, robustness to pure rotations and planar scenes, and the accuracy improvement when used with rotation averaging.

We also propose RODIAN, a robust method for averaging a set of numbers contaminated by a large proportion of outliers. In our method, we assume that the outliers are uniformly distributed within the range of the data and we search for the region that is least likely to contain outliers only. We then take the median of the data within this region. Our method is fast, robust and deterministic, and it does not rely on a known inlier error bound.

Finally, for quantitative trajectory evaluation, we point out the weakness of the commonly used Absolute Trajectory Error (ATE) and propose a novel alternative named Discernible

Trajectory Error (DTE). In the presence of just a few outliers, the ATE quickly starts losing its sensitivity to the inlier trajectory error and the number of outliers. The DTE overcomes this weakness by aligning the estimated trajectory to the ground truth using a robust method based on several different types of medians. Using similar ideas, we also propose a rotation-only metric, named Discernible Rotation Error (DRE). Additionally, we propose a simple method for calibrating the camera-to-marker rotation, which is a prerequisite for the computation of both DTE and DRE.

Resumen

Para que un computador sea capaz de entender la geometría 3D de su entorno, necesitamos derivar las relaciones geométricas entre las imágenes 2D y el mundo 3D. La geometría de múltiples vistas es el área de investigación que estudia este problema. La mayor parte de métodos existentes resuelve pequeñas partes de este gran problema minimizando una determinada función objetivo. Estas funciones normalmente se componen de errores algebraicos o geométricos que representan las desviaciones con respecto al modelo de observación. En resumen, en general tratamos de recuperar la estructura 3D del mundo y el movimiento de la cámara encontrando el modelo que minimiza la discrepancia con respecto a las observaciones.

El enfoque de esta tesis se centra principalmente en dos aspectos de los problemas de reconstrucción multivista: los criterios de error y la robustez. Primero, estudiamos los criterios de error usados en varios problemas geométricos y nos preguntamos *'¿Por qué optimizamos lo que optimizamos?'* Específicamente, analizamos sus pros y sus contras y proponemos métodos novedosos que combinan los criterios existentes o adoptan una mejor alternativa. En segundo lugar, tratamos de alcanzar el estado del arte en robustez frente a valores atípicos y escenarios desafiantes, que a menudo se encuentran en la práctica. Para ello, proponemos múltiples ideas novedosas que pueden ser incorporadas en los métodos basados en optimización.

Específicamente, estudiamos los siguientes problemas: SLAM monocular, triangulación a partir de dos y de múltiples vistas, promedio de rotaciones únicas y múltiples, ajuste de haces únicamente con rotaciones de cámara, promedio robusto de números y evaluación cuantitativa de estimación de trayectoria.

Para SLAM monocular, proponemos un enfoque híbrido novedoso que combina las fortalezas de los métodos directos y los basados en características. Los métodos directos minimizan los errores fotométricos entre los píxeles correspondientes en varias imágenes, mientras que los métodos basados en características minimizan los errores de reproyección. Nuestro método combina de manera débilmente acoplada la odometría directa y el SLAM basado en características, y demostramos que mejora la robustez en escenarios desafiantes, así como la precisión cuando el movimiento de la cámara realiza frecuentes revisitas.

Para la triangulación de dos vistas, proponemos métodos óptimos que minimizan los errores de reproyección angular en forma cerrada. Dado que el error angular es rotacionalmente invariante, estos métodos se pueden utilizar para cámaras perspectivas, lentes de ojo de pez u omnidireccionales. Además, son mucho más rápidos que los métodos óptimos existentes en la literatura. Otro método de triangulación de dos vistas que proponemos adopta un enfoque completamente diferente: Modificamos ligeramente el método clásico del punto medio y demostramos que proporciona un equilibrio superior de precisión 2D y 3D, aunque no es óptimo.

Para la triangulación multivista, proponemos un método robusto y eficiente utilizando RANSAC de dos vistas. Presentamos varios criterios de finalización temprana para RANSAC de dos vistas utilizando el método de punto medio y mostramos que mejora la eficiencia cuando la proporción de medidas espúreas es alta. Además, mostramos que la incertidumbre de un punto triangulado se puede modelar en función de tres factores: el número de cámaras, el error medio de reproyección y el ángulo de paralaje máximo. Al aprender este modelo, la incertidumbre se puede interpolar para cada caso.

Para promediar una sola rotación, proponemos un método robusto basado en el algoritmo de Weiszfeld. La idea principal es comenzar con una inicialización robusta y realizar un esquema de rechazo de valores espúreos implícito dentro del algoritmo de Weiszfeld para aumentar aún más la robustez. Además, usamos una aproximación de la mediana cordal en $SO(3)$ que proporciona una aceleración significativa del método.

Para promediar rotaciones múltiples proponemos HARA, un enfoque novedoso que inicializa de manera incremental el grafo de rotaciones basado en una jerarquía de compatibilidad con tripletas. Esencialmente, construimos un árbol de expansión priorizando los enlaces con muchos soportes triples fuertes y agregando gradualmente aquellos con menos soportes y más débiles. Como resultado, reducimos el riesgo de agregar valores atípicos en la solución inicial, lo que nos permite filtrar los valores atípicos antes de la optimización no lineal. Además, mostramos que podemos mejorar los resultados usando la función suavizada L_{0^+} en el paso de refinamiento local.

A continuación, proponemos el ajuste de haces únicamente con rotaciones, un método novedoso para estimar las rotaciones absolutas de múltiples vistas independientemente de las traslaciones y la estructura de la escena. La clave es minimizar una función de coste especialmente diseñada basada en el error epipolar normalizado, que está estrechamente relacionado con el error de reproyección angular óptimo L_1 entre otras cantidades geométricas. Nuestro enfoque brinda múltiples beneficios, como inmunidad total a translaciones y triangulaciones imprecisas, robustez frente a rotaciones puras y escenas planas, y la mejora de la precisión

cuando se usa tras el promedio de promedio de rotaciones explicado anteriormente.

También proponemos RODIAN, un método robusto para promediar un conjunto de números contaminados por una gran proporción de valores atípicos. En nuestro método, asumimos que los valores atípicos se distribuyen uniformemente dentro del rango de los datos y buscamos la región que es menos probable que contenga solo valores atípicos. Luego tomamos la mediana de los datos dentro de esta región. Nuestro método es rápido, robusto y determinista, y no se basa en un límite de error interno conocido.

Finalmente, para la evaluación cuantitativa de la trayectoria, señalamos la debilidad del Error de Trayectoria Absoluta (ATE) comúnmente utilizado y proponemos una alternativa novedosa llamada Error de Trayectoria Discernible (DTE). En presencia de solo unos pocos valores espúreos, el ATE pierde su sensibilidad respecto al error de trayectoria de los valores típicos y respecto al número de datos atípicos o espúreos. El DTE supera esta debilidad al alinear la trayectoria estimada con la verdadera (ground truth) utilizando un método robusto basado en varios tipos diferentes de medianas. Usando ideas similares, también proponemos una métrica de solo rotación, llamada Error de Rotación Discernible (DRE). Además, proponemos un método simple para calibrar la rotación de cámara a marcador, que es un requisito previo para el cálculo de DTE y DRE.

Table of Contents

Chapter 1

Introduction

Multiple view geometry is a sub-field of computer vision dealing with the geometric relations between a 3D scene and multiple cameras that project such scene into 2D images. This field has been increasingly active in the past few decades due to its relevance in many emerging technologies, such as photogrammetry (Ondrúška et al., 2015), geo-localization (Moulon et al., 2017), autonomous navigation of ground/air vehicles (Qin et al., 2018), and virtual/augmented reality (Klein and Murray, 2007). As a result of such immense research endeavors during these decades, the field of multiple view geometry has seen significant progress in both theory and practice.

One of the most influential books in our field is *Multiple View Geometry*, written by R. I. Hartley and A. Zisserman in the early 2000s (Hartley and Zisserman, 2003). In the preface of their book, they mention two important factors that led to the great achievements in multiple view geometry at that time:

> *"The achievements in multiple view geometry have been possible because of developments in our theoretical understanding, but also because of improvements in estimating mathematical objects from images. The first improvement has been an attention to the error that should be minimized in over-determined systems – whether it be algebraic, geometric or statistical. The second improvement has been the use of robust estimation algorithms (such as RANSAC), so that the estimate is unaffected by "outliers" in the data. Also these techniques have generated powerful search and matching algorithms."*

More than 20 years later since this book was published, these two topics (the error criteria and robust estimation) are still actively studied to this day and progress is being made steadily. In this thesis, we also focus on these two aspects of multiview geometry problems. The two overarching themes of this thesis are (1) the study of error criteria and cost functions, and (2) the robust estimation in the presence of outliers:

(1) Study of error criteria and cost functions:
We ask ourselves *'Why do we optimize what we optimize in multiple view geometry?'*, which is also the title of this thesis. Many existing methods in multiview reconstruction aim to recover the underlying geometric model of the world and the cameras by minimizing a certain objective function. Typically, this function is based on algebraic or geometric errors that represent the deviations of the observation from the model we are guessing. Depending on the choice and design of this function, we obtain different results and encounter different levels of difficulty solving the problem (Hartley and Zisserman, 2003). In this thesis, we analyze the role of error criteria and cost functions in several geometric problems, namely monocular simultaneous localization and mapping (SLAM), triangulation, single/multiple rotation averaging, rotation-only bundle adjustment and quantitative trajectory evaluation.

(2) Robust estimation in the presence of outliers:
When outliers in the data are not handled properly, they significantly reduce the resulting accuracy in most geometric reconstruction problems (Stewart, 1999, Antonante et al., 2022). For applications such as autonomous driving, unsuccessful handing of outliers poses a critical risk that can endanger our life and safety. For applications such as AR and VR, this can instantly ruin the user experience. Therefore, it is essential that we always take into account potential outliers when we solve an estimation problem. In this thesis, we propose several novel ideas to better handle outliers in multiview triangulation, single/multiple rotation averaging, number averaging, and quantitative trajectory evaluation.

1.1 Research Areas and Contributions

Multiple view geometry is a very broad field covering a number of diverse research and application areas. In this thesis, we focus on two main research areas: Monocular Simultaneous Localization and Mapping (SLAM) and Structure-from-Motion (SfM). Both areas consist of multiple components, each of which can be considered as a research area itself. Here we provide the complete list of the problems we consider in this thesis:

- Monocular SLAM
- Structure-from-Motion (SfM)
- Two-view and multiview triangulation
- Single and multiple rotation averaging
- Rotation-only bundle adjustment
- Number averaging
- Quantitative trajectory evaluation

In the following, we briefly explain each of the problems above and our contributions therein.

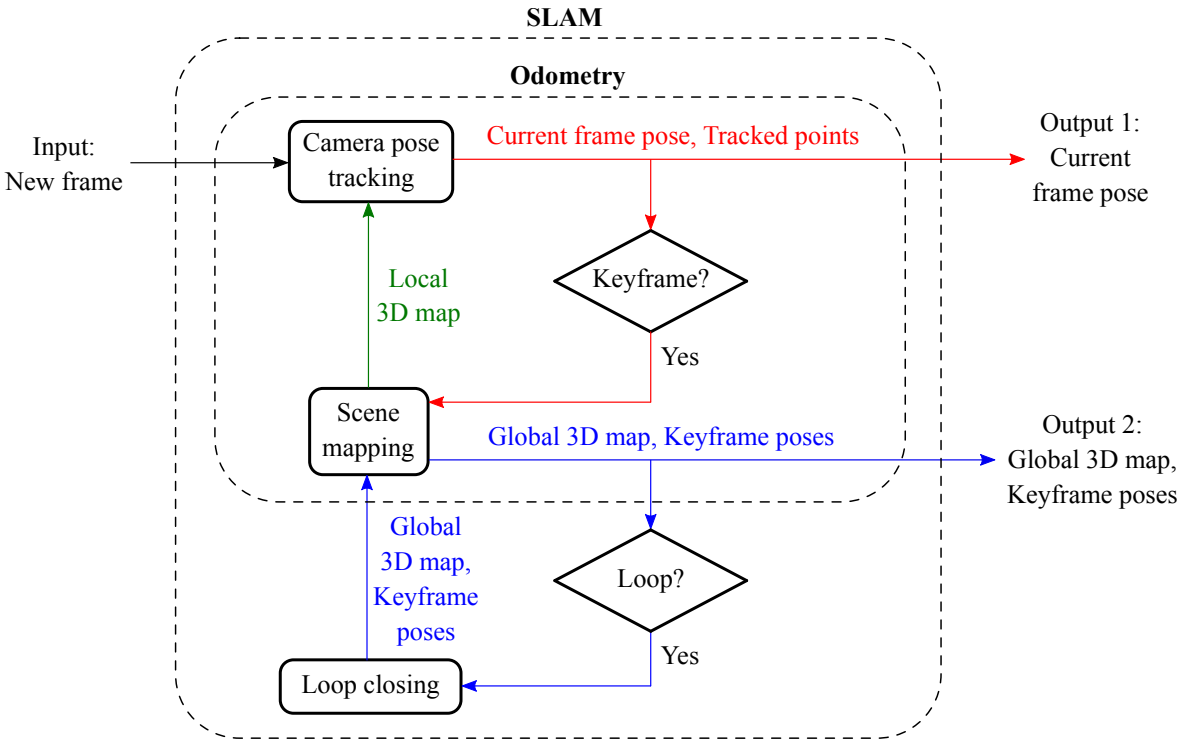


Figure 1.1: The pipeline of a keyframe-based monocular SLAM system.

1.1.1 Monocular SLAM

Monocular SLAM is a problem of simultaneously estimating the motion of a single camera and the 3D scene structure from a real-time video input. There are two main approaches for monocular SLAM: filter-based and keyframe-based. Filter-based methods such as (Davison et al., 2007) include the camera pose and 3D feature positions in a single state vector and update it sequentially using an Extended Kalman Filter (EKF). In contrast, keyframe-based methods such as (Klein and Murray, 2007) select a subset of past frames called *keyframes* and jointly optimize their poses and the 3D map, so that the resulting map can be used for real-time camera tracking. In (Strasdat et al., 2010*a*), it is shown that keyframe-based methods tend to be more accurate than the filter-based methods given a sufficient amount of processing resources. In this thesis, we focus on the keyframe-based approach. Fig. 1.1 presents a generic pipeline of a keyframe-based monocular SLAM system (*e.g.*, (Strasdat et al., 2010*b*, 2011, Mur-Artal et al., 2015)). Interested readers are referred to (Younes et al., 2017) for a survey of keyframe-based methods.

The keyframe-based methods can be further classified into three categories: feature-based (or indirect), photometric (or direct) and hybrid (or semi-direct). Feature-based methods estimate structure and motion by matching 2D features and performing geometric bundle adjustment, which minimizes the reprojection error (*e.g.*, (Klein and Murray, 2007, Mur-Artal et al., 2015)). Direct methods, on the other hand, estimate structure and motion by minimizing the photometric error (*i.e.*, intensity difference) between corresponding pixels in images (*e.g.*, (Newcombe et al., 2011, Engel et al., 2014, 2018)). Finally, hybrid methods

estimate structure and motion by combining feature-based and direct methods, taking advantage of the complementary strengths of each method (*e.g.*, (Forster et al., 2014, Forster, Zhang, Gassner, Werlberger and Scaramuzza, 2017, Krombach et al., 2016, Younes et al., 2019, Luo et al., 2021, 2022)).

Our contributions: In Chapter 2, we propose LCSD-SLAM, a novel hybrid method for monocular SLAM that inherits both the robustness of direct visual odometry and the map-reusing capability of feature-based SLAM. The novelty of our method lies in the *loose coupling* of direct and feature-based systems such that, locally, a direct method is used to track new frames rapidly and robustly with respect to a local semi-dense map, and globally, a feature-based method is used to refine the keyframe poses, perform loop closures, and build a globally consistent, reusable, sparse feature map. Our evaluation shows that LCSD-SLAM achieves state-of-the-art results on two public benchmark datasets.

1.1.2 Structure-from-Motion (SfM)

SfM refers to the problem of reconstructing the camera poses and the scene structure from the list of unordered images. Unlike visual odometry and SLAM, the input images are often taken from different cameras with different calibration parameters. Also, it is impossible to use the temporal data to relate successive frames to each other because the input is not a video. Over the years, various SfM approaches have been proposed, including incremental (Snavely et al., 2006, Agarwal et al., 2011, Schönberger and Frahm, 2016), hierarchical (Gherardi et al., 2010) and global approaches (Moulon et al., 2013, Wilson and Snavely, 2014, Cui and Tan, 2015). In this thesis, we consider the global approach, which has the following pipeline:

1. Feature extraction and matching across all images (*e.g.*, using SIFT descriptors (Lowe, 2004)).
2. Relative pose estimation between image pairs (*e.g.*, using Nistér's five-point method (Nistér, 2004) in RANSAC (Fischler and Bolles, 1981)).
3. Global rotation estimation by multiple rotation averaging.
4. Global translation estimation.
5. Multiview triangulation.
6. Bundle adjustment, *i.e.*, a joint optimization of the camera poses and the 3D points.

Our contributions: In this thesis, we study Step 3 and 5 in the pipeline above. We also propose rotation-only bundle adjustment as an additional intermediate step between Step 3 and 4. Fig. 1.2 illustrates the proposed pipeline of a global SfM system.

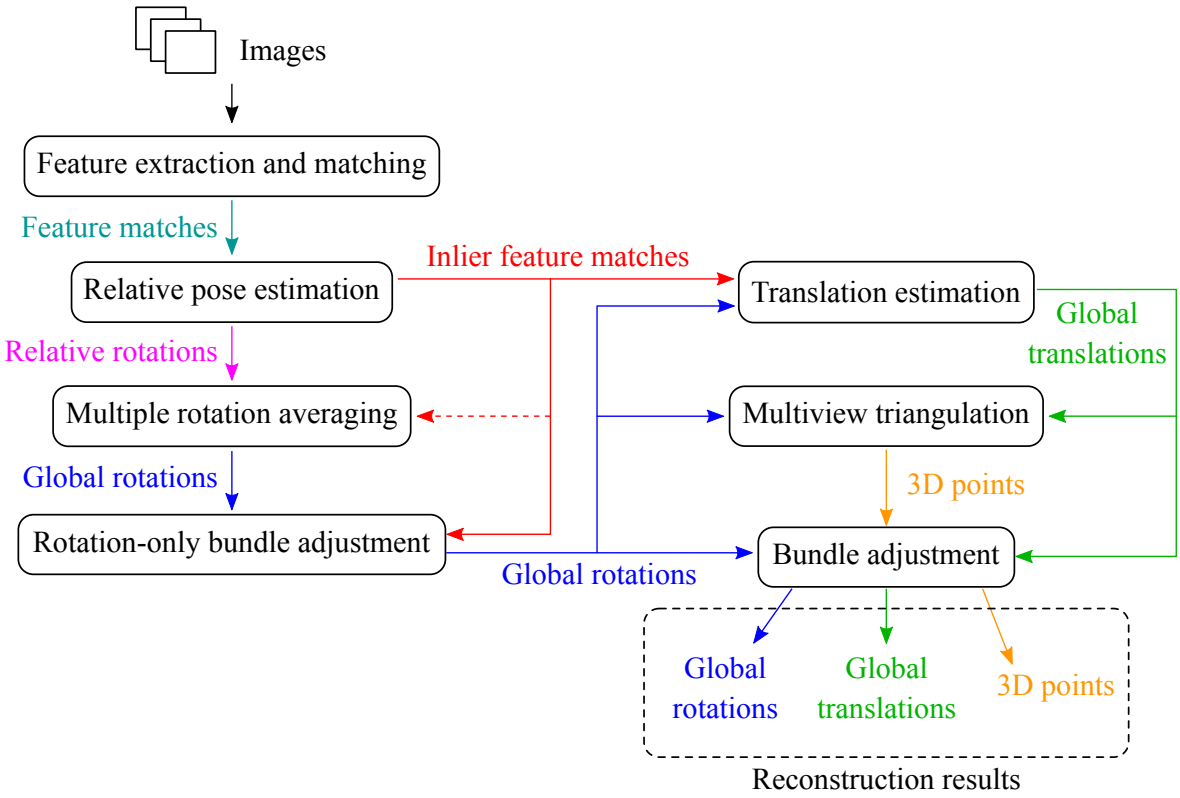


Figure 1.2: The proposed pipeline of a global SfM system.

1.1.3 Two-View Triangulation

Triangulation is the problem of estimating the position of a 3D point given its projections in two or more calibrated cameras of known pose (see Fig. 1.3). This is a fundamental problem in computer vision, and it plays an important role in geometric reconstruction, such as SLAM (Mur-Artal et al., 2015) and SfM (Schönberger and Frahm, 2016). As shown in Fig. 1.2, triangulation provides the initial seed of the 3D points for the bundle adjustment step that follows. Bundle adjustment aims to find the optimal structure and camera poses jointly by minimizing the image reprojection errors (Triggs et al., 2000). Being a nonlinear optimization, it requires a good initialization since it mostly converges to a local minimum near the starting point. Therefore, accurate triangulation is needed not only for the accurate scene reconstruction, but also for the accurate camera pose estimation in bundle adjustment.

The minimum number of views required for triangulation is two. Compared to triangulation from three or more views, two-view triangulation is considered relatively simple and straightforward. Nevertheless, its importance must not be understated, as it has unique applications in SfM (Schönberger and Frahm, 2016, Cui and Tan, 2015, Cai et al., 2021).

Our contributions: In Chapter 3, we propose, for the first time to our knowledge, the exact closed-form solutions to the L_1 and L_∞ optimal triangulation from two views based on the angular reprojection error. We also present our own deviation of the L_2 optimal solution that is much more compact and geometrically intuitive than the existing one (Oliensis, 2002).

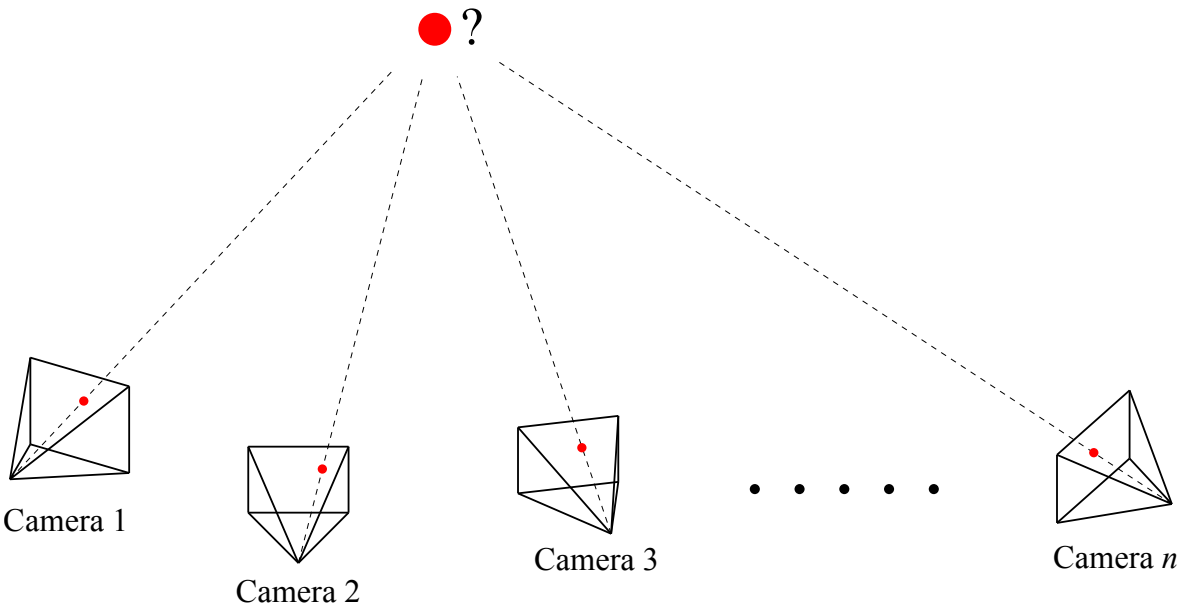


Figure 1.3: Synopsis of the triangulation problem. The unknown 3D point's position can be estimated from its projections in n views of known calibration and pose.

Since all three methods are based on the angular error, they can be applied for any type of central cameras, be it perspective, fisheye or omnidirectional. Our experiments show that our methods do indeed achieve global optimality under respective cost functions and that they are faster than the existing optimal methods.

In Chapter 4, we propose an alternative method which does not minimize geometric errors at all. This method is a variant of the classical midpoint method (Beardsley et al., 1994, 1997, Hartley and Sturm, 1997), and we show that for small parallax angles it outperforms the classical midpoint method in terms of 2D accuracy and the optimal methods in terms of 3D accuracy. We provide an in-depth discussion on the discrepancy between the 2D and 3D errors at low parallax, which was previously reported in (Hartley and Sturm, 1997).

1.1.4 Multiview Triangulation

For multiview triangulation with more than two views (as shown in Fig. 1.3), the following practical aspects should be considered: 2D and 3D accuracy, robustness to outliers and speed. Arguably, the most critical aspect is the robustness to outliers because, in practice, feature tracks often contain a very large number of outliers (Schönberger and Frahm, 2016) and failing to filter them out is detrimental to the reconstruction accuracy. (Schönberger and Frahm, 2016) proposes a robust and efficient triangulation method using two-view RANSAC.

Our contributions: In Chapter 6, we propose a robust and efficient method for multiview triangulation, also based on two-view RANSAC. Specifically, we propose several early termination criteria for two-view RANSAC using the classical midpoint method (Beardsley et al., 1994, 1997, Hartley and Sturm, 1997), and show that it improves the efficiency when

the outlier ratio is high. We also compare three different local optimization methods in terms of 2D and 3D accuracy. Lastly, we propose a novel method for estimating the uncertainty of a triangulated point. We show that the uncertainty can be modeled as a function of three factors: the number of inlier cameras, the mean reprojection error and the maximum parallax angle. Learning this model then allows us to quickly interpolate the uncertainty at test time.

1.1.5 Single Rotation Averaging

Single rotation averaging refers to the problem of averaging several noisy estimates of a single rotation to obtain its best estimate (see Fig. 1.4). This is relevant in relative pose estimation (Hartley et al., 2011) and multiple rotation averaging (Hartley and Schaffalitzky, 2004, Lee and Civera, 2022), both of which play an important role in geometric reconstruction such as SLAM and SfM. We refer to (Hartley, Trumpf, Dai and Li, 2013) for an excellent study of single rotation averaging from a theoretical perspective. In (Hartley et al., 2011), an outlier-robust method is proposed based on the geodesic median in $SO(3)$ using the Weiszfeld algorithm (Weiszfeld, 1937).

Our contributions: In Chapter 7, we propose a novel method for robust single rotation averaging, also based on the Weiszfeld algorithm (Weiszfeld, 1937). Unlike (Hartley et al., 2011), however, we implicitly disregard the outliers at each iteration of the Weiszfeld algorithm, and also, we use an approximation of the chordal median in $SO(3)$ instead of the geodesic median. Our evaluation shows that our method outperforms (Hartley et al., 2011) in terms of speed and robustness.

1.1.6 Multiple Rotation Averaging

Multiple rotation averaging (also called rotation synchronization) aims to find multiple global rotations $\mathbf{R}_i$ given noisy estimates of the relative rotations between some camera pairs $\mathbf{R}_{ij} = \mathbf{R}_i\mathbf{R}_j^\top$ (Hartley, Trumpf, Dai and Li, 2013). This problem is illustrated in Fig. 1.5. It provides the initial estimates of the global camera rotations, which directly affect all of the subsequent operations in a global SfM pipeline (see Fig. 1.2). Therefore, multiple rotation averaging plays a crucial role in achieving reliable reconstruction and it needs to be sufficiently robust to outliers in order to mitigate the propagation and amplification of large errors down the SfM pipelines. State-of-the-art methods such as (Chatterjee and Govindu, 2018) and (Shi and Lerman, 2020) formulate rotation averaging as a non-linear optimization problem and solve it iteratively starting from some initial guess of the global rotations.

Our contributions: In Chapter 8, we propose HARA, a hierarchical approach for robust multiple rotation averaging. Our main contribution is a hierarchical initialization scheme that incrementally builds a spanning tree of a rotation graph by propagating most reliable edges first and less reliable ones later. The hierarchy of reliability is established based on

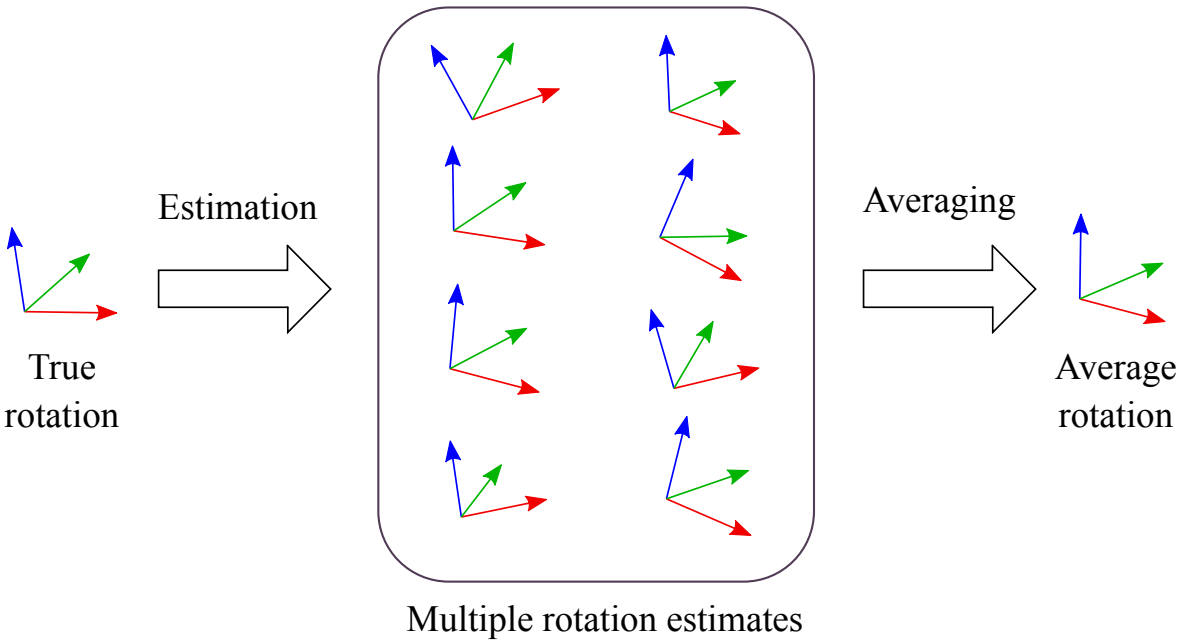


Figure 1.4: Synopsis of the single rotation averaging problem. The unknown rotation can be estimated by averaging multiple noisy estimates of it.

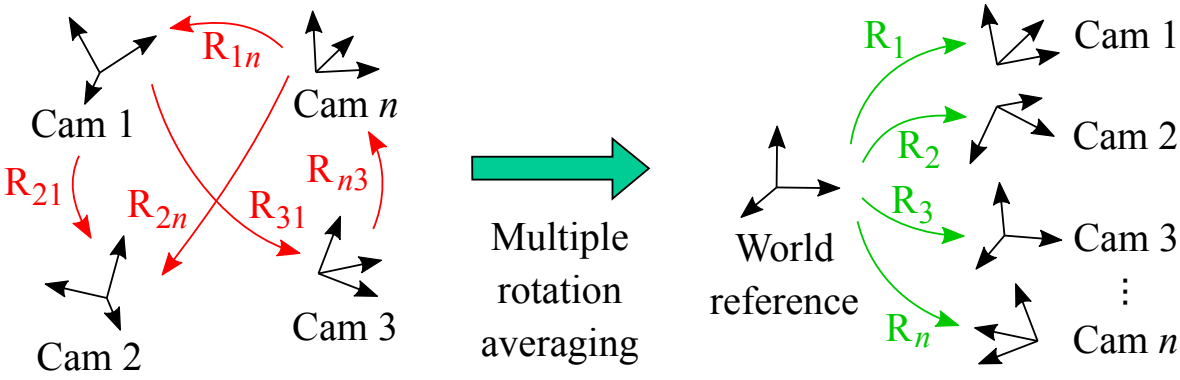


Figure 1.5: Synopsis of the multiple rotation averaging problem. The unknown global rotations can be estimated by combining noisy estimates of the relative rotations between multiple camera pairs.

the number of consistent triplet constraints, as well as their level of consistency. We also show that we can optionally incorporate the number of valid inlier feature matches into our initialization scheme (indicated by the dotted line in Fig. 1.2). Having a robust initial solution enables us to filter outlier edges prior to nonlinear optimization, and experimental results show that HARA achieves state-of-the-art results on both synthetic and real datasets. Additionally, in Chapter 9, we show that using the smoothed L_{0+} function (Peng et al., 2022) in the local refinement step of HARA further improves the results on the real datasets. We provide the probabilistic interpretations of this cost function and show that it is suitable for the outlier-prone large-scale rotation averaging problem.

1.1.7 Rotation-Only Bundle Adjustment

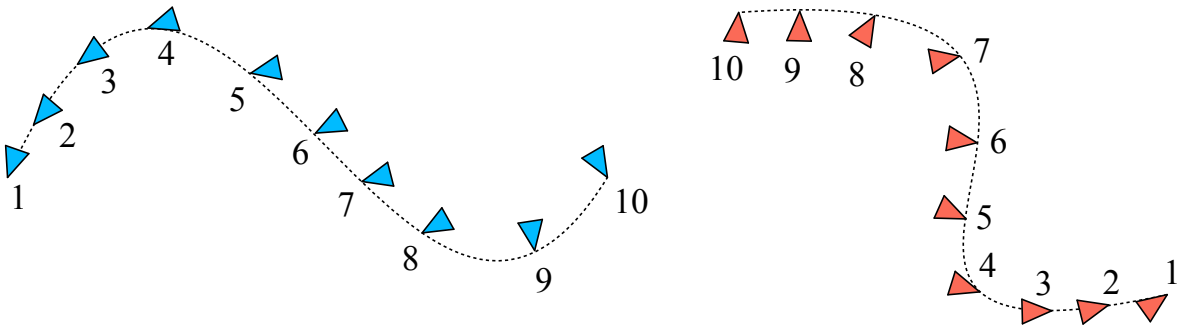
Rotation-only bundle adjustment is the problem of finding the global rotations of multiple cameras by directly leveraging the image measurements without computing the global translations or the scene structure. Since the estimated rotations are subsequently used for the translation estimation, multiview triangulation and bundle adjustment (see Fig. 1.2), it would be helpful if they can be further refined immediately after multiple rotation averaging.

Our contributions: In Chapter 10, we propose a novel method for rotation-only bundle adjustment. To the best of our knowledge, we are the first to propose a multiview rotation-only optimization method that uses the image measurements as direct input and can be generalized to both pure and non-pure rotations. The main idea is to extend the two-view rotation-only method of (Kneip and Lynen, 2013) and minimize the aggregated cost using the Adam optimizer (Kingma and Ba, 2015). Extensive evaluations on both synthetic and real datasets show that our method consistently and significantly improves the rotation accuracy when used after multiple rotation averaging. Additionally, in Chapter 5, we provide several geometric interpretations of the normalized epipolar error, which is what our aggregated cost is based on.

1.1.8 Number Averaging

Averaging numbers in the presence of a large amount of outliers is a simple yet relevant problem in a wide variety of domains, including 3D computer vision (Gesto Diaz et al., 2015, Lee and Civera, 2020*b*, Cui and Tan, 2015) among others. Commonly, the median is used for averaging outlier-prone numbers because it has a higher breakdown point (50%) than the arithmetic mean (0%). If there is an efficient method with a breakdown point even higher than 50%, then such a method could be used instead of the median, and this change could potentially improve the robustness of some of the existing reconstruction algorithms (*e.g.*, a translation averaging method proposed in (Cui and Tan, 2015)).

Our contributions: In Chapter 11, we propose RODIAN, a novel measure of central tendency that can handle more than 50% outliers in the data. Our method is inspired by MINPRAN (Stewart, 1995), but unlike MINPRAN, RODIAN is deterministic and runs in time $O(n \log n)$. Also, unlike RANSAC (Fischler and Bolles, 1981) and Huber-like cost functions (Huber, 1981), no parameter tuning is needed to account for different inlier distributions. Our extensive simulations demonstrate the superior robustness of RODIAN compared to the median and the least-median-of-squares (LMedS) (Rousseeuw, 1984).



(a) Ground truth (b) Estimation

Figure 1.6: Quantitative trajectory evaluation consists of two main steps: (1) Align the estimated trajectory (right) to the ground truth (left) as closely as possible, and (2) compute the distances between each corresponding camera pair and aggregate them into a single error metric.

1.1.9 Quantitative Trajectory Evaluation

By quantitative trajectory evaluation, we refer to the quantification of the deviation of the estimated camera poses (also called trajectory) from the known ground truth. This is an important problem because a reliable and fair evaluation is crucial not only for benchmarking purposes but also for progressive improvements of existing methods. The basic mechanism of trajectory evaluation is to align the ground-truth and estimated trajectories as closely as possible, and then aggregate the offsets between each corresponding camera pair into a single metric (see Fig. 1.6). For odometry and SLAM, a commonly used error metric is the Absolute Trajectory Error (ATE) (Sturm et al., 2012).

Our contributions: In Chapter 12, we point out a limitation of the ATE, which is that its sensitivity quickly deteriorates in the presence of just a few outliers and that it fails to robustly discern the varying accuracy as the inlier trajectory error or the number of outliers varies. To overcome this limitation, we propose an alternative error metric, named Discernible Trajectory Error (DTE). Our extensive simulations demonstrate that the DTE behaves much more favorably than the ATE in terms of sensitivity to the aforementioned factors. Additionally, we propose a rotation-only metric, named Discernible Rotation Error (DRE), and a novel calibration method that can be used for computing our metrics.

1.2 List of Publications

Some of the works presented in this thesis were peer-reviewed and published in prestigious academic conferences and journals. Among the remaining works, some were made public in the form of preprints, although they have not been published in any conferences or journals (yet). Here is the complete list of our works in order of appearance in this thesis:

- Seong Hun Lee and Javier Civera, **Loosely-Coupled Semi-Direct Monocular SLAM**, in IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 399-406, 2019,
- Seong Hun Lee and Javier Civera, **Closed-Form Optimal Two-View Triangulation Based on Angular Errors**, Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 2681-2689, 2019,
- Seong Hun Lee and Javier Civera, **Triangulation: Why Optimize?**, British Machine Vision Conference (BMVC), 2019,
- Seong Hun Lee and Javier Civera, **Geometric Interpretations of the Normalized Epipolar Error**, arXiv preprint, arXiv:2008.01254, 2020,
- Seong Hun Lee and Javier Civera, **Robust Uncertainty-Aware Multiview Triangulation**, arXiv preprint, arXiv:2008.01258, 2020,
- Seong Hun Lee and Javier Civera, **Robust Single Rotation Averaging**, arXiv preprint, arXiv:2004.00732, 2020,
- Seong Hun Lee and Javier Civera, **HARA: A Hierarchical Approach for Robust Rotation Averaging**, IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 15756-15765, 2022,
- Seong Hun Lee and Javier Civera, L_{0+} **Optimization: Application to Multiple Rotation Averaging**, unpublished, 2022,
- Seong Hun Lee and Javier Civera, **Rotation-Only Bundle Adjustment**, IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 424-433, 2021,
- Seong Hun Lee and Javier Civera, **RODIAN: Robustified Median**, arXiv preprint, arXiv:2206.02570, 2022,
- Seong Hun Lee and Javier Civera, **What's Wrong with the Absolute Trajectory Error?**, unpublished, 2022.

1.3 Code Released

- Loosely-Coupled Semi-Direct Monocular SLAM
  `https://github.com/sunghoon031/LCSD_SLAM`
- Robust Uncertainty-Aware Multiview Triangulation:
  `https://github.com/sunghoon031/RobustUncertaintyAwareMultiviewTriangulation`
- Robust Single Rotation Averaging
  `https://github.com/sunghoon031/RobustSingleRotationAveraging`
- HARA: A Hierarchical Approach for Robust Rotation Averaging
  `https://github.com/sunghoon031/HARA`
- Rotation-Only Bundle Adjustment
  `https://github.com/sunghoon031/ROBA`
- RODIAN: Robustified Median
  `https://github.com/sunghoon031/RODIAN`

1.4 Videos and Presentations

- Loosely-Coupled Semi-Direct Monocular SLAM
  `https://www.youtube.com/watch?v=j7WnU7ZpZ8c`
- Triangulation: Why Optimize?
  `https://youtu.be/K-d4EDQCpHk?t=18`
- Rotation-Only Bundle Adjustment
  `https://youtu.be/JXnEwXwVKus`
- HARA: A Hierarchical Approach for Robust Rotation Averaging
  `https://youtu.be/oAR-LMStRS4`

Chapter 2

Loosely-Coupled Semi-Direct Monocular SLAM

In this chapter, we propose a novel semi-direct approach for monocular simultaneous localization and mapping (SLAM) that combines the complementary strengths of direct and feature-based methods. The proposed pipeline loosely couples direct odometry and feature-based SLAM to perform three levels of parallel optimizations: (1) photometric bundle adjustment (BA) that jointly optimizes the local structure and motion, (2) geometric BA that refines keyframe poses and associated feature map points, and (3) pose graph optimization to achieve global map consistency in the presence of loop closures. This is achieved in real-time by limiting the feature-based operations to marginalized keyframes from the direct odometry module. Exhaustive evaluation on two benchmark datasets demonstrates that our system outperforms the state-of-the-art monocular odometry and SLAM systems in terms of overall accuracy and robustness.

2.1 Introduction

Real-time visual odometry (VO) and simultaneous localization and mapping (SLAM) play an important role in many emerging technologies, such as autonomous ground/air vehicles (Nistér et al., 2006, Lee and de Croon, 2018) and virtual/augmented reality (Klein and Murray, 2007). In particular, monocular methods have drawn significant attention due to their minimal hardware constraints.

Traditional algorithms relied heavily on feature extraction and matching to estimate structure and motion (Davison et al., 2007, Klein and Murray, 2007). In recent years, however, direct methods have gained rapidly increasing popularity (Engel et al., 2014, 2018). In contrast to feature-based ones, direct methods are capable of leveraging raw photometric information from a chosen set of pixels in the image. This removes the need for costly per-frame feature extraction and matching. Also, they are shown to be relatively more robust in low-texture scenes (Engel et al., 2018).

Although direct methods have their own merits in several aspects, they inevitably miss certain benefits of salient features. For example, feature descriptors such as SIFT (Lowe, 2004) or ORB (Rublee et al., 2011) have a high degree of invariance to viewpoint and illumination changes, and they can be matched over wide baselines. Such properties are favorable for tracking large inter-frame motions and recognizing revisited places. Recent studies indeed confirm that direct and feature-based methods have their own strengths and weaknesses in respective areas (Engel et al., 2018, Yang et al., 2018). Semi-direct methods such as (Forster et al., 2014) attempt to take advantage of such complementary characteristics by incorporating ideas from both direct and feature-based methods.

In this chapter, we propose a novel semi-direct approach for monocular SLAM that inherits both the robustness of direct VO and the map-reusing capability (*e.g.*, loop closure) of feature-based SLAM. Our contribution is a *loose coupling* between direct and feature-based algorithms such that:

1. Locally, a direct method is used to track the camera pose rapidly and robustly with respect to a locally accurate, short-term, semi-dense map.
2. Globally, a feature-based method is used to refine the keyframe poses, perform loop closures, and build a globally consistent, long-term, sparse feature map that can be reused.

This strategy allows us to complement the weaknesses of each method without compromising their real-time efficiency and performance. We implement our approach on top of DSO (Engel et al., 2018) and ORB-SLAM (Mur-Artal et al., 2015), respectively the state-of-the-art in direct and feature-based methods, and demonstrate that our system outperforms both of them on two public benchmark datasets. Fig. 2.1 shows an example snapshot of the estimated camera trajectory and associated scene reconstruction using our method. The full reconstruction process is demonstrated in the accompanying video:

`https://youtu.be/j7WnU7ZpZ8c`

2.2 Related Work

Modern keyframe-based VO/SLAM systems can be categorized into three classes:

(1) Feature-based: Feature-based (or indirect) methods recover both camera pose and scene structure by matching features and performing *geometric* bundle adjustment (BA) that minimizes the reprojection error. PTAM (Klein and Murray, 2007) is one of the most representative systems of this type, where it was first proposed to split tracking and mapping into two parallel threads. As of this writing, ORB-SLAM (Mur-Artal et al., 2015) is one of the best-performing feature-based systems. Based on multiple successful ideas of PTAM and others (Galvez-López and Tardos, 2012, Strasdat et al., 2010*b*, 2011), ORB-SLAM uses ORB features (Rublee et al., 2011) to perform tracking, mapping, relocalization and loop

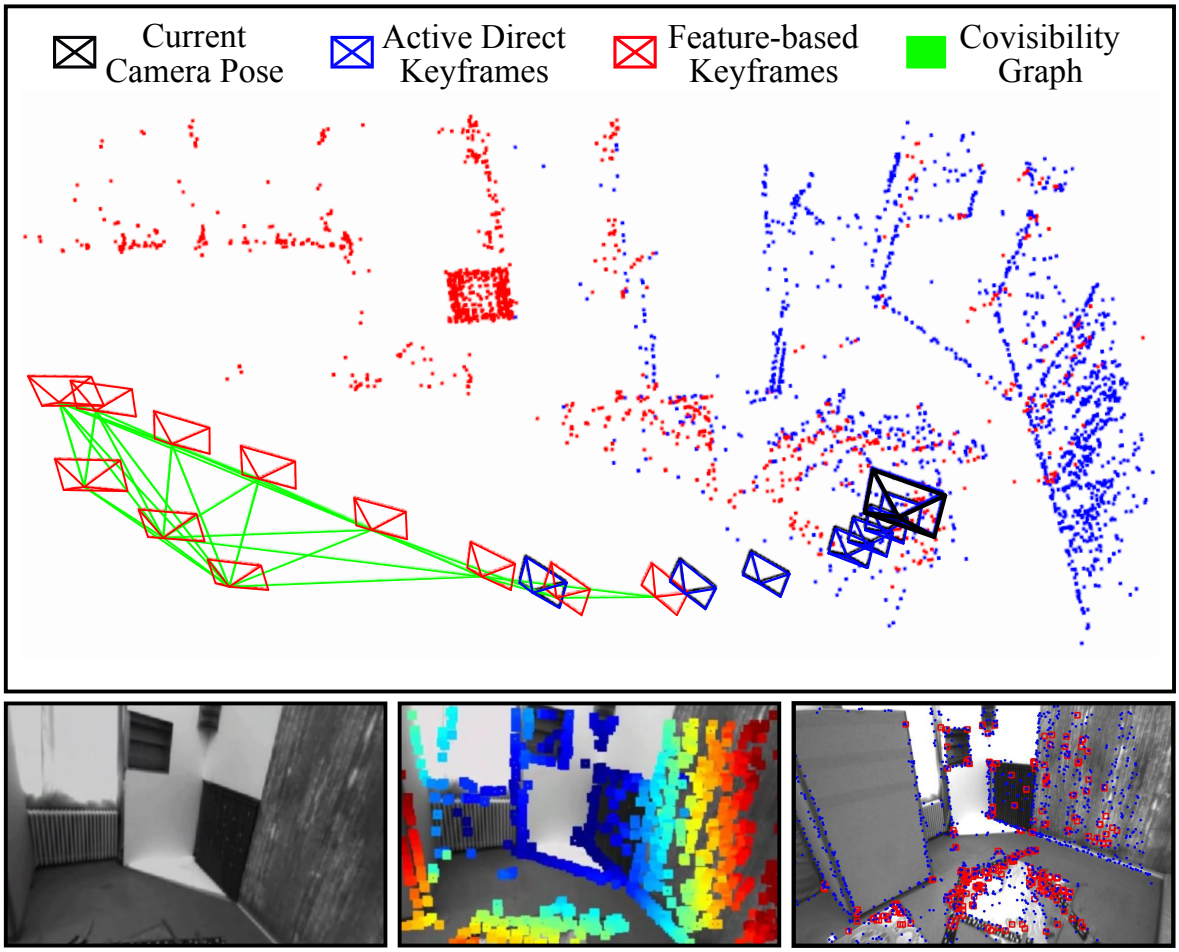


Figure 2.1: **Top:** We combine a direct and a feature-based method for monocular SLAM: the former is used for tracking and reconstructing a short-term local map (blue), and the latter for building a reusable global map (red and green). **Bottom:** (from left to right) the current frame, the latest direct keyframe with color-coded depths, and the latest feature-based keyframe with the matched features (red) and the projection of direct map points (blue).

closing in a scalable manner. In (Mur-Artal and Tardos, 2015), an extension of ORB-SLAM was proposed to generate a semi-dense reconstruction of the scene. This last method, however, does not use the resulting semi-dense map for tracking.

(2) Direct: Direct methods estimate structure and motion by minimizing the photometric error (*i.e.*, intensity difference) between corresponding pixels in images (Irani and Anandan, 1999). Unlike most feature-based methods, direct methods are not limited to a sparse map and can maintain either a sparse (Engel et al., 2018), semi-dense (Engel et al., 2013, 2014) or dense (Newcombe et al., 2011, Greene et al., 2016) map in real-time. As of this writing, one of the best-performing odometry systems is DSO (Engel et al., 2018) which performs *photometric* BA to jointly optimize camera intrinsics, extrinsics and inverse depths of sparse (or semi-dense) points in a sliding window fashion. In (Platinsky et al., 2017), it was found that, for a small number of map points (*e.g.*, < 1000), such joint optimization tends to be more accurate than alternating tracking and mapping as in, for example, LSD-SLAM (Engel et al., 2014). DSO is also the first work to demonstrate the benefits of photometric calibration

(Engel et al., 2016) in direct methods. However, it is subject to drift over time as it is a pure odometry method and does not reuse the map points that leave the field of view (FOV). (Gao et al., 2018) presented a modified DSO with the loop-closing capability similar to LSD-SLAM. Unlike our method, however, their system does not produce a reusable sparse feature map which is useful for applications such as global relocalization, fixed-map tracking and collaborative mapping.

(3) Semi-Direct: Semi-direct (or hybrid) methods estimate camera poses using both direct and feature-based methods. For example, SVO (Forster et al., 2014) performs direct sparse image alignment to estimate the initial guess of the camera pose and feature correspondences. Afterwards, it performs geometric BA to refine the pose and structure. It was shown in (Pizzoli et al., 2014) that SVO could also be used for dense mapping. In (Forster, Zhang, Gassner, Werlberger and Scaramuzza, 2017), several improvements to the original SVO were proposed. Although this system is highly efficient, it was shown to be less robust (Forster, Zhang, Gassner, Werlberger and Scaramuzza, 2017, Yang et al., 2018) than ORB-SLAM (Mur-Artal et al., 2015) and DSO (Engel et al., 2018). In (Li et al., 2019) and (Bu et al., 2017), similar approaches to SVO were proposed for monocular visual(-inertial) and RGB-D SLAM, respectively. Both methods adopt a direct method for tracking and a feature-based method for keyframe pose refinement, mapping and loop closing. At the end of Section 2.3, we briefly discuss how our method differs from these existing semi-direct methods. In (Krombach et al., 2016) and (Kim et al., 2018), different semi-direct approaches were proposed for stereo odometry. Both methods use feature-based tracking to obtain a motion prior, and then perform direct semi-dense or sparse alignment to refine the camera pose. While they were respectively shown to perform well against large inter-frame motions and illumination changes, they do not utilize the robustness of direct tracking in low-texture scenes.

2.3 System Overview

Fig. 2.2 illustrates our proposed semi-direct pipeline. The idea is to combine the currently best-performing feature-based and direct algorithms, namely ORB-SLAM (Mur-Artal et al., 2015) and DSO (Engel et al., 2018), with some modifications. To achieve real-time performance, we take inspiration from SVO (Forster et al., 2014) and apply a direct method to quickly track each frame and provide an initial seed for feature-based map optimization. Specifically, we use DSO to achieve real-time tracking and a modified version of ORB-SLAM to build a globally consistent map at a slower rate with marginalized keyframes from DSO. This is shown in Fig. 2.2 as a direct and a feature-based module, respectively. As the two separate asynchronous modules exchange information without sharing states, this approach is considered *loosely coupled*.

Our system architecture involves three different layers of optimization windows. At the most local level, a sliding window of keyframes and map points are photometrically bundle

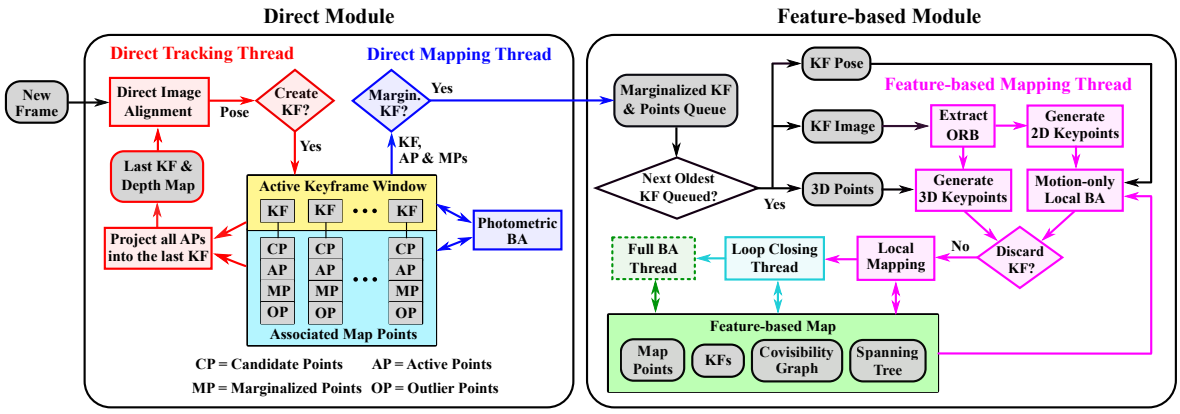


Figure 2.2: Our pipeline consists of two modules operating in parallel: One is a direct module that tracks every new frame with respect to the last keyframe and performs windowed photometric BA. The other is a feature-based module that reconstructs a globally consistent map and keyframe trajectory using the marginalized information from the direct module.

adjusted to obtain an accurate representation of the surrounding environment. New frames are tracked using direct image alignment (Baker and Matthews, 2004) with respect to the last keyframe and its depth map created by projecting active points in the window (see Fig. 2.1).

When a keyframe is marginalized from the direct module, its image and pose information is sent to the feature-based module, along with the map points within its FOV. The feature-based module extracts ORB descriptors from these keyframes and refines their poses with respect to the local feature map using motion-only BA. Some of these keyframes and map points are added to the local map and geometrically bundle adjusted for optimal accuracy. This process of feature-based mapping corresponds to the mid-level optimization.

Finally, at the most global level, a pose graph optimization (Strasdat et al., 2010*b*) is performed over Sim(3) constraints after each loop detection. Afterwards, a full BA (Mur-Artal and Tardós, 2017) optimizes all keyframes and feature points in the map for global consistency.

The key difference between our approach and SVO (Forster et al., 2014) (or the semi-direct methods in (Li et al., 2019, Bu et al., 2017)) is that we maintain in parallel two separate maps, each in the direct and the feature-based module. This allows us to utilize a locally accurate semi-dense map for fast and robust tracking, as well as a globally consistent sparse feature map for long-term reuse (*e.g.*, loop detection and closure).

2.4 Notation

Throughout the chapter, we use bold lower- and upper-case letters for vectors ($\mathbf{v}$) and matrices ($\mathbf{M}$), and light lower- and upper-case letters for scalars (s) and scalar functions (F), respectively. The intensity image is denoted by $I : \Omega \mapsto \mathbb{R}$ where $\Omega \subset \mathbb{R}^2$ is the image domain. We denote the camera intrinsic parameters with $\mathbf{c}$, and corresponding camera projection and back-projection functions with $\mathbf{\Pi_c} : \mathbb{R}^3 \mapsto \Omega$ and $\mathbf{\Pi_c^{-1}} : \Omega \times \mathbb{R} \mapsto \mathbb{R}^3$, respectively. Camera poses are represented as either rigid body or similarity transformation

matrices $\mathbf{T}_{iw} \in \text{SE(3)}$ or $\mathbf{S}_{iw} \in \text{Sim(3)}$ that transform a point from the world frame to frame i. We use $\mathcal{P}_i$ to denote the set of map points belonging to keyframe i and $\text{obs}(\mathbf{p})$ to denote the set of keyframes in which the point $\mathbf{p}$ is visible. The Euclidean and Huber norms (Huber, 1964) are denoted by $\|\cdot\|_2$ and $\|\cdot\|_\gamma$ respectively. The operator $\boxplus$ is defined as a simple addition for Euclidean parameters and a left multiplication for the pose, *i.e.*, for Lie-algebra $\mathfrak{se}(3)$ elements in twist coordinates $\mathbf{x} \in \mathbb{R}^6$, $\mathbf{x} \boxplus \mathbf{T} := \exp_{\text{SE(3)}}(\mathbf{x})\,\mathbf{T}$. We use the same notation as in (Strasdat et al., 2010*b*) for the exponential and logarithmic mapping for SE(3) and Sim(3).

2.5 Direct Module

We use the original implementation of DSO (Engel et al., 2018) as our direct module, which is responsible for initial map bootstrapping, real-time camera tracking and local mapping. In this section, we describe the windowed optimization and marginalization scheme of DSO. The reader is referred to the original work (Engel et al., 2018) for details on direct tracking and other front-end operations.

2.5.1 Windowed Photometric Bundle Adjustment

When a point $\mathbf{p}$ in a reference frame I_i is observed in another frame I_j, the photometric error is defined as the weighted SSD over the 8-point neighborhood pixels $\mathcal{N}_\mathbf{p}$ as proposed in (Engel et al., 2018):

$$E_{ij}^{\mathbf{p}} := \sum_{\tilde{\mathbf{p}} \in \mathcal{N}_\mathbf{p}} \omega_{\tilde{\mathbf{p}}} \left\| I_j\left[\tilde{\mathbf{p}}'\right] - b_j - \frac{t_j e^{a_j}}{t_i e^{a_i}} \left(I_i\left[\tilde{\mathbf{p}}\right] - b_i\right) \right\|_\gamma \tag{2.1}$$

$$\text{with} \quad \omega_{\tilde{\mathbf{p}}} := \frac{c^2}{c^2 + \|\nabla I_i(\tilde{\mathbf{p}})\|_2^2}, \tag{2.2}$$

$$\tilde{\mathbf{p}}' = \mathbf{\Pi}_\mathbf{c}\left(\mathbf{T}_{wj}^{-1}\mathbf{T}_{wi}\mathbf{\Pi}_\mathbf{c}^{-1}\left(\tilde{\mathbf{p}}, d_\mathbf{p}\right)\right) \tag{2.3}$$

where t, a and b are exposure time and affine brightness function parameters, and $d_\mathbf{p}$ is the inverse depth (Civera et al., 2008*b*) of $\mathbf{p}$ in the reference frame I_i. The weight $w_{\tilde{\mathbf{p}}}$ down-weights high-gradient pixels with some constant c. The total energy function to be minimized is given by the full photometric error plus a prior pulling the affine brightness parameters to zero:

$$E_{\text{photo}} := \sum_{i \in \mathcal{F}} \sum_{\mathbf{p} \in \mathcal{P}_i} \sum_{j \in \text{obs}(\mathbf{p})} E_{ij}^{\mathbf{p}} + \sum_{i \in \mathcal{F}} \left(\lambda_a a_i^2 + \lambda_b b_i^2\right), \tag{2.4}$$

where $\mathcal{F}$ denotes the set of all frames in the window. When exposure times are known, we set λ_a and λ_b to some constant values. Otherwise, we set $\lambda_a = \lambda_b = 0$ and $t_i = t_j = 1$ in (2.1). The optimization is performed using an iteratively reweighted Gauss-Newton or

Levenberg-Marquardt algorithm in a coarse-to-fine scheme. The update equation is given by

$$\delta\boldsymbol{\xi} = -\left(\mathbf{J}^T\mathbf{W}\mathbf{J}\right)^{-1}\mathbf{J}^T\mathbf{W}\mathbf{r} \quad \text{and} \quad \boldsymbol{\xi}^{\text{new}} \leftarrow \delta\boldsymbol{\xi} \boxplus \boldsymbol{\xi}, \tag{2.5}$$

where $\mathbf{r}$ is the stacked vector of residuals, $\mathbf{J}$ is its Jacobian and $\mathbf{W}$ is the diagonal weight matrix. The state variable $\boldsymbol{\xi}$ contains all the active variables in the window, *i.e.*, camera poses, affine brightness parameters, inverse depths and camera intrinsics.

2.5.2 Marginalization

The size of the optimization window is kept bounded by marginalizing the least useful keyframes and points using the Schur complement (Engel et al., 2018, Leutenegger et al., 2015). Points are marginalized if they are not observed in the latest two keyframes or their host keyframe is marginalized. Keyframes (other than the latest two) are marginalized if either less than 5% of its points are visible in the latest keyframe, or if it has the highest "distance score" when the window contains more than a certain number of keyframes. We refer to the original work (Engel et al., 2018) for the computation of this heuristic score.

2.6 Feature-based Module

When a keyframe is marginalized from the direct module, the feature-based module receives its image and pose information, as well as the 3D locations of both active and marginalized map points within its FOV. This information is then used for feature-based pose refinement, mapping and loop closing. Note that the marginalization strategy in Section 2.5.2 does not necessarily marginalize the oldest keyframe in the window. To avoid temporal inconsistency in such cases, we store the marginalized keyframes and points in a queue and wait until the next oldest keyframe is marginalized. If more than five keyframes are queued and the next oldest keyframe is still active, we take its latest pose and points to proceed further instead of waiting. All optimizations are performed using the original implementation in ORB-SLAM (Mur-Artal et al., 2015), which is based on the Levenberg-Marquardt algorithm in g2o (Kümmerle et al., 2011).

2.6.1 Relative Scale and Initial Pose Estimation

In our loosely-coupled approach, the direct and the feature-based modules maintain two separate maps. Due to the scale ambiguity of a monocular system, the scales of these two maps drift over time and do not converge to the same value. Therefore, we continuously compute the relative scale using Sim(3) alignment (Arun et al., 1987, Horn, 1987) between the 30 most recent keyframes in the feature-based module and their counterparts in the direct module.

Once the relative scale s is known, the incremental transformation in the direct module can be scaled appropriately and used as an initial pose guess in the feature-based module: Let i and j denote the previous and current keyframe. Then,

$$\mathbf{T}_{jw|\mathrm{F}} = \begin{bmatrix} \mathbf{R}_{ji|\mathrm{F}} & \mathbf{t}_{ji|\mathrm{F}} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \mathbf{T}_{iw|\mathrm{F}} \tag{2.6}$$

$$\text{with } \mathbf{R}_{ji|\mathrm{F}} = \mathbf{R}_{ji|\mathrm{D}} = \mathbf{R}_{jw|\mathrm{D}}(\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}, \tag{2.7}$$

$$\mathbf{t}_{ji|\mathrm{F}} = s\,\mathbf{t}_{ji|\mathrm{D}} = s\left(-\mathbf{R}_{ji|\mathrm{D}}\mathbf{t}_{iw|\mathrm{D}} + \mathbf{t}_{jw|\mathrm{D}}\right), \tag{2.8}$$

where the subscripts D and F indicate the direct and the feature-based module, respectively. We provide the derivation of (2.7) and (2.8) in Appendix A.

2.6.2 3D Keypoints Generation

The map points from the direct module are used in two ways: (1) for creating an initial set of 3D keypoints to bootstrap the feature-based module, or (2) for adding more local map points to improve the tracking robustness. Given the 2D position $\mathbf{p}$ of an ORB feature in frame i, we generate a 3D keypoint $\mathbf{x}_w$ in the world frame as

$$\mathbf{x}_w = \mathbf{T}_{iw}^{-1}\mathbf{\Pi}_{\mathbf{c}}^{-1}\left(\mathbf{p}, \frac{d_{\mathbf{p}}}{s}\right) \quad \text{with} \quad d_{\mathbf{p}} = \frac{\sum_{k\in\mathcal{P}_{\mathbf{p}}} d_k/\sigma_k^2}{\sum_k 1/\sigma_k^2}, \tag{2.9}$$

where s is the relative scale between the direct and the feature-based module (Section 2.6.1), and $\mathcal{P}_{\mathbf{p}}$ and $d_{k\in\mathcal{P}_{\mathbf{p}}}$ respectively denote the set of all map points whose projection in frame i is equal to $\mathbf{p}$ and their inverse depths in frame i. We dilate the projection by two pixels to ensure a sufficient number of valid depths for the keypoints. Note that the inverse depth $d_{\mathbf{p}}$ is computed as the inverse-variance weighted average, which is equivalent to the Kalman filter update with multiple measurements (Engel et al., 2013).

We found that extracting more features during slower camera motions often increases the number of covisibility links (Mur-Artal et al., 2015) between the keyframes and improves mapping accuracy. Therefore, we vary the number of features to extract per image as follows: Let f_{kf} be the keyframe addition frequency in the direct module. Using this as the indicator of the relative camera speed, we set $N_{\text{features}} = 2500$ if $f_{\text{kf}} < 4$Hz and $N_{\text{features}} = 1500$ if $f_{\text{kf}} > 7$Hz. Otherwise, we interpolate between the two bounds based on f_{kf}.

2.6.3 Keyframe Pose Refinement and Failure Recovery

Once the direct module provides an initial pose estimate of a new keyframe (Section 2.6.1), we refine it using motion-only geometric BA with respect to the local feature map. The total energy function is composed of the variance-normalized reprojection errors of the local map

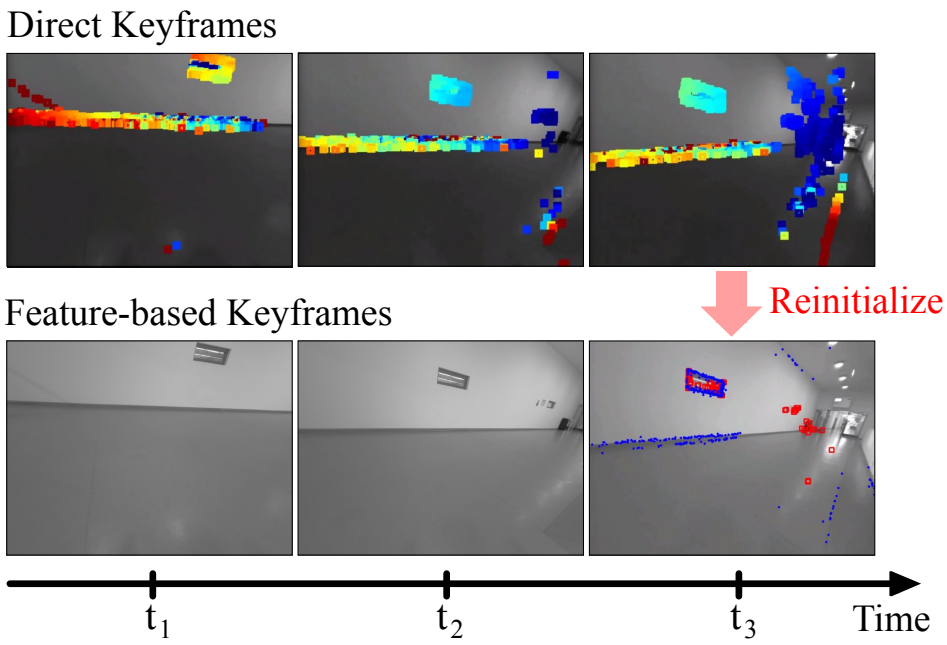


Figure 2.3: **[TUM monoVO]** Failure recovery in *sequence 40*: At t_1 and t_2, the feature-based module fails due to the lack of features in the scene, whereas the direct module is able to track the high-gradient pixels. At t_3, the scene now contains a sufficient number of features, and their depths can be initialized with the help of the direct module.

points:

$$E_{\text{reproj}} = \sum_{i \in \mathcal{F}_{\text{local}}} \sum_{\mathbf{x} \in \mathcal{P}_i} \sum_{j \in \text{obs}(\mathbf{x})} \left\| \frac{\mathbf{p}_{j,\mathbf{x}} - \boldsymbol{\Pi}_{\mathbf{c}} \left(\mathbf{T}_{iw} \mathbf{x}_w \right)}{\sigma_{\mathbf{x}}^2} \right\|_{\gamma} \tag{2.10}$$

$$\text{with} \quad \sigma_{\mathbf{x}}^2 := (\lambda_{\text{pyr}})^{2L_{\text{pyr},\mathbf{x}}}, \tag{2.11}$$

where $\mathcal{F}_{\text{local}}$ denotes the set of all local keyframes, *i.e.*, all keyframes sharing map points with the new keyframe and their neighbors in the covisibility graph (Mur-Artal et al., 2015), $\mathbf{p}_{j,\mathbf{x}} \in \mathbb{R}^2$ the match to the keypoint $\mathbf{x}$ in frame j, and $\sigma_{\mathbf{x}}^2$ the variance of the feature location in frame i. This variance depends on the constant scale factor of the image pyramid λ_{pyr} (> 1) and the pyramid level $L_{\text{pyr},\mathbf{x}}$ at which the keypoint was detected.

When the feature-based module fails due to insufficient matches, we reinitialize the module, following the procedure explained in Section 2.6.2. This is illustrated in Fig. 2.3. Since our direct module is robust to low-texture scenes, we can rely on its tracking while the feature-based module is lost. Then, as soon as we detect more features, we reinitialize the local map points using the depths from the direct module.

2.6.4 Feature-based Local Mapping and Loop Closing

After generating 3D keypoints and refining the keyframe pose, we insert them in the feature-based map if the number of matches falls below 150 or more than three keyframes passed from the last insertion. Once the keyframe and the points are added to the map, they are processed by the local mapping, as outlined in (Mur-Artal et al., 2015). This includes the local

geometric BA that minimizes (2.10) to jointly optimize the current keyframe, its neighbors in the covisibility graph, and all the map points belonging to those keyframes.

In the loop closing thread, the place recognition module (Mur-Artal and Tardós, 2014) based on DBoW2 (Galvez-López and Tardos, 2012) detects large loops by querying the keyframe database. Once a loop is detected, the keyframes and map points on each side of the loop are aligned and fused. To correct the scale drift, pose graph optimization (Strasdat et al., 2010*b*) is performed over the essential graph (Mur-Artal et al., 2015), minimizing

$$E_{\text{graph}} = \sum_{(i,j)\in\mathcal{E}_{\text{edge}}} \left\| \log_{\text{Sim(3)}} \left(\mathbf{S}_{ij,0}\, \mathbf{S}_{jw}\, \mathbf{S}_{iw}^{-1} \right) \right\|_2^2, \tag{2.12}$$

where $\mathcal{E}_{\text{edge}}$ denotes the set of edges in the essential graph, and $\mathbf{S}_{ij,0} = \mathbf{S}_{iw,0}\mathbf{S}_{jw,0}^{-1}$ is the fixed similarity transformation (with the scale 1) between the frame i and j just prior to the pose graph optimization. If the edge is created from a loop closure, this transformation is instead computed using the method of Horn (Horn, 1987). Finally, a full BA (Mur-Artal and Tardós, 2017) is performed afterwards.

2.6.5 Does Feature-based Mapping Always Improve Accuracy?

The answer is no. In general, the feature-based mapping described in the previous sections improves the accuracy if there is a loop closure or when the camera motion is mostly loopy, which enhances the covisibility of features in multiple keyframes and the reuse of map points. However, we found that it actually causes more drift when the camera motion is mostly exploratory without loop closures (a similar finding was also reported in (Engel et al., 2018)).

We solve this by keeping two versions of the keyframe trajectory: one that is initially given by the direct module and the other modified by the feature-based module. We assume that the latter is more accurate if there is a loop closure or less than a quarter of all past keyframes have *collinear covisibility links* at a given point in time. The covisibility links (*i.e.*, 3D lines connecting the keyframe to its neighbors in the covisibility graph) are considered collinear if none of them form an angle between 30° and 150°. This is illustrated in Fig. 2.4. We found that this method is especially effective for detecting exploratory translational motions.

While this strategy allows us to mitigate the odometry drift in exploratory situations, a more elegant solution would be to deal with the source of inaccuracy in the feature-based mapping that causes drift. (Yang et al., 2018) suggests a few hints on how this could be done (*e.g.*, careful point management and sub-pixel matching refinement), but it is still an open problem and remains for future work.

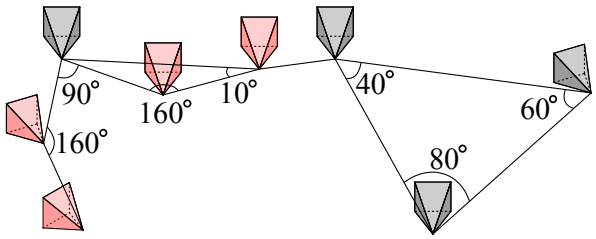


Figure 2.4: Illustration of the keyframes with collinear covisibility links (red). None of their covisibility links form an angle between 30° and 150°.

2.7 Evaluation

2.7.1 Evaluated Settings, Datasets and Methodology

We implement our method using ROS[1] and compare it against ORB-SLAM (Mur-Artal et al., 2015) and DSO (Engel et al., 2018). We evaluate each algorithm in two different settings:

- **ORB-VO** and **ORB-SLAM**: For the VO setting, we disable the loop closing thread. Relocalization is disabled for both settings to evaluate the tracking robustness. We do not apply photometric calibration (Engel et al., 2016), as we found that it worsens feature extraction and matching (similar findings were also reported in (Engel et al., 2018, Yang et al., 2018)).
- **DSO-default** and **DSO-reduced**: We use the default and reduced settings provided in the original DSO implementation. The only difference is that, for the reduced setting, we always resize the input images by half. Photometric calibration (Engel et al., 2016) is applied when available.
- **Ours-VO** and **Ours-SLAM**: The VO setting uses DSO-reduced and ORB-VO settings, whereas the SLAM setting uses DSO-reduced and ORB-SLAM settings. This means that the direct module processes photometrically calibrated images (if available) at half resolution, while the feature-based module processes photometrically non-calibrated images at full resolution. We found that using half resolution in the feature-based module significantly worsens the performance, which is in line with the observation in (Engel et al., 2016). For efficiency, we reduce the number of iterations in the local geometric BA by half.

We use two public benchmark datasets for evaluation:

1. **EuRoC MAV dataset** (Burri et al., 2016), which contains 11 indoor stereo sequences with 752×480 pixel resolution at 20 fps. As in (Engel et al., 2018), we crop the beginning and end of each sequence to disregard large occlusions due to the ground and aggressive motions meant for IMU initialization. We evaluate the tracking accuracy using the absolute trajectory RMSE (e_{ate}) of keyframe poses after Sim(3) alignment with

[1]Robot Operating System, `http://www.ros.org/`.

the ground truth. Photometric calibration and exposure times are not available for this dataset.

2. **TUM monoVO dataset** (Engel et al., 2016), which contains 50 in- and outdoor monocular sequences recorded at 20–50 fps. We use 640×480 pixel resolution for undistorted images. Since the full ground-truth data is not available for this dataset, the tracking accuracy is evaluated in terms of the alignment error (e_{align}) proposed in (Engel et al., 2016). The dataset also provides photometric calibration and exposure times.

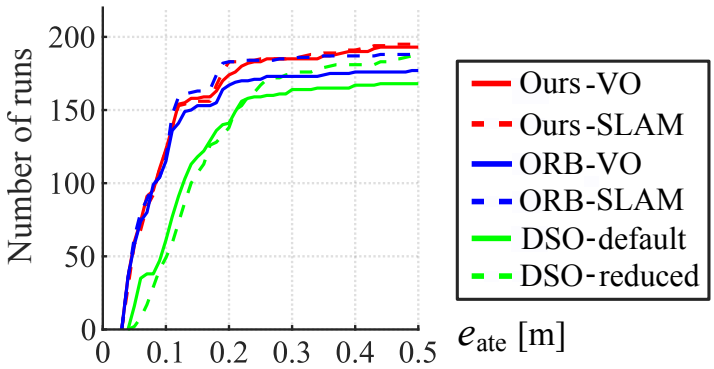
To account for non-deterministic behaviour, we run each method 10 times. This amounts to 220 runs for the EuRoC MAV dataset (*i.e.*, 110 runs for each left and right camera) and 500 runs for the TUM monoVO dataset. On the EuRoC MAV dataset, we consider that runs were unsuccessful if more than 20% of the total frames could not be tracked either due to the delayed map initialization or complete tracking failures. On the TUM monoVO dataset, we disable the keyframe culling of ORB-VO/SLAM and our systems within the start- and end-segment of each trajectory where the ground-truth data is available. This prevents the lack of keyframes when computing the alignment error on these segments. All images were preloaded into memory, but not rectified or photometrically calibrated beforehand. All results were obtained in real-time on a laptop CPU[2].

2.7.2 Results

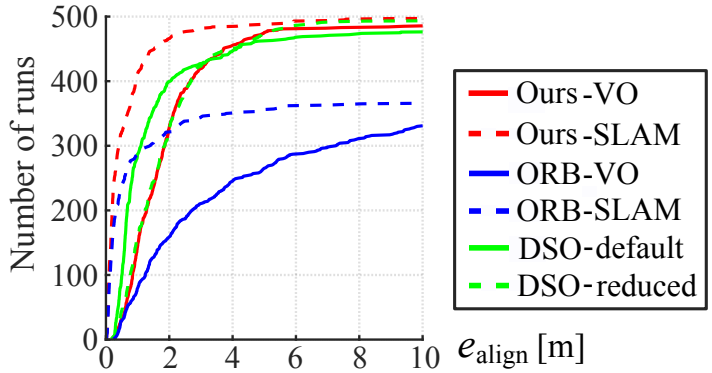
Fig. 2.9 shows the error values for each sequence of both datasets, and their median values are reported in Appendix B. The aggregated results are given in Fig. 2.5 in the form of cumulative error plots indicating how many runs have yielded error values below a certain level. On both datasets, we make three common observations: First, DSO-reduced is more robust than DSO-default, which is most likely due to the faster tracking speed, as shown in Tab. 2.1. We observe similar accuracy between the two settings on the EuRoC MAV dataset, but higher accuracy with DSO-default on the TUM monoVO dataset. Second, loop closing in ORB-SLAM improves the performance. It increases accuracy on the TUM monoVO dataset and robustness on both datasets. Third, due to the non-deterministic nature of real-time multi-threading, the occurrence of loop closure is not necessarily consistent on each sequence (see Fig. 2.9).

On the EuRoC MAV dataset, DSO (both default and reduced) yields the lowest accuracy. It was also reported in (Engel et al., 2018) that DSO was less accurate than ORB-VO on the same dataset. However, we were unable to reproduce their exact results, in particular those showing that DSO was more robust in real-time. Our systems (both VO and SLAM) and ORB-SLAM show very similar performance on this dataset, except for *V1_03_difficult* where ORB-SLAM outperforms our SLAM system. This is because in ORB-SLAM, a loop closure reintroduces the detected map points into the local map and robustifies the tracking.

[2]Intel Core i7-4810MQ, quad-core at 2.8 GHz with 15GB RAM



(a) EuRoC MAV



(b) TUM monoVO

Figure 2.5: Cumulative error plot aggregating **(a)** the absolute trajectory errors e_{ate} [m] over all runs of all EuRoC MAV sequences and **(b)** alignment errors e_{align} [m] over all runs of all TUM monoVO sequences. The closer the curve is to the y-axis, the higher the accuracy, as it means more runs with low errors. The farther the end of the curve is from the x-axis, the higher the robustness, as it means more runs without tracking failures.

On the other hand, our direct tracking does not reuse the detected map points from the loop closures, so its robustness is unaffected.

The comparison between DSO-reduced and our VO system indicates that the feature-based pose refinement (Section 2.6.3) and local mapping (Section 2.6.4) can improve the accuracy, even without loop closure. Fig. 2.6 shows this effect over time on two of the sequences of the EuRoC MAV dataset. Notice the increased amount of drift when we do not reuse the map points that leave the FOV. We also note that even with loop closure, the local geometric BA is still important because the pose graph optimization alone does not guarantee the optimal reconstruction of the local environment.

On the TUM monoVO dataset, DSO (both default and reduced) is significantly more robust than ORB-SLAM/VO, which is similar to the results reported in (Engel et al., 2018, Yang et al., 2018). Our VO system achieves very similar performance to DSO-reduced, as none of the final trajectories are affected by the feature-based module. This is because more than 75% of the total keyframes have collinear covisibility links in all sequences (see Section. 2.6.5), which is in contrast to the EuRoC MAV dataset where the opposite is true. Fig. 2.5 and 2.9 show that the loop closure significantly reduces the alignment errors for our

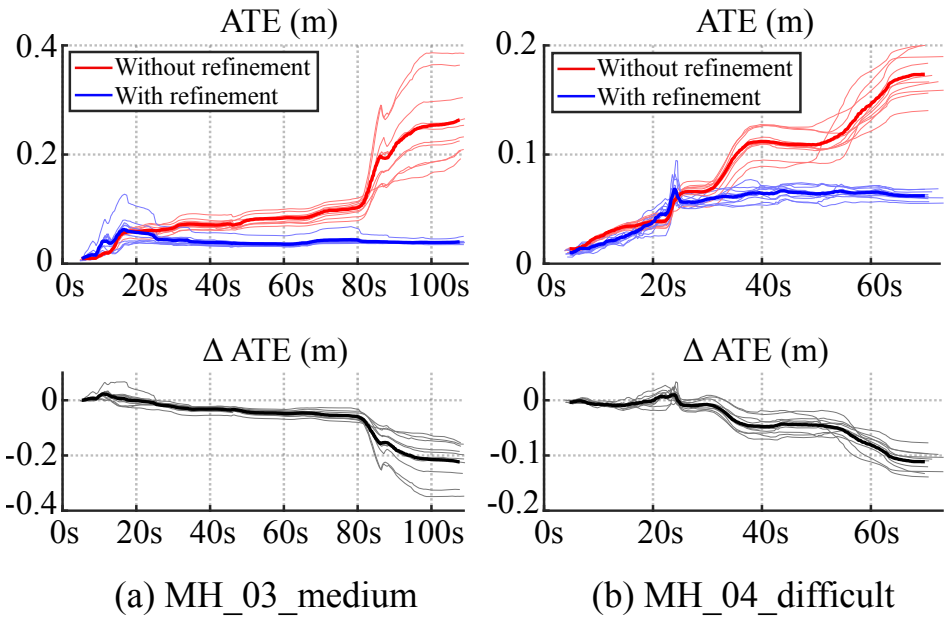


Figure 2.6: **[EuRoC MAV] Top:** Time evolution of the ATEs with and without the feature-based refinement (Section 2.6.3 and 2.6.4). The loop-closure detection is disabled. The average of 10 independent runs is shown in bold. **Bottom:** The ATE difference over time, *i.e.*, the gap between the red and the blue curves.

SLAM system in the majority of the sequences. As a result, our SLAM system achieves the best overall performance across both datasets. Fig. 2.7 compares the estimated trajectories on some of the TUM monoVO sequences. Note how ORB-SLAM fails in the middle of some sequences and DSO accumulates drift, while our SLAM system completes the whole sequences and closes loops.

Tab. 2.1 summarizes the statistics of dropped frames and tracking times[3] on the EuRoC MAV dataset. It can be seen that DSO-reduced and both our systems have lower frame drops and faster tracking speed than the rest. This shows the advantage of direct tracking, which eliminates the need to perform feature extraction and matching for every frame. In Fig. 2.8, we further show how much percentage of keyframes and map points are reduced in our systems compared to ORB-VO/SLAM. We observe average 27% (up to 46%) keyframes reduction and average 6% (up to 27%) map points reduction for the SLAM system. This suggests that our system is relatively more scalable than ORB-SLAM. Note that the sparsity of the feature-based keyframes and map points is what enabled the efficient map reuse on both local and global scale. Without relying on the sparse keypoints, it would require prohibitively more computation to maintain a global covisibility graph with the large number of map points in the direct module.

[3]This is defined as $t_i - t_{i-1}$ where t_i and t_{i-1} are the two timestamps at which the tracking thread first processes frame i and $i-1$, respectively.

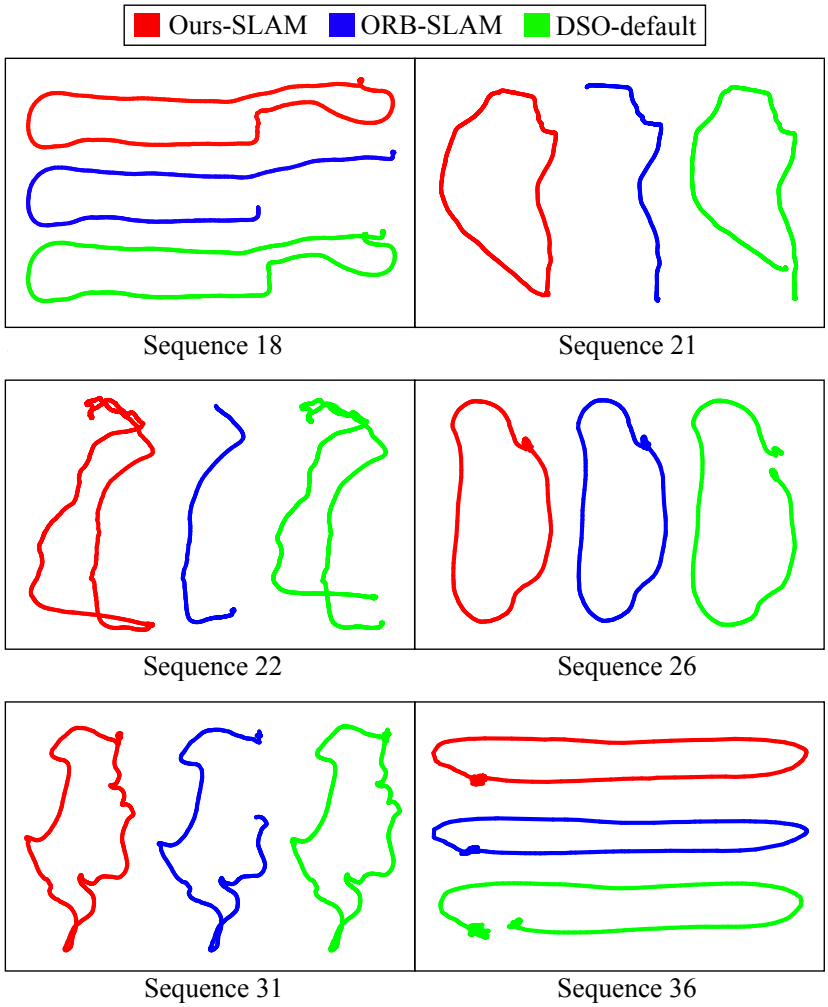


Figure 2.7: **[TUM monoVO]** Sample trajectories estimated with the median accuracy. All sequences have the ground-truth trajectories that start and end at the same positions. For several sequences, ORB-SLAM repeatedly loses tracking and DSO suffers from consistent drifts. On the other hand, our system tracks the entire trajectories and close the loops most of the time.

2.8 Conclusions

In this chapter, we proposed a loosely-coupled semi-direct method for real-time monocular SLAM. Our system consists of two modules running in parallel. One module uses a direct method to track new frames fast and robustly with respect to a local semi-dense map. The other module uses the resulting points and pose estimates as prior to build a globally consistent sparse feature-based map. We have shown on two public datasets that our method outperforms the state-of-the-art in terms of tracking accuracy and robustness.

	Dropped frames (%)			Tracking Times (ms)		
	Med*	Mean	Std	Med*	Mean	Std
ORB-VO	0.95	1.56	1.61	22.09	25.46	8.62
ORB-SLAM	1.54	2.14	1.57	22.74	26.47	9.48
DSO-default	0.81	1.16	**1.07**	7.13	9.66	17.33
DSO-reduced	**0.28**	**0.74**	**1.00**	**2.60**	**4.07**	**7.40**
Ours-VO	**0.31**	1.02	1.48	**4.62**	**6.19**	**8.62**
Ours-SLAM	0.32	**0.97**	1.41	4.69	6.23	9.48

Table 2.1: **[EuRoC MAV]** Dropped frames percentage and tracking times. The two smallest values are highlighted in bold. *This is the median of the median results in each sequence.

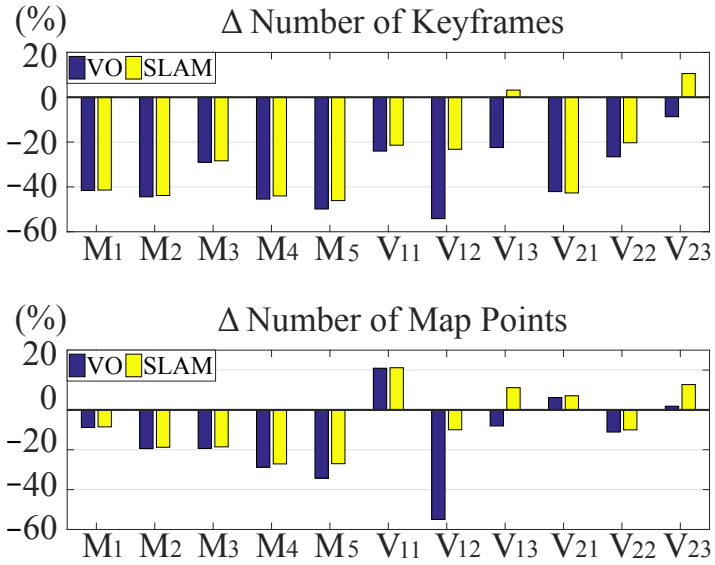


Figure 2.8: **[EuRoC MAV]** Percentage difference in the number of total KFs and map points of our VO/SLAM compared to ORB-VO/SLAM, respectively. M_{1-5}: Machine Hall sequences. $V_{11-13,\ 21-23}$: Vicon Room 1 and 2 sequences.

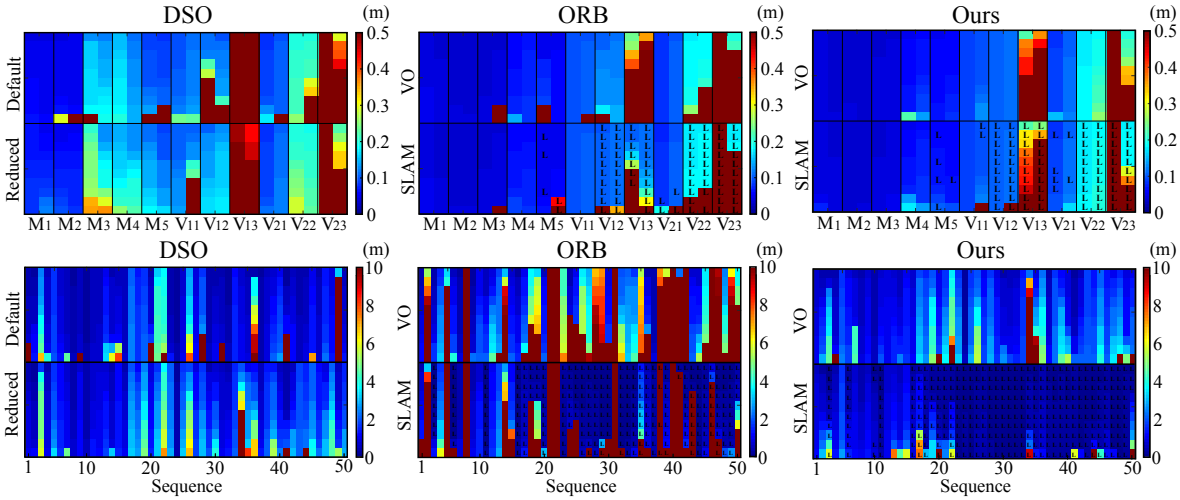


Figure 2.9: Each of the colored blocks represent the final error value from each trial on each sequence. The letter 'L' indicates the presence of one or more loop closure links. **Top row:** Absolute trajectory errors e_{ate} [m] on the EuRoC MAV dataset. M_{1-5}: Machine Hall sequences. $V_{11-13,\ 21-23}$: Vicon Room 1 and 2 sequences. **Bottom row:** Alignment errors e_{align} [m] on the TUM monoVO dataset.

Chapter 3

Closed-Form Optimal Two-View Triangulation Based on Angular Errors

In this chapter, we study closed-form optimal solutions to two-view triangulation with known internal calibration and pose. By formulating the triangulation problem as L_1 and L_∞ minimization of angular reprojection errors, we derive the exact closed-form solutions that guarantee global optimality under respective cost functions. To the best of our knowledge, we are the first to present such solutions. Since the angular error is rotationally invariant, our solutions can be applied for any type of central cameras, be it perspective, fisheye or omnidirectional. Our methods also require significantly less computation than the existing optimal methods. Experimental results on synthetic and real datasets validate our theoretical derivations.

3.1 Introduction

Recovering the position of a 3D point given its projections in two or more cameras is called triangulation. It constitutes a fundamental building block in stereo vision (Tippetts et al., 2016), simultaneous localization and mapping (SLAM) (Mur-Artal and Tardós, 2017) and structure-from-motion (SfM) pipelines (Schönberger and Frahm, 2016). For large problems, reconstructing thousands (or millions) of points is not uncommon, so achieving fast and accurate triangulation is important for the performance of such systems.

If one assumes the exact knowledge of camera matrices and noiseless feature measurements, triangulation amounts to intersecting two backprojected rays that correspond to the same 3D point. In practice, however, this assumption is unrealistic, and the rays do not necessarily intersect. Therefore, a nontrivial method is required even for just two views.

The standard approach is to find the 3D point that minimizes a chosen cost function given the feature measurements. The most common are the L_1 norm (sum of magnitude), L_2 norm (sum of squares) and L_∞ norm (maximum) of image reprojection errors. While reasonable for perspective cameras, the image reprojection error does not generalize well to different

camera types (*e.g.*, omnidirectional or fully spherical panoramic cameras). This motivates the use of *angular* reprojection error, a rotationally invariant alternative to the *image* reprojection error that is generic and independent of the projection geometry (Mouragnon et al., 2009, Oliensis, 2002, Pagani and Stricker, 2011).

In this chapter, we derive, for the first time to our knowledge, the exact closed-form solutions to the L_1 and L_∞ optimal triangulation from two views based on the angular reprojection error. Unlike iterative methods (*e.g.*, (Kanatani et al., 2008, Lindstrom, 2010)), the proposed methods guarantee global optimality without any iterations, and unlike polynomial methods (*e.g.*, (Hartley and Sturm, 1997, Níster, 2001, Stewénius et al., 2005)), they do not involve finding the roots of a higher-degree polynomial. Hence, our methods simultaneously provide the global optimality, speed and simplicity. We also present our own derivation of the L_2 optimal solution that is much more compact and geometrically intuitive than the existing one (Oliensis, 2002). Since all three methods are based on the angular error, they are not limited to standard perspective cameras and can also be used for fisheye, omnidirectional and fully spherical panoramic cameras.

The chapter is organized as follows. In the next three sections, we discuss the related work and preliminaries. Section 3.5, 3.6 and 3.7 respectively present the closed-form solutions to the L_1, L_2 and L_∞ optimal triangulation. Section 3.8 addresses the cheirality constraint. Finally, experimental results are provided in Section 3.9, followed by the conclusions in Section 3.10.

3.2 Related Work

The most widespread approach to triangulation is to find the 3D point that minimizes the L_2 norm of image reprojection errors (Hartley and Zisserman, 2003). Assuming that image points are perturbed by Gaussian noise, the L_2 optimal solution gives the maximum likelihood estimate (MLE). This can be obtained in closed form by solving a polynomial of degree 6 for two views (Hartley and Sturm, 1997) and degree 47 for three views (Stewénius et al., 2005). Such polynomial methods are, however, computationally expensive and susceptible to ill-conditioning (Lindstrom, 2010). Besides, an iterative search for the roots may converge to a local minimum (Hartley and Sturm, 1997).

Another two-view method by Kanatani et al. (Kanatani et al., 2008) iteratively corrects the 2D projections of the points. Although this method was shown to be faster than the one by Hartley and Sturm (Hartley and Sturm, 1997), it does not satisfy the epipolar constraint (Longuet-Higgins, 1981) in each iteration. Lindstrom (Lindstrom, 2010) solved this problem with an improved iterative algorithm that is even more stable and faster. However, neither his method nor Kanatani's guarantees global optimality. Oliensis (Oliensis, 2002) showed that by formulating the problem as L_2 minimization of the sine of angular reprojection errors, an exact closed-form solution can be derived for two-view triangulation.

Instead of minimizing the L_2 norm, one may choose to minimize the L_1 norm of repro-

jection errors. The advantage of L_1 norm is that it is more robust to outliers as it places less emphasis on large errors (Hartley and Sturm, 1997, Kahl et al., 2008). For two views, Hartley and Sturm (Hartley and Sturm, 1997) showed that the L_1 optimal solution can be obtained in closed form by solving a polynomial of degree 8. They also found that the L_1 optimization gives slightly more accurate 3D results than the L_2 optimization.

In geometric problems, another popular norm is the L_∞ norm. The L_∞ optimal solution corresponds to the MLE under the assumption of uniform noise in the image points (Hartley and Kahl, 2007). The advantage of the L_∞ cost function over the L_2 cost is that it is relatively simpler and has a single minimum (Hartley and Schaffalitzky, 2004). For the case of two views, Níster (Níster, 2001) showed that the optimal solution can be obtained in closed form by keeping the reprojection errors equal in the two views and solving the resulting quartic equation. A main drawback of the L_∞ cost is that it is relatively more sensitive to outliers (Hartley and Schaffalitzky, 2004). This being said, such sensitivity was shown to be useful for outlier removal (Sim and Hartley, 2006, Olsson et al., 2010, Li, 2007).

While most of the aforementioned works formulate their optimization problem in terms of the image reprojection error, the angular reprojection error is another popular choice. It embodies a better noise model for fisheye or omnidirectional cameras (Oliensis, 2002, Mičušík and Pajdla, 2006). Even for perspective cameras, the assumption of Gaussian noise is not justified (Hartley and Kahl, 2007), and the angular reprojection error is just as valid as the image reprojection error, if not more so. In the literature, it has been proposed to minimize the sine of angular reprojection errors in L_2 norm (Oliensis, 2002), the tangent in L_2 or L_∞ norm (Hartley, Kahl, Olsson and Seo, 2013, Hartley and Schaffalitzky, 2004, Kahl and Hartley, 2008), and the cosine in negative L_1 norm (Recker et al., 2013, Chesi, 2014). In contrast to these methods, our L_1 and L_∞ optimization do not involve trigonometric functions.

3.3 Preliminaries on 3D Geometry

Throughout the chapter, we adopt the following notation: We use bold letters for vectors and matrices, and light letters for scalars. The Euclidean norm of a vector $\mathbf{v}$ is denoted by $\|\mathbf{v}\|$, and the unit vector by $\widehat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$. The angle between two lines $\mathbf{L}_0$ and $\mathbf{L}_1$ is denoted by $\angle(\mathbf{L}_0,\ \mathbf{L}_1) \in [0, \pi/2]$.

The following vector identities will come in handy later:

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = \mathbf{b} \cdot (\mathbf{c} \times \mathbf{a}) = \mathbf{c} \cdot (\mathbf{a} \times \mathbf{b}) \tag{3.1}$$

$$\|\widehat{\mathbf{a}} \times \widehat{\mathbf{b}}\|^2 = 1 - (\widehat{\mathbf{a}} \cdot \widehat{\mathbf{b}})^2 \tag{3.2}$$

We also make frequent use of the following formulas:

1. The distance between a point $\mathbf{p}$ and a plane $\Pi_0(\mathbf{x}) = \mathbf{n}_0 \cdot (\mathbf{x} - \mathbf{c}_0) = 0$ is given by $\|\mathbf{p} - \mathbf{r}_0\|$

where $\mathbf{r}_0$ is the projection of $\mathbf{p}$ onto Π_0. This is computed as follows:

$$\|\mathbf{p} - \mathbf{r}_0\| = |\widehat{\mathbf{n}}_0 \cdot (\mathbf{p} - \mathbf{c}_0)|. \tag{3.3}$$

2. The distance between two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ and $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$ is given by $\|\mathbf{r}_0 - \mathbf{r}_1\|$ where $\mathbf{r}_0$ and $\mathbf{r}_1$ are the points on each line that form the closest pair. Letting $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$ and $\mathbf{q} = \mathbf{m}_0 \times \mathbf{m}_1$, this is computed as follows:

$$\|\mathbf{r}_0 - \mathbf{r}_1\| = |\mathbf{t} \cdot \widehat{\mathbf{q}}|\,. \tag{3.4}$$

The two points can also be obtained individually (Kanazawa and Kanatani, 1995):

$$\mathbf{r}_0 = \mathbf{c}_0 + \frac{\mathbf{q} \cdot (\mathbf{m}_1 \times \mathbf{t})}{\|\mathbf{q}\|^2}\mathbf{m}_0, \tag{3.5}$$

$$\mathbf{r}_1 = \mathbf{c}_1 + \frac{\mathbf{q} \cdot (\mathbf{m}_0 \times \mathbf{t})}{\|\mathbf{q}\|^2}\mathbf{m}_1. \tag{3.6}$$

Additionally, the following two lemmas will be useful later:

Lemma 1 (Cone-On-Plane Perpendicularity)
When a plane is tangent to a right circular cone, the line of intersection is the projection of cone's axis onto the plane.

Proof. Consider a cone with axis $\mathbf{L}_0$ and plane $\mathbf{\Pi}$ tangent to this cone. They are both symmetric with respect to the plane that contains $\mathbf{L}_0$ and the normal of $\mathbf{\Pi}$. Let this plane of symmetry be $\mathbf{\Pi}_{\text{sym}}$. For any circular cross-section of the cone, there is a single point touching $\mathbf{\Pi}$. Therefore, this point must lie on $\mathbf{\Pi}_{\text{sym}}$, and so must the line of intersection, $\mathbf{L}'_0$. It follows that $\mathbf{\Pi}_{\text{sym}}$ contains $\mathbf{L}_0$ and $\mathbf{L}'_0$. Since $\mathbf{\Pi}_{\text{sym}}$ is perpendicular to $\mathbf{\Pi}$, $\mathbf{L}'_0$ is a projection of $\mathbf{L}_0$ onto $\mathbf{\Pi}$. ■

Lemma 2 (Single vs Multi-Pivot for Intersection)
Given two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ and $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$, let $\mathbf{L}'_0$ be the line that forms the smallest angle $\theta_0 \in [0,\ \pi/2]$ to $\mathbf{L}_0$ among all possible lines that intersect both point $\mathbf{c}_0$ and line $\mathbf{L}_1$. For any positive integer N, consider the following arbitrary lines passing $\mathbf{c}_0$ such that

$$\mathbf{L}^*_i(s^*_i) = \begin{cases} \mathbf{L}_0 & \textit{for} \quad i = 0 \\ \mathbf{c}_0 + s^*_i\mathbf{m}^*_i & \textit{for} \quad i = 1, 2, \cdots, N \end{cases}$$

where only $\mathbf{L}^*_N$ intersects $\mathbf{L}_1$. Then,

$$\theta_0 \leq \sum_{i=1}^{N} \angle(\mathbf{L}^*_i,\ \mathbf{L}^*_{i-1}). \tag{3.7}$$

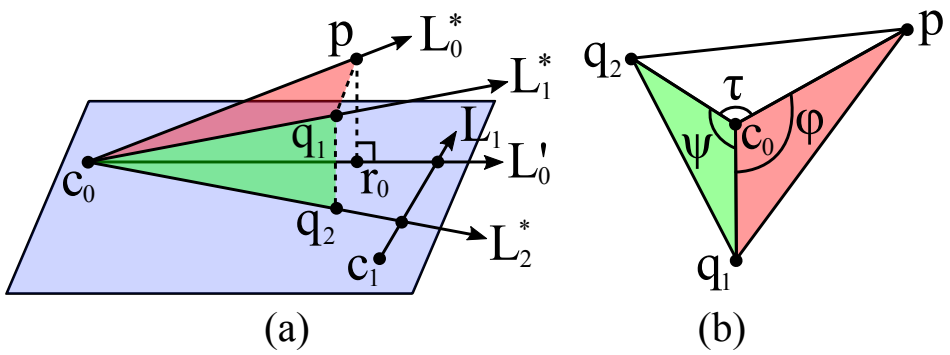


Figure 3.1: **(a)** Pivoting line $\mathbf{L}_0^*$ in two steps ($\mathbf{L}_0^* \rightarrow \mathbf{L}_1^* \rightarrow \mathbf{L}_2^*$) to make it intersect to another line $\mathbf{L}_1$. **(b)** A tetrahedron formed by point $\mathbf{c}_0$, $\mathbf{p}$, $\mathbf{q}_1$ and $\mathbf{q}_2$.

Proof. The right-hand side of (3.7) corresponds to the sum of N pivot angles that make line $\mathbf{L}_0$ to intersect $\mathbf{L}_1$. Fig. 3.1a depicts such operation for $N = 2$. Let $\phi = \angle(\mathbf{L}_0^*,\ \mathbf{L}_1^*)$, $\psi = \angle(\mathbf{L}_1^*,\ \mathbf{L}_2^*)$ and $\tau = \angle(\mathbf{L}_0^*,\ \mathbf{L}_2^*)$. Now, consider three arbitrary points $\mathbf{p}$, $\mathbf{q}_1$ and $\mathbf{q}_2$ on $\mathbf{L}_0^*$, $\mathbf{L}_1^*$ and $\mathbf{L}_2^*$, respectively. A tetrahedron formed by these three points and $\mathbf{c}_0$ are shown in Fig. 3.1b. At a vertex of a tetrahedron, the three edges form three angles such that the sum of any two angles is greater than the third one (Lines, 1935, Klamkin, 1970). Thus, $\tau \leq \phi + \psi$. Since θ_0 is the minimum pivot angle for intersection, we have $\theta_0 \leq \tau \leq \phi + \psi$, which proves (3.7) for $N = 2$. Now, for $N > 2$, we know that replacing the last two pivots by the corresponding single minimum pivot will produce $N - 1$ pivots that take equal or smaller angle. Repeating this process until $N = 1$ proves (3.7) for any $N > 2$. ■

Note that (3.4) can be interpreted as the minimum amount of translation required for the two lines to intersect. Later, it will be also important to know the minimum amount of rotation (or pivot) required for the two lines to intersect. We answer this question in the following lemma:

Lemma 3 (Minimum Pivot Angle for Intersection)
Given two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ *and* $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$, *let* $\mathbf{L}_0'$ *be the line that forms the smallest angle* $\theta_0 \in [0,\ \pi/2]$ *to* $\mathbf{L}_0$ *among all possible lines that intersect both point* $\mathbf{c}_0$ *and line* $\mathbf{L}_1$. *Then,* $\mathbf{L}_0'$ *is the projection of* $\mathbf{L}_0$ *onto the plane that contains* $\mathbf{c}_0$ *and* $\mathbf{L}_1$. *Furthermore, letting* $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$ *and* $\mathbf{n}_1 = \mathbf{m}_1 \times \mathbf{t}$,

$$\sin(\theta_0) = |\widehat{\mathbf{n}}_1 \cdot \widehat{\mathbf{m}}_0|. \tag{3.8}$$

We call θ_0 ***the minimum pivot angle for intersection****, as it represents the smallest angle required for pivoting line* $\mathbf{L}_0$ *at* $\mathbf{c}_0$ *to make it intersect* $\mathbf{L}_1$.

Proof. Consider a right circular cone with apex $\mathbf{c}_0$ and axis $\mathbf{L}_0$, lying sideways on a plane Π that contains $\mathbf{c}_0$ and line $\mathbf{L}_1$ (see Fig. 3.2). The equation of the plane is given by

$$\Pi(\mathbf{x}) = \mathbf{n}_1 \cdot (\mathbf{x} - \mathbf{c}_0) = 0 \quad \text{with} \quad \mathbf{n}_1 = \mathbf{m}_1 \times \mathbf{t}. \tag{3.9}$$

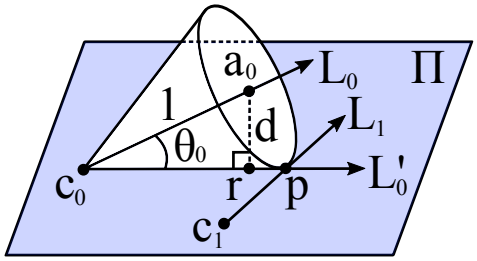


Figure 3.2: The angle θ_0 is the smallest angle required for pivoting line $\mathbf{L}_0$ at point $\mathbf{c}_0$ to make it intersect $\mathbf{L}_1$.

The line of intersection between the plane and the cone forms the smallest angle to $\mathbf{L}_0$ among all possible lines on the plane that pass $\mathbf{c}_0$. That is, it forms the smallest angle to $\mathbf{L}_0$ among all possible lines that pass both $\mathbf{c}_0$ and $\mathbf{L}_1$. Hence, this line of intersection must be $\mathbf{L}'_0$. Now, consider a point $\mathbf{a}_0 = \mathbf{c}_0 + \widehat{\mathbf{m}}_0$ located one unit away from $\mathbf{c}_0$ along $\mathbf{L}_0$. Let $\mathbf{r}$ be the projection of $\mathbf{a}_0$ onto plane Π. According to lemma 1, the point $\mathbf{r}$ must be located along $\mathbf{L}'_0$. Let $d = \|\mathbf{a}_0 - \mathbf{r}\|$, *i.e.*, the distance between $\mathbf{a}_0$ and plane Π. Then, we obtain $\sin(\theta_0)$ as follows:

$$\sin(\theta_0) = d \overset{(3.3)}{=} |\widehat{\mathbf{n}}_1 \cdot (\mathbf{a}_0 - \mathbf{c}_0)| = |\widehat{\mathbf{n}}_1 \cdot \widehat{\mathbf{m}}_0|. \qquad \blacksquare$$

3.4 Preliminaries on Two-View Triangulation

Consider two cameras C_0 and C_1 observing the same 3D world point $\mathbf{x}_w$. Let $\mathbf{c}_0$ and $\mathbf{c}_1$ be their positions in the world frame, and let $\mathbf{R}$ and $\mathbf{t}$ be the rotation matrix and translation vector that together transform a point from the camera frame C_0 to C_1, *i.e.*, $\mathbf{x}_1 = \mathbf{R}\mathbf{x}_0 + \mathbf{t}$, where $\mathbf{x}_0 = [x_0, y_0, z_0]^\intercal$ and $\mathbf{x}_1 = [x_1, y_1, z_1]^\intercal$ correspond to $\mathbf{x}_w$ in camera frame C_0 and C_1, respectively. Since triangulation is impossible for zero translation, we set $\|\mathbf{t}\| = \|\mathbf{c}_0 - \mathbf{c}_1\| = 1$ without loss of generality. Let $\mathbf{u}_0 = (u_0, v_0, 1)^\intercal$ and $\mathbf{u}_1 = (u_1, v_1, 1)^\intercal$ be the homogeneous pixel coordinates of the estimated correspondence to $\mathbf{x}_w$ in each frame. Given the camera calibration matrix $\mathbf{K}$, the normalized image coordinates $\mathbf{f}_0 = [x_0/z_0, y_0/z_0, 1]^\intercal$ and $\mathbf{f}_1 = [x_1/z_1, y_1/z_1, 1]^\intercal$ are related to $\mathbf{u}_0$ and $\mathbf{u}_1$ by $\mathbf{u}_0 = \mathbf{K}\mathbf{f}_0$ and $\mathbf{u}_1 = \mathbf{K}\mathbf{f}_1$.

The two backprojected rays in frame C_1, *i.e.*, $\mathbf{r}_1(s_1) = s_1\mathbf{f}_1$ and $\mathbf{r}_0(s_0) = s_0\mathbf{R}\mathbf{f}_0 + \mathbf{t}$, do not necessarily intersect due to inaccuracies in the image measurements and camera matrices. For the rays to intersect, $\mathbf{f}_0$ and $\mathbf{f}_1$ must be corrected to $\mathbf{f}'_0$ and $\mathbf{f}'_1$ such that the epipolar constraint (Longuet-Higgins, 1981) is satisfied. It enforces the coplanarity of $\mathbf{f}'_1$, $\mathbf{R}\mathbf{f}'_0$ and $\mathbf{t}$, and is given by

$$\mathbf{f}'_1 \cdot (\mathbf{t} \times \mathbf{R}\mathbf{f}'_0) = 0. \tag{3.10}$$

The goal of the optimal triangulation is to *minimally* correct the feature rays so that they satisfy (3.10) and intersect at some point $\mathbf{x}'_1$ in frame C_1. What is meant by "minimal" depends on the chosen cost function and error criterion. Fig. 3.3 illustrates two most popular error criteria, namely the image reprojection error and the angular reprojection error.

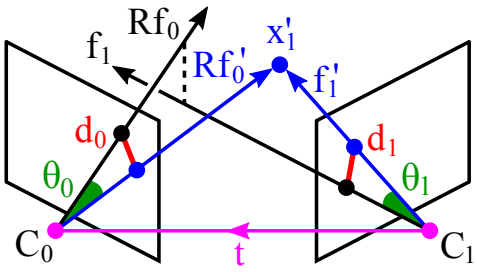


Figure 3.3: The difference between the observed features ($\mathbf{f}_0$, $\mathbf{f}_1$) and the triangulation result ($\mathbf{f}'_0$, $\mathbf{f}'_1$) can be quantified by either image reprojection errors (d_0, d_1) or angular reprojection errors (θ_0, θ_1).

Formally, they are defined as follows:

$$d_i := \|\mathbf{u}_i - \mathbf{u}'_i\| = \|\mathbf{K}\,(\mathbf{f}_i - \mathbf{f}'_i)\| \quad \text{for } i = 0, 1, \tag{3.11}$$

$$\theta_i := \angle\,(\mathbf{f}_i, \mathbf{f}'_i) = \angle\left(\mathbf{K}^{-1}\mathbf{u}_i, \mathbf{K}^{-1}\mathbf{u}'_i\right) \quad \text{for } i = 0, 1 \tag{3.12}$$

In this work, we minimize the latter in L_1, L_2 and L_∞ norms. Once we have the optimal $\mathbf{f}'_0$ and $\mathbf{f}'_1$, the point of intersection $\mathbf{x}'_1$ can be obtained using either (3.5) or (3.6):

$$\mathbf{x}'_1 = \mathbf{t} + \underbrace{\frac{\mathbf{z}\cdot(\mathbf{t}\times\mathbf{f}'_1)}{\|\mathbf{z}\|^2}}_{\lambda_0}\mathbf{R}\mathbf{f}'_0 = \underbrace{\frac{\mathbf{z}\cdot(\mathbf{t}\times\mathbf{R}\mathbf{f}'_0)}{\|\mathbf{z}\|^2}}_{\lambda_1}\mathbf{f}'_1 \tag{3.13}$$

$$\text{with} \quad \mathbf{z} = \mathbf{f}'_1 \times \mathbf{R}\mathbf{f}'_0,$$

where λ_i equals the depth multiplied by $\|\mathbf{f}'_i\|$ for $i = 0, 1$.

Note that the epipolar constraint (3.10) is a necessary condition for intersecting the two rays, but not a sufficient one. Fig. 3.4 illustrates scenarios where the two rays are coplanar, but do not intersect. This happens when the intersection requires negative depth(s), violating the *cheirality* constraint (Hartley and Zisserman, 2003). In the following analysis (until Section 3.8), we will temporarily assume that satisfying the epipolar constraint (3.10) is sufficient for intersecting the rays.

3.5 Closed-Form L_1 Triangulation

The L_1 triangulation based on the angular reprojection error (3.12) finds the feature rays $\mathbf{f}'_0$ and $\mathbf{f}'_1$ that minimize $\theta_0 + \theta_1$ subject to the epipolar constraint (3.10). The following lemma reveals a surprising fact that $(\theta_0 + \theta_1)_{\min}$ is achieved by correcting either one of $\mathbf{f}_0$ or $\mathbf{f}_1$, but not both:

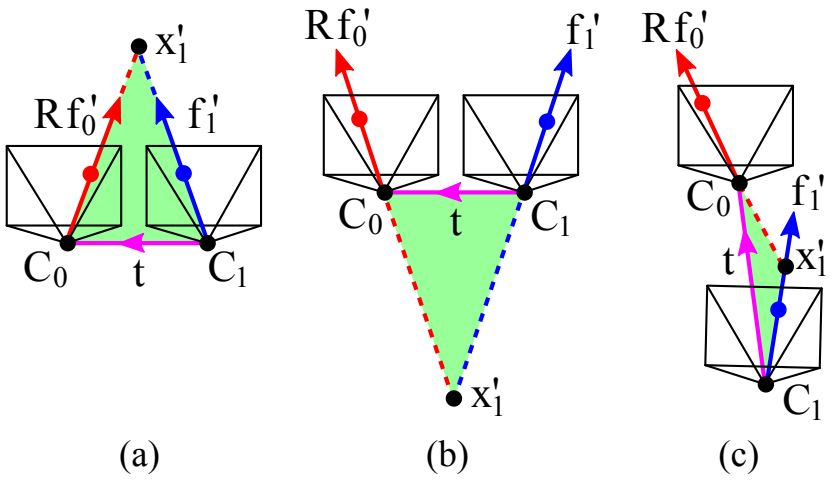


Figure 3.4: Example scenarios satisfying the epipolar constraint (3.10). The epipolar plane (shown in green) contains both the rays and the camera centers. Cheirality condition is violated in case (b) and (c).

Lemma 4 (L_1 Angle Minimization)
Given two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ *and* $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$, *consider any two intersecting lines that also pass* $\mathbf{c}_0$ *and* $\mathbf{c}_1$, *respectively, i.e.,* $\mathbf{L}'_0(s'_0) = \mathbf{c}_0 + s'_0\mathbf{m}'_0$ *and* $\mathbf{L}'_1(s'_1) = \mathbf{c}_1 + s'_1\mathbf{m}'_1$. *Let* $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$, $\mathbf{n}_0 = \mathbf{m}_0 \times \mathbf{t}$, $\mathbf{n}_1 = \mathbf{m}_1 \times \mathbf{t}$, $\theta_0 = \angle(\mathbf{L}_0, \mathbf{L}'_0)$ *and* $\theta_1 = \angle(\mathbf{L}_1, \mathbf{L}'_1)$. *Then,* $(\theta_0 + \theta_1)$ *is minimized for the following* $\mathbf{m}'_0$ *and* $\mathbf{m}'_1$*:*

If $\|\widehat{\mathbf{m}}_0 \times \mathbf{t}\| \leq \|\widehat{\mathbf{m}}_1 \times \mathbf{t}\|$,

$$\mathbf{m}'_0 = \mathbf{m}_0 - (\mathbf{m}_0 \cdot \widehat{\mathbf{n}}_1)\,\widehat{\mathbf{n}}_1 \quad \textit{and} \quad \mathbf{m}'_1 = \mathbf{m}_1. \tag{3.14}$$

Otherwise,

$$\mathbf{m}'_0 = \mathbf{m}_0 \quad \textit{and} \quad \mathbf{m}'_1 = \mathbf{m}_1 - (\mathbf{m}_1 \cdot \widehat{\mathbf{n}}_0)\,\widehat{\mathbf{n}}_0 \tag{3.15}$$

Proof. One of the following is true when $(\theta_0 + \theta_1)$ is minimized:

1. $\mathbf{L}'_0 \neq \mathbf{L}_0$ and $\mathbf{L}'_1 = \mathbf{L}_1 \longleftrightarrow \theta_0 > 0$ and $\theta_1 = 0$.
2. $\mathbf{L}'_0 = \mathbf{L}_0$ and $\mathbf{L}'_1 \neq \mathbf{L}_1 \longleftrightarrow \theta_0 = 0$ and $\theta_1 > 0$.
3. $\mathbf{L}'_0 \neq \mathbf{L}_0$ and $\mathbf{L}'_1 \neq \mathbf{L}_1 \longleftrightarrow \theta_0 > 0$ and $\theta_1 > 0$.

Suppose, for the sake of argument, that one of the first two statements is true. In the first case, lemma 3 states that $\mathbf{m}'_0$ is obtained by projecting $\mathbf{m}_0$ onto the plane with the normal $\mathbf{m}_1 \times \mathbf{t}$, which leads to (3.14) and

$$\sin(\theta_0) = \frac{|\widehat{\mathbf{m}}_0 \cdot (\mathbf{m}_1 \times \mathbf{t})|}{\|\mathbf{m}_1 \times \mathbf{t}\|} = \frac{|\widehat{\mathbf{m}}_0 \cdot (\widehat{\mathbf{m}}_1 \times \mathbf{t})|}{\|\widehat{\mathbf{m}}_1 \times \mathbf{t}\|}. \tag{3.16}$$

Likewise, in the second case, lemma 3 leads to (3.15), and

$$\sin(\theta_1) = \frac{|\widehat{\mathbf{m}}_1 \cdot (\mathbf{m}_0 \times \mathbf{t})|}{\|\mathbf{m}_0 \times \mathbf{t}\|} \overset{(3.1)}{=} \frac{|\widehat{\mathbf{m}}_0 \cdot (\widehat{\mathbf{m}}_1 \times \mathbf{t})|}{\|\widehat{\mathbf{m}}_0 \times \mathbf{t}\|}. \tag{3.17}$$

Now, the question is how to determine which of the two statements is true. Comparing the right-hand side of (3.16) and (3.17), we find that if $\|\widehat{\mathbf{m}}_0 \times \mathbf{t}\| \leq \|\widehat{\mathbf{m}}_1 \times \mathbf{t}\|$, then

$$\min_{\theta_0|\theta_1=0} \theta_0 \leq \min_{\theta_1|\theta_0=0} \theta_1, \tag{3.18}$$

and $\min(\theta_0 + \theta_1)$ is equal to the left-hand side of (3.18), indicating that the first statement is true. Naturally, the second statement is true otherwise. Note that there is an ambiguity if $\|\widehat{\mathbf{m}}_0 \times \mathbf{t}\| = \|\widehat{\mathbf{m}}_1 \times \mathbf{t}\|$, and the solution is optimal whichever case is considered. This concludes the proof of lemma 4 for the first two cases.

We will now prove that the third case never occurs. Given some angle θ_1, minimizing $(\theta_0 + \theta_1)$ is equivalent to minimizing θ_0. This is the identical situation as the first case if we replace $\mathbf{L}_1$ by $\mathbf{L}_1'$. We know from the proof of lemma 3 that pivoting a line to intersect another with minimum angle can be modeled by a cone lying sideways on a plane. The top of Fig. 3.5 illustrates this. Similarly, the bottom of Fig. 3.5 illustrates the minimization of θ_1 with respect to $\mathbf{L}_0'$ given θ_0. Now, since both planes touching each cone contain the same two intersecting lines $\mathbf{L}_0'$ and $\mathbf{L}_1'$, they must be the same plane. Let this plane be Π'. According to lemma 3, $\mathbf{L}_0'$ is the projection of $\mathbf{L}_0$ onto plane Π'. Therefore,

$$\widehat{\mathbf{m}}_0' = \frac{\widehat{\mathbf{m}}_0 - (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}'}{\|\widehat{\mathbf{m}}_0 - (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}'\|}, \tag{3.19}$$

where $\widehat{\mathbf{n}}'$ is the unit normal of plane Π'. Since Π' contains both $\mathbf{c}_0$ and $\mathbf{c}_1$, $\widehat{\mathbf{n}}'$ is perpendicular to $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$. Hence, computing the dot product with $\mathbf{t}$ on each side of (3.19) yields

$$\mathbf{t} \cdot \widehat{\mathbf{m}}_0' = \frac{\mathbf{t} \cdot \widehat{\mathbf{m}}_0}{\|\widehat{\mathbf{m}}_0 - (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}'\|}. \tag{3.20}$$

Note that $\|\widehat{\mathbf{m}}_0 - (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}'\|$ corresponds to the magnitude of the projection of $\widehat{\mathbf{m}}_0$ onto plane Π' for non-zero θ_0, so it must be smaller than $\|\widehat{\mathbf{m}}_0\| = 1$. Thus,

$$|\mathbf{t} \cdot \widehat{\mathbf{m}}_0'| = \frac{|\mathbf{t} \cdot \widehat{\mathbf{m}}_0|}{\|\widehat{\mathbf{m}}_0 - (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}'\|} > |\mathbf{t} \cdot \widehat{\mathbf{m}}_0|. \tag{3.21}$$

Using (3.2), this inequality can be written as

$$\|\mathbf{t} \times \widehat{\mathbf{m}}_0'\| < \|\mathbf{t} \times \widehat{\mathbf{m}}_0\|. \tag{3.22}$$

Analogously, we can also derive

$$\|\mathbf{t} \times \widehat{\mathbf{m}}_1'\| < \|\mathbf{t} \times \widehat{\mathbf{m}}_1\|. \tag{3.23}$$

Now, suppose that

$$\min_{\theta_0,\theta_1} (\theta_0 + \theta_1) = \theta_0^* + \theta_1^* \quad \text{with} \quad \theta_0^*,\ \theta_1^* > 0. \tag{3.24}$$

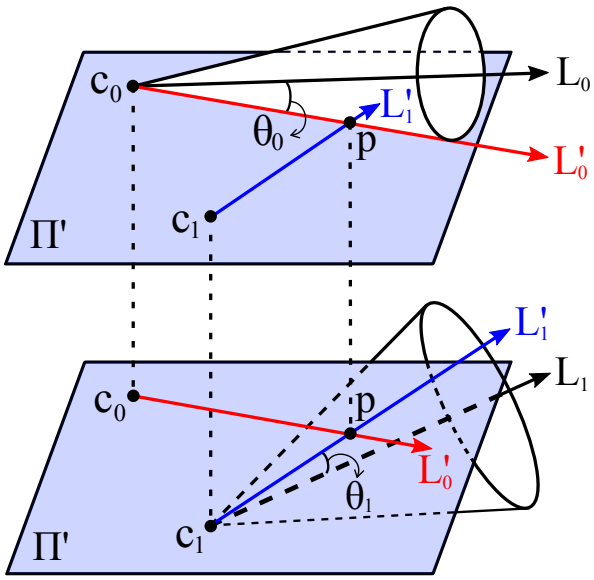


Figure 3.5: When two cones intersect at a single point $\mathbf{p}$ on their lateral surface, they are tangent to the same plane containing $\mathbf{p}$ and the apexes of each cone. For visualization purposes, we show the two cones lying on each side of the plane separately.

Without loss of generality, let us assume that $\|\mathbf{t} \times \widehat{\mathbf{m}}_0\| \leq \|\mathbf{t} \times \widehat{\mathbf{m}}_1\|$. Then, (3.22) gives $\|\mathbf{t} \times \widehat{\mathbf{m}}'_0\| < \|\mathbf{t} \times \widehat{\mathbf{m}}_1\|$. As we discussed for the first two cases, this means that pivoting $\mathbf{L}'_0$ to intersect $\mathbf{L}_1$ takes smaller angle than pivoting $\mathbf{L}_1$ to intersect $\mathbf{L}'_0$, *i.e.*, $\theta'_0 < \theta^*_1$. Thus

$$\theta^*_0 + \theta'_0 < \theta^*_0 + \theta^*_1. \tag{3.25}$$

According to lemma 2 in Section 3.3, pivoting a line twice for intersection takes equal or greater angle than the single minimum pivot angle. Therefore,

$$\min_{\theta_0|\theta_1=0} \theta_0 \leq \theta^*_0 + \theta'_0 < \theta^*_0 + \theta^*_1, \tag{3.26}$$

which contradicts (3.24). Therefore, $(\theta_0 + \theta_1)$ is minimized when either θ_0 or θ_1 is zero. ■

By substituting $\mathbf{R}\mathbf{f}_0$ and $\mathbf{f}_1$ into $\mathbf{m}_0$ and $\mathbf{m}_1$ in the above lemma, the resulting $\mathbf{m}'_0$ and $\mathbf{m}'_1$ become the corrected rays $\mathbf{R}\mathbf{f}'_0$ and $\mathbf{f}'_1$ that satisfy the L_1 optimality, and $\mathbf{n}_0$ (or $\mathbf{n}_1$) becomes the normal of the corresponding epipolar plane.

3.6 Closed-Form L_2 Triangulation

Considering that the angular errors are small in practice, the "relaxed" L_2 triangulation finds the feature rays $\mathbf{f}'_0$ and $\mathbf{f}'_1$ that minimize $\sin^2(\theta_0) + \sin^2(\theta_1)$ (instead of $\theta_0^2 + \theta_1^2$) subject to the epipolar constraint (3.10). Note that the small-angle approximation by $\sin(\theta)$ is more accurate than by $\tan(\theta)$ or $1 - \cos(\theta)$ that have been used in literature (Hartley, Kahl, Olsson and Seo, 2013, Hartley and Schaffalitzky, 2004, Kahl and Hartley, 2008, Recker et al., 2013,

Chesi, 2014). This is easily seen by comparing their Maclaurin expansions. As will be shown in the following lemma (and previously in (Oliensis, 2002)), the relaxation with the sine function allows us to derive the L_2 optimal solution in closed form.

Lemma 5 (L_2 Angle Minimization)
Given two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ *and* $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$*, consider any two intersecting lines that also pass* $\mathbf{c}_0$ *and* $\mathbf{c}_1$*, respectively, i.e.,* $\mathbf{L}'_0(s'_0) = \mathbf{c}_0 + s'_0\mathbf{m}'_0$ *and* $\mathbf{L}'_1(s'_1) = \mathbf{c}_1 + s'_1\mathbf{m}'_1$*. Let* $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$*,* $\theta_0 = \angle(\mathbf{L}_0, \mathbf{L}'_0)$ *and* $\theta_1 = \angle(\mathbf{L}_1, \mathbf{L}'_1)$*. Then,* $\left(\sin^2\theta_0 + \sin^2\theta_1\right)$ *is minimized for*

$$\mathbf{m}'_i = \mathbf{m}_i - (\mathbf{m}_i \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}' \quad \textit{for} \quad i = 0, 1, \tag{3.27}$$

where $\widehat{\mathbf{n}}'$ *is the second column of the* 3×3 *matrix* $\mathbf{V}$ *from*

$$\mathbf{U}\mathbf{S}\mathbf{V}^\mathsf{T} = \textit{SVD}\left(\begin{bmatrix}\widehat{\mathbf{m}}_0 & \widehat{\mathbf{m}}_1\end{bmatrix}^\mathsf{T}\left(\mathbf{I} - \widehat{\mathbf{t}}\,\widehat{\mathbf{t}}^\mathsf{T}\right)\right). \tag{3.28}$$

Proof. Given some angle θ_1, $\left(\sin^2(\theta_0) + \sin^2(\theta_1)\right)_{\text{min}}$ is achieved by minimizing θ_0 and vice versa. As discussed in the proof of lemma 4, this means that the underlying geometry at $\left(\sin^2(\theta_0) + \sin^2(\theta_1)\right)_{\text{min}}$ can be represented by the two cones with apex $\mathbf{c}_0$, $\mathbf{c}_1$ and skew axes $\mathbf{L}_0$, $\mathbf{L}_1$, respectively, touching each side of the same plane on their lateral surface. This is visualized in Fig. 3.5. Let $\mathbf{n}'$ be the normal of plane Π'. From Lemma 3, we know that

$$\sin(\theta_0) = |\widehat{\mathbf{n}}' \cdot \widehat{\mathbf{m}}_0| \quad \text{and} \quad \sin(\theta_1) = |\widehat{\mathbf{n}}' \cdot \widehat{\mathbf{m}}_1|. \tag{3.29}$$

Combining these two equations, we get

$$\sin^2(\theta_0) + \sin^2(\theta_1) = \|\mathbf{M}^\mathsf{T}\widehat{\mathbf{n}}'\|^2 \quad \text{with} \quad \mathbf{M} = \begin{bmatrix}\widehat{\mathbf{m}}_0 & \widehat{\mathbf{m}}_1\end{bmatrix}. \tag{3.30}$$

Since plane Π' contains both $\mathbf{c}_0$ and $\mathbf{c}_1$, $\widehat{\mathbf{n}}'$ is perpendicular to $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$. Therefore, minimizing $(\sin^2(\theta_0) + \sin^2(\theta_1))$ is equivalent to solving the following equality-constrained quadratic programming problem:

$$\underset{\widehat{\mathbf{n}}'}{\operatorname{argmin}} \|\mathbf{M}^\mathsf{T}\widehat{\mathbf{n}}'\|^2, \text{ s.t. } \|\widehat{\mathbf{n}}'\| = 1 \text{ and } \mathbf{t} \cdot \widehat{\mathbf{n}}' = 0. \tag{3.31}$$

In (Golub, 1973), it was shown that this problem can be solved using the method of Lagrange multipliers, and $\|\mathbf{M}^\mathsf{T}\widehat{\mathbf{n}}'\|^2$ is minimized when $\widehat{\mathbf{n}}'$ is the eigenvector corresponding to the smallest nontrivial eigenvalue of $\mathbf{A} = (\mathbf{I} - \widehat{\mathbf{t}}\,\widehat{\mathbf{t}}^\mathsf{T})\mathbf{M}\mathbf{M}^\mathsf{T}$. Letting $\mathbf{P} = (\mathbf{I} - \widehat{\mathbf{t}}\,\widehat{\mathbf{t}}^\mathsf{T})$, it can be easily shown that $\mathbf{P} = \mathbf{P}^\mathsf{T} = \mathbf{P}^\mathsf{T}\mathbf{P} = \mathbf{P}\mathbf{P}^\mathsf{T}$. Hence, $\mathbf{A} = \mathbf{P}\mathbf{M}\mathbf{M}^\mathsf{T} = \mathbf{P}\mathbf{P}^\mathsf{T}\mathbf{M}\mathbf{M}^\mathsf{T}$. Note that for any square matrix $\mathbf{X}$ and $\mathbf{Y}$, the eigen-decomposition of $\mathbf{X}\mathbf{Y}$ is the same as that of $\mathbf{Y}\mathbf{X}$. This means that the eigenvectors of $\mathbf{A} = \mathbf{P}\left(\mathbf{P}^\mathsf{T}\mathbf{M}\mathbf{M}^\mathsf{T}\right)$ are the same as those of $\left(\mathbf{P}^\mathsf{T}\mathbf{M}\mathbf{M}^\mathsf{T}\right)\mathbf{P} = (\mathbf{M}^\mathsf{T}\mathbf{P})^\mathsf{T}(\mathbf{M}^\mathsf{T}\mathbf{P})$, *i.e.*, the right-singular vectors of $\mathbf{M}^\mathsf{T}\mathbf{P}$. Therefore, letting $\mathbf{U}\mathbf{S}\mathbf{V}^\mathsf{T} = \text{SVD}\left(\mathbf{M}^\mathsf{T}\mathbf{P}\right)$ with the diagonal entries of $\mathbf{S}$ in descending order, the optimal $\widehat{\mathbf{n}}'$

is given by the second column of $\mathbf{V}$. Finally, projecting $\mathbf{m}_0$ and $\mathbf{m}_1$ onto plane Π' leads to (3.27). ■

Analogously to the L_1 method, substituting $\mathbf{R}\mathbf{f}_0$ and $\mathbf{f}_1$ into $\mathbf{m}_0$ and $\mathbf{m}_1$ in the above lemma gives $\mathbf{R}\mathbf{f}'_0 = \mathbf{m}'_0$ and $\mathbf{f}'_1 = \mathbf{m}'_1$ that satisfy the L_2 optimality.

3.7 Closed-Form L_∞ Triangulation

The L_∞ triangulation based on the angular reprojection error (3.12) finds the feature rays $\mathbf{f}'_0$ and $\mathbf{f}'_1$ that minimize $\max(\theta_0, \theta_1)$ subject to the epipolar constraint (3.10). The following lemma states that this is achieved when $\theta_0 = \theta_1$:

Lemma 6 (L_∞ Angle Minimization)
Given two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ and $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$, consider any two intersecting lines that also pass $\mathbf{c}_0$ and $\mathbf{c}_1$, respectively, i.e., $\mathbf{L}'_0(s'_0) = \mathbf{c}_0 + s'_0\mathbf{m}'_0$ and $\mathbf{L}'_1(s'_1) = \mathbf{c}_1 + s'_1\mathbf{m}'_1$. Let $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$, $\mathbf{n}_a = (\widehat{\mathbf{m}}_0 + \widehat{\mathbf{m}}_1) \times \mathbf{t}$, $\mathbf{n}_b = (\widehat{\mathbf{m}}_0 - \widehat{\mathbf{m}}_1) \times \mathbf{t}$, $\theta_0 = \angle(\mathbf{L}_0, \mathbf{L}'_0)$ and $\theta_1 = \angle(\mathbf{L}_1, \mathbf{L}'_1)$. Then, $\max(\theta_0, \theta_1)$ is minimized when $\theta_0 = \theta_1$. This is achieved for

$$\mathbf{m}'_i = \mathbf{m}_i - (\mathbf{m}_i \cdot \widehat{\mathbf{n}}')\widehat{\mathbf{n}}' \quad \textit{for} \quad i = 0, 1, \tag{3.32}$$

where

$$\mathbf{n}' = \begin{cases} \mathbf{n}_a & \textit{if } \|\mathbf{n}_a\| \geq \|\mathbf{n}_b\| \\ \mathbf{n}_b & \textit{ohterwise} \end{cases} \tag{3.33}$$

Proof. First, we show that $\theta_0 = \theta_1$ when $\max(\theta_0, \theta_1)$ is minimized: Consider two cones with apex $\mathbf{c}_0$, $\mathbf{c}_1$ and skew axes $\mathbf{L}_0$, $\mathbf{L}_1$. Constrain both their apertures to be 2θ. When $\theta = 0$, the they are simply two skew lines. As we gradually increase θ, they will grow at the same rate, and eventually, touch one another. Let $\theta = \theta'$ at this point. Now, suppose

$$\theta^* := \min_{\theta_0, \theta_1} \max(\theta_0, \theta_1) < \theta'. \tag{3.34}$$

The definition of θ^* implies that setting $\theta_0 = \theta_1 = \theta^*$ will make the two cones partially overlap in space (or at least meet at a point). However, the they do not meet when $\theta_0 = \theta_1 < \theta'$. This is a contradiction, so the inequality in (3.34) must be false, and θ^* must be equal to θ'. That is, $\theta_0 = \theta_1 = \theta'$ in order for $\max(\theta_0, \theta_1)$ to be minimized.

We can now represent the underlying geometry at $(\max(\theta_0, \theta_1))_{\min}$ as two congruent cones with skew axes, touching each side of the same plane Π' on their lateral surface. This is the situation shown in Fig. 3.5 for $\theta_0 = \theta_1$. Let $\mathbf{n}'$ be the normal of plane Π'. Then, from lemma 3, we get

$$\sin(\theta_0) = \sin(\theta_1) = |\widehat{\mathbf{n}}' \cdot \widehat{\mathbf{m}}_0| = |\widehat{\mathbf{n}}' \cdot \widehat{\mathbf{m}}_1|. \tag{3.35}$$

The last equality in (3.35) can be written as

$$(\widehat{\mathbf{m}}_0 + w\widehat{\mathbf{m}}_1) \cdot \widehat{\mathbf{n}}' = 0, \tag{3.36}$$

where w is -1 or 1, depending on the signs of $\widehat{\mathbf{n}}' \cdot \widehat{\mathbf{m}}_0$ and $\widehat{\mathbf{n}}' \cdot \widehat{\mathbf{m}}_1$. On the other hand, since plane Π' contains both $\mathbf{c}_0$ and $\mathbf{c}_1$, $\widehat{\mathbf{n}}'$ is perpendicular to $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$:

$$\mathbf{t} \cdot \widehat{\mathbf{n}}' = 0. \tag{3.37}$$

Combining (3.36) and (3.37), $\widehat{\mathbf{n}}'$ can be expressed as

$$\widehat{\mathbf{n}}' = \lambda(\widehat{\mathbf{m}}_0 + w\widehat{\mathbf{m}}_1) \times \mathbf{t} \tag{3.38}$$

where λ is the normalizing factor. Evaluating (3.38) at $w = 1$ and $w = -1$ gives two candidates for optimal $\widehat{\mathbf{n}}'$. The optimal solution is then determined by comparing the values of (3.35) with each candidate $\widehat{\mathbf{n}}'$, which amounts to choosing the solution with smaller λ. This procedure corresponds to (3.33). Finally, projecting $\mathbf{m}_0$ and $\mathbf{m}_1$ onto plane Π' with optimal $\widehat{\mathbf{n}}'$ leads to (3.32). ■

Analogously to the previous two methods, substituting $\mathbf{R}\mathbf{f}_0$ and $\mathbf{f}_1$ into $\mathbf{m}_0$ and $\mathbf{m}_1$ gives $\mathbf{R}\mathbf{f}'_0 = \mathbf{m}'_0$ and $\mathbf{f}'_1 = \mathbf{m}'_1$ that satisfy the L_∞ optimality.

3.8 Cheirality, Parallax and Outliers

We have used the term *lines* instead of *rays* in all lemmas so far, ignoring the cheirality constraint (Hartley and Zisserman, 2003). We argue that if the optimal solution violates the cheirality constraint, the most reasonable choice is to simply discard the result. In the following, we provide the rationale for this choice.

Fig. 3.6 illustrates five scenarios where the optimal solution violates the cheirality constraint. In case (a), both rays have negative depths at the optimal intersection. Increasing the allowed angular reprojection error, the first intersection with positive depths occurs when the two corrected rays become parallel, resulting in a point at infinity. This point cannot be triangulated, so it should be discarded.

In the remaining cases, the optimal intersection involves only one of the rays having a negative depth. Following the same procedure, the first intersection with positive depths occurs either at infinity (case (b)), at one of the camera centers (case (c)), along the ray parallel to the translation (case (d)), or at a point somewhere else (case (e)).

In case (b), (c) and (d), the newly triangulated point has either infinite, zero or ambiguous depth, so it is reasonable to discard it. In case (e), we found that reattempting the triangulation with positive depths yields either a very large error, a point near the epipole or a low parallax angle. Typically, these are the indicators of low accuracy or an outlier (Hartley and

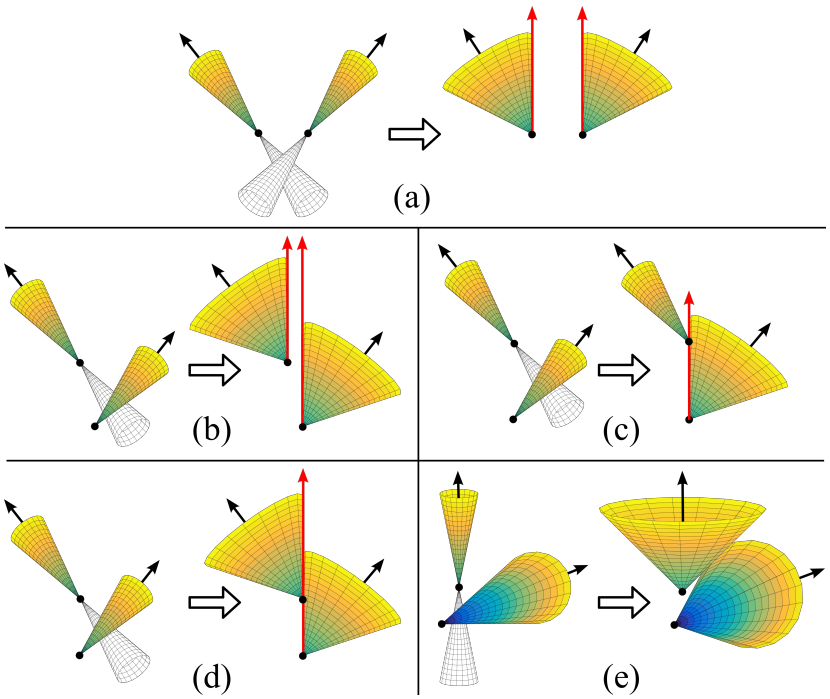


Figure 3.6: Five scenarios where the optimal solution violates the cheirality constraint, and the possible reattempts for triangulation.

Sturm, 1997, Hartley and Zisserman, 2003), so a reasonable choice is to discard the match. This procedure is outlined in Step 4–6 of Tab. 3.1.

3.9 Experimental Results

We evaluate the proposed methods ('L_1 ang', 'L_2 ang' and 'L_∞ ang') in comparison to the midpoint method, 'Midpoint' (Beardsley et al., 1994, Hartley and Sturm, 1997), Hartley and Sturm's L_1 and L_2 method, 'L_1 img' and 'L_2 img' (Hartley and Sturm, 1997), Lindstrom's L_2 method with k iterations, 'L_2 img (k)' (Lindstrom, 2010), and Níster's L_∞ method, 'L_∞ img' (Níster, 2001). The evaluation was performed on both synthetic and real datasets. We generated the synthetic datasets as follows: A set of 8×4 point clouds of 2,500 points each are generated with a Gaussian radial distribution $\mathcal{N}(0, (d/4)^2)$ where d is the distance from the world origin. Each point cloud is centered at $[0, 0, d]^\intercal$ for $d = 2^n$ with $n = -1, 0, ..., +6$, and their image projections are perturbed by Gaussian noise $\mathcal{N}(0, \sigma^2)$ for $\sigma = 0.5, 1, 2, 4, 8$. The size and the focal length of the images are $1,024 \times 1,024$ pixels and 512 pixel, respectively. We have three configurations for the camera poses: (1) "orbital" - the cameras at $[\pm 0.5, 0, 0]^\intercal$ pointing at the point cloud center, (2) "lateral" - the cameras at $[\pm 0.5, 0, 0]^\intercal$ pointing at $[0, 0, \infty]^\intercal$, and (3) "forward" - the cameras at $[0, 0, \pm 0.5]^\intercal$ pointing at the point cloud center. The poses are slightly perturbed with uniform noise $\mathcal{U}(0, 0.01)$. For real datasets, we used the Oxford Dinosaur, Model House and Corridor (*Oxford Multiview Datasets*, n.d.), Notre Dame (Snavely et al., 2006) and Fountain (Enqvist et al., 2011, Olsson et al., 2011) dataset. In total, the synthetic and real datasets provide over 5.5 million unique

Input: Calib. matrix ($\mathbf{K}$), relative pose ($\mathbf{R}$, $\mathbf{t}$), and a match ($\mathbf{u}_0$, $\mathbf{u}_1$) from two views (C_0, C_1).
Output: Triangulated 3D point ($\mathbf{x}'_1$) in ref. frame C_1.
1) $\mathbf{f}_0 \leftarrow \mathbf{K}^{-1}\mathbf{u}_0$, $\mathbf{f}_1 \leftarrow \mathbf{K}^{-1}\mathbf{u}_1$, $\mathbf{m}_0 \leftarrow \mathbf{R}\mathbf{f}_0$, $\mathbf{m}_1 \leftarrow \mathbf{f}_1$.
2) **For L_1 triangulation:**
If $\|\widehat{\mathbf{m}}_0 \times \mathbf{t}\| \leq \|\widehat{\mathbf{m}}_1 \times \mathbf{t}\|$, use (3.14) to obtain $\mathbf{m}'_0$ and $\mathbf{m}'_1$. Otherwise, use (3.15).
For L_2 triangulation:
Compute $\mathbf{m}'_0$ and $\mathbf{m}'_1$ from (3.27) and (3.28).
For L_∞ triangulation:
Compute $\mathbf{m}'_0$ and $\mathbf{m}'_1$ from (3.32) and (3.33).
3) $\mathbf{R}\mathbf{f}'_0 \leftarrow \mathbf{m}'_0$ and $\mathbf{f}'_1 \leftarrow \mathbf{m}'_1$.
4) Check cheirality:
(i) Obtain λ_0 and λ_1 from (3.13).
(ii) Discard the point and terminate if either $\lambda_0 \leq 0$ or $\lambda_1 \leq 0$.
5) Check angular reprojection errors:
(i) $\theta_0 \leftarrow \angle(\mathbf{R}\mathbf{f}_0, \mathbf{R}\mathbf{f}'_0)$ and $\theta_1 \leftarrow \angle(\mathbf{f}_1, \mathbf{f}'_1)$.
(ii) Discard the point and terminate if $\max(\theta_0, \theta_1) > \epsilon_1$ for some small ϵ_1.
6) Check parallax:
(i) $\beta \leftarrow \angle(\mathbf{R}\mathbf{f}'_0, \mathbf{f}'_1)$
(ii) Discard the point and terminate if $\beta < \epsilon_2$ for some small ϵ_2.
7) Compute and return $\mathbf{x}'_1$ from (3.13).

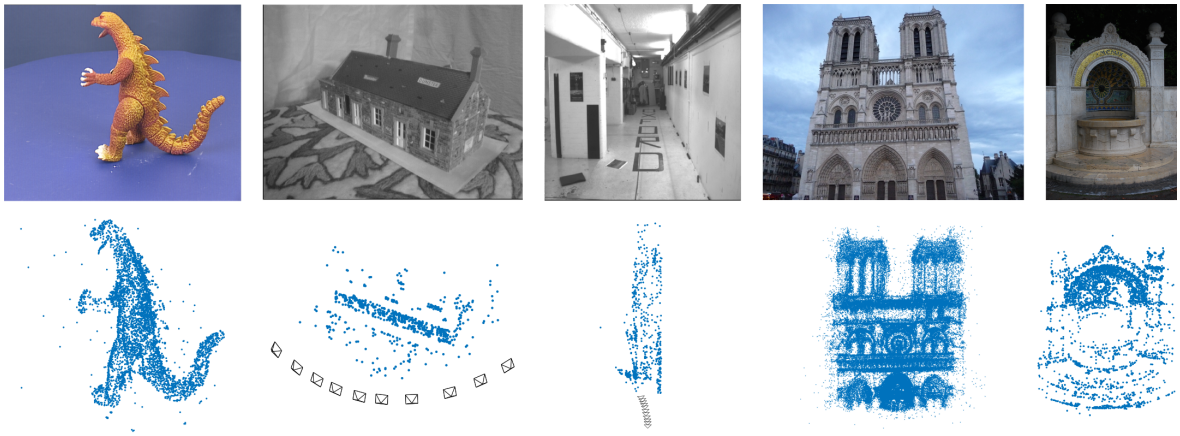
Table 3.1: Summary of the proposed optimal triangulation methods.

triangulation problems in a wide variety of geometric configurations.

Tab. 3.2 provides the percentage of the total matches (from both synthetic and real datasets) for which each method yields the lowest error in given criterion. In 100 % of the total triangulation problems, all three of our methods yield the lowest errors in their corresponding optimal criterion. We also see that minimizing $\sin^2(\theta_0) + \sin^2(\theta_1)$ is very close to minimizing $\theta_0^2 + \theta_1^2$, as discussed in Section 3.6. Since our L_1 angular method is numerically stable, it sometimes finds better solutions than Hartley-Sturm's closed-form L_1 method (Hartley and Sturm, 1997) even in the L_1 image error criterion ($d_0 + d_1$).

In Fig. 3.8, histograms are given for the 3D reconstruction errors on the synthetic datasets. It shows that 1) discarding low-parallax points (Step 6 of Tab. 3.1) helps to remove large 3D errors, and 2) all methods then exhibit similar 3D accuracy. Qualitatively, we also found that the reconstructions of the real datasets look similar for all methods. Fig. 3.7 shows the reconstruction results using the proposed L_1 method.

We compare the speed of each algorithm in Tab. 3.3. The midpoint method is the fastest, as it directly computes the 3D point using (3.13) without correcting the feature rays or image points. Among the optimal methods, our L_1 and L_∞ methods are significantly faster than the rest, *i.e.*, at least 1–2 orders of magnitude faster than the state-of-the-art (Lindstrom, 2010).



(a) Dinosaur (b) Model House (c) Corridor (d) Notre Dame (e) Fountain

Figure 3.7: **Top row:** Real dataset images. **Bottom row:** Main segments of the median reconstruction results using the proposed L_1 method.

		Midpoint	L_1 img	L_2 img	L_2 img (5)	L_∞ img	L_1 ang	L_2 ang	L_∞ ang
Error Criterion	$\theta_0 + \theta_1$	-	-	-	-	-	**100 %**	-	-
	$\theta_0^2 + \theta_1^2$	-	-	7e-5 %	5e-5 %	-	-	**99.9999 %**	-
	$\sin^2(\theta_0) + \sin^2(\theta_1)$	-	-	-	-	-	-	**100 %**	-
	$\max(\theta_0, \theta_1)$	-	-	-	-	-	-	-	**100 %**
	$d_0 + d_1$	-	**70.84%**	0.002%	0.002%	-	29.16 %	-	-
	$d_0^2 + d_1^2$	-	-	23.14 %	**76.86 %**	-	-	-	-
	$\max(d_0, d_1)$	-	-	-	-	**100 %**	-	-	-

Table 3.2: Percentage of the total matches (from all synthetic and real datasets) for which each method yields the lowest error in given criterion. "img/ang": optimal in the image/angular errors.

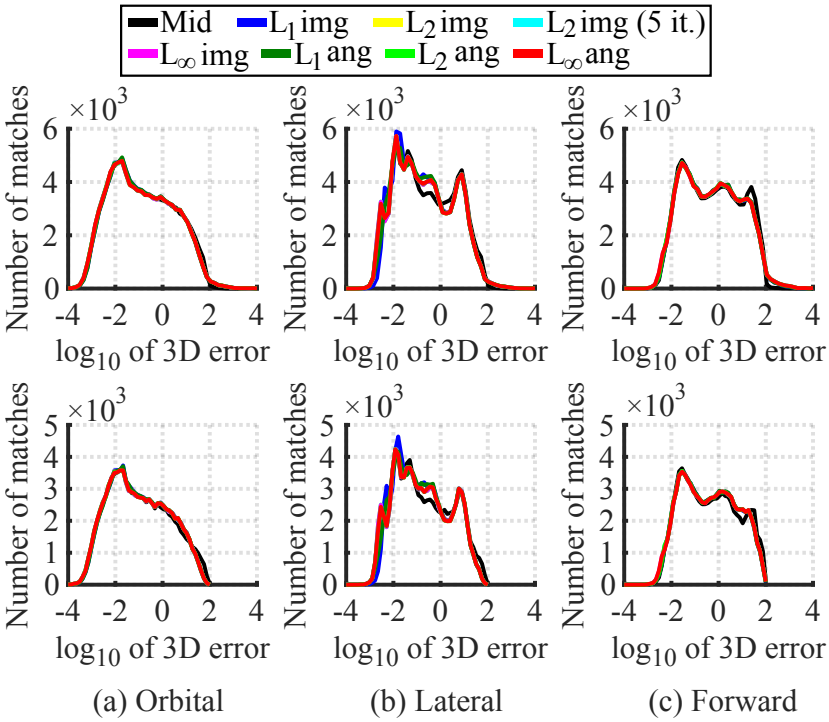


Figure 3.8: 3D triangulation errors before (top) and after (bottom) discarding the points with the lowest 5% parallax.

3.10 Conclusions

In this chapter, we derived optimal closed-form solutions to the L_1, L_2 and L_∞ stereo triangulation based on the angular reprojection error. The proposed triangulation methods are extremely simple and fast, and they guarantee global optimality under respective cost functions. We believe that our findings will be particularly useful for large-scale SfM and real-time visual SLAM algorithms.

	Midpoint	L_1 img	L_2 img	L_∞ img	L_2 img (2)	L_2 img (5)	L_1 ang	L_2 ang	L_∞ ang
Points/sec	42 M	65 K	92 K	270 K	1.4 M	520 K	29 M	670 K	14 M
Relative Speed	1.0	0.0016	0.0022	0.0064	0.033	0.013	0.71	0.016	0.33

Table 3.3: Speed of computing a 3D point. The relative speed is normalized by that of the midpoint method. Note that this does not take into account Step 4–6 of Tab. 3.1. All algorithms were implemented in C++ and run on a laptop CPU (Intel i7-4810MQ, 2.8 GHz).

Chapter 4

Triangulation: Why Optimize?

For decades, it has been widely accepted that the gold standard for two-view triangulation is to minimize the cost based on reprojection errors. In this chapter, we challenge this idea. We propose a novel alternative to the classic midpoint method that leads to significantly lower 2D errors and parallax errors. It provides a numerically stable closed-form solution based solely on a pair of backprojected rays. Since our solution is rotationally invariant, it can also be applied for fisheye and omnidirectional cameras. We show that for small parallax angles, our method outperforms the state-of-the-art in terms of combined 2D, 3D and parallax accuracy, while achieving comparable speed.

4.1 Introduction

Locating the 3D point given its projections in multiple views is called triangulation. This classic yet fundamental problem in computer vision has immediate relevance to many applications, including visual odometry (Forster et al., 2014), simultaneous localization and mapping (SLAM) (Mur-Artal et al., 2015) and structure-from-motion (SfM) (Schönberger and Frahm, 2016). As such, achieving fast and accurate triangulation has been a goal of many research endeavors in the past decades.

For two views of known calibration and pose, the problem could be solved ideally if one finds the intersection of two backprojected rays corresponding to the same point. However, the two rays are most likely skew due to noisy measurements and inaccurate camera model. Since it is not obvious how to estimate the 3D position of the point from two skew rays, different methods have been proposed. Mainly, they can be classified into three types: (1) midpoint methods (Beardsley et al., 1994, 1997, Yang et al., 2019) that find the (weighted) midpoint of the common perpendicular between the two rays, (2) linear least squares methods (Hartley and Sturm, 1997), and (3) optimal methods that "minimally" correct the two rays to make them intersect (Kanatani et al., 2008, Hartley and Sturm, 1997, Lee and Civera, 2019*a*). Note that all these three types of methods produce solutions that minimize some cost function; the (weighted) midpoint minimizes the (weighted) sum of squared distances

to each ray, linear least squares methods minimize the algebraic errors, and optimal methods minimize a cost function based on either *image* reprojection errors (Kanatani et al., 2008, Hartley and Sturm, 1997, Lindstrom, 2010) or *angular* reprojection errors (Oliensis, 2002, Lee and Civera, 2019*a*). The most common cost functions are the L_1 norm (sum of magnitude), L_2 norm (sum of squares) and L_∞ norm (maximum) of the reprojection errors.

In this chapter, we suggest a different approach. Instead of minimizing geometric or algebraic errors, we find a midpoint between a certain pair of points on each ray. Like the classic midpoint method, our method takes the two rays as input. Therefore, it is invariant to changes of camera rotation and applicable for perspective, fisheye and omnidirectional cameras. Unlike the classic method, however, the two points on each ray are not necessarily on the common perpendicular. We will see that our midpoint method bears a striking similarity to the classic method in the formulation, and yet it offers a significant performance gain in 2D and parallax accuracy. Compared to the optimal methods, our method yields much lower 3D errors at low parallax and similar 2D errors to those of L_2 and L_∞ optimal methods. This motivates the question: In two-view triangulation, why optimize if there is a better way?

The main contributions of this chapter are the following:

- We propose a novel method belonging to a group called the generalized weighted midpoint (GWM) method. We show that our method outperforms existing ones (including the classic midpoint method and the state-of-the-art optimal methods) in terms of combined 2D, 3D and parallax accuracy.
- Additionally, we propose a test of the adequacy (similar to the cheirality check) that identifies unreliable results and a weighting scheme that enhances 2D accuracy.
- We perform an extensive evaluation and analysis of various methods, revealing an intricate link between 3D accuracy and parallax estimation. This will provide an intuitive explanation of why our midpoint method performs better than the others.

4.2 Related Work

One of the earliest works that addressed the two-view triangulation problem is (Longuet-Higgins, 1981) where the depth of a 3D point is estimated using simple algebra. For robustness, later works mostly adopted geometric approaches, such as the midpoint method (Beardsley et al., 1994, 1997) and the minimization of the epipolar distance (Harris, 1987, Harris and Pike, 1988) or reprojection error (Hartley, 1994). Among those, the last approach has become the *de facto* standard in computer vision (Hartley and Zisserman, 2003).

Optimal methods refer to those triangulation methods that minimize the cost based on reprojection errors. Assuming that the image measurements are independently perturbed by the noise in the same distribution of certain types, the optimal methods find the maximum likelihood (ML) solution. For Gaussian and Laplacian distribution, the ML solution is to minimize the L_2 norm or L_1 norm of the reprojection errors, respectively (Ke and Kanade,

2003). This can be found in closed form by solving a polynomial of degree six or eight (Hartley and Sturm, 1997). Alternatively, the L_2 solution can be obtained using iterative correction methods (Kanatani et al., 2008, Lindstrom, 2010). While these iterative methods do not guarantee global optimality, they were shown to be faster and more stable. For a uniform distribution, minimizing the L_∞ norm leads to the ML estimate for the upper bound of the noise (Hartley and Kahl, 2007), and the solution is obtained by solving a quartic polynomial (Níster, 2001). Unlike the L_1 and L_2 cost, the L_∞ cost has a simple shape with a single minimum, but it is relatively more sensitive to noise and outliers (Hartley and Schaffalitzky, 2004).

These optimal methods assume that the image measurement errors follow certain distributions. However, this assumption is neither justified nor likely (Hartley and Kahl, 2007). An equally (if not more) justified alternative is to assume that the noise model applies to the bearing measurements instead of the image. For fisheye or omnidirectional cameras, the angular reprojection errors are more suitable than the image reprojection errors (Oliensis, 2002, Mičušík and Pajdla, 2006). Also, formulating the triangulation problem in terms of angular errors leads to much simpler ML solutions (Oliensis, 2002, Lee and Civera, 2019*a*).

Although the existing optimal methods can provide relatively good 3D results in many cases (Hartley and Sturm, 1997, Lee and Civera, 2019*a*), none of them are theoretically optimal in terms of 3D errors. In fact, the discrepancy between 2D optimality and 3D accuracy has already been reported by Hartley and Sturm (Hartley and Sturm, 1997). They found that in Euclidean reconstruction, the midpoint and the linear least squares method achieve higher 3D accuracy than the L_1 or L_2 optimal methods, despite consistently (and sometimes significantly) larger 2D errors. In this chapter, we provide additional insights on this matter. Furthermore, we show that a simple modification to the midpoint method can substantially reduce the 2D errors while maintaining 3D accuracy.

4.3 Preliminaries

Throughout the chapter, we use bold letters for vectors and matrices, and light letters for scalars. The angle between two lines $\mathbf{L}_0$ and $\mathbf{L}_1$ is denoted by $\angle(\mathbf{L}_0,\ \mathbf{L}_1) \in [0, \pi/2]$.

Consider a 3D point observed by two cameras C_0 and C_1. We define $\mathbf{x}_0 = [x_0, y_0, z_0]^\intercal$ and $\mathbf{x}_1 = [x_1, y_1, z_1]^\intercal$ as the unknown 3D coordinates of the point in the camera reference frame C_0 and C_1, respectively. Let $\mathbf{R}$ and $\mathbf{t}$ be the known rotation and translation between the two cameras, such that $\mathbf{x}_1 = \mathbf{R}\mathbf{x}_0 + \mathbf{t}$. Assuming that the camera calibration matrix $\mathbf{K}$ is known, the normalized image coordinates $\mathbf{f}_0 = [x_0/z_0, y_0/z_0, 1]^\intercal$ and $\mathbf{f}_1 = [x_1/z_1, y_1/z_1, 1]^\intercal$ can be obtained by $\mathbf{f}_0 = \mathbf{K}^{-1}\mathbf{u}_0$ and $\mathbf{f}_1 = \mathbf{K}^{-1}\mathbf{u}_1$, where $\mathbf{u}_0 = (u_0, v_0, 1)^\intercal$ and $\mathbf{u}_1 = (u_1, v_1, 1)^\intercal$ are the homogeneous pixel coordinates of the point observation in each frame.

In the ideal situation (Fig. 4.1a), the two backprojected rays intersect, satisfying the epipolar constraint (Longuet-Higgins, 1981), *i.e.*, $\mathbf{f}_1 \cdot (\mathbf{t} \times \mathbf{R}\mathbf{f}_0) = 0$. Then, the intersection is given by $\mathbf{x}_1 = \lambda_0 \mathbf{R}\widehat{\mathbf{f}}_0 + \mathbf{t}$ or $\mathbf{x}_1 = \lambda_1 \widehat{\mathbf{f}}_1$ for some scalar depth λ_0 and λ_1. However,

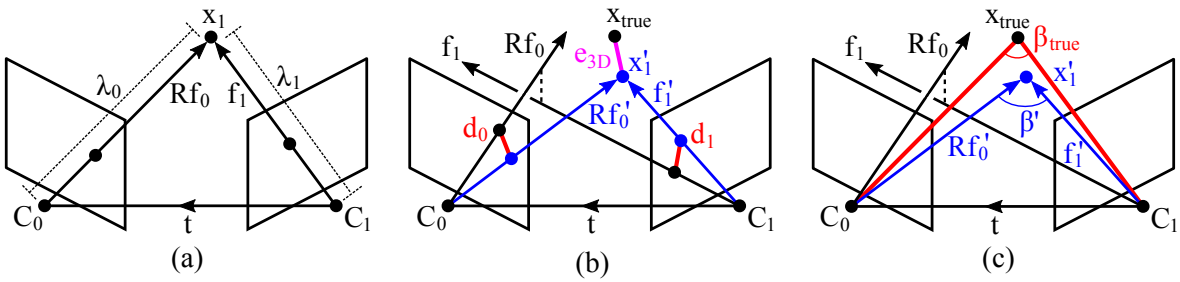


Figure 4.1: **(a)** Epipolar geometry when the two backprojected rays intersect. All vectors are in the coordinate system of C_1. **(b)** 3D error (e_{3D}) measures the Euclidean distance between the estimate ($\mathbf{x}'_1$) and the true position of the point ($\mathbf{x}_{\text{true}}$), while 2D error (d_0, d_1) measures the offset between the observation and the reprojection of the estimated point in each frame. **(c)** After the triangulation, one can estimate the parallax (β') from the corrected rays.

this rarely happens due to inaccuracies in the image measurements and the camera model. Inferring a 3D point from two skew rays requires a nontrivial method.

Once the estimate of the 3D point ($\mathbf{x}'_1$) is obtained using some triangulation method, its accuracy can be evaluated in several ways. One way is to compute the 3D error, *i.e.*, $e_{3D} = \|\mathbf{x}'_1 - \mathbf{x}_{\text{true}}\|$. Another way is to compute the 2D error (aka the reprojection error), *i.e.*,

$$d_i = \|\mathbf{K}\,(\mathbf{f}_i - \mathbf{f}_i')\| = \left\|\mathbf{K}\left(\mathbf{f}_i - ([0\ 0\ 1]\,\mathbf{x}'_i)^{-1}\,\mathbf{x}'_i\right)\right\| \quad \text{for} \quad i = 0, 1, \tag{4.1}$$

where $\mathbf{x}'_0 = \mathbf{R}^\intercal\,(\mathbf{x}'_1 - \mathbf{t})$. These two errors are illustrated in Fig. 4.1b. Note that the 2D error represents the deviation from the measurement, whereas the 3D error represents the deviation from the ground truth. Also, unlike the 3D error, the 2D error of a 3D point can be evaluated in different norms, *e.g.*, L_1 norm $(d_0 + d_1)$, L_2 norm $\left(\sqrt{d_0^2 + d_1^2}\right)$ and L_∞ norm $(\max(d_0, d_1))$. Besides 2D and 3D accuracy, we can also evaluate the accuracy of the resulting parallax angle (see Fig. 4.1c). The parallax error is defined as follows:

$$e_\beta = |\beta_{\text{true}} - \beta'| = |\angle\,(\mathbf{x}_{\text{true}}, \mathbf{x}_{\text{true}} - \mathbf{t}) - \angle\,(\mathbf{x}'_1, \mathbf{x}'_1 - \mathbf{t})\,|. \tag{4.2}$$

We define the "raw parallax" as the angle between the original backprojected rays:

$$\beta_{\text{raw}} = \angle\,(\mathbf{R}\mathbf{f}_0, \mathbf{f}_1)\,. \tag{4.3}$$

This gives a rough estimate of the parallax angle independently of the translation and the triangulation method.

4.4 Proposed Method

4.4.1 Generalized Weighted Midpoint (GWM) Method

A GWM method consists of three steps: (1) Given two backprojected rays corresponding to the same point, estimate the depth along each ray (λ_0, λ_1) using some method. (2) Compute the 3D point on each ray at depth λ_0 and λ_1, *i.e.*, $\mathbf{t} + \lambda_0 \mathbf{R}\widehat{\mathbf{f}}_0$ and $\lambda_1 \widehat{\mathbf{f}}_1$ in C_1. (3) Obtain the final estimate of the 3D point by computing their weighted average.

The classic midpoint method (Beardsley et al., 1994, 1997, Hartley and Sturm, 1997) is one such type of method where the two points on each ray are the closest pair of points with the equal weight. Fig. 4.2 shows another possible example of the generalized weighted midpoint.

4.4.2 Alternative Midpoint Method

We propose an alternative midpoint method that belongs to the GWM method. First, consider the case where the two backprojected rays happen to intersect (see Fig. 4.1a). In this case, the most sensible solution is the point of intersection, and the corresponding depths along the rays can be obtained using the sine rule:

$$\lambda_0 = \frac{\sin\left(\angle\left(\mathbf{f}_1, \mathbf{t}\right)\right)}{\sin\left(\angle\left(\mathbf{R}\mathbf{f}_0, \mathbf{f}_1, \right)\right)}\|\mathbf{t}\| = \frac{\|\widehat{\mathbf{f}}_1 \times \mathbf{t}\|}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\|}, \quad \lambda_1 = \frac{\sin\left(\angle\left(\mathbf{R}\mathbf{f}_0, \mathbf{t}\right)\right)}{\sin\left(\angle\left(\mathbf{R}\mathbf{f}_0, \mathbf{f}_1, \right)\right)}\|\mathbf{t}\| = \frac{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \mathbf{t}\|}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\|}. \tag{4.4}$$

We use this formula to estimate the depths even when the two rays are skew. Computing the 3D points on each ray at depth λ_0 and λ_1, respectively, we get

$$\mathbf{t} + \lambda_0 \mathbf{R}\widehat{\mathbf{f}}_0 = \mathbf{t} + \frac{\|\mathbf{f}_1 \times \mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0 \times \mathbf{f}_1\|}\mathbf{R}\mathbf{f}_0 \quad \text{and} \quad \lambda_1 \widehat{\mathbf{f}}_1 = \frac{\|\mathbf{R}\mathbf{f}_0 \times \mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0 \times \mathbf{f}_1\|}\mathbf{f}_1, \tag{4.5}$$

Taking the midpoint between these two points leads to

$$\mathbf{x}_1' = \frac{1}{2}\left(\mathbf{t} + \frac{\|\mathbf{f}_1 \times \mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0 \times \mathbf{f}_1\|}\mathbf{R}\mathbf{f}_0 + \frac{\|\mathbf{R}\mathbf{f}_0 \times \mathbf{t}\|}{\|\mathbf{R}\mathbf{f}_0 \times \mathbf{f}_1\|}\mathbf{f}_1\right). \tag{4.6}$$

Note that letting $\mathbf{p} = \mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1$, $\mathbf{q} = \mathbf{R}\widehat{\mathbf{f}}_0 \times \mathbf{t}$ and $\mathbf{r} = \widehat{\mathbf{f}}_1 \times \mathbf{t}$ allows us to write (4.4) as

$$\lambda_0 = \frac{\|\mathbf{r}\|}{\|\mathbf{p}\|}, \quad \lambda_1 = \frac{\|\mathbf{q}\|}{\|\mathbf{p}\|}. \tag{4.7}$$

Interestingly, these are in a similar form to the depths given by the classic midpoint method[1]:

$$\lambda_{\text{mid}0} = \frac{\widehat{\mathbf{p}} \cdot \mathbf{r}}{\|\mathbf{p}\|}, \quad \lambda_{\text{mid}1} = \frac{\widehat{\mathbf{p}} \cdot \mathbf{q}}{\|\mathbf{p}\|}. \tag{4.8}$$

[1] We provide the derivation in Appendix C.

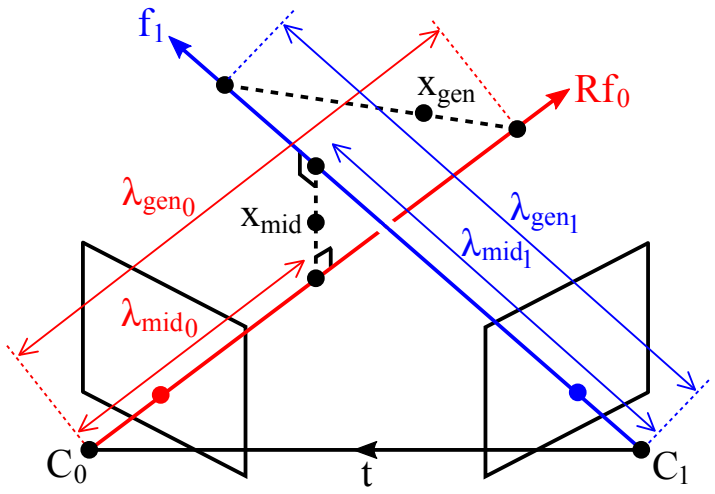


Figure 4.2: The classic midpoint and another example of the generalized weighted midpoint.

The difference is in the numerator; (4.7) has the magnitude of $\mathbf{r}$ and $\mathbf{q}$, whereas (4.8) has their projection onto $\mathbf{p}$. As a result, we always get $\lambda_0 \geq \lambda_{\text{mid}0}$ and $\lambda_1 \geq \lambda_{\text{mid}1}$. In most cases, this means that our midpoint will be located farther than the classic midpoint. Fig. 4.2 depicts one such example. When we estimate that a point is located farther away from the cameras, it usually results in a lower estimate of the parallax angle, as will be shown in Section 4.5.

4.4.3 Cheirality and Test of Adequacy

We say that the cheirality constraint (Hartley and Zisserman, 2003) is violated when a triangulated point has a negative depth. This can happen for many reasons, such as spurious data association or the noise in the image point near the epipole. Normally, it does not pose a serious problem because we can easily check the cheirality for each point and discard the bad ones. For the classic midpoint method, this can be done by checking the signs of the depths given by (4.8). For our midpoint method, however, this is not possible because the depths given by (4.7) are always positive. Fig 4.3a and 4.3b illustrate the difference between the two methods. In our method, the depths alone cannot tell us whether or not the triangulation result is reliable.

Therefore, we use a different method to test the adequacy; we discard the point correspondence if changing the sign of at least one depth to negative leads to a smaller distance between the two points on each ray, *i.e.*,

$$\|\mathbf{t}+\lambda_0\mathbf{R}\widehat{\mathbf{f}}_0-\lambda_1\widehat{\mathbf{f}}_1\|^2 \geq \min\Big(\|\mathbf{t}+\lambda_0\mathbf{R}\widehat{\mathbf{f}}_0+\lambda_1\widehat{\mathbf{f}}_1\|^2,\|\mathbf{t}-\lambda_0\mathbf{R}\widehat{\mathbf{f}}_0-\lambda_1\widehat{\mathbf{f}}_1\|^2,\|\mathbf{t}-\lambda_0\mathbf{R}\widehat{\mathbf{f}}_0+\lambda_1\widehat{\mathbf{f}}_1\|^2\Big) \tag{4.9}$$

For the classic midpoint method, letting $\lambda_0 = |\lambda_{\text{mid}0}|$ and $\lambda_1 = |\lambda_{\text{mid}1}|$ gives effectively the same result as the cheirality check. For example, (4.9) holds in Fig 4.3a because the two points are closest when $\lambda_0 = -|\lambda_{\text{mid}0}|$ and $\lambda_1 = -|\lambda_{\text{mid}1}|$.

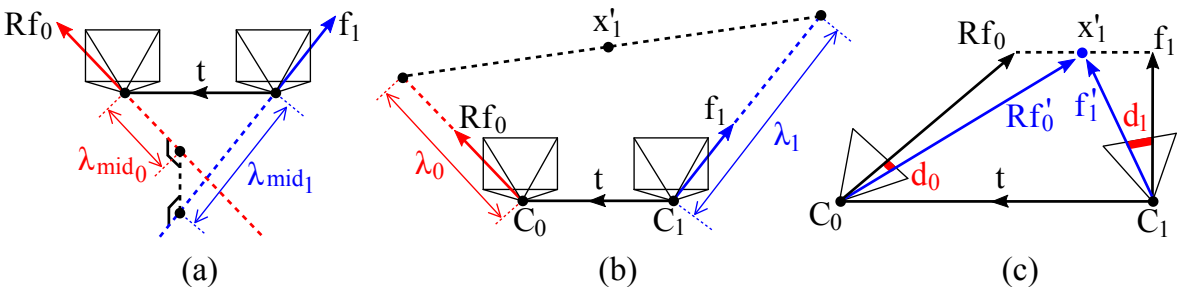


Figure 4.3: **(a)** A scenario where the classic midpoint method gives negative depths. The cheirality check will identify and remove this point. **(b)** In the same situation, our midpoint method will give positive depths. The triangulation result satisfies the cheirality constraint, but it is most likely inaccurate. **(c)** For unweighted midpoint methods, the frame with a smaller depth tends to get a larger reprojection error. In this example, $\lambda_1 < \lambda_0$ and $d_1 > d_0$.

4.4.4 Inverse Depth Weighted Midpoint

The unweighted midpoint given by (4.6) often entails disproportionate reprojection errors in the two images. Fig. 4.3c shows an example. Notice that the ray with a smaller depth tends to yield a larger reprojection error. To compensate this imbalance, we propose to use the inverse depth (Civera et al., 2008*b*) as a weight:

$$\mathbf{x}'_1 = \frac{\lambda_0^{-1}\left(\mathbf{t} + \lambda_0 \mathbf{R}\widehat{\mathbf{f}}_0\right) + \lambda_1^{-1}\left(\lambda_1 \widehat{\mathbf{f}}_1\right)}{\lambda_0^{-1} + \lambda_1^{-1}} \overset{(4.7)}{=} \frac{\|\mathbf{q}\|}{\|\mathbf{q}\| + \|\mathbf{r}\|}\left(\mathbf{t} + \frac{\|\mathbf{r}\|}{\|\mathbf{p}\|}\left(\mathbf{R}\widehat{\mathbf{f}}_0 + \widehat{\mathbf{f}}_1\right)\right). \quad (4.10)$$

4.5 Evaluation Results

We evaluate the following methods: Lee and Civera's L_1, L_2 and L_∞ optimal angular methods ('L_1 ang', 'L_2 ang', 'L_∞ ang') (Lee and Civera, 2019*a*), Hartley and Sturm's L_1 and L_2 optimal methods ('L_1 img', 'L_2 img') and linear methods ('DLT', 'LinLS') (Hartley and Sturm, 1997, Hartley and Zisserman, 2003), Lindstrom's L_2 method with five iterations ('L_2 img (5 it.)') (Lindstrom, 2010), Níster's L_∞ method ('L_∞ img') (Níster, 2001), the classic midpoint method ('Mid') (Beardsley et al., 1994, 1997, Hartley and Sturm, 1997), and our method without and with the weighting ('Mid2', 'wMid2'). The evaluation was performed on synthetic datasets generated as follows: A set of 8×8 point clouds of 5,000 points each are generated with a Gaussian radial distribution $\mathcal{N}(0, (d/4)^2)$ where d is the distance from the world origin. Each point cloud is centered at $[0, 0, d]^\intercal$ for $d = 2^n$ with $n = -1, 0, ..., +6$, and their image projections are perturbed by Gaussian noise $\mathcal{N}(0, \sigma^2)$ for $\sigma = 1, 2..., 8$. The size and the focal length of the images are $1,024^2$ pixels and 512 pixel, respectively. We have four configurations for the camera poses: (1) 'orbital' - two cameras at $[\pm 0.5, 0, 0]^\intercal$ pointing at the point cloud center, (2) 'lateral' - two cameras at $[\pm 0.5, 0, 0]^\intercal$ pointing at $[0, 0, \infty]^\intercal$, (3) 'forward' - two cameras at $[0, 0, \pm 0.5]^\intercal$ pointing at the point cloud center, and (4) 'diagonal' - two cameras at $\pm[\sqrt{3}/6, \sqrt{3}/6, \sqrt{3}/6]^\intercal$ pointing at $[0, 0, \infty]^\intercal$. The poses are slightly

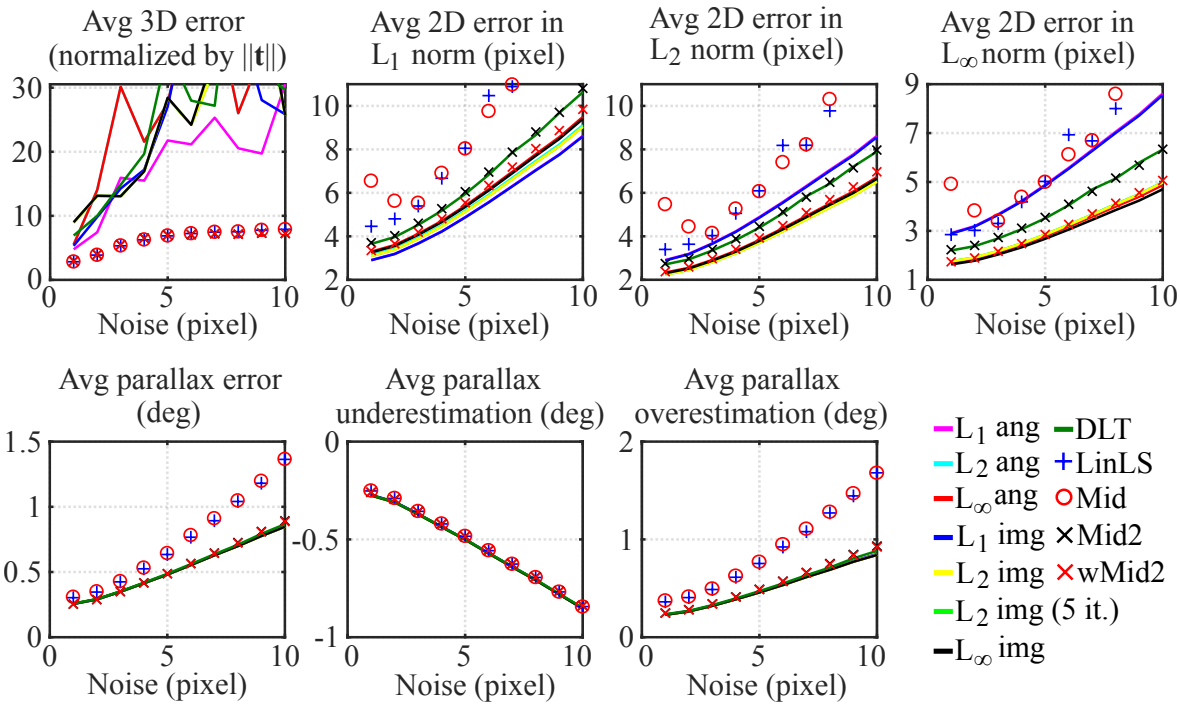


Figure 4.4: Triangulation accuracy over different noise levels in the image measurements. **3D results (top 1st column):** LinLS, Mid, Mid2 and wMid2 perform almost equally and all significantly better than the rest. Among the optimal methods, L_1 ang performs best. **2D results (top 2nd–4th column):** LinLS and Mid perform worst across all norms. Mid2 performs much better than those two, and wMid2 performs consistently better than Mid2. Expectedly, optimal methods perform best in their respective error criterion. However, the differences among them are smaller in L_1 norm than in the other two norms. **Parallax results (bottom row):** LinLS and Mid perform worst, and the rest almost equally. Looking at the under- and overestimation of the parallax separately, we notice that the low accuracy of LinLS and Mid is caused by their bias to overestimate the parallax on average.

perturbed with uniform noise $\mathcal{U}(0, 0.01)$. In total, the datasets provide over a million unique triangulation problems.

We aggregate the results in Fig. 4.4 and 4.5. Our observations agree with (Hartley and Sturm, 1997) in that:

1. Generally, greater noise and lower parallax lead to larger 3D errors. All methods yield almost equally low 3D errors for high-parallax points (> 4 deg).
2. 2D and 3D errors do not always present a strong correlation. For example, LinLS and Mid perform best in 3D, but worst in 2D.

Additionally, we report the following findings:

1. It is difficult to tell which method is the best in terms of 2D accuracy. For example, L_1 methods yield the lowest 2D errors in L_1 norm, but relatively larger errors in L_2 and L_∞ norm. It is not obvious which norm is more important. That said, some methods can still perform consistently better than others; wMid2, L_2 and L_∞ methods consistently outperform LinLS, Mid, Mid2 and DLT in all 2D error criteria.
2. As shown in Fig. 4.6, LinLS and Mid are clearly more biased to overestimate the small

parallax angles (< 4 deg). This explains their relatively low 3D errors at low parallax. Fig. 4.7 illustrates a simplified example of this effect.

3. Our methods (Mid2 and wMid2) achieve the best overall accuracy in 3D + parallax.

The last finding can be solely attributed to the low-parallax points, for which our methods show similar 3D accuracy to that of Mid, but much better parallax accuracy. The latter can be explained by the fact that Mid2 always yields larger depths than Mid (as discussed in Section 4.4.2), which in effect lowers the estimated parallax angle on average (as shown in Fig. 4.6). Since Mid tends to overestimate small parallax angles, this works to our advantage at low parallax.

The first plot in Fig. 4.6 shows that Mid, LinLS and our two methods underestimate small parallax angles slightly less than the rest. It is no coincidence that precisely these four methods achieve the best 3D accuracy; Fig. 4.7 suggests that underestimating a small parallax angle has a severe impact on 3D accuracy. At low parallax, it seems that our methods hit the sweet spot by (1) underestimating less than the optimal methods and DLT (thus achieving lower 3D errors) and (2) overestimating less than Mid and LinLS (thus achieving lower parallax errors).

We suspect that some of the large 2D errors of Mid and LinLS at low parallax are related to their large parallax overestimation (see the second row of Fig. 4.5). Large reprojection errors mean that the rays were corrected by a large amount, and pivoting two almost parallel rays (*i.e.*, rays that correspond to a low-parallax point) will most likely increase the angle between them. This link between the 2D and the parallax error could partially explain why our methods yield smaller 2D errors than Mid and LinLS for low-parallax points.

We also found that the 2D errors of wMid2 resemble those of L_∞ methods (see the bottom row of Fig. 4.5). Note that the L_∞ optimal solutions yield the equal reprojection errors in the two views (Níster, 2001, Lee and Civera, 2019*a*). This suggests that our inverse depth weighting (Section 4.4.4) not only reduces the overall 2D errors, but also balances the reprojection errors in the two images.

Fig. 4.8 compares the speed of each method. We included the test of the adequacy (Section 4.4.3) in both our methods. All methods were implemented in C++ using the Eigen library (*Eigen Library v3*, n.d.), compiled using GCC (*GNU Compiler Collection (GCC)*, n.d.) with -O3 level optimization, and run on a laptop CPU (Intel i7-4810MQ, 2.8 GHz). Although wMid2 is almost two times slower than Mid2 and three times than Mid, it is still at least ten times faster than the state-of-the-art method (Lindstrom, 2010). Notice that even though (4.7) and (4.8) are similar, Mid is still almost twice faster than Mid2. This happens for two reasons: First, we avoid computing the square root in (4.8) by multiplying the numerator and denominator by $\|\mathbf{p}\|$ and get $\lambda_{\text{mid}0} = \frac{\mathbf{p}^\intercal\mathbf{r}}{\mathbf{p}^\intercal\mathbf{p}}$ and $\lambda_{\text{mid}1} = \frac{\mathbf{p}^\intercal\mathbf{q}}{\mathbf{p}^\intercal\mathbf{p}}$. Second, the test of adequacy described in Section 4.4.3 takes longer than the standard cheirality check. Nevertheless, given that the computational cost of triangulation is relatively small compared to other operations (*e.g.*, point matching, pose estimation and structure refinement) (Hartley and Sturm, 1997), our methods provide an excellent trade-off between speed and accuracy.

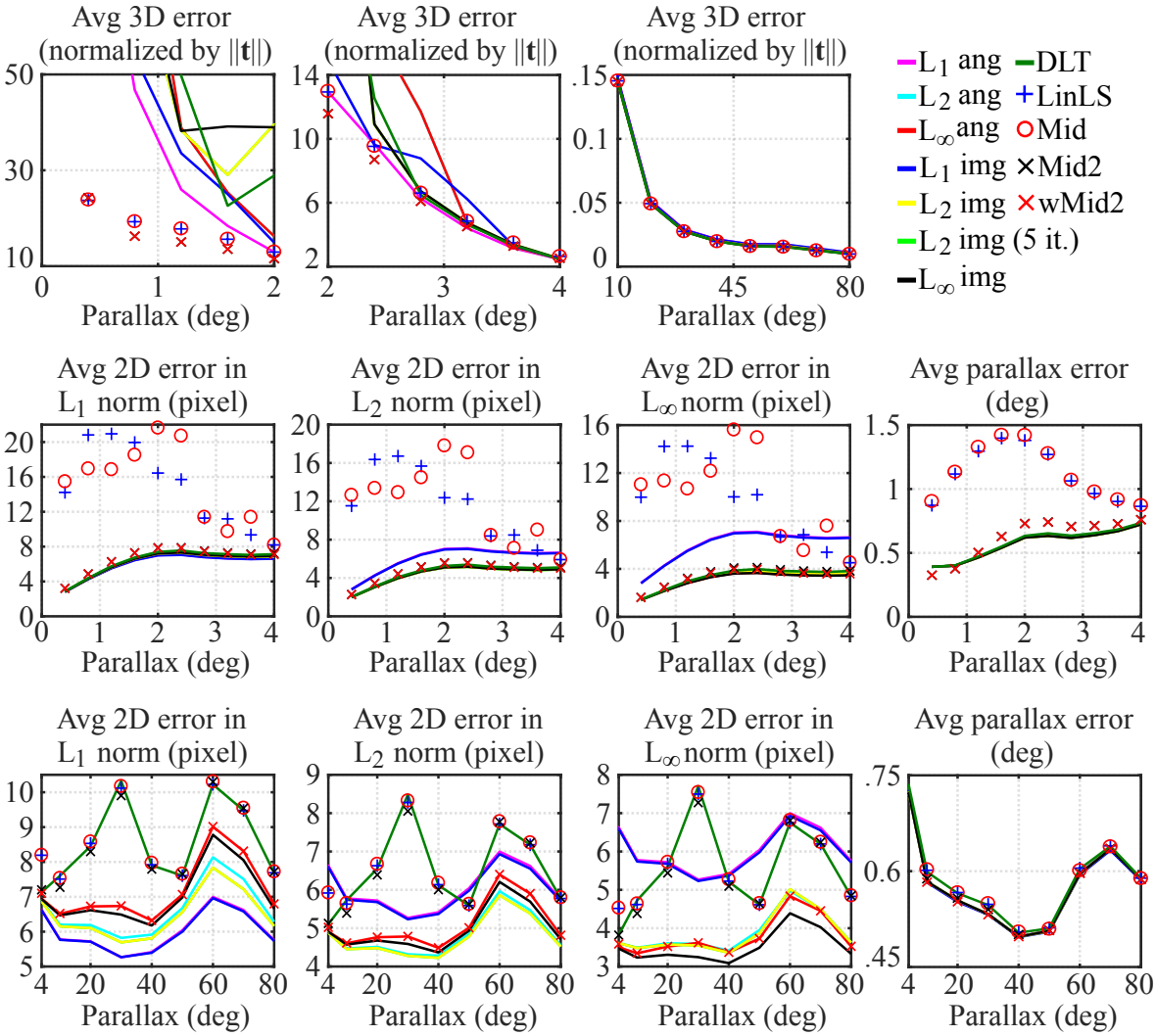


Figure 4.5: Triangulation accuracy over different parallax angles β_{raw} (4.3). **3D results (top row):** For parallax under 2 deg, LinLS, Mid, Mid2 and wMid2 outperform the rest. For parallax between 2–4 deg, L_1 ang and the four aforementioned methods perform best. For parallax over 4 deg, all methods perform almost equally. **2D results for small parallax (mid 1st–3rd column):** LinLS and Mid perform significantly worse than the rest in all norms. Aside from those two, L_1 methods perform much worse than the rest in L_2 and L_∞ norm, yet best in L_1 norm. The remaining methods perform similarly. **2D results for large parallax (bottom 1st–3rd column):** In all norms, LinLS, Mid, Mid2 and DLT perform consistently worse than wMid2, L_2 and L_∞ methods. The latter perform similarly and much better than L_1 methods in L_2 and L_∞ norm, yet worse in L_1 norm. **Parallax results (last column):** For low-parallax points, LinLS and Mid perform worst (see Fig. 4.6 for more details). For high-parallax points, all methods perform equally well.

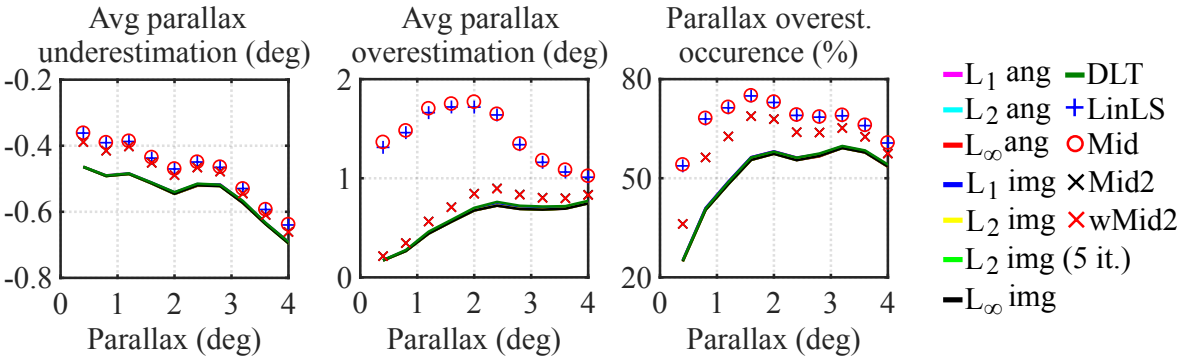


Figure 4.6: Parallax accuracy for low-parallax points. LinLS and Mid are relatively more biased to overestimate the small parallax angles, both in magnitude and frequency. The same goes for Mid2 and wMid2, but to a lesser extent.

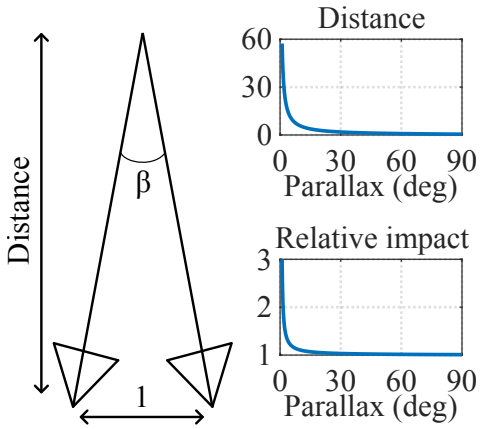


Figure 4.7: **Top:** In this 2D example, the distance is proportional to $\cot(\beta/2)$. **Bottom:** Relative impact is defined as $|f(\beta)/g(\beta)|$ where $f(\beta)$ and $g(\beta)$ are the distance errors when the parallax is under- and overestimated by 0.5 deg, respectively. Underestimating the parallax angle yields a larger distance error than overestimating it by the same degree, and especially more so for smaller parallax.

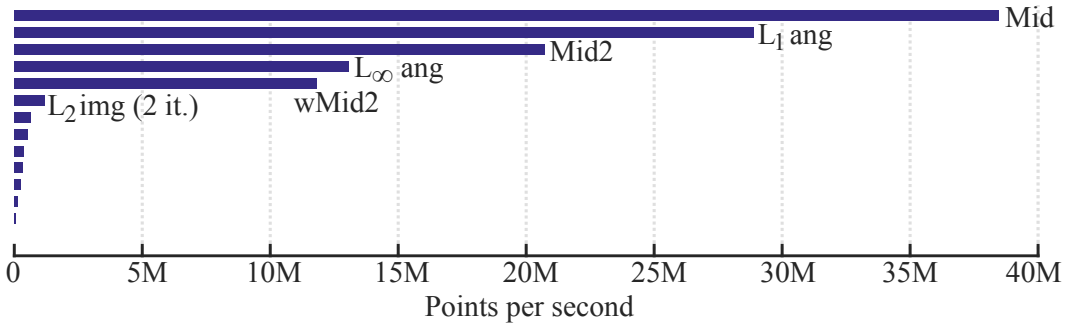


Figure 4.8: Speed of computing a 3D point. In descending order: Mid (38M), L_1 ang (29M), Mid2 (21M), L_∞ ang (13M), wMid2 (12M), L_2 img 2 it. (1.2M), L_2 ang (650K), L_2 img 5 it. (550K), LinLS (350K), DLT (330K), L_∞ img (270K), L_2 img (120K), L_1 img (65K).

4.6 Discussions

4.6.1 On Optimality

A reasonable doubt is that the proposed (weighted) midpoint method may be in fact optimal in some error criterion. If we consider algebraic errors, it is obvious that our midpoint minimizes $\|\mathbf{x}'_1 - \mathbf{g}(\mathbf{R}, \mathbf{t}, \mathbf{f}_0, \mathbf{f}_1)\|$ where $\mathbf{g}(\cdot)$ is the vector-valued function given by the right-hand side of (4.6) or (4.10). However, there are infinitely many other error functions for which the global minimum is $\mathbf{g}(\mathbf{R}, \mathbf{t}, \mathbf{f}_0, \mathbf{f}_1)$, and it is hard to tell which one is geometrically meaningful (at least, we could not find it or disprove its existence).

Another reasonable doubt is that, among the GWM methods, there could be a better one than ours, since we have not proved the optimality of the proposed weighted midpoint. A more fundamental question is then: What accuracy measure should we choose to define the optimality, knowing that there is a discrepancy between different types of errors, *e.g.*, image/angular reprojection errors in different norms, 3D and parallax errors? At least for our method, it seems that the weighting affects mostly the 2D errors, so there might be a certain weighting scheme that guarantees 2D optimality without compromising 3D and parallax accuracy. A more elaborate error analysis and the comparison of different weightings within the GWM framework remain for future work. For more discussion of the optimality in geometric vision, we refer to (Hartley and Kahl, 2007) and Appendix 3 of (Hartley and Zisserman, 2003).

4.6.2 On Practical Implications

In many SfM pipelines, two-view triangulation is used to initialize the 3D map points prior to the bundle adjustment (Snavely et al., 2006, Schönberger and Frahm, 2016, Nousias et al., 2019). Since the points with small parallax angles are associated with large 3D uncertainty, they are usually discarded. This strategy is viable if there are enough correspondences with large parallax angles. However, it is not ideal, as (1) low-parallax points can be useful for camera orientation estimation (Civera et al., 2008*a*, Pirchheim et al., 2013) and (2) if the goal itself is to reconstruct the scene with large depths compared to the baseline. For problems such as reconstruction from small-baseline (or accidental) motions (Delon and Rougé, 2007, Yu and Gallup, 2014), small parallax angles are quite common, so our method could be relevant. Extending our method to multiple views for reconstructing low-parallax scenes would be an interesting future direction.

4.7 Conclusions

Triangulation from two views with known calibration and pose is an age-old problem in computer vision. Existing methods formulate the problem as the minimization of some cost function, most commonly reprojection errors. In this chapter, we asked ourselves if this is really the best approach. To this end, we proposed a novel variant of the classic midpoint method that does not minimize geometric or algebraic errors.

We found that all the existing methods we evaluated perform poorly at low parallax, producing large errors in either 2D, 3D or parallax. On the other hand, our midpoint method achieves very good overall accuracy. We also showed that incorporating the inverse depth weighting can further reduce the 2D errors. Although our method is not theoretically optimal, it provides, with speed and simplicity, a superior balance of 2D, 3D and parallax accuracy in practice.

Chapter 5

Geometric Interpretations of the Normalized Epipolar Error

In this chapter, we provide geometric interpretations of the normalized epipolar error. Most notably, we show that it is directly related to the following quantities: (1) the shortest distance between the two backprojected rays, (2) the dihedral angle between the two bounding epipolar planes, and (3) the L_1-optimal angular reprojection error.

5.1 Introduction

Consider two cameras c_0 and c_1 observing the same 3D point $\mathbf{p}$. If the internal calibration and the relative pose of the two cameras are known, we can backproject the measured point in each image and obtain the rays from each camera, pointing to $\mathbf{p}$. Now, we define the *normalized epipolar error* as follows:

$$\widehat{e} := \left|\widehat{\mathbf{f}}_1 \cdot \left(\widehat{\mathbf{t}} \times \mathbf{R}\widehat{\mathbf{f}}_0\right)\right| = \left|\widehat{\mathbf{t}} \cdot \left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\right)\right|, \tag{5.1}$$

where $\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$ are the backprojected unit rays from c_0 and c_1, respectively, $\mathbf{R}$ is the rotation matrix and $\mathbf{t}$ is the translation vector that together transform a point from the reference frame c_0 to c_1, *i.e.*, $\mathbf{x}_1 = \mathbf{R}\mathbf{x}_0 + \mathbf{t}$ and $\widehat{\mathbf{t}} = \mathbf{t}/\|\mathbf{t}\|$. The second equality in (5.1) follows from the fact that the scalar triple product is invariant to a circular shift. In the literature, the error $\widehat{e}$ is often expressed as follows:

$$\widehat{e} = \left|\widehat{\mathbf{f}}_1^\top \mathbf{E}\widehat{\mathbf{f}}_0\right|, \tag{5.2}$$

where $\mathbf{E} = [\widehat{\mathbf{t}}]_\times \mathbf{R}$ is the *essential matrix* and $[\cdot]_\times$ is the skew-symmetric operator.

If the image measurements, calibration and pose data are all perfectly accurate, this error would be zero because $\mathbf{R}\widehat{\mathbf{f}}_0$, $\widehat{\mathbf{f}}_1$ and $\widehat{\mathbf{t}}$ would be coplanar (see Fig. 5.1). This is called the *epipolar constraint* (Longuet-Higgins, 1981). In practice, the raw data contain inaccuracies, so they do not satisfy this constraint most of the time. For this reason, many existing works in 3D vision try to solve geometric reconstruction problems by minimizing the cost based on

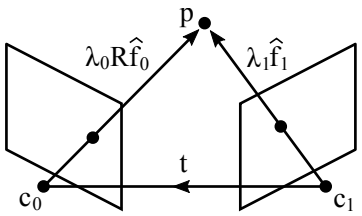


Figure 5.1: A perfectly accurate epipolar geometry. The two backprojected rays intersect at the exact position of the 3D point. The depths of each ray are denoted by λ_0 and λ_1. All vectors are expressed in the coordinate system of c_1.

this error (Briales et al., 2018, Garcia-Salguero et al., 2021, Kneip and Lynen, 2013, Lee and Civera, 2021, Pagani and Stricker, 2011, Rodríguez et al., 2011, Zhao, 2022, Spetsakis and Aloimonos, 1992) or use this error to identify outliers (Zhao, 2022, Lee and Civera, 2020*c*).

In the literature, the normalized epipolar error has mostly been treated as an algebraic quantity that has no geometric meaning (Briales et al., 2018, Garcia-Salguero et al., 2021, Kneip and Lynen, 2013, Rodríguez et al., 2011, Zhao, 2022, Yang et al., 2014). We believe that this misconception stems from the fact that the "standard" epipolar error e is an algebraic quantity (Hartley and Zisserman, 2003, Luong and Faugeras, 1996, Torr and Murray, 1997, Zhang, 1998):

$$e := \left|\mathbf{f}_1 \cdot \left(\widehat{\mathbf{t}} \times \mathbf{R}\mathbf{f}_0\right)\right| = \left|\mathbf{f}_1^\top \mathbf{E}\mathbf{f}_0\right|, \tag{5.3}$$

where $\mathbf{f}_0$ and $\mathbf{f}_1$ are the normalized image coordinates of the point in c_0 and c_1, respectively. Notice that the only difference between (5.1) and (5.3) is the way the rays are normalized: In (5.1), they are normalized by their lengths, whereas in (5.3), they are normalized by the last element in the vector.

In (Pagani and Stricker, 2011), a geometric interpretation was given for the following error:

$$e_p := \frac{\left|\widehat{\mathbf{f}}_1 \cdot \left(\widehat{\mathbf{t}} \times \mathbf{R}\widehat{\mathbf{f}}_0\right)\right|}{\|\widehat{\mathbf{t}} \times \mathbf{R}\widehat{\mathbf{f}}_0\|}, \tag{5.4}$$

which corresponds to the cosine of the angle between $\widehat{\mathbf{f}}_1$ and $\mathbf{n} = \widehat{\mathbf{t}} \times \mathbf{R}\widehat{\mathbf{f}}_0$. This is equal to the perpendicular distance between the point at $\widehat{\mathbf{f}}_1$ and the plane containing $\widehat{\mathbf{t}}$ and $\mathbf{R}\widehat{\mathbf{f}}_0$.

In this chapter, we provide geometrically intuitive interpretations of (5.1) by relating it to the following quantities:

1. The volume of the tetrahedron where $\widehat{\mathbf{f}}_0$, $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{t}}$ form the three edges meeting at one vertex.
2. The shortest distance between the two backprojected rays $\mathbf{l}_0 = \mathbf{t} + s_0\mathbf{R}\widehat{\mathbf{f}}_0$ and $\mathbf{l}_1 = s_1\widehat{\mathbf{f}}_1$.
3. The dihedral angle between the two bounding epipolar planes, *i.e.*, one plane containing $\mathbf{t}$ and $\mathbf{R}\mathbf{f}_0$ and the other containing $\mathbf{t}$ and $\mathbf{f}_1$.
4. The L_1-optimal angular reprojection error.

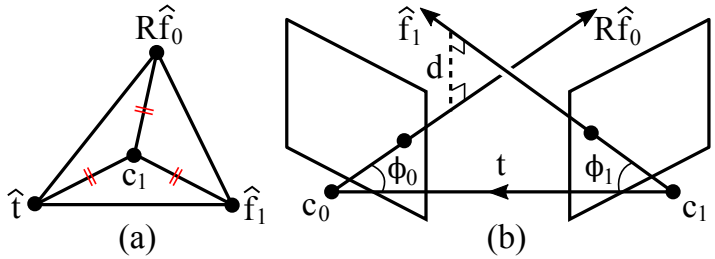


Figure 5.2: **(a)** The volume of this tetrahedron is proportional to the normalized epipolar error (5.1). **(b)** d is the shortest distance between the two backprojected rays corresponding to the same point.

5.2 Geometric Interpretations of (5.1)

5.2.1 Relation to the volume of a tetrahedron

Consider the tetrahedron shown in Fig. 5.2a. One of its vertices is placed at $\mathbf{c}_1$ (*i.e.*, the position of camera c_1, which is the origin in the reference frame of c_1), and the other three at $\widehat{\mathbf{t}}$, $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$. Then, using the well-known formula for the volume of a tetrahedron, its volume is obtained by

$$V = \frac{1}{6}\left|\widehat{\mathbf{t}} \cdot \left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\right)\right| \overset{(5.1)}{=} \frac{\widehat{e}}{6}. \tag{5.5}$$

Therefore,

$$\widehat{e} = 6V. \tag{5.6}$$

The nice thing about this interpretation is that it allows for a simple visualization of the error, as shown in Fig. 5.2a. As the degree of coplanarity increases among the three edges ($\widehat{\mathbf{t}}$, $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$), the common vertex will be "pulled" towards the opposite side, flattening the tetrahedron. When the three edges are coplanar, the tetrahedron becomes completely flat, *i.e.*, $V = 0$, and thus $\widehat{e} = 0$.

5.2.2 Relation to the distance between the two rays

We can also relate the normalized epipolar error $\widehat{e}$ (5.1) to the shortest distance between the two backprojected rays, *i.e.*, d in Fig. 5.2b. To show this, we will first derive the formula for the shortest distance between two skew lines. Consider two skew lines $\mathbf{l}_0 = \mathbf{c}_0 + s_0\mathbf{m}_0$ and $\mathbf{l}_1 = \mathbf{c}_1 + s_1\mathbf{m}_1$. The distance between them is given by the distance between the closest pair of points on each line ($\mathbf{r}_0$ and $\mathbf{r}_1$), and they lie on the common perpendicular to both lines[1]. Now, consider two parallel planes with the normal $\mathbf{n} = \mathbf{m}_0 \times \mathbf{m}_1$: plane Π_0 containing $\mathbf{l}_0$ and plane Π_1 containing $\mathbf{l}_1$, as illustrated in Fig. 5.3a. Notice that $d = \|\mathbf{r}_0 - \mathbf{r}_1\|$ is the same as the distance between the planes, which is the same as $\|\mathbf{c}_1 - \mathbf{a}_0\|$ where $\mathbf{a}_0$ is the projected

[1] We can easily prove this by contradiction. We omit the proof.

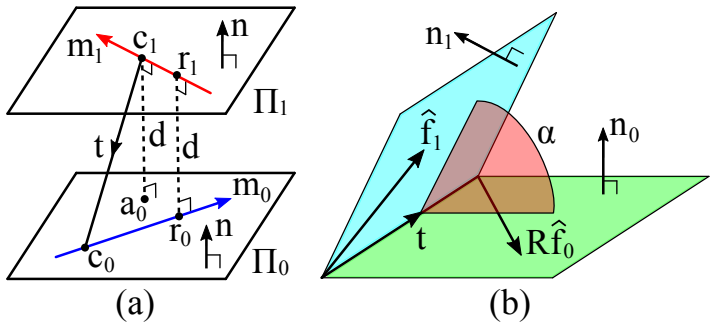


Figure 5.3: **(a)** The distance between the two skew lines is the same as the distance between the two parallel planes as shown here. The normal of the two planes is given by $\mathbf{n} = \mathbf{m}_0 \times \mathbf{m}_1$. **(b)** α is the dihedral angle between the two bounding epipolar planes.

position of $\mathbf{c}_1$ in Π_0. Since $\|\mathbf{c}_1 - \mathbf{a}_0\| = |(\mathbf{c}_0 - \mathbf{c}_1) \cdot \mathbf{n}|/\|\mathbf{n}\|$, we get

$$d = \frac{|(\mathbf{c}_0 - \mathbf{c}_1) \cdot (\mathbf{m}_0 \times \mathbf{m}_1)|}{\|\mathbf{m}_0 \times \mathbf{m}_1\|}. \tag{5.7}$$

This means that d in Fig. 5.2b is given by

$$d = \frac{\left|\mathbf{t} \cdot (\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1)\right|}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\|} \overset{(5.1)}{=} \frac{\|\mathbf{t}\|\widehat{e}}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\|}. \tag{5.8}$$

Let β be the angle between $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$ (also known as the raw parallax angle (Lee and Civera, 2019*c*)), *i.e.*,

$$\beta := \angle(\mathbf{R}\widehat{\mathbf{f}}_0, \widehat{\mathbf{f}}_1) \in [0, \pi/2]. \tag{5.9}$$

Then, (5.8) can be written as

$$\widehat{e} = \frac{\sin(\beta)}{\|\mathbf{t}\|} d. \tag{5.10}$$

Therefore, we can interpret $\widehat{e}$ as the distance between the two backprojected rays, weighted by $\sin(\beta)/\|\mathbf{t}\|$. For relative pose estimation between two views, we can assume $\|\mathbf{t}\| = 1$ without loss of generality, so minimizing the cost based on (5.10) is equivalent to minimizing the cost based on $\sin(\beta)d$. We can interpret $\sin(\beta)$ as the factor that downweights the residual d when the parallax angle is small. Note that $d \leq 1$ and the equality holds if and only if $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$ are both perpendicular to $\widehat{\mathbf{t}}$. If the two rays intersect (at infinity), then $d = 0$ (or $\beta = 0$), and thus $\widehat{e} = 0$.

As a side note, it should be mentioned that d given by (5.8) is the distance between the *lines* rather than the *rays*. Technically speaking, it is the shortest distance between line $\mathbf{l}_0 = \mathbf{t} + s_0\mathbf{R}\widehat{\mathbf{f}}_0$ for $s_0 \in \mathbb{R}$ and line $\mathbf{l}_1 = s_1\widehat{\mathbf{f}}_1$ for $s_1 \in \mathbb{R}$.

5.2.3 Relation to the angle between the two planes

In Fig. 5.2b, consider the following planes: one plane containing $\mathbf{t}$ and $\mathbf{R}\widehat{\mathbf{f}}_0$, and another containing $\mathbf{t}$ and $\widehat{\mathbf{f}}_1$. Let $\mathbf{n}_0$ and $\mathbf{n}_1$ be their normal vectors. These two planes are drawn in

Fig. 5.3b. We can think of them as the two bounding planes between which the epipolar plane is usually found. This is the case for most two-view triangulation methods (*e.g.*, midpoint methods (Lee and Civera, 2019*c*) and optimal methods (Lee and Civera, 2019*a*)). The dihedral angle between these two bounding planes is given by

$$\alpha = \angle(\mathbf{n}_0, \mathbf{n}_1) = \sin^{-1}\left(\left\| \frac{\left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right)}{\left\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right\|} \times \frac{\left(\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right)}{\left\|\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right\|} \right\|\right). \tag{5.11}$$

This can be rearranged to

$$\left\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right\| \left\|\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right\| \sin(\alpha) = \left\|\left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right) \times \left(\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right)\right\|. \tag{5.12}$$

For any 3D vector $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$, the vector quadruple product $(\mathbf{a} \times \mathbf{b}) \times (\mathbf{c} \times \mathbf{d})$ is equal to $((\mathbf{a} \times \mathbf{b}) \cdot \mathbf{d})\,\mathbf{c} - ((\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c})\,\mathbf{d}$. Therefore, the right-hand side of (5.12) can be written as

$$\left\|\left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right) \times \left(\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right)\right\| = \left\| \underbrace{\left(\left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right) \cdot \widehat{\mathbf{t}}\right)}_{0} \widehat{\mathbf{f}}_1 - \left(\left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right) \cdot \widehat{\mathbf{f}}_1\right) \widehat{\mathbf{t}} \right\| \tag{5.13}$$

$$= \left\| \left(\left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right) \cdot \widehat{\mathbf{f}}_1\right) \widehat{\mathbf{t}} \right\| = \left| \left(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right) \cdot \widehat{\mathbf{f}}_1 \right| \overset{(5.1)}{=} \widehat{e}. \tag{5.14}$$

Note that the third equality follows from the fact that $\widehat{\mathbf{t}}$ is a unit vector. Substituting (5.14) into (5.12) leads to

$$\widehat{e} = \left\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\right\| \left\|\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right\| \sin(\alpha) \tag{5.15}$$

$$= \sin(\phi_0) \sin(\phi_1) \sin(\alpha), \tag{5.16}$$

where

$$\phi_0 := \angle(\mathbf{R}\widehat{\mathbf{f}}_0, \widehat{\mathbf{t}}) \in [0, \pi/2], \tag{5.17}$$

$$\phi_1 := \angle(\widehat{\mathbf{f}}_1, \widehat{\mathbf{t}}) \in [0, \pi/2]. \tag{5.18}$$

These two angles are shown in Fig. 5.2b. From (5.16), we can interpret $\widehat{e}$ as the sine of the dihedral angle between the two bounding epipolar planes, weighted by $\sin(\phi_0)\sin(\phi_1)$. Therefore, $\widehat{e}$ would be small if either of ϕ_0, ϕ_1 or α is very small. This makes sense because the epipolar geometry degenerates as ϕ_0 or ϕ_1 approaches zero. Also, when α is small, the two bounding epipolar planes are close to coplanarity, and so do the vector $\widehat{\mathbf{t}}$, $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$.

5.2.4 Relation to the angular reprojection error

The L_1-optimal angular reprojection error (Lee and Civera, 2019*a*) is defined as follows:

$$\theta^*_{\mathrm{L1}} = \min_{\widehat{\mathbf{f}}'_0, \widehat{\mathbf{f}}'_1} \left(\angle(\widehat{\mathbf{f}}'_0, \widehat{\mathbf{f}}_0) + \angle(\widehat{\mathbf{f}}'_1, \widehat{\mathbf{f}}_1)\right) \;\; \text{s.t.} \;\; \widehat{\mathbf{f}}'_1 \cdot \left(\widehat{\mathbf{t}} \times \mathbf{R}\widehat{\mathbf{f}}'_0\right) = 0. \tag{5.19}$$

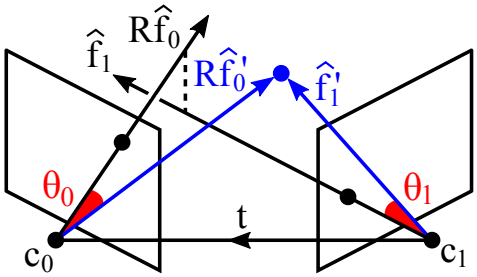


Figure 5.4: The angular reprojection errors (θ_0, θ_1) measure the angular difference between the backprojected rays ($\mathbf{R}\widehat{\mathbf{f}}_0$, $\widehat{\mathbf{f}}_1$) and the corrected rays ($\mathbf{R}\widehat{\mathbf{f}}_0'$, $\widehat{\mathbf{f}}_1'$) that are made to intersect.

In other words, it is the minimum of $\theta_0 + \theta_1$ where θ_0 and θ_1 are the angles by which we correct the backprojected rays to make them intersect. Fig. 5.4 illustrates these angles. In (Lee and Civera, 2019*a*), it was shown that

$$\sin(\theta_{\mathrm{L1}}^*) = \min\left(\frac{\left|\mathbf{R}\widehat{\mathbf{f}}_0 \cdot \left(\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right)\right|}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\|}, \frac{\left|\mathbf{R}\widehat{\mathbf{f}}_0 \cdot \left(\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\right)\right|}{\|\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\|}\right). \tag{5.20}$$

Rearranging this, we get

$$\left|\widehat{\mathbf{f}}_1 \cdot \left(\widehat{\mathbf{t}} \times \mathbf{R}\widehat{\mathbf{f}}_0\right)\right| = \sin(\theta_{\mathrm{L1}}^*) \max\left(\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{t}}\|, \|\widehat{\mathbf{f}}_1 \times \widehat{\mathbf{t}}\|\right). \tag{5.21}$$

Using (5.1), (5.17), (5.18), this can be written as

$$\widehat{e} = \sin\left(\max(\phi_0, \phi_1)\right) \sin(\theta_{\mathrm{L1}}^*). \tag{5.22}$$

Therefore, we can interpret $\widehat{e}$ as the sine of the L_1-optimal angular reprojection error, weighted by $\sin\left(\max(\phi_0, \phi_1)\right)$. It follows that $\widehat{e}$ would be small if either of θ_{L1}^* or $\max(\phi_0, \phi_1)$ is very small. This makes sense because small θ_{L1}^* means that only a little correction is needed for the two backprojected rays to intersect. Also, small $\max(\phi_0, \phi_1)$ means that the vector $\widehat{\mathbf{t}}$, $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{f}}_1$ are all close to parallelism, which brings the epipolar geometry close to degeneracy. What may seem peculiar in (5.22) is the fact that $\max(\phi_0, \phi_1)$ does not reflect the degeneracy when either ϕ_0 or ϕ_1 is zero. However, this is not an issue, because the term $\sin(\theta_{\mathrm{L1}}^*)$ is necessarily zero whenever degeneracy occurs. In the next section, we verify (5.22) using simulation.

5.3 Verification

The contributions of this work are the derivations of (5.6), (5.10), (5.16) and (5.22). No approximation is made in the derivations, so strictly speaking, experiments are redundant as long as the mathematics are correct. Having said that, we understand that some readers may have doubts about the derivations, and also, it is essential to verify the theoretical results whenever possible (as a sanity check). In the case of (5.6), (5.10) and (5.16), however,

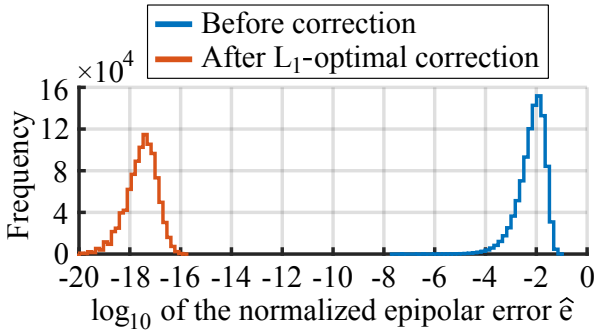


Figure 5.5: Histograms of the normalized epipolar error $\widehat{e}$ before and after the L_1-optimal correction based on angular errors (Lee and Civera, 2019*a*). The corrected rays yield minuscule error, which implies that they now intersect (within the numerical accuracy).

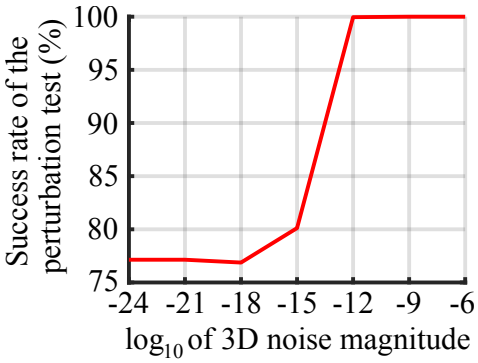


Figure 5.6: The percentage of the simulation runs where the angular error θ^*_{L1} obtained from the L_1-optimal method (Lee and Civera, 2019*a*) is smaller than that of the perturbed point. In our experiment, θ^*_{L1} is always smaller unless the triangulated point is perturbed by extremely small noise ($< 10^{-12}$ unit) beyond the numerical accuracy.

performing experiments is pointless because the only sensible method to compute the volume V, the distance d and the angle α is to use the very same formulas used in the derivations. For this reason, we only focus on the verification of (5.22) in this section.

In order to verify (5.22), we compare the values of $\widehat{e}$ computed using (5.1) and (5.22). This is done in the following steps:

1. In simulation, we create two cameras and one point. The two cameras are placed at position $\mathbf{c}_0$ and $\mathbf{c}_1$ where $\mathbf{c}_0$ is a random 3D vector of length 0.5 unit and $\mathbf{c}_1 = -\mathbf{c}_0$. This ensures that $\|\mathbf{t}\| = \|\mathbf{c}_0 - \mathbf{c}_1\| = 1$ unit. The image size is set to 640×480 pixel and the focal length to 525 pixel. We place the point at $[0, 0, D]^\top$ where D follows the uniform distribution $\mathcal{U}(1, 10)$. Then, we orient the cameras randomly until the point is visible in both views. The image coordinates of the projected point are perturbed by Gaussian noise $\mathcal{N}(0, \sigma^2)$ with $\sigma = 10$ pix.

2. We correct the backprojected rays using the L_1-optimal triangulation method described in (Lee and Civera, 2019*a*) and obtain the angular error θ^*_{L1} using (5.19). To check if this is locally optimal, we perturb the resulting 3D point by small random noise and see if we achieve smaller error. We set the noise magnitude to 10^m unit with $m \in \{-24, -21, \cdots, -6\}$, and for each magnitude, we perturb the point one hundred times

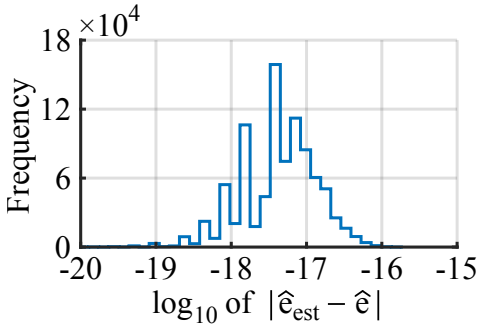


Figure 5.7: Histogram of $|\widehat{e}_{\text{est}} - \widehat{e}|$, where $\widehat{e}$ and $\widehat{e}_{\text{est}}$ are the normalized epipolar errors computed using (5.1) and (5.22) respectively.

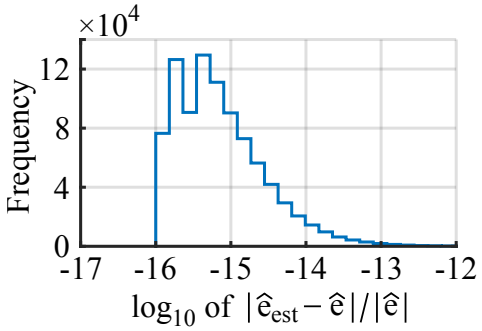


Figure 5.8: Histogram of $|\widehat{e}_{\text{est}} - \widehat{e}|/|\widehat{e}|$, where $\widehat{e}$ and $\widehat{e}_{\text{est}}$ are the normalized epipolar errors computed using (5.1) and (5.22) respectively.

independently.

3. We compute $\widehat{e}$ using (5.22) and compare it to $\widehat{e}$ from (5.1).

We repeat this procedure 10^6 times and aggregate the results. All computations are done in Matlab. Fig. 5.5 shows the histograms of the normalized epipolar error $\widehat{e}$ computed using (5.1) before and after the L_1-optimal ray correction (Lee and Civera, 2019*a*). Comparing the two histograms, we see that the corrected rays do intersect. Fig. 5.6 presents the result of the perturbation test. It shows that the angular error θ^*_{L1} of the corrected rays is (locally) minimum within the numerical accuracy. Plugging θ^*_{L1} into (5.22), we obtain an estimate of $\widehat{e}$, *i.e.*, $\widehat{e}_{\text{est}}$. In Fig. 5.7, we plot the histogram of the absolute difference $|\widehat{e}_{\text{est}} - \widehat{e}|$. Notice that it is as small as the normalized epipolar error of intersecting rays (see Fig. 5.5). Therefore, we can safely conclude that $\widehat{e}_{\text{est}} = \widehat{e}$ within the numerical accuracy. In Fig. 5.8, we provide, for completeness, the histogram of the relative difference $|\widehat{e}_{\text{est}} - \widehat{e}|/|\widehat{e}|$.

5.4 Conclusions

In this chapter, we presented several geometric interpretations of the normalized epipolar error $\widehat{e}$ defined in (5.1). Specifically, we revealed the direct relations between this error and the following quantities:

1. The volume of the tetrahedron where $\widehat{\mathbf{f}}_0$, $\mathbf{R}\widehat{\mathbf{f}}_0$ and $\widehat{\mathbf{t}}$ form the three edges meeting at one vertex (see Fig. 5.2a). The relation is given by (5.6).
2. The shortest distance between the two backprojected rays $\mathbf{l}_0 = \mathbf{t} + s_0\mathbf{R}\widehat{\mathbf{f}}_0$ and $\mathbf{l}_1 = s_1\widehat{\mathbf{f}}_1$ (see Fig. 5.2b). The relation is given by (5.10).
3. The dihedral angle between the two bounding epipolar planes, *i.e.*, one plane containing $\mathbf{t}$ and $\mathbf{R}\mathbf{f}_0$ and the other containing $\mathbf{t}$ and $\mathbf{f}_1$ (see Fig. 5.3a). The relation is given by (5.16).
4. The L_1-optimal angular reprojection error defined in (5.19). The relation is given by (5.22).

Chapter 6

Robust Uncertainty-Aware Multiview Triangulation

In this chapter, we propose a robust and efficient method for multiview triangulation and uncertainty estimation. Our contribution is threefold: First, we propose an outlier rejection scheme using two-view RANSAC with the midpoint method. By prescreening the two-view samples prior to triangulation, we achieve the state-of-the-art efficiency. Second, we compare different local optimization methods for refining the initial solution and the inlier set. With an iterative update of the inlier set, we show that the optimization provides significant improvement in accuracy and robustness. Third, we model the uncertainty of a triangulated point as a function of three factors: the number of cameras, the mean reprojection error and the maximum parallax angle. Learning this model allows us to quickly interpolate the uncertainty at test time. We validate our method through an extensive evaluation.

6.1 Introduction

Multiview triangulation refers to the problem of locating the 3D point given its projections in multiple views of known calibration and pose. It plays a fundamental role in many applications of computer vision, *e.g.*, structure-from-motion (Agarwal et al., 2011, Moulon et al., 2013, Schönberger and Frahm, 2016), visual(-inertial) odometry (Forster, Zhang, Gassner, Werlberger and Scaramuzza, 2017, Klein and Murray, 2007, Leutenegger et al., 2015) and simultaneous localization and mapping (Herrera et al., 2014, Mur-Artal and Tardós, 2017, Qin et al., 2018).

Under the assumption of perfect information (*i.e.*, image measurements, calibration and pose data without noise and outliers), triangulation simply amounts to intersecting the back-projected rays corresponding to the same point. In practice, however, noise and outliers are often inevitable. This makes the triangulation problem nontrivial. From a practical perspective, the following aspects should be taken into account when considering a triangulation method:

1. Is it applicable to multiple views? Some methods are developed specifically for two or three views (*e.g.*, two-view optimal methods (Hartley and Sturm, 1997, Kanatani et al., 2008, Lee and Civera, 2019*a*, Lindstrom, 2010, Oliensis, 2002), two-view midpoint methods (Beardsley et al., 1997, Hartley and Sturm, 1997, Lee and Civera, 2019*c*), three-view optimal methods (Byröd et al., 2007, Hedborg et al., 2014, Kukelova et al., 2013, Stewénius et al., 2005)). For more than two or three views, these methods are not directly applicable unless they are incorporated in, for example, a RANSAC (Fischler and Bolles, 1981) framework.

2. Is it robust to outliers? Many existing multiview triangulation methods are sensitive to outliers, *e.g.*, the linear methods (Hartley and Sturm, 1997), the midpoint-based methods (Ramalingam et al., 2006, Yang et al., 2019), the L_2-optimal methods (Aholt et al., 2012, Kahl et al., 2008, Kahl and Henrion, 2007, Lu and Hartley, 2007) and the L_∞-optimal methods (Agarwal et al., 2008, Dai et al., 2012, Hartley and Schaffalitzky, 2004, Kahl and Hartley, 2008). To deal with outliers, various methods have been proposed, *e.g.*, outlier rejection using the L_∞ norm (Li, 2007, Olsson et al., 2010, Sim and Hartley, 2006), k-th median minimization (Ke and Kanade, 2007) and iteratively reweighted least squares (Aftab and Hartley, 2015). We refer to (Kang et al., 2014) for a comprehensive review of the robust triangulation methods.

3. Is it fast? While the aforementioned methods can handle a moderate number of outliers, they either fail or incur excessive computational cost at high outlier ratios (Schönberger and Frahm, 2016). For this reason, RANSAC is often recommended as a preprocessing step (Li, 2007, Olsson et al., 2010, Sim and Hartley, 2006). In (Schönberger and Frahm, 2016), an efficient method using two-view RANSAC is proposed.

4. Does it estimate the uncertainty? To our knowledge, none of the aforementioned works provide the uncertainty estimate for the triangulated point. Knowing this uncertainty can be useful for point cloud denoising (Wolff et al., 2016), robust localization (Sattler et al., 2011, Svärm et al., 2014) and mapping (Concha and Civera, 2015, Mur-Artal and Tardos, 2015), among others.

In this chapter, we propose a robust and efficient method for uncertainty-aware multiview triangulation. Our contributions are summarized as follows:

1. In Section 6.3.1, we propose an outlier rejection scheme using two-view RANSAC with the midpoint method. By reformulating the midpoint, we screen out the bad samples even before computing the midpoint. This improves the efficiency when the outlier ratio is high.

2. In Section 6.3.2, we revisit three existing local optimization methods, one of which is the Gauss-Newton method. For this method, we present an efficient computation of the Jacobian matrix. We closely evaluate the three methods with an iterative update of the inlier set.

3. In Section 6.3.3, we model the uncertainty of a triangulated point as a function of three factors: the number of (inlying) cameras, the mean reprojection error and the

Algorithm 1: Proposed Multiview Triangulation

Input: $\mathcal{V}$ and $\mathbf{u}_i, \mathbf{K}_i, \mathbf{R}_i, \mathbf{t}_i$ for all $i \in \mathcal{V}$,
$\eta, \delta_{2D}, \delta_{\text{epipolar}}, \delta_{\text{lower}}, \delta_{\text{upper}}, \delta_{\text{update}}, \delta_{\text{pair}}$.
Output: $\mathbf{x}^w_{\text{est}}, \mathcal{I}, \bar{e}_{2D}, \sigma_{3D}$.

/* Initialization */
$\mathbf{x}^w_{\text{est}} \leftarrow \mathbf{0}$; $\mathcal{I} \leftarrow \{\}$; $\bar{e}_{2D} \leftarrow \infty$; $\sigma_{3D} \leftarrow \infty$;
$\widehat{\mathbf{f}}^w_i \leftarrow \mathbf{0}$, $\mathbf{c}^w_i \leftarrow -\mathbf{R}_i^\top \mathbf{t}_i$, $\mathbf{P}_i \leftarrow [\mathbf{R}_i \mid \mathbf{t}_i]$ for all $i \in \mathcal{V}$;
compute $\mathbf{M}_1, \cdots, \mathbf{M}_5$ using (6.2)–(6.6);
compute $\mathbf{b}_{1i}, \cdots, \mathbf{b}_{6i}$ for all $i \in \mathcal{V}$ using (6.24)–(6.29);
compute $\mathbf{a}_{1i}, \cdots, \mathbf{a}_{6i}$ for all $i \in \mathcal{V}$ using (6.30)–(6.32);
compute $\mathbf{A}_i$ for all $i \in \mathcal{V}$ using (6.34);
/* (1) Two-view RANSAC (Sect.6.3.1) */
$m_{\min} \leftarrow n(n-1)/2$; $C_{\min} \leftarrow \infty$;
while $m < m_{\min}$ **do**
  $m \leftarrow m + 1$;
  Pick a random pair of views $j, k \in \mathcal{V}$;
  Perform Alg. 2 for (j, k).
  if $b_{\text{good}} = \text{false}$ **then continue**;
  $\mathbf{x}^w_{\text{est}} \leftarrow \mathbf{x}^w_{\text{mid}}$;
  compute $\mathbf{M}_6$, $\mathbf{M}_7$, $\mathbf{M}_8$ using (6.7)–(6.9);
  compute $\mathbf{e}_{2D}, \mathcal{I}, C$ using (6.10), (6.11) and (6.22);
  if $C \geq C_{\min}$ **then continue**;
  $C_{\min} \leftarrow C$; $\mathbf{x}^{w*}_{\text{est}} \leftarrow \mathbf{x}^w_{\text{est}}$; $\mathcal{I}^* \leftarrow \mathcal{I}$; $\mathbf{M}^*_{6,7,8} \leftarrow \mathbf{M}_{6,7,8}$;
  $\epsilon \leftarrow \max(|\mathcal{I}^*|, 2)/n$; $m_{\min} \leftarrow \dfrac{\log(1-\eta)}{\log(1-\epsilon^2)}$;
if $C_{\min} = \infty$ **then** go to Line 23;
/* (2) Local optimization (Sect.6.3.2) */
$\mathbf{x}^w_{\text{est}} \leftarrow \mathbf{x}^{w*}_{\text{est}}$; $\mathcal{I} \leftarrow \mathcal{I}^*$; $\mathbf{M}_{6,7,8} \leftarrow \mathbf{M}^*_{6,7,8}$;
Perform Alg. 3;
/* (3) Uncertainty estimation (Sect.6.3.3) */
Perform Alg. 4;
return $\mathbf{x}^w_{\text{est}}, \mathcal{I}, \bar{e}_{2D}, \sigma_{3D}$;

maximum parallax angle. We propose to learn this model from extensive simulations, so that at test time, we can estimate the uncertainty by interpolation. The estimated uncertainty can be used to control the 3D accuracy.

The proposed approach is detailed in Alg. 1. See Appendix D for the nomenclature.

6.2 Preliminaries and Notation

We use bold lowercase letters for vectors, bold uppercase letters for matrices, and light letters for scalars. We denote the Hadamard product, division and square root by $\mathbf{A} \circ \mathbf{B}$, $\mathbf{A} \oslash \mathbf{B}$ and $\mathbf{A}^{\circ 1/2}$, respectively. The angle between two vectors $\mathbf{a}$ and $\mathbf{b}$ is denoted by $\angle(\mathbf{a}, \mathbf{b}) \in [0, \pi/2]$. We denote the vectorization of an $n \times m$ matrix by $\text{vec}(\cdot)$ and its inverse by $\text{vec}^{-1}_{n \times m}(\cdot)$.

Consider a 3D point $\mathbf{x}^w = [x^w, y^w, z^w]^\top$ in the world reference frame and a perspective camera c_i observing this point. In the camera reference frame, the 3D point is given by $\mathbf{x}_i = [x_i, y_i, z_i]^\top = \mathbf{R}_i\mathbf{x}^w + \mathbf{t}_i = \mathbf{P}_i\widetilde{\mathbf{x}}^w$, where $\mathbf{R}_i$ and $\mathbf{t}_i$ are the rotation and translation that relate the reference frame c_i to the world, $\mathbf{P}_i = [\mathbf{R}_i \mid \mathbf{t}_i]$ is the extrinsic matrix, and $\widetilde{\mathbf{x}}^w = [x^w, y^w, z^w, 1]^\top$ is the homogeneous coordinates of $\mathbf{x}^w$. In the world frame, the camera position is given by $\mathbf{c}_i^w = -\mathbf{R}_i^\top\mathbf{t}_i$. Let $\widetilde{\mathbf{u}}_i = \left[\mathbf{u}_i^\top, 1\right]^\top = [u_i, v_i, 1]^\top$ be the homogeneous pixel coordinates of the point and $\mathbf{K}_i$ the camera calibration matrix. Then, the normalized image coordinates $\mathbf{f}_i = [x_i/z_i, y_i/z_i, 1]^\top$ are obtained by $\mathbf{f}_i = \mathbf{K}_i^{-1}\widetilde{\mathbf{u}}_i$.

Let $\mathcal{V} = \{1, 2, \cdots, n\}$ be the set of all views in which the point is observed. The aim of multiview triangulation is to find the best estimate of $\mathbf{x}^w$ given that noisy $\mathbf{u}_i$, $\mathbf{R}_i$, $\mathbf{t}_i$ and $\mathbf{K}_i$ are known for all $i \in \mathcal{V}$. Once we have the estimate $\mathbf{x}^w_{\text{est}}$, the 3D error is given by $e_{3D} = \|\mathbf{x}^w_{\text{est}} - \mathbf{x}^w\|$, and the 2D error (aka the reprojection error) is given by

$$\mathbf{e}_{2D} = \left[\|\mathbf{u}_1 - \mathbf{u}_1'\|,\ \|\mathbf{u}_2 - \mathbf{u}_2'\|,\ \cdots,\ \|\mathbf{u}_n - \mathbf{u}_n'\|\right]^\top, \tag{6.1}$$

where $\mathbf{u}_i'$ is the reprojection of $\mathbf{x}^w_{\text{est}}$ in c_i. To compute $\mathbf{e}_{2D}$ compactly, we define the following matrices:

$$\mathbf{M}_1 := [k_{113} - u_1, \cdots, k_{n13} - u_n], \tag{6.2}$$

$$\mathbf{M}_2 := [k_{123} - v_1, \cdots, k_{n23} - v_n], \tag{6.3}$$

$$\begin{aligned}\mathbf{M}_3 := \big[&\left(k_{111}(\mathbf{P}_1)^\top_{\text{row1}} + k_{112}(\mathbf{P}_1)^\top_{\text{row2}}\right), \cdots, \\ &\left(k_{n11}(\mathbf{P}_n)^\top_{\text{row1}} + k_{n12}(\mathbf{P}_n)^\top_{\text{row2}}\right)\big],\end{aligned} \tag{6.4}$$

$$\begin{aligned}\mathbf{M}_4 := \big[&\left(k_{121}(\mathbf{P}_1)^\top_{\text{row1}} + k_{122}(\mathbf{P}_1)^\top_{\text{row2}}\right), \cdots, \\ &\left(k_{n21}(\mathbf{P}_n)^\top_{\text{row1}} + k_{n22}(\mathbf{P}_n)^\top_{\text{row2}}\right)\big],\end{aligned} \tag{6.5}$$

$$\mathbf{M}_5 := \left[(\mathbf{P}_1)^\top_{\text{row3}}, \cdots, (\mathbf{P}_n)^\top_{\text{row3}}\right], \tag{6.6}$$

$$\mathbf{M}_6 := \left(\widetilde{\mathbf{x}}^w_{\text{est}}\right)^\top \mathbf{M}_5, \tag{6.7}$$

$$\mathbf{M}_7 := \mathbf{M}_1 + \left(\left(\widetilde{\mathbf{x}}^w_{\text{est}}\right)^\top \mathbf{M}_3\right) \oslash \mathbf{M}_6, \tag{6.8}$$

$$\mathbf{M}_8 := \mathbf{M}_2 + \left(\left(\widetilde{\mathbf{x}}^w_{\text{est}}\right)^\top \mathbf{M}_4\right) \oslash \mathbf{M}_6. \tag{6.9}$$

where k_{ijk} is the element of $\mathbf{K}_i$ at the j-th row and k-th column. Then, $\mathbf{e}_{2D}$ can be obtained as follows:

$$\mathbf{e}_{2D}^\top = \left(\mathbf{M}_7 \circ \mathbf{M}_7 + \mathbf{M}_8 \circ \mathbf{M}_8\right)^{\circ 1/2}. \tag{6.10}$$

We provide the derivation in Appendix E. Note that $\mathbf{M}_3$, $\mathbf{M}_4$ and $\mathbf{M}_5$ are independent of the point, and thus can be precomputed for efficiency.

We define the positive z-axis of the camera as the forward direction. This means that if the i-th element of $\mathbf{M}_6$ is negative, the point is behind the camera c_i, violating the cheirality (Hartley and Zisserman, 2003). Hence, given the estimated point $\mathbf{x}^w_{\text{est}}$, the corresponding set

of inliers is obtained by

$$\mathcal{I} = \{i \in \mathcal{V} \mid (\mathbf{e}_{2D})_i < \delta_{2D} \wedge (\mathbf{M}_6)_i > 0\}, \tag{6.11}$$

where $(\cdot)_i$ indicates the i-th element, and δ_{2D} is the inlier threshold. We denote the number of elements of $\mathcal{I}$ by $|\mathcal{I}|$. We define the maximum parallax angle of $\mathcal{I}$ as follows:

$$\beta_{\text{max}} := \max \left\{ \angle \left(\mathbf{f}_j^w, \mathbf{f}_k^w \right) \mid j, k \in \mathcal{I} \right\} \tag{6.12}$$

$$= \cos^{-1} \left(\min \left\{ \left| \widehat{\mathbf{f}}_j^w \cdot \widehat{\mathbf{f}}_k^w \right| \mid j, k \in \mathcal{I} \right\} \right), \tag{6.13}$$

where $\widehat{\mathbf{f}}_j^w$ and $\widehat{\mathbf{f}}_k^w$ are the two corresponding unit rays from camera j and k expressed in the world frame, *i.e.*,

$$\widehat{\mathbf{f}}_j^w = \mathbf{R}_j^\top \widehat{\mathbf{f}}_j, \ \widehat{\mathbf{f}}_k^w = \mathbf{R}_k^\top \widehat{\mathbf{f}}_k \ \text{ with } \ \mathbf{f}_j = \mathbf{K}_j^{-1} \widetilde{\mathbf{u}}_j, \ \mathbf{f}_k = \mathbf{K}_k^{-1} \widetilde{\mathbf{u}}_k. \tag{6.14}$$

6.3 Method

6.3.1 Fast Two-View RANSAC for Outlier Rejection

To obtain the initial solution and inlier set, we perform two-view RANSAC as in (Schönberger and Frahm, 2016). Our method has two notable differences to (Schönberger and Frahm, 2016): First, instead of the DLT method (Hartley and Sturm, 1997), we use the midpoint method (Beardsley et al., 1997, Hartley and Sturm, 1997), which is faster and as accurate unless the parallax is very low (Hartley and Sturm, 1997, Lee and Civera, 2019*c*). Second, we prescreen the samples before performing the two-view triangulation. Our method consists of the following steps:

1. Check the normalized epipolar error.

Let camera j and k be the two-view sample. The normalized epipolar error (Lee and Civera, 2020*a*) of the sample is defined as

$$e_{jk} := \left| \widehat{\mathbf{t}}_{jk}^w \cdot \left(\widehat{\mathbf{f}}_j^w \times \widehat{\mathbf{f}}_k^w \right) \right|, \tag{6.15}$$

where $\widehat{\mathbf{t}}_{jk}^w$ is the unit vector of $\mathbf{t}_{jk}^w = \mathbf{c}_j^w - \mathbf{c}_k^w$ and $\widehat{\mathbf{f}}_j^w$, $\widehat{\mathbf{f}}_k^w$ are the corresponding rays in the world frame given by (6.14). If $\widehat{\mathbf{f}}_j^w$ and $\widehat{\mathbf{f}}_k^w$ are both inliers, then e_{jk} must be small (Longuet-Higgins, 1981).

2. Check the parallax angle.

The raw parallax (Lee and Civera, 2019*c*) defined as $\beta_{jk} := \angle(\widehat{\mathbf{f}}_j^w, \widehat{\mathbf{f}}_k^w)$ is a rough estimate of the parallax angle if $\widehat{\mathbf{f}}_j^w$ and $\widehat{\mathbf{f}}_k^w$ are both inliers. If β_{jk} is too small, the triangulation is inaccurate (Lee and Civera, 2019*c*). If it is too large, the sample most likely contains an outlier, because such a point is rarely matched in practice due to a severe viewpoint change.

Algorithm 2: Midpoint Method with Early Termination

Input: $\mathbf{u}_j, \mathbf{u}_k, \mathbf{K}_j, \mathbf{K}_k, \mathbf{R}_j, \mathbf{R}_k, \mathbf{P}_j, \mathbf{P}_k, \mathbf{c}_j^w, \mathbf{c}_k^w, \widehat{\mathbf{f}}_j^w, \widehat{\mathbf{f}}_k^w, \delta_{\text{epipolar}}, \delta_{\text{lower}}, \delta_{\text{upper}}, \delta_{2D}$.
Output: $\mathbf{x}_{\text{mid}}^w, \widehat{\mathbf{f}}_j^w, \widehat{\mathbf{f}}_k^w, b_{\text{good}}$.

$b_{\text{good}} \leftarrow$ false; $\mathbf{x}_{\text{mid}}^w \leftarrow \mathbf{0}$;
if $\widehat{\mathbf{f}}_j^w = \mathbf{0}$ **then** compute $\widehat{\mathbf{f}}_j^w$ using (6.14);
if $\widehat{\mathbf{f}}_k^w = \mathbf{0}$ **then** compute $\widehat{\mathbf{f}}_k^w$ using (6.14);
$\mathbf{t}_{jk}^w \leftarrow \mathbf{c}_j^w - \mathbf{c}_k^w$; $\widehat{\mathbf{t}}_{jk}^w \leftarrow \mathbf{t}_{jk}^w / \left\|\mathbf{t}_{jk}^w\right\|$;
/* Check the normalized epipolar error. */
compute e_{jk} using (6.15);
if $e_{jk} > \delta_{\text{epipolar}}$ **then** go to Line 19;
/* Check the parallax angle. */
compute p_{jk} using (6.16);
if $p_{jk} < \delta_{\text{lower}} \vee p_{jk} > \delta_{\text{upper}}$ **then** go to Line 19;
/* Check additional degeneracy. */
compute q_{jk} and r_{jk} using (6.17);
if $|q_{jk}| > \delta_{\text{upper}} \vee |r_{jk}| > \delta_{\text{upper}}$ **then** go to Line 19;
/* Check the signs of anchor depths. */
compute μ_j and μ_k using (6.20);
if $\mu_j < 0 \vee \mu_k < 0$ **then** go to Line 19;
/* Compute the midpoint. */
compute s_{jk}, λ_j, λ_k and $\mathbf{x}_{\text{mid}}^w$ using (6.19), (6.18), (6.21);
/* Check the cheirality. */
$\mathbf{x}_j \leftarrow \mathbf{P}_j \begin{bmatrix} \mathbf{x}_{\text{mid}}^w \\ 1 \end{bmatrix}$; $\mathbf{x}_k \leftarrow \mathbf{P}_k \begin{bmatrix} \mathbf{x}_{\text{mid}}^w \\ 1 \end{bmatrix}$;
if $(\mathbf{x}_j)_3 < 0 \vee (\mathbf{x}_k)_3 < 0$ **then** go to Line 19;
/* Check the reprojection error. */
$\mathbf{e}_j \leftarrow \begin{bmatrix} \mathbf{u}_j \\ 1 \end{bmatrix} - \frac{\mathbf{K}_j \mathbf{x}_j}{(\mathbf{x}_j)_3}$; $\mathbf{e}_k \leftarrow \begin{bmatrix} \mathbf{u}_k \\ 1 \end{bmatrix} - \frac{\mathbf{K}_k \mathbf{x}_k}{(\mathbf{x}_k)_3}$;
if $\mathbf{e}_j^\top \mathbf{e}_j > \delta_{2D}^2 \vee \mathbf{e}_k^\top \mathbf{e}_k > \delta_{2D}^2$ **then** go to Line 19;
$b_{\text{good}} \leftarrow$ true;
return $\mathbf{x}_{\text{mid}}^w, \widehat{\mathbf{f}}_i^w, \widehat{\mathbf{f}}_j^w, b_{\text{good}}$;

We check β_{jk} from its cosine:

$$p_{jk} := \widehat{\mathbf{f}}_j^w \cdot \widehat{\mathbf{f}}_k^w \tag{6.16}$$

3. Check additional degeneracy.

Likewise, if $\angle(\widehat{\mathbf{f}}_j^w, \widehat{\mathbf{t}}_{jk}^w)$ or $\angle(\widehat{\mathbf{f}}_k^w, \widehat{\mathbf{t}}_{jk}^w)$ is too small, the epipolar geometry degenerates (see Fig. 6.1). To avoid degeneracy, we also check these angles from their cosines:

$$q_{jk} := \widehat{\mathbf{f}}_j^w \cdot \widehat{\mathbf{t}}_{jk}^w, \quad r_{jk} := \widehat{\mathbf{f}}_k^w \cdot \widehat{\mathbf{t}}_{jk}^w. \tag{6.17}$$

4. Check the depths of the midpoint anchors.

Let λ_j and λ_k be the depths of the midpoint anchors (see Fig. 6.1). Using Lemma 7 in

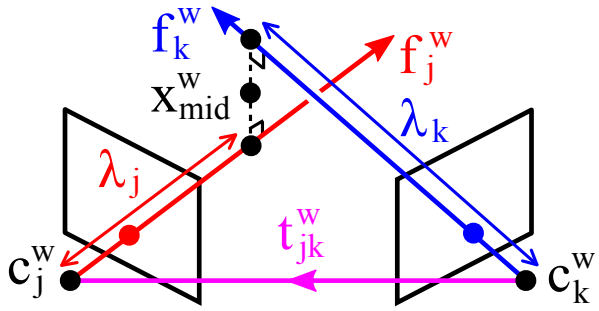


Figure 6.1: The midpoint of the two corresponding rays.

Appendix C, we can derive that

$$\lambda_j = s_{jk}\mu_j, \quad \lambda_k = s_{jk}\mu_k. \tag{6.18}$$

where

$$s_{jk} := \left\|\mathbf{t}_{jk}^w\right\| / \left(1 - p_{jk}^2\right), \tag{6.19}$$

$$\mu_j := p_{jk}r_{jk} - q_{jk}, \quad \mu_k := -p_{jk}q_{jk} + r_{jk}. \tag{6.20}$$

Since $s_{jk} \geq 0$, we check the signs of μ_j and μ_k to ensure that λ_j and λ_k are both positive.

5. Evaluate the midpoint w.r.t. the two views.

Only when the two-view sample passes all of the aforementioned checks, we compute the midpoint:

$$\mathbf{x}_{\text{mid}}^w = 0.5\left(\mathbf{c}_j^w + \lambda_j\widehat{\mathbf{f}}_j^w + \mathbf{c}_k^w + \lambda_k\widehat{\mathbf{f}}_k^w\right). \tag{6.21}$$

Then, we check the cheirality and reprojection errors in the two views. The entire procedure is detailed in Alg. 2. Each midpoint from the two-view samples becomes a hypothesis for $\mathbf{x}^w$ and is scored based on its reprojection error and cheirality. Specifically, we use the approach of (Torr and Zisserman, 1998) and find the hypothesis that minimizes the following cost:

$$C = \sum_{i=1}^{n} r_i^2 \quad \text{with} \quad r_i = \begin{cases} (\mathbf{e}_{2D})_i & \text{if } i \in \mathcal{I}, \\ \delta_{2D} & \text{otherwise.} \end{cases} \tag{6.22}$$

Once we find a hypothesis with smaller cost, we update the inlier ratio based on its support set $\mathcal{I}$ and recompute the required number of samples to be drawn (adaptive stopping criterion (Torr et al., 1998, Raguram et al., 2013, Schönberger and Frahm, 2016)). This is done in Line 18 of Alg. 1.

6.3.2 Iterative Local Optimization

Once we have the initial triangulation result and the inlier set from the two-view RANSAC, we perform local optimization for refinement. Our approach is similar to (Chum et al., 2003), except that we perform the optimization only at the end of RANSAC. We compare

three optimization methods:

- `DLT` and `LinLS` (Hartley and Sturm, 1997): These two linear methods minimize the algebraic errors in closed form. For the formal descriptions, we refer to (Hartley and Sturm, 1997). They were originally developed for two-view triangulation, but they can be easily extended to multiple views. We construct the linear system with only the inliers, solve it and update the inlier set. This is repeated until the inlier set converges.

- `GN`: This nonlinear method minimizes the geometric errors using the Gauss-Newton algorithm. After each update of the solution, we update the inlier set.

The `GN` method requires the computation of the Jacobian matrix $\mathbf{J}$ in each iteration. In the following, we present an efficient method for computing $\mathbf{J}$. Recall that we are minimizing $\mathbf{e}_{2D}^\top \mathbf{e}_{2D}$, which we know from (6.1) is equal to $\mathbf{r}^\top \mathbf{r}$, where

$$\mathbf{r} = \left[(u_{\text{error}})_1, (v_{\text{error}})_1, \cdots, (u_{\text{error}})_n, (v_{\text{error}})_n\right]^\top.$$

This means that we can obtain $\mathbf{J}$ by stacking

$$\mathbf{J}_i = \begin{bmatrix} \frac{\partial (u_{\text{error}})_i}{\partial x_{\text{est}}^w} & \frac{\partial (u_{\text{error}})_i}{\partial y_{\text{est}}^w} & \frac{\partial (u_{\text{error}})_i}{\partial z_{\text{est}}^w} \\ \frac{\partial (v_{\text{error}})_i}{\partial x_{\text{est}}^w} & \frac{\partial (v_{\text{error}})_i}{\partial y_{\text{est}}^w} & \frac{\partial (v_{\text{error}})_i}{\partial z_{\text{est}}^w} \end{bmatrix} \tag{6.23}$$

for all $i \in \mathcal{I}$. We now define the following vectors:

$$\mathbf{b}_{1i} := r_{i11}\left[0, r_{i32}, r_{i33}, t_{i3}\right]^\top - r_{i31}\left[0, r_{i12}, r_{i13}, t_{i1}\right]^\top, \tag{6.24}$$
$$\mathbf{b}_{2i} := r_{i21}\left[0, r_{i32}, r_{i33}, t_{i3}\right]^\top - r_{i31}\left[0, r_{i22}, r_{i23}, t_{i2}\right]^\top, \tag{6.25}$$
$$\mathbf{b}_{3i} := r_{i12}\left[r_{i31}, 0, r_{i33}, t_{i3}\right]^\top - r_{i32}\left[r_{i11}, 0, r_{i13}, t_{i1}\right]^\top, \tag{6.26}$$
$$\mathbf{b}_{4i} := r_{i22}\left[r_{i31}, 0, r_{i33}, t_{i3}\right]^\top - r_{i32}\left[r_{i21}, 0, r_{i23}, t_{i2}\right]^\top, \tag{6.27}$$
$$\mathbf{b}_{5i} := r_{i13}\left[r_{i31}, r_{i32}, 0, t_{i3}\right]^\top - r_{i33}\left[r_{i11}, r_{i12}, 0, t_{i1}\right]^\top, \tag{6.28}$$
$$\mathbf{b}_{6i} := r_{i23}\left[r_{i31}, r_{i32}, 0, t_{i3}\right]^\top - r_{i33}\left[r_{i21}, r_{i22}, 0, t_{i2}\right]^\top, \tag{6.29}$$
$$\mathbf{a}_{1i} := k_{i11}\mathbf{b}_{1i} + k_{i12}\mathbf{b}_{2i}, \quad \mathbf{a}_{2i} := k_{i21}\mathbf{b}_{1i} + k_{i22}\mathbf{b}_{2i}, \tag{6.30}$$
$$\mathbf{a}_{3i} := k_{i11}\mathbf{b}_{3i} + k_{i12}\mathbf{b}_{4i}, \quad \mathbf{a}_{4i} := k_{i21}\mathbf{b}_{3i} + k_{i22}\mathbf{b}_{4i}, \tag{6.31}$$
$$\mathbf{a}_{5i} := k_{i11}\mathbf{b}_{5i} + k_{i12}\mathbf{b}_{6i}, \quad \mathbf{a}_{6i} := k_{i21}\mathbf{b}_{5i} + k_{i22}\mathbf{b}_{6i}, \tag{6.32}$$

where r_{ijk} and k_{ijk} respectively indicate the elements of $\mathbf{R}_i$ and $\mathbf{K}_i$ at the j-th row and k-th column, and t_{ij} indicate the j-th element of $\mathbf{t}_i$. Then, we can rewrite (6.23) as

$$\mathbf{J}_i = \left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}_{\text{est}}^w\right)^{-2} \text{vec}_{2\times 3}^{-1}\left(\mathbf{A}_i^\top \widetilde{\mathbf{x}}_{\text{est}}^w\right) \tag{6.33}$$
$$\text{with} \quad \mathbf{A}_i = \left[\mathbf{a}_{1i}\ \mathbf{a}_{2i}\ \mathbf{a}_{3i}\ \mathbf{a}_{4i}\ \mathbf{a}_{5i}\ \mathbf{a}_{6i}\right]. \tag{6.34}$$

We provide the derivation in Appendix F. Since $\mathbf{P}_i$ and $\mathbf{A}_i$ can be precomputed independently of the point, the Jacobian can computed more efficiently. Alg. 3 summarizes the GN method.

6.3.3 Practical Uncertainty Estimation

We model the uncertainty of the triangulated point $\mathbf{x}_{\text{est}}^w$ as a function of three factors: the number of inlying views ($|\mathcal{I}|$), the mean reprojection error in those views ($\overline{e}_{\text{2D}}$) and the maximum parallax angle (β_{max}) defined by (6.12).

To this end, we run a large number of simulations in various settings and aggregate the 3D errors for each different range of factors (see Fig. 6.2). We then store these data on a 3D regular grid G that maps $(|\mathcal{I}|, \overline{e}_{\text{2D}}, \beta_{\text{max}})$ to the uncertainty σ_{3D}. At test time, we estimate the uncertainty by performing trilinear interpolation on this grid.

We point out two things in our implementation: First, to reduce the small sample bias in G, we perform monotone smoothing that enforces σ_{3D} to increase with $\overline{e}_{\text{2D}}$ and decreases with $|\mathcal{I}|$ and β_{max}. The smoothing method is described and demonstrated in Appendix G. Second, we limit the number of pairs we evaluate for computing β_{max} in (6.13). This curbs the computational cost when $\mathcal{I}$ is very large. Alg. 4 summarizes the procedure.

6.4 Results

6.4.1 Uncertainty Estimation

To find out how the different factors impact the 3D accuracy of triangulation, we run a large number of simulations in various settings configured by the following parameters:

- n: number of cameras observing the point.
- d: distance between the ground-truth point and the origin.
- σ: std. dev. of Gaussian noise in the image coordinates.
- n_{run}: number of independent simulation runs for each configuration (n, d, σ).

The parameter values are specified in Tab. 6.1. The simulations are generated as follows: We create n cameras, $n-2$ of those randomly located inside a sphere of unit diameter at the origin. We place one of the two remaining cameras at a random point on the sphere's surface and the other at its antipode. This ensures that the geometric span of the cameras is equal to one unit. The size and the focal length of the images are set to 640×480 and 525 pixel, respectively, the same as those of (Sturm et al., 2012). Next, we create a point at $[0, 0, d]^\top$ and orient the cameras randomly until the point is visible in all images. Then, we add the image noise of $\mathcal{N}(0, \sigma^2)$ to perturb the image coordinates.

Algorithm 3: GN with an iterative update of the inlier set

Input: $\mathbf{x}^w_{\text{est}}$, $\mathcal{V}$, $\mathcal{I}$, $\mathbf{M}_{1,2,\cdots,8}$, δ_{2D}, δ_{update}, $\mathbf{P}_i$ and $\mathbf{A}_i$ for all $i \in \mathcal{V}$.
Output: $\mathbf{x}^w_{\text{est}}$, $\mathcal{I}$, $\overline{e}_{2D}$.

$n_{\text{it}} \leftarrow 0$; $\overline{e}_{2D} \leftarrow 0$;
while $n_{\text{it}} < 10$ **do**
  $n_{\text{it}} \leftarrow n_{\text{it}} + 1$; $(\overline{e}_{2D})_{\text{prev}} \leftarrow \overline{e}_{2D}$; $\mathcal{I}_{\text{prev}} \leftarrow \mathcal{I}$;
  /* Obtain the residuals and Jacobian. */
  obtain $\mathbf{r}$ by stacking $\begin{bmatrix}(\mathbf{M}_7)_i \\ (\mathbf{M}_8)_i\end{bmatrix}$ for all $i \in \mathcal{I}$;
  compute $\mathbf{J}_i$ using (6.33) for all $i \in \mathcal{I}$;
  obtain $\mathbf{J}$ by stacking $\mathbf{J}_i$ for all $i \in \mathcal{I}$;
  /* Update the solution. */
  $\mathbf{x}^w_{\text{est}} \leftarrow \mathbf{x}^w_{\text{est}} - \mathbf{J}^+\mathbf{r}$;
  /* Update the inlier set. */
  compute $\mathbf{M}_6$, $\mathbf{M}_7$, $\mathbf{M}_8$ using (6.7)–(6.9);
  compute $\mathbf{e}_{2D}$ and $\mathcal{I}$ using (6.10) and (6.11);
  /* Check the convergence. */
  $\overline{e}_{2D} \leftarrow \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}(\mathbf{e}_{2D})_i$;
  if $\mathcal{I} = \mathcal{I}_{\text{prev}} \wedge |\overline{e}_{2D} - (\overline{e}_{2D})_{\text{prev}}| < \delta_{\text{update}}$ **then**
    break;
return $\mathbf{x}^w_{\text{est}}$, $\mathcal{I}$, $\overline{e}_{2D}$;

For triangulation, we initialize the point using the DLT method and refine it using the GN method. In this experiment, we assume that all points are always inliers, so we do not update the inlier set during the optimization. Fig. 6.2 shows the 3D error distribution with respect to different numbers of cameras (n), mean reprojection errors ($\overline{e}_{2D}$) and maximum parallax angle (β_{max}). In general, we observe that the 3D accuracy improves with more cameras, smaller $\overline{e}_{2D}$ and larger β_{max}. However, this effect diminishes past a certain level. For example, the difference between the 30 and 50 cameras is much smaller than the difference between the 2 and 3. Also, when β_{max} is sufficiently large, the 3D accuracy is less sensitive to the change in n, $\overline{e}_{2D}$ and β_{max}.

Fig. 6.2 clearly indicates that we must take into account all these three factors when estimating the 3D uncertainty of a triangulated point. Marginalizing any one of them would reduce the accuracy. This observation agrees with our intuition, as each factor conveys important independent information about the given triangulation problem.

6.4.2 Triangulation Performance

We evaluate the performance of our method on synthetic data. The simulation is configured in a similar way as in the previous section. The difference is that we set $n = 100$, $d = \{3, 5, 7, 9\}$, $\sigma = 3$ pixel, $n_{\text{run}} = 100$ thousand, and we perturb some of the measurements by more than 10 pixel, turning them into outliers. The outlier ratio is set to 10, 30, 50, 70 and

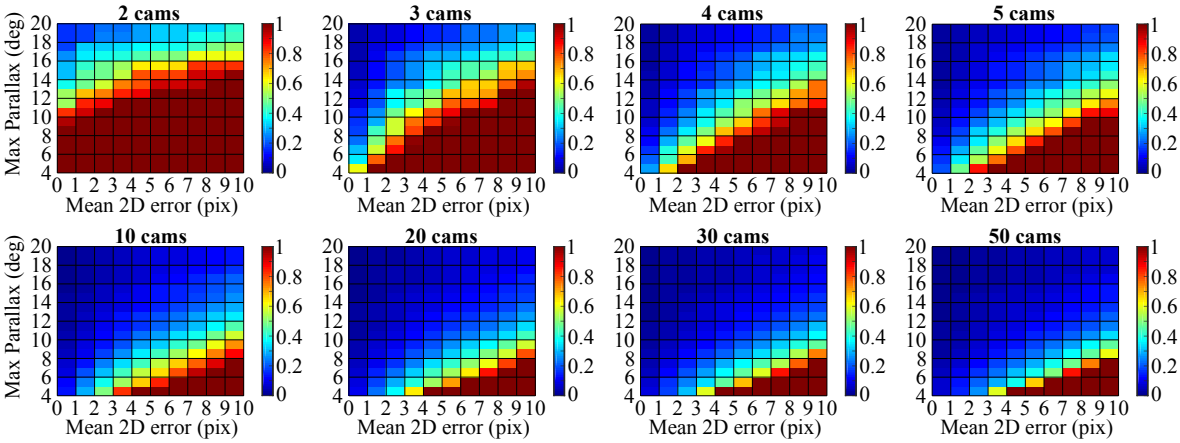


Figure 6.2: RMS of the 3D errors for different numbers of cameras, maximum parallax angles and mean 2D errors. Here, we only present the smoothed results up to 10 pixel error for the selected numbers of cameras. Any cell value above one unit is considered highly inaccurate, and thus truncated. One unit corresponds to the geometric span of the cameras.

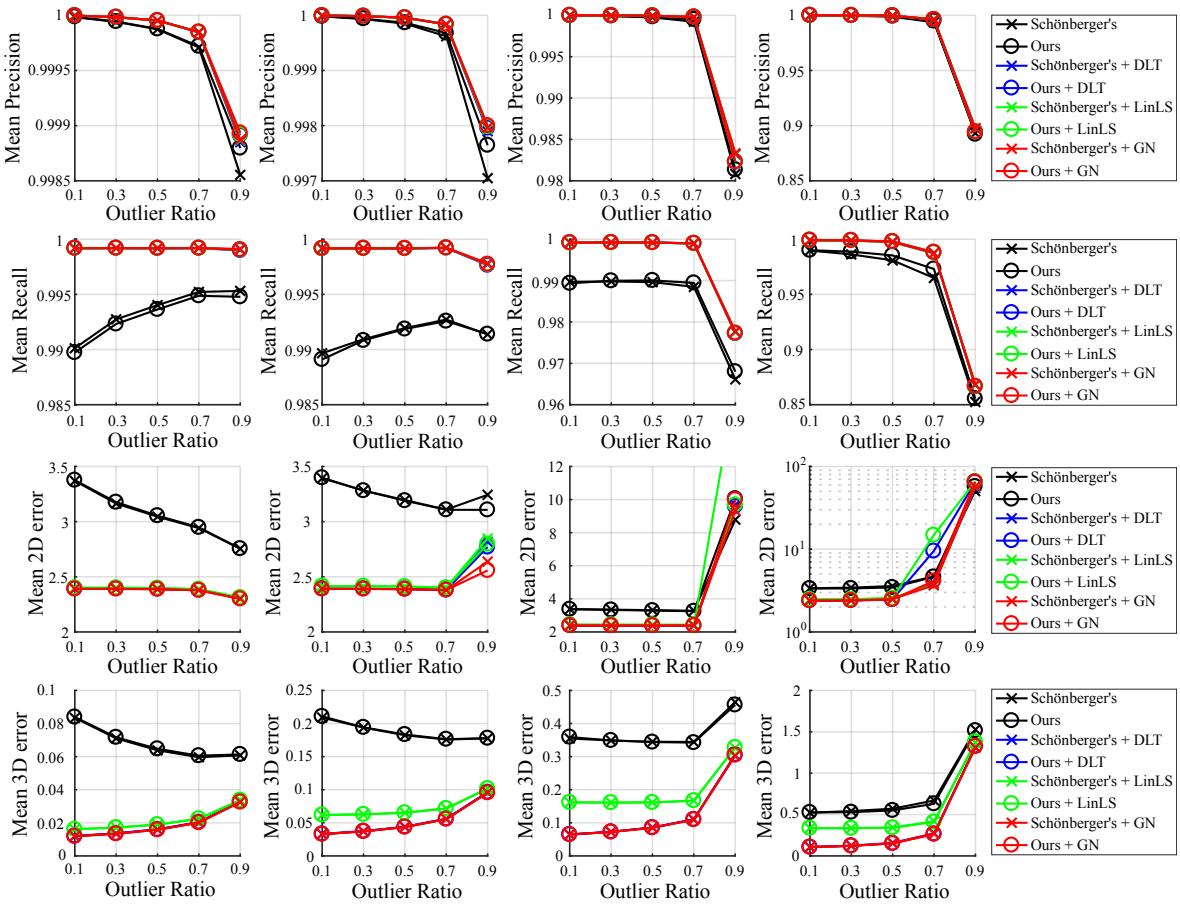


Figure 6.3: Triangulation Performance. From left to right, the columns correspond to the different point distances (3, 5, 7, 9 unit). One unit corresponds to the geometric span of the cameras. The mean 2D error is computed with respect to **all true inlying** observations.

Algorithm 4: Proposed 3D uncertainty estimation

Input: $G, \mathcal{I}, \overline{e}_{2D}, \delta_{\text{pair}}, \mathbf{u}_i, \widehat{\mathbf{f}}_i^w, \mathbf{K}_i, \mathbf{R}_i$ for all $i \in \mathcal{V}$.
Output: σ_{3D}.

/* Estimate the maximum parallax angle. */
$p_{\min} \leftarrow \infty$, $\delta_{\text{pair}} \leftarrow \min\left(\delta_{\text{pair}}, |\mathcal{I}|(|\mathcal{I}|-1)/2\right)$; $n_{\text{pair}} \leftarrow 0$
while $n_{\text{pair}} < \delta_{\text{pair}}$ **do**
  $n_{\text{pair}} \leftarrow n_{\text{pair}} + 1$;
  Pick a random pair of views $j, k \in \mathcal{I}$;
  if $\widehat{\mathbf{f}}_j^w = \mathbf{0}$ **then** compute $\widehat{\mathbf{f}}_j^w$ using (6.14);
  if $\widehat{\mathbf{f}}_k^w = \mathbf{0}$ **then** compute $\widehat{\mathbf{f}}_k^w$ using (6.14);
  $p \leftarrow \left|\widehat{\mathbf{f}}_j^w \cdot \widehat{\mathbf{f}}_k^w\right|$;
  if $p < p_{\min}$ **then** $p_{\min} \leftarrow p$;
$\beta_{\max} \leftarrow \cos^{-1}(p_{\min})$;
/* Interpolate the uncertainty. */
$n_{\text{in}} \leftarrow \min(|\mathcal{I}|, 50)$; $\overline{e}_{2D} \leftarrow \min(\overline{e}_{2D}, 20 \text{ pix})$; $\beta_{\max} \leftarrow \min(\beta_{\max}, 20°)$;
Obtain σ_{3D} by performing trilinear interpolation on the 3D grid G at $(n_{\text{in}}, \overline{e}_{2D}, \beta_{\max})$;
return σ_{3D};

n	2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 35, 40, 45, 50.
d	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 30.
σ (pix)	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30.
n_{run}	10000 if $n \leq 5$, 5000 if $6 < n \leq 20$, and 3000 if $n > 20$.

Table 6.1: Simulation setup for the 3D uncertainty estimation. n: number of cameras, d: distance between the point and the origin, σ: std. dev. of Gaussian noise in the image coordinates, n_{run}: number of independent simulation runs for each configuration (n, d, σ).

90 percent. Varying d and the outlier ratio results in 4×5 configurations, so in total, this amounts to two million unique triangulation problems.

On this dataset, we compare our method against the state of the art ((Schönberger and Frahm, 2016) by Schönberger and Frahm) with and without the local optimization (`DLT`, `LinLS` and `GN`). In Alg. 1, we set $\eta = 0.99$, $\delta_{2D} = 10$ pix, $\delta_{\text{epipolar}} = 0.01$, $\delta_{\text{update}} = 0.1$ pix, $\delta_{\text{lower}} = 0$, $\delta_{\text{upper}} = \cos(4°)$ and $\delta_{\text{pair}} = 100$. Fig. 6.3 shows the results. On average, ours and (Schönberger and Frahm, 2016) perform similarly (but ours is faster, as will be shown later). For both methods, the local optimization substantially improves the 2D and 3D accuracy. Thanks to the iterative update of the inlier set, we also see a significant gain in recall. Among the optimization methods, `DLT` and `GN` show similar performance in all criteria, while `LinLS` exhibits larger 3D error than the other two. We provide a closer comparison between `DLT` and `GN` in the next section.

In general, when the point is far and the outlier ratio is high, the performance degrades for all methods. At any fixed outlier ratio, we observe that the 3D error tends to grow with the point distance. However, the same cannot be said for the 2D error. This is because given

sufficient parallax, the 2D accuracy is mostly influenced by the image noise statistics, rather than the geometric configurations.

We also evaluate the accuracy after pruning the most uncertain points using our method (Sect. 6.4.1). In Fig. 6.4, we plot the error histograms of the points with different levels of the estimated 3D uncertainty (σ_{3D}). It shows that with a smaller threshold on σ_{3D}, we get to prune more points with larger 3D error. Fig. 6.5 shows the cumulative 3D error plots. It illustrates that thresholding on σ_{3D} gives us some control over the upper bound of the 3D error. As a result, we are able to trade off the number points for 3D accuracy by varying the threshold level. This is shown in Fig. 6.6.

To compare the timings, all methods are implemented in MATLAB and run on a laptop CPU (Intel i7-4810MQ, 2.8GHz). Tab. 6.2 provides the relative speed of our two-view RANSAC compared to (Schönberger and Frahm, 2016). It shows that ours is faster, especially when the point is far and the outlier ratio is high. This demonstrates the advantage of the early termination of two-view triangulation (Alg. 2). In Tab. 6.3, we present the timings of the local optimization and uncertainty estimation. We found that `DLT` is slightly faster than `LinLS` and almost twice faster than `GN`.

6.4.3 DLT vs. Gauss-Newton Method

To compare the accuracy of `DLT` and `GN` more closely, we perform additional simulations in outlier-free scenarios. The simulation is set up in a similar way as in Section 6.4.1.

In terms of 3D accuracy, we found that the two methods perform almost equally most of the time. The comparison is inconsistent only when the maximum parallax angle is very small (less than 6 deg or so).

As for the 2D accuracy, the difference is sometimes noticeable. Fig. 6.7 shows the mean and the maximum difference of 2D error. On average, `GN` offers less gain for more cameras, smaller noise and lower parallax. This explains why we could not see the difference between `DLT` and `GN` in Fig. 6.3. However, the bottom row of Fig. 6.7 reveals that `GN` sometimes

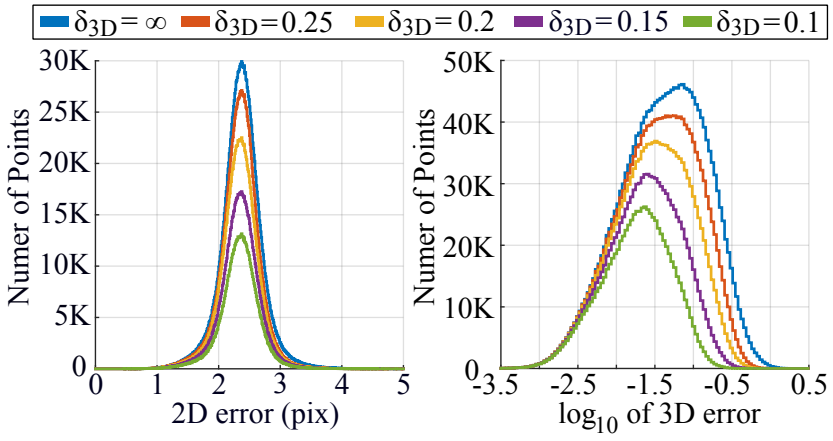


Figure 6.4: Error histograms of the triangulated points with the mean 2D error < 5 pix and the estimated uncertainty $\sigma_{3D} < \delta_{3D}$.

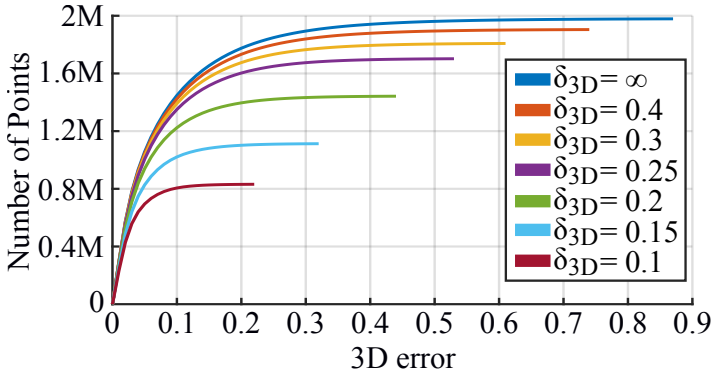


Figure 6.5: Cumulative error plots of the triangulated points with the mean 2D error < 5 pix and the estimated uncertainty $\sigma_{3D} < \delta_{3D}$. We truncate each curve at 99.9% accumulation.

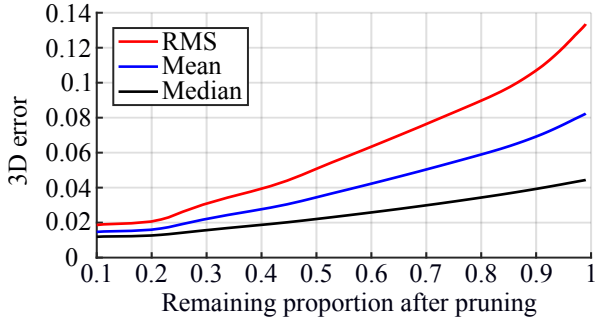


Figure 6.6: Trade-off between the 3D error and the number of points by varying the uncertainty threshold for pruning. As in Fig. 6.4 and 6.5, we only consider the points with the mean 2D error < 5 pix.

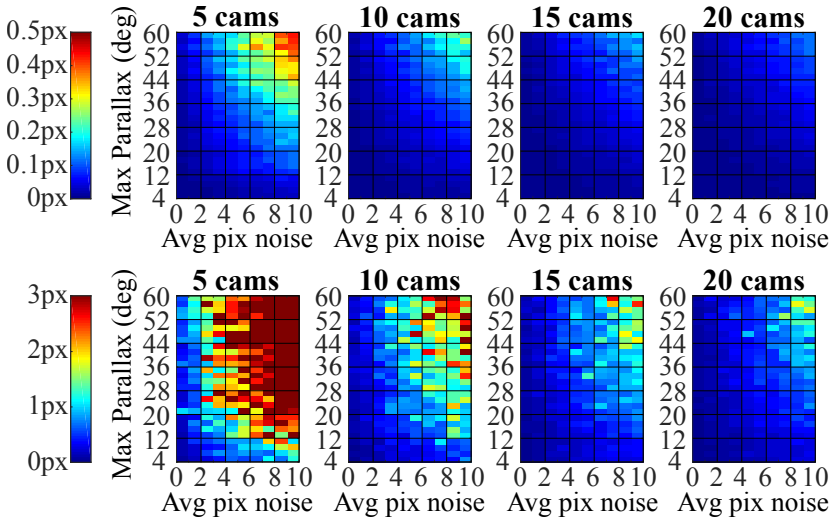


Figure 6.7: **Top row:** Mean decrease of 2D error in L_1 norm by performing `GN` in addition to `DLT`, *i.e.*, $\overline{e}_{\text{2D DLT}} - \overline{e}_{\text{2D GN}}$, for different configurations. The redder the color, the more accurate `GN` is than `DLT`. **Bottom row:** Maximum decrease of 2D error, *i.e.*, $\max(e_{\text{2D DLT}} - e_{\text{2D GN}})$.

	$d = 3$	$d = 5$	$d = 7$	$d = 9$
OR $= 10\%$	4.15, 4.15 ($\times\mathbf{1.00}$)	4.43, 4.32 ($\times\mathbf{1.03}$)	4.81, 4.40 ($\times\mathbf{1.09}$)	5.90, 4.47 ($\times\mathbf{1.32}$)
OR $= 30\%$	4.61, 4.39 ($\times\mathbf{1.05}$)	4.80, 4.45 ($\times\mathbf{1.08}$)	4.78, 4.08 ($\times\mathbf{1.17}$)	7.33, 4.79 ($\times\mathbf{1.53}$)
OR $= 50\%$	5.28, 4.56 ($\times\mathbf{1.16}$)	5.64, 4.72 ($\times\mathbf{1.20}$)	5.87, 4.38 ($\times\mathbf{1.34}$)	10.4, 5.22 ($\times\mathbf{1.99}$)
OR $= 70\%$	7.46, 5.09 ($\times\mathbf{1.46}$)	8.06, 5.30 ($\times\mathbf{1.52}$)	9.13, 4.93 ($\times\mathbf{1.85}$)	19.9, 6.45 ($\times\mathbf{3.09}$)
OR $= 90\%$	28.6, 7.64 ($\times\mathbf{3.74}$)	31.8, 8.14 ($\times\mathbf{3.91}$)	40.8, 8.86 ($\times\mathbf{4.61}$)	72.6, 13.1 ($\times\mathbf{5.56}$)

Table 6.2: RANSAC time per point (ms). The two entries respectively correspond to (Schönberger and Frahm, 2016) and ours without local optimization. The relative speed of ours compared to (Schönberger and Frahm, 2016) is given in parentheses.

	DLT	LinLS	GN	Uncertainty Est.
OR $= 10\%$	**1.57**	1.62	3.06	1.64
OR $= 30\%$	**1.18**	1.27	2.39	1.44
OR $= 50\%$	**0.84**	0.95	1.78	1.25
OR $= 70\%$	**0.57**	0.64	1.17	1.04
OR $= 90\%$	**0.25**	0.29	0.50	0.42

Table 6.3: Optimization and uncertainty estimation time per point (ms). The fastest optimization result is shown in bold.

provides a significant gain over `DLT` even when the average difference is small.

6.4.4 Results on Real Data

We evaluate our method on three real datasets: Dinosaur (*Oxford Multiview Datasets*, n.d.), Corridor (*Oxford Multiview Datasets*, n.d.) and Notre Dame (Snavely et al., 2006). We only consider the points that are visible in three or more views. In our algorithm, we use the same parameters as in Sect. 6.4.2 and discard the point that is visible in less than three views after RANSAC. Fig. 6.8 shows the 3D reconstruction results.

6.5 Conclusions

In this chapter, we presented a robust and efficient method for multiview triangulation and uncertainty estimation. We proposed several early termination criteria for two-view RANSAC using the midpoint method, and showed that it improves the efficiency when the outlier ratio is high. We also compared the three local optimization methods (`DLT`, `LinLS` and `GN`), and found that `DLT` and `GN` are similar (but better than `LinLS`) in terms of 3D accuracy, while `GN` is sometimes much more accurate than `DLT` in terms of 2D accuracy. Finally, we proposed a novel method to estimate the uncertainty of a triangulated point based on the

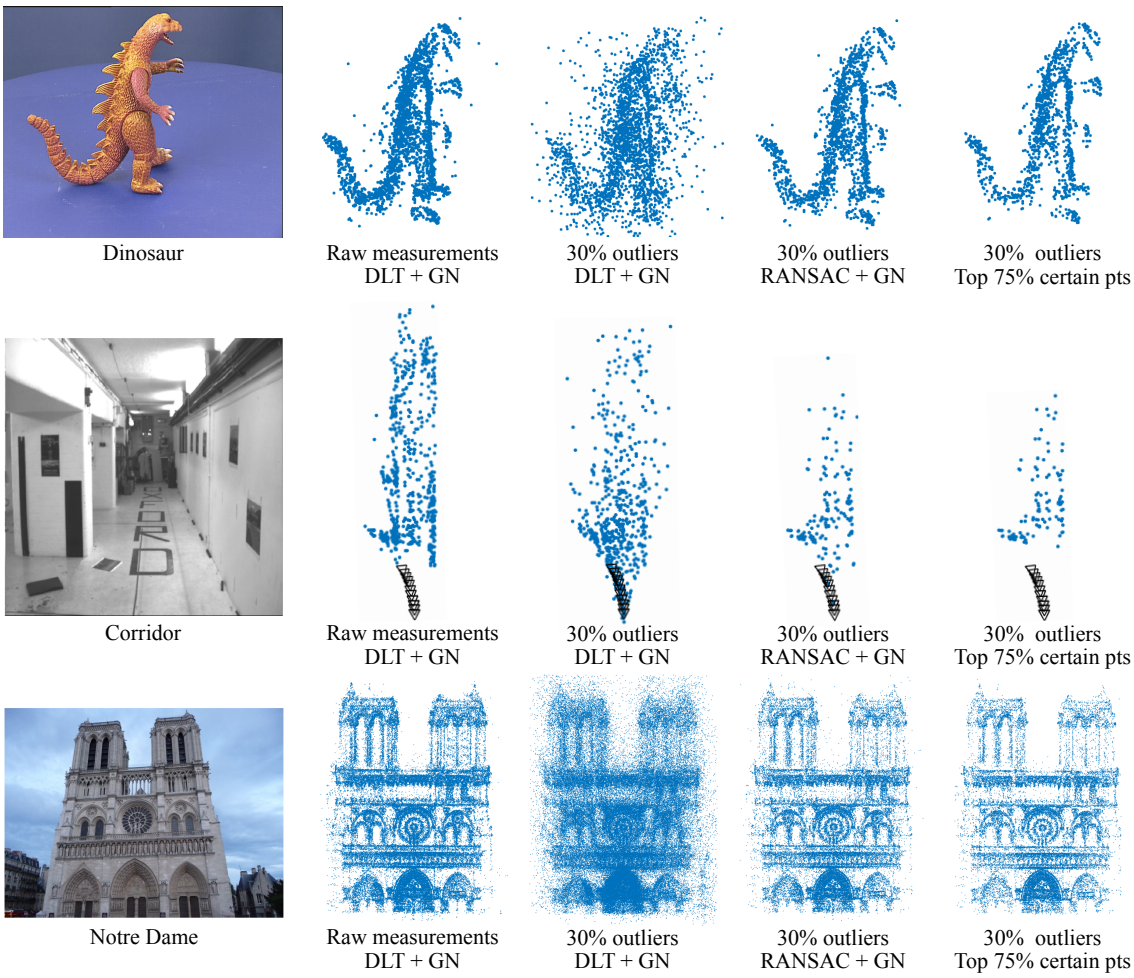


Figure 6.8: **1st column:** Sample images. **2nd column:** Reconstruction using the `GN` method (initialized by the `DLT` method), assuming no outliers. **3rd column:** Reconstruction using the same method on outlier-contaminated measurements. We perturb 30% of the measurements with uniform noise between 10 and 100 pix. **4th column:** Reconstruction using our method (Alg. 1) on the same contaminated measurements. We observe that our RANSAC method is effective against the outliers. **5th column:** From the previous result, we prune the top 25% of the most uncertain points identified by our method (Sect. 6.3.3). We use the uncertainty model we learned from the simulations in Sect. 6.4.1. Notice that some of the most inaccurate points are removed.

number of (inlying) views, the mean reprojection error and the maximum parallax angle. We showed that the estimated uncertainty can be used to control the 3D accuracy. An extensive evaluation was performed on both synthetic and real data.

Chapter 7

Robust Single Rotation Averaging

In this chapter, we propose a novel method for single rotation averaging using the Weiszfeld algorithm. Our contribution is threefold: First, we propose a robust initialization based on the elementwise median of the input rotation matrices. Our initial solution is more accurate and robust than the commonly used chordal L2-mean. Second, we propose an outlier rejection scheme that can be incorporated in the Weiszfeld algorithm to improve the robustness of L1 rotation averaging. Third, we propose a method for approximating the chordal L1-mean using the Weiszfeld algorithm. An extensive evaluation shows that both our method and the state of the art perform equally well with the proposed outlier rejection scheme, but ours is 2–4 times faster.

7.1 Introduction

consider the problem of single rotation averaging, i.e., averaging several estimates of a single rotation to obtain the best estimate. This problem is relevant in many applications such as structure from motion (SfM) (Hartley et al., 2011, Tron et al., 2016), rotation synchronization (Lee and Civera, 2022), camera rig calibration (Dai et al., 2009), motion capture (Sharf et al., 2010), satellite/spacecraft attitude determination (Lam and Crassidis, 2007, Markley et al., 2007) and crystallography (Humbert et al., 1996, Morawiec, 1998).

A standard approach for single rotation averaging is to find the rotation that minimizes a cost function based on the distance to the input rotations. We refer to (Hartley, Trumpf, Dai and Li, 2013) for an extensive study of various distance functions. The current state-of-the-art method is to minimize the sum of geodesic distances using the Weiszfeld algorithm on $SO(3)$ (Hartley et al., 2011).

In this chapter, we propose a novel method, also based on the Weiszfeld algorithm (Weiszfeld, 1937, Weiszfeld and Plastria, 2009), that is faster and more robust than (Hartley et al., 2011). Our contributions are as follows:

1. A robust initialization from the elementwise median of the input rotation matrices (Section 7.3.1).

2. An implicit outlier rejection scheme performed at each iteration of the Weiszfeld algorithm (Section 7.3.2).
3. An approximation of the chordal median in $SO(3)$ using the Weiszfeld algorithm (Section 7.3.3).

We substantiate our claim through extensive evaluation on synthetic data (Section 7.4).

7.2 Preliminaries

We denote the vectorization of an $n \times m$ matrix by $\mathrm{vec}(\cdot)$ and its inverse by $\mathrm{vec}_{n\times m}^{-1}(\cdot)$. For a 3D vector $\mathbf{v}$, we define $\mathbf{v}^\wedge$ as the corresponding 3×3 skew-symmetric matrix, and denote the inverse operator by $(\cdot)^\vee$, i.e., $(\mathbf{v}^\wedge)^\vee = \mathbf{v}$. The Euclidean, the L_1 and the Frobenius norm are respectively denoted by $\|\cdot\|$, $\|\cdot\|_1$ and $\|\cdot\|_F$. A rotation can be represented by a rotation matrix $\mathbf{R} \in SO(3)$ or a rotation vector $\mathbf{v} = \theta\hat{\mathbf{v}}$ where θ and $\hat{\mathbf{v}}$ are the angle and the unit axis of the rotation, respectively. The two representations are related by Rodrigues formula, and we denote the corresponding mapping between them by $\mathrm{Exp}(\cdot)$ and $\mathrm{Log}(\cdot)$ (Forster, Carlone, Dellaert and Scaramuzza, 2017):

$$\mathbf{R} = \mathrm{Exp}(\mathbf{v}) := \mathbf{I} + \frac{\sin\left(\|\mathbf{v}\|\right)}{\|\mathbf{v}\|}\mathbf{v}^\wedge + \frac{1-\cos\left(\|\mathbf{v}\|\right)}{\|\mathbf{v}\|^2}\left(\mathbf{v}^\wedge\right)^2, \tag{7.1}$$

$$\mathbf{v} = \mathrm{Log}(\mathbf{R}) := \frac{\theta}{2\sin(\theta)}\left(\mathbf{R}-\mathbf{R}^\top\right)^\vee \tag{7.2}$$

$$\text{with} \quad \theta = \cos^{-1}\left(\frac{\mathrm{tr}(\mathbf{R})-1}{2}\right). \tag{7.3}$$

The geodesic distance between two rotations $d_\angle(\mathbf{R}_1, \mathbf{R}_2)$ is obtained by substituting $\mathbf{R}_1\mathbf{R}_2^\top$ into $\mathbf{R}$ in (7.3). In (Hartley, Trumpf, Dai and Li, 2013), it was shown that the chordal distance is related to the geodesic distance by the following equation:

$$d_{\text{chord}}(\mathbf{R}_1, \mathbf{R}_2) := \|\mathbf{R}_1 - \mathbf{R}_2\|_F \tag{7.4}$$

$$= 2\sqrt{2}\sin\left(d_\angle(\mathbf{R}_1, \mathbf{R}_2)/2\right). \tag{7.5}$$

We define $\mathrm{proj}_{SO(3)}(\cdot)$ as the projection of the 3×3 matrix onto the special orthogonal group $SO(3)$, which gives the closest rotation in the Frobenius norm (Arun et al., 1987): For $\mathbf{M} \in \mathbb{R}^{3\times3}$,

$$\mathrm{proj}_{SO(3)}(\mathbf{M}) := \mathbf{U}\mathbf{W}\mathbf{V}^\top, \tag{7.6}$$

where

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = \mathrm{SVD}\left(\mathbf{M}\right), \tag{7.7}$$

$$\mathbf{W} = \begin{cases} \mathrm{diag}(1,1,-1) & \text{if } \det\left(\mathbf{U}\mathbf{V}^\top\right) < 0 \\ \mathbf{I}_{3\times3} & \text{otherwise} \end{cases}. \tag{7.8}$$

7.3 Method

7.3.1 Robust Initialization

In (Hartley et al., 2011), the chordal L_2-mean of the rotations is taken as the starting point of the Weiszfeld algorithm. For input rotations $\{\mathbf{R}_i\}_{i=1}^N$, it is given by $\text{proj}_{SO(3)}\left(\sum_{i=1}^N \mathbf{R}_i\right)$ (Hartley, Trumpf, Dai and Li, 2013). Although this initial solution can be obtained very fast, it is often inaccurate and sensitive to outliers. To overcome this weakness, we propose to initialize using the following matrix:

$$\mathbf{S}_0 = \underset{\mathbf{S} \in \mathbb{R}^{3\times 3}}{\text{argmin}} \sum_{i=1}^{N}\sum_{j=1}^{3}\sum_{k=1}^{3} \left|(\mathbf{R}_i - \mathbf{S})_{jk}\right|, \tag{7.9}$$

where the subscript jk denote the element at the j-th row and the k-th column of the matrix. Note that $\sum_{j,k} |\mathbf{M}_{jk}|$ is called the elementwise L_1 norm of the matrix $\mathbf{M}$. See Fig. 7.1 for the geometric interpretation of this distance metric. Since the nine entries of $\mathbf{S}$ are independent, we can consider them separately in 1D space. Then, the entry of $\mathbf{S}_0$ at location (j,k) minimizes the sum of absolute deviations from the entries of $\mathbf{R}_i$'s at (j,k), meaning that it is simply their median:

$$(\mathbf{S}_0)_{jk} = \text{median}\left(\{(\mathbf{R}_i)_{jk}\}_{i=1}^N\right) \quad \text{for all} \quad j,k \in \{1,2,3\}. \tag{7.10}$$

The initial rotation matrix is then obtained by projecting $\mathbf{S}_0$ onto $SO(3)$:

$$\mathbf{R}_0 = \text{proj}_{SO(3)}\left(\mathbf{S}_0\right). \tag{7.11}$$

7.3.2 Outlier Rejection in the Weiszfeld Algorithm

The geodesic L_1-mean (i.e., median) of the rotations is defined as

$$\mathbf{R}_{\text{gm}} = \underset{\mathbf{R}\in SO(3)}{\text{argmin}} \sum_{i=1}^{N} d_{\angle}(\mathbf{R}_i, \mathbf{R}). \tag{7.12}$$

In (Hartley et al., 2011), it was shown that this can be computed using the Weiszfeld algorithm on $SO(3)$ and that it is more robust to outliers than the L_2-mean. However, a large number of outliers is still critical to the accuracy. To further mitigate the influence of the outliers, we modify the Weiszfeld algorithm of (Hartley et al., 2011) such that the large residuals are given zero weight at each iteration. Specifically, we disregard all the residuals larger than $\max(d_{Q1}, d_{\text{max}})$ where d_{Q1} is the first quartile of the residuals at each iteration and d_{max} is some threshold we set in order to avoid discarding inliers. The details are given in Algorithm 5. A similar approach of disregarding large residuals was used in (Ferraz et al., 2014)

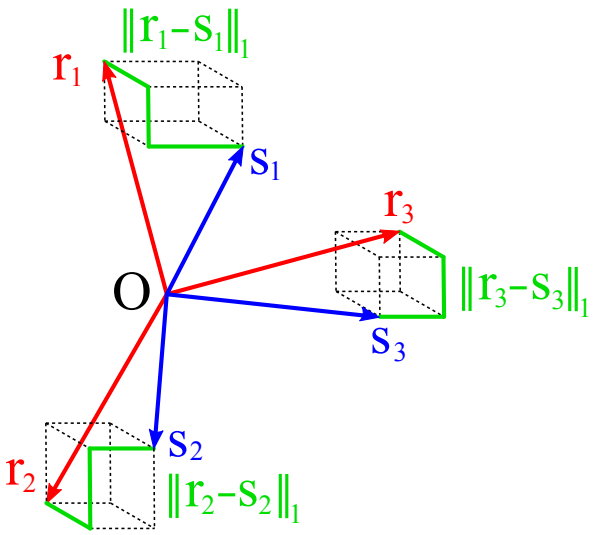


Figure 7.1: The elementwise L_1 norm of $(\mathbf{R}-\mathbf{S})$ is equal to $\sum_{i=1}^{3}\|\mathbf{r}_i-\mathbf{s}_i\|_1$ where $\mathbf{R}=[\mathbf{r}_1,\mathbf{r}_2,\mathbf{r}_3]$ and $\mathbf{S}=[\mathbf{s}_1,\mathbf{s}_2,\mathbf{s}_3]$. This can be thought as the total length of the green lines.

for robust Perspective-n-Point (PnP) problem. Note that our approach contrasts (Aftab and Hartley, 2015) where a smooth robust cost function is favored for theoretically guaranteed convergence. In practice, our method is more robust to outliers (see Section 7.4.2).

7.3.3 Approximate Chordal L_1-Mean

The chordal L_1-mean of the rotations is defined as

$$\mathbf{R}_{\text{cm}} = \underset{\mathbf{R}\in SO(3)}{\operatorname{argmin}} \sum_{i=1}^{N} d_{\text{chord}}(\mathbf{R}_i,\mathbf{R}). \tag{7.13}$$

In (Hartley, Trumpf, Dai and Li, 2013), a locally convergent algorithm on $SO(3)$ is proposed for this problem. In this chapter, we propose a different approach: Instead of iteratively updating the estimate on $SO(3)$, we first embed the rotations in a Euclidean space $\mathbb{R}^9$, find their geometric median in $\mathbb{R}^9$ using the standard Weiszfeld algorithm (Weiszfeld, 1937, Weiszfeld and Plastria, 2009) (which is globally convergent), and then project this median onto $SO(3)$. In other words, we approximate $\mathbf{R}_{\text{cm}}$ as

$$\mathbf{R}_{\text{cm}} \approx \operatorname{proj}_{SO(3)}(\mathbf{S}_{\text{cm}}) \tag{7.14}$$

$$\text{with } \mathbf{S}_{\text{cm}} = \underset{\mathbf{S}\,\in\,\mathbb{R}^{3\times 3}}{\operatorname{argmin}} \sum_{i=1}^{N} \|\mathbf{R}_i-\mathbf{S}\|_F \tag{7.15}$$

$$= \operatorname{vec}_{3\times 3}^{-1}\left(\underset{\mathbf{s}\,\in\,\mathbb{R}^{9}}{\operatorname{argmin}} \sum_{i=1}^{N} \|\operatorname{vec}(\mathbf{R}_i)-\mathbf{s}\|\right). \tag{7.16}$$

Since the optimization is performed using the Weiszfeld algorithm, we can also incorporate the initialization and outlier rejection scheme in the previous sections. Algorithm 6 summarizes our method.

We point out two things in the implementation: First, since we do not optimize on $SO(3)$, the initial estimate does not have to come from a rotation, and we omit (7.11). Second, the threshold d_{max} must be scaled appropriately when comparing Algorithm 5 and 6. Assuming that $\mathbf{s} \in \mathbb{R}^9$ at each iteration does not vastly differ from an embedding of a rotation in $\mathbb{R}^9$, we convert d_{max} from geodesic to chordal using (7.5), and vice versa. This is done in line 10 of Algorithm 6.

7.4 Results

7.4.1 Initialization

For evaluation, we generated a synthetic dataset where the inlier rotations follow a Gaussian distribution with $\sigma = 5°$, and the outliers have uniformly distributed angles $\in [0, \pi]$ at random directions. Fig. 7.2 compares the average accuracy of the proposed initial solution (Section 7.3.1) and the chordal L_2-mean (Hartley et al., 2011) over 1000 runs. It can be seen that our solution is significantly better than the chordal L_2-mean unless the outlier ratio is extremely high (i.e., above 90%). On average, the L_2 chordal method takes 0.37 μs and ours 0.83 μs per rotation. This time difference is insignificant compared to the optimization that follows (see Table 7.1).

7.4.2 Comparison against (Hartley et al., 2011)

Using the same setup as in previous section, we compare Algorithm 5 and 6, with and without the proposed outlier rejection scheme (Section 7.3.2). This time, we consider two different inlier noise levels, $\sigma = 5°$ and $15°$. The average accuracy of the evaluated methods[1] is compared in Fig. 7.3. With the outlier rejection, the geodesic L_1-mean and our approximate chordal L_1-mean are almost equally accurate. Without the outlier rejection, the geodesic L_1-mean is more accurate than our approximate chordal L_1-mean, but only for very high outlier ratios (i.e, $> 50\%$). Otherwise, there is no significant difference between the two.

The computation times are reported in Table 7.1. Our method is always faster than (Hartley et al., 2011), and is 2–4 times faster with the outlier rejection. That said, the speed is not a major advantage, since all methods can process several hundreds of rotations in less than a millisecond. In most cases, averaging rotations will take much less time than other operations, such as the computation of input rotations.

[1] We did not include the chordal L_2-mean here, since it produced much larger errors than the rest and was already reported in Fig. 7.2.

Algorithm 5: Geodesic median in $SO(3)$ (Hartley et al., 2011) with outlier rejection

Input: List of rotation matrices $\{\mathbf{R}_i\}_{i=1}^N$
Output: $\mathbf{R}_{\text{gm}}$

/* Initialize (Section 7.3.1). */
$\mathbf{S}_0 \leftarrow \mathbf{0}_{3\times 3}$;
$(\mathbf{S}_0)_{jk} \leftarrow \text{median}\left(\{(\mathbf{R}_i)_{jk}\}_{i=1}^N\right) \quad \forall j,k = 1,2,3$;
$\mathbf{R}_0 \leftarrow \text{proj}_{SO(3)}(\mathbf{S}_0)$;
/* Run the Weiszfeld algorithm on SO(3) (Hartley et al., 2011) with outlier rejection (Section 7.3.2). */
$\mathbf{R}_{\text{gm}} \leftarrow \mathbf{R}_0$;
for $\text{it} = 1, 2, \cdots, 10$ **do**
  while $\mathbf{R}_{\text{gm}} \in \{R_i\}_{i=1}^N$ **do**
    $\mathbf{R}_{\text{gm}} \leftarrow \mathbf{R}_{\text{perturb}}\mathbf{R}_{\text{gm}}$; // Perturb slightly
  $\mathbf{v}_i \leftarrow \text{Log}\left(\mathbf{R}_i\mathbf{R}_{\text{gm}}^\top\right) \quad \forall i = 1, \cdots, N$;
  $d_i \leftarrow \|\mathbf{v}_i\| \quad \forall i = 1, \cdots, N$;
  $d_{\text{Q1}} \leftarrow Q_1(\{d_1, \cdots, d_N\})$; // First quartile
  $d_{\text{max}} \leftarrow \begin{cases} 1 & \text{if } N \leq 50 \\ 0.5 & \text{otherwise} \end{cases}$
  $d_{\text{thr}} \leftarrow \max(d_{\text{Q1}}, d_{\text{max}})$;
  $w_i \leftarrow \begin{cases} 1 & \text{if } d_i \leq d_{\text{thr}} \\ 0 & \text{otherwise} \end{cases} \quad \forall i = 1, \cdots, N$;
  $\Delta\mathbf{v} \leftarrow \dfrac{\sum_{i=1}^N w_i\mathbf{v}_i/d_i}{\sum_{i=1}^N w_i/d_i}$;
  $\mathbf{R}_{\text{gm}} \leftarrow \text{Exp}(\Delta\mathbf{v})\mathbf{R}_{\text{gm}}$;
  if $\|\Delta\mathbf{v}\| < 0.001$ **then**
    break;
return $\mathbf{R}_{\text{gm}}$

7.4.3 Ablation study

In another experiment, we ran our method (with outlier rejection) initialized with the chordal L_2-mean instead of the proposed solution in Section 7.3.1. To our surprise, we found that there is almost no difference in the final accuracy. This suggests that combining the L_1-mean with the outlier rejection in Section 7.3.2 is already robust enough that a less accurate initial solution can still lead to a similar result in the end.

Algorithm 6: Approximate chordal median in $SO(3)$ with outlier rejection

Input: List of rotation matrices $\{\mathbf{R}_i\}_{i=1}^N$
Output: $\mathbf{R}_{\text{cm}}$

/* Initialize (Section 7.3.1). */
$\mathbf{S}_0 \leftarrow \mathbf{0}_{3\times 3}$;
$(\mathbf{S}_0)_{jk} \leftarrow \text{median}\left(\{(\mathbf{R}_i)_{jk}\}_{i=1}^N\right) \quad \forall j,k = 1,2,3$;
/* Run the Weiszfeld algorithm in 9D space with outlier rejection (Section 7.3.2). */
$\mathbf{s}_{\text{cm}} \leftarrow \text{vec}(\mathbf{S}_0)$;
for $\text{it} = 1, 2, \cdots, 10$ **do**
  while $\mathbf{s}_{\text{cm}} \in \{\text{vec}(\mathbf{R}_i)\}_{i=1}^N$ **do**
    $\mathbf{s}_{\text{cm}} \leftarrow \mathbf{s}_{\text{cm}} + \mathcal{U}(0, 0.001)$; // Perturb
  $\mathbf{v}_i \leftarrow \text{vec}\left(\mathbf{R}_i\right) - \mathbf{s}_{\text{cm}} \; \forall i = 1, \cdots, N$;
  $d_i \leftarrow \|\mathbf{v}_i\| \;\; \forall i = 1, \cdots, N$;
  $d_{\text{Q1}} \leftarrow Q_1\left(\{d_1, \cdots, d_N\}\right)$; // First quartile
  $d_{\text{max}} \leftarrow \begin{cases} 2\sqrt{2}\sin(1/2) \approx 1.356 & \text{if } N \leq 50 \\ 2\sqrt{2}\sin(0.5/2) \approx 0.700 & \text{otherwise} \end{cases}$
  $d_{\text{thr}} \leftarrow \max\left(d_{\text{Q1}}, d_{\text{max}}\right)$;
  $w_i \leftarrow \begin{cases} 1 & \text{if } d_i \leq d_{\text{thr}} \\ 0 & \text{otherwise} \end{cases} \quad \forall i = 1, \cdots, N$;
  $\mathbf{s}_{\text{cm,prev}} \leftarrow \mathbf{s}_{\text{cm}}$;
  $\mathbf{s}_{\text{cm}} \leftarrow \dfrac{\sum_{i=1}^N w_i \mathbf{v}_i / d_i}{\sum_{i=1}^N w_i / d_i}$;
  if $\|\mathbf{s}_{\text{cm}} - \mathbf{s}_{\text{cm,prev}}\| < 0.001$ **then**
    break;
$\mathbf{R}_{\text{cm}} = \text{proj}_{SO(3)}(\text{vec}_{3\times 3}^{-1}\left(\mathbf{s}_{\text{cm}}\right))$;
return $\mathbf{R}_{\text{cm}}$

7.5 Real-world applications

In (Lee and Civera, 2022) and (Sun, 2022), our open-source implementation was adopted in the pipeline for multiple rotation averaging (aka rotation synchronization) and point cloud registration, respectively. At the time of this writing, these two works demonstrate state-of-the-art results in respective domains. Interested readers are referred to these two works for the practical applications of our algorithm on real data.

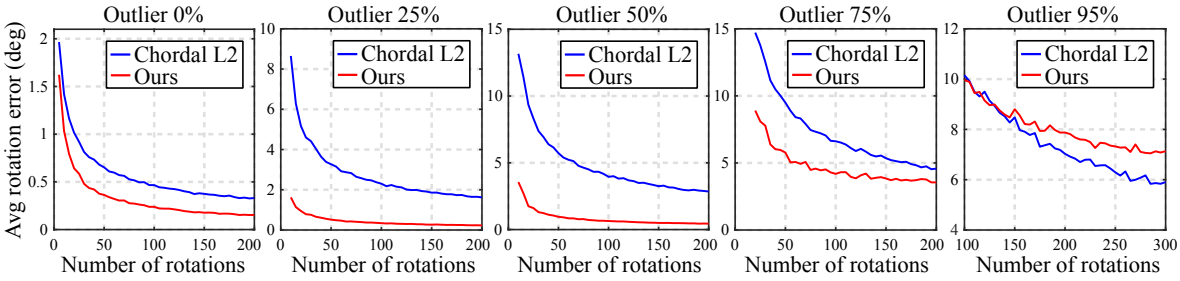


Figure 7.2: Average rotation errors of different initialization methods: Chordal L_2-mean (Hartley et al., 2011) versus ours (Section 7.3.1).

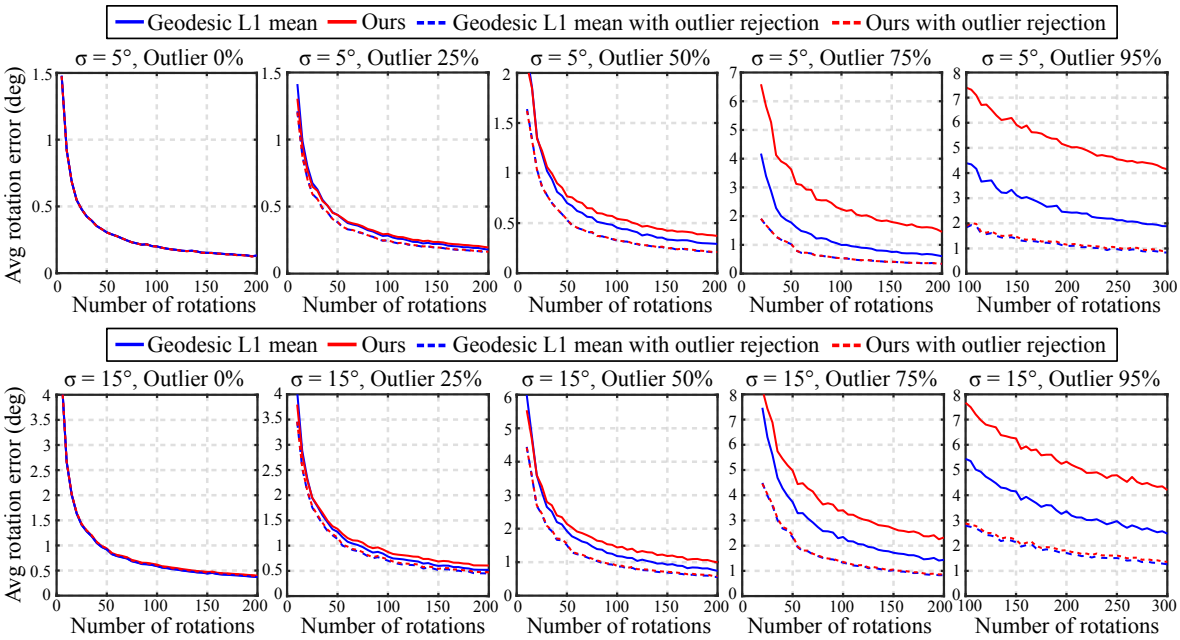


Figure 7.3: Average rotation errors of geodesic L_1-mean (Hartley et al., 2011) versus ours (Section 7.3.3), with and without the outlier rejection (Section 7.3.2).

	w/o outlier rejection		w/ outlier rejection	
	(Hartley et al., 2011)	Ours	(Hartley et al., 2011)	Ours
(5°, 0%)	8.69	**4.38** (2.0×)	9.37	**4.43** (2.1×)
(5°, 25%)	10.5	**4.47** (2.3×)	11.7	**5.68** (2.1×)
(5°, 50%)	15.2	**6.21** (2.4×)	17.2	**6.78** (2.5×)
(5°, 75%)	24.9	**15.2** (1.6×)	27.2	**7.71** (3.5×)
(5°, 95%)	32.1	**10.3** (3.1×)	31.6	**8.99** (3.5×)
(15°, 0%)	10.7	**6.00** (1.8×)	17.0	**6.02** (2.8×)
(15°, 25%)	15.0	**5.98** (2.5×)	17.0	**7.1** (2.4×)
(15°, 50%)	19.6	**7.66** (2.6×)	22.3	**8.06** (2.8×)
(15°, 75%)	24.9	**11.1** (2.2×)	28.1	**8.77** (3.2×)
(15°, 95%)	29.1	**10.2** (2.9×)	31.7	**8.50** (3.7×)

Table 7.1: Median computation time (μs/rotation) under different inlier noise levels and outlier ratios. The speedup compared to (Hartley et al., 2011) is given in parentheses. All algorithms were implemented in MATLAB and run on a laptop CPU (Intel i7-4810MQ, 2.8 GHz).

7.6 Conclusions

In this chapter, we proposed a novel alternative to the work of Hartley et al. (Hartley et al., 2011) for robust single rotation averaging. While both our method and (Hartley et al., 2011) use the Weiszfeld algorithm, there are three key differences:

1. We initialize the Weiszfeld algorithm using the elementwise median of the input rotation matrices.
2. We implicitly disregard the outliers at each iteration of the Weiszfeld algorithm.
3. We approximate the chordal median on $SO(3)$ instead of the geodesic median as in (Hartley et al., 2011).

As a result, our method achieves better performance in terms of speed and robustness to outliers. We also found that incorporating the proposed outlier rejection in the original implementation of (Hartley et al., 2011) leads to similar performance, but at 2–4 times slower speed than ours.

Chapter 8

HARA: A Hierarchical Approach for Robust Rotation Averaging

In this chapter, we propose a novel hierarchical approach for multiple rotation averaging, dubbed HARA. Our method incrementally initializes the rotation graph based on a hierarchy of triplet support. The key idea is to build a spanning tree by prioritizing the edges with many strong triplet supports and gradually adding those with weaker and fewer supports. This reduces the risk of adding outliers in the spanning tree. As a result, we obtain a robust initial solution that enables us to filter outliers prior to nonlinear optimization. With minimal modification, our approach can also integrate the knowledge of the number of valid 2D-2D correspondences. We perform extensive evaluations on both synthetic and real datasets, demonstrating state-of-the-art results.

8.1 Introduction

We consider the problem of multiple rotation averaging in the presence of outliers, *i.e.*, finding multiple absolute rotations $\mathbf{R}_i$ given a partial set of noisy, outlier-contaminated constraints on relative rotations $\mathbf{R}_{ij} = \mathbf{R}_i\mathbf{R}_j^\top$ (Hartley, Trumpf, Dai and Li, 2013). This problem has direct application to structure-from-motion (SfM) (Martinec and Pajdla, 2007, Enqvist et al., 2011, Arie-Nachimson et al., 2012, Moulon et al., 2013, Wilson and Snavely, 2014, Cui et al., 2015, Cui and Tan, 2015, Ozyesil and Singer, 2015, Cui et al., 2017, Chen et al., 2021, Zhu et al., 2018), multiple point cloud registration (Govindu and Pooja, 2014, Tang and Feng, 2015, Arrigoni et al., 2016, 2018, Huang et al., 2019, Bhattacharya and Govindu, 2019, Gojcic et al., 2020, Moreira et al., 2021*b*) and simultaneous localization and mapping (SLAM) (Carlone, Tron, Daniilidis and Dellaert, 2015, Bourmaud et al., 2014, Bourmaud, 2016, Bustos et al., 2019).

In most global SfM pipelines, multiple rotation averaging is the *de facto* standard for computing the initial orientations of the cameras: After estimating the relative poses between image pairs (*e.g.*, by matching feature descriptors such as SIFT (Lowe, 2004) and

running the 5-point algorithm (Nistér, 2004) with RANSAC (Fischler and Bolles, 1981)), one can solve the rotation averaging problem and obtain the absolute rotations with respect to a common reference frame. These initial rotations are then used in subsequent operations such as translation estimation (Wilson and Snavely, 2014, Cui et al., 2015), pose graph optimization (Carlone, Tron, Daniilidis and Dellaert, 2015, Moreira et al., 2021*b*), multiview triangulation (Lee and Civera, 2019*c*, 2020*c*) and bundle adjustment (Triggs et al., 2000, Hartley and Zisserman, 2003, Lee and Civera, 2021). As a result, all these tasks depend critically on the solution produced by the rotation averaging algorithm.

For this reason, numerous research endeavors have been made in the past decade to develop reliable and versatile rotation averaging methods. However, even without any outliers in the input, solving a large-scale rotation averaging problem is nontrivial (Wilson et al., 2016, Wilson and Bindel, 2020). The problem only gets worse when the input contains outliers, which is often the case in practice (Wilson and Snavely, 2014, Chatterjee and Govindu, 2018, Moreira et al., 2021*b*). These outliers, if not handled properly, can easily degrade the estimation accuracy.

Commonly, rotation averaging is formulated as a nonlinear optimization problem and solved iteratively starting from some initial guess of the absolute rotations (Hartley et al., 2011, Chatterjee and Govindu, 2018, Shi and Lerman, 2020, Chen et al., 2021). If, however, this initial guess is severely affected by outliers, it becomes extremely difficult to obtain an accurate result later on. Therefore, a robust initialization is essential for reliable rotation averaging in the presence of outliers.

In this chapter, we propose a novel method for robust multiple rotation averaging. Our main contribution is a hierarchical initialization scheme that constructs a spanning tree of a rotation graph by propagating most reliable constraints first and less reliable ones later. We establish the hierarchy of reliability based on the number of consistent triplet constraints, as well as their level of consistency. That is, we consider a constraint to be more reliable if it is strongly supported by many other constraints and less reliable if it has weaker or fewer supports. Optionally, we can also incorporate the number of valid 2D-2D correspondences into the hierarchy. Experimental results show that our approach can significantly improve the robustness of rotation averaging.

8.2 Related Work

Early works on motion averaging demonstrated various methods for estimating absolute rotations from pairwise constraints (Govindu, 2001, Sharp et al., 2002, Fusiello et al., 2002, Govindu, 2004, Martinec and Pajdla, 2007). In recent works, the focus has been on either (1) achieving the global optimality in the absence of outliers, or (2) obtaining a robust solution in the presence of outliers. This work belongs to the second group.

(1) Globally optimal methods in outlier-free scenarios:

In (Fredriksson and Olsson, 2012, Carlone, Rosen, Calafiore, Leonard and Dellaert, 2015),

globally optimal methods using Lagrangian duality are proposed. In later works, more advanced optimization methods have been proposed to enhance the speed and scalability, while guaranteeing the global optimality of the solution (Eriksson et al., 2018, 2021, Rosen et al., 2019). For more recent works on optimal methods, we refer to (Dellaert et al., 2020*a*, Parra et al., 2021, Moreira et al., 2021*a*).

(2) Robust methods in the presence of outliers:
Various methods have been proposed to handle outliers (see (Tron et al., 2016) for a survey). First, there are methods that attempt to detect and remove outliers, *e.g.*, (Govindu, 2006, Zach et al., 2010, Crandall et al., 2011). Govindu (Govindu, 2006) uses a RANSAC-based method by sampling random spanning trees. Zach et al. (Zach et al., 2010) employ a more tractable approach based on Bayesian inference from sampled loop inconsistencies. In (Crandall et al., 2011), Crandall et al. use discrete belief propagation on a Markov random field to obtain the initial solution and remove edges with large errors.

On the other hand, some methods do not completely remove outliers, but instead suppress large errors during optimization. For example, Hartley et al. (Hartley et al., 2011) apply single rotation averaging under the L_1 norm to update each absolute rotation in a distributed manner. Wang and Singer (Wang and Singer, 2013) use semidefinite relaxation and an alternating direction method to minimize a cost function based on the L_1 norm. Chatterjee and Govindu (Chatterjee and Govindu, 2013, 2018) apply the Lie-algebraic averaging (Govindu, 2004) using the iteratively reweighted least squares (IRLS) method with a robust loss function. In (Arrigoni et al., 2018), Arrigoni et al. demonstrate that the spectral decomposition method in (Arie-Nachimson et al., 2012) can be robustified using the IRLS method. Shi and Lerman (Shi and Lerman, 2020) proposed an alternative optimization method, called message passing least squares, and demonstrated its advantages over the IRLS approach (Chatterjee and Govindu, 2018).

Other robust methods directly model the presence of outliers in the optimization problem: Boumal et al. (Boumal et al., 2013) take into account the outliers in the noise model and compute the maximum likelihood estimate via Riemannian trust-region optimization. Arrigoni et al. (Arrigoni et al., 2014, 2018) intrinsically include the outliers in the cost function and estimate the rotations via low-rank and sparse matrix decomposition.

Another popular approach is to exploit additional visual information (*e.g.*, the number of inlier feature matches or the similarity score) to identify inlier edges and obtain a robust initial solution (Enqvist et al., 2011, Shen et al., 2016, Cui et al., 2018, Chen et al., 2021, Gao et al., 2020, 2021).

Last but not least, learning-based approaches have been proposed in (Purkait et al., 2020, Yang et al., 2021). Although these supervised methods may not always generalize well to unfamiliar settings, they show impressive performance on data similar to the training data.

8.3 Preliminaries and Notation

We denote the Euclidean and the Frobenius norm of a 3D vector $\mathbf{v}$ by $\|\mathbf{v}\|$ and $\|\mathbf{v}\|_F$, respectively. We represent a rotation with a rotation matrix $\mathbf{R} \in SO(3)$ or a rotation vector $\mathbf{u} = \theta\widehat{\mathbf{u}}$ where θ and $\widehat{\mathbf{u}}$ are the angle and the unit axis of the rotation, respectively. The two representations are related by Rodrigues' formula, and we denote the mapping between them by $\mathrm{Exp}(\cdot)$ and $\mathrm{Log}(\cdot)$ (Solà et al., 2018):

$$\mathbf{R} = \mathrm{Exp}(\mathbf{u}), \quad \mathbf{u} = \mathrm{Log}(\mathbf{R}). \tag{8.1}$$

In the context of SfM, the absolute rotation and translation of camera i are denoted as $\mathbf{R}_i$ and $\mathbf{t}_i$, respectively. Together, they transform a 3D point from the world frame to the camera reference frame: $\mathbf{x}_i = \mathbf{R}_i\mathbf{x}_w + \mathbf{t}_i$. We denote with $\mathbf{R}_{jk}$ the relative rotation between $\mathbf{R}_j$ and $\mathbf{R}_k$, *i.e.*, $\mathbf{R}_{jk} = \mathbf{R}_j\mathbf{R}_k^\top$.

The angular distance between $\mathbf{R}_j$ and $\mathbf{R}_k$ is defined as the angle of the rotation $\mathbf{R}_j\mathbf{R}_k^\top$, *i.e.*,

$$d(\mathbf{R}_j, \mathbf{R}_k) = \|\mathrm{Log}\left(\mathbf{R}_j\mathbf{R}_k^\top\right)\|. \tag{8.2}$$

The chordal distance is related to the angular distance by the following equation (Hartley, Kahl, Olsson and Seo, 2013):

$$\begin{aligned} d_{\text{chord}}(\mathbf{R}_j, \mathbf{R}_k) :=& \|\mathbf{R}_j - \mathbf{R}_k\|_F & (8.3)\\ =& 2\sqrt{2}\sin\left(d(\mathbf{R}_j, \mathbf{R}_k)/2\right). & (8.4) \end{aligned}$$

If both $\mathbf{R}_j$ and $\mathbf{R}_k$ have a small angle, their relative rotation can be approximated using the Baker-Campbell-Hausdorff (BCH) formula (Govindu, 2004):

$$\begin{aligned} \mathbf{R}_j\mathbf{R}_k^\top &\approx \mathrm{Exp}(\mathbf{u}_j - \mathbf{u}_k). & (8.5)\\ \Rightarrow \mathrm{Log}\left(\mathbf{R}_j\mathbf{R}_k^\top\right) &\approx \mathbf{u}_j - \mathbf{u}_k. & (8.6) \end{aligned}$$

In the following, we list some important terminology:

- **Nodes and edges:** Multiple absolute rotations are related to each other in pairs, so the underlying structure can be represented by a graph. In this context, the *nodes* represent the unknown absolute rotations, and the *edges* represent the known pairwise constraints.
- **Neighbors:** When two nodes are connected by an edge, they are each other's *neighbors*.
- **Fixed nodes and family:** Once a node is initialized with some absolute rotation, we call it *fixed*. A *family* refers to the set of all fixed nodes. The goal of the initialization is to have all nodes included in the family.
- **Base node:** One of the fixed nodes can be chosen as the *base node* at any time during the

initialization. This is the node from which the yet-incomplete spanning tree will branch out if a certain condition is met.

- **Consistent triplet:** Node i, j and k form a *consistent triplet* if and only if the input relative rotations satisfy

$$d_{\text{chord}}(\mathbf{R}_{ij}^{\text{in}}, \mathbf{R}_{ik}^{\text{in}}\mathbf{R}_{kj}^{\text{in}}) < \epsilon, \tag{8.7}$$

  where ϵ is called a *loop threshold*. A triplet that satisfies Eq. (8.7) under small ϵ is described as "strong", and one that does it under relatively large ϵ is described as "weak". If a triplet contains one or more outlier edges, it is most likely to be inconsistent and fail to meet Eq. (8.7).

- **Number of triplet supports:** Suppose that a base node has several non-family neighbors, including node i. The number of *triplet supports* of neighbor i refers to the number of consistent triplets formed by the base node, node i and another neighbor of the base node. A simple example is illustrated in Fig. 8.1.

8.4 Method

The proposed method consists of three steps:

1. Robust initialization of the absolute rotations by building a spanning tree in a hierarchical manner. If the number of inlier matches is known for all edges, we can optionally incorporate it in the initialization.

2. Filtering the edges that do not conform to the initial solution to remove as many outliers as possible.

3. Iterative local refinement using nonlinear optimization.

To make the initialization step easier to understand, we first describe the simplified version in Section 8.4.1 and then the full version in Section 8.4.2. We explain the edge filtering and the local refinement in Section 8.4.3 and 8.4.4, respectively.

8.4.1 Hierarchical initialization (simplified version)

We initialize the absolute rotations by constructing a spanning tree of the graph. As we expand the tree incrementally, we want to avoid as many outlier edges as possible, so we start adding the most reliable edges first. In our method, there are two modes of tree expansion: (1) based on the triplet support, or (2) via single rotation averaging.

First, we set a certain integer threshold s (called a *support threshold*) and check if the base node has any non-family neighbors with s or more triplet supports. If so, we add these

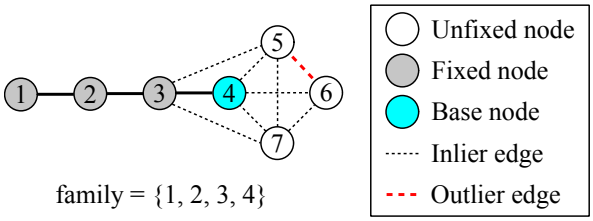


Figure 8.1: In this example, the base node (4) has three non-family neighbors (5, 6 and 7). We check the triplet consistency (Eq. (8.7)) without directly inferring the outlier edge: Node 5 has two triplet supports, *i.e.*, (3, 4, 5) and (4, 5, 7), 6 has one support, *i.e.*, (4, 6, 7), and 7 has three supports, *i.e.*, (3, 4, 7), (4, 5, 7) and (4, 6, 7).

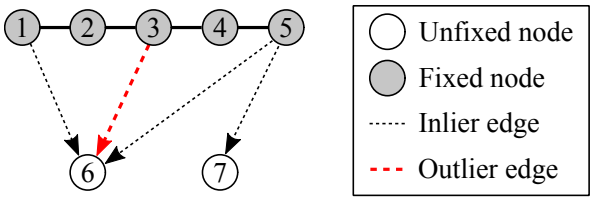


Figure 8.2: Here, no matter which family member is chosen as the base node, we cannot form a consistent triplet. In this case, we let every family member vote for their non-family neighbors and add the one with the most votes (node 6) to the family. Among the candidate rotations propagated from node 1, 3 and 5, we choose the one that is closest to their robust average (obtained using (Lee and Civera, 2020*b*)).

neighbors to the family and obtain their rotations $(\mathbf{R}_{\mathrm{N}}^{\mathrm{est}})$ by propagating from the base node $(\mathbf{R}_{\mathrm{B}}^{\mathrm{est}})$, *i.e.*,

$$\mathbf{R}_{\mathrm{N}}^{\mathrm{est}} \leftarrow \mathbf{R}_{\mathrm{NB}}^{\mathrm{in}} \mathbf{R}_{\mathrm{B}}^{\mathrm{est}}. \tag{8.8}$$

For example, if $s = 2$ in Fig. 8.1, we would add node 5 and 7 in the family, but not 6. If all non-family neighbors of the family have fewer than s triplet supports, we update $s \leftarrow s - 1$ and, for the next base node, choose the family member that has the most non-family neighbors with s or more supports. We repeat the same propagation process afterwards.

Another way to expand the tree is to add a node via single rotation averaging when all non-family neighbors of the family have zero triplet support. In this case, we let every family member vote for their non-family neighbors, and the one with the most votes is added to the family. This node also becomes the next base. To determine its rotation, we first obtain the candidate rotations by propagating from the family nodes that voted for it. Then, we average these rotations using the robust single rotation averaging method in (Lee and Civera, 2020*b*). Finally, the candidate rotation that is closest to the result is assigned to the node. Fig. 8.2 shows an example.

At each iteration, our initialization algorithm decides between the two aforementioned modes of tree expansion. We first try expanding based on the triplet support by adapting the support threshold s, and when $s = 0$, we expand the tree via voting and single rotation averaging. Every time a node is added to the family, we reset s to the initial value. Alg. 7 summarizes the procedure and Fig. 8.3 shows a toy example.

Algorithm 7: Hierarchical Initialization (simplified)

$s \leftarrow s_{\text{init}}$, `family` $\leftarrow \{\}$, `newFamily` $\leftarrow \{\}$;
Add the node with the most neighbors to `family` and `newFamily`, and set its rotation to identity;
while *not all nodes are in* `family` **do**
    while `newFamily` *is not empty* **do**
        Choose a member of `newFamily` as the base node and remove it from `newFamily`;
        Propagate away from the base node to its non-family neighbors using Eq. (8.8), and add those with s or more triplet supports to `family` and `newFamily`;
        if *at least one node is added to* `newFamily` **then**
            $s \leftarrow s_{\text{init}}$;
    For the next base node, choose the family member that has the most non-family neighbors with s or more supports;
    if *the base node has at least one non-family neighbor with* s *or more supports* **then**
        Add the base node to `newFamily`.
    else
        $s \leftarrow s - 1$;
    if $s = 0$ **then**
        Let every family member vote for their non-family neighbors, add the one with the most votes to `family` and `newFamily`, and set its rotation via single rotation averaging (see Fig. 8.2);
        $s \leftarrow s_{\text{init}}$;

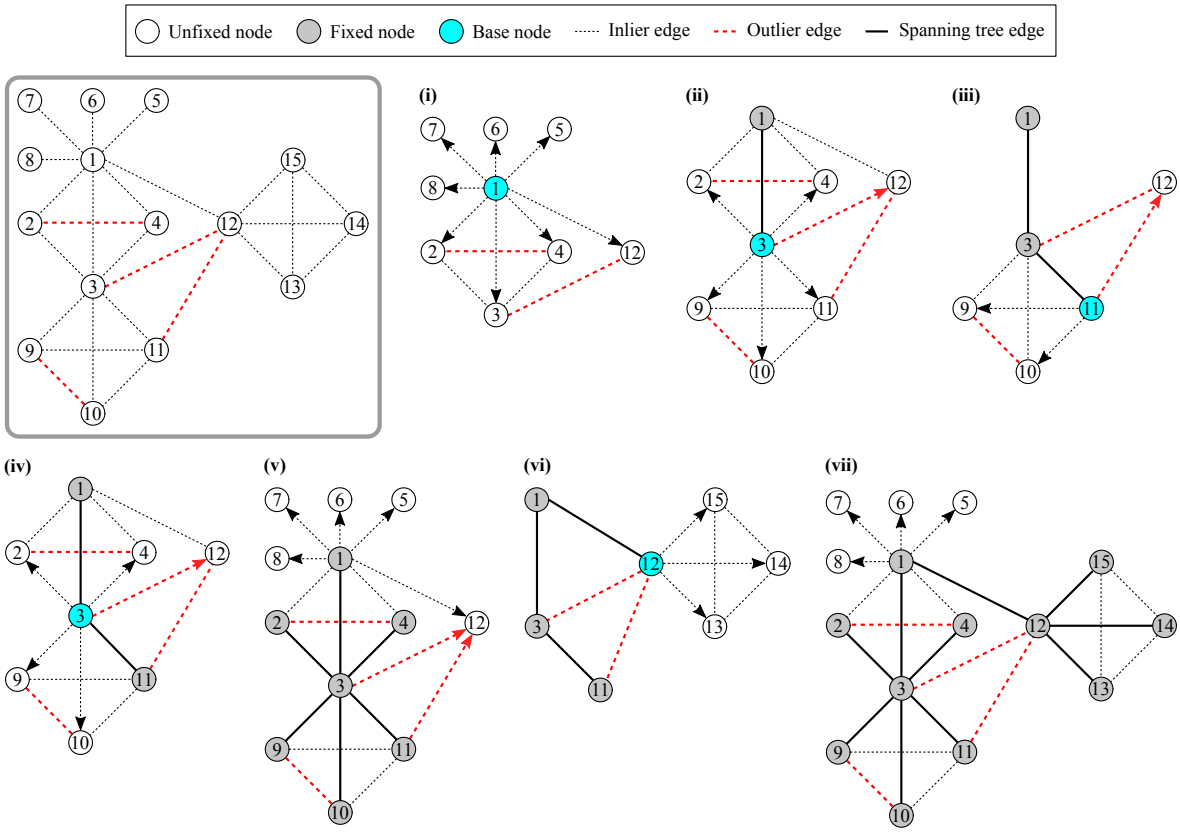


Figure 8.3: **[Top left]** A toy example. We show the steps of our initialization algorithm with $s = 2$ and a fixed loop threshold.

(i) First, we choose the node with the most neighbors (node 1) as the base node and set its rotation to identity. This node is the first member of the family. By propagating away from this node using Eq. (8.8), its neighbors get tentative rotations. For each neighbor, we count the number of triplet supports (*i.e.*, the number of other neighbors supporting it), and if it has s or more supports, we add it to the family. In this example, node 3 is supported by two other neighbors (node 2 and 4), and it is the only one added to the family. Node 3 becomes the next base node, and we fix its rotation. Also, the edge (1, 3) turns into a spanning tree edge.

(ii) We repeat the same process by propagating away from the new base node (node 3). The only non-family neighbor that has s or more supports is node 11, so we fix its rotation, add it to the family, and select it as the next base node.

(iii) We propagate away from the new base node (node 11), but no neighbor has enough supports. In this case, we update $s \leftarrow s - 1$ and check for each family member how many neighbors have s or more supports: node 1 has two (node 2 and 4), node 3 has four (node 2, 4, 9, 10), and node 11 has two (node 9 and 10). Since node 3 has the most, it becomes the next base node. Note that in the full version of the algorithm, we do this counting as soon as the base node changes and store the results for reuse (more details in Section 8.4.2).

(iv) We propagate away from the new base node (node 3), and the non-family neighbors with s (= 1) supports are node 2, 4, 9 and 10. These four nodes are added to the family, and their rotations are fixed.

(v) With node 1–4 and 9–11 in the family, none of their non-family neighbors has a single support. In this case, each family member votes for their non-family neighbors, and the one with the most votes is added to the family. This node (node 12) also becomes the next base node. To determine its rotation, we first average the candidate rotations propagated from node 1, 3 and 11 using (Lee and Civera, 2020*b*). Then, the candidate rotation that is closest to the result is assigned to node 12, and the corresponding edge becomes a spanning tree edge. In this example, let us suppose that it is the edge (1, 12).

(vi) Every time a node is added to the family, we reset s to the initial value ($s \leftarrow 2$). Afterwards, we repeat the process of propagating away from the base node and adding the neighbors with s or more supports to the family. In this example, node 12 has three non-family neighbors (node 13, 14, 15) and they all have s supports. Therefore, all three of them are added to the family and their rotations are fixed.

(vii) With node 1–4 and 9–13 in the family, none of their non-family neighbors has s (= 2) supports. We update $s \leftarrow s - 1$ and check again, but none has a single support. Now, as in Step (v), we let every family member vote for their non-family neighbors, and add the one with the most votes. Repeating this procedure adds node 5–8 to the family one by one. Finally, all nodes are in the family and their rotations are fixed. The algorithm returns the estimated rotations of all nodes.

8.4.2 Hierarchical initialization (full version)

The simplified algorithm described in the previous section constructs a spanning tree by adding the most supported edges first. In the full version, we consider two more aspects: the loop threshold ϵ in Eq. (8.7), and optionally, the number of valid 2D-2D correspondences. We highlight the differences between the two versions in Alg. 8.

In the simplified version, the consistency of a triplet depends entirely on a single threshold ϵ we set. In the full version, we set multiple thresholds (ϵ_1, ϵ_2, …, ϵ_m in ascending order) and adaptively switch between them. Specifically, we start from the smallest (the strictest) threshold and gradually move on to larger (less strict) thresholds. As a result, the following hierarchy is established:

1. Neighbor nodes with many triplet supports under small ϵ_i are added to the family first.
2. Those with many supports under large ϵ_i are added next.
3. Those with few supports under small ϵ_i are added next.
4. Those with few supports under large ϵ_i are added last.

Another change from the simplified version is that we store the number of supported neighbors each time we try propagating away from the base node. This data is stored in a *supported neighbors table* (SN table), a 3D array whose dimensions correspond to the base node index, the threshold index, and the number of triplet supports. We organize this table such that the entry at position (x, y, z) corresponds to the number of non-family neighbors of base node x that have z or more supports under the y-th threshold ϵ_y. Note that each time we update the SN table for the current base node, we update it for all $y = 1, 2, \ldots, m$ and $z = 1, 2, \ldots, s_{\text{init}}$. This is done in line 10 of Alg. 8.

The advantage of maintaining this table is that we can reuse it to promptly find the node with the most number of supported neighbors for any given s and ϵ (line 13 of Alg. 8). This operation is necessary when we have to choose the next base node after s is decremented. Although the data in the SN table may sometimes be outdated (because some non-family neighbors can turn into family members later), we can at least avoid having to evaluate the neighbors of all family members repeatedly (*i.e.*, line 9 of Alg. 7).

Our approach can also seamlessly integrate the knowledge of the number of valid 2D-2D correspondences. This can be done with minimal modification of Alg. 8: Let d_1, d_2, …, d_k (in descending order) be some thresholds we set for the number of valid 2D-2D correspondences. Then, we run the outer loop (line 6–23) while pretending that all edges whose valid correspondences are fewer than d_1 do not exist. When the number of total votes becomes zero in line 22, we reset s, ϵ and the SN table to the initial state, switch the correspondence threshold to the next one (d_2) and continue. This process ensures that the edges with very few valid correspondences are added last.

Algorithm 8: Hierarchical Initialization (full version)

1 $s \leftarrow s_{\text{init}}$, `family` $\leftarrow \{\}$, `newFamily` $\leftarrow \{\}$;
2 Add the node with the most neighbors to `family` and `newFamily`, and set its rotation to identity;
3 Determine the loop thresholds $\epsilon_1, \epsilon_2, \ldots, \epsilon_m$;
4 $i \leftarrow 1,\ \epsilon \leftarrow \epsilon_i$;
5 `snTable` $\leftarrow$ Zero 3D array of dimension $n \times m \times s$;
(n is #nodes, m is #loop thresholds, s is a support threshold)
6 **while** *not all nodes are in* `family` **do**
7 **while** `newFamily` *is not empty* **do**
8 Choose a member of `newFamily` as the base node and remove it from `newFamily`;
9 Propagate away from the base node to its non-family neighbors using Eq. (8.8) and add those with s or more triplet supports to `family` and `newFamily`;
10 Update `snTable` for the base node;
11 **if** *at least one node is added to* `newFamily` **then**
12 $s \leftarrow s_{\text{init}}$, $i \leftarrow 1,\ \epsilon \leftarrow \epsilon_i$;
13 In `snTable`, find the family member that has the most non-family neighbors with s or more triplet supports under the current threshold ϵ. Choose it as the base node;
14 **if** *the base node has at least one non-family neighbor with* s *or more supports* **then**
15 Add the base node to `newFamily`.
16 **else**
17 **if** $i < m$ **then**
18 $i \leftarrow i+1,\ \epsilon \leftarrow \epsilon_i$;
19 **else**
20 $s \leftarrow s-1,\ i \leftarrow 1,\ \epsilon \leftarrow \epsilon_i$;
21 **if** $s = 0$ **then**
22 Let every family member vote for their non-family neighbors, add the one with the most votes to `family` and `newFamily`, and set its rotation via single rotation averaging (see Fig. 8.2);
23 $s \leftarrow s_{\text{init}}$, $i \leftarrow 1,\ \epsilon \leftarrow \epsilon_i$;

Implementation details:

1. In line 5 of Alg. 7 and line 8 of Alg. 8, if `newFamily` has multiple members, we choose the one with the most neighbors as the base node. This is because we want to add well-connected nodes first to minimize drift.

2. In all of our experiments in this chapter, we fix $s_{\text{init}} = 10$, $d_1 = 5$ and $d_2 = 0$.

3. For good performance, the loop thresholds should reflect the noise level of the inlier edges. To this end, we use a simple heuristic method to determine their values in line 3 of Alg. 8: For each edge (i, j), we sample at most 10 common neighbors of node i and j, forming up to 10 triplets (i, j, k). We compute the loop errors (8.7) of all triplets from all edges and collect only those below 1. Then, we set the loop thresholds ϵ_1, ϵ_2 and ϵ_3 to the 10th, 20th and 30th percentile of the collected errors.

8.4.3 Edge filtering

Having obtained an initial solution ($\mathbf{R}_i^{\text{est}}$ for $i = 1, 2, \ldots, n$) from the spanning tree in Section 8.4.2, we next filter potential outlier edges in the full rotation graph before optimizing the solution. This is done by checking whether or not each edge conforms to the initial solution. Specifically, we consider edge (j, k) as an outlier and exclude it from the further operations if the following condition is met:

$$d_{\text{chord}}\left(\mathbf{R}_{jk}^{\text{in}}, \mathbf{R}_j^{\text{est}}\mathbf{R}_k^{\text{est}\top}\right) > \tau, \tag{8.9}$$

where τ is some threshold (we set $\tau = 1$ in this work).

While this filtering step often enhances the robustness for moderate outlier ratios (< 0.3), we found that it sometimes worsens the accuracy for higher outlier ratios. Therefore, we skip this step when we deem the outlier ratio to be too high. In practice, we assume that this is the case when the median of the loop errors from all sampled triplets is larger than 1 (see the implementation detail 3 of Section 8.4.2).

8.4.4 Local refinement

Given the initial solution and the filtered constraints ($\mathbf{R}_{jk}^{\text{in}}$ for $(j, k) \in$ filtered edges), we perform an iterative local refinement using the optimization method proposed in (Chatterjee and Govindu, 2018). In the following, we briefly summarize this method.

The goal here is to find the optimal updates such that the updated solution fits the constraints better, *i.e.*,

$$\mathbf{R}_{jk}^{\text{in}} = \left(\mathbf{R}_j^{\text{est}}\Delta\mathbf{R}_j\right)\left(\mathbf{R}_k^{\text{est}}\Delta\mathbf{R}_k\right)^\top. \tag{8.10}$$

Rearranging this and taking the Log (8.1) of both sides gives

$$\mathrm{Log}\left(\mathbf{R}_j^{\text{est}\top}\mathbf{R}_{jk}^{\text{in}}\mathbf{R}_k^{\text{est}}\right) = \mathrm{Log}\left(\Delta\mathbf{R}_j\Delta\mathbf{R}_k^\top\right). \tag{8.11}$$

Assuming that the updates are small, we can use the approximation in Eq. (8.6) on the right-hand side and obtain

$$\mathrm{Log}\left(\mathbf{R}_j^{\text{est}\top}\mathbf{R}_{jk}^{\text{in}}\mathbf{R}_k^{\text{est}}\right) \approx \Delta\mathbf{u}_j - \Delta\mathbf{u}_k, \tag{8.12}$$

where $\Delta\mathbf{u}_j$ and $\Delta\mathbf{u}_k$ are the rotation vectors of $\Delta\mathbf{R}_j$ and $\Delta\mathbf{R}_k$, respectively. Since the left-hand side of Eq. (8.12) is known, stacking these equations for all filtered edges results in a linear system of equations, which we solve using a linear algebra library. We update the rotations, *i.e.*, $\mathbf{R}_i^{\text{est}} \leftarrow \mathbf{R}_i^{\text{est}}\Delta\mathbf{R}_i$ for all i, plug them back into Eq. (8.12) and repeat the same process until convergence. In practice, we carry out the optimization using the IRLS method with the $\ell_{\frac{1}{2}}$ loss function, as in (Chatterjee and Govindu, 2018). Also, to reduce the total number of arithmetic operations, all rotations (both absolute and relative) are parameterized

as quaternions. For more details, we refer to the original work (Chatterjee and Govindu, 2018).

8.5 Results

We compare our method with the following methods: R-GoDec[1] (Arrigoni et al., 2014, 2018), Eig-IRLS[2] (Arrigoni et al., 2018), IRLS-$\ell_{\frac{1}{2}}$[3] (Chatterjee and Govindu, 2018), MPLS[4] (Shi and Lerman, 2020) and Hybrid RA[5] (Chen et al., 2021). Since the implementation of the view graph filtering (VGF) in Hybrid RA is not publicly available, we reproduced this part by ourselves. Note that this part is only applicable if 2D-2D correspondences are given for all edges. All methods are implemented in MATLAB, except Hybrid RA which is written in C++. We run all methods on a laptop with Intel's 4th Gen i7 CPU (2.8 GHz).

We evaluate the accuracy using two error metrics:

$$\theta_1 = \min_{\mathbf{R}_{\text{align}}} \frac{1}{n} \sum_{i=1}^{n} d\left(\mathbf{R}_i^{\text{gt}}, \mathbf{R}_i^{\text{est}} \mathbf{R}_{\text{align}}\right), \tag{8.13}$$

$$\theta_2 = \min_{\mathbf{R}_{\text{align}}} \sqrt{\frac{1}{n} \sum_{i=1}^{n} d\left(\mathbf{R}_i^{\text{gt}}, \mathbf{R}_i^{\text{est}} \mathbf{R}_{\text{align}}\right)^2}. \tag{8.14}$$

They respectively represent the optimal mean and RMS error after aligning the estimated rotations to the ground truth. The rotation $\mathbf{R}_{\text{align}}$ in Eq. (8.13) and (8.14) can be obtained by solving the single rotation averaging problem under the L_1 and L_2 norm, respectively (Hartley et al., 2011, Hartley, Kahl, Olsson and Seo, 2013).

8.5.1 Synthetic data

For a controlled study of various factors, we run Monte Carlo simulations in multiple settings: We generate n random rotations in a circular order and obtain the relative rotation of $p\%$ of all possible pairs. The edges are established as follows: First, we connect all successive nodes (*i.e.* node 1&2, 2&3, ..., n&1). Then, we connect those separated by one node (*i.e.* node 1&3, 2&4, ..., $n-1$&1, n&2), and afterwards, those separated by two nodes, three nodes, and so forth. We continue this process until $p\%$ are connected in total. This leads to all nodes being connected to their local neighbors in a sliding window fashion. Next, we turn $q\%$ of the edges into outliers, *i.e.*, random relative rotations. We exclude the edges between successive nodes, so that every node gets at least two inlier edges. Finally, all edges are perturbed by $\mathcal{N}(0, \sigma^2)$ and their order is randomized. These edges are used as input to the

[1] `http://www.diegm.uniud.it/fusiello/demo/gmf/`
[2] The code was kindly provided by the authors of (Arrigoni et al., 2018).
[3] `http://www.ee.iisc.ac.in/labs/cvl/research/rotaveraging/`
[4] `https://github.com/yunpeng-shi/MPLS`
[5] `https://github.com/AIBluefisher/GraphOptim`

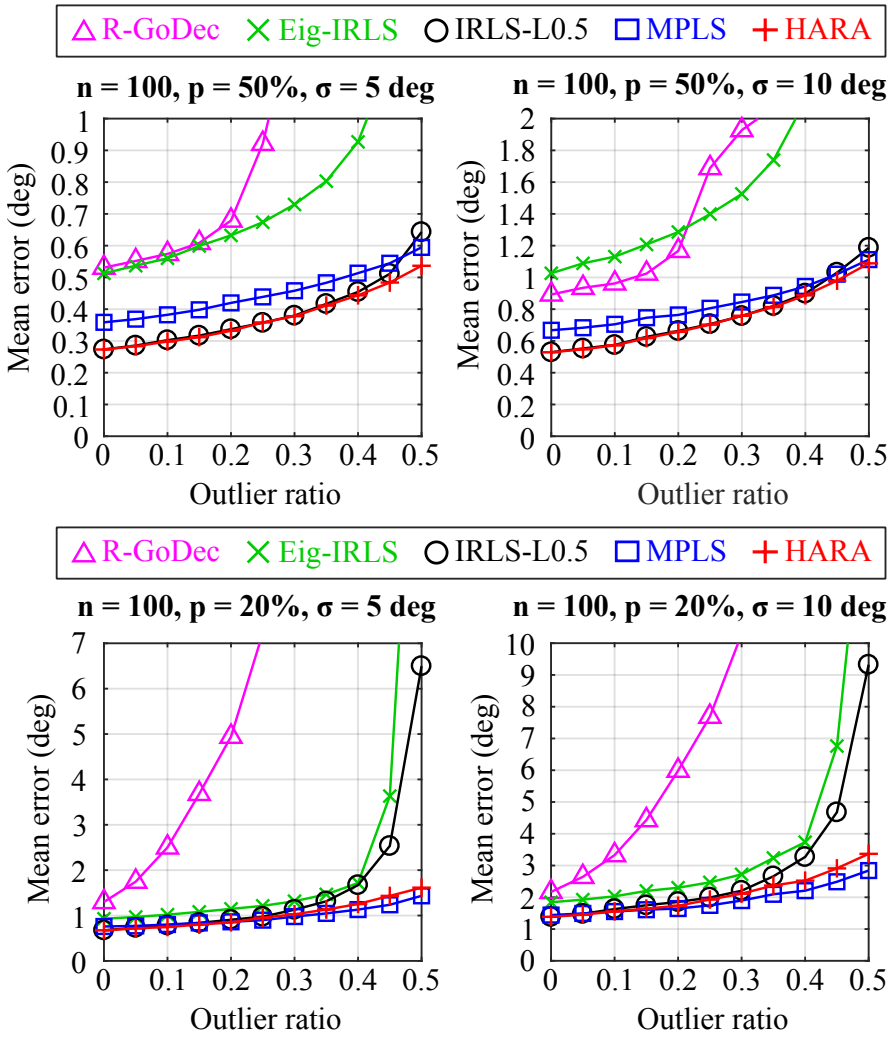


Figure 8.4: Simulation results (100 rotations): We plot the optimal mean errors, θ_1 in Eq. (8.13). For dense graphs ($p = 50\%$), IRLS-$\ell_{\frac{1}{2}}$, MPLS and HARA perform similarly well. For sparse graphs ($p = 20\%$), MPLS and HARA are more robust to outliers than the rest, with MPLS being slightly better at high outlier ratios.

rotation averaging algorithms. The simulation is configured by setting $\{n,\ p,\ q,\ \sigma\}$ to one of the following values: $n = \{100, 200\}$ rotations, $p = \{50, 20\}\%$, $q = \{0, 5, 10, ..., 50\}\%$, $\sigma = \{5, 10\}$ deg. For each setting, we generate 100 independent datasets.

Fig. 8.4 and 8.5 present the results for 100 and 200 rotations, respectively. We see that MPLS and HARA are the two best performing methods, especially at high outlier ratios.

8.5.2 Real data

Without using the number of inlier feature matches:

We evaluate the performance on the following real-world datasets: 1DSfM datasets[6] (Wilson and Snavely, 2014), 'Notre Dame 715' (ND2) dataset[3] (Chatterjee and Govindu, 2018), 'Acropolis' (ACP), 'Arts Quad' (ARQ) and 'San Francisco' (SNF) datasets[7] (Crandall et al., 2011). As in (Chatterjee and Govindu, 2018), only those cameras whose ground truth is

[6] http://www.cs.cornell.edu/projects/1dsfm/
[7] http://vision.soic.indiana.edu/projects/disco/

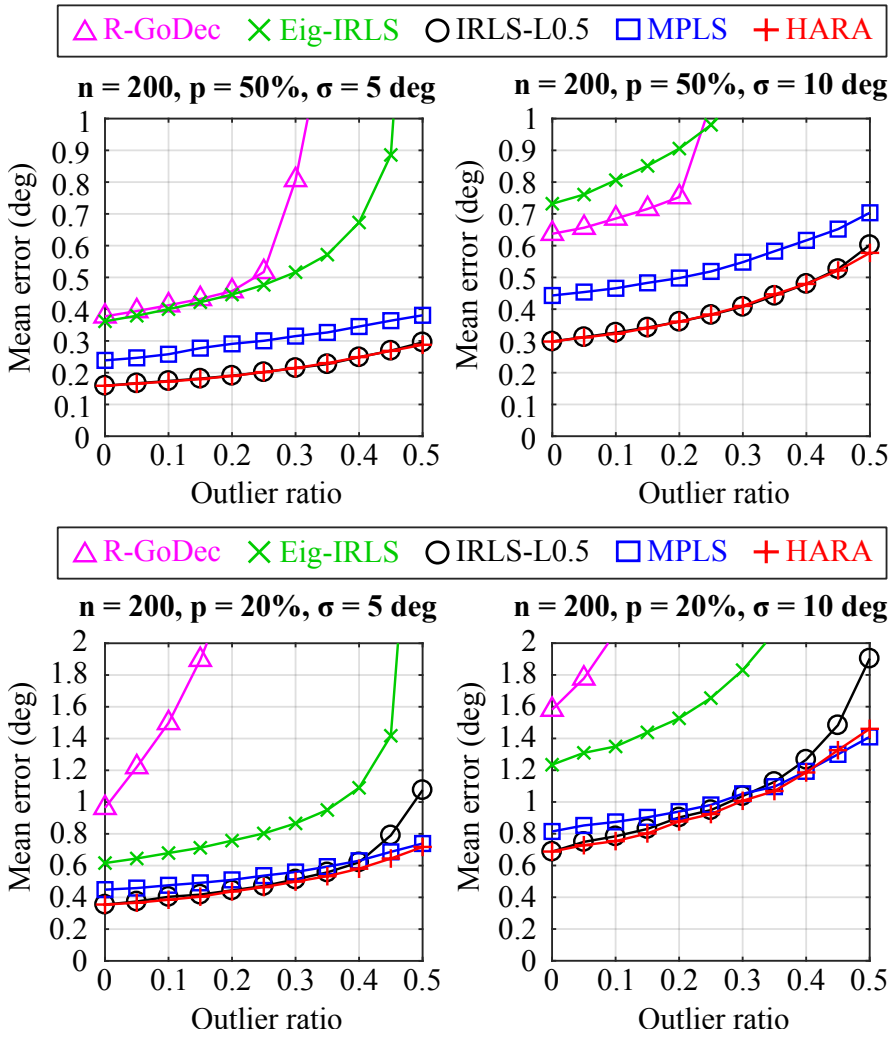


Figure 8.5: Simulation results (200 rotations): We plot the optimal mean errors, θ_1 in Eq. (8.13). IRLS-$\ell_{\frac{1}{2}}$, MPLS and HARA perform similarly well, except that for sparse graphs ($p = 20\%$), IRLS-$\ell_{\frac{1}{2}}$ is outperformed by the other two at high outlier ratios.

available are used to evaluate the accuracy, even though those without the ground truth are still included in the input for rotation averaging. Table 8.1 reports the results. It shows that in most cases HARA achieves state-of-the-art accuracy at a comparable speed.

Using the number of inlier feature matches:

For this experiment, we only use the 1DSfM dataset[6] (Wilson and Snavely, 2014), as the other datasets do not provide the 2D-2D correspondences. The feature matches are only available for those cameras with the ground truth, so we disregard the rest. To check the validity of the correspondences, we put a threshold (0.01) on the sine of the L_1-optimal angular reprojection error (Lee and Civera, 2019*a*, 2020*a*). Table 8.2 presents the results. It shows that HARA achieves state-of-the-art results, with or without incorporating the number of inlier matches.

8.5.3 Ablation study

We perform an ablation study to see how each component of HARA contributes to the final accuracy. We compare three different variations of HARA against the baseline version:

1. HARA without the local refinement (Section 8.4.4),
2. HARA without the edge filtering (Section 8.4.3),
3. HARA without the triplet-based propagation (Alg. 8 line 7–21): That is, the initial solution is obtained via a series of voting + single rotation averaging only.

Fig. 8.6 and Table 8.3 present the results on the synthetic and the real datasets, respectively. These results clearly show that the best performance can be achieved by utilizing all the components.

8.6 Limitations

The main limitation of our method is that it is sensitive to the parameters we set, especially the loop thresholds. Currently, we determine their values using a simple heuristic based on the sampled loop errors (Section 8.4.2). We noticed that, in some of the real datasets, a small change in this heuristic introduces a non-negligible fluctuation in the initialization accuracy. In future work, we plan to replace this heuristic with a more robust and reliable method.

8.7 Conclusions

We presented HARA, a hierarchical approach for robust multiple rotation averaging. For robust initialization of the rotation graph, we incrementally build a spanning tree based on a hierarchy of triplet support. That is, the edges supported by many strong triplets are added in the tree sooner than those with fewer or weaker triplets. This approach significantly reduces the influence of outliers on the initial solution, allowing us to filter outliers prior to nonlinear optimization. Also, we showed that we can optionally integrate the knowledge of the number of valid 2D-2D correspondences into our approach. An extensive evaluation demonstrates that HARA achieves state-of-the-art results.

Datasets			R-GoDec			Eig-IRLS			IRLS-$\ell_{\frac{1}{2}}$			MPLS*			Hybrid RA w/o VGF†			HARA w/o #inlier matches		
Name	#views	%edges	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time
ALM	627	49.5%	6.3	16.3	4s	3.9	12.2	21s	4.2	12.6	27s	3.7	12.1	29s	4.3	12.7	–	**3.5**	**11.5**	45s
ELS	247	66.8%	4.1	10.7	1s	3.3	11.5	3s	2.9	10.3	4s	2.8	10.9	7s	3.1	10.5	–	**2.1**	**7.4**	7s
GDM	742	17.5%	51.3	64.8	17s	65.8	74.2	29s	**37.5**	62.3	12s	40.7	68.7	74s	44.7	**60.0**	–	43.8	72.5	26s
MDR	394	30.7%	10.2	18.5	1s	10.9	22.4	5s	7.0	17.1	4s	5.2	14.7	5s	6.4	16.2	–	**4.8**	**14.5**	14s
MND	474	46.8%	6.2	18.4	3s	1.9	11.2	7s	1.5	7.4	10s	1.2	3.9	9s	1.5	6.9	–	**1.1**	**2.1**	20s
ND1	553	68.1%	5.2	15.5	6s	3.6	14.8	16s	3.5	14.6	28s	2.7	13.5	27s	3.5	14.7	–	**1.6**	**6.3**	55s
NYC	376	29.3%	6.4	9.9	6s	3.8	8.2	5s	3.0	**7.0**	3s	3.0	8.2	7s	3.2	7.4	–	**2.9**	7.7	10s
PDP	354	39.5%	11.4	22.0	2s	4.0	9.3	8s	4.1	8.1	4s	3.5	8.2	4s	5.3	10.4	–	**3.4**	**7.4**	9s
PIC	2508	10.2%	24.8	40.0	150s	81.0	91.2	687s	6.8	18.6	467s	4.6	14.6	295s	7.0	20.1	–	**4.4**	**13.1**	289s
ROF	1134	10.9%	12.6	19.5	61s	3.4	10.4	52s	3.1	10.2	12s	2.8	10.0	13s	3.1	9.2	–	**2.7**	**8.5**	31s
TOL	508	18.5%	6.4	13.1	8s	4.5	10.7	10s	**3.9**	**9.0**	2s	4.0	9.4	6s	4.4	10.5	–	4.3	10.0	13s
TFG	5433	4.6%	42.1	54.2	722s	59.4	67.1	833s	3.6	**9.8**	976s	4.5	10.8	1945s	15.1	17.8	–	**3.5**	10.7	925s
USQ	930	5.9%	12.0	23.7	27s	6.7	12.8	22s	9.3	22.2	10s	6.3	14.7	11s	9.3	21.7	–	**6.0**	**12.3**	9s
VNC	918	24.6%	16.1	36.9	25s	8.6	28.4	27s	8.3	27.5	28s	6.2	18.2	53s	8.4	27.2	–	**6.1**	**18.1**	47s
YKM	458	26.5%	6.1	11.3	7s	3.8	9.4	8s	3.5	8.4	3s	3.5	9.2	7s	3.5	8.4	–	**3.0**	**6.9**	16s
ND2	715	25.3%	2.7	10.0	10s	1.2	4.0	13s	**1.1**	**3.5**	13s	**1.1**	4.0	12s	**1.1**	**3.5**	–	1.3	5.5	23s
ACP	463	10.7%	**0.8**	**1.2**	6s	1.1	1.7	4s	1.2	1.7	1s	1.4	2.0	2s	1.2	1.7	–	1.2	1.7	7s
ARQ	5530	1.5%	29.7	56.5	1111s	70.1	79.7	4894s	4.0	7.1	173s	**3.2**	**6.3**	118s	3.9	6.9	–	3.6	6.8	137s
SNF	7866	0.3%	Out of memory			77.3	87.3	3.8h	**3.6**	**4.2**	180s	4.4	5.5	154s	4.3	6.2	–	**3.6**	**4.2**	44s

θ_1 (deg): Optimal mean error in Eq. (8.13), θ_2 (deg): Optimal RMS error in Eq. (8.14), %edges = #edges/#possible pairs of views in %.

*Due to the non-deterministic nature of MPLS (Shi and Lerman, 2020), we report the median of five independent runs.

†The computation times of Hybrid RA (Chen et al., 2021) are not included for comparison, since it is the only method implemented in C++.

Table 8.1: Results on the real datasets **without** the knowledge of the 2D-2D correspondences: For all datasets, HARA gives either better or comparable results to the state of the art. Interestingly, for the SNF dataset, it takes substantially less time than the rest. The results of MPLS (Shi and Lerman, 2020) are mostly competitive with ours, except for the ELS, ND1, TFG and SNF datasets where HARA performs noticeably better. We run Hybrid RA (Chen et al., 2021) without view graph filtering because this requires the number of valid 2D-2D correspondences. As a result, this method does not provide much gain in accuracy compared to IRLS-$\ell_{\frac{1}{2}}$ (Chatterjee and Govindu, 2018), even though it performs an additional global optimization prior to local refinement. In fact, Hybrid RA performs much worse than IRLS-$\ell_{\frac{1}{2}}$ on the TFG dataset.

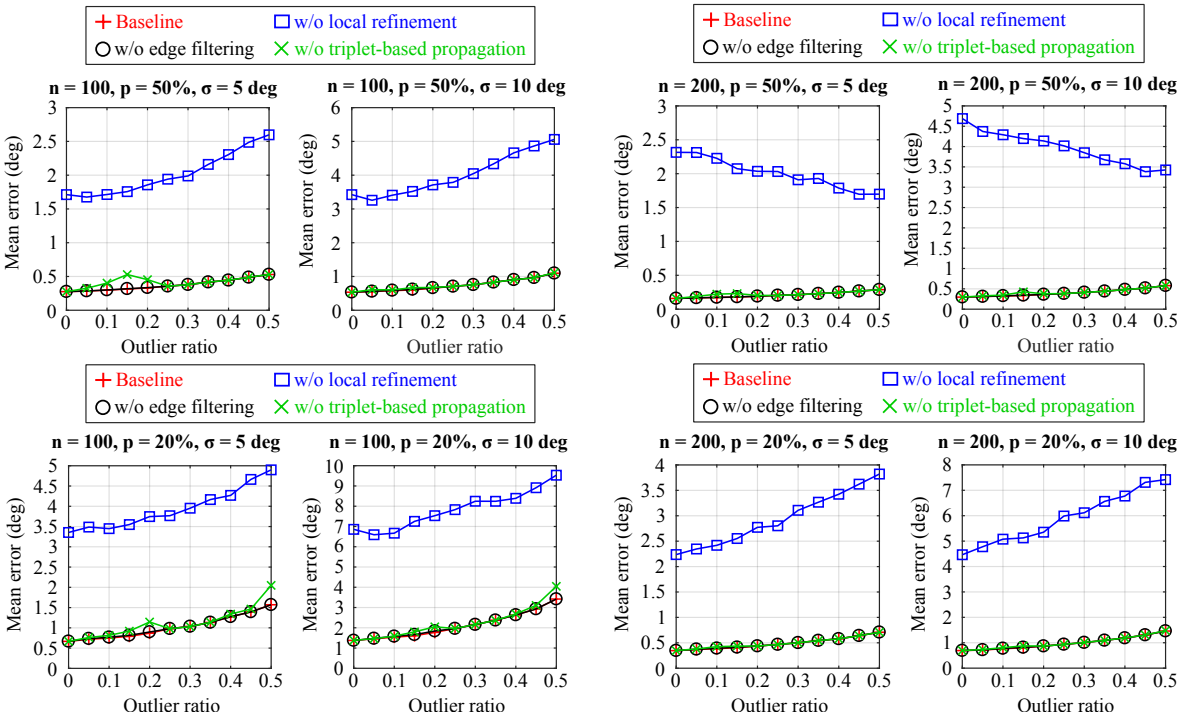
Datasets			IRLS-$\ell_{\frac{1}{2}}$			MPLS*			Hybrid RA w/o VGF†			Hybrid RA with VGF†			HARA w/o #inlier matches			HARA with #inlier matches		
Name	#views	%edges	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time
ALM	577	58.4%	4.0	12.4	28s	3.7	12.0	24s	4.3	12.8	–	**3.0**	**10.2**	–	3.5	11.5	41s	3.4	11.0	40s
ELS	227	78.0%	2.8	10.1	2s	3.0	11.7	6s	3.0	9.9	–	2.1	7.1	–	2.1	7.2	8s	**1.8**	**4.8**	6s
GDM	677	20.9%	37.4	62.2	8s	40.7	68.6	81s	**34.5**	**55.3**	–	39.9	68.7	–	44.1	72.5	23s	44.3	72.8	16s
MDR	341	40.7%	6.7	16.7	4s	5.1	14.4	4s	6.3	15.8	–	**4.4**	**13.1**	–	4.8	14.5	13s	4.8	14.8	11s
MND	450	51.8%	1.5	7.4	6s	1.2	3.9	8s	1.5	6.9	–	**1.1**	2.2	–	**1.1**	**2.1**	21s	**1.1**	**2.1**	18s
ND1	553	68.1%	3.5	14.6	29s	2.8	13.6	30s	3.5	14.7	–	1.7	6.1	–	1.6	6.3	61s	**1.5**	**5.9**	44s
NYC	332	37.4%	3.1	7.1	3s	3.1	8.2	7s	3.4	7.8	–	3.0	7.1	–	2.9	7.8	8s	**2.6**	**5.8**	7s
PDP	338	43.3%	4.1	8.2	4s	3.5	8.2	4s	5.2	10.3	–	**3.1**	**6.4**	–	3.5	7.8	8s	3.3	6.6	7s
PIC	2152	13.4%	6.2	17.0	419	4.7	14.6	254s	6.3	18.6	–	4.3	12.1	–	4.1	**11.3**	269s	**4.0**	**11.3**	247s
ROF	1084	11.9%	3.1	10.2	16s	2.8	9.7	13s	3.1	9.4	–	**2.5**	**6.7**	–	2.7	8.7	30s	**2.5**	7.6	25s
TOL	472	21.4%	**3.9**	**8.9**	2s	4.0	9.4	4s	4.4	10.4	–	4.0	9.4	–	4.3	10.0	8s	4.0	**8.9**	11s
TFG	5058	5.3%	3.5	**8.9**	881s	5.3	11.1	1466s	4.0	9.8	–	5.3	11.8	–	3.5	10.1	948s	**3.4**	9.6	902s
USQ	789	7.9%	6.7	14.2	5s	6.2	12.9	7s	7.9	17.0	–	6.6	14.8	–	5.9	11.2	10s	**5.8**	**10.8**	9s
VNC	836	29.6%	8.4	27.5	32s	**6.2**	18.2	59s	8.4	27.1	–	6.3	**18.0**	–	**6.2**	18.1	54s	**6.2**	18.1	52s
YKM	437	29.1%	3.5	8.4	2s	3.6	9.4	4s	3.6	8.4	–	3.9	11.8	–	**3.0**	**6.9**	11s	3.4	11.2	12s

θ_1 (deg): Optimal mean error in Eq. (8.13), θ_2 (deg): Optimal RMS error in Eq. (8.14), %edges = #edges/#possible pairs of views in %.

*Due to the non-deterministic nature of MPLS (Shi and Lerman, 2020), we report the median of five independent runs.

†The computation times of Hybrid RA (Chen et al., 2021) are not included for comparison, since it is the only method implemented in C++.

Table 8.2: Results on the real datasets **with** the knowledge of the number of valid 2D-2D correspondences: The best performing methods are Hybrid RA (Chen et al., 2021) (with VGF) and HARA with and without using #inlier feature matches. All three of these methods give competitive results, but on the TFG and USQ datasets, HARA outperforms Hybrid RA by a noticeable margin.



(a) 100 rotations (b) 200 rotations

Figure 8.6: Ablation study on the synthetic dataset.

Datasets			HARA (baseline)			w/o local refinement			w/o edge filtering			w/o triplet-based propagation		
Name	#views	%edges	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time
ALM	627	49.5%	**3.5**	**11.5**	41s	4.4	12.6	26s	4.1	12.4	51s	3.9	12.3	30s
ELS	247	66.8%	**2.1**	**7.4**	6s	2.6	7.8	5s	2.5	7.7	9s	3.0	11.6	4s
GDM	742	17.5%	43.8	72.5	21s	44.1	72.0	20s	**37.7**	**62.4**	33s	46.6	71.7	12s
MDR	394	30.7%	**4.8**	**14.5**	14s	6.8	15.2	13s	6.5	16.4	13s	7.7	20.2	5s
MND	474	46.8%	**1.1**	**2.1**	28s	1.6	2.8	26s	1.5	7.4	23s	1.4	7.5	11s
ND1	553	68.1%	**1.6**	**6.3**	48s	2.3	6.7	38s	3.2	12.4	55s	3.7	15.8	31s
NYC	376	29.3%	**2.9**	7.7	10s	3.3	8.0	9s	3.0	**7.0**	8s	3.3	8.6	6s
PDP	354	39.5%	**3.4**	**7.4**	7s	3.5	7.5	6s	4.0	8.0	10s	3.6	8.5	5s
PIC	2508	10.2%	**4.4**	**13.1**	279s	5.7	13.7	140s	5.5	14.5	437s	5.9	18.4	220s
ROF	1134	10.9%	**2.7**	8.5	31s	3.3	9.1	26s	3.0	8.6	30s	**2.7**	**7.9**	15s
TOL	508	18.5%	4.3	10.0	8s	4.6	10.3	8s	**4.0**	**9.2**	14s	4.7	11.6	4s
TFG	5433	4.6%	**3.5**	10.7	924s	5.5	11.6	325s	3.6	10.0	1049s	3.6	**9.7**	1014s
USQ	930	5.9%	**6.0**	12.3	8s	7.1	14.1	7s	7.3	14.7	11s	6.1	**11.4**	5s
VNC	918	24.6%	**6.1**	**18.1**	52s	6.6	18.6	40s	8.0	26.3	56s	8.2	28.3	32s
YKM	458	26.5%	**3.0**	6.9	17s	3.1	**6.5**	16s	3.5	8.4	14s	3.6	9.6	5s
ND2	715	25.3%	1.3	5.5	23s	1.7	5.5	19s	**1.1**	**3.5**	31s	1.3	5.3	17s
ACP	463	10.7%	**1.2**	**1.7**	6s	2.0	2.4	6s	**1.2**	**1.7**	4s	**1.2**	**1.7**	3s
ARQ	5530	1.5%	**3.6**	6.8	136s	5.1	8.1	104s	3.7	**6.6**	169s	4.4	11.2	98s
SNF	7866	0.3%	**3.6**	**4.2**	35s	4.8	6.7	32s	**3.6**	**4.2**	44s	**3.6**	**4.2**	17s

Table 8.3: Ablation study on the real datasets **without** the knowledge of the 2D-2D correspondences.

Chapter 9

L_{0+} Optimization: Application to Multiple Rotation Averaging

9.1 Introduction

In the previous chapter, we used the $L_{0.5}$ cost function proposed in (Chatterjee and Govindu, 2018) for the local refinement step of HARA. In this chapter, we propose to use a different cost function, namely the smoothed L_{0+} function introduced in (Peng et al., 2022). Compared to the commonly used Huber loss (Huber, 1964), L_1 or $L_{0.5}$ norm, this function places smaller weights on large residuals, thereby further reducing the influence of outliers. Also, unlike the truncated least squares (Blake and Zisserman, 1987) or Tukey's biweight function (Beaton and Tukey, 1974), it does not completely discard them. We show that by adopting the smoothed L_{0+} function in the local refinement step of HARA, we can further improve the results.

This chapter is organized as follows: In Section 9.2, we review the smoothed L_p cost function proposed in (Peng et al., 2022). We provide the probabilistic interpretations of this function in Section 9.3. In Section 9.4, we apply our method on HARA and evaluate its rotation averaging accuracy on the real datasets. Finally, conclusions are given in Section 9.5.

9.2 The Smoothed L_{0+} Cost Function

(Peng et al., 2022) proposed the following family of robust cost functions parameterized by two parameters, *i.e.*, p and c ($c > 0$):

$$\rho(r) = \begin{cases} \dfrac{r^2}{2} & \text{if } |r| < c \\ c^{2-p}\left(\dfrac{|r|^p}{p} - \dfrac{c^p}{p} + \dfrac{c^p}{2}\right) & \text{otherwise} \end{cases} \tag{9.1}$$

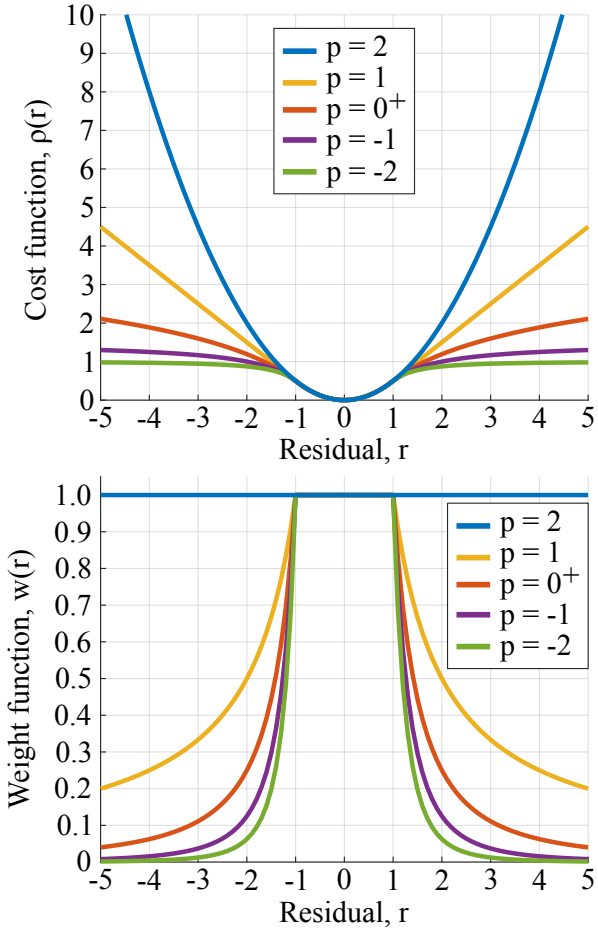


Figure 9.1: Robust cost functions defined by (9.1) with different values of p **(top)** and their corresponding weight functions **(bottom)**. In this example, the parameter c in (9.1) is set to be 1.

The parameter c represents the threshold below which the function behaves as the L_2 norm. Above this threshold, the function behaves as the L_p norm: the smaller the parameter p, the smaller the cost of large residuals. This behaviour is illustrated in Fig. 9.1. Note that when $p = 1$, this function is called the Huber loss (Huber, 1964). The weight function corresponding to (9.1) can be obtained as $w(r) = \frac{d\rho(r)/dr}{r}$ (Zhang, 1997), which gives

$$w(r) = \begin{cases} 1 & \text{if } |r| < c \\ c^{2-p}|r|^{p-2} & \text{otherwise.} \end{cases} \tag{9.2}$$

Now, we are specifically interested in a special case of (9.1) when $p \to 0^+$. Note that this is not the same as setting $p = 0$, because doing so would turn the second part of (9.1) into a constant term (if not ill-defined), leading to a truncated least squares function. In (Peng et al., 2022), it was noted that when $p \to 0^+$, the cost function (9.1) can be written as follows:

$$\rho(r) = \begin{cases} \dfrac{r^2}{2} & \text{if } |r| < c \\ c^2\left(\ln|r| - \ln c + \dfrac{1}{2}\right) & \text{otherwise.} \end{cases} \tag{9.3}$$

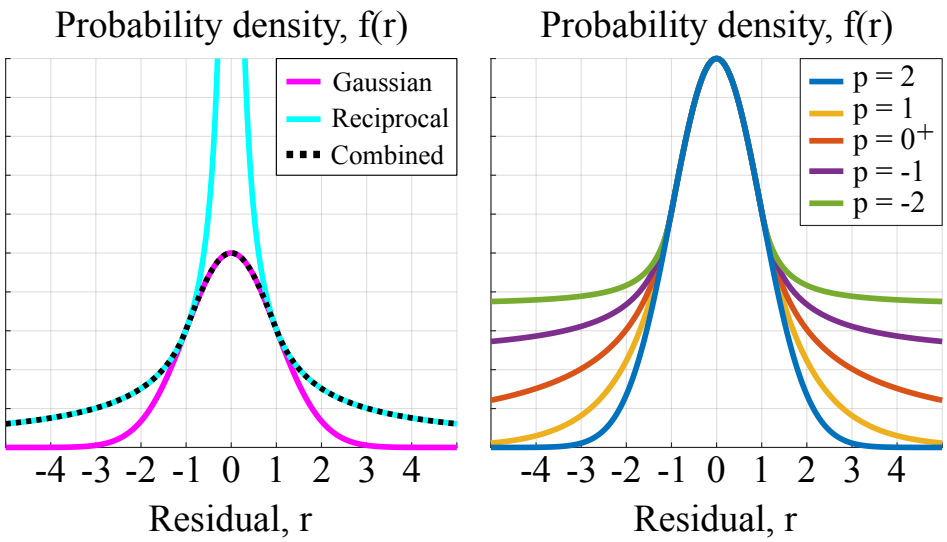


Figure 9.2: **[Left]:** The PDF (up to scale) defined by (9.6) with $c = 1$. The inner part behaves like a Gaussian density and the outer part behaves like a reciprocal function. **[Right]:** PDFs (up to scale) for which minimizing the sum of (9.1) leads to the maximum likelihood estimate.

In the following, we provide the proof, which is omitted in the original work:

Proof. As $p \to 0^+$, the second part of $\rho(r)$ in (9.1) becomes

$$\lim_{p\to 0^+} c^{2-p}\left(\frac{|r|^p}{p} - \frac{c^p}{p} + \frac{c^p}{2}\right) = \frac{c^2}{2} + c^2 \lim_{p\to 0^+} \frac{|r|^p - c^p}{p}.$$

Using L'Hôpital's Rule, this can be written as

$$\frac{c^2}{2} + c^2 \lim_{p\to 0^+} \frac{|r|^p \ln|r| - c^p \ln c}{1} = \frac{c^2}{2} + c^2(\ln|r| - \ln c) = c^2\left(\ln|r| - \ln c + \frac{1}{2}\right).$$

■

The weight function corresponding to (9.3) is then given by

$$w(r) = \begin{cases} 1 & \text{if } |r| < c \\ c^2|r|^{-2} & \text{otherwise.} \end{cases} \tag{9.4}$$

By setting a single parameter c to some constant, we can readily adopt this function in a standard IRLS scheme.

9.3 Probabilistic Interpretations of (9.3)

Using the smoothed L_{0^+} cost function (9.3), the optimization problem can be formulated as follows:

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_i \rho(r(x_i|\boldsymbol{\theta})), \tag{9.5}$$

where $\boldsymbol{\theta}$ is the set of parameters we want to estimate and $r(x_i, \boldsymbol{\theta})$ corresponds the residual from the i-th measurement x_i given $\boldsymbol{\theta}$. The solution $\boldsymbol{\theta}^*$ then corresponds to the maximum likelihood estimate of $\boldsymbol{\theta}$ when the probability density function (PDF) for the residual takes the following form:

$$f(r) = \begin{cases} k \exp\left(-\dfrac{r^2}{2}\right) & \text{if } |r| < c \\ k \left(\dfrac{c}{\sqrt{e}}\right)^{c^2} \left(\dfrac{1}{|r|}\right)^{c^2} & \text{otherwise,} \end{cases} \tag{9.6}$$

where k is the normalizing factor.

Proof. Let $f(r(x_i|\boldsymbol{\theta}))$ be the probability density of the residual corresponding to the measurement x_i given the parameter $\boldsymbol{\theta}$. The likelihood function is given by $\prod_i f(r(x_i|\boldsymbol{\theta}))$, and finding $\boldsymbol{\theta}$ that maximizes this function is equivalent to finding $\boldsymbol{\theta}$ that minimizes the negative log-likelihood, *i.e.*, $\sum_i -\ln f(r(x_i|\boldsymbol{\theta}))$. Now, if $f(r) \propto \exp(-\rho(r))$, this is equivalent to minimizing $\sum_i \rho(r(x_i|\boldsymbol{\theta}))$. In other words, the parameter $\boldsymbol{\theta}$ that minimizes $\sum_i \rho(r(x_i|\boldsymbol{\theta}))$ is the maximum likelihood estimate when the probability density takes the form $f(r) \propto \exp(-\rho(r))$ (Huber, 1964). Defining $\rho(r)$ by (9.3), the corresponding PDF is given by

$$f(r) = \begin{cases} k \exp\left(-\dfrac{r^2}{2}\right) & \text{if } |r| < c \\ k \exp\left(-c^2 \left(\ln|r| - \ln c + \dfrac{1}{2}\right)\right) & \text{otherwise.} \end{cases}$$

Simplifying the second part of this function leads to (9.6). ■

This PDF in (9.6) behaves like a Gaussian density for small $|r|$, and like a reciprocal function for large $|r|$ (see the left panel of Fig. 9.2). Following a similar derivation to that of (9.6), we can also derive the underlying PDFs associated with (9.1). For example, we obtain a Gaussian density when $p = 2$ and the Huber density (Meyer, 2021) when $p = 1$. The right panel of Fig. 9.2 compares the resulting probability densities for different values of p. Notice that the L_{0^+} function has heavier tails than the Gaussian and the Huber density.

Note that when we make a log-log plot of the PDF (9.6), a straight line of gradient $-c^2$ should appear for large $|r|$. This is interesting for two reasons: First, we can use this fact to determine for which optimization problem the L_{0^+} cost function (9.3) is beneficial. If a negatively sloped straight line appears in a log-log plot of a typical residual distribution, then it would be more appropriate to model the PDF with (9.6) than with a uniform outlier distribution, for example (see Fig. 9.3). In Section 9.4.1, we show that large-scale rotation averaging is one of the problems that meet this criterion. Second, should such a straight line do appear in the log-log plot, we can also deduce the value of c in (9.6) by fitting a line to it. In Section 9.4.2, we show that setting the value of c this way and using it in the weight function (9.4) leads to good optimization results.

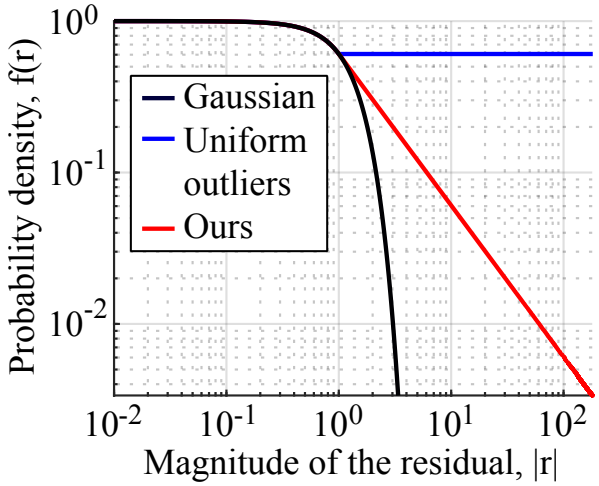


Figure 9.3: Log-log plot of three different PDFs (up to scale): (1) a Gaussian density, (2) a Gaussian density for small $|r|$ and a uniform density for large $|r|$, and (3) the one we use, *i.e.*, (9.6). The third one is characterized by a negatively sloped straight line for large $|r|$.

9.4 Application to Multiple Rotation Averaging

In the IRLS step of HARA (Lee and Civera, 2022), we replace the original reweighting function with (9.4). In all our experiments, we set $c = 1^\circ$ unless otherwise stated. We evaluate our method on the following real-world datasets: 1DSfM datasets[1] (Wilson and Snavely, 2014), 'Notre Dame 715' (ND2), 'Acropolis' (ACP), 'Arts Quad' (ARQ) and 'San Francisco' (SNF) datasets[2] (Crandall et al., 2011). As in (Chatterjee and Govindu, 2018, Lee and Civera, 2022), we use all images in the connected graph for rotation averaging, but only those with the ground truth data are used for evaluating the accuracy. We align the estimated rotations to the ground truth and compute the optimal mean and RMS errors (Lee and Civera, 2022).

9.4.1 Error distribution

Before evaluating the rotation averaging results, we first analyze the error distribution of the tested datasets. The errors are computed by comparing the relative rotation estimates against the ground truth, both of which are provided in the datasets. Here, we only consider the subset of errors corresponding to the filtered edges from HARA (Lee and Civera, 2022) because only those edges are used for local optimization. Fig. 9.3 shows the distributions on a log-log scale. In most of the graphs, a straight segment appears for large errors, which is in accordance with the observation in (Chatterjee and Govindu, 2018). This suggests that our PDF (9.6) is a better model than a commonly used uniform outlier distribution, as illustrated previously in Fig. 9.3. Inferring the median value of c from the gradient of the fitted lines (according to (9.6)), we get $c \approx 1.2$ deg.

[1] `http://www.cs.cornell.edu/projects/1dsfm/`
[2] `http://vision.soic.indiana.edu/projects/disco/`

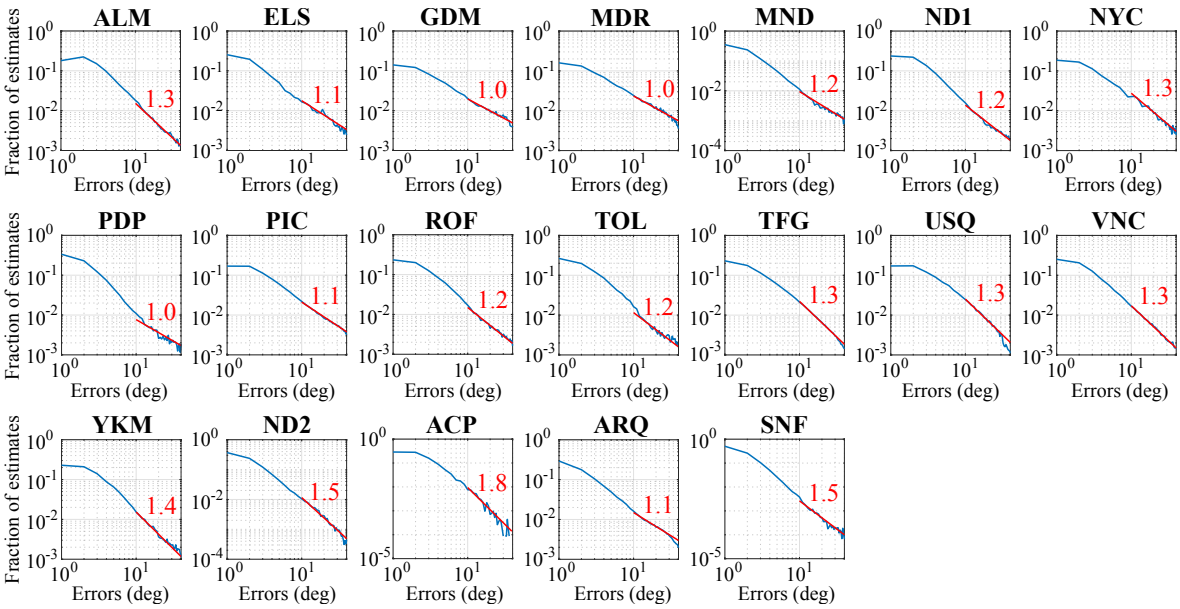


Figure 9.4: Log-log plot of the distribution of errors in the relative rotation estimates (provided in the real datasets in Section 9.4). In most graphs, a linear behavior is observed for large errors. We fit a line (in red) to the points between 10 and 40 deg using the least-median-of-squares method (Rousseeuw, 1984). The number in red is the value of c which we estimate as the square-root of the negative gradient (according to (9.6)).

9.4.2 Parameter study

Next, we investigate the influence of p in (9.1) while keeping c at 1 deg. This is done by setting $p = \{-8, -7.5, -7, \cdots, -0.5, 0^+, 0.5, 1\}$ in the weight function (9.2) and using it for the reweighting step instead of (9.4). In order to aggregate the results from multiple datasets, we shift the errors for each dataset such that the minimum is zero. That is, from the errors obtained from each p, we subtract the smallest one among them. In Fig. 9.5, we show how different values of p affect the mean and the median of the shifted errors from 18 datasets, *i.e.* all but GDM in Tab. 9.1. We excluded GDM because its result was constantly dictated by outliers and deemed unreliable. Fig. 9.5 clearly shows that the best overall result is obtained when $p \to 0^+$, which supports our choice of using (9.4).

In a similar manner, we also study the influence of the parameter c in (9.4). The result is shown in Fig. 9.6. It suggests that good performance is achieved when $c \approx 1$ deg. This agrees with the value we estimated from the error distribution in Section 9.4.1.

9.4.3 Comparison to the state of the art

Table 9.1 presents the results of our method (HARA-L_{0+}) and other recent works, namely R-GoDec[3] (Arrigoni et al., 2014, 2018), Eig-IRLS[4] (Arrigoni et al., 2018), IRLS-$L_{0.5}$[5] (Chatterjee and Govindu, 2018), MPLS[6] (Shi and Lerman, 2020) and the original HARA[7](Lee and

[3] `http://www.diegm.uniud.it/fusiello/demo/gmf/`
[4] The code was kindly provided by the authors of (Arrigoni et al., 2018).
[5] `http://www.ee.iisc.ac.in/labs/cvl/research/rotaveraging/`
[6] `https://github.com/yunpeng-shi/MPLS`
[7] `https://github.com/sunghoon031/HARA`

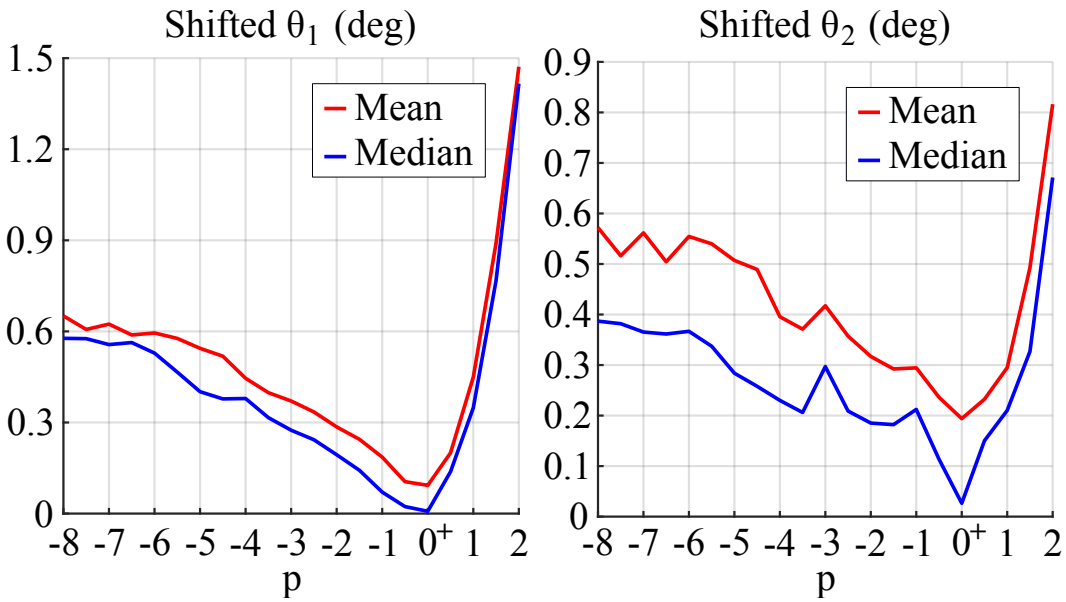


Figure 9.5: The influence of p in (9.2) on the shifted optimal mean errors θ_1 **(left)** and RMS errors θ_2 **(right)** from the 18 datasets (*i.e.* all but GDM in Tab. 9.1). Here, we fixed the parameter c in (9.2) at 1 deg. The mean and the median of both error metrics are minimum when $p \to 0^+$.

Civera, 2022) (without using the number of inlier matches). It shows that in most datasets HARA with our optimization achieves the highest accuracy at a comparable speed.

9.5 Conclusion

In this chapter, we proposed to use the smoothed L_{0^+} function introduced in (Peng et al., 2022) in the local refinement step of HARA (Lee and Civera, 2022). We presented the mathematical proof that the L_{0^+} part of the function is equivalent to $A \ln |x| + B$ for some constants $A > 0$ and B. Also, we provided the probabilistic interpretations of this cost function, outlining how we can use the log-log plot of the PDF to determine whether or not this function is suitable for the problem at hand, and if so, how we can set the value of its tuning parameter. Our evaluation on the real datasets demonstrates the effectiveness of the proposed method for robust multiple rotation averaging.

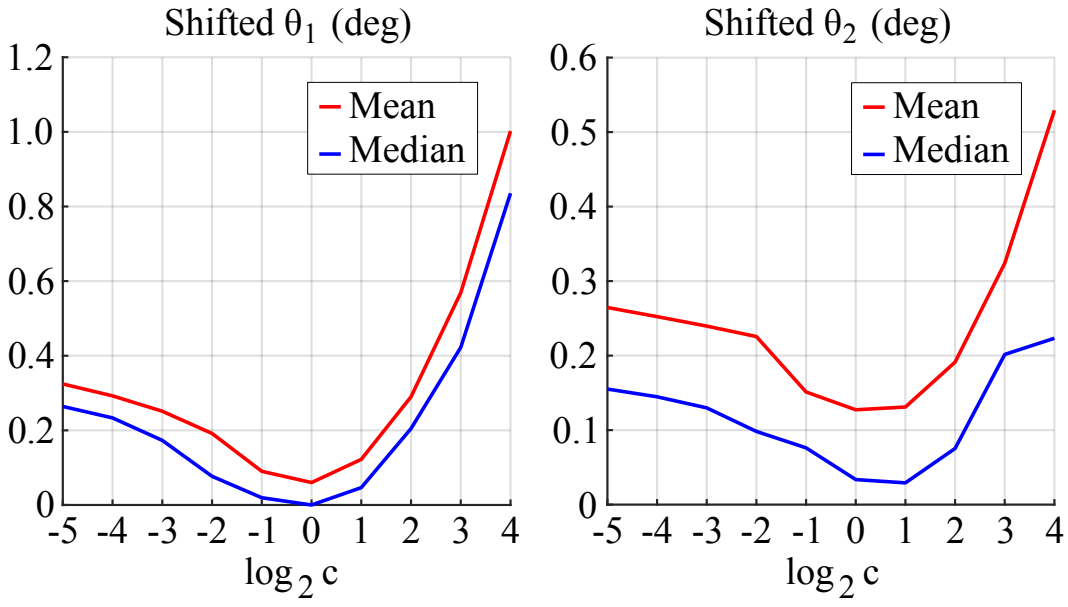


Figure 9.6: The influence of c in (9.4) on the shifted optimal mean errors θ_1 **(left)** and RMS errors θ_2 **(right)** from the 18 datasets (*i.e.* all but GDM in Tab. 9.1). The errors are minimum around $\log_2 c \approx 0$, that is, $c \approx 1$ deg.

Datasets			R-GoDec			Eig-IRLS			IRLS-$L_{0.5}$			MPLS*			HARA			HARA-L_{0+}		
Name	#views	%edges	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time	θ_1	θ_2	Time
ALM	627	49.5%	6.3	16.3	4s	3.9	12.2	21s	4.2	12.6	27s	3.7	12.1	29s	3.5	11.5	61s	**3.1**	**11.0**	62s
ELS	247	66.8%	4.1	10.7	1s	3.3	11.5	3s	2.9	10.3	4s	2.8	10.9	7s	**2.1**	**7.4**	9s	**2.1**	**7.4**	9s
GDM	742	17.5%	51.3	64.8	17s	65.8	74.2	29s	**37.5**	**62.3**	12s	40.7	68.7	74s	44.3	72.8	30s	44.3	73.1	30s
MDR	394	30.7%	10.2	18.5	1s	10.9	22.4	5s	7.0	17.1	4s	5.2	14.7	5s	4.8	14.5	16s	**4.5**	**14.4**	16s
MND	474	46.8%	6.2	18.4	3s	1.9	11.2	7s	1.5	7.4	10s	1.2	3.9	9s	**1.1**	**2.1**	27s	**1.1**	**2.1**	28s
ND1	553	68.1%	5.2	15.5	6s	3.6	14.8	16s	3.5	14.6	28s	2.7	13.5	27s	1.6	6.3	70s	**1.5**	**6.2**	71s
NYC	376	29.3%	6.4	9.9	6s	3.8	8.2	5s	3.0	**7.0**	3s	3.0	8.2	7s	**2.9**	7.7	10s	**2.9**	7.8	10s
PDP	354	39.5%	11.4	22.0	2s	4.0	9.3	8s	4.1	8.1	4s	3.5	8.2	4s	3.5	**7.8**	10s	**3.4**	7.9	10s
PIC	2508	10.2%	24.8	40.0	150s	81.0	91.2	687s	6.8	18.6	467s	4.6	14.6	295s	4.4	**12.9**	371s	**4.2**	**12.9**	356s
ROF	1134	10.9%	12.6	19.5	61s	3.4	10.4	52s	3.1	10.2	12s	2.8	10.0	13s	2.8	8.7	37s	**2.7**	**8.6**	39s
TOL	508	18.5%	6.4	13.1	8s	4.5	10.7	10s	**3.9**	**9.0**	2s	4.0	9.4	6s	4.3	10.0	11s	4.1	10.1	11s
TFG	5433	4.6%	42.1	54.2	722s	59.4	67.1	833s	3.6	**9.8**	976s	4.5	10.8	1945s	3.5	10.4	1210s	**3.3**	10.4	1215s
USQ	930	5.9%	12.0	23.7	27s	6.7	12.8	22s	9.3	22.2	10s	6.3	14.7	11s	6.0	12.3	12s	**5.9**	**12.2**	12s
VNC	918	24.6%	16.1	36.9	25s	8.6	28.4	27s	8.3	27.5	28s	6.2	18.2	53s	**6.1**	**18.1**	65s	**6.1**	**18.1**	89s
YKM	458	26.5%	6.1	11.3	7s	3.8	9.4	8s	3.5	8.4	3s	3.5	9.2	7s	**3.0**	**6.8**	14s	**3.0**	7.0	14s
ND2	715	25.3%	2.7	10.0	10s	1.2	4.0	13s	**1.1**	**3.5**	13s	**1.1**	4.0	12s	1.3	5.5	29s	1.3	5.5	29s
ACP	463	10.7%	**0.8**	**1.2**	6s	1.1	1.7	4s	1.2	1.7	1s	1.4	2.0	2s	1.2	1.7	4s	1.2	1.7	4s
ARQ	5530	1.5%	29.7	56.5	1111s	70.1	79.7	4894s	4.0	7.1	173s	**3.2**	**6.3**	118s	3.6	6.8	166s	3.4	6.6	168s
SNF	7866	0.3%	Out of memory			77.3	87.3	3.8h	3.6	4.2	180s	4.4	5.5	154s	3.6	4.2	50s	**2.9**	**3.5**	51s

θ_1 (deg): Optimal mean error, θ_2 (deg): Optimal RMS error, %edges = #edges/#possible pairs of views in %.

*Due to the non-deterministic nature of the algorithm, we take the median of five runs.

Blue: Best result, Underline: Better result between the original HARA and ours.

Table 9.1: Results on the real-world benchmark datasets: For all datasets, our optimization leads to either better or comparable results to the original HARA (Lee and Civera, 2022). Especially, the improvement is significant for the SNF dataset. All computational times are taken from (Lee and Civera, 2022), except for HARA with/without our optimization. These last two methods were run on a laptop with Intel's 4th Gen i7 CPU (2.8 GHz).

Chapter 10

Rotation-Only Bundle Adjustment

In this chapter, we propose a novel method for estimating the global rotations of the cameras independently of their positions and the scene structure. When two calibrated cameras observe five or more of the same points, their relative rotation can be recovered independently of the translation. We extend this idea to multiple views, thereby decoupling the rotation estimation from the translation and structure estimation. Our approach provides several benefits such as complete immunity to inaccurate translations and structure, and the accuracy improvement when used with rotation averaging. We perform extensive evaluations on both synthetic and real datasets, demonstrating consistent and significant gains in accuracy when used with the state-of-the-art rotation averaging method.

10.1 Introduction

Bundle adjustment is the problem of reconstructing the camera poses (*i.e.*, rotations and translations) and the 3D scene structure from the image measurements. It plays a crucial role in many areas of 3D vision, such as structure from motion (Hartley and Zisserman, 2003), visual odometry (Scaramuzza and Fraundorfer, 2011), and simultaneous localization and mapping (Cadena et al., 2016). For this reason, significant research endeavors have been devoted to this problem, which led to tremendous progress over the past two decades.

Bundle adjustment aims to obtain jointly optimal structure and camera poses by minimizing the image reprojection errors (Triggs et al., 2000). Being a nonlinear optimization problem, it requires a good initialization to ensure the convergence to the statistically optimal solution (Hartley and Zisserman, 2003). A common strategy involves the following steps: (1) Estimate the pairwise motions. (2) Estimate the global rotations through rotation averaging (*e.g.*, (Arrigoni et al., 2018, Chatterjee and Govindu, 2018)). (3) Estimate the global translations (*e.g.*, (Govindu, 2001, Wilson and Snavely, 2014)). (4) Triangulate the points (*e.g.*, (Kang et al., 2014, Lee and Civera, 2020*c*)).

In such a pipeline, it is important to make an accurate initial guess of the rotations, as the subsequent steps directly depend on it. To this end, one could try to improve the rotation

averaging method or its input (*i.e.*, the relative pairwise motion estimates). Recent examples of the former include (Arrigoni et al., 2018, Chatterjee and Govindu, 2018, Dellaert et al., 2020*b*, Eriksson et al., 2021, Purkait et al., 2020) and the latter include (Brachmann and Rother, 2019, Briales et al., 2018, Garcia-Salguero et al., 2021, Zhao, 2022).

These two types of approaches are certainly useful for initializing the rotations. However, relative pose estimation is limited to two views only, while rotation averaging does not directly leverage the image measurements. That is, it treats all relative rotations equally even if they were estimated from different numbers of points with different noise statistics and distributions. To our knowledge, no previous work has addressed this limitation for rotation estimation.

In this chapter, we present a novel method that, given the initial estimates of the rotations, performs rotation-only optimization using the image measurements as direct input. Our work is based on (Kneip and Lynen, 2013), where it was proposed to optimize the rotation between two views independently of the translation. We extend this idea to multiple views. We call our approach *rotation-only bundle adjustment* because it can be seen as the decoupling of the rotation estimation from the translation and structure estimation in bundle adjustment. This provides the following advantages:

- The rotations are estimated without requiring the knowledge of the translations and structure. This greatly simplifies the optimization problem.
- The rotations are immune to inaccurate estimation of the translations and structure.
- Both pure and non-pure rotations are treated in a unified manner, as we do not need to triangulate and discard the low-parallax points.
- It can be used after rotation averaging to improve the accuracy of the rotation estimates.

Table 10.1 summarizes the differences between our method and the related methods.

The chapter is organized as follows: In the next two sections, we review the related work and the preliminaries. Section 10.4 reviews the two-view rotation-only method by Kneip and Lynen (Kneip and Lynen, 2013). We describe our method in Section 10.5 and show the experimental results in Section 10.6. Finally, Section 10.7 and 10.8 present discussions and conclusions.

10.2 Related Work

Our work is related to several areas of study in 3D vision and robotics, namely structure from motion, simultaneous localization and mapping, bundle adjustment, rotation averaging, and relative pose estimation.

Structure from motion (SfM) is the problem of recovering the camera poses and the 3D scene from an unordered set of images. Large-scale systems may handle from hundreds of thousands (Agarwal et al., 2011) to millions of images (Heinly et al., 2015, Zhu et al.,

	Independent of the translations and the 3D scene structure?	Directly using the image measurements as input?	Applicable to n views?
Full bundle adjustment (*e.g.*, (Triggs et al., 2000))	✗	✓	✓
Rotation averaging (*e.g.*, (Chatterjee and Govindu, 2018))	✓	✗	✓
Direct rotation optimization (Kneip and Lynen, 2013)	✓	✓	✗
Rotation-only bundle adjustment (proposed in this work)	✓	✓	✓

Table 10.1: Comparison between the related methods. To the best of our knowledge, we are the first to propose a multiview rotation-only optimization method using the image measurements as direct input. Our method can be generalized to both pure and non-pure rotations.

2018). We refer to (Ozyesil et al., 2017, Schönberger and Frahm, 2016) for excellent reviews of the SfM literature. The backbone of most SfM systems is bundle adjustment, the joint optimization of the camera poses and the 3D structure. To obtain the optimal solution, it requires good initial estimates of the poses and points (Hartley and Zisserman, 2003). In many works (Arie-Nachimson et al., 2012, Cui et al., 2015, Cui and Tan, 2015, Enqvist et al., 2011, Govindu, 2001, Martinec and Pajdla, 2007, Moulon et al., 2013, Wilson and Snavely, 2014, Ozyesil and Singer, 2015), the initialization consists of the following steps: (1) Estimate the relative poses between the camera pairs observing many points in common. (2) Perform multiple rotation averaging. (3) Estimate the camera locations (and the 3D points). In such a pipeline, one may use our method as Step 2.5 to refine the rotations.

Simultaneous localization and mapping (SLAM) is the problem of estimating the camera motion and the 3D scene in real time. Like SfM, the modern SLAM systems rely on bundle adjustment to jointly optimize the keyframe poses and the map points (Campos et al., 2021, Engel et al., 2018, Lee and Civera, 2019*b*, Mur-Artal and Tardós, 2017). In (Chng et al., 2020, Bustos et al., 2019), it was suggested that decoupling the rotation estimation through rotation averaging improves the efficiency and the handling of pure rotations.

Bundle adjustment is mainly classified into geometric and photometric methods. The former minimizes the reprojection errors (Agarwal et al., 2010, Hartley and Zisserman, 2003, Lourakis and Argyros, 2009, Triggs et al., 2000), and the latter minimizes the photometric errors (Alismail et al., 2017, Delaunoy and Pollefeys, 2014, Engel et al., 2018, Woodford and Rosten, 2020). The bundle adjustment problems have been studied extensively for several decades, which led to diverse techniques for improving the scalability (*e.g.*, (Agarwal et al., 2010, Konolige, 2010, Kümmerle et al., 2011, Lourakis and Argyros, 2009)) and the accuracy (*e.g.*, (Triggs et al., 2000, Zach, 2014, Zach and Bourmaud, 2018)). In (Hong and Zach, 2018), an initialization-free approach was proposed. To our knowledge, however, no

previous work has completely decoupled the rotation estimation in bundle adjustment.

Rotation averaging takes two forms: single rotation averaging that averages several estimates of a single rotation to obtain the best estimate (Hartley et al., 2011, Lee and Civera, 2020*b*), and multiple rotation averaging that finds the multiple rotations $\mathbf{R}_i$ given several noisy constraints on the relative rotations $\mathbf{R}_i\mathbf{R}_j^\top$ (Arie-Nachimson et al., 2012, Arrigoni et al., 2018, Chatterjee and Govindu, 2018, Dellaert et al., 2020*a*, Eriksson et al., 2021, Hartley et al., 2011, Martinec and Pajdla, 2007). We refer to (Hartley, Trumpf, Dai and Li, 2013, Tron et al., 2016) for an excellent tutorial and survey on the topic. As discussed earlier, multiple rotation averaging has wide application to SfM. This problem differs from bundle adjustment in that (1) only rotations are estimated, and (2) the input is the relative rotation estimates, not the image measurements. Since the states and the input are small compared to bundle adjustment, the computation process is faster and simpler. However, the downside is that it does not directly reflect the errors with respect to the image measurements. In contrast, our optimization method directly uses the image measurements as input, while maintaining only rotations in the state space. In this context, our method can be seen as the middle ground between rotation averaging and bundle adjustment.

Relative pose estimation Given a set of five or more point matches between two calibrated views, their relative pose can be obtained using a minimal method (*e.g.*, (Fathian et al., 2018, Hongdong Li and Hartley, 2006, Kukelova et al., 2008, Nistér, 2004)) with RANSAC (Brachmann and Rother, 2019, Fischler and Bolles, 1981, Raguram et al., 2013) or a non-minimal method (*e.g.*, (Briales et al., 2018, Hartley, 1997, Kneip and Lynen, 2013, Zhao, 2022)). In (Kneip and Lynen, 2013, Kneip et al., 2012), it was shown that the rotation can be estimated independently of the translation. In this work, we extend the idea of (Kneip and Lynen, 2013) to $n \geq 2$ views by aggregating multiple two-view costs and minimizing it through iterative nonlinear optimization.

10.3 Preliminaries and Notation

We use bold lowercase letters for vectors, bold uppercase letters for matrices, and light letters for scalars. We denote the Hadamard product, division and square root by $\mathbf{A} \circ \mathbf{B}$, $\mathbf{A} \oslash \mathbf{B}$ and $\mathbf{A}^{\circ 1/2}$, respectively. For a 3D vector $\mathbf{v}$, we define $\mathbf{v}^\wedge$ as the corresponding 3×3 skew-symmetric matrix, and denote the inverse operator by $(\cdot)^\vee$, i.e., $(\mathbf{v}^\wedge)^\vee = \mathbf{v}$. A rotation matrix $\mathbf{R} \in SO(3)$ can be represented by the corresponding rotation vector $\mathbf{u} = \theta\widehat{\mathbf{u}}$, where θ and $\widehat{\mathbf{u}}$ represent the angle and the unit axis of the rotation, respectively. The two representations are related by Rodrigues formula, and we denote the mapping between them by $\mathrm{Exp}(\cdot)$ and $\mathrm{Log}(\cdot)$ (Forster, Carlone, Dellaert and Scaramuzza, 2017):

$$\mathbf{R} = \mathrm{Exp}(\mathbf{u}) := \mathbf{I} + \frac{\sin\left(\|\mathbf{u}\|\right)}{\|\mathbf{u}\|}\mathbf{u}^\wedge + \frac{1-\cos\left(\|\mathbf{u}\|\right)}{\|\mathbf{u}\|^2}\left(\mathbf{u}^\wedge\right)^2, \tag{10.1}$$

$$\mathbf{u} = \mathrm{Log}(\mathbf{R}) := \frac{\theta}{2\sin(\theta)}\left(\mathbf{R} - \mathbf{R}^\top\right)^\vee \tag{10.2}$$

$$\text{with} \quad \theta = \arccos\left(\left(\mathrm{tr}(\mathbf{R}) - 1\right)/2\right). \tag{10.3}$$

We denote the 3D position of a point with index i in a world reference frame w as $(\mathbf{x}_i)_w = [(x_i)_w, (y_i)_w, (z_i)_w]^\top$ and a perspective camera with index j as c_j. In the reference frame of c_j, the position of $\mathbf{x}_i$ is given by $(\mathbf{x}_i)_j = [(x_i)_j, (y_i)_j, (z_i)_j]^\top = \mathbf{R}_j(\mathbf{x}_i)_w + \mathbf{t}_j$, where $\mathbf{R}_j$ and $\mathbf{t}_j$ are the rotation and translation that relate the local reference frame of c_j to the world. The projection of $\mathbf{x}_i$ in the image plane of c_j has the pixel coordinates

$$\begin{bmatrix}(u_i)_j\\(v_i)_j\end{bmatrix} = \begin{bmatrix}1 & 0 & 0\\0 & 1 & 0\end{bmatrix}\mathbf{K}_j(\mathbf{f}_i)_j,$$

where $\mathbf{K}_j$ is the camera calibration matrix of c_j and $(\mathbf{f}_i)_j = [(x_i)_j/(z_i)_j, (y_i)_j/(z_i)_j, 1]^\top$ is the normalized image coordinates of $(\mathbf{x}_i)_j$.

Then, $(\mathbf{f}_i)_j$ can be obtained by $(\mathbf{f}_i)_j = \mathbf{K}_j^{-1}[(u_i)_j, (v_i)_j, 1]^\top$. We denote the rotation and the translation between camera j and k as $\mathbf{R}_{jk}$ and $\mathbf{t}_{jk}$. The point $\mathbf{x}_i$ in the reference frame of c_j and c_k is related by $(\mathbf{x}_i)_j = \mathbf{R}_{jk}(\mathbf{x}_i)_k + \mathbf{t}_{jk}$. This means that $\mathbf{R}_{jk} = \mathbf{R}_j\mathbf{R}_k^\top$ and $\mathbf{t}_{jk} = -\mathbf{R}_j\mathbf{R}_k^\top\mathbf{t}_k + \mathbf{t}_j$.

10.4 Review of Two-view Rotation-Only Method

In this section, we review the two-view rotation-only optimization method proposed in (Kneip and Lynen, 2013). Consider two views with known internal calibration, c_j and c_k, observing $m \geq 5$ common points with index $i \in \{1, 2, \ldots, m\}$. The normalized epipolar error (Lee and Civera, 2020*a*) associated with each point i is defined as

$$(e_i)_{(j,k)} = \left|\widehat{\mathbf{t}}_{jk} \cdot \left(\left(\widehat{\mathbf{f}}_i\right)_j \times \mathbf{R}_{jk}\left(\widehat{\mathbf{f}}_i\right)_k\right)\right|, \tag{10.4}$$

where $\left(\widehat{\mathbf{f}}_i\right)_j$ and $\left(\widehat{\mathbf{f}}_i\right)_k$ are the unit bearing vectors corresponding to the ith point in c_j and c_k, respectively. The sum of squares of all these errors is then given by

$$\sum_{i=1}^{m}(e_i)_{(j,k)}^2 = \widehat{\mathbf{t}}_{jk}^\top\mathbf{M}_{jk}\widehat{\mathbf{t}}_{jk}, \tag{10.5}$$

where

$$\mathbf{M}_{jk} = \sum_{i=1}^{m}\left(\left(\widehat{\mathbf{f}}_i\right)_j \times \mathbf{R}_{jk}\left(\widehat{\mathbf{f}}_i\right)_k\right)\left(\left(\widehat{\mathbf{f}}_i\right)_j \times \mathbf{R}_{jk}\left(\widehat{\mathbf{f}}_i\right)_k\right)^\top. \tag{10.6}$$

In (Kneip et al., 2012), it was shown that the 3×3 matrix $\mathbf{M}_{jk}$ can also be computed as follows: denoting the entries of $\left(\widehat{\mathbf{f}}_i\right)_j$ as $[(f_{xi})_j, (f_{yi})_j, (f_{zi})_j]^\top$, the following matrices are defined:

$$(\mathbf{F}_{xx})_{jk} = \sum_{i=1}^{m} (f_{xi})_j^2 \left(\widehat{\mathbf{f}}_i\right)_k \left(\widehat{\mathbf{f}}_i\right)_k^\top, \tag{10.7}$$

$$(\mathbf{F}_{xy})_{jk} = \sum_{i=1}^{m} (f_{xi})_j (f_{yi})_j \left(\widehat{\mathbf{f}}_i\right)_k \left(\widehat{\mathbf{f}}_i\right)_k^\top, \tag{10.8}$$

$$(\mathbf{F}_{xz})_{jk} = \sum_{i=1}^{m} (f_{xi})_j (f_{zi})_j \left(\widehat{\mathbf{f}}_i\right)_k \left(\widehat{\mathbf{f}}_i\right)_k^\top, \tag{10.9}$$

$$(\mathbf{F}_{yy})_{jk} = \sum_{i=1}^{m} (f_{yi})_j^2 \left(\widehat{\mathbf{f}}_i\right)_k \left(\widehat{\mathbf{f}}_i\right)_k^\top, \tag{10.10}$$

$$(\mathbf{F}_{yz})_{jk} = \sum_{i=1}^{m} (f_{yi})_j (f_{zi})_j \left(\widehat{\mathbf{f}}_i\right)_k \left(\widehat{\mathbf{f}}_i\right)_k^\top. \tag{10.11}$$

Let $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ be each row of $\mathbf{R}_{jk}$, and m_{ab} be the element of $\mathbf{M}_{jk}$ at the ath row and bth column. (Notice that we omitted the subscript jk here for simplicity). Then,

$$m_{11} = \mathbf{r}_3 \mathbf{F}_{yy} \mathbf{r}_3^\top - 2\mathbf{r}_3 \mathbf{F}_{yz} \mathbf{r}_2^\top + \mathbf{r}_2 \mathbf{F}_{zz} \mathbf{r}_2^\top, \tag{10.12}$$

$$m_{22} = \mathbf{r}_1 \mathbf{F}_{zz} \mathbf{r}_1^\top - 2\mathbf{r}_1 \mathbf{F}_{xz} \mathbf{r}_3^\top + \mathbf{r}_3 \mathbf{F}_{xx} \mathbf{r}_3^\top, \tag{10.13}$$

$$m_{33} = \mathbf{r}_2 \mathbf{F}_{xx} \mathbf{r}_2^\top - 2\mathbf{r}_1 \mathbf{F}_{xy} \mathbf{r}_2^\top + \mathbf{r}_1 \mathbf{F}_{yy} \mathbf{r}_1^\top, \tag{10.14}$$

$$m_{12} = \mathbf{r}_1 \mathbf{F}_{yz} \mathbf{r}_3^\top - \mathbf{r}_1 \mathbf{F}_{zz} \mathbf{r}_2^\top - \mathbf{r}_3 \mathbf{F}_{xy} \mathbf{r}_3^\top + \mathbf{r}_3 \mathbf{F}_{xz} \mathbf{r}_2^\top, \tag{10.15}$$

$$m_{13} = \mathbf{r}_2 \mathbf{F}_{xy} \mathbf{r}_3^\top - \mathbf{r}_2 \mathbf{F}_{xz} \mathbf{r}_2^\top - \mathbf{r}_1 \mathbf{F}_{yy} \mathbf{r}_3^\top + \mathbf{r}_1 \mathbf{F}_{yz} \mathbf{r}_2^\top, \tag{10.16}$$

$$m_{23} = \mathbf{r}_1 \mathbf{F}_{xz} \mathbf{r}_2^\top - \mathbf{r}_1 \mathbf{F}_{yz} \mathbf{r}_1^\top - \mathbf{r}_3 \mathbf{F}_{xx} \mathbf{r}_2^\top + \mathbf{r}_3 \mathbf{F}_{xy} \mathbf{r}_1^\top, \tag{10.17}$$

and $m_{21} = m_{12}$, $m_{31} = m_{13}$, $m_{32} = m_{23}$. This is a more efficient computation of $\mathbf{M}_{jk}$ than (10.6), as (10.7)–(10.11) can be precomputed regardless of $\mathbf{R}_{jk}$, reducing the number of operations during the rotation optimization (Kneip and Lynen, 2013). Given the set of corresponding unit bearing vectors, one can jointly optimize the relative rotation and translation by minimizing (10.5) with respect to $\mathbf{R}_{jk}$ and $\widehat{\mathbf{t}}_{jk}$. In (Kneip and Lynen, 2013), it was shown that this problem can be transformed into a rotation-only form:

$$\mathbf{R}_{jk}^* = \underset{\mathbf{R}_{jk}}{\operatorname{argmin}} \, \lambda_{\mathbf{M}}(\mathbf{R}_{jk}), \tag{10.18}$$

where $\lambda_{\mathbf{M}}(\mathbf{R}_{jk})$ is the smallest eigenvalue of $\mathbf{M}_{jk}$ (which is a function of $\mathbf{R}_{jk}$). This eigenvalue can be obtained in closed form (Kneip and Lynen, 2013):

$$b_1 = -m_{11} - m_{22} - m_{33}, \tag{10.19}$$

$$b_2 = -m_{13}^2 - m_{23}^2 - m_{12}^2 + m_{11}m_{22} + m_{11}m_{33} + m_{22}m_{33}, \tag{10.20}$$

$$b_3 = m_{22}m_{13}^2 + m_{11}m_{23}^2 + m_{33}m_{12}^2 - m_{11}m_{22}m_{33} - 2m_{12}m_{23}m_{13}, \tag{10.21}$$

$$s = 2b_1^3 - 9b_1b_2 + 27b_3, \tag{10.22}$$

$$t = 4(b_1^2 - 3b_2)^3, \tag{10.23}$$

$$k = \left(\sqrt{t}/2\right)^{1/3} \cos\left(\arccos\left(s/\sqrt{t}\right)/3\right), \tag{10.24}$$

$$\lambda_{\mathbf{M}}(\mathbf{R}_{jk}) = (-b_1 - 2k)/3. \tag{10.25}$$

To summarize, the rotation part of the optimal solution $(\mathbf{R}_{jk}^*, \widehat{\mathbf{t}}_{jk}^*)$ that minimizes (10.5) is obtained by solving (10.18).

10.5 Rotation-Only Bundle Adjustment

10.5.1 Cost function

We extend the idea of (Kneip and Lynen, 2013) for n views. Let $\mathcal{E}$ be the set of all edges, *i.e.*, camera pairs (j, k) observing a sufficient number of points in common (> 10 in our implementation). Then, we formulate the optimization problem as follows:

$$\{\mathbf{R}_1^*, \cdots, \mathbf{R}_n^*\} = \underset{\mathbf{R}_1, \cdots, \mathbf{R}_n}{\operatorname{argmin}} C(\mathbf{R}_1, \cdots, \mathbf{R}_n) \tag{10.26}$$

with

$$C(\mathbf{R}_1, \cdots, \mathbf{R}_n) = \sum_{(j,k)\in\mathcal{E}} \underbrace{\sqrt{\lambda_{\mathbf{M}}(\mathbf{R}_{jk})}}_{c_{jk}}, \tag{10.27}$$

where $\lambda_{\mathbf{M}}(\mathbf{R}_{jk})$ is the same cost function used in (10.18) for the two-view case and $c_{jk} = \sqrt{\lambda_{\mathbf{M}}(\mathbf{R}_{jk})}$ is our edge cost. We empirically found that this square rooting improves the convergence rate (see Table 10.5), which we presume is due to the downweighted influence of outliers. Alg. 9 summarizes the steps for computing the edge cost c_{jk}.

10.5.2 Optimization

To solve (10.26) iteratively, we use Adam (Kingma and Ba, 2015), a first-order gradient-based optimization algorithm for stochastic objective functions. Adam has been widely used in deep learning, and we found that it also works well for our geometric optimization problem. Given the initial estimates of $\mathbf{R}_1, \cdots, \mathbf{R}_n$, let $\mathbf{s}_0$ be the initial state vector formed by stacking $\operatorname{Log}(\mathbf{R}_1), \cdots, \operatorname{Log}(\mathbf{R}_n)$ in one column. Let $\mathbf{m}_0 = \mathbf{0}_{3n\times 1}$, $\mathbf{v}_0 = \mathbf{0}_{3n\times 1}$, $t = 0$ and $\boldsymbol{\epsilon} = (10^{-8})\mathbf{1}_{3n\times 1}$. Then, using Adam, we repeat the following steps at each iteration t of our optimization:

$$t \leftarrow t + 1, \tag{10.28}$$

$$\mathbf{g}_t \leftarrow \nabla_{\mathbf{s}} C(\mathbf{R}_1, \cdots, \mathbf{R}_n), \tag{10.29}$$

$$\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1)\mathbf{g}_t, \tag{10.30}$$

$$\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2)(\mathbf{g}_t \circ \mathbf{g}_t), \tag{10.31}$$

$$\mathbf{m}_t' \leftarrow \mathbf{m}_t / (1 - \beta_1^t), \tag{10.32}$$

$$\mathbf{v}_t' \leftarrow \mathbf{v}_t / (1 - \beta_2^t), \tag{10.33}$$

$$\mathbf{s}_t \leftarrow \mathbf{s}_{t-1} - \alpha \mathbf{m}_t' \oslash (\mathbf{v}_t'^{\circ 1/2} + \boldsymbol{\epsilon}), \tag{10.34}$$

$$\mathbf{u}_i \leftarrow [(\mathbf{s}_t)_{3i-2}, (\mathbf{s}_t)_{3i-1}, (\mathbf{s}_t)_{3i}]^\top \text{ for } i = 1, \cdots, n, \tag{10.35}$$

$$\mathbf{R}_i \leftarrow \operatorname{Exp}(\mathbf{u}_i) \text{ for } i = 1, \cdots, n. \tag{10.36}$$

Algorithm 9: Computation of edge cost c_{jk}

Input: Relative rotation $\mathbf{R}_{jk}$, Precomputed matrices $(\mathbf{F}_{xx})_{jk}, (\mathbf{F}_{xy})_{jk}, (\mathbf{F}_{xz})_{jk}, (\mathbf{F}_{yy})_{jk}, (\mathbf{F}_{yz})_{jk}$.
Output: Edge cost c_{jk}.
$\mathbf{r}_i \leftarrow (\mathbf{R}_{jk})_{i\text{th row}}$ for $i = 1, 2, 3$;
compute $m_{11}, \cdots, m_{23}$ using (10.12)–(10.17);
compute $\lambda_{\mathbf{M}}(\mathbf{R}_{jk})$ using (10.19)–(10.25);
$c_{jk} \leftarrow \sqrt{\lambda_{\mathbf{M}}(\mathbf{R}_{jk})}$;
return c_{jk};

We detail the computation of the gradient (*i.e.*, (10.29)) in the next section. For the hyper-parameters β_1 and β_2, we use the default values given in (Kingma and Ba, 2015): $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For the step size α, we use $\alpha = 0.01$ at the beginning and switch to $\alpha = 0.001$ permanently once the cost increases in five successive iterations. We empirically found that this switching sometimes helps the convergence (see Table 10.5). Alg. 10 summarizes our method.

10.5.3 Gradient computation

We compute the gradient $\mathbf{g}_t$ in (10.29) numerically[1]. This can be done efficiently by slightly perturbing each rotation parameter in $\mathbf{s}_t$ and summing the resulting changes of all the edge costs c_{jk} (10.27) as we traverse the edge set $\mathcal{E}$. Since we need to run Alg. 9 seven times for each edge (*i.e.*, 1 from the unperturbed state, 3×2 from perturbing $\mathbf{R}_j$ and $\mathbf{R}_k$), if there are $n_{\mathcal{E}}$ edges, this method will require $7n_{\mathcal{E}}$ computations of edge costs. To reduce the computation time, we make the following approximation:

$$c_{jk}\left(\mathbf{R}_j(\mathbf{R}_k)_{x+\Delta x}^{\top}\right) - c_{jk}(\mathbf{R}_j\mathbf{R}_k^{\top}) \approx c_{jk}(\mathbf{R}_j\mathbf{R}_k^{\top}) - c_{jk}\left((\mathbf{R}_j)_{x+\Delta x}\mathbf{R}_k^{\top}\right), \tag{10.37}$$

where $(\mathbf{R}_j)_{x+\Delta x}$ and $(\mathbf{R}_k)_{x+\Delta x}$ respectively denote $\mathbf{R}_j$ and $\mathbf{R}_k$ after being perturbed (by the same magnitude) in the x component of the rotation vector. That is, we assume that Δc_{jk} due to $(\mathbf{R}_k)_{x+\Delta x}$ is approximately equal to the negative of Δc_{jk} due to $(\mathbf{R}_j)_{x+\Delta x}$. We make analogous approximations for the perturbations in the y and z component of the rotation vector. By approximating the gradient of $\mathbf{R}_k$ using that of $\mathbf{R}_j$, we reduce the number of edge cost computations from $7n_{\mathcal{E}}$ to $4n_{\mathcal{E}}$. Empirically, we found that this improves the efficiency significantly at a relatively small loss of accuracy (see Table 10.6). Alg. 11 summarizes the procedure for computing the gradient and the total cost.

[1]It is possible to compute it analytically. However, the closed-form expressions involve more operations than the numerical method (see the supplementary material of (Kneip and Lynen, 2013)). We empirically found that this takes approximately 1.8 times longer, while the numerical difference is negligible.

Algorithm 10: Rotation-Only Bundle Adjustment

Input: Initial rotations $\mathbf{R}_1, \cdots, \mathbf{R}_n$, Edges $\mathcal{E}$,
Matched unit bearing vectors $\left(\left(\widehat{\mathbf{f}}_i\right)_j, \left(\widehat{\mathbf{f}}_i\right)_k\right)$
for all $i \in 1, \cdots, m_{(j,k)}$ for all $(j,k) \in \mathcal{E}$,
Number of iterations n_{it}.

Output: Final rotations $\mathbf{R}_1, \cdots, \mathbf{R}_n$.

/* Initialization: */

compute $(\mathbf{F}_{xx})_{jk}, \cdots, (\mathbf{F}_{yz})_{jk}$ for all $(j,k) \in \mathcal{E}$ using (10.7)–(10.11);
compute C using (10.27) and Alg. 9;
obtain $\mathbf{s}_0$ by stacking $\text{Log}(\mathbf{R}_1), \cdots, \text{Log}(\mathbf{R}_n)$ in one column;
$\beta_1 \leftarrow 0.9$; $\beta_2 \leftarrow 0.999$; $\alpha \leftarrow 0.01$; $\boldsymbol{\epsilon} \leftarrow (10^{-8})\mathbf{1}_{3n\times 1}$;
$\mathbf{m}_0 \leftarrow \mathbf{0}_{3n\times 1}$; $\mathbf{v}_0 \leftarrow \mathbf{0}_{3n\times 1}$; $t \leftarrow 0$;

/* Optimization: */

while $t < n_{it}$ **do**
    $t \leftarrow t + 1$;
    compute C and $\mathbf{g}_t$ using Alg. 11;
    Perform (10.30)–(10.36);
    if C *increased in five successive iterations* **then**
        $\alpha \leftarrow 0.001$;

return $\mathbf{R}_1, \cdots, \mathbf{R}_n$;

Algorithm 11: Cost and gradient computation

Input: Current rotations $\mathbf{R}_1, \cdots, \mathbf{R}_n$, Edges $\mathcal{E}$, $(\mathbf{F}_{xx})_{jk}, (\mathbf{F}_{xy})_{jk}, (\mathbf{F}_{xz})_{jk}, (\mathbf{F}_{yy})_{jk}, (\mathbf{F}_{yz})_{jk}$ for all $(j,k) \in \mathcal{E}$

Output: Cost C, Gradient $\mathbf{g}$.

$C \leftarrow 0$; $\mathbf{g} \leftarrow \mathbf{0}_{3n\times 1}$; $\delta \leftarrow 10^{-4}$;

$\boldsymbol{\delta}_x \leftarrow [\delta, 0, 0]^\top$; $\boldsymbol{\delta}_y \leftarrow [0, \delta, 0]^\top$; $\boldsymbol{\delta}_z \leftarrow [0, 0, \delta]^\top$;

$\mathbf{u}_i \leftarrow \mathrm{Log}(\mathbf{R}_i)$ for all $i \in 1, \cdots, n$;

/* Perturb the rotations: */

$(\mathbf{R}_i)_{x+\Delta x} \leftarrow \mathrm{Exp}\left(\mathbf{u}_i + \boldsymbol{\delta}_x\right)$ for all $i \in 1, \cdots, n$;

$(\mathbf{R}_i)_{y+\Delta y} \leftarrow \mathrm{Exp}\left(\mathbf{u}_i + \boldsymbol{\delta}_y\right)$ for all $i \in 1, \cdots, n$;

$(\mathbf{R}_i)_{z+\Delta z} \leftarrow \mathrm{Exp}\left(\mathbf{u}_i + \boldsymbol{\delta}_z\right)$ for all $i \in 1, \cdots, n$;

/* Sum the resulting changes of each c_{jk}: */

for $(j,k) \in \mathcal{E}$ **do**

- $\mathbf{R}_{jk} \leftarrow \mathbf{R}_j \mathbf{R}_k^\top$;
- $(\mathbf{R}_{jk})_{x+\Delta x} \leftarrow (\mathbf{R}_j)_{x+\Delta x} \mathbf{R}_k^\top$;
- $(\mathbf{R}_{jk})_{y+\Delta y} \leftarrow (\mathbf{R}_j)_{y+\Delta y} \mathbf{R}_k^\top$;
- $(\mathbf{R}_{jk})_{z+\Delta z} \leftarrow (\mathbf{R}_j)_{z+\Delta z} \mathbf{R}_k^\top$;
- obtain c_{jk} using Alg. 9 with $\mathbf{R}_{jk}$.
- obtain $(c_{jk})_{x+\Delta x}$ using Alg. 9 with $(\mathbf{R}_{jk})_{x+\Delta x}$.
- obtain $(c_{jk})_{y+\Delta y}$ using Alg. 9 with $(\mathbf{R}_{jk})_{y+\Delta y}$.
- obtain $(c_{jk})_{z+\Delta z}$ using Alg. 9 with $(\mathbf{R}_{jk})_{z+\Delta z}$.
- $\Delta(c_{jk})_x \leftarrow (c_{jk})_{x+\Delta x} - c_{jk}$;
- $\Delta(c_{jk})_y \leftarrow (c_{jk})_{y+\Delta y} - c_{jk}$;
- $\Delta(c_{jk})_z \leftarrow (c_{jk})_{z+\Delta z} - c_{jk}$;
- $\mathbf{g}_{3j-2} \leftarrow \mathbf{g}_{3j-2} + \Delta(c_{jk})_x$;
- $\mathbf{g}_{3j-1} \leftarrow \mathbf{g}_{3j-1} + \Delta(c_{jk})_y$;
- $\mathbf{g}_{3j} \leftarrow \mathbf{g}_{3j} + \Delta(c_{jk})_z$;
- $\mathbf{g}_{3k-2} \leftarrow \mathbf{g}_{3k-2} - \Delta(c_{jk})_x$;
- $\mathbf{g}_{3k-1} \leftarrow \mathbf{g}_{3k-1} - \Delta(c_{jk})_y$;
- $\mathbf{g}_{3k} \leftarrow \mathbf{g}_{3k} - \Delta(c_{jk})_z$;
- $C \leftarrow C + c_{jk}$;

$\mathbf{g} \leftarrow \mathbf{g}/\delta$;

return C and $\mathbf{g}$;

10.6 Results

10.6.1 Evaluation method

We compare our method (henceforth ROBA) against the state-of-the-art rotation averaging method by Chatterjee and Govindu (Chatterjee and Govindu, 2018) (henceforth RA). Both methods are implemented in MATLAB and run on a laptop CPU (Intel i7-4710MQ, 2.8GHz). For RA, we use the code publicly shared by the authors of (Chatterjee and Govindu, 2018)[2] with the $L_{\frac{1}{2}}$ loss function, as recommended in (Chatterjee and Govindu, 2018). We use the output of RA as input to ROBA, so that we can compare RA versus RA + ROBA.

In Alg. 10, the bottleneck is the gradient computation (line 8), where the predominant part is the computation of the edge costs using Alg. 9. To speed up this part, we implement Alg. 9 in a C++ MEX function. We set the number of iterations (n_{it}) to 100 in Alg. 10. Note that, in practice, it would be sensible to adopt some stopping criteria to detect the convergence (*e.g.*, based on the relative change of the total cost or the angular change in the rotations). In our experiment, however, we aim to investigate the convergence behavior of ROBA, so we are agnostic about this heuristics.

Finally, we draw attention to the error metrics for evaluating the rotation estimates $(\mathbf{R}_1, \cdots, \mathbf{R}_n)$. Since they do not share the same reference frame as the ground-truth rotations $(\mathbf{R}_1^{\text{gt}}, \cdots, \mathbf{R}_n^{\text{gt}})$, we must first align them with the ground truth to evaluate the accuracy. Commonly, this is done by rotating them with one of the following rotations:

$$\mathbf{R}_{L1} = \underset{\mathbf{R}_{L1}}{\operatorname{argmin}} \sum_{j=1}^{n} d\left(\mathbf{R}_{L1}, \mathbf{R}_j^\top \mathbf{R}_j^{\text{gt}}\right), \tag{10.38}$$

$$\mathbf{R}_{L2} = \underset{\mathbf{R}_{L2}}{\operatorname{argmin}} \sum_{j=1}^{n} d\left(\mathbf{R}_{L2}, \mathbf{R}_j^\top \mathbf{R}_j^{\text{gt}}\right)^2, \tag{10.39}$$

where $d(\cdot,\cdot)$ denotes the geodesic distance between the two rotations, *i.e.*, $d(\mathbf{R}_1, \mathbf{R}_2) = \arccos((\text{tr}(\mathbf{R}_1\mathbf{R}_2^\top) - 1)/2)$. Note that (10.38) and (10.39) are single rotation averaging problems, and they can be solved using iterative algorithms (Hartley et al., 2011, Hartley, Trumpf, Dai and Li, 2013). Afterwards, we rotate the estimates as follows[3]:

$$\mathbf{R}_j \leftarrow \mathbf{R}_j\mathbf{R}_{L1}, \quad \text{or} \quad \mathbf{R}_j \leftarrow \mathbf{R}_j\mathbf{R}_{L2}. \tag{10.40}$$

Since $\mathbf{R}_{L1}$ minimizes the sum of absolute distances and $\mathbf{R}_{L2}$ minimizes the sum of squares, we call the first method L_1 *alignment* and the second L_2 *alignment*. In our evaluation, we report the mean and median angular errors using these two alignment methods.

[2] `http://www.ee.iisc.ac.in/labs/cvl/research/rotaveraging/`

[3] We must use right multiplication here in order to make sure that $(\mathbf{R}_{12})_{\text{after}} = (\mathbf{R}_1\mathbf{R}_{L1})(\mathbf{R}_2\mathbf{R}_{L1})^\top = \mathbf{R}_1\mathbf{R}_2^\top = (\mathbf{R}_{12})_{\text{before}}$.

Baseline setting	$n = 100$, $n_{\text{cov}} = 50$, $\sigma = 1$px, $d_{\text{min}} = 2$, $d_{\text{max}} = 5$, Views are uniformly spaced by 1 unit on a circle.
Variations	**More points** ($n_{\text{cov}} = 100$), **Fewer views** ($n = 30$), **More views** ($n = 300$), **Closer points** ($d_{\text{max}} = 3$), **Farther points** (d_{max}=10), **Less noise** ($\sigma = 0.5$px), **More noise** ($\sigma = 2$px), **Planar scene** ($d_{\text{min}} = 5$). **Pure rotations** (all views are placed at the origin), **Pure rotations + Planar scene** ($d_{\text{min}} = 5$), **Mixed rotations** (20 groups of views are uniformly spaced by 1 unit on a circle. A group consists of five views at the same location).

Table 10.2: Simulation settings.

10.6.2 Synthetic data

To study how different factors affect our method, we run Monte Carlo simulations in controlled settings: we uniformly distribute n cameras on a circle on the xy-plane such that the neighbors are 1 unit apart. After aligning their optical axes with the z-axis, we perturb the rotations by random angles $\theta \sim \mathcal{U}(0, 20^\circ)$. We set the image size to 640×480 pixels and the focal length to 525 pixels, the same as those in (Sturm et al., 2012). We create 3D points at random distances $d \sim \mathcal{U}(d_{\text{min}}, d_{\text{max}})$ from the xy-plane, ensuring that every neighboring view observes at least n_{cov} points in common. We perturb the image coordinates of the points by $\mathcal{N}(0, \sigma^2)$. For every pair of views observing n_{cov} or more points in common, we estimate the relative pose and the inlying points. If there are at least 10 inliers, we add the pair as an edge in $\mathcal{E}$. Table 10.2 specifies the configuration parameters we set for our simulations. For each parameter setting, we generate 100 different datasets, each with randomly sampled camera rotations, 3D points and 2D measurements.

To obtain the relative rotation estimates, we use the following method: first, we obtain 100 pose samples around the ground-truth relative pose. We do this by perturbing the rotation and the translation by two arbitrary angles (< 20 deg). Then, we evaluate the L_1-optimal angular reprojection errors (Lee and Civera, 2019*a*)[4] for each pose and choose the one that yields the most inliers. This method is similar to the standard method of using a minimal solver (*e.g.*, five-point algorithm (Nistér, 2004)) in RANSAC (Fischler and Bolles, 1981), except that our samples are simulated, not estimated. We use this method because our focus is on the optimization of the rotations and we are agnostic about the relative pose estimation method.

Fig. 10.1 and Table 10.3 present the results in each setting. Notice that ROBA improves the results of RA in all scenarios considered. In Fig. 10.2, we show the evolution of our cost function (10.27) and the rotation error in the baseline setting.

[4]This can be computed using (10.4) and their relation derived in (Lee and Civera, 2020*a*). We do not consider cheirality (Hartley and Zisserman, 2003), because otherwise we would end up discarding many inlying low-parallax points that appear in pure rotations.

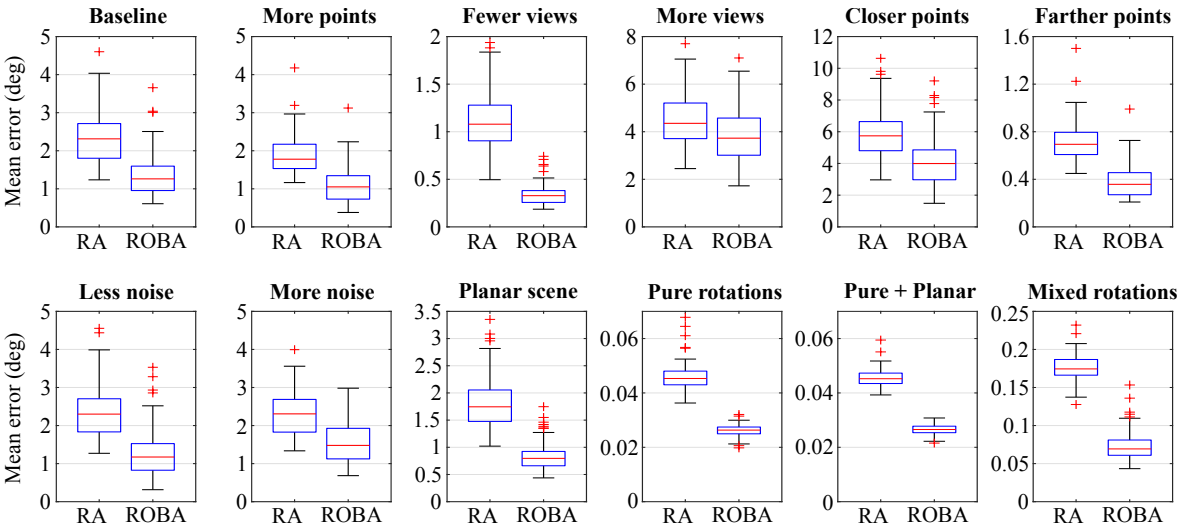


Figure 10.1: Results on the synthetic data (see Table 10.2 for the settings). We compare RA (Chatterjee and Govindu, 2018) and ROBA (initialized by RA) in terms of the mean angular error after the L_1 alignment (see Appendix H for the other error metrics). It shows that ROBA improves the results of RA in all scenarios considered. In particular, the relative error reduction is large for fewer views, farther points and pure rotations, all of which lead to a denser view-graph (see Table 10.3).

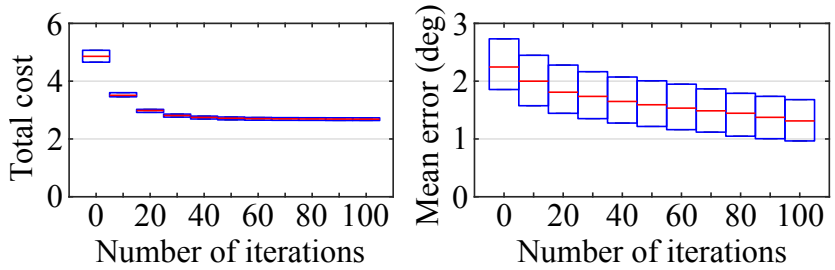


Figure 10.2: Evolution of the cost function (left) and the mean angular errors after the L_1 alignment (right) in the baseline setting. Here, we only show the interquartile range. Notice that while our cost (10.27) seems to plateau after 30–40 iterations, the actual rotation errors continue to decrease. This is discussed in Section 10.7.2.

10.6.3 Real data

We also perform an evaluation on the real-world datasets publicly shared by the authors of (Wilson and Snavely, 2014)[5], which include:

- internal camera calibration and radial distortion data,
- SIFT (Lowe, 2004) feature tracks and their image coordinates,
- estimated relative rotations,
- reconstruction made with `Bundler` (Snavely et al., 2006), consisting of the camera poses and a sparse set of 3D points.[6]

[5] `http://www.cs.cornell.edu/projects/1dsfm/`

[6] The pose estimates are not available for some of the cameras. Also, some of the SIFT features are not associated with the available 3D points.

Settings	$\%_{\mathcal{E}}$	$\overline{e}_{\mathcal{E}}$	$\widetilde{e}_{\mathcal{E}}$	RA	RA + ROBA	$\%_{\text{better}}$
Baseline	6.0%	2.21	1.39	2.31	**1.26**	100%
More points	7.2%	2.51	1.52	1.78	**1.05**	100%
Fewer views	21%	2.23	1.38	1.08	**0.33**	100%
More views	2.0%	2.24	1.40	4.35	**3.73**	100%
Closer points	4.0%	2.57	1.65	5.74	**3.99**	100%
Farther points	10%	1.97	1.22	0.69	**0.36**	100%
Less noise	6.0%	2.25	1.40	2.30	**1.17**	100%
More noise	5.9%	2.23	1.38	2.31	**1.48**	100%
Planar scene	7.4%	2.61	1.56	1.74	**0.80**	100%
Pure rotations	100%	0.89	0.74	0.045	**0.026**	100%
Pure + Planar	100%	0.89	0.74	0.045	**0.027**	100%
Mixed rotations	37%	3.13	1.66	0.17	**0.069**	100%

$\%_{\mathcal{E}}$, $\%_{\text{better}}$: proportion of existing edges and improved results,
$\overline{e}_{\mathcal{E}}$, $\widetilde{e}_{\mathcal{E}}$: mean and median angular errors (in deg) of the relative rotations from all edges.

Table 10.3: Median results of the 100 simulations in each setting. The 5th and 6th columns give the median errors shown in Fig. 10.1. Overall, the denser the view-graph, the more accurate both methods are. ROBA improves upon RA in 1200 out of 1200 simulations.

As in (Lee and Civera, 2020*b*), we use the provided reconstruction model as the ground truth in our experiment. We undistort all image measurements when we process the input. Additionally, some preprocessing is required for these datasets because (1) they do not provide the SIFT IDs of the reconstructed 3D points, and (2) some edges (*i.e.*, camera pairs for which the estimated relative rotations are given) lack covisible 3D points because some of the tracks were disregarded during the reconstruction process (*e.g.*, outliers and low-parallax points).

We address the first issue by projecting each 3D point in its associated images and finding the track that yields the smallest mean reprojection error. Since most reconstructed points have small reprojection errors ($<$1–2 pixels), this can associate the points quite accurately, which we verified qualitatively. We address the second issue by removing all edges with less than 10 covisible 3D points. As a result, some cameras get disconnected from the main view-graph. In our experiment, we disregard all such cameras, as well as those without the available ground truth.

Table 10.4 presents the results. It shows that ROBA offers a consistent and significant gain in accuracy. In Fig. 10.3, we plot the evolution of the relative errors aggregated from all datasets. Table 10.5 and 10.6 present the ablation study results.

Datasets		RA				RA + ROBA (100 iter)				Computation time (s)			
Name	$(n, n_\mathcal{E}, \%_\mathcal{E})$	mn1	md1	mn2	md2	mn1	md1	mn2	md2	RA	Init	Opti	Total
ALM	(577, 96653, 58%)	4.08	1.11	4.74	2.17	**2.35**	**0.42**	**3.20**	**1.47**	16	35	216	267
ELS	(227, 18709, 73%)	2.10	0.50	2.37	0.93	**1.06**	**0.10**	**1.48**	**0.57**	1	5	42	48
GDM	(677, 33662, 15%)	6.05	2.78	6.14	3.12	**2.43**	**1.13**	**2.44**	**1.15**	3	8	76	87
MDR	(341, 23228, 40%)	6.20	1.27	7.23	2.88	**4.42**	**0.61**	**5.71**	**2.50**	2	7	52	61
MND	(480, 51172, 45%)	1.46	0.51	1.59	0.72	**0.82**	**0.26**	**1.01**	**0.50**	4	24	114	142
NTD	(553, 96672, 63%)	2.08	0.64	2.31	0.88	**1.27**	**0.30**	**1.47**	**0.57**	14	60	217	291
NYC	(332, 18787, 34%)	2.87	1.32	2.99	1.40	**1.03**	**0.20**	**1.20**	**0.45**	1	6	42	49
PDP	(338, 24121, 42%)	3.86	0.91	4.67	2.48	**2.18**	**0.33**	**2.92**	**1.64**	1	6	54	61
PIC	(2151, 275895, 12%)	4.14	2.28	4.18	2.38	**1.58**	**0.29**	**1.75**	**0.49**	220	62	617	899
ROF	(1083, 68379, 12%)	2.94	1.52	2.99	1.56	**2.18**	**0.27**	**2.28**	**0.61**	6	26	154	186
TOL	(472, 23379, 21%)	3.83	2.33	3.85	2.38	**1.15**	**0.16**	**1.20**	**0.24**	1	10	53	64
TFG	(5057, 663755, 5%)	3.40	2.34	3.42	2.28	**2.76**	**2.09**	**2.78**	**1.79**	553	153	1488	2194
USQ	(787, 23639, 8%)	5.59	4.03	5.60	4.06	**3.26**	**0.90**	**3.46**	**1.26**	1	7	54	62
VNC	(836, 98999, 28%)	6.12	1.33	7.96	4.06	**4.96**	**0.25**	**7.31**	**3.47**	15	53	221	289
YKM	(437, 27039, 28%)	3.67	1.60	3.72	1.61	**1.66**	**0.19**	**1.74**	**0.28**	1	12	61	74

n: # connected views with known ground truth, $n_\mathcal{E}$: # edges with at least 10 covisible 3D points, $\%_\mathcal{E} = n_\mathcal{E}/n\text{C2}$ in %,
mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,
Init: Initialization (line 1–5 of Alg. 10), Opti: Optimization (line 6–11 of Alg. 10).

Table 10.4: Results on the real data (Wilson and Snavely, 2014). For all datasets, ROBA improves the results of RA (Chatterjee and Govindu, 2018). This is shown across all error criteria, and often, the relative error reduction is significant. See Appendix I for the evolution of the total cost (10.27) and the errors.

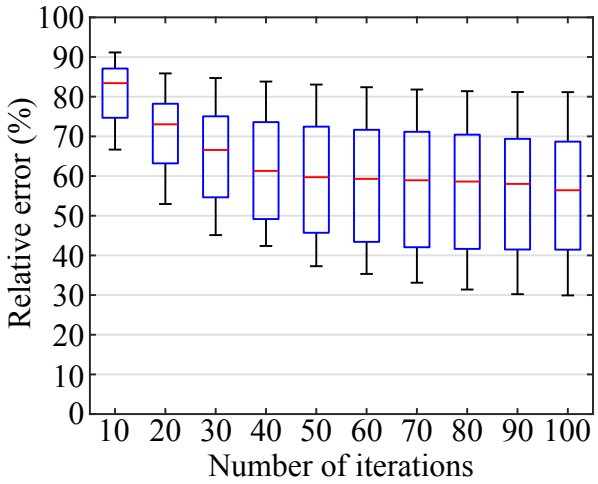


Figure 10.3: Relative errors with respect to the initial values for the real-world datasets (Wilson and Snavely, 2014). Here, we use the mean angular errors after the L_1 alignment (see Appendix I for the other metrics). For more than half of the datasets, the error decreases to less than 60% of the initial value after 50 iterations. After 100 iterations, we observe error reductions up to 70%.

	Baseline		Without $\sqrt{}$ in (10.27)		Without switching α	
Datasets	mn1	mn2	mn1	mn2	mn1	mn2
ALM	**2.35**	**3.20**	2.48	3.35	**2.35**	**3.20**
ELS	1.06	1.48	0.95	1.32	**0.94**	**1.27**
GDM	**2.43**	**2.44**	2.73	2.77	2.45	2.47
MDR	**4.42**	**5.71**	4.82	6.02	**4.42**	**5.71**
MND	**0.82**	**1.01**	0.94	1.11	**0.82**	**1.01**
NTD	**1.27**	1.47	1.47	1.75	1.28	**1.41**
NYC	**1.03**	**1.20**	1.23	1.44	**1.03**	**1.20**
PDP	**2.18**	**2.92**	2.57	3.38	**2.18**	**2.92**
PIC	**1.58**	1.75	2.43	2.73	1.62	**1.72**
ROF	**2.18**	**2.28**	2.92	3.09	3.53	3.83
TOL	1.15	1.20	1.98	2.04	**1.10**	**1.14**
TFG	**2.76**	**2.78**	3.07	3.11	3.53	3.59
USQ	3.26	3.46	4.32	4.41	**3.08**	**3.21**
VNC	**4.96**	**7.31**	5.70	7.93	5.09	7.48
YKM	1.66	1.74	1.85	1.93	**1.59**	**1.66**

Baseline: RA + ROBA (100 iterations),
mn1/2: mean angular error (deg) after L_1/L_2 alignment

Table 10.5: Ablation study I: Undoing the square-rooting in (10.27) (line 4 of Alg. 9) worsens the accuracy for all datasets except ELS. Also, disabling the change of the step size α (line 11 of Alg. 10) significantly worsens the accuracy for ROF and TFG.

	Baseline			Without approximating $\mathbf{g}$		
Datasets	mn1	mn2	Opti	mn1	mn2	Opti
ALM	2.35	3.20	**216**	**2.23**	**3.09**	389
ELS	1.06	1.48	**42**	**0.97**	**1.32**	76
GDM	2.43	2.44	**76**	**2.36**	**2.40**	135
MDR	4.42	5.71	**52**	**4.21**	**5.41**	93
MND	0.82	1.01	**114**	**0.76**	**0.97**	207
NTD	1.27	1.47	**217**	**1.14**	**1.33**	384
NYC	1.03	**1.20**	**42**	**1.00**	1.21	75
PDP	2.18	2.92	**54**	**1.99**	**2.67**	96
PIC	1.58	1.75	**617**	**1.43**	**1.61**	1103
ROF	2.18	2.28	**154**	**1.17**	**1.29**	274
TOL	1.15	1.20	**53**	**1.10**	**1.13**	94
TFG	2.76	2.78	**1488**	**2.54**	**2.55**	2664
USQ	3.26	3.46	**54**	**2.68**	**2.95**	95
VNC	4.96	7.31	**221**	**4.39**	**7.03**	395
YKM	1.66	1.74	**61**	**1.56**	**1.64**	109

Baseline: RA + ROBA (100 iterations),
mn1/2: mean angular error (deg) after L_1/L_2 alignment
Opti: Optimization time (in s) (line 6–11 of Alg. 10).

Table 10.6: Ablation study II: In Alg. 11, computing $\mathbf{g}$ without using the approximation (10.37) improves the accuracy slightly. Exceptions are ROF and USQ where we observe large gains. In all cases, however, this significantly increases the optimization time.

10.7 Discussions

10.7.1 On error metrics

As shown in Table 10.4, the different alignment methods in (10.40) result in different error values. For example, the mean errors are always lower with the L_1 alignment because it yields the theoretically minimal mean error among all possible alignments. Also, the L_1 alignment often gives very small median errors because using both median and the L_1 alignment significantly diminishes the influence of large errors. This was also reported in (Chatterjee and Govindu, 2018). In Section 10.6, we used the mean error after the L_1 alignment as our primary metric due to its moderate sensitivity to large errors.

10.7.2 On convergence

Most of the time, ROBA converges after 30–40 iterations, but it is subject to local minima. Generally, the better the initial rotations, the better the final result. Since rotation averaging depends entirely on the relative rotation estimates, these input rotations must be accurate enough to achieve good results. Empirically, we found that it is much better to have noisier input with fewer outliers than the other way around. We also observed that sometimes a relatively small change in the total cost (10.27) induces a non-negligible change in the rotational accuracy (see Fig. 10.3 and Appendix I). This is somewhat in line with the observation in (Briales et al., 2018) for the two-view case.

10.7.3 On robustness to outliers

In this work, we did not consider outliers in the input, *i.e.*, the image measurements and the initial rotation estimates. We assumed that the outliers in the former have already been dealt with by a robust pose estimator and the latter by robust rotation averaging. Taking into account also the outliers in our optimization is left for future work.

10.7.4 On speed and scalability

As discussed in Section 10.6.1, we fixed the number of iterations at 100. In some of the datasets, we observe that ROBA converges in much fewer iterations (see Appendix I), so implementing the stopping criteria would reduce the runtime. We also note that our code was not highly optimized. Possibly, one could increase the speed by vectorizing line 2 in Al. 9 using SIMD instructions.

As can be seen from Table 10.4, the complexity of ROBA is linear in the number of edges. To enhance the scalability, one could partition the view-graph and perform the local and the global optimization in parallel, as in (Cui et al., 2017, Zhu et al., 2018). This is left for future work.

10.8 Conclusions

In this chapter, we presented rotation-only bundle adjustment, a novel method for estimating the global rotations of multiple views independently of the translations and the scene structure. We formulate the optimization problem by extending the two-view rotation-only method of (Kneip and Lynen, 2013) and solve it using the Adam optimizer (Kingma and Ba, 2015). As we decouple the rotation estimation from the translation and structure estimation, it is completely immune to their inaccuracies. Our evaluation shows that (1) our method is robust to challenging configurations such as pure rotations and planar scenes, and (2) it consistently and significantly improves the accuracy when used after rotation averaging.

Chapter 11

RODIAN: Robustified Median

In this chapter, we propose a robust method for averaging numbers contaminated by a large proportion of outliers. Our method, dubbed RODIAN, is inspired by the key idea of MINPRAN (Stewart, 1995): We assume that the outliers are uniformly distributed within the range of the data and we search for the region that is least likely to contain outliers only. The median of the data within this region is then taken as RODIAN. Our approach can accurately estimate the true mean of data with more than 50% outliers and runs in time $O(n \log n)$. Unlike other robust techniques, it is completely deterministic and does not rely on a known inlier error bound. Our extensive evaluation shows that RODIAN is much more robust than the median and the least-median-of-squares. This result also holds in the case of non-uniform outlier distributions.

11.1 Introduction

Averaging means finding the most representative value of a given set of data points. For one-dimensional numbers, it can be the arithmetic mean, the median or other measures of central tendency. All these measures have different properties, one of which is the robustness to outliers. This is an important property to consider because outliers can severely degrade the averaging accuracy if not handled properly. Robust averaging is a useful technique in a wide variety of domains, including pattern recognition (Hauberg et al., 2014, Li et al., 2009, Lewis et al., 1999), image processing (Vaish et al., 2006, Chaudhury and Singer, 2012, Khatoonabadi and Bajic, 2013), 3D computer vision (Gesto Diaz et al., 2015, Lee and Civera, 2020*b*, Cui and Tan, 2015), biomedical engineering (Leonowicz et al., 2005, Leski, 2002, Kotowski et al., 2019), economics/econometrics (Bryan et al., 1997, Mykland and Zhang, 2016, Dias Curto, 2021), information science (Angelov and Yager, 2013, Garcin et al., 2009, Beliakov et al., 2016), environmental studies (Zhang and Zhang, 1996, Merchant et al., 2018), geochemistry (Rock et al., 1987, Rock, 1988), forensic science (Illes and Boué, 2013), psychology (S. Courvoisier and Renaud, 2010) and database research (Hellerstein, 2008).

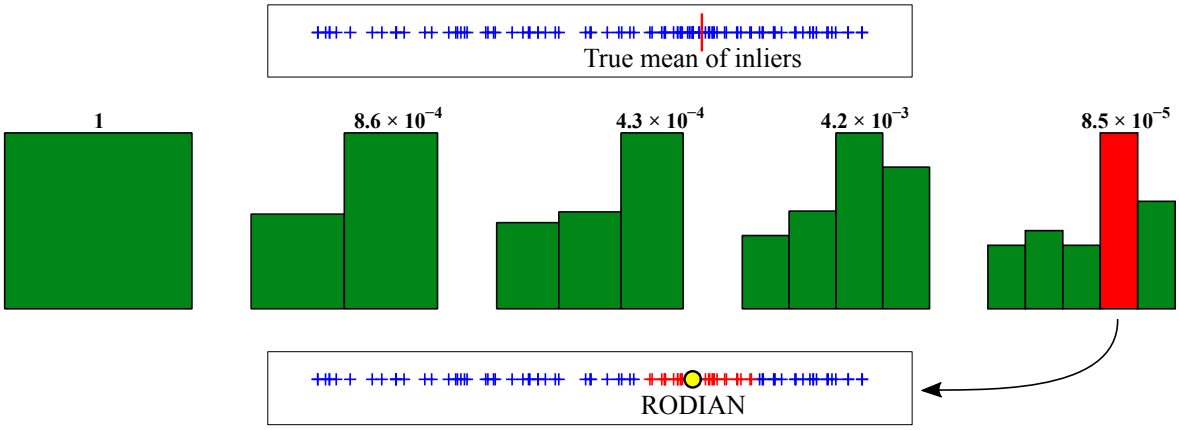


Figure 11.1: **Top:** 100 numbers used as input for a toy example. 80% of the numbers are outliers, uniformly distributed between 0 and 100. The inliers follow $\mathcal{N}(70, 5^2)$. **Middle:** We build multiple histograms with a varying number of bins (1–5 in this example). For each histogram, we find the highest bin and evaluate the probability of this occurring purely by chance, *assuming that the outliers are uniformly distributed.* These probabilities of randomness are given above each histogram. We find the bin that produces the smallest probability (shown in red). **Bottom:** RODIAN is the median of the numbers in this bin.

If the given data set contains a small number of outliers, it may be sufficient to use the median, as, contrary to the mean, it is robust up to a certain outlier ratio (Davis and Jr., n.d.). The median can be considered as a specific case of the alpha-trimmed mean with $\alpha = 0.5$. The alpha-trimmed mean (Bednar and Watt, 1984) of n numbers is defined as the arithmetic mean after truncating the largest $\frac{\alpha}{2}n$ and the smallest $\frac{\alpha}{2}n$ elements. This method assumes that the outliers are likely to be located at the high and low ends of the sorted data. As a result, it fails when a large number of outliers are located mostly on one side of the long tails.

To handle such cases, a more elaborate method should be used. One popular example is the maximum likelihood-type estimator (M-estimator) (Huber, 1981). Paired with iteratively reweighted least squares (IRLS) (Holland and Welsch, 1977), it can effectively downweight the influence of outliers. However, M-estimators, such as the Huber function, often require a control parameter to be carefully tuned to the inlier error distribution. Also, their robustness strongly depends on the initial seed, and accurate initialization in the presence of many outliers is already a non-trivial problem in and of itself.

Another popular robust estimation method is RANSAC (Fischler and Bolles, 1981). It involves random sampling, but it can be made deterministic for the 1D averaging problem if we simply pick every number as a sample once. While this method can handle a very large number of outliers, it incurs a computational cost of $O(n^2)$ and requires the prior knowledge of the inlier error bound.

The least-median-of-squares (LMedS) (Rousseeuw, 1984), on the other hand, does not require any prior knowledge. For the 1D averaging problem, the LMedS can be obtained by finding the data point that yields the smallest median deviation from the rest. This would involve $O(n^2 \log n)$ computations. Like the median, the LMedS has a breakdown point of 50%.

Another method that does not rely on a known inlier error bound is MINPRAN (Stewart,

1995). This method is more robust than the LMedS, as it can handle more than 50% outliers. However, it is slower than the LMedS and has a random nature.

In this chapter, we propose RODIAN, a novel robust measure of central tendency. Our method is inspired by the core idea of MINPRAN (Stewart, 1995): We assume that the outliers follow a uniform distribution and find the median in the bounded region that is least likely to contain outliers only. Unlike MINPRAN, however, our method is deterministic and runs in time $O(n \log n)$.[1] Also, unlike RANSAC (Fischler and Bolles, 1981) and Huber-like cost functions (Huber, 1981), no parameter tuning is needed to account for different inlier distributions. Our experiments show that RODIAN can handle more than 50% outliers, outperforming the median and the LMedS (Rousseeuw, 1984) in terms of robustness.

11.2 Method

11.2.1 Main Idea

Suppose that we are given a set of numbers. Each number is either an inlier or an outlier, but we do not know which is which. Assuming that the inliers are scattered around a certain number μ, how can we estimate μ from this noisy, outlier-contaminated data? Our approach is to find the most densely populated region in the data and take the median value in that region. Now the question is how to determine this region.

One simple heuristic approach is to build a histogram and find the tallest bin. Then, the edges of this bin correspond to the upper and lower bounds of the densest region. This is a reasonable approach, but there is one problem: The histogram can be constructed in many different ways. If we constrain the lower edge of the first bin to be the minimum value and the upper edge of the last bin to be the maximum value, then we can obtain multiple histograms by varying the number of bins. So, which histogram is the right one to use?

Our answer to this question is that we choose the histogram with the bin that is least likely to have occurred by chance. For any histogram, each of its bins have its associated probability of randomness. For example, if all bins have the exact same height, we can deduce that the numbers are uniformly distributed, and thus random in this sense. By the same token, if one of the bins is significantly taller than the others, then it is unlikely that it occurred due to the randomness. In other words, this very tall bin has a low probability of randomness.

Essentially, what we propose is to build multiple histograms with the different numbers of bins, evaluate the probability of randomness associated with the tallest bin of each histogram, and choose the one that yields the smallest probability of randomness. This is because the minimum probability of randomness implies the maximum probability of containing mostly inliers. This process is illustrated in Fig. 11.1.

[1] This is made possible because, unlike MINPRAN, we do not use random sampling and we fix the number of inlier bounds we evaluate (by fixing the number of histograms, as will be explained in Section 11.2).

The remaining question is how exactly we compute this probability of randomness. Basically, we adopt a similar idea proposed by Stewart (Stewart, 1995) and compute the probability in a binomial distribution, assuming that the random outliers are uniformly distributed across the entire range. Note that if a trial has a probability of success p, the probability of obtaining k successes from n trials is given by

$$P(k) = \binom{n}{k} p^k (1-p)^{n-k}. \tag{11.1}$$

In our problem, p is the probability of a random outlier falling inside the tallest bin, k is the frequency of this bin, and n is the size of the data. Since an outlier can fall inside any other bins with an equal probability, we have $p = 1/b$ where b is the number of bins. Therefore, the probability of randomness associated with the bin containing k numbers is given by

$$P(k) = \binom{n}{k} \left(\frac{1}{b}\right)^k \left(1 - \frac{1}{b}\right)^{n-k}. \tag{11.2}$$

In summary, we vary b, find k by building a histogram, compute $P(k)$, and repeat this process until we find the value of b that leads to the smallest $P(k)$. In the next section, we discuss several strategies we came up with to improve the efficiency of the algorithm.

We note that, in contrast to our approach, MINPRAN (Stewart, 1995) uses the probability that *at least* k data points fall within the inlier region, which is technically different from (11.2). Empirically, we found that using this probability instead of (11.2) leads to similar results, but at a slower speed.

11.2.2 Implementation Details

1. How many histograms do we consider?:
According to the pigeonhole principle, if we set the number of bins to $n - 1$, at least one bin will contain more than one number. Therefore, one could find the theoretically optimal number of bins, b^*, by varying b from 1 to $n - 1$, searching for the minimum $P(k)$ in Eq. (11.2). However, in our experiment described in Section 11.3, we found that setting $b > 20$ hardly makes any difference in the final accuracy. We also empirically found that there is no need to try all integers between 1 and 20, as similar results could be obtained faster with $b \in \{2, 3, 4, 5, 7, 9, 11, 14, 17, 20\}$.

2. How to accelerate the histogram building process:
Building multiple histograms one by one can take a long time. For efficiency, we precompute a table that matches the bin edges and the bin indices of all the histograms. This process is explained Fig. 11.2. In order to reuse this table on any data, it must be agnostic of the input. To this end, we normalize the input data such that its range becomes $[0, 1]$. This way, all edges get predetermined values between 0 and 1.

11.2.3 Summary

1. Precompute a table, as described in Fig. 11.2, with $A = 0$, $B = 0.5$, $C = 1$, etc. For the number of bins, we use $b \in \{2, 3, 4, 5, 7, 9, 11, 14, 17, 20\}$.
2. Sort and normalize the input such that its range is between 0 and 1, *i.e.,*

$$x_i \leftarrow \frac{x_i - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \quad \text{for } i = 1, 2, \cdots, n. \tag{11.3}$$

3. For each data point, use the precomputed table to find the corresponding bin index in each histogram.
4. For each histogram, find the frequency of the tallest bin and the associated probability of randomness (Eq. (11.2)).
5. Find the histogram that leads to the smallest probability.
6. In that histogram, find the median of the numbers that fall inside the tallest bin.
7. Unnormalize this median. The final value is RODIAN.

In Step 4, we discard a histogram if multiple bins have the same maximum frequency. If, by any chance, all histograms are discarded, we simply take the median of the original input.

Time analysis: The time complexity of Step 2 and 6 is $O(n \log n)$. The other steps run in either $O(1)$ or $O(n)$. Hence, the total time complexity is $O(n \log n)$.

11.3 Results

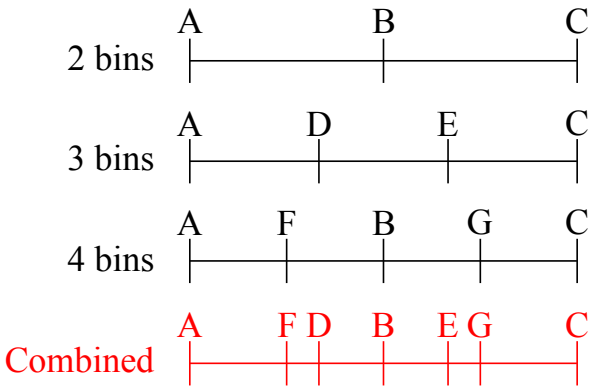
We compare RODIAN with three other methods:

1. Median,
2. Least-median-of-squares (LMedS) (Rousseeuw, 1984), estimated as the data point with the smallest median (squared) distance to the rest, *i.e.*,

$$\text{LMedS} = \underset{x_i}{\text{argmin}} \underset{j}{\text{med}} \left(x_i - x_j\right)^2, \tag{11.4}$$

3. Median of the tallest bin of a fixed histogram, obtained in the following way: (i) Build a histogram with a fixed number of bins, (ii) Collect all the numbers that fall inside the tallest bin, (iii) Compute their median.

We present the results on synthetic data with two different outlier distributions: a uniform distribution (Fig. 11.3) and a mixture of a uniform and a Gaussian distribution (Fig. 11.4). In both cases, RODIAN outperforms the rest in terms of robustness. Especially, Fig. 11.4 shows that even though we assumed a uniform outlier distribution in our derivation of RODIAN, it can still handle non-uniform outliers relatively well if σ_{outlier} is larger than σ_{inlier}.



(a)

		A-F	F-D	D-B	B-E	E-G	G-C
# bins	2	1	1	1	2	2	2
	3	1	1	2	2	3	3
	4	1	2	2	3	3	4

(b)

Figure 11.2: This example illustrates the process of precomputing the tabular data for building histograms: **(a)** Suppose that we want to build three histograms with 2, 3 and 4 bins. We collect the bin edges of all histograms (without duplicates) and sort them in a single array, *i.e.*, $[A, F, D, B, E, G, C]$. **(b)** We assign the corresponding bin index to each region bounded by two successive edges in the combined array. For example, a number between edge F and D would fall inside the 1^{st} bin in the 2-bin and 3-bin histogram, and 2^{nd} bin in the 4-bin histogram. These indices are given in the F-D column. By checking which region a number falls in, we can immediately find the corresponding bin index in each histogram.

Table 11.1 compares the accuracy of RODIAN and the fixed-histogram approach for low to moderate outlier ratios. It again demonstrates the advantage of using RODIAN over a fixed histogram. In Fig. 11.5, we plot the mean computation times of the median, LMedS and RODIAN. We observe that RODIAN is much more scalable than the LMedS.

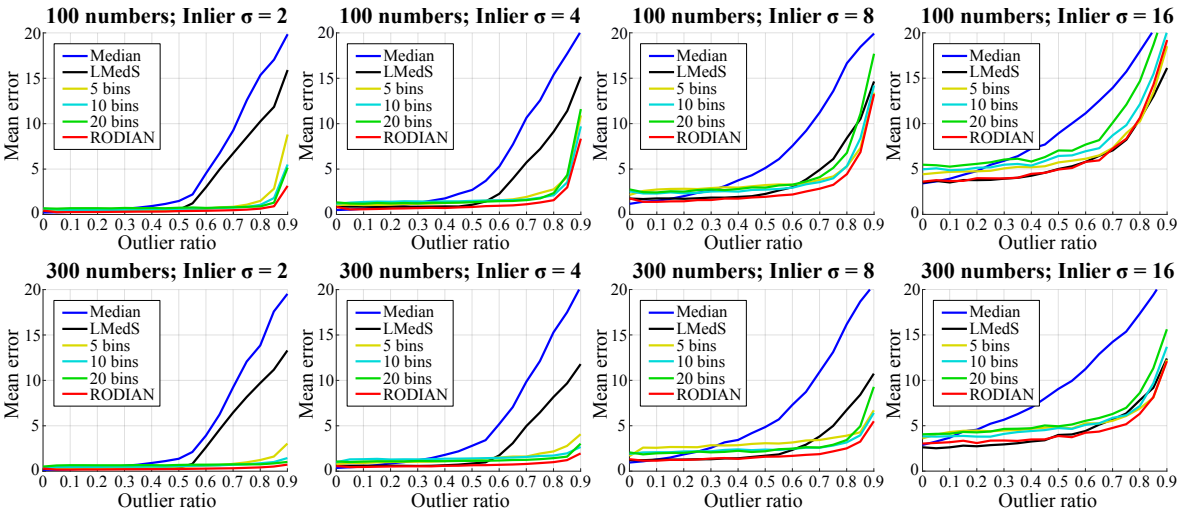


Figure 11.3: Mean error comparison under a **uniform outlier distribution**: The evaluation is carried out with 100 numbers (top row) and 300 numbers (bottom row). The numbers are generated such that the inliers follow $\mathcal{N}(\mu, \sigma^2)$ where $0 < \mu < 100$ and $\sigma \in \{2, 4, 8, 16\}$, and the outliers follow $\mathcal{U}(0, 100)$. If any number is outside a range $[0, 100]$, it is removed and regenerated until all numbers are within this range. Each data point in the graph represents the mean error of 1000 independent runs. It shows that, across different inlier noise levels, RODIAN is the most robust to outliers. When the inlier noise is small, the breakdown point of RODIAN is well over 80% (see the first column). Also, RODIAN is generally more accurate than the fixed-histogram approach.

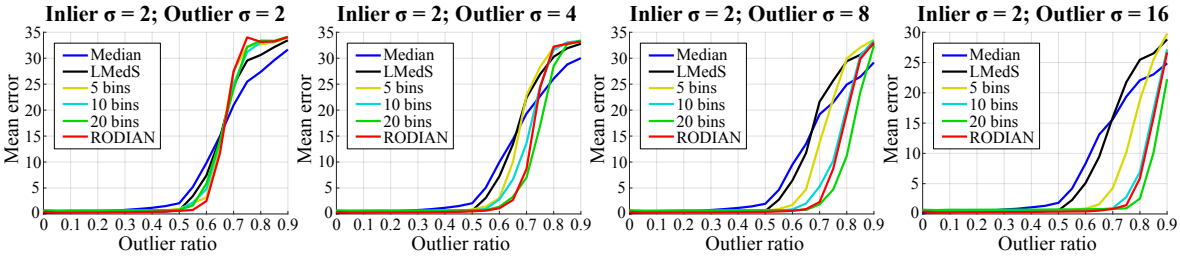


Figure 11.4: Mean error comparison under a **uniform + Gaussian outlier distribution**: The evaluation is carried out with 100 numbers. The numbers are generated using the same procedure described in Fig. 11.3, except that half of the outliers now follow $\mathcal{N}(\mu_{\text{outlier}}, \sigma^2_{\text{outlier}})$, $0 < \mu_{\text{outlier}} < 100$. We observe that RODIAN has a higher breakdown point than the median and the LMedS. When the outlier ratio is very high, the fixed-histogram approach with 20 bins produces smaller errors than RODIAN. However, they both are well beyond the breakdown point there and comparisons are meaningless, as errors are driven by outliers.

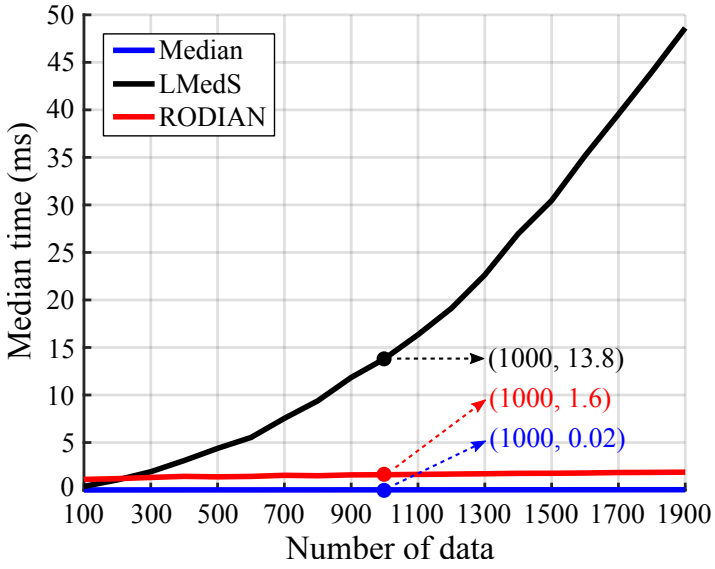


Figure 11.5: Median computation times (ms) of 1000 runs: The time complexity of the median, LMedS and RODIAN is $O(n \log n)$, $O(n^2 \log n)$ and $O(n \log n)$, respectively. The median is always the fastest, and RODIAN is faster than the LMedS when #data $\gtrapprox$ 200. All methods are implemented in MATLAB and run on a laptop with an Intel's 4th Gen i7 CPU (2.8 GHz).

Outliers		5 bins	10 bins	20 bins	30 bins	50 bins	RODIAN
Uniform[a]	0%	0.54	0.64	0.69	0.74	0.80	**0.42**
	10%	0.36	0.50	0.69	0.61	0.61	**0.26**
	20%	0.39	0.51	0.69	0.63	0.62	**0.28**
	30%	0.43	0.54	0.70	0.64	0.63	**0.30**
	40%	0.48	0.57	0.71	0.67	0.66	**0.32**
	50%	0.56	0.61	0.72	0.68	0.68	**0.36**
Gaussian[b]	0%	0.54	0.63	0.69	0.73	0.79	**0.42**
	10%	0.46	0.64	0.63	0.64	0.68	**0.28**
	20%	0.47	0.64	0.65	0.66	0.69	**0.30**
	30%	0.47	0.65	0.68	0.67	0.72	**0.33**
	40%	0.50	0.71	0.77	0.77	0.85	**0.38**
	50%	4.36	2.33	2.29	1.93	2.49	**1.34**

Table 11.1: Using fixed histogram vs. RODIAN: We generate 100 numbers within a range $[0, 100]$ and average them using either a fixed histogram or RODIAN. This is repeated 10000 times. RODIAN produces the smallest mean error.
[a]Inliers follow $\mathcal{N}(\mu, 2^2)$ with $0 < \mu < 100$.
[b]Inliers and outliers follow $\mathcal{N}(\mu_1, 2^2)$ and $\mathcal{N}(\mu_2, 4^2)$ with $0 < \mu_1, \mu_2 < 100$. (Note: This dataset is different from that of Fig. 11.4.)

11.4 Limitation

The main limitation of RODIAN is that its accuracy slightly drops when there are too few outliers (see Table 11.1). This happens because the densest region of the inlier distribution is not always well aligned with the location of the true mean. While this is certainly not a favorable property, the average error increase is relatively small (around 10% of the standard deviation of the inliers in Table 11.1). We believe that this is a tolerable level in outlier-prone situations, which is the main domain we target in this work.

One potential solution is to detect when the data is outlier-free and switch to a traditional median. If the type of the inlier distribution is known (*e.g.,* Gaussian), one can use a statistical test to check if the whole data follow the inlier distribution (*e.g.,* normality test (Thode, 2002, Shapiro and Wilk, 1965)). In this work, however, we aim to make our method generalizable to any types of inlier distribution as long as it is unimodal. Discerning outlier-free data in such a general scenario is left for future work.

11.5 Conclusion

In this chapter, we presented RODIAN, a novel method for averaging outlier-contaminated numbers. It consists of two main steps: (1) determine a bounded region in the range that would contain mostly inliers, and (2) find the median within that region. The key idea of the first step is to assume a uniform outlier distribution and search for the region that is least likely to have occurred due to outliers. Unlike MINPRAN (Stewart, 1995), where a similar idea was used, our method is deterministic and runs in time $O(n \log n)$. Unlike RANSAC (Fischler and Bolles, 1981) and Huber-like loss functions (Huber, 1981), we do not need to tune a control parameter to adapt to different inlier error distributions. Finally, unlike the median and the LMedS (Rousseeuw, 1984), RODIAN can handle more than 50% outliers. An extensive evaluation demonstrates its excellent robustness, versatility and scalability.

Chapter 12

What's Wrong With the Absolute Trajectory Error?

One of the main limitations of the commonly used Absolute Trajectory Error (ATE) is that it is highly sensitive to outliers. As a result, in the presence of just a few outliers, it often fails to reflect the varying accuracy as the inlier trajectory error or the number of outliers varies. In this chapter, we propose an alternative error metric for evaluating the accuracy of the reconstructed camera trajectory. Our metric, named Discernible Trajectory Error (DTE), is computed in four steps: (1) Shift the ground-truth and estimated trajectories such that both of their geometric medians are located at the origin. (2) Rotate the estimated trajectory such that it minimizes the sum of geodesic distances between the corresponding camera orientations. (3) Scale the estimated trajectory such that the median distance of the cameras to their geometric median is the same as that of the ground truth. (4) Compute the distances between the corresponding cameras, and obtain the DTE by taking the average of the mean and root-mean-square (RMS) distance. This metric is an attractive alternative to the ATE, in that it is capable of discerning the varying trajectory accuracy as the inlier trajectory error or the number of outliers varies. Using the similar idea, we also propose a novel rotation error metric, named Discernible Rotation Error (DRE), which has similar advantages to the DTE. Furthermore, we propose a simple yet effective method for calibrating the camera-to-marker rotation, which is needed for the computation of our metrics. Our methods are verified through extensive simulations.

12.1 Introduction

Reconstructing a set of camera poses from images (and other sensors) is an important problem in computer vision and robotics. It has direct application to autonomous navigation, photogrammetry, and AR/VR. For this reason, significant research endeavors have been devoted to improving the performance of reconstruction algorithms. Active research areas include odometry (Engel et al., 2018, Leutenegger et al., 2015, Zhang and Singh, 2014),

simultaneous localization and mapping (SLAM) (Lee and Civera, 2019*b*, Campos et al., 2021, Teed and Deng, 2021), visual localization (Sattler et al., 2018, Lynen et al., 2020, Toft et al., 2022) and structure-from-motion (Agarwal et al., 2011, Schönberger and Frahm, 2016, Moulon et al., 2013).

In pursuit of developing better reconstruction algorithms, it is also very important to ask what error metric should be used to evaluate the accuracy of the results. If the ground-truth data for the camera poses is given, the current *de facto* standard in the robotics community is the Absolute Trajectory Error (ATE). An early work that analyzed this metric is (Sturm et al., 2012). Since the ATE is a single number metric that could be intuitively understood and easily used for comparison, it has quickly become a popular choice of metric for the the evaluation of camera localization systems.

The basic idea of the ATE is to translate, rotate, and optionally, scale the estimated trajectory (*i.e.*, 3D positions of the cameras), such that it is as closely aligned to the ground-truth trajectory as possible. This is done by minimizing the root mean square (RMS) of the distances between the corresponding cameras in each trajectory, typically using Horn's (Horn, 1987) and Arun's (Arun et al., 1987) methods. The ATE is then given by the optimal RMS value.

Note that the trajectory alignment relies on the optimization under the L_2 norm. This means that the ATE is inevitably sensitive to outliers. This, in and of itself, is not really a problem, because we want the error metric to clearly indicate whether or not any localization failure has occurred throughout the trajectory. That said, the actual problem is that, with just a few outliers, it quickly starts losing its sensitivity to the inlier trajectory error and the number of outliers. In other words, the ATE reacts much less sharply to the changes in these two factors when the estimation contains just a few percent of outliers.

For benchmarking, this is certainly not a desirable property as an error metric. For example, suppose we are comparing several localization methods on the same dataset and all of the methods happen to produce grossly erroneous estimates at the same set of locations in the trajectory. In this case, it would be a loss if we cannot know which methods are more accurate within the "good" part of the trajectory. Also, if some of the methods fail more times than others, we would want to know in which of them this happens. Such information is even more relevant when developing one's own system. Suppose that a baseline algorithm fails to reconstruct some of the camera poses because the dataset involves difficult scenarios. Suppose we make certain changes in the algorithm that enhance its overall accuracy, but we are still unable to save it from failing. In this case, we would want the error metric to reflect the overall increase in accuracy, even if there is no change in robustness. In this regard, ATE may not be the most suitable choice of metric.

In this chapter, we propose a novel alternative to the ATE that addresses this problem. Our metric, called Discernible Trajectory Error (DTE), is computed using robust trajectory alignment based on the idea of median. We show that, unlike the ATE, the DTE can reliably discern the varying accuracy as the inlier trajectory error or the number of outliers varies.

Also, we extend this idea to rotations and propose a novel rotation error metric called Discernible Rotation Error (DRE), which has similar advantages to the DTE. Furthermore, we propose a simple method for calibrating the camera-to-marker rotation, which needs to be known for the computation of our metrics.

12.2 Related Work

An early work by Sturm et al. (Sturm et al., 2012) provides a discussion comparing the ATE and the relative error. The relative error can be useful for odometry systems (Küemmerle et al., 2009, Burgard et al., 2009, Geiger et al., 2012), but it cannot be used for evaluating the reconstruction from unordered image collections. Zhang and Scaramuzza (Zhang and Scaramuzza, 2018*a*) analyze the properties of the ATE and the relative errors for visual(-inertial) systems. They point out that the ATE decreases when more cameras are used for its computation. In (Zhang and Scaramuzza, 2018*b*), the same authors propose a continuous-time approach for trajectory evaluation, tackling the potential issue of imperfect temporal association between the ground truth and the estimation.

To our knowledge, in the field of quantitative trajectory evaluation, no previous study has addressed the sensitivity issue caused by outliers (as described in Section 12.1).

12.3 Notation

When a variable $\mathbf{v}$ involves physical measurements, we denote it by $\widetilde{\mathbf{v}}$. If a rotation matrix $\mathbf{R}$ has the angle θ and the unit axis of rotation $\widehat{\mathbf{v}}$, then $\mathbf{R} = \mathrm{Exp}(\theta\widehat{\mathbf{v}})$ where $\mathrm{Exp}(\cdot)$ is the mapping defined by Rodrigues' formula (Forster, Zhang, Gassner, Werlberger and Scaramuzza, 2017). The geodesic (or angular) distance between two rotation matrices $\mathbf{R}_j$ and $\mathbf{R}_k$ is defined as the angle of the rotation $\mathbf{R}_j\mathbf{R}_k^\top$ and is denoted by $d(\mathbf{R}_j, \mathbf{R}_k)$. Let $\mathbf{p}_i$ be a 3D point in the reference frame i. When this point is viewed in another reference frame j, its coordinates are given by $\mathbf{p}_j = s_{ji}\mathbf{R}_{ji}\mathbf{p}_i + \mathbf{t}_{ji}$, where $s_{ji} \in \mathbb{R}$, $\mathbf{R}_{ji} \in \mathbf{SO}(3)$ and $\mathbf{t}_{ji} \in \mathbb{R}^3$ are respectively the relative scale, the rotation matrix and the translation vector that relate the reference frame i and j. In the reference frame j, the position of the reference frame i is defined as the position of its origin, which is given by $s_{ji}\mathbf{R}_{ji}\mathbf{0} + \mathbf{t}_{ji} = \mathbf{t}_{ji}$. Likewise, the position of the reference frame j in frame i is given by $\mathbf{t}_{ij}$. We define the orientation of the reference frame i with respect to frame j as follows: Imagine three arrows fixed to the reference frame i, each pointing in the positive x, y and z direction of frame i. Let $\mathbf{x}_j$, $\mathbf{y}_j$ and $\mathbf{z}_j$ be the vectors representing these arrows viewed in the reference frame j, *i.e.*, $\mathbf{x}_j = s_{ji}\mathbf{R}_{ji}[1,0,0]^\top$, $\mathbf{y}_j = s_{ji}\mathbf{R}_{ji}[0,1,0]^\top$ and $\mathbf{z}_j = s_{ji}\mathbf{R}_{ji}[0,0,1]^\top$. Then, the orientation of the reference frame i with respect to frame j is defined by $[\widehat{\mathbf{x}}_j, \widehat{\mathbf{y}}_j, \widehat{\mathbf{z}}_j] = \mathbf{R}_{ji}\mathbf{I}_{3\times3} = \mathbf{R}_{ji}$. Likewise, the orientation of the reference frame j with respect to i is given by $\mathbf{R}_{ij}$. For clarity, we summarize the notations in Tab. 12.1.

	Interpretation based on the transformation of a 3D point	Interpretation based on the view from another reference frame
$\mathbf{R}_{ij}$	The rotation that, together with the translation $\mathbf{t}_{ij}$, transforms a 3D point from the reference frame j to i: $\mathbf{x}_i = \mathbf{R}_{ij}\mathbf{x}_j + \mathbf{t}_{ij}$.	The orientation of the reference frame j with respect to frame i. In other words, the three columns of $\mathbf{R}_{ij}$ correspond to the directions of the standard basis of frame j viewed in frame i.
$\mathbf{t}_{ij}$	The translation that, together with the rotation $\mathbf{R}_{ij}$, transforms a 3D point from the reference frame j to i: $\mathbf{x}_i = \mathbf{R}_{ij}\mathbf{x}_j + \mathbf{t}_{ij}$.	The position of the reference frame j in frame i. More specifically, it is the position of the origin of frame j viewed in frame i.

Table 12.1: Different interpretations of $\mathbf{R}_{ij}$ and $\mathbf{t}_{ij}$.

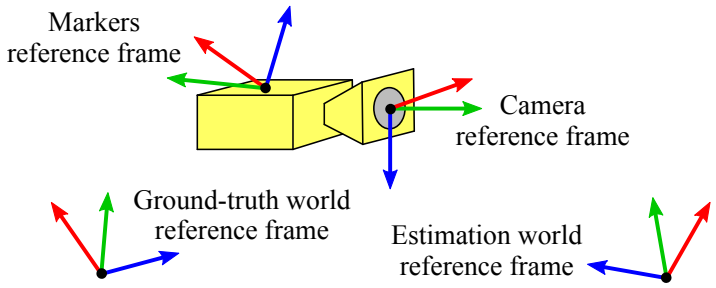


Figure 12.1: Reference frames that are relevant for describing the camera pose.

12.4 Preliminaries: ATE

Suppose that we have measured the 3D positions of n cameras in the ground-truth world reference frame, *i.e.*, $\widetilde{\mathbf{t}}_{gc,i}$ for $i = 1, 2, \cdots, n$. Also, suppose that we have used some computer vision algorithm and obtained the up-to-scale[1] positions of those cameras in the estimation world reference frame, *i.e.*, $\mathbf{t}_{ec,i}$ for $i = 1, 2, \cdots, n$. Now, the ATE between those two sets of camera positions is defined as follows (Sturm et al., 2012):

$$\text{ATE} = \min_{s_{ge},\mathbf{R}_{ge},\mathbf{t}_{ge}} \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|\widetilde{\mathbf{t}}_{gc,i} - (s_{ge}\mathbf{R}_{ge}\mathbf{t}_{ec,i} + \mathbf{t}_{ge})\|^2}. \tag{12.1}$$

Essentially, this is the minimum RMS of the distances between the ground-truth trajectory and the estimated trajectory after aligning the latter to the former using $\mathbf{SIM}(3)$ transformation.

Since it is impossible to measure $\widetilde{\mathbf{t}}_{gc}$ directly, we use reflective markers (or other measuring tools) instead to track the position of the cameras. As a result, the reference frame defined by the markers may not be exactly the same as the camera reference frame (see Fig. 12.1). These two reference frames are related by the following transformation of a 3D point: $\mathbf{p}_m = s_{mc}\mathbf{R}_{mc}\mathbf{p}_c + \mathbf{t}_{mc}$. Combining this with the transformation between the markers and

[1] In this work, we assume that the positions are estimated up to scale. For applications where the positions are estimated at an absolute scale, one can simply fix the relative scale to 1 in the remainder of the paper.

the ground-truth world reference frame, we get

$$\mathbf{p}_g = \mathbf{R}_{gm}\mathbf{p}_m + \mathbf{t}_{gm} \tag{12.2}$$

$$= s_{mc}\underbrace{\mathbf{R}_{gm}\mathbf{R}_{mc}}_{\mathbf{R}_{gc}}\mathbf{p}_c + \underbrace{\mathbf{R}_{gm}\mathbf{t}_{mc} + \mathbf{t}_{gm}}_{\mathbf{t}_{gc}}. \tag{12.3}$$

This means that $\widetilde{\mathbf{t}}_{gc,i}$ in (12.1) can be obtained as follows:

$$\widetilde{\mathbf{t}}_{gc,i} = \widetilde{\mathbf{R}}_{gm,i}\mathbf{t}_{mc} + \widetilde{\mathbf{t}}_{gm,i} \tag{12.4}$$

In practice, if $\mathbf{t}_{mc}$ is unknown, we often assume that it is zero, as the markers are usually placed near the camera[2].

In (Horn, 1987) and (Arun et al., 1987), the closed-form optimal solution for (12.1) was derived. We refer to the original works for the derivations. In the following, we summarize the steps needed to obtain the solution:

1. Compute the centroids of the two trajectories (*i.e.*, the ground truth, $\widetilde{\mathbf{t}}_{gc,i}$, and the estimation, $\mathbf{t}_{ec,i}$ for all i):

$$\overline{\mathbf{t}}_{gc} = \frac{\sum_{i=1}^{n}\widetilde{\mathbf{t}}_{gc,i}}{n}, \quad \overline{\mathbf{t}}_{ec} = \frac{\sum_{i=1}^{n}\mathbf{t}_{ec,i}}{n}, \tag{12.5}$$

   Let $\mathbf{G}$ and $\mathbf{E}$ be the $3 \times n$ matrices that have $\widetilde{\mathbf{t}}_{gc,i} - \overline{\mathbf{t}}_{gc}$ and $\mathbf{t}_{ec,i} - \overline{\mathbf{t}}_{ec}$ as their columns, respectively.

2. Compute the singular value decomposition of $\mathbf{E}\mathbf{G}^\top$:

$$\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top = \mathrm{SVD}\left(\mathbf{E}\mathbf{G}^\top\right). \tag{12.6}$$

   Then, the optimal rotation $\mathbf{R}_{ge}$ is given by

$$\mathbf{R}^*_{ge} = \mathbf{V}\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & \mathrm{sign}\left(\det\left(\mathbf{V}\mathbf{U}^\top\right)\right)\end{bmatrix}\mathbf{U}^\top. \tag{12.7}$$

3. The optimal relative scale s_{ge} is given by

$$s^*_{ge} = \frac{\sum_{i=1}^{n}\mathbf{G}_i^\top\mathbf{R}^*_{ge}\mathbf{E}_i}{\sum_{i=1}^{n}\mathbf{E}_i^\top\mathbf{E}_i}, \tag{12.8}$$

[2]It would be problematic to assume that $\mathbf{t}_{mc}$ is zero when it is non-negligible. Here is a simple example: Imagine we have a long rod. We attach the camera at one end, A, and the markers at the other end, B. Now, suppose that we move A in a circle while keeping B at the center of the circle. Replacing $\widetilde{\mathbf{t}}_{gc,i}$ with $\widetilde{\mathbf{t}}_{gm,i}$ in (12.1), we get ATE $= 0$ because $\widetilde{\mathbf{t}}_{gm,i} = \mathbf{0}$, and thus, $s_{ge} = 0$ and $\mathbf{t}_{ge} = \mathbf{0}$. This does not happen if we use (12.4) without assuming that $\mathbf{t}_{mc} = \mathbf{0}$.

where $\mathbf{G}_i$ and $\mathbf{E}_i$ denote the ith column of each matrix.

4. The optimal translation $\mathbf{t}_{ge}$ is given by

$$\mathbf{t}^*_{ge} = \bar{\mathbf{t}}_{gc} - s^*_{ge}\mathbf{R}^*_{ge}\bar{\mathbf{t}}_{ec}. \tag{12.9}$$

The ATE is then obtained by plugging these results into (12.1). Note that all of these steps is vulnerable to potential outliers in the estimated trajectory. In the next section, we propose an alternative error metric that can better handle outliers.

12.5 Discernible Trajectory Error (DTE)

The key idea behind our method is to robustify each step in aligning the two trajectories. The proposed steps are as follows:

1. Compute the geometric medians of the two trajectories (*i.e.*, the ground truth, $\widetilde{\mathbf{t}}_{gc,i}$, and the estimation, $\mathbf{t}_{ec,i}$ for all i):

$$\mathbf{t}^*_{gc} = \underset{\mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^{n} \|\widetilde{\mathbf{t}}_{gc,i} - \mathbf{t}\|, \tag{12.10}$$

$$\mathbf{t}^*_{ec} = \underset{\mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^{n} \|\mathbf{t}_{ec,i} - \mathbf{t}\|. \tag{12.11}$$

   These can be found efficiently using the Weiszfeld algorithm (Weiszfeld, 1937, Weiszfeld and Plastria, 2009).

2. Find the optimal rotation $\mathbf{R}_{ge}$ that rotates the estimated trajectory such that it minimizes the sum of geodesic distances between the corresponding camera orientations in each trajectory:

$$\mathbf{R}^*_{ge} = \underset{\mathbf{R}_{\text{align}}}{\operatorname{argmin}} \sum_{i=1}^{n} d\left(\widetilde{\mathbf{R}}_{gc,i}, \mathbf{R}_{\text{align}}\mathbf{R}_{ec,i}\right). \tag{12.12}$$

   Since the geodesic distance is invariant to rotation, we can rearrange this equation into the following form:

$$\mathbf{R}^*_{ge} = \underset{\mathbf{R}_{\text{align}}}{\operatorname{argmin}} \sum_{i=1}^{n} d\left(\widetilde{\mathbf{R}}_{gc,i}\mathbf{R}^\top_{ec,i}, \mathbf{R}_{\text{align}}\right). \tag{12.13}$$

   This is the single rotation averaging problem on $\mathbf{SO}(3)$ under the L_1 norm, and $\mathbf{R}^*_{ge}$ corresponds to the geodesic median of the rotations $\widetilde{\mathbf{R}}_{gc,i}\mathbf{R}^\top_{ec,i}$ for all i. We solve this using the method proposed in (Hartley et al., 2011). Note that the ground-truth camera

orientations $\widetilde{\mathbf{R}}_{gc,i}$ are obtained as follows:

$$\widetilde{\mathbf{R}}_{gc,i} = \widetilde{\mathbf{R}}_{gm,i}\widetilde{\mathbf{R}}_{mc}, \tag{12.14}$$

where $\widetilde{\mathbf{R}}_{mc}$ is the rotation between the camera and the markers reference frame. For now, we assume that this is already known (see Section 12.7 for its calibration).

3. Compute the relative scale s_{ge} as follows:

$$s_{ge} = \frac{\underset{i}{\text{med}} \, \|\widetilde{\mathbf{t}}_{gc,i} - \mathbf{t}^*_{gc}\|}{\underset{i}{\text{med}} \, \|\mathbf{t}_{ec,i} - \mathbf{t}^*_{ec}\|}. \tag{12.15}$$

Note that the numerator and the denominator correspond to the median absolute deviations (MAD) (Hampel, 1974) to the geometric median of the ground-truth and estimated camera positions, respectively.

4. Compute the translation $\mathbf{t}_{ge}$ as follows:

$$\mathbf{t}_{ge} = \mathbf{t}^*_{gc} - s_{ge}\mathbf{R}^*_{ge}\mathbf{t}^*_{ec}. \tag{12.16}$$

5. The DTE is then defined by

$$\text{DTE} = \frac{\epsilon_{\text{mean}} + \epsilon_{\text{rms}}}{2}, \tag{12.17}$$

where

$$\epsilon_{\text{mean}} = \frac{1}{n}\sum_{i=1}^{n} \|\widetilde{\mathbf{t}}_{gc,i} - \left(s_{ge}\mathbf{R}^*_{ge}\mathbf{t}_{ec,i} + \mathbf{t}_{ge}\right)\|, \tag{12.18}$$

$$\epsilon_{\text{rms}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|\widetilde{\mathbf{t}}_{gc,i} - \left(s_{ge}\mathbf{R}^*_{ge}\mathbf{t}_{ec,i} + \mathbf{t}_{ge}\right)\|^2}. \tag{12.19}$$

That is, the DTE is the average of the mean and the RMS trajectory error after aligning the estimated trajectory using the transformation obtained from the previous steps.

In the following, we provide the rationale behind our approach: First, we use the geometric median of the trajectory instead of the centroid when computing the translation. This is because the geometric median is significantly more robust to outliers than the centroid (Lopuhaa and Rousseeuw, 1991). As a result, our translation (12.16) is more robust than that used for the ATE (12.9).

Second, we obtain the rotation by aligning the camera orientations using the geodesic median in $\mathbf{SO}(3)$. For the same reason as previously stated, this is more robust than (12.7) which assigns equal weights to both inliers and outliers.

Third, we compute the scale using the MAD, instead of the variance (as in (12.8)) or the standard deviation (as in the symmetrical scale proposed in (Horn, 1987)). The inherent robustness of the MAD (Hampel, 1974) makes our scale estimate (12.15) more reliable than (12.8) which is highly sensitive to outliers.

Finally, instead of simply taking the mean or the RMS error after alignment, we take their average in (12.17). This way, the DTE behaves favorably in terms of sensitivity to the inlier trajectory accuracy and the number of outliers. We elaborate on this behavior in Section 12.8.2.

12.6 Discernible Rotation Error (DRE)

We can apply the similar principle when evaluating the accuracy of the rotations separately: We define the Discernible Rotation Error (DRE) as the average of the mean and RMS rotation error after aligning the camera orientations using the same rotation $\mathbf{R}_{ge}^{*}$ defined in (12.12). Like the DTE, the DRE is capable of discerning the varying accuracy as the inlier orientation error or the number of outliers varies. The experimental results are provided in Section 12.8.3.

12.7 Calibration of Camera-to-Marker Rotation

So far, we have assumed that the camera-to-marker rotation $\widetilde{\mathbf{R}}_{mc}$ in (12.14) is known. In this section, we propose a simple method for calibrating this rotation. The idea is to optimize two rotations simultaneously by solving the following problem:

$$\underset{\mathbf{R}_{\text{align}},\, \widetilde{\mathbf{R}}_{mc}}{\operatorname{argmin}} \sum_{i=1}^{n} d\left(\widetilde{\mathbf{R}}_{gm,i}\widetilde{\mathbf{R}}_{mc}\mathbf{R}_{ec,i}^{\top}, \mathbf{R}_{\text{align}}\right). \tag{12.20}$$

Notice that this is a simple extension of (12.13) which now involves another unknown variable, $\widetilde{\mathbf{R}}_{mc}$. Assuming that the estimated rotations $\mathbf{R}_{ec,i}$ are reasonably accurate and that the camera orientations are not in any degenerate configurations (which will be discussed later), solving this problem will lead to an accurate estimate of $\widetilde{\mathbf{R}}_{mc}$ (and $\mathbf{R}_{\text{align}}$ if needed).

Our strategy for solving (12.20) is as follows:

1. Set $\mathbf{R}_{\text{est}} = \mathbf{I}_{3\times 3}$ and $\theta_{\max} = 360^{\circ}$.
2. Set $\theta = k\theta_{\max}$ where k is a random number between 0 and 1. Also, set $\widehat{\mathbf{v}}$ to a random unit vector.
3. Update $\widetilde{\mathbf{R}}_{mc} \leftarrow \operatorname{Exp}(\theta\widehat{\mathbf{v}})\mathbf{R}_{\text{est}}$.
4. Since $\widetilde{\mathbf{R}}_{mc}$ is now fixed, (12.20) becomes the single rotation averaging problem on $\mathbf{SO}(3)$ under the L_1 norm. Solve it using the method proposed in (Hartley et al.,

2011).

5. Repeat Step 2–4 one thousand times. Whenever (12.20) yields a smaller cost than the smallest value ever obtained so far, update $\mathbf{R}_{\text{est}} \leftarrow \widetilde{\mathbf{R}}_{mc}$.

6. Repeat Step 2–5 four more times. For each further iteration, update $\theta_{\max}$ to 30°, 10°, 3° and 1°, respectively. Return the final $\widetilde{\mathbf{R}}_{mc}$ in the end.

Essentially, we update the estimate of $\widetilde{\mathbf{R}}_{mc}$ by searching for a better rotation within the ball of a certain radius around the current estimate. In the outer loop, the radius of the ball is decreased at each iteration, and in the inner loop, the search is done by simple random sampling within the ball.

A word of caution: The calibration method described above assumes that the estimated rotations $\mathbf{R}_{ec,i}$ are accurate. Indeed, our experiment shows that their overall accuracy directly affects the calibration result (see Section 12.8.4). That said, since our method is based on the robust rotation averaging algorithm (Hartley et al., 2011), the calibration error will be well within a tolerable limit as long as the estimation is reasonably accurate. For example, the calibration error is mostly less than 1° when the average estimation error is less than 10°. To prevent a potential calibration failure in the first place, one may consider performing the calibration procedure in a separate session using, for example, a checkerboard pattern.

Another important aspect that must be taken into account during calibration is that certain camera orientations can lead to degeneracy. Specifically, this happens when all cameras have the same fixed orientation, or when they all rotate around same axis. The proof is given in Appendix J.

12.8 Results

12.8.1 Comparison between ATE and DTE

We compare the ATE and the DTE in simulation: We generate 100 cameras with random rotations and positions inside a $1 \times 1 \times 1$ cube centered at the origin. Then, we corrupt this ground truth to obtain the estimated trajectory. First, we perturb the camera positions with Gaussian noise $\mathcal{N}(0, \sigma^2)$ where $\sigma = 0, 0.01, 0.02, \ldots, 0.1$ unit. Their rotations are perturbed by Gaussian noise with $\sigma = 5^\circ$. Next, we turn some of the cameras into outliers by assigning to them random rotations and positions inside a $10 \times 10 \times 10$ cube centered at the origin. We vary the number of outliers between 0 and 10 in our experiment.

Next, we rotate the entire trajectory by a random rotation, scale it by a random number, and shift it by a random translation vector. The resulting trajectory is taken as our estimation. Note that here we assume that $\widetilde{\mathbf{R}}_{mc}$ in (12.14) is the identity matrix and that this is known in advance through calibration. We compute the DTE and the ATE by comparing the estimated trajectory with the ground truth. This procedure gives us errors in 121 settings because we

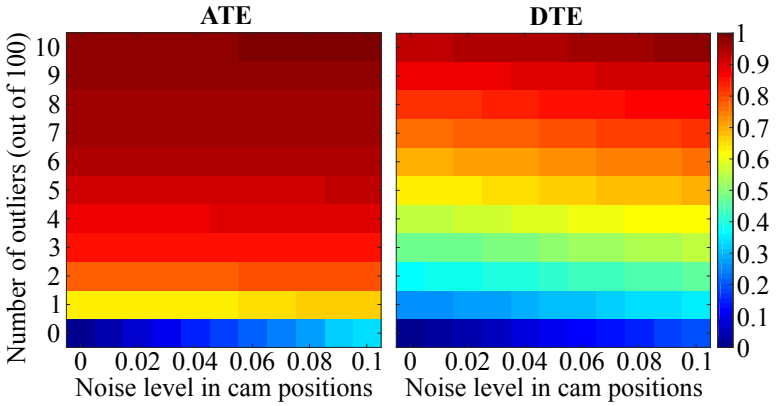


Figure 12.2: Each of the colored blocks represents the normalized ATE and DTE obtained with the given number of outliers and noise level in camera positions. In the presence of outliers, the DTE shows a more pronounced gradation than the ATE. This means that it is better at capturing the varying trajectory accuracy as the number of outliers and the noise level varies.

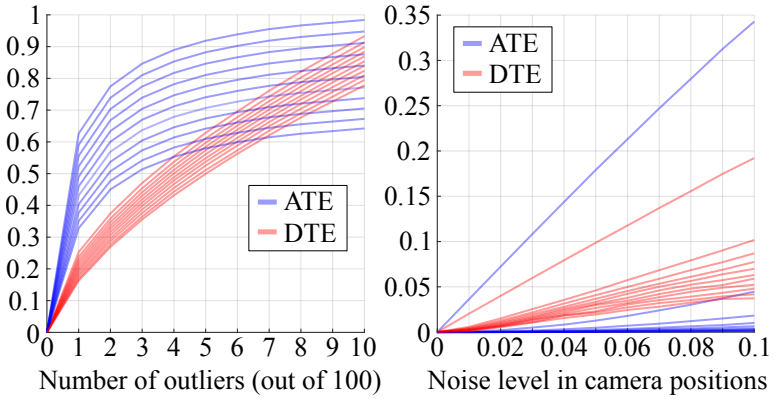


Figure 12.3: **[Left]** The errors corresponding to each column of Fig. 12.2, showing the effect of a varying number of outliers. We shift each column values such that the error in the outlier-free case is 0. As the number of outliers increases, the ATE curves flatten, which signals a decreasing level of sensitivity. In contrast, the DTE maintains a relatively high level of sensitivity. **[Right]** The errors corresponding to each row of Fig. 12.2, showing the effect of a varying noise level in camera positions. We shift each row values such that the error in the zero-noise case is 0. As the number of outliers increases, the slope of the ATE curve drastically decreases, and with just three outliers, it becomes almost insensitive to the noise level. In contrast, the DTE can still maintain a moderate level of sensitivity, even with 10 outliers.

have 11 different numbers of outliers and 11 different noise levels in camera positions. We repeat this 1000 times with independently generated ground-truth trajectories.

In order to aggregate the results from these 1000 runs, we normalize the 121 ATEs and 121 DTEs from each run by dividing them by the maximum ATE and DTE, respectively. This gives us, per run, 121 errors up to the magnitude of 1, which can be averaged over the 1000 runs.

Fig. 12.2 shows the results. We observe that the ATE quickly becomes insensitive to the number of outliers and the noise level as the outlier ratio increases. On the other hand, the DTE can discern the effect of a varying number of outliers and noise level relatively well across the whole domain of settings. This contrast in their sensitivity is more clearly shown in Fig. 12.3, where we plot the error values of each row and column of Fig. 12.2 separately.

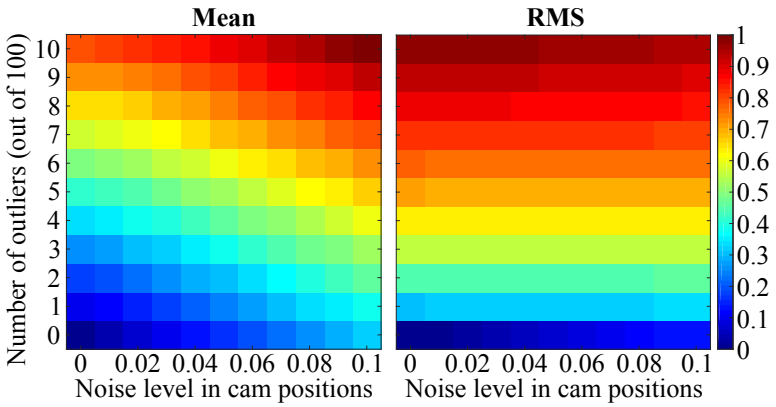


Figure 12.4: Each of the colored blocks represents the normalized mean and RMS trajectory errors obtained with the given number of outliers and noise level in camera positions. The mean error shows a high level of sensitivity to both the number of outliers and the noise level, whereas the RMS error is sensitive to the number of outliers, but not so much to the noise level.

12.8.2 Why take the average of the mean and the RMS?

In this section, we explain why we take the average of the mean and RMS error in (12.17). Fig. 12.4 shows the results if we choose either only the mean or only the RMS. In the left panel, we observe that the mean error is sensitive both to the number of outliers and the noise level. This is not a problem per se, but it is when the error reacts similarly to the change in the number of outliers and the change in the noise level. For many robotics applications, the estimator's robustness is generally considered more critical than its overall accuracy. Therefore, we want the error metric to be substantially more sensitive to a camera tracking failure than to a slight loss in accuracy. For the RMS error, on the other hand, this is not an issue (see the right panel of Fig. 12.4), but it is rather insensitive to the inlier noise level. To overcome the respective weaknesses of the mean and the RMS error, we combine the two together. This way, the mean error provides the sensitivity to the noise level, while the RMS error accentuates the effect of outliers.

We point out that (12.17) can be seen as an instance of a more general form involving variable weights, *i.e.,*

$$\frac{1}{1+\alpha}\epsilon_{\text{mean}} + \frac{\alpha}{1+\alpha}\epsilon_{\text{rms}}, \tag{12.21}$$

where $\alpha \geq 0$ is the relative weight of ϵ_{rms} compared to ϵ_{mean}. The greater the value of α, the stronger the influence of outliers. While the appropriate value for α was 1 in our experiment, it is also possible to choose a different value for other applications, depending on the noise/outlier sensitivity.

12.8.3 Evaluation of the DRE

We compare the DRE and the following error metrics:

- Median-1 (Chatterjee and Govindu, 2018), Mean-1 (Lee and Civera, 2021), RMS-1: These are respectively the median, mean and RMS rotation error after aligning the

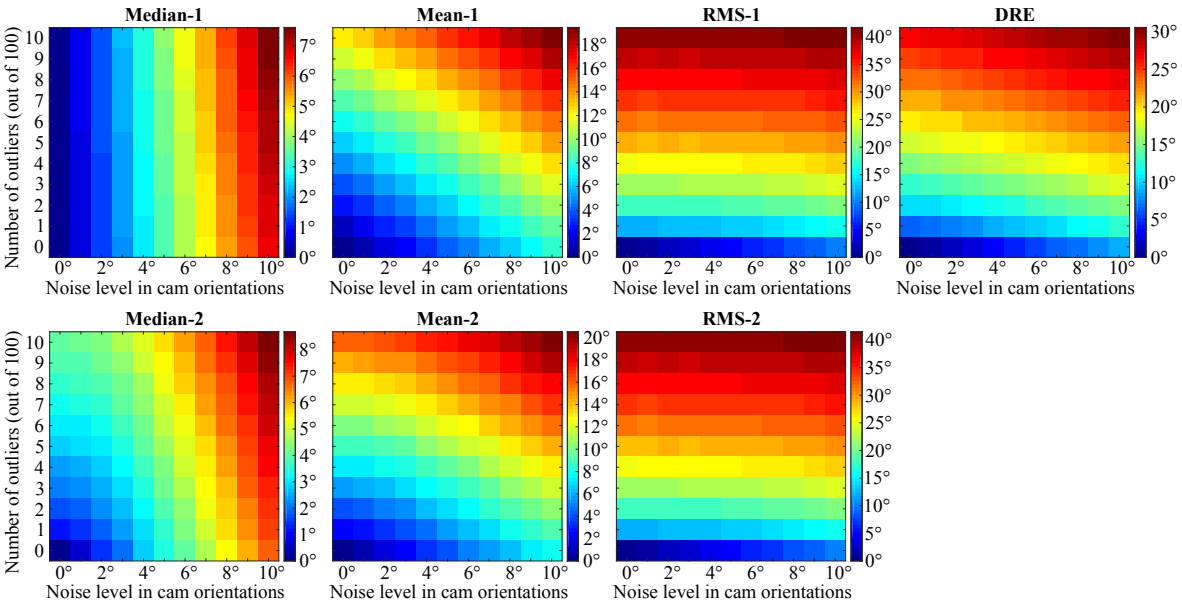


Figure 12.5: Each of the colored blocks represents the rotation error obtained with the given number of outliers and noise level in camera orientations. Compared to the RMS errors, the median and mean errors have relatively low sensitivity to the number of outliers and relatively high sensitivity to the noise level. The DRE is sufficiently sensitive to the noise level, but more to the number of outliers, which is a desirable property.

orientations by minimizing the geodesic distances under the L_1 norm.

- Median-2 (Chatterjee and Govindu, 2018), Mean-2 (Lee and Civera, 2021), RMS-2 (Lee and Civera, 2022): These are the counterparts of the previous three, obtained using the alignment under the L_2 norm.

Note that the DRE is basically the average of Mean-1 and RMS-1 errors. To compare these error metrics in a controlled manner, we generate 100 random ground-truth orientations and obtain the estimated orientations by (1) rotating the ground truth by some random rotation, (2) perturbing them with Gaussian noise, and (3) turning some of them into outliers by setting them to random orientations. We vary the noise level up to 10° and the outlier ratio up to 10%. For each configuration, we run 1000 independent simulations. The average results are shown in Fig. 12.5. From this figure, we can see that the DRE has similar advantages to the DTE.

12.8.4 Evaluation of our calibration method

To evaluate our calibration method in Section 12.7, we use a similar setup in simulation: First, we generate 100 random ground-truth orientations, $\widetilde{\mathbf{R}}_{gm,i}$, and two random rotations, $\mathbf{R}_{\text{align}}$ and $\widetilde{\mathbf{R}}_{mc}$. Next, we generate the estimated orientations, $\mathbf{R}_{ec,i}$, by setting them to $\mathbf{R}_{\text{align}}^\top \widetilde{\mathbf{R}}_{gm,i} \widetilde{\mathbf{R}}_{mc}$. Finally, we perturb these orientations with Gaussian noise and turn some of them into outliers by setting them to random orientations. We vary the noise level up to 10° and the outlier ratio up to 20%. For each setting, we generate 100 independent datasets. Then, using the ground-truth and estimated orientations as input, we solve (12.20) and obtain the estimates of $\mathbf{R}_{\text{align}}$ and $\widetilde{\mathbf{R}}_{mc}$.

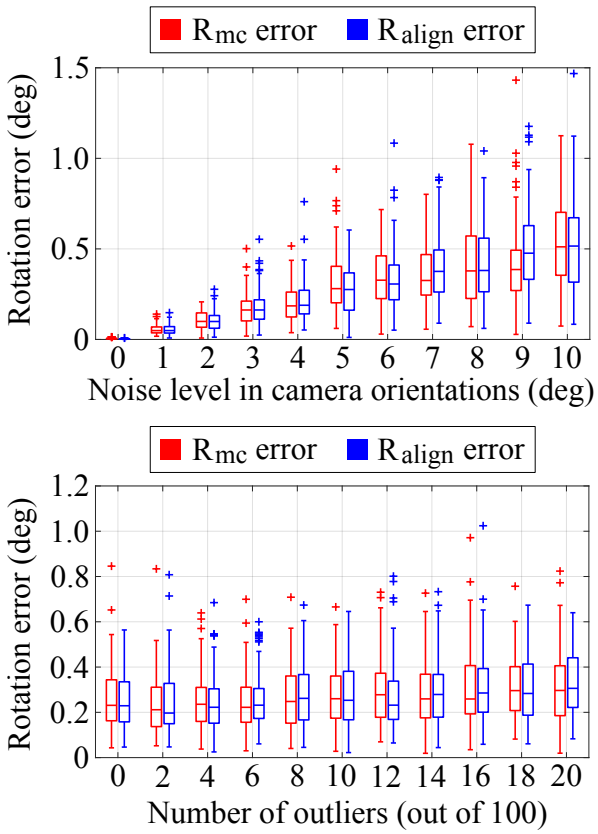


Figure 12.6: **[Top]** Calibration errors under a varying noise level in the estimated camera orientations. We fix the number of outliers to be 5 out of 100. The median error increases somewhat linearly to the noise level. **[Bottom]** Calibration errors under a varying number of outliers in the estimated camera orientations. We fix the noise level to be 5°. The effect of outliers is relatively small when the outlier ratio is moderate (*i.e.*, at least up to 20%).

Fig. 12.6 presents the results. It shows that the median calibration error is less than 0.5° when the estimation is reasonably accurate (*i.e.*, within 10° margin). Also, our method seems to be robust to a moderate amount of outliers, probably due to the inherent robustness of the L_1 optimization.

Additionally, to check if our estimates converged to a near-optimal solution, we compare them to the solution we would have got if we had used the ground truth as the initial seed. We obtain this solution using the same method described in Section 12.7, with the difference that we set $\mathbf{R}_{\text{est}}$ to true $\widetilde{\mathbf{R}}_{mc}$ and $\theta_{\max}$ to 1° in Step 1 and skip Step 6. The result is shown in Fig. 12.7. We see that the angular differences between our estimates and the ground-truth-initialized ones are never above 0.04°. This demonstrates that our calibration algorithm has a reliable convergence behavior.

12.9 Discussion

Possibly the biggest limitation of the DTE is that it requires the knowledge of the camera-to-marker rotation $\widetilde{\mathbf{R}}_{mc}$ when obtaining the ground-truth camera orientations using (12.14). Although our calibration method in Section 12.7 works quite well under a reasonable amount

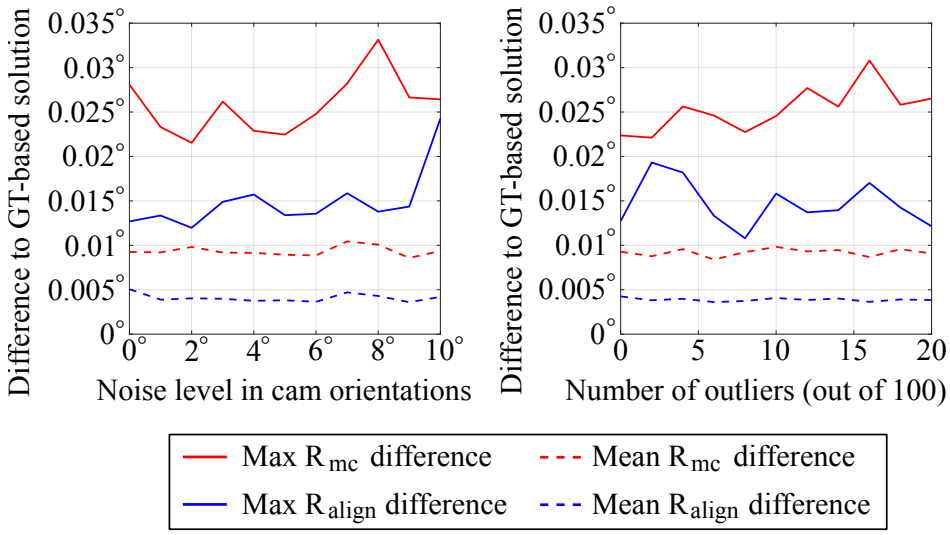


Figure 12.7: The angular difference between our calibration results and the results we get when we use the ground truth as the initial seed: **[Left]** We vary the noise level in the camera orientations, while fixing the number of outliers to be 10 out of 100. **[Right]** We vary the number of outliers, while fixing the noise level in the camera orientations to be 5 deg.

of noise and outliers, the fact that it requires an additional data inquisition step may be considered bothersome, especially when compared to the relatively simple process of computing the ATE. Worse yet, if the ground-truth data had already been collected and if it does not contain the markers' orientations at all, then it would be simply impossible to compute the DTE.

That said, it would be an overstatement to say that this limitation alone is a major deterrent to the wide adoption of the DTE. First, there are cases where the ground-truth camera orientations $\widetilde{\mathbf{R}}_{gc,i}$ in (12.13) are directly available. Examples include synthetic datasets (*e.g.*, ICL-NUIM (Handa et al., 2014), TartanAir (Wang et al., 2020)) and real-world datasets where pseudo ground truth data is obtained by a structure-from-motion system[3] (*e.g.*, 1DSfM (Wilson and Snavely, 2014), MVS (Zhou et al., 2018)). Second, some of the most popular public datasets do provide the information about the relative rotation between the camera and the markers (or some sort of trackers with the same role). Examples include the TUM RGB-D (Sturm et al., 2012), KITTI (Geiger et al., 2013) and EuRoC MAV (Burri et al., 2016) dataset. Lastly, even if a public dataset provides only the markers' orientations without their relation to the camera's, it is still possible to estimate the camera-to-marker rotation using our calibration method in Section 12.7 as long as the cameras are sufficiently rotated with a varying axis direction and the estimated trajectory is reasonably accurate. To ensure sufficient estimation accuracy, one could check if the ATE is small enough. If it is too large, then an accurate part of the trajectory can be extracted manually and used for calibration.

[3]We refer to (Brachmann et al., 2021) for an excellent discussion of this approach in visual relocalization.

12.10 Conclusion

In this chapter, we proposed DTE, a novel metric for evaluating the trajectory estimation accuracy. Unlike the ATE whose sensitivity quickly deteriorates in the presence of a few outliers, the DTE can robustly capture the varying accuracy as the inlier trajectory error or the number of outliers varies. This is made possible by aligning the estimated trajectory to the ground truth using a robust method. The key difference from the ATE is that we compute each of the translation, rotation and scale involved in the trajectory transformation using the geometric, geodesic and arithmetic median, respectively. Also, instead of simply taking the mean or the RMS trajectory error as in the ATE, we take the average of the mean and the RMS. This helps the DTE behave favorably in terms of sensitivity to the inlier trajectory error, as well as the number of outliers. We also proposed the DRE, a rotation-only metric that is based on the similar idea and has similar advantages to the DTE. Lastly, we proposed a simple yet effective method for calibrating the camera-to-marker rotation, which is necessary for the computation of our metrics. In our calibration method, we showed that degeneracy occurs when the relative camera rotations have the same axis of rotation (up to a sign). We evaluated our metrics and the calibration method through extensive simulations, demonstrating the effectiveness of our approach.

Chapter 13

Conclusions

13.1 Summary of Contributions

In this thesis, we studied two aspects that continue to be relevant in many problems of multiple view geometry: the error criteria and the robustness to outliers. First, we reviewed the error criteria and cost functions that are commonly used in some of the multiview geometry problems and asked ours *'Why do we optimize what we optimize?'* Specifically, we discussed their implications in practice and proposed novel methods that either combine them or replace them with better alternatives. Second, we proposed multiple novel ideas in order to better handle outliers in geometric reconstruction and achieve state-of-the-art accuracy and robustness in challenging scenarios with high outlier ratio.

The problems we studied in this thesis are monocular SLAM, two-view and multiview triangulation, single and multiple rotation averaging, rotation-only bundle adjustment, number averaging, and quantitative trajectory evaluation.

For monocular SLAM, we proposed LCSD-SLAM, a novel hybrid method that has the robustness of direct visual odometry and the map-reusing capability of feature-based SLAM (Chapter 2). The proposed system consists of two loosely-coupled modules running in parallel: One module uses a direct method of (Engel et al., 2018) to track the current camera pose with respect to a local semi-dense map. The other module uses a feature-based method of (Mur-Artal et al., 2015) to build a globally consistent sparse feature-based map. Our evaluation on two public datasets showed that LCSD-SLAM outperforms the state-of-the-art in terms of tracking accuracy and robustness.

For two-view triangulation, our contributions are two-fold. First, in Chapter 3, we derived the exact closed-form solutions that guarantee global optimality of angular reprojection errors under the L_1 and L_∞ norms. These methods are generalizable to any types of central camera and significantly more efficient than the existing optimal methods. Second, in Chapter 4, we proposed a novel variant of the midpoint method that does not involve the optimization of geometric errors. Although this method is not theoretically optimal, it outperforms existing methods in terms of overall 2D and 3D accuracy at low parallax.

In Chapter 6, we proposed an efficient and robust multiview triangulation method. Our method incorporates an outlier rejection scheme using two-view RANSAC with the midpoint method. By prescreening the two-view samples prior to triangulation, we improve the efficiency when the outlier ratio is high. We also compared three different local optimization methods (DLT, LinLS and Gauss-Newton) and found that the Gauss-Newton method is the most accurate in terms of combined 2D and 3D accuracy. Furthermore, we proposed a novel method for modeling the uncertainty of the triangulated point and showed that this uncertainty can be used to control the 3D accuracy of the scene reconstruction.

For single rotation averaging, we proposed a fast and robust method using the Weiszfeld algorithm (Chapter 7). Specifically, we first obtain a robust initial solution by computing the elementwise median of the input rotation matrices, and then perform the Weiszfeld algorithm in combination with an outlier rejection scheme. To improve the efficiency, we compute the approximation of the chordal L_1-mean on $SO(3)$ instead of computing the geodesic L_1-mean. Our evaluation showed that our method outperforms (Hartley et al., 2011) in terms of speed and robustness.

For multiple rotation averaging, we proposed HARA, a novel hierarchical approach based on a robust incremental initialization of the rotation graph (Chapter 8). The key idea is to build a spanning tree by first propagating the edges with many strong triplet supports and later those with gradually weaker and fewer supports. This reduces the risk of adding outliers in the initial solution, which then enables us to filter outliers prior to nonlinear optimization. Additionally, we showed that using the smoothed L_{0^+} function (Peng et al., 2022) in the IRLS step of HARA further improves the results on the real datasets.

In Chapter 10, we proposed ROBA, a novel rotation-only method that optimizes the global rotations of multiple cameras directly using the image measurements and independently of the translations and the scene structure. This is made possible by extending the two-view rotation-only method of (Kneip and Lynen, 2013) and minimizing the aggregated cost using the Adam optimizer (Kingma and Ba, 2015). Our evaluation on both synthetic and real datasets demonstrated that our method improves the rotation accuracy when used with multiple rotation averaging. In Chapter 5, we also provided several geometric interpretations of the normalized epipolar error, which is what the cost function of ROBA is based on.

In Chapter 11, we proposed RODIAN, a novel method for robustly averaging numbers contaminated by a large proportion of outliers. Inspired by MINPRAN (Stewart, 1995), we assume that the outliers are uniformly distributed within the range of the data and search for the region that is least likely to contain outliers only. The median of the numbers within this region is then taken as RODIAN. Compared to existing methods, RODIAN has the advantage that it is fast, deterministic and highly robust without relying on a known inlier error bound.

Finally, in Chapter 12, we proposed the DTE, a novel error metric for quantitative trajectory evaluation. Unlike the commonly used ATE whose sensitivity quickly deteriorates in the presence of just a few outliers, the DTE can robustly discern the varying accuracy as the inlier trajectory error or the number of outliers varies. This is mainly because we use more

robust, median-based methods when computing the translation, rotation and scale of the trajectory alignment. In addition, we proposed the DRE, a rotation-only metric that uses the similar idea and has similar advantages to the DTE. We also proposed a simple yet effective method for calibrating the camera-to-marker rotation, which is required for computing both DTE and DRE.

13.2 Future Work

In this section, we discuss some of the possible directions for future work.

- LCSD-SLAM proposed in Chapter 2 has two main limitations: First, the relative scale and initial pose estimation described in Section 2.6.1 does not resolve the scale discrepancy between the direct and the feature-based map until the keyframes are marginalized. The real-time tracking accuracy can be improved by continuously adjusting the scale of one map to match that of the other. Second, the robustness of the LCSD-SLAM depends entirely on the robustness of DSO-reduced described in Section 2.7.1. The robustness could be improved by using DSO-default as the direct module, but it may incur excessive computational cost. A more practical solution would be to consider using an additional sensor, such as an inertial measurement unit (IMU). Visual-inertial SLAM systems are generally more robust than vision-only systems (Von Stumberg et al., 2018, Mur-Artal and Tardós, 2017), and at the same time, the absolute scale of the map is also estimated. For this reason, it would be interesting to develop the visual-inertial LCSD-SLAM and compare it against the vision-only LCSD-SLAM, as well as other existing visual-inertial methods.

- The uncertainty estimation method proposed for multiview triangulation in Chapter 6 has not been compared against other methods. It would be interesting to compare this method with the uncertainty estimation using the inverse of the Gauss-Newton Hessian matrix (Hartley and Zisserman, 2003, p.144). Also, our method uses the mean reprojection error to compute the uncertainty, which is not the ideal strategy if cameras have different resolutions. A possible direction for future work would then be to estimate the triangulation uncertainty based angular errors, such we can take into account the different resolutions and focal lengths of the cameras.

- The L_{0^+} optimization proposed in Chapter 9 was effective for multiple rotation averaging on the tested datasets because the error distributions of the datasets happen to match the PDF associated with the cost function. Unfortunately, there is no guarantee that this would be the case for other datasets. A more elegant solution would be to adapt c and p in (9.2) according to the error distributions at each iteration of the IRLS.

- Our work on rotation-only bundle adjustment in Chapter 10 has three main limitations:

1. The effectiveness of ROBA has not been demonstrated in a full SfM pipeline. Would the final results be significantly different after bundle adjustment with and without ROBA in the pipeline? Answering this specific question is left for future work.
2. We have not compared our optimization method with other methods. For example, is it really better to use the Adam optimizer (Kingma and Ba, 2015) instead of the commonly used Levenberg-Marquardt algorithm (Moré, 1978)? What about other stochastic optimization methods, such as AMSGrad (Reddi et al., 2018) and AdamW (Loshchilov and Hutter, 2019)? Again, answering these questions is left for future work.
3. The current version of ROBA is not robust to outliers, and it is not straightforward to robustify it without incurring excessive computational cost. We speculate that it may be more effective to handle outliers prior to ROBA than within. This could be achieved by further boosting the robustness of the multiple rotation averaging and/or the relative pose estimation step in the pipeline shown in Fig. 1.2.

- We believe that RODIAN proposed in Chapter 11 can be useful in some geometric problems, such as translation averaging. Application of RODIAN to motion averaging in SfM is left for future work.

- Another interesting future work would be to evaluate existing translation estimation algorithms using the trajectory evaluation method proposed in Chapter 12. Ultimately, the best outcome would be the development of a novel translation estimation method that gives smaller DTEs than the existing methods.

Appendix

A Derivation of (2.7) and (2.8)

Let subscript w, i and j denote the world reference frame, the previous and the current keyframe, respectively. Also, let subscript D and F denote the direct and the feature-based module, respectively. Suppose that two modules have the same camera axes convention. Now, let the relative scale between the two modules be s, such that for an arbitrary point $\mathbf{p}$,

$$s\mathbf{p}_{i|\mathrm{D}} = \mathbf{p}_{i|\mathrm{F}}, \tag{A.1}$$

$$s\mathbf{p}_{j|\mathrm{D}} = \mathbf{p}_{j|\mathrm{F}} = \mathbf{R}_{ji|\mathrm{F}}\mathbf{p}_{i|\mathrm{F}} + \mathbf{t}_{ji|\mathrm{F}}. \tag{A.2}$$

Then, we have

$$s\mathbf{p}_{j|\mathrm{D}} = s(\mathbf{R}_{ji|\mathrm{D}}\mathbf{p}_{i|\mathrm{D}} + \mathbf{t}_{ji|\mathrm{D}}) \tag{A.3}$$

$$= \mathbf{R}_{ji|\mathrm{D}}(s\mathbf{p}_{i|\mathrm{D}}) + s\mathbf{t}_{ji|\mathrm{D}} \tag{A.4}$$

$$\overset{\text{(A.1)}}{=} \mathbf{R}_{ji|\mathrm{D}}\mathbf{p}_{i|\mathrm{F}} + s\mathbf{t}_{ji|\mathrm{D}} \tag{A.5}$$

Subtracting (A.5) from (A.2) leads to

$$(\mathbf{R}_{ji|\mathrm{F}} - \mathbf{R}_{ji|\mathrm{D}})\mathbf{p}_{i|\mathrm{F}} + \mathbf{t}_{ji|\mathrm{F}} - s\mathbf{t}_{ji|\mathrm{D}} = \mathbf{0}. \tag{A.6}$$

For (A.6) to hold for $\mathbf{p}_{i|\mathrm{F}} = \mathbf{0}$:

$$\mathbf{t}_{ji|\mathrm{F}} = s\mathbf{t}_{ji|\mathrm{D}}. \tag{A.7}$$

Likewise, for (A.6) to hold for $\mathbf{p}_{i|\mathrm{F}} \neq \mathbf{0}$,

$$\mathbf{R}_{ji|\mathrm{F}} = \mathbf{R}_{ji|\mathrm{D}}. \tag{A.8}$$

On the other hand, we know that

$$\mathbf{p}_{i|\mathrm{D}} = \mathbf{R}_{iw|\mathrm{D}}\mathbf{p}_{w|\mathrm{D}} + \mathbf{t}_{iw|\mathrm{D}}, \tag{A.9}$$

$$\mathbf{p}_{j|\mathrm{D}} = \mathbf{R}_{jw|\mathrm{D}}\mathbf{p}_{w|\mathrm{D}} + \mathbf{t}_{jw|\mathrm{D}}. \tag{A.10}$$

Hence,

$$\mathbf{p}_{w|\mathrm{D}} \overset{(A.9)}{=} (\mathbf{R}_{iw|\mathrm{D}})^{-1}(\mathbf{p}_{i|\mathrm{D}} - \mathbf{t}_{iw|\mathrm{D}}) = (\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}(\mathbf{p}_{i|\mathrm{D}} - \mathbf{t}_{iw|\mathrm{D}}) = (\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}\mathbf{p}_{i|\mathrm{D}} - (\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}\mathbf{t}_{iw|\mathrm{D}}, \tag{A.11}$$

$$\mathbf{p}_{w|\mathrm{D}} \overset{(A.10)}{=} (\mathbf{R}_{jw|\mathrm{D}})^{-1}(\mathbf{p}_{j|\mathrm{D}} - \mathbf{t}_{jw|\mathrm{D}}). \tag{A.12}$$

Equating the right-hand sides of (A.11) and (A.12), we get

$$\mathbf{p}_{j|\mathrm{D}} = \mathbf{R}_{jw|\mathrm{D}}(\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}\mathbf{p}_{i|\mathrm{D}} - \mathbf{R}_{jw|\mathrm{D}}(\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}\mathbf{t}_{iw|\mathrm{D}} + \mathbf{t}_{jw|\mathrm{D}}. \tag{A.13}$$

This shows that

$$\mathbf{R}_{ij|\mathrm{D}} = \mathbf{R}_{jw|\mathrm{D}}(\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}} \tag{A.14}$$

and

$$\mathbf{t}_{ji|\mathrm{D}} = -\mathbf{R}_{jw|\mathrm{D}}(\mathbf{R}_{iw|\mathrm{D}})^{\mathrm{T}}\mathbf{t}_{iw|\mathrm{D}} + \mathbf{t}_{jw|\mathrm{D}} \tag{A.15}$$

$$\overset{(A.14)}{=} -\mathbf{R}_{ji|\mathrm{D}}\mathbf{t}_{iw|\mathrm{D}} + \mathbf{t}_{jw|\mathrm{D}}. \tag{A.16}$$

Finally, substituting (A.14) and (A.16) into (A.7) and (A.8) leads to (2.7) and (2.8).

B The Median Errors of the Tested VO/SLAM Methods

	ORB VO	ORB SLAM	DSO default	DSO reduced	Ours VO	Ours SLAM
M1L	0.046	0.046	0.053	**0.076**	**0.044**	0.046
M2L	0.038	**0.037**	0.050	**0.071**	0.038	**0.037**
M3L	**0.039**	**0.039**	0.166	**0.249**	0.040	0.040
M4L	0.064	0.065	**0.172**	0.163	**0.060**	0.064
M5L	0.078	**0.051**	0.101	**0.144**	0.061	0.065
V11L	0.096	0.097	0.104	**0.115**	**0.089**	0.090
V12L	0.127	**0.103**	**0.516**	0.136	0.104	0.105
V13L	0.938	0.423	×	0.757	0.608	**0.412**
V21L	**0.080**	**0.080**	0.089	**0.090**	0.089	0.088
V22L	0.185	**0.178**	0.210	**0.233**	0.181	**0.178**
V23L	×	×	1.429	**1.055**	1.174	1.062

	ORB VO	ORB SLAM	DSO default	DSO reduced	Ours VO	Ours SLAM
M1R	0.040	**0.038**	0.052	**0.076**	**0.038**	0.039
M2R	0.038	0.038	0.052	**0.071**	**0.036**	**0.036**
M3R	0.045	**0.043**	0.149	**0.185**	**0.043**	0.045
M4R	0.074	**0.069**	0.129	**0.174**	0.070	0.074
M5R	0.056	**0.055**	0.095	**0.122**	0.065	0.060
V11R	0.103	0.104	0.119	**0.215**	0.100	**0.099**
V12R	0.125	0.112	0.133	**0.161**	**0.111**	**0.111**
V13R	×	0.130	×	**0.614**	1.277	0.825
V21R	**0.103**	0.104	0.104	**0.116**	0.113	0.114
V22R	0.198	**0.183**	**0.255**	0.209	0.201	0.191
V23R	×	×	0.707	0.432	0.345	**0.238**

Table B.1: **[EuRoC MAV]** Median absolute trajectory errors e_{ate} [m] over 10 runs on each sequence. The best and the worst results are in blue and red, respectively. The mark × indicates that more than half of the total runs lost tracking due to challenging scene textures, camera motions or illumination changes.

	ORB VO	ORB SLAM	DSO default	DSO reduced	Ours VO	Ours SLAM
1	0.744	**1.063**	**0.526**	0.799	0.885	0.860
2	**19.08**	**56.03**	0.570	**0.487**	1.412	0.908
3	0.731	**0.510**	3.393	**4.747**	3.283	0.529
4	**2.458**	2.234	0.689	0.770	**0.677**	0.764
5	×	×	**1.718**	2.004	1.921	1.880
6	1.328	0.037	0.808	1.328	**1.358**	**0.522**
7	**0.406**	0.454	0.604	**0.969**	0.926	0.931
8	**596.9**	**425.0**	**0.353**	0.483	0.600	0.625
9	0.534	**0.393**	0.617	**1.069**	1.057	0.994
10	**1.632**	1.312	**0.289**	0.427	0.400	0.564
11	**1.086**	**0.152**	0.592	0.847	0.961	0.172
12	**2.533**	1.646	**0.597**	0.937	1.000	0.812
13	2.232	**2.366**	1.388	1.312	**1.130**	1.216
14	**10.05**	**20.63**	0.718	**0.666**	0.863	0.847
15	1.241	**1.498**	**0.832**	1.310	1.340	1.323
16	0.832	**0.086**	0.514	0.739	**0.931**	0.603
17	1.777	**0.417**	2.300	**2.684**	2.681	1.300
18	5.381	×	1.579	2.135	1.986	**0.411**
19	**6.493**	5.621	1.898	3.534	3.673	**1.232**
20	1.266	**0.273**	0.754	1.095	**1.331**	0.295
21	×	×	4.219	2.707	2.504	**0.230**
22	×	×	3.954	4.785	4.732	**1.131**
23	**5.140**	0.161	0.478	2.344	2.538	**0.061**
24	**5.731**	0.263	0.297	0.361	0.275	**0.156**
25	1.657	0.099	0.821	**2.275**	2.070	**0.081**

	ORB VO	ORB SLAM	DSO default	DSO reduced	Ours VO	Ours SLAM
26	**5.386**	0.323	3.336	4.756	3.705	**0.241**
27	**3.810**	**0.198**	0.965	1.496	1.530	0.321
28	**9.517**	0.433	2.185	2.309	2.186	**0.121**
29	**8.255**	0.325	0.430	1.232	1.311	**0.039**
30	1.278	0.046	0.663	**1.919**	0.821	**0.037**
31	×	×	0.593	0.770	0.738	**0.078**
32	**3.094**	0.077	0.320	0.816	0.876	**0.070**
33	**2.726**	**0.074**	1.449	1.730	1.632	0.078
34	2.687	**0.105**	0.884	**9.434**	**22.54**	0.355
35	**7.046**	0.705	0.578	2.620	2.758	**0.290**
36	1.465	0.590	**5.851**	2.898	3.661	**0.528**
37	0.442	**0.103**	0.379	0.685	**0.709**	0.180
38	×	×	0.768	1.662	1.966	**0.171**
39	**32.58**	0.314	1.293	2.221	2.896	**0.210**
40	×	×	1.805	1.185	1.007	**0.164**
41	×	×	0.897	0.542	0.449	**0.195**
42	×	0.547	0.889	1.293	1.158	**0.084**
43	1.388	0.181	0.458	**1.963**	1.902	**0.102**
44	1.388	**0.065**	0.552	**1.780**	1.713	0.120
45	**4.085**	**0.204**	1.258	2.442	2.693	0.227
46	**12.09**	1.345	0.608	1.415	1.605	**0.403**
47	**12.68**	0.252	1.519	2.019	2.234	**0.122**
48	**3.753**	0.208	1.089	3.314	2.621	**0.060**
49	**11.40**	**0.164**	×	1.023	1.172	0.195
50	**137.2**	4.557	**0.771**	1.830	1.999	1.586

Table B.2: **[TUM monoVO]** Median alignment errors e_{align} [m] over 10 runs on each sequence. The best and the worst results are in blue and red, respectively. The mark × indicates that more than half of the total runs lost tracking due to challenging scene textures, camera motions or illumination changes.

C Derivation of the Depths given by the Classic Midpoint Method

In this section, we derive the equation for computing the depths in the classic midpoint method, *i.e.*, $\lambda_{\text{mid}0}$ and $\lambda_{\text{mid}1}$ in Fig. 4.2. In literature, the formula has been used many times without derivation (Kanazawa and Kanatani, 1995, Lindstrom, 2010, Lee and Civera, 2019*a*). For the sake of completeness, we present the full derivations here. In doing so, we will use the following properties of the dot product and the cross product operations:

$$\widehat{\mathbf{a}} \times (\widehat{\mathbf{a}} \times \mathbf{b}) = \widehat{\mathbf{a}}(\widehat{\mathbf{a}} \cdot \mathbf{b}) - \mathbf{b}. \tag{C.17}$$

$$(\widehat{\mathbf{a}} \times \mathbf{b}) \cdot (\widehat{\mathbf{a}} \times \mathbf{c}) = \mathbf{b} \cdot \mathbf{c} - (\widehat{\mathbf{a}} \cdot \mathbf{b})(\widehat{\mathbf{a}} \cdot \mathbf{c}). \tag{C.18}$$

$$(\mathbf{a} \times \mathbf{b}) \times (\mathbf{a} \times \mathbf{c}) = (\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}))\, \mathbf{a}. \tag{C.19}$$

Next, we introduce the following lemma:

Lemma 7 (The Closest Pair of Points on Two Skew Lines)
Consider two skew lines $\mathbf{L}_0(s_0) = \mathbf{c}_0 + s_0\mathbf{m}_0$ *and* $\mathbf{L}_1(s_1) = \mathbf{c}_1 + s_1\mathbf{m}_1$ *in 3D space. Let* $\mathbf{t} = \mathbf{c}_0 - \mathbf{c}_1$ *and* $(\mathbf{r}_0, \mathbf{r}_1)$ *be the two points on each line that form the closest pair. Then,*

$$\mathbf{r}_0 = \mathbf{c}_0 + \frac{(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_1 \times \mathbf{t})}{\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2} \widehat{\mathbf{m}}_0 \tag{C.20}$$

and

$$\mathbf{r}_1 = \mathbf{c}_1 + \frac{(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_0 \times \mathbf{t})}{\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2} \widehat{\mathbf{m}}_1. \tag{C.21}$$

Proof. In geometry, it is a well-known fact that the closest pair of points on two skew lines lie on the common perpendicular to both lines. In other words, the vector $\mathbf{r}_0 - \mathbf{r}_1$ is perpendicular to both $\mathbf{L}_0$ and $\mathbf{L}_1$. Therefore, for some scalar τ,

$$\mathbf{r}_0 - \mathbf{r}_1 = \tau\, (\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1)\,. \tag{C.22}$$

Since point $\mathbf{r}_0$ and $\mathbf{r}_1$ are respectively located along $\mathbf{L}_0$ and $\mathbf{L}_1$, we can write

$$\mathbf{r}_0 = \mathbf{c}_0 + \lambda_0 \widehat{\mathbf{m}}_0 \quad \text{and} \quad \mathbf{r}_1 = \mathbf{c}_1 + \lambda_1 \widehat{\mathbf{m}}_1. \tag{C.23}$$

for some scalar λ_0 and λ_1. Then, (C.22) becomes

$$\mathbf{t} + \lambda_0 \widehat{\mathbf{m}}_0 - \lambda_1 \widehat{\mathbf{m}}_1 = \tau \mathbf{n}, \tag{C.24}$$

where $\mathbf{n} = \widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1$. This makes a system of three equations (in each coordinate x, y and z) with three unknowns λ_0, λ_1, and τ. Removing τ from the equations leads to

$$\frac{t_x + \lambda_0 m_{0x} - \lambda_1 m_{1x}}{n_x} = \frac{t_y + \lambda_0 m_{0y} - \lambda_1 m_{1y}}{n_y} \tag{C.25}$$

and

$$\frac{t_y + \lambda_0 m_{0y} - \lambda_1 m_{1y}}{n_y} = \frac{t_z + \lambda_0 m_{0z} - \lambda_1 m_{1z}}{n_z}. \tag{C.26}$$

Note that $\mathbf{t} = [t_x, t_y, t_z]^\intercal$, $\mathbf{n} = [n_x, n_y, n_z]^\intercal$, $\widehat{\mathbf{m}}_0 = [m_{0x}, m_{0y}, m_{0z}]^\intercal$ and $\widehat{\mathbf{m}}_1 = [m_{1x}, m_{1y}, m_{1z}]^\intercal$. From (C.25) and (C.26), we get

$$\lambda_0 = \frac{\lambda_1 \left(m_{1x} n_y - m_{1y} n_x\right) + t_y n_x - t_x n_y}{m_{0x} n_y - m_{0y} n_x} \tag{C.27}$$

and

$$\lambda_0 = \frac{\lambda_1 \left(m_{1y} n_z - m_{1z} n_y\right) + t_z n_y - t_y n_z}{m_{0y} n_z - m_{0z} n_y}. \tag{C.28}$$

Equating the right-hand sides of (C.27) and (C.28) leads to

$$\lambda_1 = \frac{A - B}{C - D}, \tag{C.29}$$

where

$$\begin{aligned}
A &= (t_z n_y - t_y n_z)(m_{0x} n_y - m_{0y} n_x) \\
B &= (t_y n_x - t_x n_y)(m_{0y} n_z - m_{0z} n_y) \\
C &= (m_{1x} n_y - m_{1y} n_x)(m_{0y} n_z - m_{0z} n_y) \\
&= \left(\widehat{\mathbf{m}}_1 \times \mathbf{n}\right)_z \left(\widehat{\mathbf{m}}_0 \times \mathbf{n}\right)_x, \\
D &= (m_{1y} n_z - m_{1z} n_y)(m_{0x} n_y - m_{0y} n_x) \\
&= \left(\widehat{\mathbf{m}}_1 \times \mathbf{n}\right)_x \left(\widehat{\mathbf{m}}_0 \times \mathbf{n}\right)_z.
\end{aligned}$$

We can rearrange $A - B$ into

$$A - B = n_y \mathbf{t} \cdot \begin{pmatrix} m_{0y} n_z - m_{0z} n_y \\ m_{0z} n_x - m_{0x} n_z \\ m_{0x} n_y - m_{0y} n_x \end{pmatrix}. \tag{C.30}$$

The latter term in the dot product of (C.30) is equal to $\widehat{\mathbf{m}}_0 \times \mathbf{n}$. Thus,

$$\begin{aligned} A - B &= n_y \mathbf{t} \cdot (\widehat{\mathbf{m}}_0 \times (\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1)) \\ &\overset{(C.17)}{=} n_y \mathbf{t} \cdot (\widehat{\mathbf{m}}_0 (\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{m}}_1) - \widehat{\mathbf{m}}_1) \\ &= n_y ((\widehat{\mathbf{m}}_0 \cdot \widehat{\mathbf{m}}_1) (\widehat{\mathbf{m}}_0 \cdot \mathbf{t}) - \widehat{\mathbf{m}}_1 \cdot \mathbf{t}) \\ &\overset{(C.18)}{=} -n_y (\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_0 \times \mathbf{t}) . \end{aligned} \tag{C.31}$$

We can rearrange $C - D$ into

$$\begin{aligned} C - D &= (\widehat{\mathbf{m}}_1 \times \mathbf{n})_z (\widehat{\mathbf{m}}_0 \times \mathbf{n})_x - (\widehat{\mathbf{m}}_1 \times \mathbf{n})_x (\widehat{\mathbf{m}}_0 \times \mathbf{n})_z \\ &= ((\widehat{\mathbf{m}}_1 \times \mathbf{n}) \times (\widehat{\mathbf{m}}_0 \times \mathbf{n}))_y \\ &\overset{(C.19)}{=} ((\mathbf{n} \cdot (\widehat{\mathbf{m}}_1 \times \widehat{\mathbf{m}}_0)) \mathbf{n})_y \\ &= \left(-\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2 \mathbf{n}\right)_y \\ &= -\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2 n_y. \end{aligned} \tag{C.32}$$

Substituting (C.31) and (C.32) into (C.29) gives

$$\lambda_1 = \frac{(\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1) \cdot (\widehat{\mathbf{m}}_0 \times \mathbf{t})}{\|\widehat{\mathbf{m}}_0 \times \widehat{\mathbf{m}}_1\|^2}. \tag{C.33}$$

Finally, substituting (C.33) into (C.23) leads to (C.21). Equation (C.20) can be derived analogously. ■

By substituting $\mathbf{R}\mathbf{f}_0$ and $\mathbf{f}_1$ into $\mathbf{m}_0$ and $\mathbf{m}_1$, we can use lemma 7 to obtain $\lambda_{\text{mid}0}$ and $\lambda_{\text{mid}1}$ in Fig 4.2:

$$\lambda_{\text{mid}0} = \frac{(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1) \cdot (\widehat{\mathbf{f}}_1 \times \mathbf{t})}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\|^2}, \quad \lambda_{\text{mid}1} = \frac{(\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1) \cdot (\mathbf{R}\widehat{\mathbf{f}}_0 \times \mathbf{t})}{\|\mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1\|^2}. \tag{C.34}$$

Finally, letting $\mathbf{p} = \mathbf{R}\widehat{\mathbf{f}}_0 \times \widehat{\mathbf{f}}_1$, $\mathbf{q} = \mathbf{R}\widehat{\mathbf{f}}_0 \times \mathbf{t}$ and $\mathbf{r} = \widehat{\mathbf{f}}_1 \times \mathbf{t}$, we get (4.8).

D Nomenclature for Chapter 6

Symbol	Description
c_i	Camera/View with index i.
$\mathcal{V}$	Set of all cameras/views observing the point to be triangulated.
$\mathbf{x}^w$	Ground-truth 3D point to be triangulated, expressed in the world reference frame.
$\mathbf{x}^w_{\text{est}}$	Estimated 3D point, expressed in the world reference frame.
$\mathbf{x}_i$	Estimated 3D point, expressed in the camera reference frame of c_i.
$\mathbf{R}_i$ and $\mathbf{t}_i$	Rotation matrix and translation vector that together transform the vector in the world frame to the reference frame of c_i, *e.g.*, $\mathbf{x}_i = \mathbf{R}_i\mathbf{x}^w_{\text{est}} + \mathbf{t}_i$.
$\mathbf{P}_i$	Extrinsic matrix of c_i, *i.e.*, $\mathbf{P}_i = [\mathbf{R}_i \mid \mathbf{t}_i]$.
$\mathbf{c}^w_i$	The position of c_i in the world reference frame, *i.e.*, $\mathbf{c}_i = -\mathbf{R}_i^\top \mathbf{t}_i$.
$\mathbf{K}_i$	Camera calibration matrix of c_i.
$\mathbf{u}_i$	Noisy pixel coordinates of the point observed in c_i, *i.e.*, $\mathbf{u}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix}$.
$\mathbf{u}'_i$	Pixel coordinates of the projection of $\mathbf{x}^w_{\text{est}}$ in c_i, *i.e.*, $\mathbf{u}'_i = \begin{bmatrix} u'_i \\ v'_i \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \frac{\mathbf{K}_i\mathbf{x}_i}{(\mathbf{x}_i)_3}$.
$\mathbf{f}_i$	Normalized image coordinates of the point observed in c_i, *i.e.*, $\mathbf{f}_i = \mathbf{K}_i^{-1} \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix}$.
$\mathbf{f}^w_i$	Feature ray of the point observed in c_i, expressed in the world reference frame *i.e.*, $\mathbf{f}^w_i = \mathbf{R}_i^\top \mathbf{f}_i$.
$\mathcal{I}$	Set of all inlying cameras/views for a given $\mathbf{x}^w_{\text{est}}$ that meet the following conditions: (1) It has a small reprojection error, and (2) it satisfies the cheirality.
$\beta_{\max}$	Maximum parallax angle of $\mathcal{I}$, *i.e.*, $\beta_{\max} = \max\left\{\angle\left(\mathbf{f}^w_j, \mathbf{f}^w_k\right) \mid j,k \in \mathcal{I}\right\}$.
$\mathbf{t}^w_{jk}$	Vector difference of the camera position j and k, *i.e.*, $\mathbf{t}^w_{jk} = \mathbf{c}^w_j - \mathbf{c}^w_k$.
e_{jk}	Normalized epipolar error for the point observed in c_j and c_k, *i.e.*, $e_{jk} := \left\lvert \widehat{\mathbf{t}}^w_{jk} \cdot \left(\widehat{\mathbf{f}}^w_j \times \widehat{\mathbf{f}}^w_k\right)\right\rvert$.
p_{jk}, q_{jk} and r_{jk}	Quantities for the point observed in c_j and c_k, defined as follows: $p_{jk} = \widehat{\mathbf{f}}^w_j \cdot \widehat{\mathbf{f}}^w_k$, $q_{jk} = \widehat{\mathbf{f}}^w_j \cdot \widehat{\mathbf{t}}^w_{jk}$, $r_{jk} = \widehat{\mathbf{f}}^w_k \cdot \widehat{\mathbf{t}}^w_{jk}$.
λ_j and λ_k	Depths of the midpoint anchors for the point observed in c_j and c_k.
b_{good}	Boolean that indicates the validity of the resulting midpoint.
$\mathbf{e}_{2D}$	2D reprojection error of $\mathbf{x}^w_{\text{est}}$ in $\mathcal{V}$, *i.e.*, $\mathbf{e}_{2D} = \left[\cdots, \lVert\mathbf{u}_i - \mathbf{u}'_i\rVert, \cdots\right]^\top$ for all $i \in \mathcal{V}$.
$\overline{e}_{2D}$	Mean 2D reprojection error of $\mathbf{x}^w_{\text{est}}$ in $\mathcal{I}$.
σ_{3D}	Estimated magnitude of the 3D uncertainty of $\mathbf{x}^w_{\text{est}}$.
$\mathbf{J}$	Jacobian matrix in the Gauss-Newton algorithm.
$\mathbf{r}$	Residual vector in the Gauss-Newton algorithm.
G	3D regular grid that maps $(\lvert\mathcal{I}\rvert, \overline{e}_{2D}, \beta_{\max})$ to σ_{3D}.
η	Desired probability that at least one outlier-free pair is sampled in the two-view RANSAC.
ϵ	Estimated inlier ratio.
$m_{\min}$	Minimum number of pairs to be sampled in RANSAC to achieve the desired η.
C	Cost function of the hypothesis in RANSAC.
δ_{2D}	Inlier threshold for the reprojection error.
δ_{epipolar}	Threshold for the normalized epipolar error.
δ_{update}	Threshold for detecting the convergence of the mean reprojection error in the Gauss-Newton algorithm.
δ_{lower}	Lower threshold for the cosine of the considered angle.
δ_{upper}	Upper threshold for the cosine of the considered angle.
δ_{pair}	Maximum number of pairs to be sampled for computing the maximum parallax angle.

E Derivation of (6.10)

For camera c_i, the 2D reprojection error of point $\mathbf{x}_{\text{est}}^w$ is obtained by

$$\begin{bmatrix} (u_{\text{error}})_i \\ (v_{\text{error}})_i \end{bmatrix} = - \begin{bmatrix} u_i \\ v_i \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{K}_i \begin{bmatrix} [(\mathbf{P}_i)_{\text{row1}} \widetilde{\mathbf{x}}_{\text{est}}^w] \,/\, [(\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}_{\text{est}}^w] \\ [(\mathbf{P}_i)_{\text{row2}} \widetilde{\mathbf{x}}_{\text{est}}^w] \,/\, [(\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}_{\text{est}}^w] \\ 1 \end{bmatrix}, \tag{E.35}$$

where $\begin{bmatrix} u_i \\ v_i \end{bmatrix}$ is the image coordinates of the observed point, $\mathbf{K}_i = \begin{bmatrix} k_{i11} & k_{i12} & k_{i13} \\ k_{i21} & k_{i22} & k_{i23} \\ k_{i31} & k_{i32} & k_{i33} \end{bmatrix}$ is the intrinsic matrix[1], $\mathbf{P}_i = [\mathbf{R}_i \mid \mathbf{t}_i]$ is the extrinsic matrix, and $\widetilde{\mathbf{x}}_{\text{est}}^w = \begin{bmatrix} \mathbf{x}_{\text{est}}^w \\ 1 \end{bmatrix}$ is the homogeneous coordinates of $\mathbf{x}_{\text{est}}^w$. Rewriting (E.35), we get

$$\begin{bmatrix} (u_{\text{error}})_i \\ (v_{\text{error}})_i \end{bmatrix} = \begin{bmatrix} k_{i13} - u_i \\ k_{i23} - v_i \end{bmatrix} + \frac{1}{(\widetilde{\mathbf{x}}_{\text{est}}^w)^\top (\mathbf{P}_i)_{\text{row3}}^\top} \begin{bmatrix} (\widetilde{\mathbf{x}}_{\text{est}}^w)^\top \left(k_{i11} (\mathbf{P}_i)_{\text{row1}}^\top + k_{i12} (\mathbf{P}_i)_{\text{row2}}^\top \right) \\ (\widetilde{\mathbf{x}}_{\text{est}}^w)^\top \left(k_{i21} (\mathbf{P}_i)_{\text{row1}}^\top + k_{i22} (\mathbf{P}_i)_{\text{row2}}^\top \right) \end{bmatrix}. \tag{E.36}$$

Now, for n views observing the same point, define the following matrices:

$$\mathbf{M}_1 := [k_{113} - u_1, \cdots, k_{n13} - u_n], \tag{E.37}$$

$$\mathbf{M}_2 := [k_{123} - v_1, \cdots, k_{n23} - v_n], \tag{E.38}$$

$$\mathbf{M}_3 := \left[\left(k_{111} (\mathbf{P}_1)_{\text{row1}}^\top + k_{112} (\mathbf{P}_1)_{\text{row2}}^\top \right), \cdots, \left(k_{n11} (\mathbf{P}_n)_{\text{row1}}^\top + k_{n12} (\mathbf{P}_n)_{\text{row2}}^\top \right) \right], \tag{E.39}$$

$$\mathbf{M}_4 := \left[\left(k_{121} (\mathbf{P}_1)_{\text{row1}}^\top + k_{122} (\mathbf{P}_1)_{\text{row2}}^\top \right), \cdots, \left(k_{n21} (\mathbf{P}_n)_{\text{row1}}^\top + k_{n22} (\mathbf{P}_n)_{\text{row2}}^\top \right) \right], \tag{E.40}$$

$$\mathbf{M}_5 := \left[(\mathbf{P}_1)_{\text{row3}}^\top, \cdots, (\mathbf{P}_n)_{\text{row3}}^\top \right] \in \mathbb{R}^{3 \times n}, \tag{E.41}$$

$$\mathbf{M}_6 := (\widetilde{\mathbf{x}}_{\text{est}}^w)^\top \mathbf{M}_5, \tag{E.42}$$

$$\mathbf{M}_7 := \mathbf{M}_1 + \left((\widetilde{\mathbf{x}}_{\text{est}}^w)^\top \mathbf{M}_3 \right) \oslash \mathbf{M}_6, \tag{E.43}$$

$$\mathbf{M}_8 := \mathbf{M}_2 + \left((\widetilde{\mathbf{x}}_{\text{est}}^w)^\top \mathbf{M}_4 \right) \oslash \mathbf{M}_6. \tag{E.44}$$

Then, by stacking (E.36) for all views, we obtain the following equation:

$$\begin{bmatrix} (u_{\text{error}})_1 & (u_{\text{error}})_2 & \cdots & (u_{\text{error}})_n \\ (v_{\text{error}})_1 & (v_{\text{error}})_2 & \cdots & (v_{\text{error}})_n \end{bmatrix} = \begin{bmatrix} \mathbf{M}_7 \\ \mathbf{M}_8 \end{bmatrix}, \tag{E.45}$$

which leads to

$$\left[(u_{\text{error}})_1^2 + (v_{\text{error}})_1^2, \cdots, (u_{\text{error}})_n^2 + (v_{\text{error}})_n^2 \right] = \mathbf{M}_7 \circ \mathbf{M}_7 + \mathbf{M}_8 \circ \mathbf{M}_8. \tag{E.46}$$

Therefore, the reprojection errors can be obtained as vector $\mathbf{e}_{2D}$ in the following form:

$$\mathbf{e}_{2D}^\top = \left(\mathbf{M}_7 \circ \mathbf{M}_7 + \mathbf{M}_8 \circ \mathbf{M}_8 \right)^{\circ 1/2}. \tag{E.47}$$

[1]Here, we treat the matrix $\mathbf{K}_i$ as if it is a full matrix, but it is in fact an upper triangular matrix: k_{i11} and k_{i22} are the focal lengths in the horizontal and vertical direction, respectively. k_{i13} and k_{i23} are the horizontal and vertical coordinate of the principal point, respectively. k_{i12} is the skew coefficient. k_{i21}, k_{i31} and k_{i32} are equal to zero, and k_{i33} is equal to one.

F Derivation of (6.33)

Expanding (E.35), we get

$$
\begin{aligned}
&(u_{\text{error}})_i \\
&= -u_i + \frac{k_{i11}(r_{i11}x^w_{\text{est}} + r_{i12}y^w_{\text{est}} + r_{i13}z^w_{\text{est}} + t_{i1}) + k_{i12}(r_{i21}x^w_{\text{est}} + r_{i22}y^w_{\text{est}} + r_{i23}z^w_{\text{est}} + t_{i2})}{r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}} + k_{i13}, \\
&= -u_i + k_{i13} + \frac{(k_{i11}r_{i11} + k_{i12}r_{i21})x^w_{\text{est}} + (k_{i11}r_{i12} + k_{i12}r_{i22})y^w_{\text{est}} + (k_{i11}r_{i13} + k_{i12}r_{i23})z^w_{\text{est}} + k_{i11}t_{i1} + k_{i12}t_{i2}}{r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}}, \\
&(v_{\text{error}})_i \\
&= -v_i + \frac{k_{i21}(r_{i11}x^w_{\text{est}} + r_{i12}y^w_{\text{est}} + r_{i13}z^w_{\text{est}} + t_{i1}) + k_{i22}(r_{i21}x^w_{\text{est}} + r_{i22}y^w_{\text{est}} + r_{i23}z^w_{\text{est}} + t_{i2})}{r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}} + k_{i23} \\
&= -v_i + k_{i23} + \frac{(k_{i21}r_{i11} + k_{i22}r_{i21})x^w_{\text{est}} + (k_{i21}r_{i12} + k_{i22}r_{i22})y^w_{\text{est}} + (k_{i21}r_{i13} + k_{i22}r_{i23})z^w_{\text{est}} + k_{i21}t_{i1} + k_{i22}t_{i2}}{r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}},
\end{aligned}
$$

where r_{ijk} and k_{ijk} respectively indicate the elements of $\mathbf{R}_i$ and $\mathbf{K}_i$ at the j-th row and k-th column, and t_{ij} indicate the j-th element of $\mathbf{t}_i$. To compute the Jacobian, we take the partial derivatives with respect to x^w_{est}, y^w_{est} and z^w_{est}.

Using the fact that $\dfrac{d}{dx}\left(\dfrac{ax+b}{cx+d}\right) = \dfrac{ad-bc}{(cx+d)^2}$, we obtain the following:

$$
\begin{aligned}
&\frac{\partial(u_{\text{error}})_i}{\partial x^w_{\text{est}}} \\
&= \frac{(k_{i11}r_{i11} + k_{i12}r_{i21})(r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}) - r_{i31}\left[(k_{i11}r_{i12} + k_{i12}r_{i22})y^w_{\text{est}} + (k_{i11}r_{i13} + k_{i12}r_{i23})z^w_{\text{est}} + k_{i11}t_{i1} + k_{i12}t_{i2}\right]}{(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3})^2} \\
&\frac{\partial(v_{\text{error}})_i}{\partial x^w_{\text{est}}} \\
&= \frac{(k_{i21}r_{i11} + k_{i22}r_{i21})(r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}) - r_{i31}\left[(k_{i21}r_{i12} + k_{i22}r_{i22})y^w_{\text{est}} + (k_{i21}r_{i13} + k_{i22}r_{i23})z^w_{\text{est}} + k_{i21}t_{i1} + k_{i22}t_{i2}\right]}{(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3})^2} \\
&\frac{\partial(u_{\text{error}})_i}{\partial y^w_{\text{est}}} \\
&= \frac{(k_{i11}r_{i12} + k_{i12}r_{i22})(r_{i31}x^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}) - r_{i32}\left[(k_{i11}r_{i11} + k_{i12}r_{i21})x^w_{\text{est}} + (k_{i11}r_{i13} + k_{i12}r_{i23})z^w_{\text{est}} + k_{i11}t_{i1} + k_{i12}t_{i2}\right]}{(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3})^2} \\
&\frac{\partial(v_{\text{error}})_i}{\partial y^w_{\text{est}}} \\
&= \frac{(k_{i21}r_{i12} + k_{i22}r_{i22})(r_{i31}x^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3}) - r_{i32}\left[(k_{i21}r_{i11} + k_{i22}r_{i21})x^w_{\text{est}} + (k_{i21}r_{i13} + k_{i22}r_{i23})z^w_{\text{est}} + k_{i21}t_{i1} + k_{i22}t_{i2}\right]}{(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3})^2} \\
&\frac{\partial(u_{\text{error}})_i}{\partial z^w_{\text{est}}} \\
&= \frac{(k_{i11}r_{i13} + k_{i12}r_{i23})(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + t_{i3}) - r_{i33}\left[(k_{i11}r_{i11} + k_{i12}r_{i21})x^w_{\text{est}} + (k_{i11}r_{i12} + k_{i12}r_{i22})y^w_{\text{est}} + k_{i11}t_{i1} + k_{i12}t_{i2}\right]}{(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3})^2} \\
&\frac{\partial(v_{\text{error}})_i}{\partial z^w_{\text{est}}} \\
&= \frac{(k_{i21}r_{i13} + k_{i22}r_{i23})(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + t_{i3}) - r_{i33}\left[(k_{i21}r_{i11} + k_{i22}r_{i21})x^w_{\text{est}} + (k_{i21}r_{i12} + k_{i22}r_{i22})y^w_{\text{est}} + k_{i21}t_{i1} + k_{i22}t_{i2}\right]}{(r_{i31}x^w_{\text{est}} + r_{i32}y^w_{\text{est}} + r_{i33}z^w_{\text{est}} + t_{i3})^2}
\end{aligned}
$$

Letting $\widetilde{\mathbf{x}}^w_{\text{est}} = \begin{bmatrix} \mathbf{x}^w_{\text{est}} \\ 1 \end{bmatrix}$ and $\mathbf{P}_i = [\mathbf{R}_i \mid \mathbf{t}_i]$, we can rearrange these equations as follows:

$$\frac{\partial (u_{\text{error}})_i}{\partial x^w_{\text{est}}} = \frac{\mathbf{a}_{1i}^\top \widetilde{\mathbf{x}}^w_{\text{est}}}{\left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}^w_{\text{est}}\right)^2}, \quad \frac{\partial (v_{\text{error}})_i}{\partial x^w_{\text{est}}} = \frac{\mathbf{a}_{2i}^\top \widetilde{\mathbf{x}}^w_{\text{est}}}{\left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}^w_{\text{est}}\right)^2}, \tag{F.48}$$

$$\frac{\partial (u_{\text{error}})_i}{\partial y^w_{\text{est}}} = \frac{\mathbf{a}_{3i}^\top \widetilde{\mathbf{x}}^w_{\text{est}}}{\left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}^w_{\text{est}}\right)^2}, \quad \frac{\partial (v_{\text{error}})_i}{\partial y^w_{\text{est}}} = \frac{\mathbf{a}_{4i}^\top \widetilde{\mathbf{x}}^w_{\text{est}}}{\left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}^w_{\text{est}}\right)^2}, \tag{F.49}$$

$$\frac{\partial (u_{\text{error}})_i}{\partial z^w_{\text{est}}} = \frac{\mathbf{a}_{5i}^\top \widetilde{\mathbf{x}}^w_{\text{est}}}{\left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}^w_{\text{est}}\right)^2}, \quad \frac{\partial (v_{\text{error}})_i}{\partial z^w_{\text{est}}} = \frac{\mathbf{a}_{6i}^\top \widetilde{\mathbf{x}}^w_{\text{est}}}{\left((\mathbf{P}_i)_{\text{row3}} \widetilde{\mathbf{x}}^w_{\text{est}}\right)^2}, \tag{F.50}$$

with

$$\mathbf{a}_{1i} = \begin{bmatrix} 0 \\ k_{i11}(r_{i11}r_{i32} - r_{i31}r_{i12}) + k_{i12}(r_{i21}r_{i32} - r_{i31}r_{i22}) \\ k_{i11}(r_{i11}r_{i33} - r_{i31}r_{i13}) + k_{i12}(r_{i21}r_{i33} - r_{i31}r_{i23}) \\ k_{i11}(r_{i11}t_{i3} - r_{i31}t_{i1}) + k_{i12}(r_{i21}t_{i3} - r_{i31}t_{i2}) \end{bmatrix}, \quad \mathbf{a}_{2i} = \begin{bmatrix} 0 \\ k_{i21}(r_{i11}r_{i32} - r_{i31}r_{i12}) + k_{i22}(r_{i21}r_{i32} - r_{i31}r_{i22}) \\ k_{i21}(r_{i11}r_{i33} - r_{i31}r_{i13}) + k_{i22}(r_{i21}r_{i33} - r_{i31}r_{i23}) \\ k_{i21}(r_{i11}t_{i3} - r_{i31}t_{i1}) + k_{i22}(r_{i21}t_{i3} - r_{i31}t_{i2}) \end{bmatrix}, \tag{F.51}$$

$$\mathbf{a}_{3i} = \begin{bmatrix} k_{i11}(r_{i12}r_{i31} - r_{i32}r_{i11}) + k_{i12}(r_{i22}r_{i31} - r_{i32}r_{i21}) \\ 0 \\ k_{i11}(r_{i12}r_{i33} - r_{i32}r_{i13}) + k_{i12}(r_{i22}r_{i33} - r_{i32}r_{i23}) \\ k_{i11}(r_{i12}t_{i3} - r_{i32}t_{i1}) + k_{i12}(r_{i22}t_{i3} - r_{i32}t_{i2}) \end{bmatrix}, \quad \mathbf{a}_{4i} = \begin{bmatrix} k_{i21}(r_{i12}r_{i31} - r_{i32}r_{i11}) + k_{i22}(r_{i22}r_{i31} - r_{i32}r_{i21}) \\ 0 \\ k_{i21}(r_{i12}r_{i33} - r_{i32}r_{i13}) + k_{i22}(r_{i22}r_{i33} - r_{i32}r_{i23}) \\ k_{i21}(r_{i12}t_{i3} - r_{i32}t_{i1}) + k_{i22}(r_{i22}t_{i3} - r_{i32}t_{i2}) \end{bmatrix}, \tag{F.52}$$

$$\mathbf{a}_{5i} = \begin{bmatrix} k_{i11}(r_{i13}r_{i31} - r_{i33}r_{i11}) + k_{i12}(r_{i23}r_{i31} - r_{i33}r_{i21}) \\ k_{i11}(r_{i13}r_{i32} - r_{i33}r_{i12}) + k_{i12}(r_{i23}r_{i32} - r_{i33}r_{i22}) \\ 0 \\ k_{i11}(r_{i13}t_{i3} - r_{i33}t_{i1}) + k_{i12}(r_{i23}t_{i3} - r_{i33}t_{i2}) \end{bmatrix}, \quad \mathbf{a}_{6i} = \begin{bmatrix} k_{i21}(r_{i13}r_{i31} - r_{i33}r_{i11}) + k_{i22}(r_{i23}r_{i31} - r_{i33}r_{i21}) \\ k_{i21}(r_{i13}r_{i32} - r_{i33}r_{i12}) + k_{i22}(r_{i23}r_{i32} - r_{i33}r_{i22}) \\ 0 \\ k_{i21}(r_{i13}t_{i3} - r_{i33}t_{i1}) + k_{i22}(r_{i23}t_{i3} - r_{i33}t_{i2}) \end{bmatrix}. \tag{F.53}$$

The vector $\mathbf{a}_{1i}, \cdots, \mathbf{a}_{6i}$ can also be written in the following form:

$$\mathbf{a}_{1i} = k_{i11}\mathbf{b}_{1i} + k_{i12}\mathbf{b}_{2i}, \quad \mathbf{a}_{2i} = k_{i21}\mathbf{b}_{1i} + k_{i22}\mathbf{b}_{2i}, \quad \mathbf{a}_{3i} = k_{i11}\mathbf{b}_{3i} + k_{i12}\mathbf{b}_{4i}, \tag{F.54}$$

$$\mathbf{a}_{4i} = k_{i21}\mathbf{b}_{3i} + k_{i22}\mathbf{b}_{4i}, \quad \mathbf{a}_{5i} = k_{i11}\mathbf{b}_{5i} + k_{i12}\mathbf{b}_{6i}, \quad \mathbf{a}_{6i} = k_{i21}\mathbf{b}_{5i} + k_{i22}\mathbf{b}_{6i}, \tag{F.55}$$

with

$$\mathbf{b}_{1i} := r_{i11}\,[0, r_{i32}, r_{i33}, t_{i3}]^\top - r_{i31}\,[0, r_{i12}, r_{i13}, t_{i1}]^\top, \quad \mathbf{b}_{2i} := r_{i21}\,[0, r_{i32}, r_{i33}, t_{i3}]^\top - r_{i31}\,[0, r_{i22}, r_{i23}, t_{i2}]^\top, \tag{F.56}$$

$$\mathbf{b}_{3i} := r_{i12}\,[r_{i31}, 0, r_{i33}, t_{i3}]^\top - r_{i32}\,[r_{i11}, 0, r_{i13}, t_{i1}]^\top, \quad \mathbf{b}_{4i} := r_{i22}\,[r_{i31}, 0, r_{i33}, t_{i3}]^\top - r_{i32}\,[r_{i21}, 0, r_{i23}, t_{i2}]^\top, \tag{F.57}$$

$$\mathbf{b}_{5i} := r_{i13}\,[r_{i31}, r_{i32}, 0, t_{i3}]^\top - r_{i33}\,[r_{i11}, r_{i12}, 0, t_{i1}]^\top, \quad \mathbf{b}_{6i} := r_{i23}\,[r_{i31}, r_{i32}, 0, t_{i3}]^\top - r_{i33}\,[r_{i21}, r_{i22}, 0, t_{i2}]^\top. \tag{F.58}$$

Putting (F.48)–(F.50) together in one vector, we get

$$\begin{bmatrix} \partial(u_{\text{error}})_i/\partial x^w_{\text{est}} \\ \partial(v_{\text{error}})_i/\partial x^w_{\text{est}} \\ \partial(u_{\text{error}})_i/\partial y^w_{\text{est}} \\ \partial(v_{\text{error}})_i/\partial y^w_{\text{est}} \\ \partial(u_{\text{error}})_i/\partial z^w_{\text{est}} \\ \partial(v_{\text{error}})_i/\partial z^w_{\text{est}} \end{bmatrix} = ((\mathbf{P}_i)_{\text{row3}}\widetilde{\mathbf{x}}^w_{\text{est}})^{-2}\, \mathbf{A}_i^\top \widetilde{\mathbf{x}}^w_{\text{est}} \quad \text{with} \quad \mathbf{A}_i = [\mathbf{a}_{1i}\ \mathbf{a}_{2i}\ \mathbf{a}_{3i}\ \mathbf{a}_{4i}\ \mathbf{a}_{5i}\ \mathbf{a}_{6i}]\,. \tag{F.59}$$

Therefore, the Jacobian matrix for camera c_i is given by

$$\mathbf{J}_i = \begin{bmatrix} \dfrac{\partial(u_{\text{error}})_i}{\partial x^w_{\text{est}}} & \dfrac{\partial(u_{\text{error}})_i}{\partial y^w_{\text{est}}} & \dfrac{\partial(u_{\text{error}})_i}{\partial z^w_{\text{est}}} \\ \dfrac{\partial(v_{\text{error}})_i}{\partial x^w_{\text{est}}} & \dfrac{\partial(v_{\text{error}})_i}{\partial y^w_{\text{est}}} & \dfrac{\partial(v_{\text{error}})_i}{\partial z^w_{\text{est}}} \end{bmatrix} \tag{F.60}$$

$$= \text{vec}^{-1}_{2\times 3}\left(((\mathbf{P}_i)_{\text{row3}}\widetilde{\mathbf{x}}^w_{\text{est}})^{-2}\, \mathbf{A}_i^\top \widetilde{\mathbf{x}}^w_{\text{est}}\right) \tag{F.61}$$

$$= ((\mathbf{P}_i)_{\text{row3}}\widetilde{\mathbf{x}}^w_{\text{est}})^{-2}\, \text{vec}^{-1}_{2\times 3}\left(\mathbf{A}_i^\top \widetilde{\mathbf{x}}^w_{\text{est}}\right). \tag{F.62}$$

G Monotone Smoothing used in Section 6.3.3

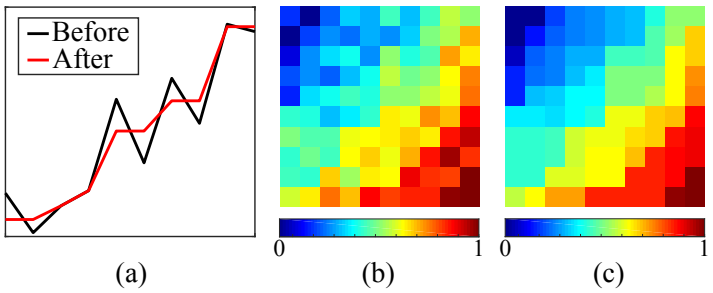


Figure G.1: (a): Monotone smoothing example on 1D data. (b)–(c): 2D example before and after smoothing.

We propose a simple smoothing method that enforces the monotonicity. Our method iteratively updates the data using the following rule:

$$x_{\text{new}} \leftarrow \frac{1}{|\mathcal{N}|} \sum_{i \in \mathcal{N}} f(x_{\text{old}}, x_i) \tag{G.63}$$

$$\text{with} \quad f(x_{\text{old}}, x_i) = \begin{cases} x_{\text{old}} & \text{if } x_{\text{old}} \text{ and } x_i \text{ fulfills the desired monotonicity,} \\ 0.5(x_{\text{old}} + x_i) & \text{otherwise,} \end{cases}$$

where x_{old} and x_{new} are the data values before and after the update, $\mathcal{N}$ is the set of its neighbors, and $x_{i \in \mathcal{N}}$ is the data value of the i-th neighbor. Basically, (G.63) means that each neighbor of the data point x_{old} votes on how much x_{old} should change. If the neighbor x_i is already fulfilling the desired monotonicity, it votes for no change. Otherwise, it votes for changing x_{old} to $0.5(x_{\text{old}} + x_i)$ in order to bring it closer to itself. Once every neighbor casts a vote, we average it to determine x_{new}. Since we do not want the update order to influence the result, we use the pre-update values for the neighbor x_i. Only after every data point is assigned a new value, we update them altogether at once. We repeat this process until convergence.

Fig. G.1(a) illustrates an example on 1D data. Each data point has either one neighbor (if it is located at the boundary) or two neighbors. Notice that our method flattens out the "bumps" that violate the desired monotonicity. Fig. G.1(b)–(c) shows an example on 2D data. In this case, each data point has either two neighbors (if it is located at the corner), three neighbors (if it is located at the edge), or four neighbors. In contrast to the noisy input, the output is both smooth and monotonic.

In Section 6.3.3, we apply the smoothing on a 3D grid.

H More Results from ROBA on the Synthetic Dataset

In Chapter 10, we compared the rotational accuracy of RA and ROBA (initialized by RA) in terms of the mean angular error after the L_1 alignment. Here, we show additional comparisons in other metrics, namely the median angular error after the L_1 alignment, and the mean/median angular error after the L_2 alignment. Fig. H.2–H.4 and Table H.3 present the results. With these additional metrics, we reach the same conclusion as in Chapter 10. That is,

1. ROBA improves the results of RA in all scenarios considered.

2. Both RA and ROBA yield more accurate results for fewer views, farther points and pure rotations, all of which lead to a denser view-graph. In these cases, the relative error reduction by ROBA is also larger.

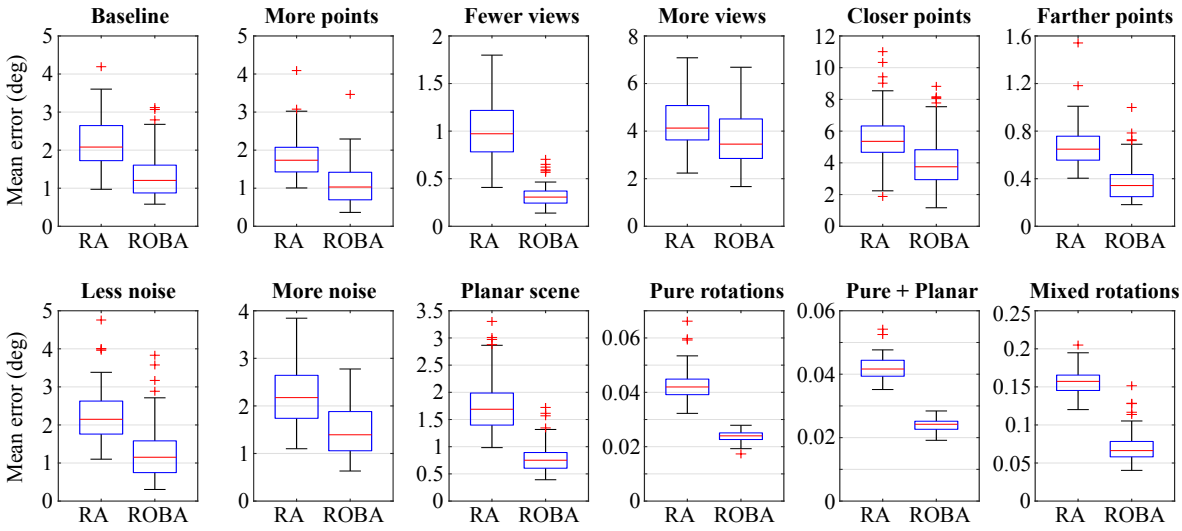


Figure H.2: **[Synthetic - md1]** Comparison of RA and ROBA (initialized by RA) in terms of the median angular error after L_1 alignment.

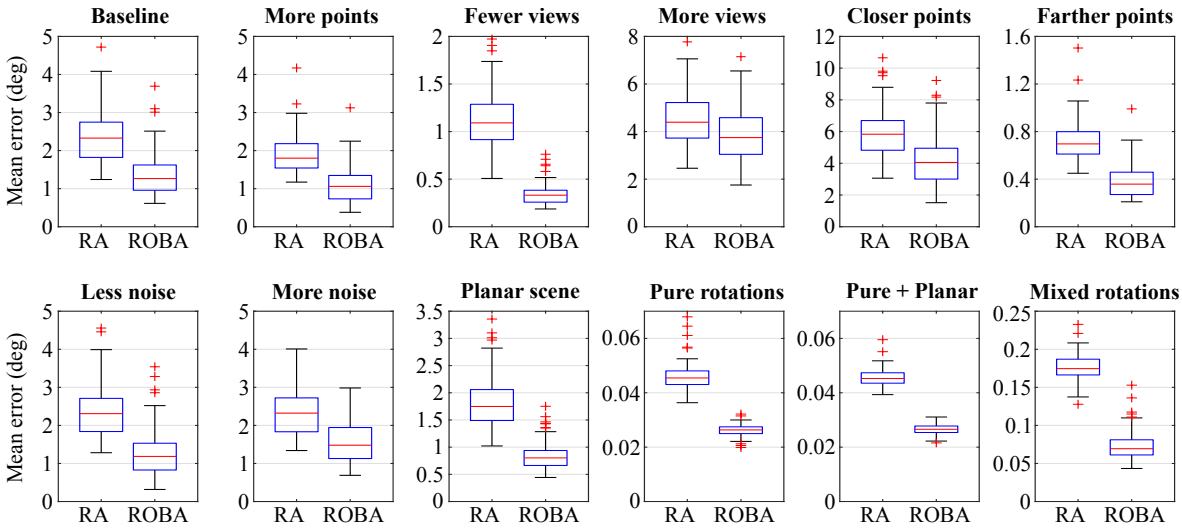


Figure H.3: **[Synthetic - mn2]** Comparison of RA and ROBA (initialized by RA) in terms of the mean angular error after L_2 alignment.

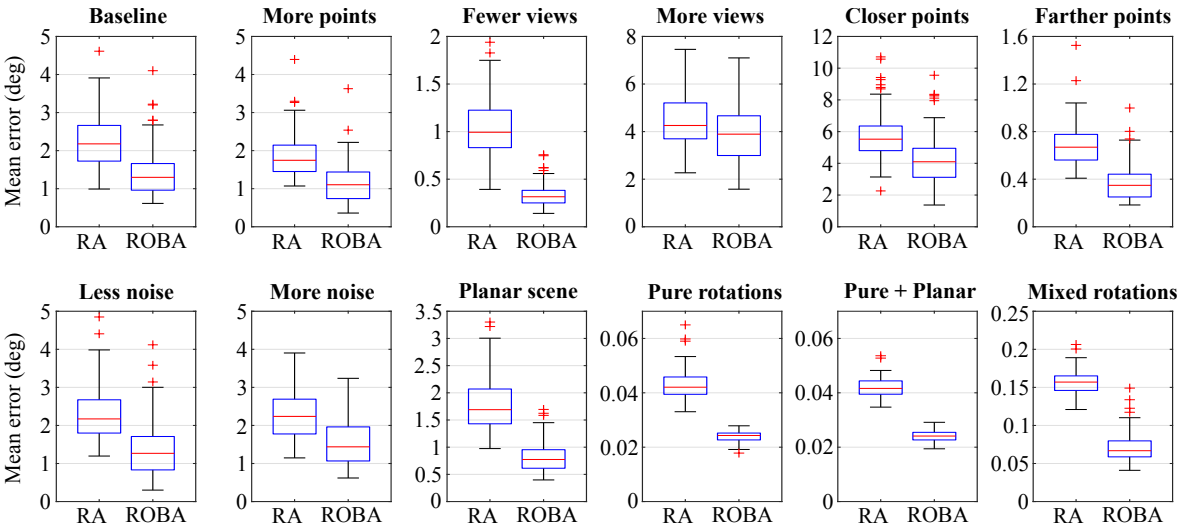


Figure H.4: **[Synthetic - md2]** Comparison of RA and ROBA (initialized by RA) in terms of the median angular error after L_2 alignment.

Settings	$\%_{\mathcal{E}}$	$\bar{e}_{\mathcal{E}}$	$\tilde{e}_{\mathcal{E}}$	RA				RA + ROBA (100 iter)				$\%_{\text{better}}$			
				mn1	md1	mn2	md2	mn1	md1	mn2	md2	mn1	md1	mn2	md2
Baseline	6.0%	2.21	1.39	2.31	2.08	2.33	2.18	**1.26**	**1.20**	**1.26**	**1.30**	100%	99%	100%	100%
More points	7.2%	2.51	1.52	1.78	1.73	1.80	1.75	**1.05**	**1.03**	**1.06**	**1.10**	100%	100%	100%	100%
Fewer camera	21%	2.23	1.38	1.08	0.97	1.09	0.99	**0.33**	**0.31**	**0.33**	**0.32**	100%	100%	100%	100%
More views	2.0%	2.24	1.40	4.35	4.13	4.39	4.26	**3.73**	**3.45**	**3.75**	**3.89**	100%	98%	100%	97%
Closer points	4.0%	2.57	1.65	5.74	5.35	5.83	5.52	**3.99**	**3.75**	**4.05**	**4.10**	100%	97%	100%	99%
Farther points	10%	1.97	1.22	0.69	0.65	0.70	0.67	**0.36**	**0.34**	**0.36**	**0.35**	100%	100%	100%	100%
Less noise	6.0%	2.25	1.40	2.30	2.15	2.31	2.17	**1.17**	**1.15**	**1.19**	**1.27**	100%	99%	100%	100%
More noise	5.9%	2.23	1.38	2.31	2.17	2.32	2.24	**1.48**	**1.39**	**1.48**	**1.44**	100%	98%	100%	99%
Planar scene	7.4%	2.61	1.56	1.74	1.69	1.75	1.69	**0.80**	**0.75**	**0.80**	**0.77**	100%	100%	100%	100%
Pure rotations	100%	0.89	0.74	0.045	0.042	0.045	0.042	**0.026**	**0.024**	**0.026**	**0.024**	100%	100%	100%	100%
Pure + Planar	100%	0.89	0.74	0.045	0.042	0.045	0.042	**0.027**	**0.024**	**0.027**	**0.024**	100%	100%	100%	100%
Mixed rotations	37%	3.13	1.66	0.17	0.16	0.17	0.16	**0.069**	**0.066**	**0.069**	**0.067**	100%	100%	100%	100%

$\%_{\mathcal{E}}$, $\%_{\text{better}}$: proportion of existing edges and improved results,
$\bar{e}_{\mathcal{E}}$, $\tilde{e}_{\mathcal{E}}$: mean and median angular errors (in deg) of the relative rotations from all edges.

Table H.3: **[Synthetic datasets]** Median results of the 100 simulations in each setting.

I Evolution of Total Cost and Rotation Errors of ROBA on the Real Datasets

In Fig. I.5, we plot the evolution of the relative errors aggregated from all 15 real-world datasets. Notice that most of the improvement occurs in less than 50 iterations. In Fig. I.6–I.20, we show the evolution of the total cost and the different rotation error metrics for each dataset. We make the following observations:

1. The cost always overshoots after the first rotation update. From the second update onward, it starts decreasing with occasional "mini" overshoots. From additional experiments, we found that reducing the step size α in the Adam optimizer prevents this overshoot, but this leads to a slower rate of convergence. Interestingly, the overshoot of the total cost does not necessarily lead to the overshoot of the rotation errors.

2. On most datasets, the cost converges after 30–40 iterations. However, a relatively small change in the total cost can sometimes lead to a non-negligible change in the angular errors. For example, see the results for MDR, PDP and TOL in Fig. I.9, I.13, I.16 after 40 iterations.

3. For ROF dataset in Fig. I.15, the final total cost after 100 iterations is larger than the initial total cost. However, all of the rotation error metrics are reduced.

These observations indicate that there is a discrepancy between the total cost and the rotation errors. This suggests that when we implement the stopping criteria for the iterations, we need to consider not only the change in the total cost, but also the number of iterations and the angular change in the rotations.

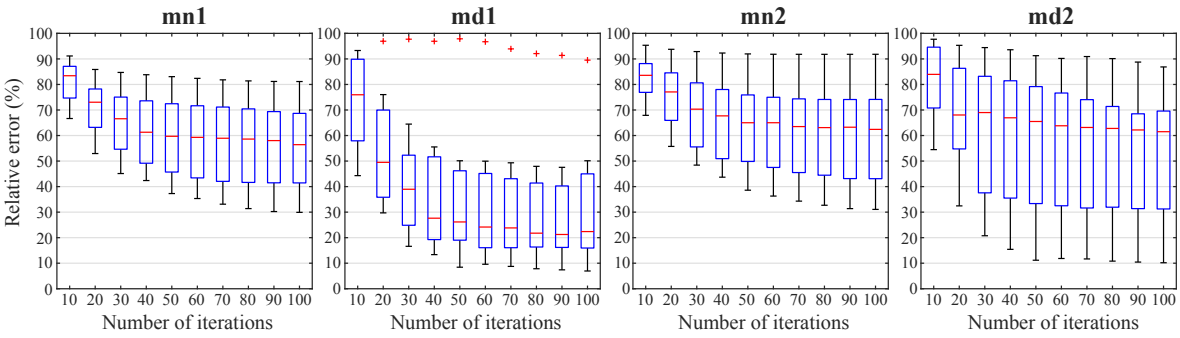


Figure I.5: **[Real datasets]** Relative errors with respect to the initial errors. mn/md/1/2: mean/median angular error after L_1/L_2 alignment.

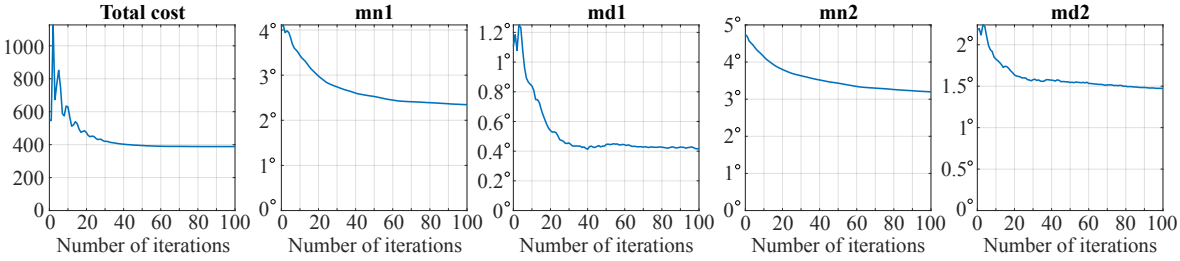


Figure I.6: **[ALM - Alamo]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

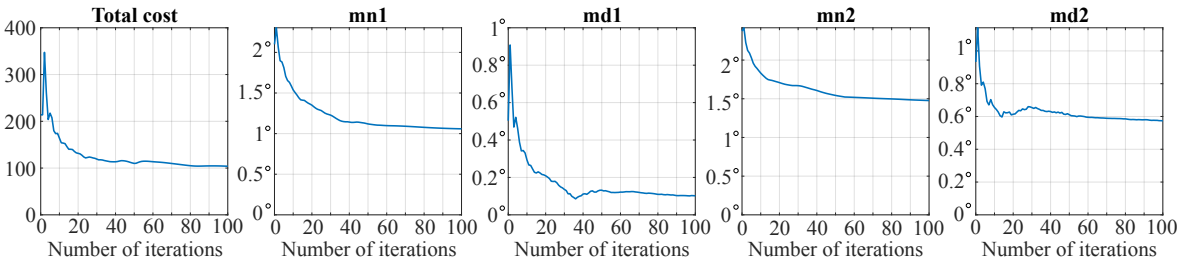


Figure I.7: **[ELS - Ellis Island]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

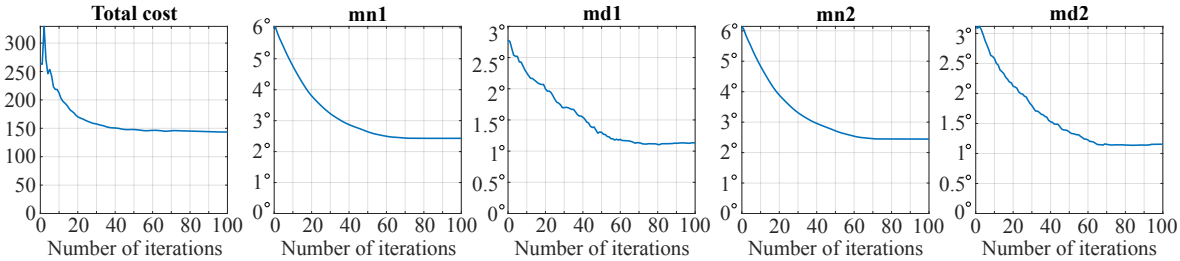


Figure I.8: **[GDM - Gendarmenmarkt]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

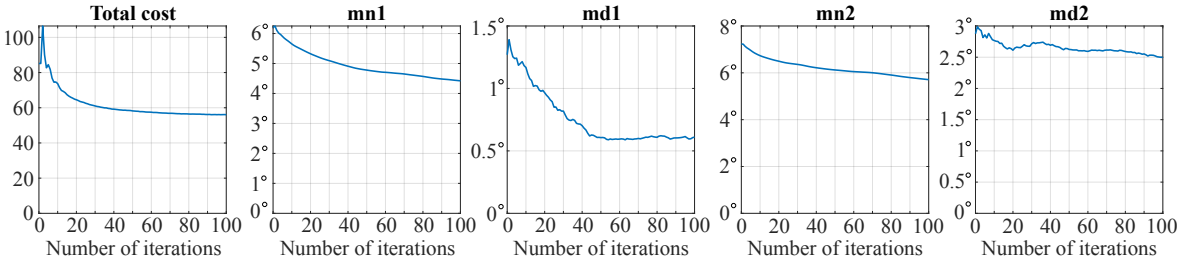


Figure I.9: **[MDR - Madrid Metropolis]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

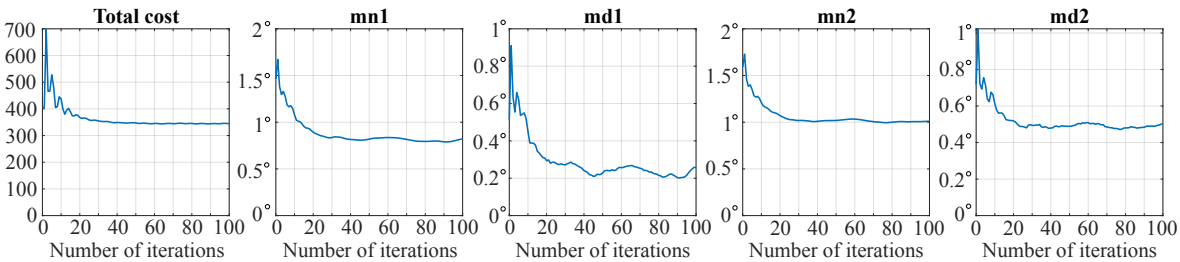


Figure I.10: **[MND - Montreal Notre Dame]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

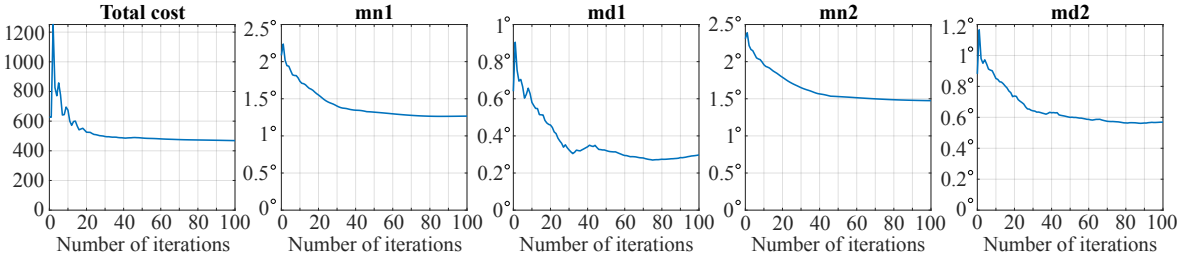


Figure I.11: **[NTD - Notre Dame]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

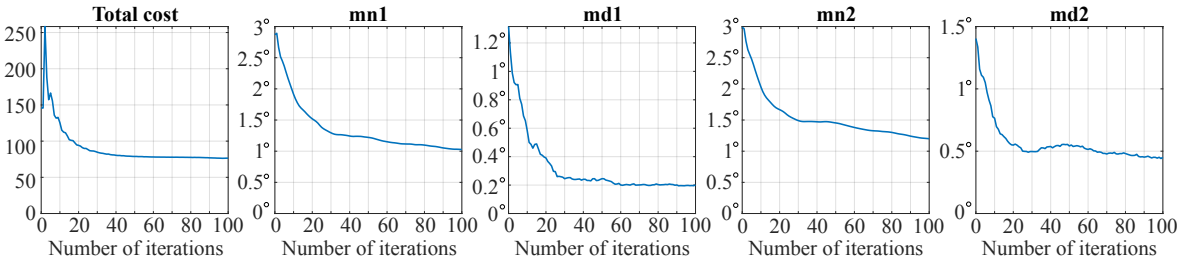


Figure I.12: **[NYC - NYC Library]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

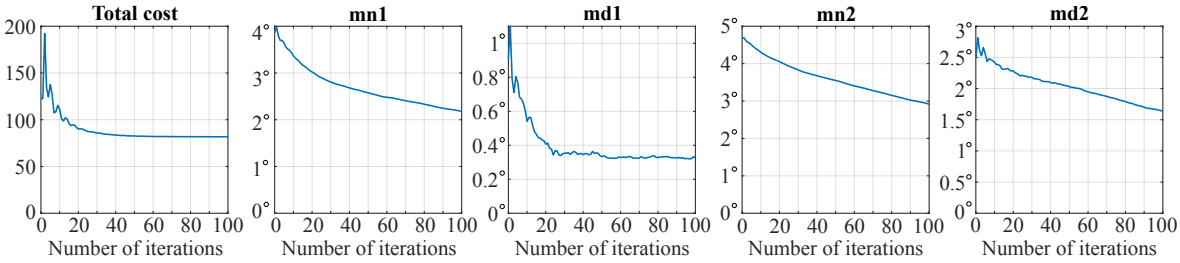


Figure I.13: **[PDP - Piazza del Popolo]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

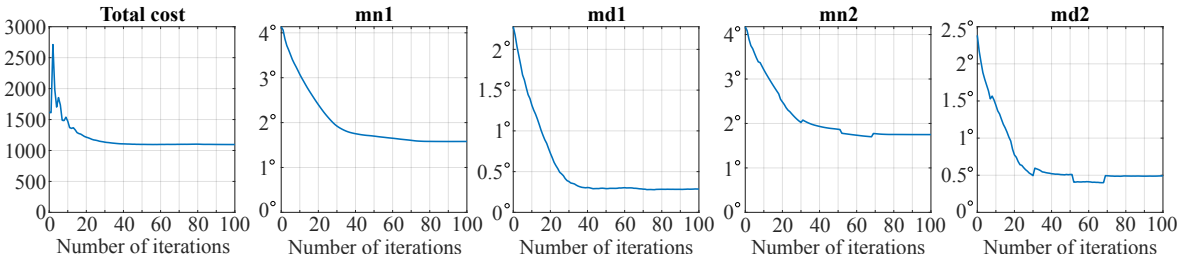


Figure I.14: **[PIC - Picadilly]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

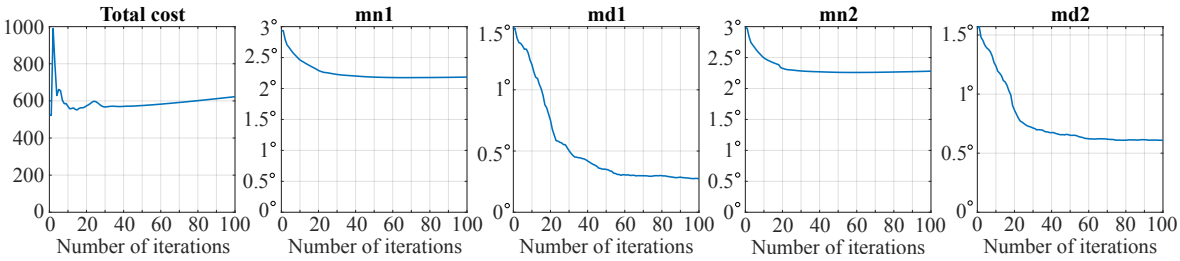


Figure I.15: **[ROF - Roman Forum]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

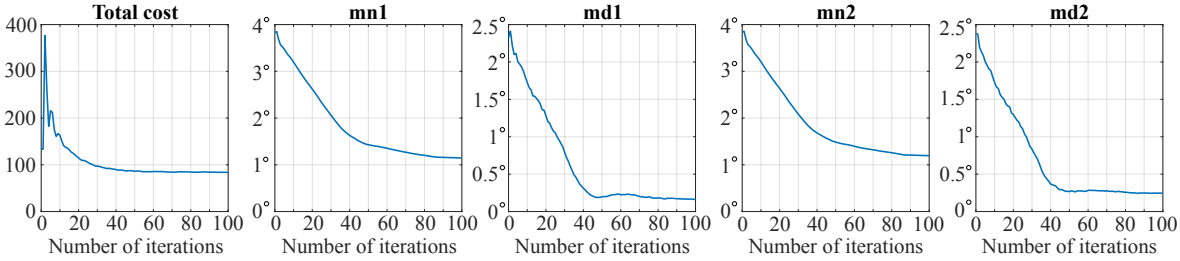


Figure I.16: **[TOL - Tower of London]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

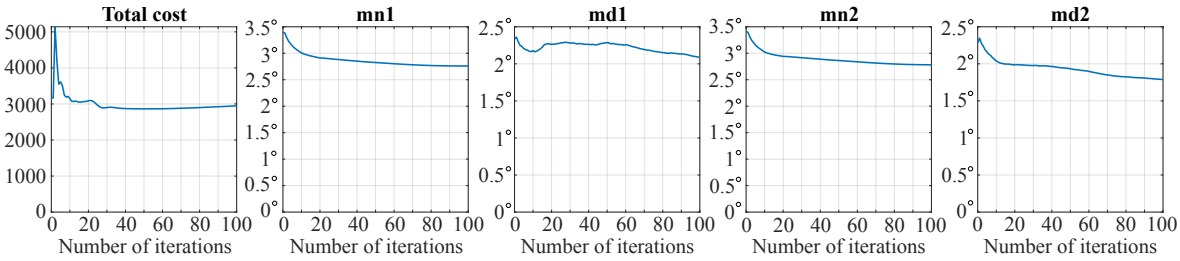


Figure I.17: **[TFG - Trafalgar]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

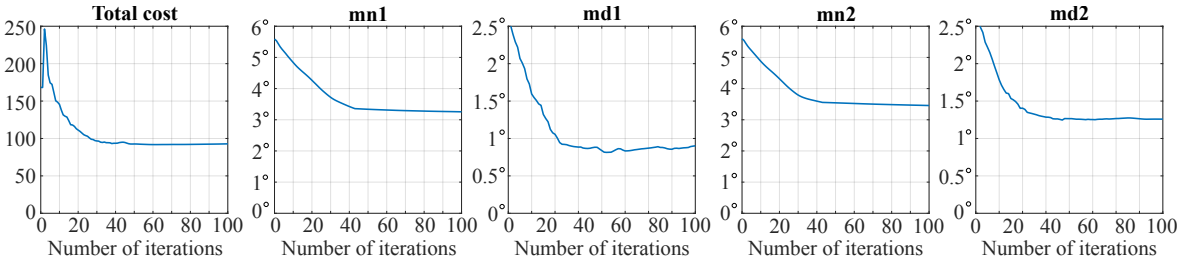


Figure I.18: **[USQ - Union Square]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

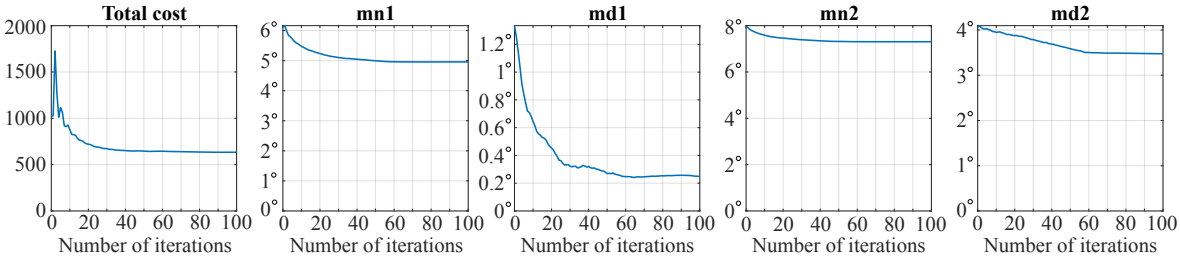


Figure I.19: **[VNC - Vienna Cathedral]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

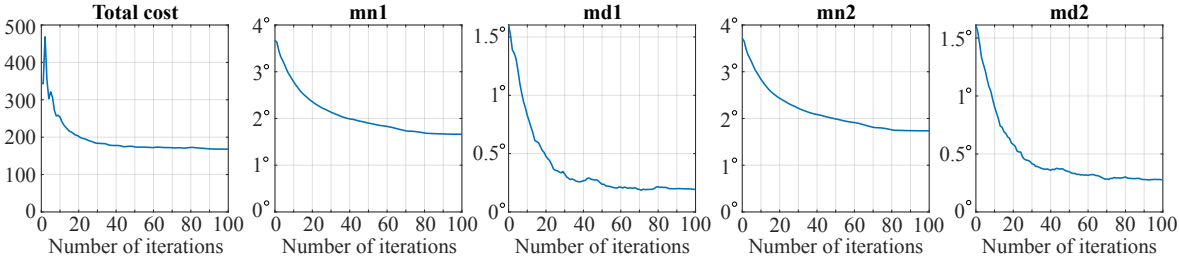


Figure I.20: **[YKM - Yorkminster]** mn/md/1/2: mean/median angular error (in deg) after the L_1/L_2 alignment,

J Proof of Degeneracy in Section 12.7

First, it is useful to know the following identity (Eade, 2017):

$$\mathbf{R}\mathrm{Exp}(\mathbf{v})\mathbf{R}^\top = \mathrm{Exp}\left(\mathbf{R}\mathbf{v}\right). \tag{J.64}$$

This means that for any rotation matrices $\mathbf{R}_1$ and $\mathbf{R}_2$,

$$\angle\left(\mathbf{R}_1\mathbf{R}_2\mathbf{R}_1^\top\right) = \angle\left(\mathbf{R}_2\right), \tag{J.65}$$

where $\angle(\cdot)$ denotes the angle of the rotation. When orientations are related by rotations around the same axis, they can be parameterized in two different ways using (J.64):

$$\mathbf{R}_i = \mathrm{Exp}(\theta_i\widehat{\mathbf{v}})\mathbf{R}_1 = \mathbf{R}_1\mathrm{Exp}(\theta_i\widehat{\mathbf{w}}) \quad \text{for all } i, \tag{J.66}$$

where θ_i is the signed angle of rotation between the first and the ith orientation, and $\widehat{\mathbf{v}}$ and $\widehat{\mathbf{w}}$ are the axes of rotation that are related by $\widehat{\mathbf{v}} = \mathbf{R}_1\widehat{\mathbf{w}}$.

Next, we prove the following proposition: When solving (12.20), degeneracy occurs if all cameras have the same fixed orientation, or if the axes of their relative rotations are all the same. Since the former case is a special instance of the latter case (*i.e.*, when the rotation angle is 0), we focus on the latter here. First, we rewrite the summand in (12.20) as

$$s_i = \angle\left(\widetilde{\mathbf{R}}_{gm,i}\widetilde{\mathbf{R}}_{mc}\mathbf{R}_{ec,i}^\top\mathbf{R}_{\text{align}}^\top\right). \tag{J.67}$$

If the ground-truth orientations $\widetilde{\mathbf{R}}_{gm,i}$ are related to each other by rotations around the same axis (say $\widehat{\mathbf{v}}$), we have

$$\widetilde{\mathbf{R}}_{gm,i} \overset{\text{(J.66)}}{=} \mathrm{Exp}(\theta_i\widehat{\mathbf{v}})\widetilde{\mathbf{R}}_{gm,1}. \tag{J.68}$$

Now, consider the following update:

$$\left(\widetilde{\mathbf{R}}_{mc}\right)_{\text{new}} \leftarrow \widetilde{\mathbf{R}}_{gm,1}^\top\mathrm{Exp}(a\widehat{v})\widetilde{\mathbf{R}}_{gm,1}\widehat{\mathbf{R}}_{mc}, \tag{J.69}$$

$$\left(\mathbf{R}_{\text{align}}\right)_{\text{new}} \leftarrow \mathrm{Exp}(a\widehat{v})\mathbf{R}_{\text{align}}, \tag{J.70}$$

where a is an arbitrary angle. Substituting (J.68), (J.69) and (J.70) into (J.67) and using the fact that the rotations are commutative if they share the same axis, we get

$$s_i = \angle\left(\mathrm{Exp}(a\widehat{\mathbf{v}})\widetilde{\mathbf{R}}_{gm,i}\widetilde{\mathbf{R}}_{mc}\mathbf{R}_{ec,i}^\top\mathbf{R}_{\text{align}}^\top\mathrm{Exp}(a\widehat{\mathbf{v}})^\top\right). \tag{J.71}$$

According to (J.65), the angle given by (J.71) must be the same as that given by (J.67). Thus, by varying a in (J.69) and (J.70), we get infinitely many solutions for $\widetilde{\mathbf{R}}_{mc}$ and $\mathbf{R}_{\text{align}}$ that lead to the exact same cost in (12.20), resulting in degeneracy.

A similar argument can be made when the estimated orientations $\mathbf{R}_{ec,i}$ are related to

one another by rotations around the same axis (say $\widehat{\mathbf{w}}$). In this case, it is convenient to parameterize $\mathbf{R}_{ec,i}$ as follows:

$$\mathbf{R}_{ec,i} \overset{\text{(J.66)}}{=} \mathbf{R}_{ec,1}\text{Exp}(\theta_i\widehat{\mathbf{w}}). \tag{J.72}$$

Now, consider the following update:

$$\left(\widetilde{\mathbf{R}}_{mc}\right)_{\text{new}} \leftarrow \widehat{\mathbf{R}}_{mc}\text{Exp}(a\widehat{\mathbf{w}}), \tag{J.73}$$

$$\left(\mathbf{R}_{\text{align}}\right)_{\text{new}} \leftarrow \mathbf{R}_{\text{align}}\mathbf{R}_{ec,1}\text{Exp}(a\widehat{\mathbf{w}})\mathbf{R}_{ec,1}^{\top}, \tag{J.74}$$

where a is an arbitrary angle. Substituting (J.72), (J.73) and (J.74) into (J.67) and again using the fact that the rotations are commutative if they share the same axis, we end up with the same expression as (J.67). Thus, by varying a in (J.73) and (J.74), we get infinitely many solutions that lead to the exact same cost in (12.20), resulting in degeneracy.

References

Aftab, K. and Hartley, R. (2015), Convergence of iteratively re-weighted least squares to robust M-estimators, *in* 'IEEE Winter Conf. Appl. Comput. Vis.', pp. 480–487.

Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S. M. and Szeliski, R. (2011), 'Building rome in a day', *Commun. ACM* **54**(10), 105–112.

Agarwal, S., Snavely, N. and Seitz, S. M. (2008), Fast algorithms for L_∞ problems in multiview geometry, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1–8.

Agarwal, S., Snavely, N., Seitz, S. M. and Szeliski, R. (2010), Bundle adjustment in the large, *in* 'Eur. Conf. Comput. Vis.', pp. 29–42.

Aholt, C., Agarwal, S. and Thomas, R. (2012), A QCQP approach to triangulation, *in* 'Eur. Conf. Comput. Vis.', pp. 654–667.

Alismail, H., Browning, B. and Lucey, S. (2017), Photometric bundle adjustment for vision-based slam, *in* 'Asian Conf. Comput. Vis.', pp. 324–341.

Angelov, P. and Yager, R. (2013), 'Density-based averaging – a new operator for data fusion', *Information Sciences* **222**, 163–174. Including Special Section on New Trends in Ambient Intelligence and Bio-inspired Systems.

Antonante, P., Tzoumas, V., Yang, H. and Carlone, L. (2022), 'Outlier-robust estimation: Hardness, minimally tuned algorithms, and applications', *IEEE Trans. Robot.* **38**(1), 281–301.

Arie-Nachimson, M., Kovalsky, S. Z., Kemelmacher-Shlizerman, I., Singer, A. and Basri, R. (2012), Global motion estimation from point matches, *in* 'Int. Conf. 3D Imaging, Modeling, Processing, Visualization & Transmission', pp. 81–88.

Arrigoni, F., Magri, L., Rossi, B., Fragneto, P. and Fusiello, A. (2014), Robust absolute rotation estimation via low-rank and sparse matrix decomposition, *in* 'IEEE Int. Conf. 3D Vis.', Vol. 1, pp. 491–498.

Arrigoni, F., Rossi, B., Fragneto, P. and Fusiello, A. (2018), 'Robust synchronization in SO(3) and SE(3) via low-rank and sparse matrix decomposition', *Comput. Vis. and Image Underst.* **174**, 95–113.

Arrigoni, F., Rossi, B. and Fusiello, A. (2016), Global registration of 3D point sets via LRS decomposition, *in* 'Eur. Conf. Comput. Vis.', pp. 489–504.

Arun, K. S., Huang, T. S. and Blostein, S. D. (1987), 'Least-squares fitting of two 3-D point sets', *IEEE Trans. Pattern Anal. Mach. Intell.* **9**(5), 698–700.

Baker, S. and Matthews, I. (2004), 'Lucas-Kanade 20 years on: A unifying framework', *Int. J. Comput. Vis.* **56**(3), 221–255.

Beardsley, P. A., Zisserman, A. and Murray, D. W. (1994), Navigation using affine structure from motion, *in* 'Eur. Conf. Comput. Vis.', pp. 85–96.

Beardsley, P. A., Zisserman, A. and Murray, D. W. (1997), 'Sequential updating of projective and affine structure from motion', *Int. J. Comput. Vis.* **23**(3), 235–259.

Beaton, A. E. and Tukey, J. W. (1974), 'The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data', *Technometrics* **16**(2), 147–185.

Bednar, J. and Watt, T. (1984), 'Alpha-trimmed means and their relationship to median filters', *IEEE Transactions on Acoustics, Speech, and Signal Processing* **32**(1), 145–153.

Beliakov, G., Sola, H. B. and Calvo, T. (2016), *A Practical Guide to Averaging Functions*, Vol. 329 of *Studies in Fuzziness and Soft Computing*, Springer.

Bhattacharya, U. and Govindu, V. M. (2019), Efficient and robust registration on the 3D special euclidean group, *in* 'Int. Conf. Comput. Vis.'.

Blake, A. and Zisserman, A. (1987), *Visual Reconstruction*, MIT Press, Cambridge, MA, USA.

Boumal, N., Singer, A. and Absil, P.-A. (2013), Robust estimation of rotations from relative measurements by maximum likelihood, *in* 'IEEE Conf. Decision Control', pp. 1156–1161.

Bourmaud, G. (2016), Online variational bayesian motion averaging, *in* 'Eur. Conf. Comput. Vis.', pp. 126–142.

Bourmaud, G., Mégret, R., Giremus, A. and Berthoumieu, Y. (2014), Global motion estimation from relative measurements in the presence of outliers, *in* 'Asian Conf. Comput. Vis.', pp. 366–381.

Brachmann, E., Humenberger, M., Rother, C. and Sattler, T. (2021), On the limits of pseudo ground truth in visual camera re-localisation, *in* 'IEEE Int. Conf. Coput. Vis.'.

Brachmann, E. and Rother, C. (2019), Neural-guided RANSAC: Learning where to sample model hypotheses, *in* 'Int. Conf. Comput. Vis.', pp. 4322–4331.

Briales, J., Kneip, L. and Gonzalez-Jimenez, J. (2018), A certifiably globally optimal solution to the non-minimal relative pose problem, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 145–154.

Bryan, M. F., Cecchetti, S. G. and Wiggins, R. L. (1997), Efficient inflation estimation, Technical Report 6183, National Bureau of Economic Research.

Bu, S., Zhao, Y., Wan, G., Li, K., Cheng, G. and Liu, Z. (2017), 'Semi-direct tracking and mapping with RGB-D camera for MAV', *Multimedia Tools and Applications* **76**(3), 4445–4469.

Burgard, W., Stachniss, C., Grisetti, G., Steder, B., Kümmerle, R., Dornhege, C., Ruhnke, M., Kleiner, A. and Tardös, J. D. (2009), A comparison of SLAM algorithms based on a graph of relations, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.', pp. 2089–2095.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W. and Siegwart, R. (2016), 'The EuRoC micro aerial vehicle datasets', *Int. J. Robot. Res.* .

Bustos, A. P., Chin, T.-J., Eriksson, A. and Reid, I. (2019), Visual SLAM: Why bundle adjust?, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 2385–2391.

Byröd, M., Josephson, K. and Åström, K. (2007), Fast optimal three view triangulation, *in* 'Asian Conf. Comput. Vis.', pp. 549–559.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I. and Leonard, J. J. (2016), 'Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age', *IEEE Trans. Robot.* **32**(6), 1309–1332.

Cai, Q., Zhang, L., Wu, Y., Yu, W. and Hu, D. (2021), 'A pose-only solution to visual reconstruction and navigation', *IEEE Trans. Pattern Anal. Mach. Intell.* pp. 1–1.

Campos, C., Elvira, R., Rodríguez, J. J. G., M. Montiel, J. M. and D. Tardós, J. (2021), 'ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM', *IEEE Trans. Robot.* **37**(6), 1874–1890.

Carlone, L., Rosen, D. M., Calafiore, G., Leonard, J. J. and Dellaert, F. (2015), Lagrangian duality in 3D slam: Verification techniques and optimal solutions, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.', pp. 125–132.

Carlone, L., Tron, R., Daniilidis, K. and Dellaert, F. (2015), Initialization techniques for 3D SLAM: A survey on rotation estimation and its use in pose graph optimization, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 4597–4604.

Chatterjee, A. and Govindu, V. M. (2013), Efficient and robust large-scale rotation averaging, *in* 'Int. Conf. Comput. Vis.', pp. 521–528.

Chatterjee, A. and Govindu, V. M. (2018), 'Robust relative rotation averaging', *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(4), 958–972.

Chaudhury, K. N. and Singer, A. (2012), 'Non-local Euclidean medians', *IEEE Sign. Process. Letters* **19**(11), 745–748.

Chen, Y., Zhao, J. and Kneip, L. (2021), Hybrid rotation averaging: A fast and robust rotation averaging approach, *in* 'IEEE Conf. Comput. Vis. Pattern Recognit.', pp. 10358–10367.

Chesi, G. (2014), 'LMI-based estimation of scene points in vision systems with generalized cameras', *IEEE Trans. Automatic Control* **59**(11), 2996–3001.

Chng, C.-K., Parra, A., Chin, T.-J. and Latif, Y. (2020), Monocular rotational odometry with incremental rotation averaging and loop closure, *in* 'Digital Image Computing: Techniques and Applications', pp. 1–8.

Chum, O., Matas, J. and Kittler, J. (2003), Locally optimized RANSAC, *in* 'Pattern Recognition', pp. 236–243.

Civera, J., Davison, A. J. and Montiel, J. M. M. (2008*a*), Interacting multiple model monocular slam, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 3704–3709.

Civera, J., Davison, A. J. and Montiel, J. M. M. (2008*b*), 'Inverse depth parametrization for monocular SLAM', *IEEE Trans. Robot.* **24**(5), 932–945.

Concha, A. and Civera, J. (2015), DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.', pp. 5686–5693.

Crandall, D., Owens, A., Snavely, N. and Huttenlocher, D. (2011), Discrete-continuous optimization for large-scale structure from motion, *in* 'IEEE Conf. Comput. Vis. Pattern Recognit.', pp. 3001–3008.

Cui, H., Gao, X., Shen, S. and Hu, Z. (2017), HSfM: Hybrid Structure-from-Motion, *in* 'IEEE Conf. Comput. Vis. Pattern Recognit.'.

Cui, H., Shen, S. and Gao, W. (2018), Voting-based incremental structure-from-motion, *in* 'Int. Conf. Pattern Recog.', pp. 1929–1934.

Cui, Z., Jiang, N., Tang, C. and Tan, P. (2015), Linear global translation estimation with feature tracks, *in* 'Brit. Mach. Vis. Conf.', pp. 46.1–46.13.

Cui, Z. and Tan, P. (2015), Global structure-from-motion by similarity averaging, *in* 'IEEE Int. Conf. Coput. Vis.', pp. 864–872.

Dai, Y., Trumpf, J., Li, H., Barnes, N. and Hartley, R. (2009), Rotation averaging with application to camera-rig calibration, *in* 'Asian Conf. Comput. Vis.', pp. 335–346.

Dai, Z., Wu, Y., Zhang, F. and Wang, H. (2012), A novel fast method for L_∞ problems in multiview geometry, *in* 'Eur. Conf. Comput. Vis.', pp. 116–129.

Davis, M. A. and Jr., D. L. P. (n.d.), 'Central tendencies, measures of', International Encyclopedia of the Social Sciences dictionary.
URL: *https://www.encyclopedia.com/social-sciences/applied-and-social-sciences-magazines/central-tendencies-measures*

Davison, A. J., Reid, I. D., Molton, N. D. and Stasse, O. (2007), 'MonoSLAM: Real-time single camera SLAM', *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(6), 1052–1067.

Delaunoy, A. and Pollefeys, M. (2014), Photometric bundle adjustment for dense multi-view 3D modeling, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1486–1493.

Dellaert, F., Rosen, D. M., Wu, J., Mahony, R. and Carlone, L. (2020*a*), Shonan rotation averaging: Global optimality by surfing $\mathrm{SO(p)^n}$, *in* 'Eur. Conf. Comput. Vis.', pp. 292–308.

Dellaert, F., Rosen, D. M., Wu, J., Mahony, R. and Carlone, L. (2020*b*), Shonan rotation averaging: Global optimality by surfing $so(p)^n$, *in* A. Vedaldi, H. Bischof, T. Brox and J.-M. Frahm, eds, 'Eur. Conf. Comput. Vis.', pp. 292–308.

Delon, J. and Rougé, B. (2007), 'Small baseline stereovision', *Journal of Mathematical Imaging and Vision* **28**(3), 209–223.

Dias Curto, J. (2021), 'Averages: There is still something to learn', *Computational Economics* .

Eade, E. (2017), Lie groups for 2D and 3D transformations, Technical report.

Eigen Library v3 (n.d.), `http://eigen.tuxfamily.org`.

Engel, J., Koltun, V. and Cremers, D. (2016), 'A photometrically calibrated benchmark for monocular visual odometry', *CoRR* **abs/1607.02555**.

Engel, J., Koltun, V. and Cremers, D. (2018), 'Direct sparse odometry', *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(3), 611–625.

Engel, J., Schöps, T. and Cremers, D. (2014), LSD-SLAM: Large-scale direct monocular SLAM, *in* 'Eur. Conf. Comput. Vis.', pp. 834–849.

Engel, J., Sturm, J. and Cremers, D. (2013), Semi-dense visual odometry for a monocular camera, *in* 'IEEE Int. Conf. Comput. Vis.', pp. 1449–1456.

Enqvist, O., Kahl, F. and Olsson, C. (2011), Non-sequential structure from motion, *in* 'IEEE Int. Conf. Comput. Vis.', pp. 264–271.

Eriksson, A., Olsson, C., Kahl, F. and Chin, T.-J. (2018), Rotation averaging and strong duality, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 127–135.

Eriksson, A., Olsson, C., Kahl, F. and Chin, T.-J. (2021), 'Rotation averaging with the chordal distance: Global minimizers and strong duality', *IEEE Trans. Pattern Anal. Mach. Intell.* **43**(1), 256–268.

Fathian, K., Ramirez-Paredes, J. P., Doucette, E. A., Curtis, J. W. and Gans, N. R. (2018), 'QuEst: A Quaternion-based approach for camera motion estimation from minimal feature points', *IEEE Robot. Automat. Lett.* **3**(2), 857–864.

Ferraz, L., Binefa, X. and Moreno-Noguer, F. (2014), Very fast solution to the pnp problem with algebraic outlier rejection, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 501–508.

Fischler, M. A. and Bolles, R. C. (1981), 'Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography', *Commun. ACM* **24**(6), 381–395.

Forster, C., Carlone, L., Dellaert, F. and Scaramuzza, D. (2017), 'On-manifold preintegration for real-time visual-inertial odometry', *IEEE Trans. Robot.* **33**(1), 1–21.

Forster, C., Pizzoli, M. and Scaramuzza, D. (2014), SVO: fast semi-direct monocular visual odometry, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 15–22.

Forster, C., Zhang, Z., Gassner, M., Werlberger, M. and Scaramuzza, D. (2017), 'SVO: semidirect visual odometry for monocular and multicamera systems', *IEEE Trans. Robot.* **33**(2), 249–265.

Fredriksson, J. and Olsson, C. (2012), Simultaneous multiple rotation averaging using lagrangian duality, *in* 'Asian Conf. Comput. Vis.', Vol. 7726, pp. 245–258.

Fusiello, A., Castellani, U., Ronchetti, L. and Murino, V. (2002), Model acquisition by registration of multiple acoustic range views, *in* 'Eur. Conf. Comput. Vis.', pp. 805–819.

Galvez-López, D. and Tardos, J. D. (2012), 'Bags of binary words for fast place recognition in image sequences', *IEEE Trans. Robot.* **28**(5), 1188–1197.

Gao, X., Luo, J., Li, K. and Xie, Z. (2020), 'Hierarchical RANSAC-based rotation averaging', *IEEE Sign. Process. Letters* **27**, 1874–1878.

Gao, X., Wang, R., Demmel, N. and Cremers, D. (2018), LDSO: Direct sparse odometry with loop closure, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.', pp. 2198–2204.

Gao, X., Zhu, L., Xie, Z., Liu, H. and Shen, S. (2021), 'Incremental rotation averaging', *Int. J. Comput. Vis.* **129**(4), 1202–1216.

Garcia-Salguero, M., Briales, J. and Gonzalez-Jimenez, J. (2021), 'Certifiable relative pose estimation', *Image and Vision Computing* **109**, 104142.

Garcin, F., Faltings, B. and Jurca, R. (2009), 'Aggregating reputation feedback', *Proceedings of the First International Conference on Reputation: Theory and Technology* .

Geiger, A., Lenz, P., Stiller, C. and Urtasun, R. (2013), 'Vision meets robotics: The KITTI dataset', *Int. J. Robot. Res.* .

Geiger, A., Lenz, P. and Urtasun, R. (2012), Are we ready for autonomous driving? the KITTI vision benchmark suite, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 3354–3361.

Gesto Diaz, M., Tombari, F., Rodriguez-Gonzalvez, P. and Gonzalez-Aguilera, D. (2015), 'Analysis and evaluation between the first and the second generation of RGB-D sensors', *IEEE Sensors Journal* **15**(11), 6507–6516.

Gherardi, R., Farenzena, M. and Fusiello, A. (2010), Improving the efficiency of hierarchical structure-and-motion, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1594–1600.

GNU Compiler Collection (GCC) (n.d.), `https://www.gnu.org/software/gcc`.

Gojcic, Z., Zhou, C., Wegner, J. D., Guibas, L. J. and Birdal, T. (2020), Learning multiview 3D point cloud registration, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1756–1766.

Golub, G. H. (1973), 'Some modified matrix eigenvalue problems', *SIAM Review* **15**(2), 318–334.

Govindu, V. M. (2001), Combining two-view constraints for motion estimation, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', Vol. 2, pp. 218–225.

Govindu, V. M. (2004), Lie-algebraic averaging for globally consistent motion estimation, *in* 'IEEE Conf. Comput. Vis. Pattern Recognit.', Vol. 1, pp. 684–691.

Govindu, V. M. (2006), Robustness in motion averaging, *in* 'Asian Conf. Comput. Vis.', pp. 457–466.

Govindu, V. M. and Pooja, A. (2014), 'On averaging multiview relations for 3D scan registration', *IEEE Trans. Image Process.* **23**(3), 1289–1302.

Greene, W. N., Ok, K., Lommel, P. and Roy, N. (2016), Multi-level mapping: Real-time dense monocular SLAM, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 833–840.

Hampel, F. R. (1974), 'The influence curve and its role in robust estimation', *Journal of the American Statistical Association* **69**(346), 383–393.

Handa, A., Whelan, T., McDonald, J. and Davison, A. (2014), A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM, *in* 'IEEE Int. Conf. on Robotics and Automation'.

Harris, C. G. (1987), Determination of ego-motion from matched points, *in* 'Third Alvey Vision Conference', pp. 189–192.

Harris, C. and Pike, J. (1988), '3D positional integration from image sequences', *Image and Vision Computing* **6**(2), 87 – 90.

Hartley, R. I. (1994), Euclidean reconstruction from uncalibrated views, *in* 'Proceedings of the Second Joint European - US Workshop on Applications of Invariance in Computer Vision', pp. 237–256.

Hartley, R. I. (1997), 'In defense of the eight-point algorithm', *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(6), 580–593.

Hartley, R. I., Aftab, K. and Trumpf, J. (2011), L1 rotation averaging using the Weiszfeld algorithm, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 3041–3048.

Hartley, R. I. and Schaffalitzky, F. (2004), L_∞ minimization in geometric reconstruction problems, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 504–509.

Hartley, R. I. and Sturm, P. (1997), 'Triangulation', *Comput. Vis. Image Underst.* **68**(2), 146–157.

Hartley, R. and Kahl, F. (2007), Optimal algorithms in multiview geometry, *in* Y. Yagi, S. B. Kang, I. S. Kweon and H. Zha, eds, 'Asian Conf. Comput. Vis.', pp. 13–34.

Hartley, R., Kahl, F., Olsson, C. and Seo, Y. (2013), 'Verifying global minima for L2 minimization problems in multiple view geometry', *Int. J. Comput. Vis.* **101**(2), 288–304.

Hartley, R., Trumpf, J., Dai, Y. and Li, H. (2013), 'Rotation averaging', *Int. J. Comput. Vis.* **103**, 267–305.

Hartley, R. and Zisserman, A. (2003), *Multiple View Geometry in Computer Vision*, 2 edn, Cambridge University Press.

Hauberg, S., Feragen, A. and Black, M. J. (2014), Grassmann averages for scalable robust PCA, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 3810–3817.

Hedborg, J., Robinson, A. and Felsberg, M. (2014), Robust three-view triangulation done fast, *in* 'IEEE Conf. Comput. Vis. Pattern Recog. Workshops', pp. 152–157.

Heinly, J., Schönberger, J. L., Dunn, E. and Frahm, J.-M. (2015), Reconstructing the world* in six days, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 3287–3295.

Hellerstein, J. M. (2008), 'Quantitative data cleaning for large databases'.

Herrera, D. C., Kim, K., Kannala, J., Pulli, K. and Heikkilä, J. (2014), DT-SLAM: Deferred triangulation for robust SLAM, *in* 'IEEE Int. Conf. 3D Vis.', pp. 609–616.

Holland, P. W. and Welsch, R. E. (1977), 'Robust regression using iteratively reweighted least-squares', *Communications in Statistics - Theory and Methods* **6**(9), 813–827.

Hong, J. H. and Zach, C. (2018), pOSE: Pseudo object space error for initialization-free bundle adjustment, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.'.

Hongdong Li and Hartley, R. (2006), Five-point motion estimation made easy, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', Vol. 1, pp. 630–633.

Horn, B. K. P. (1987), 'Closed-form solution of absolute orientation using unit quaternions', *J. Opt. Soc. Am. A* **4**(4), 629–642.

Huang, X., Liang, Z., Zhou, X., Xie, Y., Guibas, L. J. and Huang, Q. (2019), Learning transformation synchronization, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.'.

Huber, P. J. (1964), 'Robust estimation of a location parameter', *Annals of Mathematical Statistics* **35**(1), 73–101.

Huber, P. J. (1981), *Robust Statistics*, Wiley Series in Probability and Statistics, Wiley.

Humbert, M., Gey, N., Muller, J. and Esling, C. (1996), 'Determination of a Mean Orientation from a Cloud of Orientations. Application to Electron Back-Scattering Pattern Measurements', *Journal of Applied Crystallography* **29**(6), 662–666.

Illes, M. and Boué, M. (2013), 'Robust estimation for area of origin in bloodstain pattern analysis via directional analysis', *Forensic Science International* **226**(1), 223–229.

Irani, M. and Anandan, P. (1999), About direct methods, *in* 'Proc. Workshop Vision Algorithms', pp. 267–277.

Kahl, F., Agarwal, S., Chandraker, M. K., Kriegman, D. and Belongie, S. (2008), 'Practical global optimization for multiview geometry', *Int. J. Comput. Vis.* **79**(3), 271–284.

Kahl, F. and Hartley, R. (2008), 'Multiple-view geometry under the L_∞-norm', *IEEE Trans. Pattern Anal. Mach. Intell.* **30**(9), 1603–1617.

Kahl, F. and Henrion, D. (2007), 'Globally optimal estimates for geometric reconstruction problems', *Int. J. Comput. Vis.* **74**(1), 3–15.

Kanatani, K., Sugaya, Y. and Niitsuma, H. (2008), Triangulation from two views revisited: Hartley-Sturm vs. optimal correction, *in* 'Brit. Mach. Vis. Conf.', pp. 173–182.

Kanazawa, Y. and Kanatani, K. (1995), 'Reliability of 3-D reconstruction by stereo vision', *IECE Trans. Inf. & Syst.* **E78-D**(10), 1301–1306.

Kang, L., Wu, L. and Yang, Y.-H. (2014), 'Robust multi-view L_2 triangulation via optimal inlier selection and 3D structure refinement', *Pattern Recognition* **47**(9), 2974 – 2992.

Ke, Q. and Kanade, T. (2003), *Robust subspace computation using L1 norm*, School of Computer Science, Carnegie Mellon University Pittsburgh, PA.

Ke, Q. and Kanade, T. (2007), 'Quasiconvex optimization for robust geometric reconstruction', *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(10), 1834–1847.

Khatoonabadi, S. H. and Bajic, I. V. (2013), 'Video object tracking in the compressed domain using spatio-temporal Markov random fields', *IEEE Trans. Image Process.* **22**(1), 300–313.

Kim, P., Lee, H. and Kim, H. J. (2018), 'Autonomous flight with robust visual odometry under dynamic lighting conditions', *Autonomous Robots* .

Kingma, D. P. and Ba, J. L. (2015), ADAM: A method for stochastic optimization, *in* 'Int. Conf. Learn. Represent.'.

Klamkin, M. S. (1970), 'Vector proofs in solid geometry', *The American Mathematical Monthly* **77**(10), 1051–1065.

Klein, G. and Murray, D. (2007), Parallel tracking and mapping for small AR workspaces, *in* 'Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)'.

Kneip, L. and Lynen, S. (2013), Direct optimization of frame-to-frame rotation, *in* 'IEEE Int. Conf. on Comput. Vis.', pp. 2352–2359.

Kneip, L., Siegwart, R. and Pollefeys, M. (2012), Finding the exact rotation between two images independently of the translation, *in* 'Eur. Conf. Comput. Vis.', pp. 696–709.

Konolige, K. (2010), Sparse sparse bundle adjustment, *in* 'Brit. Mach. Vis. Conf.', pp. 102.1–102.11.

Kotowski, K., Stapor, K. and Leski, J. (2019), 'Improved robust weighted averaging for event-related potentials in EEG', *Biocybernetics and Biomedical Engineering* **39**(4), 1036–1046.

Krombach, N., Droeschel, D. and Behnke, S. (2016), Combining feature-based and direct methods for semi-dense real-time stereo visual odometry, *in* 'Int. Conf. on Intelligent Autonomous Systems', pp. 855–868.

Küemmerle, R., Steder, B., Dornhege, C., Ruhnke, M., Grisetti, G., Stachniss, C. and Kleiner, A. (2009), 'On measuring the accuracy of SLAM algorithms', *Autonomous Robots* **27**(4), 387–407.

Kukelova, Z., Bujnak, M. and Pajdla, T. (2008), Polynomial eigenvalue solutions to the 5-pt and 6-pt relative pose problems, *in* 'Brit. Mach. Vis. Conf.', pp. 56.1–56.10.

Kukelova, Z., Pajdla, T. and Bujnak, M. (2013), Fast and stable algebraic solution to L_2 three-view triangulation, *in* 'IEEE Int. Conf. 3D Vis.', pp. 326–333.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K. and Burgard, W. (2011), g2o: A general framework for graph optimization, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 3607–3613.

Lam, Q. M. and Crassidis, J. L. (2007), Precision attitude determination using a multiple model adaptive estimation scheme, *in* 'IEEE Aerospace Conference', pp. 1–20.

Lee, S. H. and Civera, J. (2019*a*), Closed-form optimal two-view triangulation based on angular errors, *in* 'IEEE Int. Conf. Comput. Vis.'.

Lee, S. H. and Civera, J. (2019*b*), 'Loosely-coupled semi-direct monocular SLAM', *IEEE Robot. Autom. Lett.* **4**(2), 399–406.

Lee, S. H. and Civera, J. (2019*c*), Triangulation: Why optimize?, *in* 'Brit. Mach. Vis. Conf.'.

Lee, S. H. and Civera, J. (2020*a*), 'Geometric interpretations of the normalized epipolar error', *CoRR* **abs/2008.01254**.

Lee, S. H. and Civera, J. (2020*b*), 'Robust single rotation averaging', *CoRR* **abs/2004.00732**.

Lee, S. H. and Civera, J. (2020*c*), 'Robust uncertainty-aware multiview triangulation', *CoRR* **abs/2008.01258**.

Lee, S. H. and Civera, J. (2021), Rotation-only bundle adjustment, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 424–433.

Lee, S. H. and Civera, J. (2022), HARA: A hierarchical approach for robust rotation averaging, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 15777–15786.

Lee, S. H. and de Croon, G. (2018), 'Stability-based scale estimation for monocular SLAM', *IEEE Robot. Automat. Lett.* **3**(2), 780–787.

Leonowicz, Z., Karvanen, J. and Shishkin, S. L. (2005), 'Trimmed estimators for robust averaging of event-related potentials', *Journal of neuroscience methods* **142**(1), 17–26.

Leski, J. M. (2002), 'Robust weighted averaging [of biomedical signals]', *IEEE Trans. Biomed. Eng.* **49**(8), 796–804.

Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R. and Furgale, P. (2015), 'Keyframe-based visual-inertial odometry using nonlinear optimization', *Int. J. Robot. Res.* **34**(3), 314–334.

Lewis, T., Owens, R. and Baddeley, A. (1999), 'Averaging feature maps', *Pattern Recognition* **32**(9), 1615–1630.

Li, H. (2007), 'A practical algorithm for L_∞ triangulation with outliers', *IEEE Conf. Comput. Vis. Pattern Recog.* pp. 1–8.

Li, S., Zhang, T., Gao, X., Wang, D. and Xian, Y. (2019), 'Semi-direct monocular visual and visual-inertial SLAM with loop closure detection', *Robotics and Autonomous Systems* **112**, 201 – 210.

Li, X., Fei, S. and Zhang, T. (2009), 'Median MSD-based method for face recognition', *Neurocomputing* **72**(16), 3930–3934.

Lindstrom, P. (2010), Triangulation made easy, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1554–1561.

Lines, L. (1935), *Solid Geometry*, Macmillan and co., limited.

Longuet-Higgins, H. C. (1981), 'A computer algorithm for reconstructing a scene from two projections', *Nature* **293**(5828), 133–135.

Lopuhaa, H. P. and Rousseeuw, P. J. (1991), 'Breakdown points of affine equivariant estimators of multivariate location and covariance matrices', *The Annals of Statistics* **19**(1), 229 – 248.

Loshchilov, I. and Hutter, F. (2019), Decoupled weight decay regularization, *in* 'Int. Conf. Learn. Represent.'.

Lourakis, M. I. A. and Argyros, A. A. (2009), 'SBA: A software package for generic sparse bundle adjustment', *ACM Trans. Math. Softw.* **36**(1).

Lowe, D. G. (2004), 'Distinctive image features from scale-invariant keypoints', *Int. J. Comput. Vis.* **60**(2), 91–110.

Lu, F. and Hartley, R. (2007), A fast optimal algorithm for L_2 triangulation, *in* 'Asian Conf. Comput. Vis.', pp. 279–288.

Luo, D., Zhuang, Y. and Wang, S. (2022), 'Hybrid sparse monocular visual odometry with online photometric calibration', *Int. J. Robot. Res.* **41**(11-12), 993–1021.

Luo, H., Pape, C. and Reithmeier, E. (2021), 'Hybrid monocular SLAM using double window optimization', *IEEE Robot. Automat. Lett.* **6**(3), 4899–4906.

Luong, Q. and Faugeras, O. D. (1996), 'The fundamental matrix: Theory, algorithms, and stability analysis', *Int. J. Comput. Vis.* **17**(1), 43–75.

Lynen, S., Zeisl, B., Aiger, D., Bosse, M., Hesch, J., Pollefeys, M., Siegwart, R. and Sattler, T. (2020), 'Large-scale, real-time visual–inertial localization revisited', *Int. J. Robot. Res.* **39**(9), 1061–1084.

Markley, L., Cheng, Y., Crassidis, J. and Oshman, Y. (2007), 'Averaging quaternions', *Journal of Guidance, Control, and Dynamics* **30**, 1193–1196.

Martinec, D. and Pajdla, T. (2007), Robust rotation and translation estimation in multiview reconstruction, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1–8.

Merchant, N., Farcas, A. and Powell, C. (2018), Acoustic metric specification, Technical report, Centre for Environment, Fisheries & Aquaculture Science (Cefas), UK.

Meyer, G. P. (2021), An alternative probabilistic interpretation of the Huber loss, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 5261–5269.

Mičušík, B. and Pajdla, T. (2006), 'Structure from motion with wide circular field of view cameras', *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(7), 1135–1149.

Morawiec, A. (1998), 'A note on mean orientation', *Journal of Applied Crystallography* **31**(5), 818–819.

Moré, J. J. (1978), The Levenberg-Marquardt algorithm: Implementation and theory, *in* 'Numerical Analysis', pp. 105–116.

Moreira, G., Marques, M. and Costeira, J. a. P. (2021*a*), Rotation averaging in a split second: A primal-dual method and a closed-form for cycle graphs, *in* 'Int. Conf. Comput. Vis.', pp. 5452–5460.

Moreira, G., Marques, M. and Costeira, J. P. (2021*b*), Fast pose graph optimization via Krylov-Schur and Cholesky factorization, *in* 'Winter Conf. Appl. Comput. Vis.', pp. 1897–1905.

Moulon, P., Monasse, P. and Marlet, R. (2013), Global fusion of relative motions for robust, accurate and scalable structure from motion, *in* 'IEEE Int. Conf. Comput. Vis.', pp. 3248–3255.

Moulon, P., Monasse, P., Perrot, R. and Marlet, R. (2017), OpenMVG: Open multiple view geometry, *in* 'Reproducible Research in Pattern Recognition', pp. 60–74.

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. and Sayd, P. (2009), 'Generic and real-time structure from motion using local bundle adjustment', *Image Vision Comput.* **27**(8), 1178–1193.

Mur-Artal, R., Montiel, J. M. M. and Tardós, J. D. (2015), 'ORB-SLAM: A versatile and accurate monocular SLAM system', *IEEE Trans. Robot.* **31**(5), 1147–1163.

Mur-Artal, R. and Tardós, J. D. (2014), Fast relocalisation and loop closing in keyframe-based SLAM, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 846–853.

Mur-Artal, R. and Tardos, J. D. (2015), Probabilistic semi-dense mapping from highly accurate feature-based monocular SLAM, *in* 'Proc. Robotics: Science and Systems'.

Mur-Artal, R. and Tardós, J. D. (2017), 'ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras', *IEEE Trans. Robot.* **33**(5), 1255–1262.

Mur-Artal, R. and Tardós, J. D. (2017), 'Visual-inertial monocular SLAM with map reuse', *IEEE Robot. Automat. Lett.* **2**(2), 796–803.

Mykland, P. A. and Zhang, L. (2016), 'Between data cleaning and inference: Pre-averaging and robust estimators of the efficient price', *Journal of Econometrics* **194**(2), 242–262.

Newcombe, R. A., Lovegrove, S. and Davison, A. J. (2011), DTAM: dense tracking and mapping in real-time, *in* 'IEEE Int. Conf. Comput. Vis.', pp. 2320–2327.

Níster, D. (2001), Automatic Dense Reconstruction from Uncalibrated Video Sequences, PhD thesis, KTH.

Nistér, D. (2004), 'An efficient solution to the five-point relative pose problem', *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(6), 756—-777.

Nistér, D., Naroditsky, O. and Bergen, J. (2006), 'Visual odometry for ground vehicle applications', *J. Field Robot.* **23**(1), 3–20.

Nousias, S., Lourakis, M. and Bergeles, C. (2019), Large-scale, metric structure from motion for unordered light fields, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.'.

Oliensis, J. (2002), 'Exact two-image structure from motion', *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(12), 1618–1633.

Olsson, C., Enqvist, O. and Kahl, F. (2011), Stable structure from motion for unordered image collections, *in* 'Image Analysis', pp. 524–535.

Olsson, C., Eriksson, A. and Hartley, R. (2010), Outlier removal using duality, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 1450–1457.

Ondrúška, P., Kohli, P. and Izadi, S. (2015), 'MobileFusion: Real-time volumetric surface reconstruction and dense tracking on mobile phones', *IEEE Transactions on Visualization and Computer Graphics* **21**(11), 1251–1258.

Oxford Multiview Datasets (n.d.), `https://www.robots.ox.ac.uk/~vgg/data/mview/`.

Ozyesil, O. and Singer, A. (2015), Robust camera location estimation by convex programming, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 2674–2683.

Ozyesil, O., Voroninski, V., Basri, R. and Singer, A. (2017), 'A survey of structure from motion.', *Acta Numerica* **26**, 305–364.

Pagani, A. and Stricker, D. (2011), Structure from motion using full spherical panoramic cameras, *in* 'IEEE Int. Conf. on Computer Vision Workshops', pp. 375–382.

Parra, A., Chng, S.-F., Chin, T.-J., Eriksson, A. and Reid, I. (2021), Rotation coordinate descent for fast globally optimal rotation averaging, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 4298–4307.

Peng, L., Kümmerle, C. and Vidal, R. (2022), Global linear and local superlinear convergence of IRLS for non-smooth robust regression, *in* 'Advances in Neural Information Processing Systems'.

Pirchheim, C., Schmalstieg, D. and Reitmayr, G. (2013), Handling pure camera rotation in keyframe-based SLAM, *in* 'IEEE International Symposium on Mixed and Augmented Reality', pp. 229–238.

Pizzoli, M., Forster, C. and Scaramuzza, D. (2014), REMODE: Probabilistic, monocular dense reconstruction in real time, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 2609–2616.

Platinsky, L., Davison, A. J. and Leutenegger, S. (2017), Monocular visual odometry: Sparse joint optimisation or dense alternation?, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 5126–5133.

Purkait, P., Chin, T. and Reid, I. (2020), Neurora: Neural robust rotation averaging, *in* 'Eur. Conf. Comput. Vis.', pp. 137–154.

Qin, T., Li, P. and Shen, S. (2018), 'VINS-Mono: A robust and versatile monocular visual-inertial state estimator', *IEEE Trans. Robot.* **34**(4), 1004–1020.

Raguram, R., Chum, O., Pollefeys, M., Matas, J. and Frahm, J. (2013), 'Usac: A universal framework for random sample consensus', *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 2022–2038.

Ramalingam, S., Lodha, S. K. and Sturm, P. (2006), 'A generic structure-from-motion framework', *Comput. Vis. Image Underst.* **103**(3), 218 – 228.

Recker, S., Hess-Flores, M. and Joy, K. I. (2013), Statistical angular error-based triangulation for efficient and accurate multi-view scene reconstruction, *in* 'IEEE Workshop on Applications of Computer Vision', pp. 68–75.

Reddi, S. J., Kale, S. and Kumar, S. (2018), On the convergence of Adam and beyond, *in* 'Int. Conf. Learn. Represent.'.

Rock, N. M. S. (1988), 'Summary statistics in geochemistry: A study of the performance of robust estimates', *Mathematical Geology* **20**(3), 243–275.

Rock, N., Webb, J., McNaughton, N. and Bell, G. (1987), 'Nonparametric estimation of averages and errors for small data-sets in isotope geoscience: a proposal', *Chemical Geology: Isotope Geoscience section* **66**(1), 163–177.

Rodríguez, A. L., López-de-Teruel, P. E. and Ruiz, A. (2011), Reduced epipolar cost for accelerated incremental SfM, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 3097–3104.

Rosen, D. M., Carlone, L., Bandeira, A. S. and Leonard, J. J. (2019), 'SE-Sync: A certifiably correct algorithm for synchronization over the special Euclidean group', *Int. J. Robot. Res.* **38**(2–3), 95–125.

Rousseeuw, P. J. (1984), 'Least median of squares regression', *Journal of the American Statistical Association* **79**(388), 871–880.

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. (2011), ORB: An efficient alternative to SIFT or SURF, *in* 'IEEE Int. Conf. Comput. Vis.', pp. 2564–2571.

S. Courvoisier, D. and Renaud, O. (2010), 'Robust analysis of the central tendency, simple and multiple regression and ANOVA: a step by step tutorial.', *International Journal of Psychological Research* **3**(1), 78–87.

Sattler, T., Leibe, B. and Kobbelt, L. (2011), Fast image-based localization using direct 2D-to-3D matching, *in* 'Int. Conf. Comput. Vis.', pp. 667–674.

Sattler, T., Maddern, W., Toft, C., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., Kahl, F. and Pajdla, T. (2018), Benchmarking 6DOF outdoor visual localization in changing conditions, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.'.

Scaramuzza, D. and Fraundorfer, F. (2011), 'Visual odometry [Tutorial]', *IEEE robotics & automation magazine* **18**(4), 80–92.

Schönberger, J. L. and Frahm, J.-M. (2016), Structure-from-motion revisited, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 4104–4113.

Shapiro, S. S. and Wilk, M. B. (1965), 'An analysis of variance test for normality (complete samples)', *Biometrika* **52**(3/4), 591–611.

Sharf, I., Wolf, A. and Rubin, M. (2010), 'Arithmetic and geometric solutions for average rigid-body rotation', *Mechanism and Machine Theory* **45**(9), 1239 – 1251.

Sharp, G. C., Lee, S. W. and Wehe, D. K. (2002), Multiview registration of 3D scenes by minimizing error between coordinate frames, *in* 'Eur. Conf. Comput. Vis.', pp. 587–597.

Shen, T., Zhu, S., Fang, T., Zhang, R. and Quan, L. (2016), Graph-based consistent matching for structure-from-motion, *in* 'Eur. Conf. Comput. Vis.', pp. 139–155.

Shi, Y. and Lerman, G. (2020), Message passing least squares framework and its application to rotation synchronization, *in* 'Int. Conf. Mach. Learning', Vol. 119, pp. 8796–8806.

Sim, K. and Hartley, R. (2006), Removing outliers using the L_∞ norm, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', Vol. 1, pp. 485–494.

Snavely, N., Seitz, S. M. and Szeliski, R. (2006), 'Photo tourism: exploring photo collections in 3D', *ACM Trans. Graph.* **25**(3), 835–846.

Solà, J., Deray, J. and Atchuthan, D. (2018), 'A micro Lie theory for state estimation in robotics', *CoRR* **abs/1812.01537**.

Spetsakis, M. and Aloimonos, Y. (1992), 'Optimal visual motion estimation: a note', *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(9), 959–964.

Stewart, C. (1995), 'MINPRAN: a new robust estimator for computer vision', *IEEE Trans. Pattern Anal. Mach. Intell.* **17**(10), 925–938.

Stewart, C. V. (1999), 'Robust parameter estimation in computer vision', *SIAM Rev.* **41**(3), 513—-537.

Stewénius, H., Schaffalitzky, F. and Nister, D. (2005), How hard is 3-view triangulation really?, *in* 'IEEE Int. Conf. Comput. Vis.', Vol. 1, pp. 686–693.

Strasdat, H., Davison, A. J., Montiel, J. M. M. and Konolige, K. (2011), Double window optimisation for constant time visual SLAM, *in* 'IEEE Int. Conf. Comput. Vis.', pp. 2352–2359.

Strasdat, H., Montiel, J. M. M. and Davison, A. J. (2010*a*), Real-time monocular slam: Why filter?, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 2657–2664.

Strasdat, H., Montiel, J. M. M. and Davison, A. J. (2010*b*), Scale drift-aware large scale monocular SLAM, *in* 'Proc. Robotics: Science and Systems'.

Sturm, J., Engelhard, N., Endres, F., Burgard, W. and Cremers, D. (2012), A benchmark for the evaluation of rgb-d slam systems, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.'.

Sun, L. (2022), 'Practical, fast and robust point cloud registration for scene stitching and object localization', *IEEE Access* **10**, 3962–3978.

Svärm, L., Enqvist, O., Oskarsson, M. and Kahl, F. (2014), Accurate localization and pose estimation for large 3D models, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 532–539.

Tang, Y. and Feng, J. (2015), 'Hierarchical multiview rigid registration', *Computer Graphics Forum* **34**.

Teed, Z. and Deng, J. (2021), DROID-SLAM: Deep visual slam for monocular, stereo, and RGB-D cameras, *in* 'Advances in Neural Information Processing Systems', Vol. 34, pp. 16558–16569.

Thode, H. (2002), *Testing For Normality*, Statistics, textbooks and monographs, CRC Press.

Tippetts, B., Lee, D. J., Lillywhite, K. and Archibald, J. (2016), 'Review of stereo vision algorithms and their suitability for resource-limited systems', *Journal of Real-Time Image Processing* **11**(1), 5–25.

Toft, C., Maddern, W., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., Pajdla, T., Kahl, F. and Sattler, T. (2022), 'Long-term visual localization revisited', *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(4), 2074–2088.

Torr, P. H. S. and Zisserman, A. (1998), Robust computation and parameterization of multiple view relations, *in* 'IEEE Int. Conf. Comput. Vis.'.

Torr, P. and Murray, D. (1997), 'The development and comparison of robust methods for estimating the fundamental matrix', *Int. J. Comput. Vis.* **24**(3), 271–300.

Torr, P., Zisserman, A. and Maybank, S. (1998), 'Robust detection of degenerate configurations while estimating the fundamental matrix', *Comput. Vis. Image Underst.* **71**(3), 312 – 333.

Triggs, B., McLauchlan, P., Hartley, R. and Fitzgibbon, A. (2000), Bundle adjustment – a modern synthesis, *in* 'Vision Algorithms: Theory and Practice', pp. 298–375.

Tron, R., Zhou, X. and Daniilidis, K. (2016), A survey on rotation optimization in structure from motion, *in* 'IEEE Conf. Comput. Vis. Pattern Recog. Workshops', pp. 1032–1040.

Vaish, V., Levoy, M., Szeliski, R., Zitnick, C. and Kang, S. B. (2006), Reconstructing occluded surfaces using synthetic apertures: Stereo, focus and robust measures, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', Vol. 2, pp. 2331–2338.

Von Stumberg, L., Usenko, V. and Cremers, D. (2018), Direct sparse visual-inertial odometry using dynamic marginalization, *in* 'IEEE Intl. Conf. Robot. Automat.', pp. 2510–2517.

Wang, L. and Singer, A. (2013), 'Exact and stable recovery of rotations for robust synchronization', *Inf. Inference J. IMA* **2**(2), 145–193.

Wang, W., Zhu, D., Wang, X., Hu, Y., Qiu, Y., Wang, C., Hu, Y., Kapoor, A. and Scherer, S. (2020), TartanAir: A dataset to push the limits of visual SLAM, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.'.

Weiszfeld, E. (1937), 'Sur le point pour lequel la somme des distances de n points donnés est minimum', *Tohoku Mathematical Journal* **43**, 355–386.

Weiszfeld, E. and Plastria, F. (2009), 'On the point for which the sum of the distances to n given points is minimum', *Annals of Operations Research* **167**(1), 7–41.

Wilson, K. and Bindel, D. (2020), On the distribution of minima in intrinsic-metric rotation averaging, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 6030–6038.

Wilson, K., Bindel, D. and Snavely, N. (2016), When is rotations averaging hard?, *in* 'Eur. Conf. Comput. Vis.', pp. 255–270.

Wilson, K. and Snavely, N. (2014), Robust global translations with 1DSfM, *in* 'Eur. Conf. Comput. Vis.', pp. 61–75.

Wolff, K., Kim, C., Zimmer, H., Schroers, C., Botsch, M., Sorkine-Hornung, O. and Sorkine-Hornung, A. (2016), Point cloud noise and outlier removal for image-based 3D reconstruction, *in* 'Int. Conf. 3D Vis.'.

Woodford, O. J. and Rosten, E. (2020), Large scale photometric bundle adjustment, *in* 'Brit. Mach. Vis. Conf.', pp. 1–12.

Yang, J., Li, H. and Jia, Y. (2014), Optimal essential matrix estimation via inlier-set maximization, *in* 'Eur. Conf. Comput. Vis.', pp. 111–126.

Yang, K., Fang, W., Zhao, Y. and Deng, N. (2019), 'Iteratively reweighted midpoint method for fast multiple view triangulation', *IEEE Robot. Autom. Lett.* **4**(2), 708–715.

Yang, L., Li, H., Rahim, J. A., Cui, Z. and Tan, P. (2021), End-to-end rotation averaging with multi-source propagation, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 11774–11783.

Yang, N., Wang, R., Gao, X. and Cremers, D. (2018), 'Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect', *IEEE Robot. Automat. Lett.* **3**(4), 2878–2885.

Younes, G., Asmar, D., Shammas, E. and Zelek, J. (2017), 'Keyframe-based monocular SLAM: design, survey, and future directions', *Robotics and Autonomous Systems* **98**, 67–88.

Younes, G., Asmar, D. and Zelek, J. (2019), A unified formulation for visual odometry, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.', pp. 6237–6244.

Yu, F. and Gallup, D. (2014), 3D reconstruction from accidental motion, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 3986–3993.

Zach, C. (2014), Robust bundle adjustment revisited, *in* 'Eur. Conf. Comput. Vis.', pp. 772–787.

Zach, C. and Bourmaud, G. (2018), Descending, lifting or smoothing: Secrets of robust cost optimization, *in* 'Eur. Conf. Comput. Vis.'.

Zach, C., Klopschitz, M. and Pollefeys, M. (2010), Disambiguating visual relations using loop constraints, *in* 'IEEE Conf. Comput. Vis. Pattern Recognit.', pp. 1426–1433.

Zhang, C. and Zhang, S. (1996), 'A robust-symmetric mean: A new way of mean calculation for environmental data', *GeoJournal* **40**, 209–212.

Zhang, J. and Singh, S. (2014), LOAM: Lidar odometry and mapping in real-time, *in* 'Proceedings of Robotics: Science and Systems'.

Zhang, Z. (1997), 'Parameter estimation techniques: a tutorial with application to conic fitting', *Image and Vision Computing* **15**(1), 59–76.

Zhang, Z. (1998), 'Determining the epipolar geometry and its uncertainty: A review', *Int. J. Comput. Vis.* **27**(2), 161–195.

Zhang, Z. and Scaramuzza, D. (2018*a*), A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry, *in* 'IEEE/RSJ Int. Conf. Intell. Robots Syst.', pp. 7244–7251.

Zhang, Z. and Scaramuzza, D. (2018*b*), A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry, *in* 'IEEE Int. Conf. on Robotics and Automation', pp. 1–5.

Zhao, J. (2022), 'An efficient solution to non-minimal case essential matrix estimation', *IEEE Trans. Pattern Anal. Mach. Intell.* **44**(4), 1777–1792.

Zhou, H., Ummenhofer, B. and Brox, T. (2018), DeepTam: Deep tracking and mapping, *in* 'Eur. Conf. Comput. Vis.', pp. 851–868.

Zhu, S., Zhang, R., Zhou, L., Shen, T., Fang, T., Tan, P. and Quan, L. (2018), Very large-scale global SfM by distributed motion averaging, *in* 'IEEE Conf. Comput. Vis. Pattern Recog.', pp. 4568–4577.