SLAM in Robotics and Autonumous Driving

Xiang Gao

Last update: March 18, 2024

To my beloved Lilian and Shenghan

Preface

About this book

Well, autonomous driving is a such cool stuff, isn't it?

You've probably seen scenes of self-driving cars in science fiction movies. In these vehicles, the steering wheel turns by itself, the throttle and brakes are controlled automatically, freeing people from the monotony of driving so they can enjoy their time more freely. In fact, some Level 2 vehicles have already achieved partial self-driving capabilities in simple road conditions. They can help drivers keep the vehicle centered in the lane or maintain a certain distance from the preceding vehicle. These systems are called **Advanced Driver Assistance Systems (ADAS)**. And for more advanced self-driving systems (Level 4), computers can fully take over, not only assisting drivers but also controlling buses, delivery vehicles, robots, robotic dogs, and even bicycles, enabling many functionalities we never imagined. Over time, these sci-fi-like scenes have gradually become reality. Self-driving jobs have also emerged as a new industry, attracting young talents from all sectors of society. It's truly an exciting development!

The field of autonomous driving encompasses many emerging technologies, with Simultaneous Localization and Mapping (SLAM) being a key focus. Since completing my PhD, I have been involved in the research and development of SLAM in the autonomous driving industry. It's an intriguing area because whether it's large passenger vehicles, small low-speed vehicles, or even sweepers, SLAM is certainly a fundamental technology. Most of the automation features we see are actually embedded within the vehicle's map data. For instance, the map might instruct the vehicle to turn left at the upcoming intersection and merge into the right lane of the opposite road; or, for cleaning a plaza ahead, the vehicle should drive along the right boundary in circles while avoiding the flower bed area in the middle. To achieve these functionalities, we need to integrate various types of data such as GPS, inertial navigation, laser point clouds, visual images, etc., to construct maps and then perform localization on these maps.

If you work in this area, you'll find that it brings together people with diverse backgrounds. People working on inertial navigation are familiar with strapdown inertial navigation systems. They enjoy writing matrix and vector computation programs on a small embedded CPU, often scratching their heads over errors of one or two points aboue accuracy. Those involved in processing laser point clouds engage in meticulous map reconstruction, displaying beautiful three-dimensional point clouds on the screen. Meanwhile, those working on vision spend their days manipulating images on the imaging plane, producing impressive results but not focusing much on accuracy issues. Well, I mean, accuracy is of course an important issue, but

the accuracy on a two-dimensional imaging plane is not the same as the accuracy of three-dimensional world points or localization accuracy. Cameras usually don't have fixed accuracy metrics like other sensors. They can either focus on nearby objects for local high precision or distant objects for a broad field of view, just like the difference between a microscope and a telescope. Thanks to the collaboration among these individuals and effective communication from management, vehicles are able to navigate the roads stably. However, most of the time, we aren't entirely clear about what other people are doing or how they're doing it. This is one of the motivations for me to write this book.

I hope to introduce readers to the localization and mapping technologies related to autonomous driving and robotics in this book, which include the sensors we use in our daily lives. Although there is currently no unified opinion on what constitutes a vehicle or a robot, they can even be seen as wheeled smartphones. However, these intelligent machines all use similar sensors, and the underlying theories are basically the same. I hope that through this book, researchers in this field can enhance mutual understanding, or colleagues and students outside the field can understand what work we are doing. I believe many people will be interested in these technologies.

Content of This Book

This book introduces SLAM-related technologies used in autonomous driving and robotics. The SLAM discussed here is quite general. We will cover sensors and processing methods related to localization and mapping. A typical autonomous vehicle will include various sensors such as IMU, wheel encoders, vehicle speed sensors, and multi-line laser sensors, so our localization and mapping will also involve methods for these sensors. Therefore, readers will encounter topics like basic algorithms for inertial navigation, filters represented by Kalman filtering, laser point cloud matching methods, trajectory fusion algorithms, and more in this book. Overall, we will introduce these contents in the following order:

1. The first part covers **Basic Mathematical Knowledge**. We will start with basic coordinate system definitions, rotational geometry, and quickly introduce some mathematical background knowledge used in this book. Since most of this background knowledge can be found in other books and materials, we will only provide a brief introduction. Chapter 1 of Part 1 provides an overview of autonomous driving, Chapter 2 introduces basic geometry and kinematics, Chapter 3 covers the error Kalman filter used for integrated navigation, and Chapter 4 introduces pre-integration systems and optimization methods. Readers don't need to worry about the specialized terminology mentioned here; we will delve into them in detail in specific chapters.

2. The second part is about **Laser Localization and Mapping**. This section introduces 2D and 3D laser localization technologies, with the former mainly used in robots represented by sweepers, and the latter being one of the foundational technologies for autonomous driving vehicles. We will detail representative techniques for processing laser point clouds and demonstrate their applications through code implementation. Chapter 5 of Part 2 introduces basic point cloud processing algorithms (nearest neighbor structures, KD trees, etc.), Chapter 6 discusses 2D laser localization and mapping, and Chapter 7 covers 3D laser localization and mapping.

3. The third part focuses on **Application**. We will discuss the process of building high-precision point cloud maps for autonomous driving and how to use point cloud maps for real-time localization. Chapter 8 introduces tightly coupled laser-inertial odometry methods, Chapter 9 presents offline point cloud mapping systems, and Chapter 10 introduces online fusion localization systems.

Similar to my previous book (`https://github.com/gaoxiang12/slambook-en`), this book emphasizes the unity of theory and practice and pays close attention to the implementation of principles in code. All algorithms mentioned in this book will have code implementations provided in the corresponding chapters. Readers will work with us to implement those important and foundational algorithmic structures in this field from scratch, using modern programming techniques and fully exploiting parallelization principles to make our algorithms run smoother than classical implementations. Consequently, we will not limit ourselves to specific implementations of open-source code. For example, we will avoid discussing what LOAM does from line X to line Y or which library Cartographer references in a particular cpp file. We will refrain from discussing engineering details like thread pools or parameter file formats. Yes, such discussions can be too detailed, and everyone's implementation may differ. We will strive to retain only the core algorithmic code, allowing readers to debug and understand the entire process themselves.

In terms of style, I will continue to use my familiar writing style. Readers who are familiar with me should quickly adapt, while those who are not should not find it overly challenging. I hope my writing is as clear and straightforward as a conversation. Throughout the introduction of content, I hope the reading process reflects a complete train of thought, rather than simply compiling information together. Although this writing style may result in some verbosity, I believe it is beneficial.

The majority of the key contents in this book will be accompanied by corresponding implementation code. This is one of the major features of this book. I believe that for a comprehensive book, providing code that demonstrates the concepts is always a wise choice. However, despite our efforts to streamline the code, the code section of this book is still much larger than my previous book.

Here is our code repository:

`https://github.com/gaoxiang12/slam_in_autonomous_driving`

Please checkout the "en" branch if you only want to read English comments.

All code and data for this book are open-source and freely accessible to readers. The PDF file of this book will be continuously updated within the code repository. We use C++ as the primary programming language. Please don't ask me why I didn't use more concise languages like Python or Matlab because the programs running in actual vehicles or robots are still primarily C++ programs, and I don't want our experiments to deviate too far from industrial applications. Please note that the code, errata, and other files for this book will be updated on GitHub first, while the published version of the book may have a certain lag time due to the printing schedule of the publishing house. If readers find any discrepancies between the content in the book and the code repository, please consider the implementation in the code repository as authoritative.

We welcome readers to ask questions or answer questions from other readers in the code repository of this book. We encourage readers to communicate in English to facilitate sharing your experiences with international friends.

How to Use This Book

The content of this book follows a process of gradually deepening complexity, but even basic topics like those in Chapter 2 require some groundwork. Personally, I hope this book serves as a sequel to my previous book [1] (`https://github.com/gaoxiang12/slambook-en`). You should at least read the first 6 chapters of that book to familiarize yourself with some basic mathematical principles and the basic usage of optimization libraries. However, if readers have not read that book, you should at least have knowledge in the following areas:

- Basic undergraduate-level mathematics such as calculus, linear algebra, and probability theory.
- Mathematics at the graduate level: optimization, matrix theory, a small amount of knowledge about Lie groups and Lie algebras.
- Computer science: Linux system operations, C++ programming language.

If readers find certain parts of this book difficult to understand, they can refer to corresponding reference books for supplementary learning. Overall, this book will be slightly more challenging and the pace of introduction will be somewhat faster.

The code for this book is organized by chapter. For example, the code for Chapter 3 will be located in src/ch3. The code for each chapter will be compiled into separate library files and executable files. Additionally, shared code will be placed in src/common (such as some common structures, message definitions, UI, etc.). There is a certain degree of dependency between the code of different chapters, with later chapters reusing the results of earlier chapters. The code for this book needs to be compiled using ROS, but the actual running and testing processes do not require the use of ROS mechanisms; only ROS data packages are used for storage. Readers only need to understand the installation process of ROS and do not need to familiarize themselves with the details of ROS in advance.

Notations

The mathematical symbols in this book follow the international standard. In general, scalars are expressed in slanted font, such as a; matrices and vectors are represented in bold font, such as $\mathbf{A}$; special sets are denoted in hollow sans-serif font, such as $\mathbb{R}$; and Lie algebra-related sets are expressed in Gothic font, such as $\mathfrak{so}(3)$. We aim to maintain consistency throughout the book in terms of symbols, with additional explanations provided where ambiguity may arise.

Relationship with Other Books and Papers

Autonomous driving localization technology involves many active research fields. For example, Professor Barfoot's "State Estimation for Robotics" [2] focuses on introducing state estimation theory. Its Chinese translation was also translated by our team. On the one hand, it provides a comparative introduction to the differences and similarities between traditional filtering theory and modern optimization theory, and on the other hand, it provides an excellent introduction to Lie groups and Lie algebra for engineering readers. This book will partially use some conclusions from the book on state estimation, mainly the part related to Lie groups and Lie algebras, to support some of our formula derivations.

Professor Ma's "An invitation to 3-d vision: from images to geometric models" [3] is also an excellent book that introduces knowledge of 3D vision, with many similarities in the basic knowledge of 3D geometry.

Joan Solà's "Error State Kalman Filter" [4] provides a very concise and precise theory of quaternion-based error Kalman filters. Although it is not lengthy, it discusses quaternions and Kalman filters very thoroughly, and most derivations in this field are based on this material. This book will also use some of its results, but we will mainly derive various filter formulas based on Lie groups rather than quaternion forms.

Professor Thrun's "Probabilistic Robotics" [5] is also a well-known classic book in the field of robotics. It introduces some results related to SLAM in the field of robotics, and provides a very detailed introduction to traditional filters, 2D grid maps, and other content. This book will also introduce 2D grid localization and mapping methods, with the theoretical part also referencing this book's content.

Professor Qin Yongyuan and Professor Yan Gongmin's works in the field of inertial navigation, including "Inertial Navigation" [6], "Kalman Filter Algorithm and Combination Navigation Principle for Strapdown Inertial Navigation" [7], and "Inertial Instrument Testing and Data Analysis" [8], are classic textbooks in this field, and many teachers and students studying inertial navigation will refer to their derivation process. This book also refers to these books in the field of inertial navigation, but compared to specialized textbooks on inertial navigation, the content introduced in this book will be relatively basic. We mainly introduce the basic principles of inertial navigation, without involving complex parameter compensation or discussions on various subdivided motion states. However, in contrast, the preintegration principle and nonlinear optimization part introduced in this book are not fully introduced in these traditional textbooks.

Finally, compared to the books and materials mentioned above, the biggest feature of this book is still the unity of code and theory. It can be said that most books are for reading, while this book can be **executed**. I believe that understanding many algorithmic aspects requires readers to participate in the debugging and running process.

Environment

This book uses Ubuntu 20.04 as the experimental environment. Readers can use their personal computers as development environments. If familiar with Docker, they can also use Docker environments. The book primarily utilizes **C++17** as the C++ standard, which may be relatively new to some readers. Older machines or environments may not necessarily support it well. We recommend readers to use software environments above Ubuntu 20.04 to run the code in this book; otherwise, you may need to address some minor issues regarding C++ standard support.

This book comes with a considerable amount of test data, which is quite large (approximately 270GB). We suggest that readers allocate at least 100GB of space to run the code in this book. Readers can download the test data through the links provided in the book's repository.

Acknowledgement

1. Considering the confidentiality of geographical information, this book avoids using domestic data and tends to use open-source datasets worldwide. Readers

can consider the trajectories or point clouds provided in the book as data in a general spatial coordinate system, without concerning themselves with the actual geographical locations of this data.

2. Similarly, unless necessary, the data provided in this book will not specify geographical information such as place names or ranges. Readers can regard them as general road, plaza, or building scenes.

3. Some of the images used in this book are sourced from internet search engines and are used solely for educational purposes, with no intention of infringing on the original authors' copyrights. Some of the images used in this book may contain logos of commercial companies or may be images used in promotional materials for some companies. These images are sourced from public search engines and do not imply any cooperation or competition relationship between the authors and the companies. The author will strive to obtain authorization for images that may have commercial copyrights. If there is any dispute, please inform us.

4. Each chapter of this book uses datasets from different sources, mainly including the NCLT dataset from the University of Michigan [9], the UTBM dataset from Montbéliard, France [10], and the UrbanLoco dataset mainly from Hong Kong, China [11] (ULHK), among others. This book has designed a unified interface for them programmatically, making it convenient for readers to test the performance of algorithms on different datasets.

5. The English version of this book is translated with the help of ChatGPT and I would like to thank their great work here.

Contents

Bibliography

[1] X. Gao and T. Zhang, *Introduction to visual SLAM: from theory to practice*. Springer Nature, 2021.

[2] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.

[3] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry, *An invitation to 3-d vision: from images to geometric models*, vol. 26. Springer Science & Business Media, 2012.

[4] J. Solà, "Quaternion kinematics for the error-state kalman filter," *CoRR*, vol. abs/1711.02508, 2017.

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

[6] Y. Qin, *Inertial Navigation*. Science Press, 2014.

[7] G. Yan and J. Weng, *Kalman Filter Algorithm and Combination Navigation Principle for Strapdown Inertial Navigation*. Northwestern Polytechnical University Press, 2019.

[8] S. L. Gongmin Yan and Y. Qin, *Inertial Instrument Testing and Data Analysis*. National Defense Industry Press, 2012.

[9] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of Michigan North Campus long-term vision and lidar dataset," *International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2015.

[10] Z. Yan, L. Sun, T. Krajnik, and Y. Ruichek, "EU long-term dataset with multiple sensors for autonomous driving," in *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[11] W. Wen, Y. Zhou, G. Zhang, S. Fahandezh-Saadi, X. Bai, W. Zhan, M. Tomizuka, and L.-T. Hsu, "Urbanloco: A full sensor suite dataset for mapping and localization in urban scenes," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2310–2316, IEEE, 2020.