Visual and Inertial Motion Estimation and Calibration

DISS. ETH NO. 25894

VISUAL AND INERTIAL MOTION ESTIMATION AND CALIBRATION

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

THOMAS SCHNEIDER

MSc. Mechanical Engineering, ETH Zurich

born on August 25, 1986
citizen of Rüthi SG

accepted on the recommendation of

Prof. Dr. Roland Siegwart, Examiner
Prof. Dr. Stephan Weiss, Co-examiner

2019

Autonomous Systems Lab
Department of Mechanical and Process Engineering
ETH Zurich
Switzerland

Abstract

Mobile robots have the potential to automate many tasks of everyday life and in the industry. For instance, robots can deliver goods or organize warehouses more efficiently. Self-driving cars promise to navigate autonomously and help to increase road safety in the near future. Micro Aerial Vehicles (MAVs) could perform regular industrial inspections in otherwise hard to reach places. All these applications require a perception system with an accurate and robust motion estimation at its core. Only such spatial-awareness enables key functionalities such as path planning, navigation or obstacle avoidance. Outdoors, these devices can rely on a Global Navigation Satellite System (GNSS) to determine their position but often the limited accuracy restricts its usage to certain applications. Indoors, however, the GNSS will fail completely. In this case, different sensor modalities can be used, for example, cameras, Inertial Measurement Units (IMUs), Light Detection and Ranging (LIDARs) or Radio Detection and Ranging (RADARs). While the latter two are still too expensive and heavy for many mobile applications, cameras and IMUs constitute an ideal sensor suite as they are light-weight, small and inexpensive. In this dissertation, we focus on developing accurate and robust visual-inertial motion estimation, mapping and localization methods that can be used in a wide variety of applications.

The promising performance of visual-inertial estimation has led to a widespread deployment of these sensors into (mass-)consumer products even though visual-inertial odometry (VIO) methods inherently accumulate drift over time. Many applications, however, require highly accurate and smooth motion estimates, for instance, pose tracking in Augmented Reality/Virtual Reality (AR/VR) headsets. In our first publication, we propose to tightly integrate localization information into the VIO by establishing and processing matches of current visual observations against the landmarks of a previously recorded localization map. Our proposed formulation provides motion estimates with increased accuracy and smoothness while at the same time reducing the drift considerably or even eliminating it entirely in certain situations.

Most of the currently available mapping and localization frameworks are supporting only single session use-cases or are tailored to very specific applications. Due to this lack of a flexible framework that can support a wide range of research in visual-inertial Simultaneous Localization and Mapping (SLAM) we developed maplab: a research-oriented framework for rapid prototyping of novel motion estimation, mapping, and localization algorithms. maplab provides a set of robust and well-tested implementations of the most important visual-inertial algorithms and tools which can be used as building blocks to accelerate the development of new algorithms, hence, reducing redundant work in the research community. On the other hand, maplab also provides a visual-inertial motion estimation, mapping and localization method that can be used out-of-the-box and has been well-tested on a variety of robots including MAVs, legged robots and ground robots. The framework is publicly available as open-source software.

Currently, most visual-inertial sensors are calibrated in a tedious manual process by experts using special equipment such as checkerboard patterns. The increasing deployment of this tech-

nology into (mass-)consumer products creates a need for continuous calibration methods that do not require equipment nor expert knowledge. In this context, we propose an observability-aware self-calibration architecture that runs in parallel to an existing motion estimation pipeline. The dataset collection process is automated by selecting informative motion segments in the background without requiring any user intervention. Once enough segments have been collected, the calibration parameters are updated using a self-calibration formulation without the need for a calibration target.

All contributions have been extensively tested and validated in real-world scenarios and on various platforms. We demonstrate accurate and robust motion estimation performance in challenging environments while the tight integration of localization constraints into the VIO reduces the drift considerably. The advantage of the proposed formulation is demonstrated in comparison to related loosely-coupled approaches on motion capture ground-truth. The efficient large-scale and multi-session mapping capabilities of our proposed framework maplab are demonstrated in a real-world mapping use-case in which we have mapped the old town of Zurich using multiple sessions recorded on hand-held tablets. The proposed life-long mapping approach has been validated in two different use-cases and four environments - showing that the achievable motion estimation performance is comparable to full-batch calibrations but at a reduced and constant computational complexity. As a whole, our contributions constitute a complete motion estimation, mapping, and localization system including continuous sensor calibration. The system can be deployed to a wide variety of mobile robots and devices while delivering good performance over the entire life of a device.

Zusammenfassung

Mobile Roboter haben das Potenzial viele Aufgaben aus dem Alltag oder in der Industrie zu automatisieren. Zum Beispiel können solche Roboter autonom Waren ausliefern oder Logistikzentren effizient organisieren. In der nahen Zukunft werden selbstfahrende Autos autonom navigieren und dazu beitragen die Sicherheit im Strassenverkehr zu verbessern. Fliegende Roboter können Inspektionsaufgaben in der Industrie übernehmen und Orte erreichen, welche sonst nur schwer zugänglich sind. Eine grundlegende Voraussetzung für solche Anwendungen ist ein zuverlässiges sensorbasiertes Wahrnehmungssystem, welches genaue Schätzungen der Bewegung und Position zur Verfügung stellt. Nur mit einer solchen räumlichen Wahrnehmung können wichtige Funktionen wie die Planung von Bewegungspfaden, eine autonome Navigation oder das Ausweichen von Hindernissen realisiert werden. In Aussenbereichen können diese Systeme Positionsdaten von GNSS-Geräten verwenden, welche aber oft nur eine beschränkte Genauigkeit aufweisen und somit die Anwendbarkeit einschränken. In Gebäuden, jedoch, versagt das GNSS komplett und liefert keine verwendbaren Daten. In diesem Fall können andere Sensortechnologien eingesetzt werden, wie zum Beispiel, Kameras, Inertialmesssysteme (IMU), LIDAR oder RADAR. Während die letzten beiden Technologien für viele mobile Anwendungen immer noch zu teuer und schwer sind, stellt die Kombination von Kameras und IMUs eine ideale Sensorplatform dar, da diese kompakt, leicht und günstig hergestellt werden können. In dieser Dissertation fokussieren wir auf die Entwicklung einer Lösung zur zuverlässigen und genauen Bewegungsschätzung, Kartografierung sowie Lokalisierung basierend auf den Daten von Kameras und IMUs für mobile Roboter.

Die bemerkenswerte Leistung von kamera- und IMU-basierter Bewegungsschätzung hat dazu geführt, dass diese Sensoren vermehrt in Produkte eingebaut werden trotz des unvermeidlichen Drifts dieser Odometriemethoden. Viele Anwendungen benötigen jedoch eine hochpräzise und möglichst rauschfreie Bewegungsschätzung, wie zum Beispiel, die Bewegungsverfolgung in Geräten der Erweiterten oder Virtuellen Realität (AR/VR). In unserer ersten Publikation entwickeln wir eine Methode mit welcher Lokalisierungsinformation direkt mit den Kamera- und IMU-Messungen zur Bewegungsschätzung fusioniert werden. Dazu vergleichen wir aktuelle Messungen von Landmarken mit jenen in einer vorhanden Lokalierungskarte, um bereits beobachtete Landmarken zu erkennen und in den Bewegungsschätzungsprozess mit einzubeziehen. Die vorgeschlagene Methode kann den Odometriedrift reduzieren, oder in gewissen Situationen sogar komplett verhindern, und somit die Genauigkeit der Bewegungsschätzung erhöhen und gleichzeitig das Rauschen reduzieren.

Die meisten heute frei verfügbaren Kartografierungs- und Lokalisierungslösungen unterstützen keine Kartografierung über mehreren Aufnahmesessionen und sind meist auf bestimmte Anwendungen zugeschnitten. Somit fehlt eine flexible Lösung, die in vielen Forschungsbereichen eingesetzt werden kann. Aus diesem Grund haben wir maplab entwickelt – eine Lösung zur schnellen Entwicklung von neuen kamera- und IMU-basierten Methoden zur Bewe-

gungsschätzung, Kartografierung sowie Lokalisierung in der Forschung. maplab bietet einen kompletten Satz von bewährten Implementierungen der wichtigsten Methoden und Werkzeugen der kamera- und IMU-basierten Bewegungsschätzung. Diese können zur Entwicklung von neuen Algorithmen verwendet werden und helfen somit unnötige und redundante Arbeit in der Forschungsgemeinschaft zu vermeiden. Andererseits kann maplab auch als fertige Lösung zur Bewegungsschätzung, Kartografierung sowie Lokalisierung verwendet werden, wo es sich bereits auf verschiedensten Robotern, wie kleine Multikopter oder Laufroboter, bewährt hat. maplab ist als quelloffene Software frei verfügbar.

Meist werden Kameras und IMUs in langwieriger Handarbeit von Experten mit spezialisierter Ausrüstung kalibriert. Aus diesem Grund erfordert der Einsatz dieser Technologie in (Massen-)Produkten neue Methoden zur kontinuierlichen Kalibration, welche keine spezialisierte Ausrüstung oder Wissen voraussetzen. Aus dieser Motivation heraus entwickelten wir eine Methode zur Selbstkalibrierung der Sensoren, welche die Beobachtbarkeit der geschätzten Parameter sicherstellt. Die vorgeschlagene Methode läuft als Hintergrundprozess parallel zu einem existierenden Bewegungsschätzer und automatisiert die Aufnahme eines Kalibrierungsdatensatzes durch die Auswahl von Segmenten mit informativer Bewegung. Dabei ist kein Benutzereingriff notwendig. Sobald genügend informative Segmente gesammelt wurden, werden die Kalibrationsparameter mithilfe einer Selbstkalibrierungs-Formulierung aktualisiert. Somit ist kein Kalibrationsobjekt, wie ein Schachbrettmuster, erforderlich.

Alle theoretischen Beiträge dieser Dissertation wurden in Anwendungen der realen Welt und auf verschiedensten Robotern ausführlich getestet und validiert. Wir zeigen auf, dass wir eine genaue und zuverlässige Bewegungsschätzung auch in schwierigen Bedingungen erreichen können, wobei die direkte Fusion von Odometrie und Lokalisierungsinformation den Drift drastisch reduzieren kann. Wir vergleichen die vorgeschlagene Methode anhand von hochgenauen Referenzmessungen und mit verwandten Methoden, welche Odometrie- und Lokalisierungsinformationen nur indirekt fusionieren, und zeigen so dessen Vorteile auf. Die Leistungsfähigkeit von maplab wurde anhand eines Experiments demonstriert, in welchen die Zürcher Altstadt grossflächig und in mehreren Sitzungen basierend auf den Sensordaten von tragbaren Tablets kartografiert wurde. Der vorgeschlagene Ansatz zur kontinuierlichen Sensorkalibration wurde in zwei verschiedenen Anwendungen und vier Umgebungen validiert. Dabei konnten wir aufzeigen, dass die Bewegungsschätzung eine vergleichbare Genauigkeit erreicht, wie wenn konventionelle Kalibrationsmethoden eingesetzt werden. Jedoch benötigt die Kalibration signifikant weniger Rechenleistung und kann in konstanter Zeit durchgeführt werden. Der Beitrag dieser Dissertation besteht aus einer kompletten Lösung zur Bewegungsschätzung, Kartografierung sowie Lokalisierung mit der Fähigkeit zur kontinuierlichen Kalibration. Das System kann auf verschiedensten mobilen Robotern eingesetzt werden, wo es über die ganze Lebensdauer des Roboters gleichbleibend genaue Bewegungsschätzungen liefern kann.

Acknowledgements

I'd like to express my sincere gratitude to my thesis supervisor Prof. Roland Siegwart, the head of the Autonomous Systems Lab at ETH Zurich, for his support and encouragement throughout this doctoral thesis. His open-minded approach combined with great trust created a wonderful environment for research and engineering and I am deeply grateful that I could be a part of it. Similarly, I would like to thank Prof. Stephan Weiss for co-supervising this dissertation.

My deepest thanks go to Dr. Paul Furgale and Dr. Simon Lynen for their guidance during the initial phase of the dissertation and for laying the foundation of this work. I also would like to thank my supervisors Dr. Igor Gilitschenski, Dr. Cesar Cadena and Dr. Juan Nieto for the countless discussions and invaluable feedback on my work. I am grateful to Luciana Borsatti and Cornelia Della Casa for their generous help and support. Similarly, I would like to thank Michael Riner and Stefan Bertschi for their excellent support in electronic and IT needs. I would also like to acknowledge the great help of Markus Bühler from the workshop team. Further, I wish to thank Dr. Simon Lynen, Dr. Mingyang Li and Dr. Konstantine Tsotsos for the great collaboration during the Google Project Tango and for giving me the opportunity to visit and work at Google for some time.

Unfortunately, there is no space to list all of my great colleagues at the lab. I'm very grateful to all of you for the countless coffee beaks, lunches and after-work beers. It was a great experience and a lot of fun to collaborate with such a great and talented team. Further, I would like to thank all of you that have helped and contributed to this work. My special appreciation, however, goes to the mapping team: Dr. Marcin Dymczyk, Marius Fehr, Titus Cieslewski, Mathias Bürki, Timo Hinzmann, Florian Tschopp, Andrei Cramariuc, Lukas Bernreiter, Mathias Gehrig and Kevin Egger. I am deeply grateful for all your help, efforts and the endless discussions that have greatly influenced this work. It was a pleasure to work with all of you and I truly enjoyed the great atmosphere in the mapping office. I am also indebted to the entire rotary wing team for their support when deploying our algorithms on their drones - including Dr. Michael Burri, Pascal Gohl, Dr. Markus Achtelik, Dr. Zachary Taylor, Dr. Helen Oleynikova, and Alex Millane. Further, I am thankful to Dr. Michael Burri, Dr. Jörn Rehder, Dr. Janosch Nikolic and Hannes Sommer for the inspiring discussions on sensor calibration. I am also deeply grateful to all the students that helped us pursue our ideas or explore alternative research directions.

My deepest thanks, however, go to my family, friends, and loved ones, especially, to my parents and grandmother for their support and for sparking my interest in engineering and building things.

February 19, 2019 *Thomas Schneider*

Financial Support

The research leading to the results presented in this thesis has partially been funded by Google's Project Tango.

Contents

Preface

This thesis is a cumulative dissertation and as such includes a selection of the most relevant publications. The publications are collected into two parts and included at the end of the thesis in full-length. Chapter 1 provides an introduction to the research topics, states the objectives and finally discusses the approach used to fulfill them. In Chapter 2, we discuss each of the included publications in terms of contribution, how it fits into the overall goal of the thesis and how it relates to other publications within the dissertation. Chapter 3 summarizes the findings and provides an outlook on future work.

Chapter 1

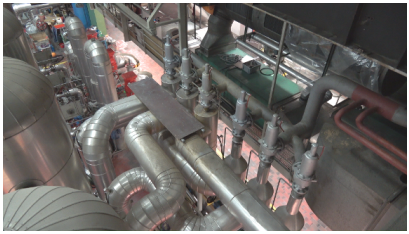
Introduction

Mobile robotics has seen tremendous progress over the last few years and has the potential to automate many tasks in industry and everyday life. Robots can assist human workers or even relieve them completely of tedious and repetitive tasks letting them focus on more important tasks. This becomes especially important in hazardous environments that pose risks to humans workers. Machines can perform tasks with constant high precision, in most cases faster than humans, and without breaks. These benefits will soon lead to a widespread deployment of such systems into public spaces, homes, and factories.

In warehouses, for example, mobile robots already organize logistics at blazing fast speeds without mistakes, or at home, they can clean floors autonomously. In the future, driver-less cars promise to increase the capacity of the road network, relieve the passengers from the monotonic driving task and most importantly improve road safety. Another promising application can be found in agriculture, where robots, for example, can monitor crops using aerial vehicles and detect deficiencies and diseases early, hence reducing the use of fertilizers and pesticides by only applying them locally where required. Autonomous machines could harvest, plant and maintain crops and orchards, thus providing ecological food at more affordable prices. After disasters, robots will soon assist search-and-rescue missions where they often can operate in dangerous environments such as collapsed buildings or in contaminated environments e.g. after radioactive or chemical leakage.

Most of current applications, however, are still restricted to structured environments, that are somewhat adapted to the robots, and are often expected to be static. Many tasks, however, require mobile robots, to operate in unstructured and complex environments and even cope with dynamic changes and obstacles. An example of such challenges is shown in Fig. 1.1. For instance, self-driving cars need to perceive the world in real-time and cannot rely on markers for navigation, and must expect obstacles (e.g. pedestrians) at any time – creating the need for accurate, robust and low-latency perception systems.

Higher level tasks such as path planning, obstacle avoidance, navigation, and manipulation are in most cases only possible when the position and motion of the robot are known. One of



(a) Complex industrial environment used in our aerial teach-and-repeat inspection demonstrator [29]. Video: https://goo.gl/LoNz1j

(b) Robots deployed for search-and-rescue missions need to navigate highly complex environments.

Figure 1.1: In many applications, mobile robots will have to operate in challenging environments – creating a need for advanced perception systems. A key building block of such a system is the motion estimation, localization, and mapping component that enables a safe navigation and obstacle avoidance in complex environments. In this work, we aim to develop accurate and robust methods for visual-inertial motion estimation and calibration to enable applications including robot navigation, teach-and-repeat scenarios or AR/VR headset tracking.

the core functionalities of the perception system in mobile robotics is to provide such spatial-awareness. Many sensor modalities are available to implement such a motion estimation and localization system, including cameras, Inertial Measurement Units (IMUs), Light Detection and Ranging (LIDARs), Radio Detection and Ranging (RADARs) and Global Navigation Satellite System (GNSS). While a GNSS can provide limited position information outdoors, its accuracy is often not sufficient for robotic needs especially in urban canyons or close to structures and will completely fail indoors. Although LIDARs and RADARs can provide highly accurate 3d information – also indoors – they are still prohibitively expensive for many applications.

On the other hand, a camera can estimate local motion without metric scale in a process called visual odometry. For most applications, however, a metric scale is desirable or even required. In this case, an IMU can provide such metric information and, additionally, increase the bandwidth of the estimates to capture highly-dynamic motions that a slower camera would miss. These complementary properties of the two sensor modalities make this pair an ideal sensor-suit allowing for accurate and high dynamic motion estimation. Additionally, the camera images can be used to build maps of the local scene structure which can be used to recognize areas that have been visited in earlier missions. Furthermore, a visual-inertial sensor system is inexpensive, compact and light-weight, and can thus enable spatial-awareness in a wide variety of applications.

Accurate and reliable measurements, from both the IMU and camera, can only be obtained by carefully modeling their sensing process and by compensating effects including lens distortion and manufacturing inaccuracies (e.g. misaligned sensor axis). The parameters of these sensor models have to be estimated in a process called ‘sensor calibration’. Usually, the parameters vary over time either by a slow drift caused by environmental effects (e.g. temperature, vibrations, etc.) or by sudden changes, for example, by shocks. As a result, the sensors must be

continuously (re-)calibrated over the entire lifetime of a device to ensure accurate and reliable motion estimation.

The goal of this thesis is to develop a framework for accurate visual-inertial motion estimation, mapping, and localization to provide spatial-awareness to a wide variety of platforms and thus increase their autonomy. We also aim at providing the framework as a research platform that can serve as a base for convenient rapid prototyping of new visual-inertial algorithms. The second goal of the thesis is to develop life-long calibration methods that enable the continuous (re-)calibration of devices in real-world settings without relying on calibration equipment nor expert knowledge. With our efforts, we aim to increase the accuracy of visual-inertial Simultaneous Localization and Mapping (SLAM) and accelerating the transition of this technology from the lab into real-world products.

1.1 Motivation and Objectives

The combination of IMUs and cameras has proved to be a lightweight and cost-effective sensor suite that can provide accurate motion estimation and localization and has thus been widely adopted by the industry. For example, it has been used for navigation or pose tracking in AR/VR headsets. The deployed visual-inertial SLAM systems, however, were previously operated under the supervision of experts and often in controlled (lab) environments. A deployment of this technology to mass-consumer products and the inherent transition from the traditional lab environment to the real-world brings a set of new challenges.

For some applications, such as virtual reality, a good performance can only be achieved with highly accurate and smooth motion estimation. To achieve these goals new formulations for the motion estimation are required, for example, using concurrent mapping and localization. An integration of localization information into the online front-end can then help to reduce the inherent drift of the odometry methods and provide motion estimates at a higher accuracy.

Another source for errors in the visual-inertial motion estimation pipeline can be found in inaccurate or out-dated sensor calibrations. Often the devices are calibrated already at the factory but external factors, such as temperature variations, vibrations, shocks, etc., make periodic re-calibrations necessary to ensure accurate and reliable operation over the lifetime of the device. In the traditional lab setting, these (re-)calibrations were performed by experts in a tedious manual process. In a mass-consumer product, however, it is desirable to perform calibrations in the background without any user intervention for three main reasons: First, the users often have no access to specialized equipment e.g. a checkerboard pattern. Second, reliable calibration depends heavily on the performed motion which, without expert knowledge, might lead to inconsistent calibrations. And finally, the user experience would suffer if periodic calibrations have to be performed manually.

The research presented in this thesis required the implementation of a multitude of tools such as camera/lens models, an efficient map structure, visual-inertial bundle-adjustment, geometric vision methods feature tracking, loop closure, and more. While we found some implementations that could be re-used, there was no framework available that combined all of these core tools for visual-inertial motion estimation, mapping, and localization in a single flexible framework. We believe that the lack of such a common framework leads to avoidable redundant work in the research community. Besides providing an out-of-the-box pose tracking solution, the

framework should also serve as a rapid prototyping platform for new methods – making the research process more efficient, facilitate the development of novel algorithms and, consequently, accelerate the transition of the technology into real-world products.

In the scope of this dissertation, we aim to advance the state-of-the-art in visual-inertial motion estimation and calibration with the following objectives:

Accurate Motion Estimation and Localization Many of today's applications require highly accurate and smooth motion estimates, for instance, the aerial inspection teach-and-repeat use-case demonstrated in [29]. Often these methods only make use of local odometry constraints and do not leverage mapping and localization methods. Using (re-)localization and fusing previously seen landmarks into the motion estimation process can help to reduce drift or even eliminate it completely. In this work, we want to investigate new motion estimation formulations that directly make use of such localization information during the online motion estimation process.

Efficient and Flexible Mapping Tools Pose estimation and mapping are still limiting many applications in mobile robotics. Most of current systems only support a limited map size hence restricting their use to small-scale scenarios. Our goal is to research efficient mapping algorithms that can cope with larger environments and support multi-session mapping by co-registering multiple sessions into a single global map. Further, the research community is lacking a set of core tools that can be used to efficiently prototype new visual-inertial motion estimation, mapping, and localization algorithms. We feel that providing the most important tools in a single framework would help avoid redundant work in future research and thus make the research process more efficient. Additionally, such a framework would allow for a more consistent comparison of different methods with minimal or no re-implementation efforts.

Life-long Sensor Calibration Visual-inertial SLAM technology drives many novel applications including mobile robotics or AR/VR headsets. In these real-world applications, the expert knowledge and equipment that was previously required to calibrate the sensors are often missing. A constant performance of the motion estimation, however, can only be ensured if the calibration is always kept up-to-date. For this reason, new calibration methods are required that continuously (re-)calibrate devices without the need for equipment or user-intervention.

Additionally, the motion estimation accuracy often suffers from modeling errors due to the use of very basic inertial models which neglect effects such as the scale distortion or misalignment of individual sensor axis. For this reason, we also investigate more detailed models to increase motion estimation performance.

1.2 Approach

The objectives of the dissertation are achieved in two parts that are also reflected in the organization of this thesis. In Part A, we develop a framework for accurate and efficient motion estimation, mapping and localization that can also be used as a research test-bed for rapid prototyping of novel algorithms. And in Part B, we focus on life-long calibration systems that

continuously calibrate sensor models without requiring any equipment or expert knowledge. The architecture of such a system is shown in Fig. 1.2.

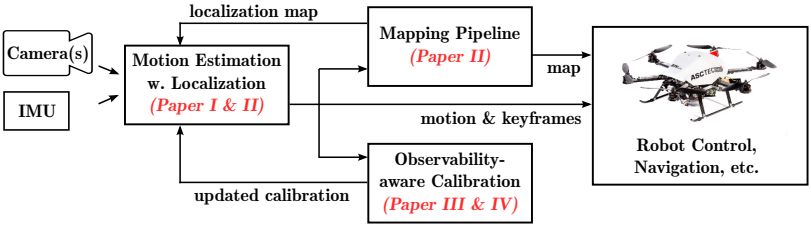


Figure 1.2: The architecture of the perception system that is being developed in this dissertation. Most of the modules are part of the open-source framework maplab except for the calibration methods that are not (yet) publicly released.

Part A: Motion Estimation and Mapping

The goal of this part is to developed a framework for visual-inertial motion estimation, mapping, and localization. A special focus is put on keeping the framework flexible with well-defined interfaces such that it can serve as a research test-bed when developing new algorithms.

Online Motion Estimation and Localization Visual-inertial odometry inherently accumulates drift over time. In many situations, re-localization against a local map can be used to limit or even eliminate this drift completely, for example, when using an AR/VR headset in a single room. We, therefore, propose a motion estimation and mapping architecture that matches current visual observations against the landmarks of a localization map. In previous approaches, the localization information was, often, integrated in a loosely-coupled fashion by fusing global with local visual-inertial odometry (VIO) pose estimates which leads to a sub-optimal estimation performance. Therefore, we propose a tight fusion of the odometry and localization information by integrating 2d-3d localization matches with visual feature tracks and inertial data. Such an online front-end provides motion estimates in a global frame with increased smoothness and accuracy. A fixed-lag smoother-based implementation is proposed in Paper I and an Extended Kalman Filter (EKF)-based version in Paper II.

Flexible Large-scale Mapping Framework Our goal is to develop an efficient multi-session mapping framework that can be deployed to various platforms and provide out-of-the-box mapping and localization capabilities. A special focus is put on efficient data structures and robust algorithms that support large-scale mapping. The framework provides implementations for the most important algorithms including camera/lens models, map optimization, loop-closure, visualization, introspection, and evaluations, etc. All these algorithms and tools are integrated into a single framework to accelerate future

research by providing a rapid prototyping environment where the existing algorithms can serve as a base for new developments. The framework is described in Paper II and is publicly available as open-source software: `https://github.com/ethz-asl/maplab`.

Part B: Sensor Calibration

In this part, we want to develop life-long calibration methods to keep the sensor calibrations up-to-date over the entire life of a device. We specifically address the challenges when deploying visual-inertial calibration into (mass-consumer) products that are operated by non-expert users. First, we need to automate the dataset collection process as end-users rarely have the knowledge on how to excite all sensors properly. Second, the calibration should be performed as a background process without any user intervention and without requiring any equipment or tools. And finally, we want the process to be efficient such that it can be used on mobile devices with limited resources.

Continuous Self-Calibration Accurate motion estimation is only possible if the sensor calibration is accurate and up-to-date. These calibrations, however, vary over time due to various effects such as temperature changes, vibrations or shocks. For this reason, we propose a calibration architecture that continuously (re-)calibrates the sensors to track these changes. This is especially important when using cheaper sensors, such as Micro Electro-Mechanical Systems (MEMS) IMUs, for which the calibrations are less stable. In practice, continuous calibration can only be achieved using self-calibration formulations that do not rely on external markers such as checkerboard patterns. With this approach, a degradation of the sensor calibrations can be avoided and a constant motion estimation performance can be ensured.

The proposed framework is described in Paper III and Paper IV.

Calibration on Informative Motion Often calibrations have to be performed on mobile devices with limited computational power. For this reason, we propose to sparsify calibration datasets to only retain the most informative sections of a given trajectory. This is possible as the information is often not equally distributed in visual-inertial datasets as illustrated on an example in Fig. 1.3. Such an approach enables consistent calibrations even on mobile devices with limited computational resources.

Second, the dataset collection process can be automated by selecting informative segments while the device is being used. This not only renders (re-)calibrations into a background process but also avoids having the user to perform consistent exciting motion deliberately which might be hard for non-experts. Further, this method facilitates the use of more advanced sensor models that might be hard to excite manually.

The informative sparsification is proposed in Paper III for single-session datasets and extended to the multi-session use-case in Paper IV. The latter paper also introduces additional information theoretic metrics to select informative motion and extends the evaluation of the first publication.

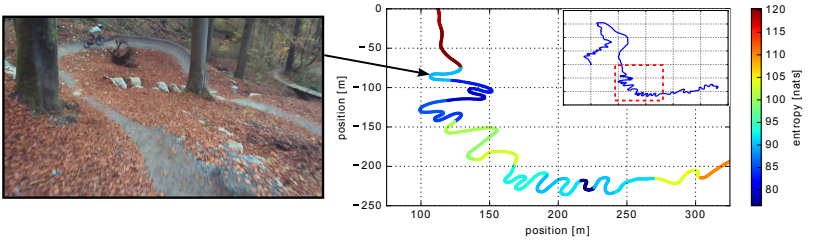


Figure 1.3: This dataset of a bike ride is a good illustration that the information is often not uniformly distributed along a trajectory in visual-inertial datasets [90]. The color indicates the information content for calibration within a segment (where a lower entropy indicates more information). Our proposed calibration method leverages this non-uniformity to: (a) sparsify the calibration dataset to only include informative sections and thus enable efficient calibration on resource-constrained systems, and (b) to automate the dataset collection by selecting informative motion as it occurs, and thus removing the need to perform it consciously by the user – which might be difficult to perform consistently for non-experts.

Improved Sensor Models Most visual-inertial estimation frameworks use very basic sensor models for the inertial sensors. Especially, cheaper sensors, which are widely deployed in mobile devices, are more sensitive to external effects, such as, temperature variations and, often, contain manufacturing inaccuracies, such as, non-orthogonal sensing axis. It has been shown in e.g. [75] that a calibration of the intrinsics can increase the performance of the motion estimation. For this reason, we have included the misalignment and scale factors of each inertial axis as well as a rotation between the gyroscopes and the accelerometer similarly to [50]. Furthermore, we calibrate these parameters online in a self-calibration formulation without relying on calibration targets.

These more complete models and their calibration are described in Paper III and Paper IV.

Chapter 2

Contribution

In this chapter, we discuss the scientific contributions achieved within the scope of this dissertation. We provide the context for each publication, explain how the dissertation objectives are addressed and explain their core contributions as well as how they relate to other publications. Finally, a list of all publications, student projects, and open-source releases is given.

2.1 Part A: Motion Estimation and Mapping

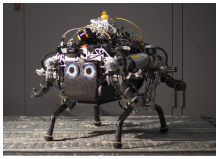
The publications in the first part focus on visual-inertial motion estimation and mapping. The first publication discusses a motion estimation framework which tightly integrates localization constraints into a visual-inertial odometry framework. The second work introduces maplab – an open-source visual-inertial motion estimation, localization, and mapping framework that was developed for different robotic platforms some of which are shown in Fig. 2.1. Most of the contributions are released as part of the open-source project maplab: `https://github.com/ethz-asl/maplab`.

Paper I

Thomas Schneider, Marcin Dymczyk, Timo Hinzmann, Igor Gilitschenski, Simon Lynen, Roland Siegwart, "Real-time Visual-inertial Localization using Summary Maps". In *Tech Report, Autonomous Systems Lab, ETH Zurich*, 2015.

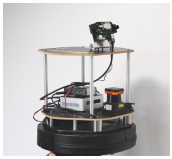
Context

Being able to precisely track the motion of a robot is crucial for many applications in robotics such as autonomous navigation or obstacle avoidance. However, (visual-inertial) odometry methods inherently accumulate drift over time. In this work, we match visual observations against a localization map, that was built in a previous session, to reduce this drift or even elim-

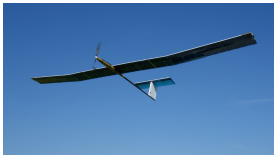


(a) Legged Robot: Starleth

(b) Hexacopter: Auk

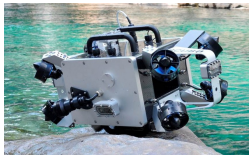


(c) Ground Vehicle: Turtlebot



(d) Fixed-wing Aircraft

(e) Mobile Manipulation Platform: Mobmi



(f) Under-water Vehicle: Scubo

Figure 2.1: Some of the robotic platforms at the Autonomous Systems Lab on which the proposed motion estimation, mapping, and localization methods have been developed, tested and deployed.

inate it completely in certain areas. In contrast to previous work (e.g. [78]), we tightly integrate the map matches into a sliding-window-based visual-inertial odometry estimation framework. Using such a tightly-coupled approach, we are able to provide smoother and more accurate motion estimates while at the same time being more robust against outlier map matches.

Often, the matching of visual observations against the localization map becomes prohibitively expensive for larger maps. For this reason, we employ the summarization scheme developed in [20] to only retain the most informative landmarks of a given localization map and thus enable localization on resource-constrained systems.

Contribution

We propose a formulation to tightly fuse the 2d-3d map matches with the visual odometry feature tracks and the inertial data in a fixed-lag-smoother architecture. Such a formulation allows to jointly estimate the local motion of the sensor system as well as a global map alignment while exploiting all cross-terms. We evaluate the performance in a collaborative robot scenario in which an Unmanned Ground Vehicle (UGV), first, navigates the environment to build a localization map, and, second, an Micro Aerial Vehicle (MAV) localizes within the shared map. The ground-truth data of a Vicon motion capture system is used to evaluate motion estimation errors. A comparison against a related loosely-coupled approach shows increased smoothness and accuracy of the proposed method. Finally, we perform a study to investigate the effects of increasing map summarization on motion estimation accuracy. We can show that the proposed method allows for more aggressive summarization while retaining the accuracy achieved with larger localization maps.

Interrelations

The proposed visual-inertial motion estimation and localization framework was successfully used in multiple projects at our lab e.g. in a collaborative mapping and localization scenario using a UGV and an MAV [27], in fixed-wing applications including [42] and [43]. The modular and flexible implementation has also served as a base to include additional sensors into the visual-inertial odometry (VIO) estimator e.g. wheel odometry in a car application [12] and [13]. More recently the framework has been licensed by a spin-off company from ETH Zurich where it is deployed on ground-robots in a teach-and-repeat setting.

The system was a predecessor of maplab (see Paper II) in which we have integrated additional algorithms and replaced the fixed-lag-smoother-based front-end with an Extended Kalman Filter (EKF) implementation. While this framework does not support online calibration, we explore such methods in the following publications Paper III and Paper IV.

A video demonstrating the proposed method in a collaborative mapping and localization use-case using a walking and a flying robot can be found at `http://goo.gl/DgqxWv`.

Paper II

Thomas Schneider*, Marcin Dymczyk*, Marius Fehr*, Kevin Egger, Simon Lynen, Igor Gilitschenski, Roland Siegwart
(* contributed equally), "maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization". In *IEEE Robotics and Automation Letters (Volume: 3, Issue: 3, July 2018)*, 2018.

Context

maplab was created in a collaborative effort to combine all visual(-inertial) mapping and localization research within the Autonomous Systems Lab into a single framework. An important goal was to provide a framework that could be deployed on most of the robotic platforms at the lab to conveniently provide access to the current research efforts in motion estimation, mapping, and localization. We also aim at reducing redundant work by sharing the implementations of basic tools including map optimization, geometric vision algorithms, visualization and introspection tools, etc. Further, the framework facilitates collaboration by providing a common map format that allows for easy exchange and comparison of data.

Contribution

maplab introduces an open-source visual-inertial mapping and localization framework which is mainly focused at research by providing a test-bed for rapid prototyping of new algorithms. Its modular design and flexible interfaces make it easy to extend and integrate new algorithms. But it can also be seen as a ready-to-use (multi-session) visual-inertial mapping and localization system providing a front-end to estimate ego-motion and perform online localization as well as a console-based back-end for map building and manipulation.

The framework facilitates prototyping of new algorithms by providing a set of the most important tools used in visual-inertial mapping and localization including implementations for visual-inertial bundle-adjustment, pose graph relaxation, loop-closure using binary descriptors, multi-session map merging, map compression, visualization and introspection tools, and dense

reconstruction. A console interface provides easy access to the algorithms where they can be applied on the loaded (multi-session) maps using simple commands without the need of modifying any code. New algorithms can be integrated into the console using a plug-in architecture. Additionally, maplab tightly integrates with aslam_cv2 – a modular computer vision library providing data structures, camera and distortion models, geometric vision methods, feature tracking and matching.

maplab provides an efficient and extensible map structure that can be serialized to disk to enable easy data storage and exchange. The map structure uses a resource management system to attach large objects to the pose-graph e.g. laser data or dense reconstructions. The most common map queries, such as geometry, nearest-neighbor, search, etc., are already implemented and available to develop new algorithms.

maplab introduces ROVIOLI as a visual-inertial odometry and localization front-end. It can be used to build maps and to localize against a previously built map online – either from a previous ROVIOLI sessions or a processed (multi-session) map exported from the maplab console. ROVIOLI is built around the popular visual-inertial odometry framework ROVIO [7] and tightly integrates localization constraints using 2d-3d matches of visual observations against landmarks of a localization map. In this work, we do not model the uncertainty of the localization landmark position. While this simplification seems to work well in practice, it would be interesting to investigate the effects on the consistency of the global pose estimate in future work.

The framework has been well-tested in various projects and on various platforms including walking robots, flying robots (rotary and fixed wing), an underwater robot, wheeled ground-based robots and hand-held devices. The presented evaluations include a mapping experiment of Zurich's old town to showcase the large-scale mapping capabilities as well as an evaluation of the motion estimation and localization accuracy on the EuRoC datasets [11]. A later publication, further, demonstrates the long-term localization capabilities of the maplab localization system [23].

We hope that maplab will be useful for the research community and help to make research in visual-inertial localization and mapping accessible without having to (re-)build common tools and algorithms.

Interrelations

An early version of maplab was presented in Paper I where it was used for research in visual-inertial localization. The work introduced an estimator that tightly fuses visual and inertial information with 2d-3d matches against a previously built localization map. In this context, maplab was used for mapping, visual-inertial bundle-adjustment and matching visual observations against the landmarks of the localization map.

Additionally, maplab has been used in multiple co-authored publications including research and experiments on map compression by selecting informative landmarks [22], long-term large-scale mapping and localization [23], collaborative mapping using a legged robot and a flying MAV [27], topological mapping and navigation [6], an evaluation of current VIO frameworks in train applications [99], on fixed-wing platforms for mapping and localization [42, 43] and for map quality evaluation [64].

The capabilities of maplab are demonstrated in the following video in a large-scale multi-

session mapping and localization use-case: http://goo.gl/XvS1Hp.

2.2 Part B: Sensor Calibration

This part of the thesis discusses our publications on visual-inertial sensor calibration. The first publication introduces an observability-aware self-calibration framework which collects informative motion segments in a background process to update sensor calibrations online. The second publication extends the first and investigates additional metrics to select informative motion and performs an extensive evaluation of the method in different scenarios and use-cases.

Paper III

Thomas Schneider, Mingyang Li, Michael Burri, Juan Nieto, Roland Siegwart and Igor Gilitschenski, "Visual-inertial self-calibration on informative motion segments". In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

Context

Visual-inertial motion estimation requires accurate and up-to-date sensor calibrations. In a traditional lab setting, visual-inertial sensors have been calibrated by an expert often using a calibration target (e.g. checkerboard). When deploying these sensors systems to consumer products, however, we face new challenges as the end-users usually lack the knowledge on how to excite all modes of the system properly. Further, requiring external calibration tools will negatively impact the user experience or often such equipment is just not accessible when needed.

A popular approach for actuated systems is to plan and execute 'informative trajectories' to automate the dataset collection and thus guarantee good and consistent calibrations as proposed, for example, in [5] or [81]. However, in this work, we focus on non-actuated systems such as Augmented Reality/Virtual Reality (AR/VR) devices or mobile phones. We, therefore, analyze the information content of small trajectory segments in a background process that runs in parallel to an existing motion estimation pipeline. The most informative segments are then maintained in a database for later calibration. Our method bases on the assumption that informative motion will occur eventually and thus takes the burden from the user to perform such motion consciously.

Contribution

We have developed a calibration framework that runs in parallel to an existing motion estimation pipeline. We use an efficient information theoretic metric based on the entropy to collect the most informative segments of a trajectory. A segment-based calibration is triggered once enough segments have been collected. We use a self-calibration formulation to estimate the intrinsics and extrinsics of the visual-inertial sensor system and thus do not rely on any external markers (e.g. checkerboard pattern). In our evaluations, we can show that the proposed method achieves comparable results to a full-batch calibration while requiring significantly less data and thus less computational resources.

With this method, we are able to render the dataset collection and calibration into a background process that does not require any user intervention. Further, the observability-aware selection of segments facilitates the use of more sophisticated sensor models which might be hard to excite manually.

The effects of the dataset sparsification on the estimator consistency were not investigated in this publication and should be part of future work.

Interrelations

Paper II provides offline sensor re-calibration through the visual-inertial bundle-adjustment but it does not include online calibration capabilities. The proposed method can automate this task and provide up-to-date calibrations without any user intervention. Thus, we can increase the accuracy of visual-inertial motion estimation and mapping capabilities.

We calibrate a similar inertial model as in our framework kalibr (see Section 2.5) which is described in our previous co-authored publication [84]. Instead of relying on the observations of a known calibration target, we use a self-calibration formulation to allow calibrations without the need for any equipment. Additionally, we sparsify the calibration dataset based on information-theoretic metrics to reduce the complexity of the problem and make online calibration accessible to resource-constrained mobile platforms.

It is important to note that we explored the questions, of what good calibration motion is, for the visual-inertial use-case, however, the method can be extended to arbitrary calibration problems (as long as the underlying uncertainties can be represented correctly).

Paper IV

Thomas Schneider, Mingyang Li, Cesar Cadena, Juan Nieto, and Roland Siegwart, "Observability-aware Self-Calibration of Visual and Inertial Sensors for Ego-Motion Estimation". In *IEEE Sensors Journal*, 2019.

Context

We build on Paper III in which we have sparsified (single-session) calibration datasets to only retain the most informative segments. Often the individual sessions, however, are too short to excite all modes of the sensors properly, for example, in augmented-reality navigation use-cases where a device might be used frequently but only for a short amount of time. Therefore, we explore the feasibility of accumulating informative segments from multiple sessions to calibrate the visual and inertial sensor models. Further, we introduce new metrics to select informative trajectory segments and provide an extended evaluation on motion capture ground-truth.

Contribution

We use the same observability-aware calibration architecture as in Paper III and propose three information-theoretic metrics to assess the information of segments. We perform a study on new datasets to compare the metrics in four different environments and two use-cases. An experimental evaluation shows that the proposed metrics outperform random selection when

used for calibration dataset sparsification. However, all metrics perform similar and further evaluation is required to make conclusive statements whether one of the metrics performs better in certain scenarios.

Additionally, we perform an evaluation on motion capture ground-truth to assess the motion estimation accuracy that can be achieved using calibrations obtained with the proposed method. The results show that comparable performance to full-batch calibrations can be achieved. However, the dataset size can be limited by only considering the most informative part of a dataset and thus enable calibrations on resource-constrained systems where a full-batch calibration would be prohibitively expensive. Further, the evaluations show that an accumulation of information from multiple short sessions is feasible and leads to reliable calibrations. Therefore, the method can provide consistent calibrations in use-cases with frequent short sessions where an individual session would not provide enough excitation. Finally, a comparison against a related EKF-based method, which jointly estimates motion and calibration, demonstrates the benefits of the proposed method.

Interrelations

This work bases on Paper III and additionally proposes three different information metrics to select informative segments, extends the evaluations to two different use-cases in four environments and adds a comparison against a related EKF-based method. While Paper III focused on single-session datasets, Paper IV also considers the collection of informative segments from multiple sessions.

2.3 List of Publications

This section provides a list of the publications that were made in the scope of this dissertation. The most relevant papers have been selected and are included in full-length in this thesis.

Journal articles:

- T. Schneider, M. Li, C. Cadena, J. Nieto, and R. Siegwart. Observability-aware self-calibration of visual and inertial sensors for ego-motion estimation. *IEEE Sensors Journal*, 19(10):3846–3860, May 2019

- T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart. maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425, July 2018

- F. Tschopp, T. Schneider, A. W. Palmer, N. Nourani-Vatani, C. Cadena, R. Siegwart, and J. Nieto. Experimental comparison of visual-aided odometry methods for rail vehicles. *IEEE Robotics and Automation Letters*, 2019. (accepted for publication)

- M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The euroc micro aerial vehicle datasets. *IJRR*, 35(10):1157–1163, Sept. 2016

Conference papers:

- T. Schneider, M. Li, M. Burri, J. Nieto, R. Siegwart, and I. Gilitschenski. Visual-inertial self-calibration on informative motion segments. In *IEEE Int. Conf. on Robotics and Automation*, pages 6487–6494, 2017

- F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart. Topomap: Topological mapping and navigation based on visual slam maps. *IEEE Int. Conf. on Robotics and Automation*, pages 1–9, May 2018

- J. Rehder, J. Nikolic, T. Schneider, and R. Siegwart. A direct formulation for camera calibration. In *IEEE Int. Conf. on Robotics and Automation*, pages 6479–6486, 2017

- J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart. Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. In *IEEE Int. Conf. on Robotics and Automation*, 2016

- M. Dymczyk, T. Schneider, I. Gilitschenski, R. Siegwart, and E. Stumm. Erasing bad memories: Agent-side summarization for long-term mapping. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 4572–4579, Oct 2016

- P. Fankhauser, M. Bloesch, P. Krüsi, R. Diethelm, M. Wermelinger, T. Schneider, M. Dymczyk, M. Hutter, and R. Siegwart. Collaborative navigation for flying and walking robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2859–2866, Oct 2016

- T. Hinzmann, T. Schneider, M. Dymczyk, A. Melzer, T. Mantel, R. Siegwart, and I. Gilitschenski. Robust map generation for fixed-wing uavs with low-cost highly-oblique monocular cameras. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3261–3268, Oct 2016

- T. Hinzmann, T. Schneider, M. Dymczyk, A. Schaffner, S. Lynen, R. Siegwart, and I. Gilitschenski. Monocular visual-inertial SLAM for fixed-wing UAVs using sliding window based nonlinear optimization. In *IEEE International Symposium On Visual Computing*, pages 569–581, Cham, 2016. Springer

- L. Traffelet, T. Eppenberger, A. Millane, T. Schneider, and R. Siegwart. Target-based calibration of underwater camera housing parameters. In *IEEE Int. Symposium on Safety, Security, and Rescue Robotics*, pages 201–206, Oct 2016

Workshop papers:

- M. Fehr, T. Schneider, M. Dymczyk, J. Sturm, and R. Siegwart. Visual-inertial teach and repeat for aerial inspection. *Workshop International Conference on Robotics and Automation (ICRA)*, 2018

- M. Dymczyk, M. Fehr, T. Schneider, and R. Siegwart. Long-term large-scale mapping and localization using maplab. *Workshop International Conference on Robotics and Automation (ICRA)*, 2018

2.4 List of Supervised Students

This section lists all student projects that have been conducted in collaboration with the author and are sorted by the type of the project.

Master Thesis:

Master student, 6 months full time

- Ghosh, Partha (2015): "Improving Scalability Of Bundle Adjustment"
- Boada, Ricard (2016): "Active Camera Calibration for Robotic Systems"
- Morara , Elena (2016): "Odometry and Mapping Inside Pipes"
- Egger, Kevin (2016): "Vision Based Person Following for a Quadrupedal Robot"
- Radomski, Adam (2017): "Closed-Loop Multi-Sensor Simultaneous Localization and Mapping (SLAM) for Fixed-Wing UAVs"
- Blöchliger, Fabian (2017): "Topological Mapping and Navigation based on visual SLAM Maps" [6]
- Eiholzer, Flavio, (2018): "Learning a Descriptor for Visual Sequence Matching"
- Bögli, Josua, (2018): "Development of an online SLAM System for maplab"
- Schilling, Milan, (2019): "Real-time Motion Estimation using Visual, Inertial and LIDAR Data"
- Benjamin, Hahn, (2019): "Segment-based re-localization using stereo cameras"

Semester Thesis:

Master student, 3-4 months part time

- Gehrig, Matthias (2015): "Daily Autonomous Mapping of an Indoor Environment"
- Huber, Marius (2015): "Deep Learning for Map Summarization"
- Ye, Yawei (2016): "Feasibility Study of GNSS Outlier Rejection Using Additional Pose Information"

- Eiholzer, Flavio (2017): "Semantic mapping for VI maps: Enriching sparse maps with semantic information"
- Rosinol Vidal, Antoni (2017): "Autonomous Navigation using Sparse Visual Inertial Maps with a Computationally Constrained MAV"
- Yu, Yilun (2017): "Deep Scale Learning for Visual Motion Estimation"

Bachelor Thesis:

Bachelor student, 3-4 months part time

- Strässle, Timo and Bischoff, Reto (2016): "Underwater Visual Odometry for an Omnidirectional Submersible Robot" [97]
- Eppenberger, Thomas and Traffelet, Leonie (2016) "Camera Calibration for an Underwater Robot"
- Schulz, Yannick (2016): "Direct, Radiometric Calibration of RGB Cameras"

2.5 List of Open-source Software

The author has (co-)developed multiple frameworks including visual-inertial motion estimation, mapping and localization software, and calibration algorithms. The most relevant open-source releases within the scope of the dissertation include:

- **maplab**: maplab is a research platform for visual and inertial estimation and mapping that has been tested extensively on real robots. It facilitates rapid prototyping of new mapping and localization methods by providing a set of the most common algorithms to build upon including robust visual-inertial odometry with localization, large-scale multi-session mapping and optimization, and methods for dense reconstruction.
  `https://www.github.com/ethz-asl/maplab`
- **aslam_cv2**: A general computer vision library including convenient data structures, camera/projection models and algorithms for geometric vision, feature tracking and matching, etc.
  `https://www.github.com/ethz-asl/aslam_cv2`
- **kalibr**: A toolbox to calibrate the intrinsic and extrinsic parameters of an Inertial Measurement Unit (IMU) and camera system in a continuous-time batch formulation. [84]
  `https://www.github.com/ethz-asl/kalibr`

Chapter 3

Conclusion and Future Directions

In this dissertation, we have investigated methods for accurate visual-inertial motion estimation, localization and calibration. This chapter provides a brief summary of the findings and discusses potential future directions for research in these topics.

3.1 Part A: Motion Estimation and Mapping

In the first part of the dissertation, we built a framework for visual-inertial motion estimation, localization, and mapping.

The first publication proposed a visual-inertial motion estimation and localization estimator that tightly integrates 2d-3d map matches with sparse feature tracks in a fixed-lag smoother architecture. This formulation jointly estimates the local motion as well as the global pose within the localization map. The evaluations on motion capture ground-truth demonstrate the increased accuracy of such a tight integration when compared to a loosely-coupled fusion of visual-inertial odometry (VIO) poses with 6-Degrees-of-Freedom (DoF) localization constraints. A study shows that the proposed formulation allows for a more aggressive map compression at the same localization accuracy when compared to a loosely-coupled approach. Our implementation has successfully been used in several publications and on different platforms e.g. on an Micro Aerial Vehicle (MAV) and an Unmanned Ground Vehicle (UGV) [27], on a fixed-wing platform [42, 43] and on a car [12, 13]. More recently the framework has been licensed by spin-off company that deploys it on ground-robots in a teach-and-repeat setting.

The second publication introduced maplab – an open-source visual-inertial motion estimation, mapping, and localization framework. maplab provides visual-inertial mapping and localization capabilities for robotic platforms which can be used to estimate ego-motion, build multi-session maps and localize online in such maps. An implementation of the most important tools is provided including bundle-adjustment, pose-graph relaxation, loop-closure, multi-session map merging, map summarization, and visualization and introspection tools. The modular design allows for convenient rapid prototyping of new algorithms which has led to the framework being

used in multiple publications including research in map compression [22], long-term localization experiments [23], collaborative mapping [27] or topological mapping for navigation [6].

We believe that current (visual-inertial) motion estimation, mapping, and localization system, such as the one proposed in this thesis, can provide accurate pose tracking in many environments and already enable a lot of useful applications such as indoor robot navigation and Augmented Reality/Virtual Reality (AR/VR) headset tracking. However, there are many interesting directions for future research some of which are just opening up now:

Multi-modal Perception Although visual-inertial sensing can capture highly dynamic motion and cover a wide range of applications, it still has its weaknesses. The camera, for example, can fail in poor illumination conditions, during fast motions leading motion blur or in low-textured environments. The accelerometers require enough excitation in all directions for the biases to be observable and only then provide meaningful information for 6-DoF motion estimation – making their use in e.g. ground-vehicle with planar, often even constant-velocity, motion questionable.

Multi-modal perception system can mitigate such weaknesses by including complementary sensor modalities. We believe that a Light Detection and Ranging (LIDAR) could be a complementary sensor as it directly provides metric 3d information of its surroundings even in complete darkness. The LIDAR will fail in symmetric environments (e.g. in a corridor or on a large open plane), however, the camera could provide useful constraints in this situation. Another useful sensor could be a Radio Detection and Ranging (RADAR) system which can still provide information in dense fog or rain where both the LIDAR and camera could fail. For high dynamic applications, an event camera might be an interesting option but will require very fast low-latency processing of the data most likely in hardware to deal with the high frequency of data.

Overall, multi-modal perception pipelines can greatly increase the robustness by fusing sensors with complementary characteristics or just by plain redundancy in case a sensor should fail. Such improvements will be especially important in safety-critical applications, for instance, in autonomous driving.

Novel Hardware and Technical Standards Robotics would vastly benefit from a standardized communication bus for the data exchange between sensors and processing units. Similarly, an ever reoccurring issue – the temporal synchronization of multiple sensors – could be addressed by a standard and implemented on such a common bus. We believe that such a standard would greatly boost the adoption of advanced perception systems, especially, the use of multi-modal sensor systems.

LIDARs are still quite heavy and expensive limiting their use in consumer-products e.g. autonomous cars or MAVs. The solid-state LIDAR technology might address some of these issues and hopefully enable their use in more applications. However, the interference between multiple LIDAR units operating in the same environment has to be addressed before a large-scale deployment is possible.

Thermal imaging is a promising sensing modality for low-light or foggy conditions, however, the module size, low-resolution, and cost currently limit their application in products. Some companies just started to miniaturize the sensors but the resolution is still

quite low for motion estimation applications. Also, many devices are still export controlled which limits its adoption further.

Map Representations Although the proposed framework can handle map sizes spanning several city blocks, the time required for operations on the map, such as optimization or localization, increases with size. For most applications, such a global metric map is not required and could be split up into smaller sub-maps [26]. This might be a way of addressing the scalability problem as only a small part of the actual map must be considered during optimization, localization or path planning. Further, it enables an efficient dense mapping approach where 3d data is attached to the sparse pose-graph. When the pose-graph is updated, e.g. with loop closures, the dense data can just be moved around to build a globally consistent 3d map. For navigation applications, hybrid topological-metric maps might serve as a good representation. The metric sub-maps could be connected through topological links to enable efficient path planning and navigation [6].

Scene Understanding The recent progress in machine learning, especially deep learning, has improved scene understanding (semantic, geometric, context, etc.) tremendously. We can now understand the content and context of images or even 3d point clouds which provides a new dimension of information next to the raw measurements. This information can be used to robustify the motion estimation, mapping, and localization process, for example, by removing dynamic objects or by improving image localization under vast appearance or viewpoint changes. The understanding of high-level objects and their relations might lead to a very compact representation of the world by only including a set of primitives and their connections/relations.

3.2 Part B: Sensor Calibration

In the first publication, we proposed an observability-aware self-calibration method for visual and inertial sensor systems. The dataset collection is automated by selecting informative motion in a background process and thus provides consistent calibrations without requiring expert knowledge on how to excite the sensors properly. The self-calibration formulation removes the need for any external calibration targets (e.g. checkerboard patters) and enables (re-)calibrations in-field e.g. on mass-consumer products. We show that the sparsified calibration problem, which only includes the most informative portion of a dataset, estimates similar calibration parameters to full-batch solutions but at a constant computational complexity (independent of the duration of a session). Thus the method enables calibration even on long datasets and resource-constrained platforms where a full-batch calibration would be infeasible. Further, the observability-aware selection of calibration motion facilitates the use of more advanced sensor models which might be difficult to manually excite consistently.

In our second publication, we have investigated additional information metrics for the selection of informative motion. An evaluation on motion capture ground-truth validates the calibration approach in four different environments and two use-cases and demonstrates that the proposed method achieves comparable motion estimation performance to a full-batch calibration. Further, we can not only sparsify long calibration datasets but also obtain consistent

calibrations by accumulating informative segments from multiple short sessions where a single session would not provide enough excitation for a reliable calibration. Thus, the proposed framework ensures accurate motion estimation over the entire lifetime of a device. Although we have developed the calibration routines for visual-inertial sensors systems, the method is also applicable for arbitrary sensors system.

We believe the following research directions are worth exploring in the future:

Detection of Miscalibration and Sensor Failures Due to unmodeled sensor effects, for instance temperature variations, aging, vibrations or shocks, a calibration can either slowly drift or change abruptly. These dynamics can either be captured using periodic re-calibrations or by detecting changes in the calibration parameters to trigger a re-calibration. The latter would avoid unnecessary calibrations and thus save resources and potentially battery power.

Additionally, in multi-modal sensor setups, it is of great importance that sensor failures and miscalibration can be detected fast and reliably. This information can then be used to reconfigure the estimation framework online to provide accurate state estimates without interruption. In a next step the failing sensor could be either re-calibrated or taken offline completely.

Advanced Sensor Modelling More advanced sensor models could further improve the estimation accuracy and also decrease the frequency of required (re-)calibrations. For instance, the effect of temperature variations on the measurements of inertial sensors is usually neglected in current models. Such effects are usually not calibrated for as they are device specific and require a large amount of measurement data. In this context, it might be interesting to explore cloud-based calibration techniques where the data of many devices are accumulated to learn a data-driven model. A motivating example for the use of more advanced models can be found with the gyroscopes: high-quality and well-calibrated gyroscopes can directly measure the rotation of the earth and thus can sense an earth-fixed reference vector. This information can be used to determine the heading of the unit (gyro-compass) and eliminate the inherent yaw drift of VIO methods completely.

Efficient Classification of Informative Motion The selection of informative motion has to be as efficient as possible as it must be evaluated for each motion segment online. This becomes especially important when deploying these algorithms on mobile platforms with limited battery capacity. Instead of using the proposed model-based approach (Paper III, Paper IV), it would be interesting to investigate whether a simple classifier could be trained to determine whether a given motion segment is informative. In this case, the potentially more expensive model-based approach could serve as a mean to create/label training data. Such an approach might be more efficient especially for models with a large parameter space.

Suggesting Informative Motion for Calibration In this dissertation, we focused on the observability-aware calibration of non-actuated systems. For this scenario, we propose to analyze the occurring motion and collect informative trajectory segments for later

calibration. This method relies on the assumption that such exciting motion occurs eventually.

For actuated platforms, the research community has presented motion planning methods that deviate trajectories in favor of better observability thus finding a compromise between the mission and keeping the sensor calibrations up-to-date (e.g. [5] or [81]). Such methods could be combined with the ideas presented in this thesis to suggest informative motion to the user in case the occurring motion is not sufficiently informative. One could even integrate such an approach into applications, for instance AR/VR games or navigation applications, such that the user performs motion that leads informative trajectories without doing it intentionally.

Part A

MOTION ESTIMATION AND MAPPING

Paper

Real-time Visual-inertial Localization using Summary Maps

Thomas Schneider, Marcin Dymczyk, Timo Hinzmann, Igor Gilitschenski, Simon Lynen, Roland Siegwart

Abstract

Localization in a global reference frame constitutes a fundamental milestone towards high-level applications in robotics such as autonomous navigation and obstacle avoidance. Visual-inertial Simultaneous Localization and Mapping (SLAM) became a compelling method for this task, despite its inherent drift and local pose estimation. An approach to these shortcomings, however, can be achieved by matching against maps built in previous sessions. Commonly, a careful data selection is performed to keep the map size traceable and thus enable localization in real-time. Although, such a map summarization usually guarantees global localization coverage, the accuracy suffers due to fewer matches.

In this work, we aim at mitigating this effect by directly integrating the scarce 2d-3d matches with visual feature tracks and inertial measurements in the framework of a sliding-window based optimization. We compare our approach to motion tracking data and demonstrate that such a joint estimation yields smoother and more accurate global pose estimates than related methods that loosely integrate 6-Degrees-of-Freedom (DoF) localization poses with visual-inertial odometry (VIO). Finally, we evaluate the impact of varying map summarization parameters on the trade-off between map-size and localization accuracy and demonstrate that our approach allows for a more aggressive summarization while retaining the robustness and accuracy achieved with larger maps.

Published in:
Tech Report, Autonomous Systems Lab, ETH Zurich, 2015

1 Introduction and Related Work

Being able to precisely determine the location of agents w.r.t. a known frame of reference, for instance using Global Positioning System (GPS) is key to nearly all robotic applications. However, when such global positioning methods are unavailable alternative sensor modalities, such as cameras and time-of-flight sensors in conjunction with state estimation algorithms, become essential. In contrast to GPS signals, these sensor modalities estimate the pose in a local frame of reference that is tied to a map built using Simultaneous Localization and Mapping (SLAM). This lack of a common frame of reference has pivotal impact on collaborative robotic applications such as industrial inspection or disaster relief [65]. In recent years, several approaches have been proposed to co-localize multiple agents in a shared map: Strategies for collaborative mapping can involve a central server for Micro Aerial Vehicles (MAVs) [30], or teams of ground and aerial robots [31]. On the other hand, collaborative mapping can be performed in a distributed fashion as proposed in [15, 86]. In particular, a team of heterogeneous robots can profit from collaborative mapping and localization as different viewpoints and sensor modalities can be combined into a joint map. The achieved synergetic effects can be diverse for instance extending the range of perception [65].

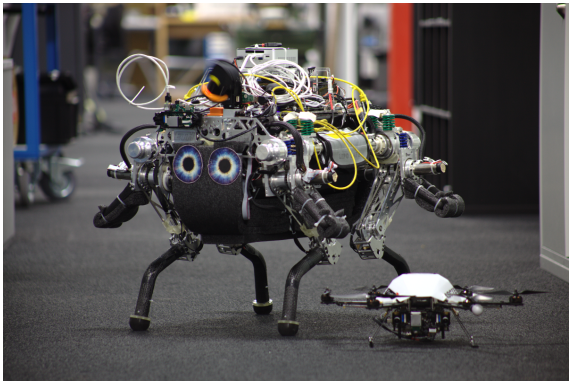


Figure 4.1: Team of heterogeneous robots running the visual-inertial estimator in real-time to localize against a highly compact localization summary map. The evaluations show more accurate and smooth global pose estimates with a tight integration of 2d-3d localization matches against a map, visual feature-tracks and inertial measurements.

Fusing camera and inertial sensor measurements has proven to yield highly accurate motion estimates [41, 54]. Therefore, such visual-inertial odometry (VIO) systems have become a common choice for robotic navigation. In parallel, the research community has developed high performance algorithms for loop-closure [16, 58] and vision-based localization against a known 3d-model [88, 100, 101]. Furthermore, variants for real-time operation with [66] and without [59] the need for a server connection have been demonstrated. Combining a visual SLAM

system with efficient large-scale localization [59, 66] provides a compelling approach to localize multiple agents in a common frame of reference. These methods localize in maps consisting of 3d-points and descriptors representing the appearance of these landmarks from observing keyframes. These localization maps are often built from multiple VIO trajectories that are merged into a global map and refined using visual-inertial weighted least-squares [26]. Several approaches have been described for merging these sub-maps in a subsequent post-processing step [31, 38, 86], which may also incorporate additional sensors [101] besides vision.

Localization of multiple-robots in a common frame-of-reference relies on a compact representation of the localization map. This is important to keep the memory requirement to a minimum and thus allow for low transmission times when sharing maps with other agents. Recently, several methods were presented to select and only retain the most informative localization landmarks of a map [20, 72, 94]. Such methods usually guarantee full localization coverage at a minimal count of localization landmarks. The number of 2d-3d matches available for localization, however, will drop and lead to a decrease in smoothness and accuracy of the global pose estimates.

In contrast to state of the art approaches [66, 78], we aim at a tight integration of inertial measurements with visual feature-tracks and 2d-3d constraints to the global localization map. For this reason, we use a fixed-lag-smoother as our state estimation framework instead of the filter formulation proposed in [59]. The proposed joint estimation of local VIO and global pose improves the smoothness and accuracy of the global pose when localizing against such highly compact maps. To sparsify landmarks and thus reduce the map size, we employ the method of map summarization described in our previous work [20]. This method only retains the most informative landmarks from a map that was initially built from several VIO trajectories and refined with a visual-inertial least squares optimization. During runtime the system concurrently performs localization to the known reference model and visual-odometry in yet unmapped areas. This allows both having an accurate estimate w.r.t. other agents and also mapping previously unvisited areas.

The contributions of this paper are:

- a localization estimator that tightly integrates inertial measurements and visual feature tracks with 2d-3d matches for a global pose w.r.t. a summary map and a local VIO pose estimate,
- an evaluation of the localization error against motion-capture data and a related loosely-coupled approach,
- a demonstration of the improvements in smoothness and accuracy of the global poses estimated in highly compact maps,
- and a validation of the summary map concept, presented in our previous work [20], against absolute ground-truth data in a real-world scenario with a team of heterogeneous robots shown in Fig. 4.1.

The remainder of the paper is structured as follows. Section 2 describes the mapping process that is used to create the compact localization summary maps from several VIO trajectories. Section 3 introduces the visual-inertial localization that tightly integrates visual feature tracks

and inertial data with 2d-3d map matches into a global pose estimate. Finally, Section 4 evaluates the localization error against ground-truth from a motion-capture system and compares it against a related loosely-coupled approach.

2 Mapping Backend

A global localization map is created by, first, running several mapping sessions using visual-inertial odometry to create several local sub-maps. This is carried out using the proposed Visual-Inertial Localization (VIL) system from Section 3 without a reference map. Next, the resulting sub-maps are merged into one global map using appearance-based matching followed by a refinement using a visual-inertial least-squares minimization. And finally, the global map is summarized to only retain the most-informative landmarks for localization. The VIL estimator can then be used to localize against the resulting summary map in real-time. An overview of this process is shown in Fig. 4.3.

2.1 Map Representation

The map consists of several sub-maps, so-called *missions* that each represent a separate agent trajectory. Every mission has its own local frame of reference M_i. These frames are anchored w.r.t. the global frame of reference by a transformation T_{GM_i}. By introducing such additional frames of reference, we can align individual trajectories without modifying keyframe and landmark positions. The transformations T_{GM_i} are estimated as part of a least-squares minimization involving loop-closure constraints to other missions and/or GPS signals.

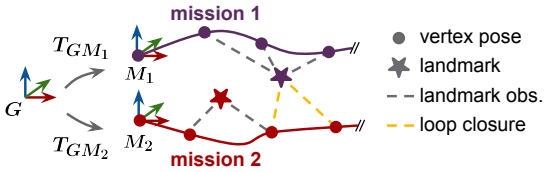


Figure 4.2: The map structure consists of sub-maps (missions), base-frames M_i, keyframes (vertices), landmarks with observation and loop-closure constraints.

Each mission stores a graph of keyframes and a set of corresponding visual landmarks. The vertices of these graphs correspond to keyframes containing visual measurements, the local pose T_{M_iB}, and the inertial states (velocity and Inertial Measurement Unit (IMU) biases). These keyframes are connected by edges holding IMU measurements. Landmarks are observed from and thus associated with multiple keyframes which are part of one mission or in case of loop-closures from multiple missions.

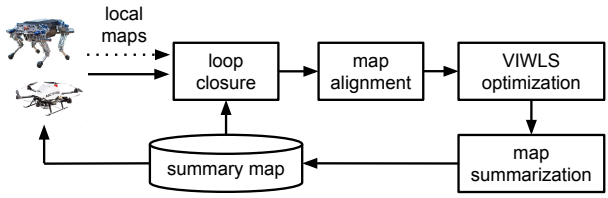


Figure 4.3: Overview of the mapping process: (i) multiple local maps from VIO are loop-closed, (ii) aligned w.r.t. each other, (iii) refined in a visual-inertial least-squares optimization and (iv) finally summarized to yield a compact localization map.

2.2 Merging of Local Maps

In order to create a global localization map we merge several local VIO maps that have been obtained by running the visual-inertial localization estimator without a reference map (Section 3). This is achieved by loop-closing all individual trajectories and performing an alignment based on these constraints. We utilize an implementation of the visual descriptor based loop-closure system described in [58] that associates 2d-image descriptors and 3d-points in the maps. First, the high dimensional binary BRISK descriptors [51] are projected to a lower dimensional real-valued space (here: 10 dimensions) and then inserted into an index. Depending on the map size this is either formed by a KD-tree [24] or a multi-dimensional product vocabulary [4] augmented by KD-trees as proposed in [58]. The raw 2d-3d matches are first filtered using a covisibility graph [89] – an approximate set-cover problem is solved in which only 3d landmarks that form a cluster in the covisibility graph are returned. The matches from the dominant cluster are passed to a Perspective-n-Point (PnP) solver in a RANSAC loop to ensure geometric consistency. Once loop-closure correspondences are established an alignment transformation T_{GM_i} for each mission M_i is estimated. Therefore, a least-squares problem is solved using all loop-closures correspondences as constraints. Landmarks are merged if the verified matches indicate that two or more landmarks are actually the same physical landmark. After this process, a joint visual-inertial least-squares is solved to further refine the global map.

2.3 Visual-inertial Weighted Least-Squares

A non-linear visual-inertial least-squares optimization (VIWLS) problem is solved to obtain a consistent global map. We use the Ceres solver [1] to minimize the cost $J(x)$ comprised of visual and inertial error terms. Visual error terms penalize the reprojection error, i.e., the image plane distance between the reprojected 3d landmark position and the measured keypoint location. Inertial error terms penalize the temporal error between two vertices; that is the difference between states of the two vertices and the integrated IMU measurement. The optimization

objective is then given by:

$$J(x) = \sum_{i=1}^{N} \sum_{j=1}^{n(i)} \sum_{k \in \mathcal{K}(i)} \mathbf{e}_r^{i,j,k^T} \mathbf{W}_r^{i,j,k} \mathbf{e}_r^{i,j,k} + \sum_{i=1}^{N-1} \mathbf{e}_s^{i^T} \mathbf{W}_s^i \mathbf{e}_s^i \tag{4.1}$$

where N denotes the number of keyframes, $n(i)$ the number of cameras for the i-th keyframe, $\mathcal{K}(i)$ the set of landmarks visible in camera j of keyframe i, $\mathbf{e}_r^{i,j,k}$ the reprojection error of landmark k in camera j of keyframe i and $\mathbf{e}_s^i$ the temporal IMU error between keyframe i and $i+1$. Terms $\mathbf{W}_r^{i,j,k}$ and $\mathbf{W}_s^i$ denote the weighting information matrices calculated as the inverse of the covariance matrices: keypoint measurement and IMU integration covariance respectively.

2.4 Map Summarization

In order to reduce the size of the global localization map we apply a compression and data selection step called *summarization*. This describes the process of pruning as many landmarks from the map as possible while ensuring sufficient localization coverage over the entire mapped area. Landmark removal is vital, as their 3d positions and especially descriptors constitute the majority of the map size.

To obtain the best possible performance while keeping the computational effort limited, an Integer Linear Programming (ILP) approach is used to select the subset of landmarks which are most informative for localization [20]. The resulting optimization problem can be defined as:

$$\begin{aligned} \min \ & \mathbf{q}^T \mathbf{x} + \lambda \mathbf{1}^T \boldsymbol{\zeta} \\ \text{s.t.} \ & \mathbf{A}\mathbf{x} + \boldsymbol{\zeta} \geq b\mathbf{1}, \qquad \boldsymbol{\zeta} \in \{\{0\} \cup \mathbb{Z}^+\}^M \\ & \sum_{i=1}^{N} \mathbf{x}_i = n_{desired}, \quad \mathbf{x} \in \{0,1\}^N \end{aligned} \tag{4.2}$$

where N is the initial number of landmarks, M is the total number of keyframes, x is a vector of binary switch variables associated with landmarks (1 means the landmark should be retained, 0 otherwise), $\mathbf{q}$ is a vector of scores associated with landmarks (based on the number of observations and the stability of descriptors), $\mathbf{A}$ is a $M \times N$ visibility matrix, b is setting a keypoints-per-keyframe threshold, ζ is a slack variable and $n_{desired}$ is the desired number of retained landmarks. This process selects the subset of $n_{desired}$ landmarks with highest scores, while ensuring localizability of each keyframe, modelled as heuristic constraints (i.e. keyframe observes at least $b - \zeta$ landmarks).

3 Visual-inertial Localization

Due to highly compact maps, often, there is only a very limited number of 2d-3d matches to the localization map available. A global pose estimator solely based on scarce 2d-3d matches would suffer from non-smooth and less accurate estimates. Therefore, we propose to augment the global pose estimation with visual feature tracks and inertial data in the framework of a sliding-window based optimization. This joint estimation improves accuracy and noise characteristics of the global pose when used with highly compact localization maps. Moreover, such a joint estimation of the local VIO and the global pose allows for a seamless switching between localization and exploration mode.

First, we introduce the vision frontend that detects and tracks point-features from image-to-image. Second, we describe the matching stage which establishes correspondences between visual feature tracks and map landmarks. Last, we present the visual-inertial estimator which is used to jointly estimate the robot motion in a local and global frame of reference.

3.1 Vision Frontend

The vision frontend establishes 2d-2d correspondences of keypoints over time. Therefore, we detect AGAST [60] keypoints and assign them to bins in a uniform grid on the image plane. In each bin we only retain the N strongest keypoints according to their detector response. This enforces minimal feature coverage across the image and ensures a good sampling of the observed structure. It is important to note that localization and ego-motion estimation have somewhat different requirements regarding feature selection. The point-based localization relies on a high repeatability of the feature detections whereas the ego-motion estimation requires a uniform sampling of the observed structure. For this reason, a very coarse grid is used when selecting new keypoints for tracking.

The remaining keypoints are tracked between consecutive images using the Lucas-Kanade [9] method. The integrated angular measurements from the gyroscopes are used to predict the keypoint locations between consecutive frames to facilitate and robustify the data association. Finally, a 2-point RANSAC scheme is used to detect and reject outlier matches.

3.2 Matching Feature Tracks to the Map

All terminated feature tracks (or track above a certain length) are matched against the localization map. In order to find correspondences between 2d observations of the feature tracks to map landmarks, we use an implementation of the method proposed in [59]. To allow matching against a large map, we build a multi-dimensional (product) vocabulary [4] with KD-tree augmentation as an index of all descriptors of the localization map. All descriptors of the feature track are queried for its nearest neighbors to identify landmarks with similar descriptors. These raw matches are clustered based on landmark covisibility [89]. The biggest cluster is then passed to a PnP solver in a RANSAC loop to identify the consistent subset of matches. Finally, the orientation of the recovered camera pose w.r.t. gravity is compared to the current estimate of the VIO. As the localization maps are gravity-aligned, any RANSAC result with a gravity alignment error above a predefined threshold (here: 5 deg) is rejected. A single matched observation of a feature track is used to associate the entire track with this map landmark. Further, the

map alignment T_{GM} obtained in the RANSAC step is used to initialize its linearisation point in the non-linear optimization.

3.3 Sliding-window Optimization

To allow real-time operation on a robotic platform we solve an approximation of the visual-inertial SLAM problem posed as a non-linear fixed-lag smoother. Therefore, only a fixed number of the most recent visual-inertial keyframes (here: 5) is kept in the optimization window. Older keyframes are marginalized as they are pushed out of the window by new keyframes. The problem is formulated as a factor-graph with the assumption of Gaussian noise. We use GTSAM for solving the resulting non-linear least-squares problem and to marginalize-out old states [17].

At each update, the state is augmented with a new keyframe containing its pose T_{MB}, velocity v and IMU bias b. The pose is initialized by propagating the pose of last the keyframes using the integrated inertial measurements. We establish an inertial constraint between the current and the last keyframe using the formulation of [33].

All terminated feature tracks obtained by the vision frontend (Section 3.1) are separated into odometry and localization tracks. Localization tracks are the feature tracks that were successfully matched to a map landmark (Section 3.2). Processing terminated feature tracks ensures good constraints between all keyframes in the window while still being able to sub-select the set of tracks used for the update. To maintain a bounded and nearly constant computational complexity only a limited number of the available tracks are used for the update (here: 50). We use a heuristic score for this selection based on the distance to the triangulated landmark positions and the disparity angle spanned by all observation rays to the landmark.

Processing odometry tracks: A new landmark is initialized for each odometry track by triangulating its position ${}_M l$ (Fig. 4.4). All keypoint measurements of the track are added to the optimization to form constraints between the observing keyframes and the landmark. We use a weighted re-projection error $\mathbf{e_o}$ that takes the following form in the non-linear least-squares problem:

$$\mathbf{e}_o \left(T_{MB}, {}_M l \right) = \frac{1}{\sigma_o} \left[f_{\text{proj}} \left(T_{MB}^T \cdot {}_M l \right) - z_i \right] \tag{4.3}$$

where T_{MB} is the local pose of the body frame B expressed in the mission frame of reference M, ${}_M l$ is the landmark position expressed in the mission frame M, f_{proj} is the non-linear function projecting a 3d point in the camera frame onto the image plane, z_i is the keypoint measurement on the image plane, and σ_o the keypoint measurement uncertainty. Additionally, this error is weighted by a Huber loss-function to achieve robustness against errors in tracking and data association.

Processing localization tracks: Each localization track has an associated map landmark that is used to formulate a re-projection error similar to the one used for the odometry tracks but with a known landmark position ${}_G l$. We assume a constant and isotropic measurement noise for these re-projection errors (here: 1.0 px). The error $\mathbf{e}_l$ for one observation can be written as:

$$\mathbf{e}_l \left(T_{GM}, T_{MB} \right) = \frac{1}{\sigma_l} \left[f_{\text{proj}} \left(T_{MB}^T \cdot T_{GM}^T \cdot {}_G l \right) - z_i \right] \tag{4.4}$$

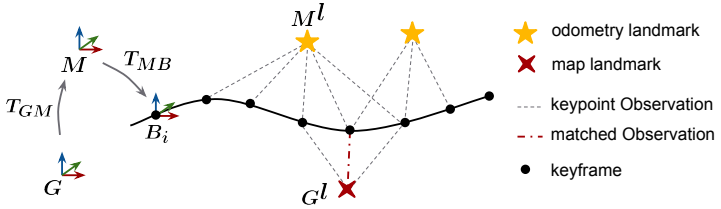


Figure 4.4: Frames of reference in visual-inertial localization: (i) T_{GM}: alignment of the local mission frame M to the global localization map frame G, (ii) T_{MB}: local pose estimate in mission frame M, (iii) ${}_M l$ odometry landmark in mission frame and (iv) ${}_G l$ map landmark in global frame G.

where T_{GM} is the transformation that aligns the local mission frame M with the global map frame G, ${}_G l$ is the position of the associated map landmark and all other terms are the same as in 4.3. Note that this error term constrains the transformation T_{GM} and T_{MB}; whereas the error term for odometry observations (4.3) only constrains the local pose T_{MB}. This allows for a tightly-coupled estimation of the local pose T_{MB} and the global pose T_{GB} that is given by $T_{GB} = T_{GM} \cdot T_{MB}$. Special care must be taken to avoid re-use of information. To this end, we need to make sure that each keypoint measurement and map landmark is not processed more than once in a localization update.

The total cost $J(x)$ of the non-linear least squares problem can be written as:

$$J(x) = \sum_{i=1}^{N} \sum_{j=1}^{n(i)} \left[\sum_{k \in \mathcal{K}(i)} \mathbf{e}_o^{i,j,k^T} \mathbf{W}_o^{i,j,k} \mathbf{e}_o^{i,j,k} + \sum_{m \in \mathcal{M}(i)} \mathbf{e}_l^{i,j,m^T} \mathbf{W}_l^{i,j,m} \mathbf{e}_l^{i,j,m} \right] + \sum_{i=1}^{N-1} \mathbf{e}_s^{i^T} \mathbf{W}_s^i \mathbf{e}_s^i \tag{4.5}$$

where N denotes the number of keyframes in the window, $n(i)$ the number of cameras for the i-th keyframe, $\mathcal{K}(i)$ the set of odometry, $\mathcal{M}(i)$ the set of localization landmarks visible in camera j of keyframe i, $\mathbf{e}_o^{i,j,k}$ the reprojection error of odometry landmark k in camera j of keyframe i, analogously $\mathbf{e}_l^{i,j,m}$ the reprojection error of the localization landmark m and $\mathbf{e}_s^i$ the temporal IMU error between keyframe i and $i+1$. The terms $\mathbf{W}_o^{i,j,k}$, $\mathbf{W}_l^{i,j,k}$, $\mathbf{W}_s^i$ denote the inverse of the corresponding measurement covariance matrices.

Processed odometry and localization landmarks get marginalized after each update of the factor graph and keyframes once they are pushed out of the window by new keyframes. After marginalization an additional linear term is introduced to 4.5 to account for the influence of the removed states at the time of marginalization.

3.4 Handling of Degenerate Motion

Since the estimator processes terminated feature tracks, (quasi-)stationary motions need to be handled explicitly as no or very few tracks terminate during such phases. Such phases are detected by heuristics based on inertial measurements, the output of the 2-pt RANSAC used in the vision frontend and the method from [49]. An artificial zero-velocity measurement is added to the inertial states if rotation-only motion was detected during that time. Additionally, a small number of landmarks (here: 15) remain in the state for multiple estimator updates; whereas all other landmarks are marginalized after each update. These local SLAM landmarks are only marginalized once tracking is lost. This allows for a stable ego-motion estimation during slow motion and transitions to and from stationary phases.

4 Experimental Evaluation

We evaluate the presented visual-inertial localization method in a collaborative robot scenario involving a MAV and an Unmanned Ground Vehicle (UGV) [45]. Both platforms are equipped with a VI-Sensor [74], a sensor system containing an IMU and two global shutter cameras.

(a) Asctec Firefly Hexacopter

(b) StarlETH Quadruped [45]

Figure 4.5: Robot platforms used to collect the evaluation datasets.

For simplicity both platforms were manually controlled to follow a path through an indoor environment while recording visual and inertial data. A motion capture system was used to obtain accurate ground-truth data for position and orientation. The poses from the motion capture system were spatially and temporally aligned with the inertial data of the robots using a full-batch maximum-likelihood estimator instead of the filter formulation of [55].

Additionally, our previous work on map summarization [20] is validated under real-world conditions with a team of heterogeneous robots. Furthermore, the influence of parameters in the map summarization is investigated as a trade-off between map-size and localization accuracy.

4.1 Absolute Localization Error

Here, we evaluate the absolute localization error of the presented visual-inertial localization estimator (VIL) against a related sliding-window localization method (SWL) [78]. The SWL

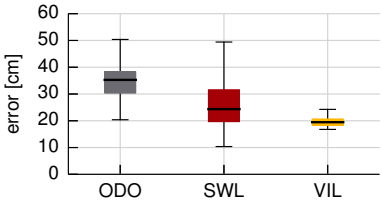


Figure 4.6: Median and standard deviation of the absolute localization error are lower for the visual-inertial estimator (VIL), as compared to the sliding-window localization (SWL), and the drifting visual-inertial odometry (ODO) with initial alignment.

method estimates a map alignment transformation over a sliding-window of recent keyframes and associated map landmarks. The keyframe poses and landmark positions are fixed and only the global map alignment is estimated. The VIO estimates (by running the VIL without a map) are used to forward propagate the state from the most recent localization estimate. Therefore, the SWL is a method that utilizes the same information as the VIL (inertial data, feature tracks and 2d-3d matches) but in a loosely-coupled formulation.

In this scenario, a single indoor MAV mission is used to build a map of the environment. Subsequently, the UGV localizes against this map built by the MAV using its on-board sensors. The global position estimates of both methods are evaluated against the motion tracking ground-truth.

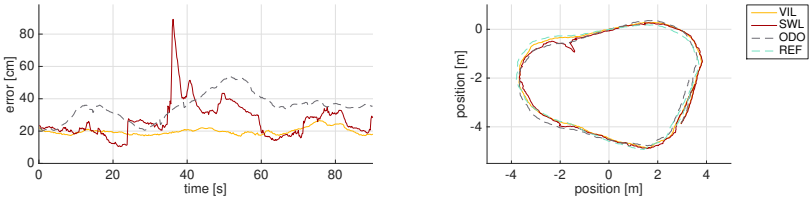


Figure 4.7: The absolute localization error (left) w.r.t. ground-truth is shown when localizing a UGV mission against a map built from MAV data. This experiment shows an improvement in smoothness and accuracy of the visual-inertial localization (VIL) achieved by jointly estimating the VIO and global poses whereas the sliding-window localization (SWL) and pure visual-inertial odometry (ODO) yield higher variance and mean error. The global poses (left) are compared against ground-truth from motion tracking (REF).

Fig. 4.6 and Fig. 4.7 show the error and statistics of the global pose estimates and demonstrate the robustness of the presented VIL method. It can be seen that a joint estimation of VIO and global pose yields a smoother and more accurate estimate as compared to the loosely-coupled

method.

4.2 Influence of Map Summarization

This analysis investigates the influence of the localization map size on the global localization error. The setup aims at simulating a typical collaborative mapping and localization scenario between an MAV and a UGV. First, six local maps are built from independent MAV missions using the visual-inertial localization estimator (without a reference map). In a next step, all local maps are merged into a single global map and refined in a visual-inertial least-squares optimization. The resulting global map is then summarized multiple times with increasing levels of landmark removal. A separate UGV dataset is then used to localize against these summarized maps and evaluate the resulting absolute global localization error w.r.t. the motion tracking ground-truth. This is an ideal scenario to assess, not only the visual-inertial localization performance, but also the concept as a whole as it involves all components: local map creation using VIO, map merging, map summarization and finally localization in the global map using a different robotic platform.

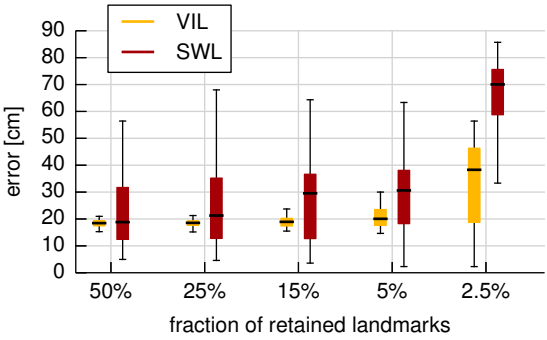


Figure 4.8: Statistics of the absolute localization error for different levels of map summarization. The landmarks are reduced to a varying fraction of the initially $200'000$ landmarks. The accuracy and smoothness of the VIL estimates remains almost constant up to a summarization level of 5% whereas the SWL shows a significant increase of the median error starting at 25%.

Fig. 4.8 and Table 4.1 show that roughly 90 percent of all landmarks can be pruned from the localization map with only a minimal increase of the localization error when using the proposed visual-inertial localization (VIL) method. The loosely-coupled method (SWL), however, shows a significant increase of the mean and the variance of the localization error at the same summarization levels. This study demonstrates the benefit of the proposed joint estimation of VIO and the global pose when used with highly compact maps. Furthermore, it justifies a more aggressive map summarization for the VIL method and thus allows for more compact localization

	fraction of initial landmark count				
[cm]	**50%**	**25%**	**15%**	**5%**	**2.5%**
VIL					
median	18.46	18.55	18.93	20.05	38.29
std.	1.28	1.23	1.81	3.57	13.89
SWL					
median	18.83	21.28	29.54	30.61	70.04
std.	14.48	14.37	13.48	12.39	18.96

Table 4.1: Absolute localization error statistics for the experiment of Section 4.2 for the visual-inertial estimator (VIL) and sliding-window localization (SWL).

maps while retaining the robustness and accuracy of larger maps.

4.3 Timings of the VIL-Estimator

Timings are given in Table 4.2 for the UGV dataset of Section 4.2 using a localization map containing $30'000$ landmarks. Both methods have been evaluated on an Intel *i7-3740QM* using 2 cores. The sliding-window localization (SWL) runtime includes matching against the map, global pose estimation and visual-inertial odometry (ODO) to propagate global poses between localizations. The visual-inertial estimator (VIL) contains only map matching and estimation as its tightly-coupled formulation jointly estimates local and global pose. The proposed VIL exhibits very similar run-times compared to the SWL approach.

[ms]	**VIL**	**SWL**	**ODO**
map query			
matching	7.38 ± 1.26		-
RANSAC	2.53 ± 2.90		-
estimator			
localization	36.89 ± 9.34	10.29 ± 12.87	-
odometry		34.72 ± 8.22	34.72 ± 8.22
complete update	$\mathbf{46.80 \pm 9.86}$	$\mathbf{54.92 \pm 15.59}$	$\mathbf{34.72 \pm 8.22}$

Table 4.2: Processing time for one update profiled on an i7-3740QM using 2 cores. The joint optimization of VIO and localization leads to a lower run-time in the VIL as compared to the loosely-coupled SWL.

5 Conclusions

In this work, we presented a real-time visual-inertial localization method that is particularly suitable for use with highly compact localization maps. Although, the employed map summa-

rization process guarantees a good localization coverage over the mapped space, it generally reduces the number of potential 2d-3d matches during localization. Commonly, a less smooth global pose estimate is expected if it is solely based on these matches. Therefore, we augment the localization problem (based on 2d-3d map matches) with the additional information from visual feature tracks and inertial measurements. The proposed formulation as a sliding-window based optimization jointly estimates the local VIO and the global pose in the map's frame of reference. Moreover, it is worth noting that this leads to a seamless switching between localization and exploration mode.

A series of experiments with a team of heterogeneous robots validate the proposed visual-inertial localization (VIL) and the concept of summary maps as a whole. When comparing the VIL against a related method that loosely integrates 6-Degrees-of-Freedom (DoF) localizations with VIO poses, we can show that in our method smoothness and accuracy of the global pose estimates remain nearly unaffected up to a high level of summarization. The compared loosely-coupled method, however, does not exhibit the same robustness. Thus, the proposed method can tolerate a more aggressive summarization while still maintaining nearly the same performance.

For future work, we plan to perform experiments at a larger scale and investigate possible benefits arising from using projected landmark uncertainties in the localization.

Acknowledgements

The authors thank Samuel Bachmann, Michael Burri, Remo Diethelm and Péter Fankhauser for their help with the experiments and Janosch Nikolic for tools to process the motion capture data. The research leading to these results has received funding from Google's Project Tango.

Paper II

maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization

Thomas Schneider*, Marcin Dymczyk*, Marius Fehr*, Kevin Egger, Simon Lynen, Igor Gilitschenski, Roland Siegwart
(* contributed equally)

Abstract

Robust and accurate visual-inertial estimation is crucial to many of today's challenges in robotics. Being able to localize against a prior map and obtain accurate and drift-free pose estimates can push the applicability of such systems even further. Most of the currently available solutions, however, either focus on a single session use case, lack localization capabilities, or don't provide an end-to-end pipeline. We believe that only a complete system, combining state-of-the-art algorithms, scalable multi-session mapping tools, and a flexible user interface, can become an efficient research platform. We, therefore, present maplab, an open, research-oriented visual-inertial mapping framework for processing and manipulating multi-session maps, written in C++. On the one hand, maplab can be seen as a ready-to-use visual-inertial mapping and localization system. On the other hand, maplab provides the research community with a collection of multi-session mapping tools that include map merging, visual-inertial batch optimization, and loop closure. Furthermore, it includes an online frontend that can create visual-inertial maps and also track a global drift-free pose within a localization map. In this paper, we present the system architecture, five use cases, and evaluations of the system on public datasets. The source code of maplab is freely available for the benefit of the robotics research community.

Published in:
IEEE Robotics and Automation Letters (Volume: 3, Issue: 3, July 2018), 2018
10.1109/LRA.2018.2800113

1 Introduction

The ever growing deployment of simultaneous localization and mapping (SLAM) systems poses novel challenges for the robotics community. Availability of precise, drift-free pose estimates both outdoors and indoors has become a vital requirement of numerous robotics applications, such as navigation or manipulation. The increasing popularity of visual-inertial estimation systems created a strong incentive to improve their robustness to viewpoint and appearance changes (daylight, weather, seasons, etc.) or rapid motion. Current research efforts aim to collect data using heterogeneous agents, build maps of larger scale, cover various visual appearance conditions and maintain maps over a long time horizon. Investigating these and many related challenges requires a multi-session end-to-end mapping system that can be easily deployed on various robotic platforms and provides ready-to-use algorithms with state-of-the-art performance. At the same time it needs to offer high flexibility necessary for conducting research.

Most openly available frameworks for visual and visual-inertial Simultaneous Localization and Mapping (SLAM) either focus on a single-session case [52] or only provide large-scale batch optimization without an online frontend [93]. Usually, they are crafted for a very specific pipeline without a separation between the map structure and algorithms. They often lack completeness and will not offer a full workflow such that a map can be created, manipulated, merged with previous sessions and reused in the frontend within a single framework. This impairs the flexibility of such systems, a key for rapid development and research.

This work addresses this problem by introducing maplab[1], an open visual-inertial mapping framework, written in C++. In contrast to existing visual-inertial SLAM systems, maplab does not only provide tools to create and localize from visual-inertial maps but also provides map maintenance and processing capabilities. These capabilities are offered as a set of tools accessible in a convenient console that can easily be extended through a plugin system. These tools involve multi-session merging, sparsification, loop closing, dense reconstruction and visualization of maps. Additionally, maplab includes ROVIOLI (ROVIO with Localization Integration), a mapping and localization frontend based on ROVIO [7], a patch-based visual-inertial odometry system.

Maplab has been extensively field tested and has been deployed on a variety of robotic platforms including micro aerial vehicles [10], autonomous planes [42, 43], autonomous cars [12], autonomous underwater vehicles [97], and walking robots [27]. It has also served as a research platform for map summarization [19–22], map quality evaluation [64], multi-session 3d reconstruction [28], topological mapping [6], visual localization [36, 58, 78], and decentralized mapping [15].

To the best of our knowledge, maplab is the first visual-inertial mapping framework that integrates a wide variety of use cases within a single system. Maplab is free, open-source, and has already proved to be of great use for various research and industry projects. We strongly believe that the robotics community will harness it both as an off-the-shelf mapping and localization solution, as well as a mapping research testbed. The contributions of this work can be summarized as follows:

- it introduces a general purpose visual-inertial mapping framework using feature-based maps with multi-session support;

[1] Maplab is available at: www.github.com/ethz-asl/maplab

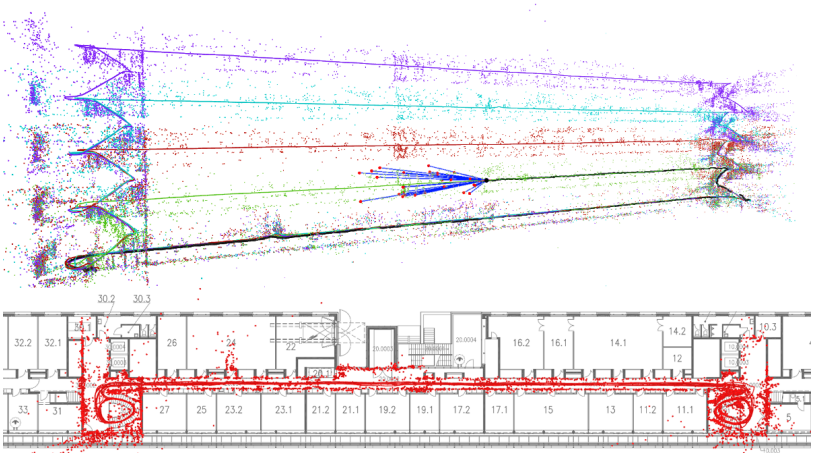


Figure 5.1: The maplab framework can build consistent visual-inertial maps from multiple mapping sessions. Here, 4 separate sessions are merged and jointly refined. The global map can then be used by odometry and localization frontend to correct for any drift when revisiting the area. The floorplan is overlayed with the landmarks of all floors demonstrating the accuracy and consistency of the map alignment.

- it introduces ROVIOLI, a robust visual-inertial estimator tightly coupled with a localization system;
- it presents examples of algorithms and data structures for modifying and maintaining maps including map merging, sparsification, place recognition, and visualization;
- it highlights the extensibility of the system that makes it well suited for research;
- it provides evaluation of selected components of the framework.

2 Related Work

There are several openly available visual and visual-inertial SLAM systems. One of the earliest examples is PTAM [48], a lightweight approach for mapping and tracking a local map in parallel. It was originally developed for augmented reality applications so it offers neither large-scale localization nor any offline processing tools. More recent examples include OKVIS [52], a visual-inertial keyframe-based estimator. This approach tracks a local map built from recently acquired keyframes, which minimizes the drift locally. Similarly, semi-dense [32] and

dense [25] odometry frameworks achieve high-quality pose estimates by using photometric error formulations instead of feature-based matching. None of these methods, however, supports global localization against a previously recorded map.

ORB-SLAM [71] and ORB-SLAM2 [70] are vision-based frameworks that offer the possibility to create a map of the environment and then reuse it in a consecutive session, which closely relates to the workflow we propose here. In contrast to these systems, maplab offers an offline processing toolkit centered around a console user interface, which guarantees high flexibility and permits users to add their own extensions or modify the processing pipelines. We consider the ability to merge multiple mapping sessions into a single, consistent map and to refine it using a visual-inertial least-squares optimization a core capability of maplab that differentiates it from ORB-SLAM. Another difference worth emphasizing is the online frontend of maplab, ROVIOLI. Using image intensity within patches instead of point features guarantees a high level of robustness, even in the presence of motion blur [7].

Incorporating the capability to process multiple maps has received considerable attention in the SLAM research community with [8] being one of the earliest works incorporating multiple maps in a hybrid metric-topological approach to multi-session mapping. Use of anchor nodes to stitch together posegraphs from multiple mapping sessions is proposed in [63]. Trying to establish topological associations between maps is also proposed in [14], where maps are stored as a set of experiences. In contrast, maplab stores a unified localization map allowing to use a carefully selected subset of features, e.g. based on the current appearance conditions [12].

Systems that aim to reconstruct the 3d structure from large collections of unordered images [68, 93, 96] also contain functionalities similar to maplab. They typically offer efficient implementations of large-scale bundle adjustment optimization and advanced image and feature matching techniques. They lack, however, algorithms that process inertial data and cannot be run directly on a robotic platform in order to provide pose estimates online.

3 The maplab Framework

From the user perspective, the framework consists of two major parts:

i. The online **VIO (Visual Inertial Odometry) and localization frontend**, ROVIOLI, that takes raw visual-inertial sensor data. It outputs (global) pose estimates and can be used to build visual-inertial maps.

ii. The (offline) **maplab-console** that lets the user apply various algorithms on maps in an offline batch fashion. It does also serve as a research testbed for new algorithms that operate on visual-inertial data.

The maplab framework follows an extensible and modular design. All software components are organized in packages, which are built using catkin, the official build system of ROS [83]. The C++11 standard is used throughout the framework and third-party dependencies are limited to popular and well-maintained libraries, among others Eigen [37] for linear algebra and Ceres [1] for non-linear optimization. Additionally, the framework provides ROS interfaces to conveniently input raw sensor data and output the results, such as pose estimates for an easy deployment on a robotic systems. The framework uses RViz as a 3d visualization tool to both

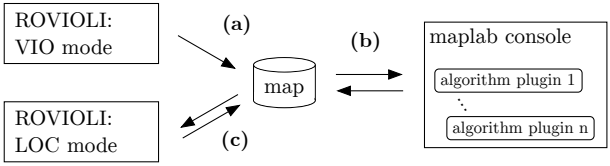


Figure 5.2: Typical workflow in maplab: (a) In VIO mode, ROVIOLI estimates the pose of an agent w.r.t. a (drifting) local frame; additionally a map is built based on these estimates. (b) Resulting maps can be loaded in the maplab-console where all of the available algorithms can be applied, e.g. map alignment and merging, VI optimization, loop closure. (c) In LOC mode, ROVIOLI can load the updated map to track a global (drift-free) pose online.

visualize the state of the online mapping algorithms and the results of the offline processing from the maplab console.

3.1 Notation

Throughout this document and the source-code, we use the notation as defined in this section. A transformation matrix $\mathbf{T}_{AB} \in \mathrm{SE}(3)$ takes a vector ${}_B\mathbf{p} \in \mathbb{R}^3$ from the frame of reference $\mathcal{F}_B$ to the frame of reference $\mathcal{F}_A$. It can be partitioned into a rotation matrix $\mathbf{R}_{AB} \in \mathrm{SO}(3)$ and a translation vector ${}_A\mathbf{p}_{AB} \in \mathbb{R}^3$ as:

$$\begin{bmatrix} {}_A\mathbf{p} \\ 1 \end{bmatrix} = \mathbf{T}_{AB} \cdot \begin{bmatrix} {}_B\mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{AB} & {}_A\mathbf{p}_{AB} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} {}_B\mathbf{p} \\ 1 \end{bmatrix} \tag{5.1}$$

The operator $\mathbf{T}_{AB}(\cdot)$ is defined to transform a vector in $\mathbb{R}^3$ from $\mathcal{F}_B$ to the frame of reference $\mathcal{F}_A$ as ${}_A\mathbf{p} = \mathbf{T}_{AB}\,({}_B\mathbf{p})$ according to 5.1.

3.2 Workflow for multi-session mapping and localization

The typical workflow for a mapping and localization session within the maplab system is illustrated in Fig. 5.2. Often, it is beneficial to build a single localization map from multiple mapping sessions to ensure a good spatial and temporal (i.e. different appearances) coverage of the area. An initial, open loop map is built in each session using ROVIOLI in visual-inertial odometry (VIO) mode and stored to disk. The maps can then be refined using various (offline) tools such as loop closure detection, visual-inertial optimization or co-registration of multiple sessions (map merging). Detailed inspection of the maps is possible using a large set of different visualizations, statistics and queries. More advanced modules allow, e.g., to create a dense representation (Truncated Signed Distance Field (TSDF), occupancy, etc.) of the environment using data from a depth sensor or from stereo.

The resulting (multi-session) map can then be exported as a compact localization map and used by ROVIOLI (in LOC mode) for online localization during a second visit to the same place. Continuous online localization enables accurate tracking of a global pose w.r.t. a known 3d structure and thus compensates for drift in the visual-inertial state estimation.

3.3 maplab Console: The Offline User Interface

The maplab framework uses a console user interface to manipulate maps offline. Multiple maps can be loaded into the console simultaneously, facilitating multi-session mapping experiments. All algorithms are available through console commands and can be applied to the loaded maps. Parameters specific to each algorithm are set by console flags or a flag file and can be modified at runtime. Combined with the real-time visualization of the map in RViz, this greatly facilitates algorithm prototyping and parameter tuning. It is possible to combine multiple algorithms and experiment with entire processing pipelines. Changes can be easily reverted by saving and reloading intermediate states of a map from disk.

The console uses a plugin architecture[1] and automatically detects all available plugins within the build workspace at run time. Therefore, the integration of a new algorithm or functionality is possible without any changes to the core packages. For algorithms that operate on the standard visual-inertial map datatype (see Section 3.4), no interfacing work will be necessary.

3.4 Map Structure

The framework uses a data structure, called VI-map, for visual-inertial mapping data. The VI-map contains the raw measurements of all sensors and a sparse reconstruction of the covered environment. Each map may contain multiple *missions* where each is based on a single recording session. The core structure of a mission is a graph consisting of *vertices* and *edges*. A *vertex* corresponds to a state captured at a certain point in time. It contains a state estimate (pose $\mathbf{T}_{MI_k}$, Inertial Measurement Unit (IMU) biases, velocity) and visual information from the (multi-) camera system including keypoints, descriptors (BRISK [51] or FREAK [2]), tracking information and images. An *edge* connects two neighboring vertices. While there are a few different types of edges in maplab, the most common type is the IMU edge. It contains the inertial measurements recorded between the vertices that the edge connects. Visual observations tracked by multiple vertices are triangulated as 3d landmarks. The landmark itself is stored within the vertex that first observed it. Loop closures might link observations of one mission to a landmark stored in another mission.

Fig. 5.3 illustrates the map structure and introduces the relevant coordinate frames. Each mission is anchored in the global coordinate frame $\mathcal{F}_G$ using a transformation $\mathbf{T}_{GM_i}$. The poses $\mathbf{T}_{M_iI_j}$ of mission i are expressed w.r.t. the mission frame $\mathcal{F}_{M_i}$. Therefore, it suffices to manipulate the transformation $\mathbf{T}_{GM_i}$ to anchor multiple missions in a single global coordinate system without the need for updating any vertex poses or landmark positions.

The map structure can be serialized to the Google Protobuf format, enabling portable file serialization and network transmission. Furthermore, data-intensive objects (such as images, dense reconstructions, etc.) can be attached to the maps using a resource management system. Resources are linked to either a *vertex* or a set of *missions* or simply a timestamp, and are stored on the file system separate from the main mapping data. This architecture allows for (cached) loading such (potentially large) objects on demand, effectively reducing the peak memory usage. This facilitates research in areas such as dense reconstruction and image-based/enhanced localization on large-scale maps that might otherwise exhaust the available memory on certain

[1] For more details, tutorials and documentation, please visit our wiki page: www.github.com/ethz-asl/maplab/wiki

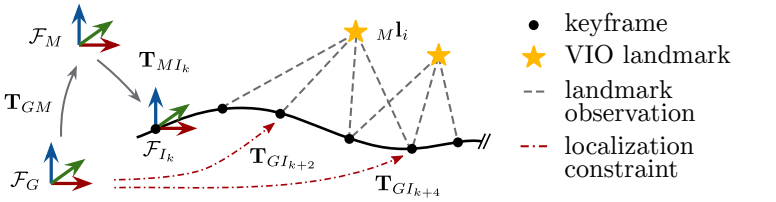


Figure 5.3: Coordinate frames used in maplab and ROVIOLI: $\mathcal{F}_G$: global, gravity-aligned map frame; all missions are anchored in this frame. $\mathcal{F}_{M_k}$: gravity-aligned frame that represents the origin of a mission k equivalent to the origin of the VIO. $\mathcal{F}_{I_k}$: IMU frame at time stamp k (body frame).

platforms.

3.5 Core Packages of maplab

The maplab framework incorporates implementations of several state-of-the-art algorithms. All of them are conveniently accessible from the maplab console. We only briefly highlight the ones that, in our opinion, bring a particular value to the robotics community:

visual-inertial least-squares optimization (VIWLS): least-squares optimization with cost terms similar to [52]. The main batch optimization algorithm of the framework is used to refine maps e.g. after initialization with ROVIOLI or after loop closures have been established. By default, the optimization problem is constructed using visual and inertial data, but optionally it can include wheel odometry, Global Positioning System (GPS) measurements or other types of pose priors.

Loop closure/localization: a complete loop closure and localization system based on binary descriptors. The search backend uses an inverted multi-index for efficient nearest neighbor retrieval on projected binary descriptors. The algorithm is a (partial) implementation of [59].

ROVIOLI: online visual-inertial mapping and localization frontend, see Section 3.6 for details.

Posegraph relaxation: posegraph optimization using edges introduced by the loop closure system. The algorithm is similar to [95]. Optionally, a Cauchy loss might be used to increase the robustness against false loop closures.

aslam_cv2: a collection of computer vision data structures and algorithms. It includes various camera and distortion models as well as algorithms for feature detection, extraction, tracking and geometric vision.

Map sparsification: algorithms to select the best landmarks for localization [19, 20] and keyframe selection to sparsify the pose graph. Useful for processing large-scale maps or for lifelong mapping.

Dense reconstruction: a collection of dense reconstruction, depth fusion and surface reconstruction [79] algorithms. Also includes an interface to CMVS/PMVS2 [35]. See Section 4.5 for details.

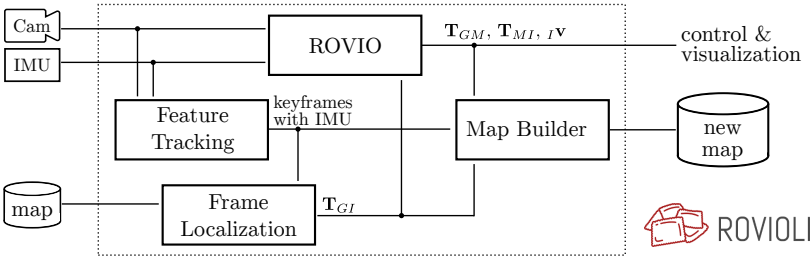


Figure 5.4: Modules and data flows within ROVIOLI (ROVIO [7] with Localization Integration).

3.6 ROVIOLI: Online VIO and Localization Frontend

ROVIOLI (ROVIO with Localization Integration) is maplab's mapping and localization frontend which is used to build maps from raw visual and inertial data and also localize w.r.t. existing maps online. It is built around the visual-inertial odometry framework ROVIO [7] and extends it with localization and mapping capabilities. The following two modes of operation are available: (i) *VIO mode* in which a map is built based on the VIO estimates and (ii) *LOC mode* where additionally localization constraints are processed to track a (drift-free) global pose estimate w.r.t. a given map. The localization maps are either created directly in a previous (single-session) of ROVIOLI or are exported from the maplab-console. The preparation of a localization map within the console allows for building complex processing pipelines (e.g. multi-session maps, data selection and compression).

An overview of the (main) data flows and modules within ROVIOLI are shown in Fig. 5.4. The *Feature Tracking* module detects and tracks BRISK [51] or FREAK [2] keypoints. Feature correspondences between frames are established by matching descriptors from frame to frame. The expected matching window is predicted based on integrated gyroscope measurements to increase the efficiency and robustness. In *LOC mode*, keyframes containing feature points and descriptors are processed by the *Frame Localization* module to establish 2d-3d matches against the provided localization map. These 2d-3d matches are used to obtain a global pose estimate $\mathbf{T}_{GI_k}$ w.r.t. the map's frame of reference (see Fig. 5.3) using a P3P algorithm within a RANSAC scheme. The raw global pose estimates are fed to ROVIO where they are fused with the odometry constraints to estimate a transformation $\mathbf{T}_{GM}$ in addition to the local odometry pose $\mathbf{T}_{MI}$. The outputs of all modules are synchronized within the *Map Builder* to construct a visual-inertial map (VI-map). The resulting map can serve as a localization map in subsequent sessions or can be loaded into the maplab console for further processing.

A process-internal publisher-subscriber data exchange layer manages the data flows between all modules within ROVIOLI. This architecture makes it easy to extend the current online pipeline with new algorithms, e.g. for online multiagent mapping, semantic SLAM, or localization research.

4 Use-cases

This section gives an overview of five common use cases of maplab: online mapping and localization, multi-session mapping, map maintenance, large-scale mapping and dense reconstruction. While maplab offers much more than that, we believe these examples highlight the capabilities of the system, the expected performance and its scalability.

Furthermore, we provide the related console commands to reproduce every example. The intention is to show that the following results can be obtained by relying solely on the user interface, without any additional code development. For more documentation, updated commands, datasets and tutorials, please visit our wiki page: `www.github.com/ethz-asl/maplab/wiki`.

4.1 Online Mapping and Localization with ROVIOLI

For many robotic applications it is of high importance to have access to (drift-free) global pose estimates. Such capability enables, e.g., teach and repeat scenarios, robotic manipulation and precise navigation. Within maplab, as a first step, we use ROVIOLI to create an initial VI-map of the desired area of operation. The sensor data can be provided either offline in a Rosbag or online using ROS topics. Upon completion, the VI-map is automatically loop closed, optimized and optionally keyframed and summarized to obtain a compact localization map. In a second session the localization map can be passed to ROVIOLI to obtain drift-free global pose estimates in the mapped area.

We evaluated the ROVIOLI estimates against plain ROVIO [7] results and the estimates from a full-batch optimization on the EuRoC datasets [11]. To that end, in a first step, we created a localization map using one of the datasets. Then in a second step we processed a second EuRoC dataset using both ROVIOLI (using the previously built map) and ROVIO. The results are presented in Fig. 5.5 and Table 5.1, where we compare the groundtruth error of ROVIO, ROVIOLI, and the full-batch optimized trajectory. These experiments demonstrate the drift-free performance of the system and the improvements upon the regular VIO estimation. Additionally, Table 5.2 shows timing information of ROVIO and ROVIOLI compared to ORB-SLAM2 [70].

4.2 Multi-session Mapping

In many mapping applications, it is not possible to cover the entire environment within a single mapping session. Apart from that, it might be desirable to capture the environment in as many differing visual appearance conditions as possible [12]. Therefore, maplab offers tools to co-register maps from multiple sessions together and jointly refine them to obtain a single, consistent map.

Hence this use case demonstrates the process of creating a map of a university building from four individual trajectories. Each trajectory passes through the ground floor, staircases and one other floor of a building. Combined, they cover over 1,000 meters and contain about 463,000 landmarks. On such large maps, many of the common operations such as optimization or loop closure quickly become intractable without a careful selection of the data. For this reason, we employ a keyframing scheme using heuristics based on vertex distance, orientation, and

Table 5.1: Global position and orientation RMSEs on EuRoC datasets [11] for ROVIO (only VIO), ROVIOLI using one of the datasets as a localization map and ROVIO+VIWLS that corresponds to a full batch VIWLS). Additionally, the results of ORB-SLAM2 [70] (in batch and real-time) are compared. ROVIO and ROVIOLI use a single camera and IMU data whereas ORB-SLAM2 uses a stereo camera. The localization map for ORB-SLAM2 has been built in SLAM mode whereas the localization evaluation has been performed in localization mode. For V2-medium, we were unable to build a map with ORB-SLAM2's real-time mode as the estimator diverged (marked with X).

	MH1 *LOC: MH2		V2-easy *LOC: V2-medium	
	position	orientation	position	orientation
ROVIO	0.178 m	1.49 deg	0.064 m	0.90 deg
ROVIOLI*	0.082 m	1.43 deg	0.057 m	1.57 deg
ROVIO+ VIWLS	0.036 m	1.29 deg	0.027 m	1.06 deg
ORB-SLAM2* (batch mode)	0.084 m	0.78 deg	0.121 m	1.14 deg
ORB-SLAM2* (real-time)	0.464 m	13.34 deg	X	X

Table 5.2: *(a)* Timing and CPU load for ROVIO, ROVIOLI and ORB-SLAM2 on EuRoC MH1 dataset processed at 20 Hz. In case of ROVIOLI and ORB-SLAM2 (marked with *), the estimator was set to localize against a map built from EuRoC MH2. All reported values have been measured on an Intel Xeon E3-1505M@2.8Ghz. A CPU load of 800% corresponds to fully utilizing **all** 8 (logical) cores of the CPU. *(b)* Single frame processing times for the individual blocks of ROVIOLI. The total time does not correspond to the sum of the individual blocks as they run in parallel. Instead, it is the time it takes for a single frame to be fully processed.

(a)

	Frame update	CPU load
ROVIO	23 ms	56%±7.7%
ROVIOLI*	44 ms	105%±14.8%
ORB-SLAM2* (batch mode)	63 ms	162%±10.9%

(b)

ROVIOLI frame update	
ROVIO update	22.7 ms
Feature tracking	20.6 ms
Localization	20.4 ms
Map building	3.2 ms
Total	**44.2 ms**

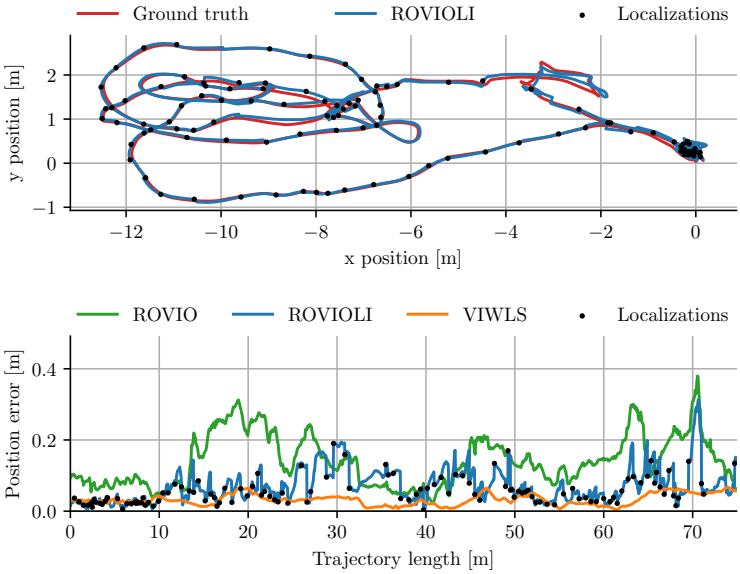


Figure 5.5: Evaluation of ROVIOLI on the EuRoC machine hall dataset [11]. *Top*: Ground-truth positions overlayed with the ROVIOLI position estimates. *Bottom*: Position error of the visual-inertial odometry pipeline ROVIO [7], ROVIOLI and the optimized VI-map (VIWLS) compared to the ground truth.

landmark covisibility. The loop closure algorithm of maplab correctly identifies the geometric transformations between all missions and the non-linear optimization refines the geometry. The result is a compact, geometrically-consistent localization map of 8.2 MB ready to be used by ROVIOLI for localization within the entire building as shown in Fig. 5.1.

This use case can be reproduced using the following commands in the maplab console:

```
# Load multiple single session maps from ROVIOLI.
load_merge_all_maps --maps_folder YOUR_MAPS_FOLDER
# Keyframing and initial optimization.
kfh
optvi
# Set one mission as base, anchor the others.
set_mission_baseframe_to_known
anchor_all_missions
# Pose-graph relaxation, loop-closure, optimization.
```

```
relax
lc
optvi
```

4.3 Map Maintenance

Large feature-based models, potentially built in multiple sessions, easily comprise thousands of landmarks and reach considerable storage size. However, it is not really necessary to keep all of the landmarks to guarantee good localization quality with ROVIOLI. Maplab offers a map summarization functionality based on [19] that uses an integer-based optimization to perform the landmark selection. The algorithm attempts to remove the least commonly seen landmarks but at the same time maintain a balanced coverage of the environment. Maplab also includes a keyframing algorithm to remove redundant vertices and only keep the ones necessary for an efficient and accurate state estimation. By removing the vertices we also eliminate many vertex-landmark associations that contain descriptors of considerable size. Both summarization and keyframing permit to significantly reduce the model size without a large loss in pose estimation quality.

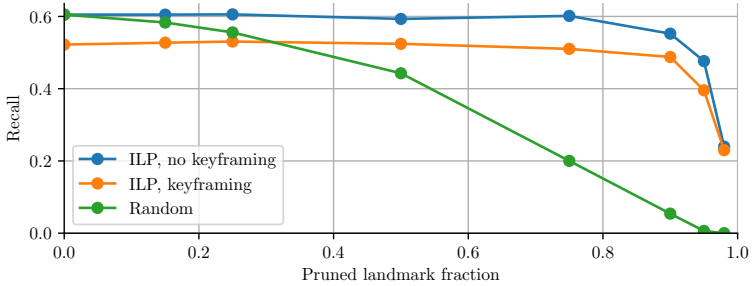
The map maintenance is demonstrated on a database map built from four mapping sessions recorded on the ground floor of the building introduced in Section 4.2. Each mapping session covers about 90 meters and contains about 20,000 landmarks, out of which about 5,000 are considered reliable. A fifth dataset is used as a query – we try to localize each vertex against the database, built from the four datasets, and verify if the position error is smaller than 50cm. We compare the recall of localization maps that were pre-processed in different ways, either summarized, keyframed or both.

Fig. 5.6 presents the influence of landmark summarization and keyframing on the localization map size and demonstrates how those approaches affect the localization. The results confirm that keyframing significantly reduces the localization map size with a rather marginal loss of localization quality. Similarly, summarization can reduce the total amount of landmarks by 90% without grave consequences. When these methods are combined we can reduce the map size 13 times and keep the recall level at 51%, compared to 60% for the full map.

```
# Keyframe the map and sparsify landmarks to 10,000.
kfh
landmark_sparsify --num_landmarks_to_keep=10000
```

4.4 Large-scale Mapping

In this use case we would like to demonstrate the large-scale mapping capabilities of maplab and the applicability to a sensor other than the VI-sensor [74]. To that end we used the publicly available Google Tango tablets, and recorded a large-scale, multi-session map of the old town of Zurich. We exported the raw visual-inertial data and processed it with ROVIOLI to obtain the initial open loop maps. We then loaded these maps into the maplab console for alignment and optimization and applied the same tools as described in Section 4.2. The bundle adjustment



pruning fraction	0	0.5	0.75	0.9	0.95	0.98
# landmarks	18,316 (16,088)	8,824 (9,148)	4,259 (4,570)	1,818 (1,822)	899 (906)	349 (358)
map size [MB]	34.559 (3.740)	29.217 (3.209)	24.028 (2.602)	17.619 (1.837)	12.203 (1.214)	6.707 (0.712)

Figure 5.6: The localization performance and map size after ILP landmark summarization and keyframing+summarization (in brackets). Keyframing removes vertices including vertex-landmark associations, effectively making the map smaller. The original map had $6,258$ vertices whereas the keyframed map contains 760. Keyframing consistently reduces the recall by a few percent while summarization only affects the quality when the pruned landmark fraction exceeds 85%. For comparison, we provide a recall curve for a random selection of landmarks to be removed.

and pose-graph relaxation was performed on a desktop computer with 32 GB RAM overnight. An orthographic projection of the optimized VI-map onto the map of Zurich, as well as further details about the map can be found in Fig. 5.7. The figure shows that the resulting map is consistent with the building and streets across most of the map with some minor inconsistencies in areas of low coverage.

4.5 Dense Reconstruction

Many applications in robotics, such as path planning, inspection and object detection require a more dense 3d representation of the environment. Maplab offers several dense reconstruction tools, which use the optimized *vertex* poses of the sparse map to compute dense depth information based on camera images attached to the VI-map.

Stereo Dense Reconstruction

In order to compute depth maps from multi-camera systems, this tool first identifies stereo cameras that are suitable for planar rectification. It then utilizes a (semi-global) block matcher

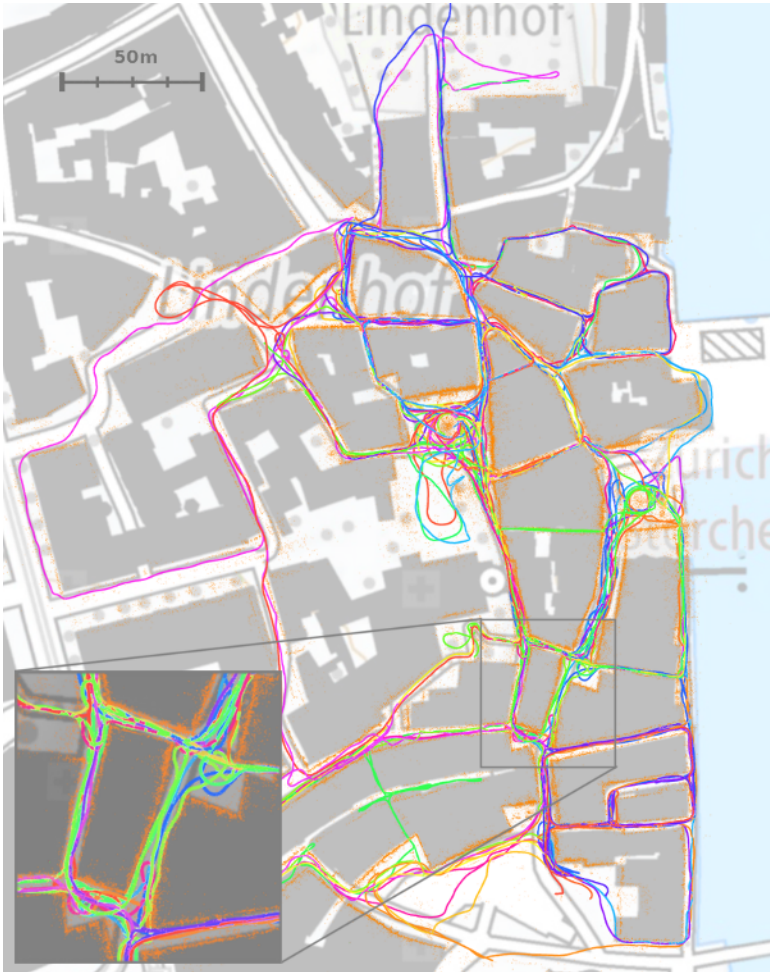


Figure 5.7: Large-scale, multi-session VI-map of Zurich's old town. Built from the raw visual-inertial data recorded in 45 sessions using Google Tango tablets on two different days (sunny and cloudy). The total duration of the recordings is 231 min. The final map contains trajectories with a total length of 16.48 km, 435k landmarks with 7.3M observations and has a size of 480 MB. The map is available on the maplab wiki page for download.

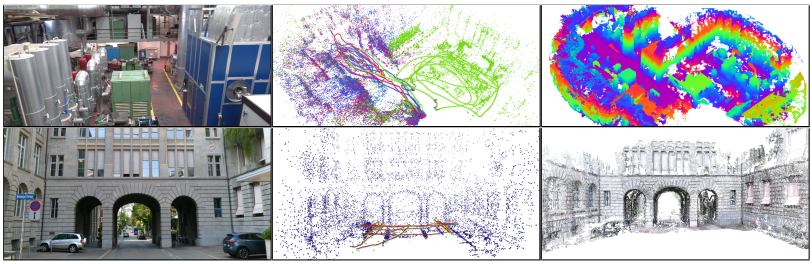


Figure 5.8: Two different dense reconstruction tools are available in maplab. *Top:* stereo dense reconstruction is used to compute depth maps based on grayscale images and optimized camera poses. They are then fused in voxblox [79] to create a surface mesh. 3 EuRoC datasets [10] (MH1-3) are combined to create an aligned and optimized VI-map. *Bottom:* CMVS/PMVS2 [35] reconstruction results based on a single recording session using a multi-camera system with a RGB camera.

to compute depth maps for every stereo pair along the trajectory. The resulting depth maps (or point clouds) are attached to the VI-map and stored in the resource system. The following commands assume that the maps are already aligned, loop closed and optimized as described in Section 4.2.

```
stereo_dense_reconstruction
```

TSDF-based Depth Fusion

Once the VI-map contains depth information, e.g. obtained using the above described commands or an RGB-D sensor, the globally consistent camera poses of the VI-map can be utilized to create an equally consistent global 3d reconstruction. To that end, maplab employs voxblox [79], a volumetric mapping library, for TSDF-based depth fusion and surface reconstruction. The following commands will insert depth maps or point cloud data into a voxblox grid and store a surface mesh to the file system. The top row of Fig. 5.8 shows the reconstruction results of three combined EuRoC machine hall datasets [11].

```
create_tsdf_from_depth_resource
   --dense_tsdf_voxel_size_m 0.10
   --dense_tsdf_truncation_distance_m 0.30
export_tsdf
   --dense_result_mesh_output_file YOUR_FILE
```

Export to CMVS/PMVS2

For more accurate dense reconstructions maplab offers an export command to convert the sparse VI-map and images to the input data format for the open-source multi-view-stereo pipeline, CMVS/PMVS2 [35]. Even though the export of grayscale images is supported, the best results are obtained using RGB images. The VI-map and the resulting 3d reconstruction can be seen in the bottom row of Fig. 5.8.

```
export_for_pmvs
   --pmvs_reconstruction_folder EXPORT_FOLDER
```

5 Using maplab for Research

All the algorithms and console commands required for the use cases in Section 4 are available in maplab and constitute most of the basic tools needed in visual-inertial mapping and localization. Furthermore, a rich set of helper functions, queries, and manipulation tools are provided to ease rapid prototyping of new algorithms. The plugin architecture of the console allows for an easy integration of new algorithms into the system. Examples demonstrating how to extend the framework are provided in the project's wiki pages. We would like to invite the community to take advantage of this research-friendly design.

6 Conclusions

This work presents maplab, an open framework for visual-inertial mapping and localization with the goal of making research in this field more efficient by providing a collection of basic algorithms and letting researchers focus on actual tasks. All components in maplab are written in a flexible and extensible way such that novel algorithms that rely on visual-inertial state estimates or localization can be integrated and tested easily. For this reason, the framework provides an implementation of the most important tools required in mapping and localization related research such as visual-inertial optimization, a loop closure/localization backend, multi-session map merging, pose-graph relaxation and extensive introspection and visualization tools. All these algorithms are made accessible from a console-based user interface where they can be applied to single or multi-session maps. Such a workflow has proven to be very efficient when prototyping new algorithms or tuning parameters.

Secondly, the framework contains an online visual-inertial mapping and localization front-end, named ROVIOLI. It can build new maps from raw visual and inertial sensor data and additionally track a global (drift-free) pose in real-time if a localization map is provided. Previous work made use of this capability on different robotic platforms and demonstrated its ability of accurately tracking a global pose for a multitude of applications, including navigation and trajectory following.

Acknowledgement

We would like to acknowledge the many other contributors of maplab, most importantly: Titus Cieslewski, Mathias Bürki, Timo Hinzmann, Mathias Gehrig and Nicolas Degen. Furthermore we would also like to thank Michael Blösch the author of ROVIO. The research leading to these results has received funding from Google Tango and the EU H2020 project UP-Drive under grant no. 688652.

Part B

SENSOR CALIBRATION

Paper III

Visual-inertial self-calibration on informative motion segments

Thomas Schneider, Mingyang Li, Michael Burri, Juan Nieto, Roland Siegwart and Igor Gilitschenski

Abstract

Environmental conditions and external effects, such as shocks, have a significant impact on the calibration parameters of visual-inertial sensor systems. Thus long-term operation of these systems cannot fully rely on factory calibration. Since the observability of certain parameters is highly dependent on the motion of the device, using short data segments at device initialization may yield poor results. When such systems are additionally subject to energy constraints, it is also infeasible to use full-batch approaches on a big dataset and careful selection of the data is of high importance.

In this paper, we present a novel approach for resource efficient self-calibration of visual-inertial sensor systems. This is achieved by casting the calibration as a segment-based optimization problem that can be run on a small subset of informative segments. Consequently, the computational burden is limited as only a predefined number of segments is used. We also propose an efficient information-theoretic selection to identify such informative motion segments. In evaluations on a challenging dataset, we show our approach to significantly outperform state-of-the-art in terms of computational burden while maintaining a comparable accuracy.

Published in:
2017 IEEE International Conference on Robotics and Automation (ICRA), 2017
DOI: 10.1109/ICRA.2017.7989766

1 Introduction

In this work, we address the problem of sensor self-calibration of a visual-inertial tracking system, i.e., a state estimation system that fuses measurements from an inertial measurement unit (IMU) and one/multiple cameras to compute pose (position and orientation) estimates of a moving platform. In recent years visual-inertial tracking has witnessed an ever increasing gain in popularity and is used in numerous mobile devices, virtual and augmented reality systems, and robotic platforms. This success story results in large-scale projects such as Google Tango or Microsoft's HoloLens promising to make these complex systems available as part of consumer devices with a limited energy supply and may be operated by inexperienced users over a potential lifespan of several years. These developments pose novel technical challenges to ensure accurate calibration of extrinsics and intrinsics of the underlying sensor systems.

Outside a lab environment, varying environmental conditions (such as temperature) and a long lifespan result in changing calibration parameters that make permanent use of factory calibration infeasible even when assuming all parameters to be constant over a short or medium timespan. In the absence of experienced engineers with access to special calibration routines and calibration patterns, the systems need to be capable of calibrating automatically in a potentially unknown environment. Even though it was shown that calibration is also possible by using natural visual landmarks only [56], parameters such as axis misalignment of the Inertial Measurement Unit (IMU) can only be observed under certain motion. One possible solution is to run a full-batch calibration procedure over as much data as possible. However, this results in a huge computational load making this methodology infeasible for consumer devices with limited computational resources and a limited power supply.

This work makes use of the fact that visual-inertial estimation systems typically run for a sufficiently long time to perform a lot of different types of motion eventually. Therefore, our system is designed to automatically select informative motion segments, that are well suited for calibration. The information measure, used for this identification, is illustrated on an example trajectory in Fig. 6.1. The most informative segments are then stored in a database and used to refine the calibration from time to time. This not only helps in getting good calibration data, but also reduces the size of the calibration problem considerably. Furthermore, we show that the results of our calibration, using only a small number of segments, is comparable in accuracy to the results obtained with a full batch approach over all the data collected.

This paper makes the following contributions:

- We present an efficient information-theoretic procedure to identify the most informative segments of a trajectory.
- We propose a segment-based method for self-calibration of the intrinsic and extrinsic parameters of visual-inertial sensor systems.
- In thorough evaluations we show that the proposed methodology achieves comparable results to a full batch approach and state-of-the-art while at the same time requires a significant lower complexity and computational effort.

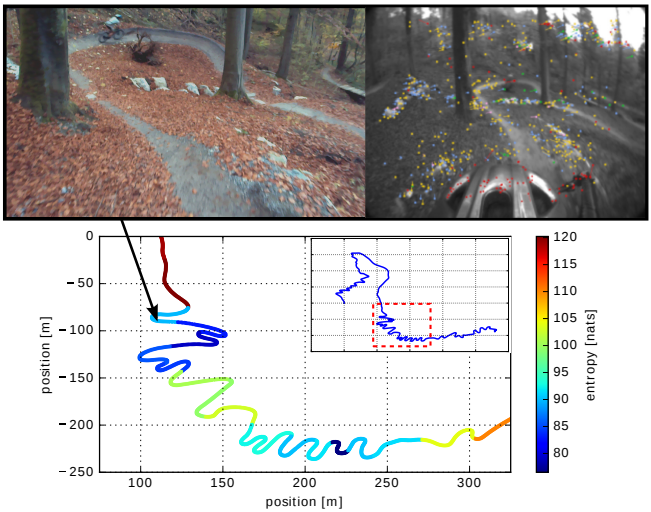


Figure 6.1: Riding down Mount Uetliberg on a mountain-bike with a camera and IMU attached to the rider's helmet: This dataset is a good illustration of the vastly varying amount of information available in different segments of the trajectory. The color indicates the information content of the segment w.r.t. the sensor calibration parameters (intrinsics and extrinsics of camera/IMU) where a lower value indicates more information. Consequently, the information measure is used to sparsify the sensor self-calibration problem by excluding less informative portions of the dataset.

2 Related Work

Over the last decade, visual-inertial Simultaneous Localization and Mapping (SLAM) has received great attention from the research community and tremendous progress has been achieved. For example, the work of [52] demonstrates a fixed-lag-smoother based visual-inertial odometry (VIO) framework, that achieves accuracies in the sub-percent range over the travelled distance. However, on constrained platforms such as mobile phones, filtering based algorithms are preferred such as [53] and [7] that show similar accuracies at lower computational complexity.

To achieve such accuracies, precise calibration of the sensor models is required. Traditionally, camera models are calibrated using a calibration target such as in the work of [104]. It has also been shown that camera models can be obtained using natural features only [18]. The increasing usage of low-cost Micro Electro-Mechanical Systems (MEMS) IMUs further requires calibration of the inertial sensors, referred to as IMU intrinsics. The work of [50] presents an inertial model, which we will adopt in this work, that considers scale inaccuracies and misalignments of individual sensors axes. In [75] a batch estimator is presented that calibrates the

latter model relying on a calibration pattern. The model of [84] additionally considers the location of individual accelerometer axes where the parameters are estimated in a continuous-time formulation using a parametric estimation framework.

The recent roll-out of advanced SLAM systems to a wide audience creates a need for simple calibration algorithms accessible to users without access to special equipment such as calibration targets. The work of [56] mitigates these short-comings by including the calibration parameters directly into an Extended Kalman Filter (EKF)-based VIO estimator and performing visual and inertial self-calibration solely based on natural features. Visual inertial systems, however, require special motion in order to render all calibration parameters observable [67]. Therefore, observability-aware calibration methods have been developed to aid non-expert users in collecting a complete dataset of minimal size and improve the estimation quality. In [62], a set of informative segments is selected using an information-gain measure to consequently perform a calibration over this set. Further, a truncated QR solver is used to constrain parameter updates to the observable sub-space. The generality of this method makes it applicable to a wide-range of estimation problems. Unfortunately, the evaluation of the utilized information metric is expensive and can prevent its use especially on resource constrained platforms. In our work, we follow a similar approach and identify informative motion segments to build a sparser but complete calibration dataset. Similarly to the work of [46], we use the entropy to efficiently approximate the information content of segments but calibrate the full visual-inertial model instead of just a camera. Additionally, we extend the information measure and evaluate the informativeness of segments w.r.t. to subgroups of the high-dimensional parameter vector and thus mitigate the drawback of using a scalar measure. Another interesting approach approximates the information of a trajectory segment by the local observability Gramian, as described in [40], where it is used in an active calibration setting.

3 Visual-inertial Models and Calibration

In this section, we will introduce the sensor models for the camera and IMU and formulate the batch estimation problem for self-calibration. [1]

3.1 Notation and Frames of Reference

A transformation matrix $\mathbf{T}_{AB} \in \mathbb{SE}^3$ takes vector ${}_B\mathbf{p} \in \mathbb{R}^3$ from the frame of reference B to the frame of reference A and can be further partitioned into a rotation matrix $\mathbf{R}_{AB} \in \mathbb{SO}^3$ and a translation vector ${}_A\mathbf{p}_{AB} \in \mathbb{R}^3$ as follows:

$$\begin{bmatrix} {}_A\mathbf{p} \\ 1 \end{bmatrix} = \mathbf{T}_{AB} \cdot \begin{bmatrix} {}_B\mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{AB} & {}_A\mathbf{p}_{AB} \\ \mathbf{0} & 1 \end{bmatrix} \cdot \begin{bmatrix} {}_B\mathbf{p} \\ 1 \end{bmatrix} \tag{6.1}$$

Further, the unit quaternion $\mathbf{q}_{AB}$ represents the rotation corresponding to $\mathbf{R}_{AB}$ as defined in [98]. The operator $\mathbf{T}_{AB}(\cdot)$ is defined to transform a vector in $\mathbb{R}^3$ from B to the frame of reference A as ${}_A\mathbf{p} = \mathbf{T}_{AB}\,({}_B\mathbf{p})$ according to 6.1.

[1] It is important to note that the method described in this paper generalizes to arbitrary problems, however it is presented on the application of visual-inertial self-calibration.

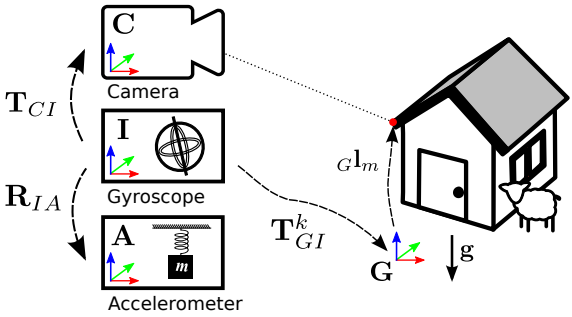


Figure 6.2: Frame of reference definitions for the visual-inertial system. A camera, 3-DoF accelerometer and 3-DoF gyroscope are rigidly attached to an agent. The estimated pose of the agent at timestep k is expressed by the transformation $\mathbf{T}^k_{GI}$. A 6-DoF transformation matrix $\mathbf{T}_{CI}$ relates the gyroscope's frame I to the camera's frame C. The accelerometer frame A is only rotated w.r.t. gyroscope's frame I by $\mathbf{R}_{IA}$, since ${}_I\mathbf{p}_{IA}$ in single-chip MEMS IMUs is typically close to zero.

The Fig. 6.2 illustrates the relevant coordinate frames used within this work. The frame G denotes a gravity aligned (${}_G\mathbf{e}_z = -\mathbf{g}$) inertial frame and is used to express the estimated pose of the agent $\mathbf{T}^k_{GI}$ and the position of the estimated landmarks ${}_G\mathbf{l}_m$. The frame I coincides with the sensing axes of the gyroscope and is chosen as the body frame of the agent. The camera frame C and accelerometer frame A are rigidly attached to the body frame. The extrinsic calibration transformations for the camera $\mathbf{T}_{CI}$ and the rotation matrix for the accelerometer $\mathbf{R}_{IA}$ are to be estimated and are both defined relative to the frame of the gyroscope I that is used as the body frame.

3.2 Inertial Model

A triad of (ideally) orthogonal gyroscopes are used to sense the true angular velocities ${}_I\boldsymbol{\omega}_{GI}$ of the body frame I w.r.t. the world-fixed inertial frame G. The gyroscope measurements $\tilde{\boldsymbol{\omega}}$ are modeled similar to [50, 56] as:

$$\tilde{\boldsymbol{\omega}} = \mathbf{T}_g \cdot {}_I\boldsymbol{\omega}_{GI} + \mathbf{b}_g + \boldsymbol{\eta}_g \tag{6.2}$$

where the bias $\mathbf{b}_g$ follows a random walk process as $\dot{\mathbf{b}}_g = \boldsymbol{\eta}_{bg}$ and $\boldsymbol{\eta}_g$ and $\boldsymbol{\eta}_{bg}$ are zero-mean, white Gaussian noise processes. The matrix $\mathbf{T}_g$ accounts for scale errors and sensor axis misalignments present in cheaper sensors. It is assumed to be a constant over time and is structured as:

$$\mathbf{T}_g = \begin{bmatrix} s_g^x & m_g^x & m_g^y \\ 0 & s_g^y & m_g^z \\ 0 & 0 & s_g^z \end{bmatrix}, \mathbf{s}_g = \begin{bmatrix} s_g^x \\ s_g^y \\ s_g^z \end{bmatrix}, \mathbf{m}_g = \begin{bmatrix} m_g^x \\ m_g^y \\ m_g^z \end{bmatrix} \tag{6.3}$$

with $\mathbf{s}_g$ and $\mathbf{s}_g$ denoting the collection of all parameters from $\mathbf{T}_g$.

Similarly, the specific force measurements $\tilde{\mathbf{a}}$ of the accelerometer are modeled as:

$$\tilde{\mathbf{a}} = \mathbf{T}_a \cdot \mathbf{R}_{AI} \cdot \mathbf{R}_{IG}^k \cdot ({}_G\mathbf{a}_{GI} - {}_G\,\mathbf{g}) + \mathbf{b}_a + \boldsymbol{\eta}_a \tag{6.4}$$

where $\mathbf{T}_a$ is a calibration matrix and $\mathbf{b}_a$ defines a random walk process analog to the gyroscope model. The calibration states for the IMU models can be summarized as:

$$\boldsymbol{\theta}_i = \begin{bmatrix}\mathbf{s}_g^T & \mathbf{m}_g^T & \mathbf{s}_a^T & \mathbf{m}_a^T & \mathbf{q}_{AI}^T\end{bmatrix}^T \tag{6.5}$$

It is important to note that the values of the scale parameters s_i can't be used directly to correct the scales of each individual axis, instead a linear combination of all factors applies. Further details can be found in [56].

3.3 Camera Model

Let ${}_G\mathbf{l}_m$ denote a 3-d landmark observed from keyframe k that is projected into a 2-d point $\mathbf{z}_{k,m}$ on the image plane of the camera as follows:

$$\mathbf{z}_{k,m}(\mathbf{T}_{IG}, \mathbf{l}_m, \boldsymbol{\theta}_c) = f_p(\boldsymbol{\theta}_c, \mathbf{T}_{CI}(\mathbf{T}_{IG}({}_G\mathbf{l}_m))) + \boldsymbol{\eta}_c \tag{6.6}$$

where $f_p(\cdot)$ denotes the perspective projection function and $\boldsymbol{\eta}_c \sim \mathcal{N}(\mathbf{0}, \sigma_c^2 \cdot \mathbf{I}_2)$ a white Gaussian noise process.

For the evaluations, we parametrize the projection function f_p using a pinhole camera model and field-of-view (FOV) distortion model of [18]. The calibration state relevant for the camera model then is:

$$\boldsymbol{\theta}_c = \begin{bmatrix}\mathbf{q}_{CI}{}^T & {}_C\mathbf{p}_{CI}{}^T & \mathbf{f}^T & \mathbf{c}^T & w\end{bmatrix}^T$$

where $\mathbf{q}_{CI}$ and ${}_C\mathbf{p}_{CI}$ are the extrinsic calibration of the camera w.r.t. IMU, $\mathbf{f} = \begin{bmatrix}f_x & f_y\end{bmatrix}^T$ the focal lengths, $\mathbf{c} = \begin{bmatrix}c_x & c_y\end{bmatrix}^T$ the principal point and w a distortion parameter.

3.4 Maximum-likelihood Estimator

The framework of maximum-likelihood estimation is used to jointly estimate all keyframe states $\mathbf{x}_k$ (6.7), the scene as a set of observed point landmarks ${}_G\mathbf{l}_m$, the calibration parameters of the camera $\boldsymbol{\theta}_c$ and the IMU $\boldsymbol{\theta}_i$ with the keyframe state $\mathbf{x}_k$ being defined as:

$$\mathbf{x}_k = \begin{bmatrix}\mathbf{q}_{GI}^k{}^T & {}_G\mathbf{p}_{GI}^k{}^T & {}_G\mathbf{v}_I^k{}^T & \mathbf{b}_a^k{}^T & \mathbf{b}_g^k{}^T\end{bmatrix}^T \tag{6.7}$$

where $\mathbf{q}_{GI}^k$ and ${}_G\mathbf{p}_{GI}^k$ denote the pose of the agent, ${}_G\mathbf{v}_I^k$ the velocity of the IMU expressed in frame G, and $\mathbf{b}_\cdot^k$ the biases for the gyroscope or accelerometer. For convenience of notation, the individual states are stacked into vectors as follows:

$$\hat{\mathbf{X}} = \begin{bmatrix}\hat{\mathbf{x}}_0^T & \dots & \hat{\mathbf{x}}_K^T\end{bmatrix}^T, \quad \hat{\mathbf{L}} = \begin{bmatrix}{}_G\hat{\mathbf{l}}_M{}^T & \dots & {}_G\hat{\mathbf{l}}_M{}^T\end{bmatrix}^T,$$
$$\hat{\boldsymbol{\theta}} = \begin{bmatrix}\hat{\boldsymbol{\theta}}_c^T & \hat{\boldsymbol{\theta}}_i^T\end{bmatrix}^T$$

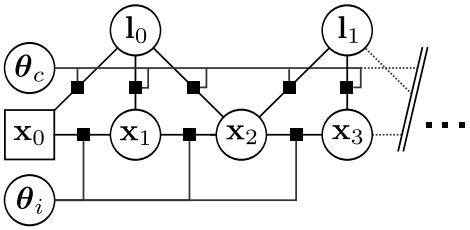


Figure 6.3: Calibration problem in factor graph representation that contains visual-inertial keyframe states $\mathbf{x}_k$ (pose, velocity and IMU biases), landmarks $\mathbf{l}_m$ and calibration states for the camera $\boldsymbol{\theta}_c$ and IMU $\boldsymbol{\theta}_i$. The square around the initial node $\mathbf{x}_0$ denotes a gauge fix of the position ${}_G\mathbf{p}_{GI}$ and rotation around the gravity vector. Integrated IMU measurements constitute the inertial factor $g_k^{imu}(\mathbf{x}_k, \mathbf{x}_{k-1}, \boldsymbol{\theta}_i, \mathbf{u}_k)$ and the landmark projection factors $g_{k,m}^{cam}(\mathbf{x}_k, \mathbf{l}_m, \boldsymbol{\theta}_c, \mathbf{z}_{k,m})$ models the camera measurements.

where K denotes the number of keyframes and M the number of landmarks. Additionally $\hat{\boldsymbol{\pi}}$ defines the collection of all estimated quantities as:

$$\hat{\boldsymbol{\pi}} = \begin{bmatrix} \hat{\boldsymbol{\theta}}^T & \hat{\mathbf{X}}^T & \hat{\mathbf{L}}^T \end{bmatrix}^T$$

We want to infer $\boldsymbol{\pi}$ from measurements $\mathbf{z}_{k,m}$ made by a camera and measurements $\mathbf{u}_k$ of an IMU. The stacked vector forms of the measurements are defined as follows:

$$\begin{aligned} \mathbf{Z} &= \{\mathbf{z}_{k,m} | k \in [0, K], m \in [0, M(k)]\} \\ \mathbf{U} &= \{\mathbf{u}_k | k \in [0, K-1]\} \end{aligned}$$

Following the sensor models described in Section 3.2-3.3, a probability model is defined as shown in Fig. 6.3. Probabilistic inertial constraints g_k^{imu} between consecutive keyframe states k and $k+1$ are formed as a function of the integrated IMU measurements and the corresponding measurement uncertainties [53]. The likelihood $p(\cdot)$ of this model can be expressed as:

$$\begin{aligned} p(\boldsymbol{\pi}|\mathbf{Z}, \mathbf{U}) \propto & \prod_{k=1}^{K} p(\mathbf{x}_k | \mathbf{x}_{k-1}, \boldsymbol{\theta}_i, \mathbf{u}_k) \\ & \cdot \prod_{k=0}^{K} \prod_{m=0}^{M(k)} p(\mathbf{z}_{k,m} | \mathbf{x}_k, \mathbf{l}_m, \boldsymbol{\theta}_c) \end{aligned} \tag{6.8}$$

where $p(\mathbf{x}_k | \mathbf{x}_{k-1}, \boldsymbol{\theta}_i, \mathbf{u}_i)$ denotes the inertial constraints between two consecutive keyframe states as a function of integrated IMU measurements $\mathbf{u}_k$ and $p(\mathbf{z}_{k,m} | \mathbf{x}_k, \mathbf{l}_m)$ the measurement model of the point landmark observation $\mathbf{z}_{k,m}$ of the m-th landmark observed from the k-th keyframe. More details on the derivation can be found in [75].

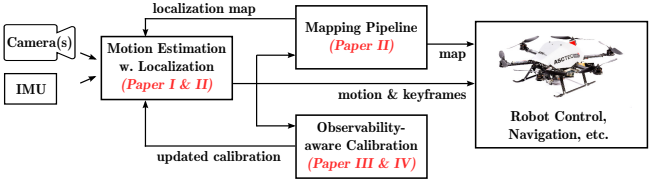


Figure 6.4: System overview and context: informative motion segments are identified from the output of an existing ego-motion estimator (COM) and maintained in a database for future calibration.

The maximum-likelihood (ML) estimate $\hat{\boldsymbol{\pi}}_{ML}$ is obtained by solving the optimization problem that maximizes the likelihood of 6.8:

$$\hat{\boldsymbol{\pi}}_{ML} = \underset{\boldsymbol{\pi}}{\operatorname{argmax}}\, p(\boldsymbol{\pi}|\mathbf{Z}, \mathbf{U}) \tag{6.9}$$

With the assumptions of Gaussian noise for all sensor models, as discussed in Section 3.2 and Section 3.3, the optimization problem defined in 6.9 is equivalent to a non-linear least squares problem. This problem can be solved using numerical minimization approaches, where standard methods include Gauss-Newton, Levenberg-Marquardt, Dogleg, etc. In our implementation, we use the Levenberg-Marquardt implementation of the Ceres framework [1].

4 Method

The proposed self-calibration method aims at being run in parallel to an existing visual-inertial SLAM system that provides motion estimates as shown in Fig. 6.4. In our implementation we use a concurrent odometry and mapping (COM) framework consisting of [53], [73] and [59] but it is important to note that the proposed algorithms are not tied to a particular SLAM formulation. The SLAM system uses a calibration from previous runs or nominal values for the device at hand. [2] The stream of estimated keyframes $\hat{\mathbf{x}}_i$ and landmarks $\hat{\mathbf{l}}_i$ leaving the COM module is partitioned into motion segments $\mathcal{S}^i$ of a predefined size N as follows:

$$\begin{aligned} \hat{\mathbf{X}}^i_{\mathcal{S}} &= \left[\hat{\mathbf{x}}_i^T, \quad \cdots, \quad \hat{\mathbf{x}}_{i+(N-1)}^T\right]^T \\ \hat{\mathbf{L}}^i_{\mathcal{S}} &= \left[\hat{\mathbf{l}}_i^T, \quad \cdots, \quad \hat{\mathbf{l}}_{i+(N-1)}^T\right]^T \end{aligned} \tag{6.10}$$

where $\hat{\mathbf{X}}^i_{\mathcal{S}}$ denotes the keyframes within the i-th segment and $\hat{\mathbf{L}}^i_{\mathcal{S}}$ the landmarks observed by the i-th segment. An efficient information-theoretic measure is used to evaluate each new candidate segment for their information content w.r.t. the calibration parameters and the most informative segments are maintained in a database. Once enough segments have been collected,

[2] If no priors are available, a complete self-calibration may be difficult and specialized initialization techniques should be used beforehand e.g. [44, 104].

an ML-based calibration is triggered to estimate the calibration parameters. An overview of the algorithm is shown in Alg. 1. The remainder of this section will discuss the algorithm in more detail.

Algorithm 1 Method shown for a single parameter group

Input: Initial calibration: $\hat{\boldsymbol{\theta}}_{init}$
Output: Updated calibration: $\hat{\boldsymbol{\theta}}$

Loop
  // Initialize motion segments of size N from COM output.
  $\mathcal{S}_i \leftarrow \{\}$
  repeat
    $data$ = `WaitForNewSensorData()`
    $\hat{\mathbf{x}}_j, \hat{\mathbf{l}}_j \leftarrow$ `RunCOM(`$data, \hat{\theta}_{init}$`)`
    $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup (\hat{\mathbf{x}}_j, \hat{\mathbf{l}}_j)$
  until $dim(\mathcal{S}_i) == N$;
  $H(\boldsymbol{\theta}) \leftarrow$ `EvaluateSegmentInformation(`$\mathcal{S}_i$`)` // Section 4.1
  `UpdateDatabase(`$\mathcal{S}_i, H(\boldsymbol{\theta})$`)` // Section 4.2
  if `EnoughSegmentsInDatabase()` **then**
    $\mathbf{S}_{info} \leftarrow$ `GetAllSegmentsFromDatabase()`
    $\hat{\boldsymbol{\theta}} \leftarrow$ `RunOptimization(`$\mathbf{S}_{info}$`)` // Section 4.3
    return $\hat{\boldsymbol{\theta}}$
  end
  $i \leftarrow i + 1$
EndLoop

4.1 Evaluating Information Content of Segments

We use the differential entropy to quantify the information content of the i-the candidate segment $\mathcal{S}_i$ w.r.t. the calibration parameters $\boldsymbol{\theta}$ by considering only the constraints within each segment. Using the entropy to evaluate the information of a candidate segments, as a score that is independent of all other segments, makes its evaluation very efficient at the cost that information coming from other segments is neglected. For example, loop-closure constraints cannot be considered in the score, however, loop-closures are considered during calibration.

To calculate the segment entropy, we first approximate the covariance matrix of all states in the segment by the inverse of the Fisher Information Matrix as:

$$\boldsymbol{\Sigma}_{\mathbf{XL}\boldsymbol{\theta}} = \text{Cov}\left[p(\mathbf{X}_{\mathcal{S}}^i, \mathbf{L}_{\mathcal{S}}^i, \boldsymbol{\theta} | \mathbf{U}_i, \mathbf{Z}_i)\right] = (\mathbf{J}_i^T \mathbf{T}_i^{-1} \mathbf{J}_i)^{-1} \tag{6.11}$$

where $\mathbf{J}_i$ denotes the Jacobian of all error-terms in the segment and $\mathbf{T}_i$ the stacked error-term covariance where the column ordering is chosen that the calibration parameters $\boldsymbol{\theta}$ lie on the right side. To avoid a costly inversion of 6.11, which becomes intractable for larger problems, we make use of a rank-revealing QR decomposition to obtain $\mathbf{Q_i R_i} = \mathbf{L}_i \mathbf{J}_i$ where $\mathbf{T}_i^{-1} = \mathbf{L}_i^T \mathbf{L}_i$ denotes the Cholesky decomposition of the error-term covariance matrix. 6.11 can then be rewritten as:

$$\boldsymbol{\Sigma}_{\mathbf{XL}\boldsymbol{\theta}} = (\mathbf{R}_i^T \mathbf{R}_i)^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{XL}} & \boldsymbol{\Sigma}_{\mathbf{XL},\boldsymbol{\theta}} \\ \boldsymbol{\Sigma}_{\mathbf{XL},\boldsymbol{\theta}}^T & \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \end{bmatrix} \tag{6.12}$$

In the context of sensor calibration, the keyframe $\mathbf{X}_{\mathcal{S}}^i$ and landmark states $\mathbf{L}_{\mathcal{S}}^i$ are considered nuisance variables and we are only interested in the marginal covariance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}} =$

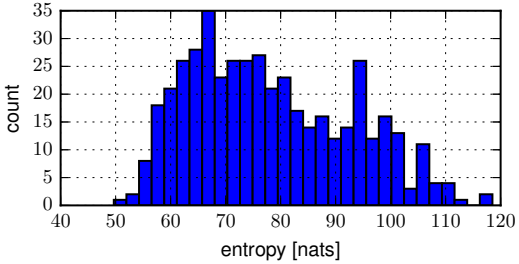


Figure 6.5: Histogram of normalized segment entropies $H(\boldsymbol{\theta})$ over 450 segments from 15 datasets.

$\mathrm{Cov}\left[p(\boldsymbol{\theta}|\mathbf{U}_i, \mathbf{Z}_i)\right]$ of the calibration parameters $\boldsymbol{\theta}$. As $\mathbf{R}_i$ is an upper-triangular matrix, we can efficiently obtain the marginal covariance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ by back-substitution.

Before calculating the entropy, we normalize the marginal covariance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}$ to account for the different scales of the calibration parameters. The normalized covariance $\bar{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}$ is calculated as:

$$\bar{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}} = \mathrm{diag}(\boldsymbol{\sigma}_{ref})^{-1} \cdot \boldsymbol{\Sigma}_{\boldsymbol{\theta}} \cdot \mathrm{diag}(\boldsymbol{\sigma}_{ref})^{-1} \tag{6.13}$$

where $\boldsymbol{\sigma}_{ref}$ is the expected standard deviation of $\hat{\boldsymbol{\theta}}$ and was obtained from statistics over multiple reference segments. The differential entropy $H(\boldsymbol{\theta})$ of the normalized multivariate normal distribution $\bar{p}_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \bar{p}(\boldsymbol{\theta}|\mathbf{U}_i, \mathbf{Z}_i)$ can then be calculated as:

$$\begin{aligned} H(\boldsymbol{\theta}) &= -\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \bar{p}_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \ln \bar{p}_{\boldsymbol{\theta}}(\boldsymbol{\theta})\, d\boldsymbol{\theta} \\ &= \frac{1}{2} \ln\left((2\pi e)^k \cdot \det\left(\bar{\boldsymbol{\Sigma}}_{\boldsymbol{\theta}}\right)\right), \end{aligned} \tag{6.14}$$

where k is the dimension of the normal distribution.

The segment entropy $H(\boldsymbol{\theta})$ is not a directional measure and thus summarizes the information of all parameters $\boldsymbol{\theta}$ in a single scalar value. For high-dimensional calibration vectors $\boldsymbol{\theta}$, however, the contribution of well-observable modes to the entropy might shadow weaker modes despite normalization. This effect causes the distribution of the entropies to remain multimodal (as shown in Fig. 6.5) because the number of informative segments vs. less informative segments is in general not distributed equally within a given dataset.

For this reason, the vector of calibration parameters $\boldsymbol{\theta}$ is partitioned into Q sub-vectors $\boldsymbol{\theta}_q$ as:

$$\boldsymbol{\theta} = \begin{bmatrix} \tilde{\boldsymbol{\theta}}_0^T & \dots & \tilde{\boldsymbol{\theta}}_Q^T \end{bmatrix}^T \tag{6.15}$$

The marginal entropy is calculated for each parameter group q using the corresponding marginal covariance $\bar{\boldsymbol{\Sigma}}_{\tilde{\boldsymbol{\theta}}_q}$ as described in 6.14. The marginal segment entropies $H(\boldsymbol{\theta}_q)$ are then directly

used as a measure of information contained in the segment w.r.t. to the parameters of group q (where a lower entropy corresponds to richer information).

In this work, we partition the parameters $\boldsymbol{\theta}$ into three groups by sensor:

$$\begin{aligned}\tilde{\boldsymbol{\theta}}_0 &= \begin{bmatrix}\mathbf{s}_g^T & \mathbf{m}_g^T & \mathbf{s}_a^T & \mathbf{m}_a^T & \mathbf{q}_{AI}^T\end{bmatrix}^T \\ \tilde{\boldsymbol{\theta}}_1 &= \begin{bmatrix}\mathbf{f}^T & \mathbf{c}^T & w\end{bmatrix}^T \\ \tilde{\boldsymbol{\theta}}_2 &= \begin{bmatrix}\mathbf{q}_{CI}{}^T & {}_C\mathbf{p}_{CI}{}^T\end{bmatrix}^T\end{aligned} \tag{6.16}$$

This follows the intuition that the problem exhibits different co-observability structures i.e. a set of parameters is always observable as a group e.g. the camera model requires only minimal motion (once the landmarks are initialized) whereas the inertial model requires sufficient excitation. A more thorough analysis of how to identify the co-observability structure and thus optimally group the parameters should be part of future work.

4.2 Collecting Informative Segments in a DB

A database with Q tables is maintained where each table retains the N most informative segments for the corresponding parameter group q. Segments can be in multiple tables if it is informative w.r.t. multiple parameter groups. Therefore, the complexity of the calibration problem has an upper bound, as the max. number of segments in the database can be $Q \cdot N$ (or less if segments are in multiple tables).

It is important to note that the sum of segment entropies is a conservative approximation to the true information in the database for two reasons: First, the entropy is a scalar that "summarizes" the information of several parameters and thus does not contain any directional information. Second, for efficiency, the segment entropy is calculated by neglecting the cross-terms to other segments. This approximation of the information in the database can lead to the collection of redundant segments in the database. Nevertheless, the very efficient evaluation of the segment entropies outweighs the run-time penalty from including such redundant segments into the optimization

4.3 Sparsified Problem using Informative Segments

The calibration over the set of informative segments differs from the full batch problem, described in Section 3, in that non-informative segments have been removed. This results in missing inertial constraints between the remaining segments as shown in Fig. 6.6 (e.g. between keyframe 6/10 and 12/16). The set of segments can then contain partitions that are neither constrained to other partitions through inertial constraints nor by joint landmark observations. Each of these partitions can be seen as a (nearly) independent calibration problem only sharing calibration states with other partitions.

If we assume the availability of sufficient landmark constraints and non-degenerate motion (e.g. only rotation), then the visual-inertial calibration problem contains two structurally unobservable states: the global orientation around the gravity vector and the global position. To ensure an optimal and efficient convergence of the iterative solvers these redundant degrees of

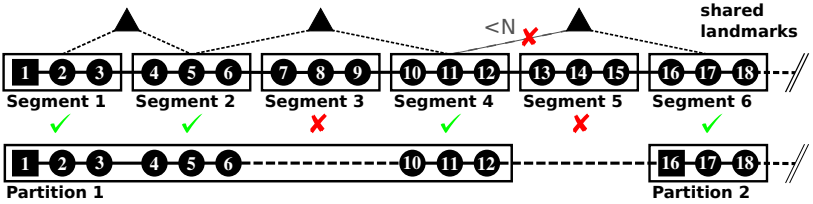


Figure 6.6: The upper graph shows the full keyframe/landmark graph where the keyframes are assigned into fixed-size segments and the segments found to be informative are marked by green check-marks The motion segments of the sparsified problem (shown below) can be partitioned into disjoint sets that are neither connected through inertial constraints nor share more than N landmark observations with other partitions. During optimization, the structurally unobservable states are fixed for exactly one keyframe per partition (marked by a square: e.g. 1)

.

freedom need to be held constant during optimization for exactly one keyframe in each of the partitions.

Algorithm 2 Partitioning segments on landmark co-visibility

Input: Set of motion segments $\mathbf{S} = \{\mathcal{S}_0, \ldots, \mathcal{S}_K\}$
Input: Max. co-observed landmarks between partitions N
Result: Set of motion segment partitions $\mathcal{P}$

$\mathbf{P} \leftarrow \{\}$
foreach $\mathcal{S}_k \in \mathbf{S}$ **do**
  $\mathbf{C} \leftarrow \{\{\mathcal{S}_k\}\}$
  foreach $p \in \mathbf{P}$ **do**
    if `CountSharedLandmarks`$(p, \mathcal{S}_k) > N$ **then**
      $\mathbf{C} \leftarrow \mathbf{C} \cup \{p\}$
    end
  end
  $p_\mathcal{C} \leftarrow$ `MergePartitions`$(\mathbf{C})$
  $\mathbf{P} \leftarrow (\mathbf{P} \setminus \mathbf{C}) \cup \{p_\mathcal{C}\}$
end

Consequently, we identify these partitions by first joining all motion segments that have direct temporal neighbors into bigger segments (Fig. 6.6: e.g. segment 1 and 2). At this point, all keyframes within the joined segments are constrained through inertial constraints. The union-find algorithm, shown in Alg. 2, is then used to iteratively partition the segments into disjoint sets such that the count of co-observed landmarks between the partitions lies below a given threshold N (here: 15). This ensures that all keyframes within these partitions are either connected through inertial constraints or share sufficient landmark observations with other keyframes of the same partition. Degenerate landmark configurations are theoretically possible, when using such a heuristic landmark threshold, but are highly unlikely and would only affect the convergence rate but not bias the estimates.

Additionally, a constraint between two bias states is introduced if keyframes were removed

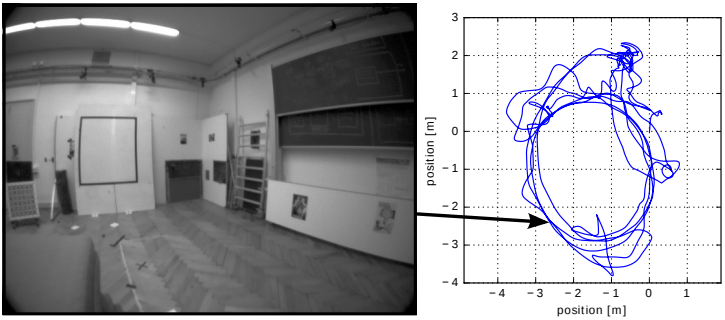


Figure 6.7: Top-down view on the estimated trajectory of one of the evaluation datasets.

between the two (Fig. 6.6: e.g. between keyframe 6-10 and 12-16). The bias evolution is modeled using a random walk as described in 3.2.

5 Experiments and Results

A collection of 15 datasets is used to assess the performance of the proposed visual-inertial self-calibration method. The datasets were collected using a Google Tango Dev. Kit. tablet equipped with a MEMS IMU and a global shutter fisheye camera. The device was hand-held while recording multiple trajectories of 3 min duration while freely moving in a room of approx. 8x6 m with a height of 4 m. The trajectories consist of calmer sections and sections that excite all rotational and translational degrees of freedom. Fig. 6.7 shows an image of the experimental environment together with a top-down view on one of the recorded trajectories.

In this section, we discuss our evaluation results based on these datasets along the following questions:

- Does the sparsified calibration problem yield comparable results to the batch solution (Section 5.1)?
- Is the proposed measure capable of identifying informative segments (Section 5.2)?
- Can the estimation be improved by grouping certain parameters and collecting segments for each group separately (Section 5.3)?
- How does the proposed approach perform against comparable state-of-the-art methods in terms of run-time and estimation results (Section 5.4)?

5.1 Performance and Repeatability of the Calibration

We compare the estimated parameters of the sparsified problem to the batch solution that uses all keyframes. The sparsified problem, here, denotes the calibration problem that only contains the most informative segments as described in Section 4. The initial calibration states were set to the CAD values, if available, otherwise to the expected nominal values (i.e. no sensor misalignment, unit scale factors). Table 6.1 shows the mean and standard deviation over the estimated parameters of the 15 different datasets and the convergence of the estimator is shown in Fig. 6.8. The mean of rotation parameters corresponds to the Rodrigues angle $\gamma(\cdot)$ of the averaged quaternion [61] over all data points and the standard deviation is calculated from the Rodrigues angles between the data points and the averaged quaternion.

These experiments show that the deviation between the sparsified estimation and the batch solution remains insignificant, in both the mean and standard deviation, even though large portions of the trajectory have been removed. This indicates that the proposed method can sparsify the problem while retaining an estimation quality close the batch solution at a drastically reduced run-time. It is important to note, that we cannot evaluate the accuracy of the estimated parameters as no ground-truth data is available, the statistics, however, give a good indication of the precision that can be achieved.

5.2 Evaluation of Segment Entropy to Select Informative Segments

In this section, we conduct an experiment to investigate the suitability of the segment entropy to identify informative segments. For this reason, we let the estimator from Section 4 collect the least informative segments and compare the convergence of two parameters with results obtained by selecting the most informative segments. In both cases, we collect 8 segments each of which consists of 40 keyframes resulting in a total of 320 keyframes which is equal to the data of 32 s. The convergence is shown in Fig. 6.9 together with the statistics on the final calibration states. The calibration using the set of least-informative segments yields a higher estimation error w.r.t. the batch solution and a higher variance than the estimation using the most-informative segments. This can be seen as an indication that:

1. the selection using the entropy identifies segments containing relevant information for sensor calibration,

2. the ratio of the number of selected segments to the total count of segments in the dataset is sufficiently low such that a careful selection is actually necessary.

5.3 Influence of Multiple Parameter Groups

In this section, we analyze the effect on the estimation performance when collecting segments for individual parameter groups. The estimator has been run with the parameter groups described in Section 4.2 and as a comparison with a single group that contains all calibration parameters.

The Table 6.2 lists statistics of two estimated calibration parameters over 15 datasets. The results show that the variance of the estimates can be reduced by using multiple groups whereas

Table 6.1: Estimated calibration parameters for three different algorithms. The statistics are taken over 15 datasets and show the mean and standard deviation. The number of used segments and run-time can be found in Table 6.3.

parameter	informative (proposed method)	all segments (batch)	related work [62]
$\mathbf{f}$ [px]	254.71 ± 0.28	254.50 ± 0.13	254.89 ± 0.35
	254.63 ± 0.29	254.47 ± 0.14	254.68 ± 0.32
$\mathbf{c}$ [px]	317.26 ± 0.32	317.51 ± 0.18	317.74 ± 0.35
	244.61 ± 0.45	244.56 ± 0.21	242.87 ± 0.67
w [-]	0.9222 ± 0.0003	0.9222 ± 0.0003	0.9227 ± 0.0007
$\mathbf{s}_g - 1$ [-]	6.17e-04 ± 9.61e-04	4.45e-05 ± 3.52e-04	5.40e-04 ± 1.10e-03
	5.80e-03 ± 8.51e-04	5.56e-03 ± 5.36e-04	4.78e-03 ± 1.53e-03
	8.54e-04 ± 3.89e-04	8.44e-04 ± 1.97e-04	4.00e-04 ± 6.74e-04
$\mathbf{s}_a - 1$ [-]	-2.07e-02 ± 2.28e-03	-2.07e-02 ± 1.47e-03	-2.15e-02 ± 2.33e-03
	-1.73e-02 ± 1.25e-03	-1.77e-02 ± 5.21e-04	-1.82e-02 ± 1.01e-03
	-1.42e-02 ± 1.34e-03	-1.49e-02 ± 5.39e-04	-1.45e-02 ± 1.08e-03
$\mathbf{m}_g$ [-]	3.44e-04 ± 6.48e-04	7.42e-05 ± 3.23e-04	2.94e-04 ± 7.52e-04
	1.07e-03 ± 8.49e-04	1.23e-03 ± 4.75e-04	1.42e-03 ± 1.16e-03
	7.38e-04 ± 8.36e-04	4.31e-04 ± 5.02e-04	6.75e-04 ± 5.51e-04
$\gamma(\mathbf{q}_{GA})$ [deg]	1.467 ± 0.141	1.498 ± 0.056	1.501 ± 0.060
$\mathbf{m}_a$ [-]	1.78e-02 ± 4.58e-03	1.79e-02 ± 2.10e-03	1.82e-02 ± 1.88e-03
	-2.91e-02 ± 3.02e-03	-2.95e-02 ± 1.35e-03	-3.00e-02 ± 1.75e-03
	-1.18e-05 ± 1.80e-03	1.13e-04 ± 1.14e-03	-1.62e-03 ± 1.32e-03
${}_C\mathbf{p}_{IC}$ [m]	2.92e-03 ± 2.79e-03	4.12e-03 ± 1.01e-03	-3.43e-03 ± 3.42e-03
	1.25e-02 ± 2.55e-03	1.34e-02 ± 1.37e-03	1.38e-02 ± 1.94e-03
	-5.47e-03 ± 2.86e-03	-5.68e-03 ± 1.14e-03	-2.81e-03 ± 3.01e-03
$\gamma(\mathbf{q}_{IC})$ [deg]	0.311 ± 0.062	0.306 ± 0.019	0.170 ± 0.047

Table 6.2: Estimated calibration parameters: single parameter group vs. multiple groups.

	multiple groups	single group	batch
$\mathbf{c}$ [px]	317.35 ± 0.21	317.31 ± 0.35	317.51 ± 0.18
	244.61 ± 0.29	244.51 ± 0.52	244.56 ± 0.21
$\mathbf{m}_g$ [-]	1.64e-04 ± 4.54e-04	2.60e-04 ± 7.69e-04	7.42e-05 ± 3.23e-04
	1.09e-03 ± 5.78e-04	1.05e-03 ± 9.96e-04	1.23e-03 ± 4.75e-04
	5.33e-04 ± 5.18e-04	8.68e-04 ± 9.33e-04	4.31e-04 ± 5.02e-04

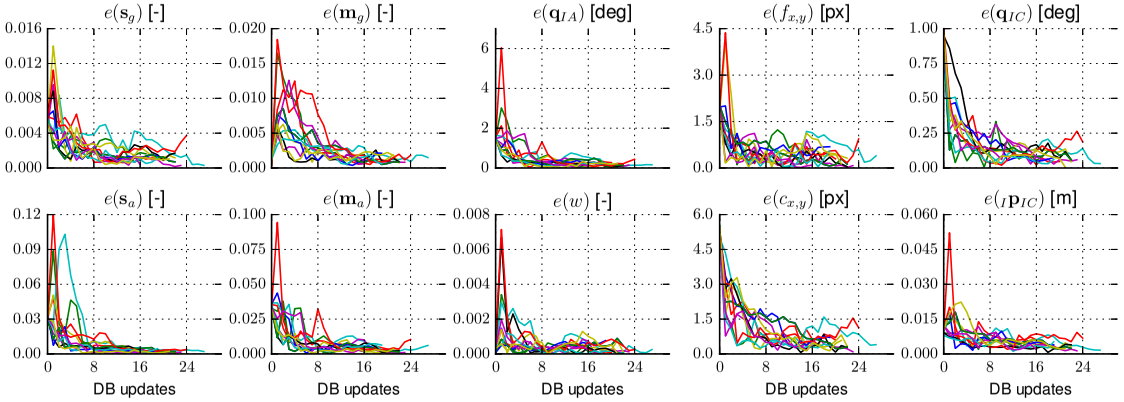
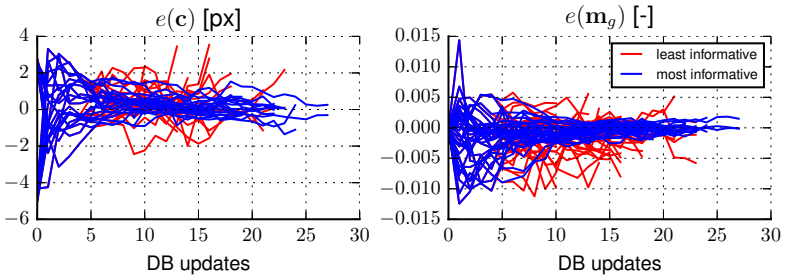


Figure 6.8: Convergence of the calibration parameters for each update of the database i.e. when a new segment is added or a less informative segment is replaced. The y-axis shows the deviation to the batch solution as: $e\left(\mathbf{x}\right) = \|\hat{\mathbf{x}}(k) \boxminus \hat{\mathbf{x}}_{batch}\|$.



	most informative	least informative	batch
$\mathbf{c}$ [px]	317.26 $\pm$ 0.32	318.24 $\pm$ 0.96	317.51 $\pm$ 0.18
	244.61 $\pm$ 0.45	244.85 $\pm$ 0.81	244.56 $\pm$ 0.21
$\mathbf{m}_g$ [-]	3.44e-04 $\pm$ 6.48e-04	-9.71e-04 $\pm$ 2.60e-03	7.42e-05 $\pm$ 3.23e-04
	1.07e-03 $\pm$ 8.49e-04	1.57e-04 $\pm$ 2.59e-03	1.23e-03 $\pm$ 4.75e-04
	7.38e-04 $\pm$ 8.36e-04	-1.25e-03 $\pm$ 3.75e-03	4.31e-04 $\pm$ 5.02e-04

Figure 6.9: Estimator performance when collecting the most-informative vs. the least-informative segments over 15 datasets. The plots show the deviation $e(\cdot)$ of the proposed method to the batch solution.

the averaged error remains less affected. Intuitively, this effect can be explained as follows: If the problem structure contains groups of parameters that are rendered observable by different motion patterns and only a single informative database table is used then the chances are higher that only one type of motion is kept. If multiple groups are used, however, the content in the database gets more stable and thus leads to a lower variance of the estimates.

5.4 Comparison to Related Method and Run-time

In this section, we compare the proposed method against the work of [62] in terms of estimation performance and run-time. The latter work follows a similar approach that maintains a database of informative segments. A calibration is run over the candidate segment and all segments already contained in the database. The candidate is found to be informative if the information gain w.r.t. a calibration without the candidate segment lies above a certain threshold. Since a complete calibration must be run for each candidate evaluation the complexity grows with each new segment in the database. In contrast to the proposed method, this algorithm does consider all constraints when evaluating the information content of a candidate segment and does not make the assumption of segment independence as outlined in Section 4.1. Furthermore, they use a truncated QR instead of the Cholseky solver therefore it is more general and applicable for a wider range of problems although at a higher computational cost.

Two time points are given for the related work as it doesn't use an upper bound on the number

Table 6.3: Run-time and number of processed segments for the three estimators. Each segment contains 40 keyframes and corresponds to the data of 4 s. Statistics are collected over 15 datasets.

	our method	**related work** [62]	**batch**
num. segments	8.7 $\pm$1.5	9.0	38.0 $\pm$3.2
run-time [s]	31.2 $\pm$5.6	395.9 $\pm$319.6 (7745.3 $\pm$4601.7)	178.5 $\pm$94.0

of selected informative segments. The first until the same amount of informative segments are collected as in the proposed method ($\approx$ 9) and the second (in brackets) until the information measure has been evaluated for each segment which is done in the proposed method by default. The same 15 datasets, used in the previous sections, have been processed with both methods. The run-times are shown in Table 6.3 and the estimated parameters in Table 6.1. The results show that the run-time of our algorithm is considerably lower than the full-batch and related work at very similar estimation performance.

6 Conclusions

In this work, we presented a novel method for efficient self-calibration of visual-inertial sensor systems that runs in parallel to an existing SLAM system. An information-theoretic measure is introduced to evaluate the information content of motion segments keeping a fixed number of the most-informative segments in a database. The proposed measure can be efficiently evaluated without running an expensive batch calibration beforehand. Once the database contains enough data, an optimization is run over these segments to update the calibration parameters.

Real-world experiments show that the sparsified problem yields similar results to the full batch solution at a significantly reduced computational cost. Even, when compared to previous work on segment based calibration, our approach shows a reduction of the run-time by a factor of approx. 10. Therefore, the proposed method is well suited for performing self-calibration on resource constrained platforms and can enable accurate operation over the entire lifespan.

Acknowledgement

The research leading to these results has received funding from Google's project Tango.

Paper IV

Observability-aware Self-Calibration of Visual and Inertial Sensors for Ego-Motion Estimation

Thomas Schneider, Mingyang Li, Cesar Cadena, Juan Nieto, and Roland Siegwart

Abstract

External effects such as shocks and temperature variations affect the calibration of visual-inertial sensor systems and thus they cannot fully rely on factory calibrations. Re-calibrations performed on short user-collected datasets might yield poor performance since the observability of certain parameters is highly dependent on the motion. Additionally, on resource-constrained systems (e.g mobile phones), full-batch approaches over longer sessions quickly become prohibitively expensive.

In this paper, we approach the self-calibration problem by introducing information theoretic metrics to assess the information content of trajectory segments, thus allowing to select the most informative parts from a dataset for calibration purposes. With this approach, we are able to build compact calibration datasets either: (a) by selecting segments from a long session with limited exciting motion or (b) from multiple short sessions where a single session does not necessarily excite all modes sufficiently. Real-world experiments in four different environments show that the proposed method achieves comparable performance to a batch calibration approach, yet, at a constant computational complexity which is independent of the duration of the session.

Published in:
IEEE Sensors Journal, 2019
DOI: 10.1109/JSEN.2019.2893809

1 Introduction

In this work, we present a sensor self-calibration method for visual-inertial ego-motion estimation frameworks i.e. systems that fuse visual information from one or multiple cameras with an Inertial Measurement Unit (IMU) to track the pose (position and orientation) of the sensors over time. Over the last years, visual-inertial tracking has become an increasingly popular method and is being deployed into a big variety of products including Augmented Reality/Virtual Reality (AR/VR) headsets, mobile devices, and robotic platforms. Large-scale projects, such as Microsoft's HoloLens, make these complex systems available as part of mass-consumer devices operated by non-experts over the entire life-span of the product. This transition from the traditional lab environment to the consumer market poses new technical challenges to keep the calibration of the sensors up-to-date.

Traditionally, visual-inertial sensors are calibrated in a laborious manual process by an expert often using specialized tools and external markers such as checkerboard patterns (e.g. [84]). Aside from a lack of equipment, the lack of knowledge on how to properly excite all modes usually renders these methods infeasible for consumers as specific motion is required to obtain a consistent calibration. However, it can be used at the factory to provide an initial calibration for the device. Due to varying conditions (e.g. temperature, shocks, etc.) such calibrations degrade over time and periodic re-calibrations become necessary. A straightforward approach to this problem would be to run a calibration over a long dataset, hoping it is rich enough to excite all modes of the system. Yet, the large computational requirement of such a batch method might render this approach infeasible on constrained platforms without careful data selection.

This work exploits that information is usually not distributed uniformly along the trajectory of most visual-inertial datasets, as illustrated in Fig. 7.1 for a mountain-bike dataset. trajectory segments with higher excitation provide more information for sensor calibration whereas segments with weak excitation can lead to a non-consistent or even wrong calibration. Consequently, we propose a calibration architecture that evaluates the information content of trajectory segments in a background process alongside an existing visual-inertial estimation framework. A database maintains the most informative segments that have been observed either in a single-session or over multiple sessions to accumulate relevant calibration data over time. Subsequently, the collected segments are used to update the calibration parameters using a segment-based calibration formulation.

By only including the most informative portion of the trajectory, we are able to reduce the size of the calibration dataset considerably. Further, we can collect exciting motion in a background process assuming such motion occurs eventually and thus take the burden from the users to perform them consciously (which might be hard for non-experts). With this approach we can automate the traditional tedious calibration task and perform a re-calibration without any user intervention e.g. while playing an AR/VR video game or while navigating a car through the city. Additionally, our method facilitates the use of more advanced sensor models (e.g. IMU intrinsics) with potentially weakly observable modes that require specific motion for a consistent calibration.

This article is an extension of our previous work [90] where we presented the following:

- an efficient information-theoretic metric to identify informative segments for calibration,
- a segment-based self-calibration method for the intrinsic and extrinsic parameters of a

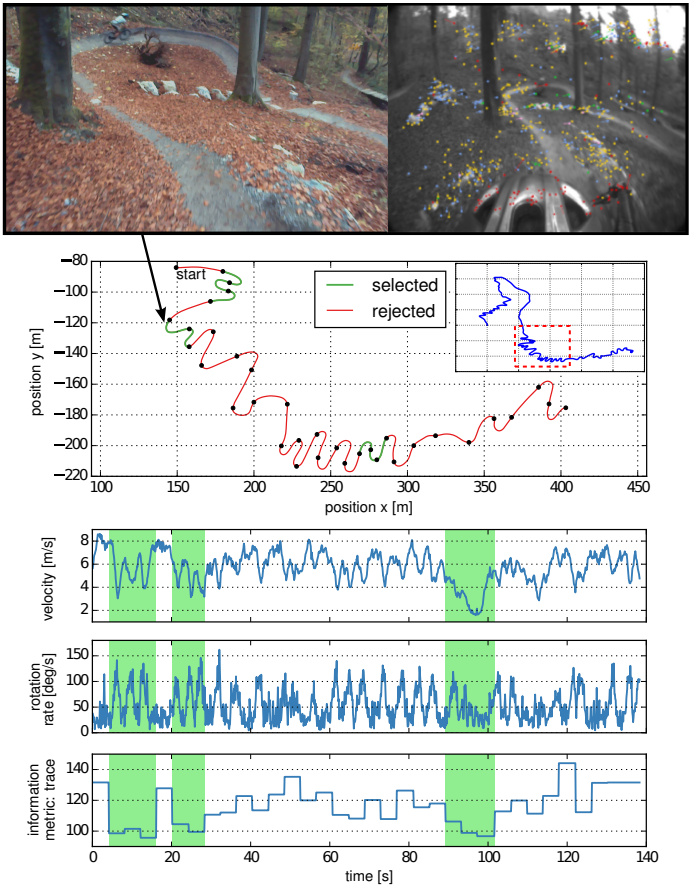


Figure 7.1: Dataset recorded while riding down Mount Uetliberg on a mountain-bike with a Tango Tablet strapped to the rider's head. This trajectory is a good example of the varying amount of information within different segments of a visual-inertial dataset. Our method identifies the most informative segments in a background process alongside an existing visual-inertial motion estimation framework. Consequently, we sparsify the dataset to ensure an efficient calibration of the camera and IMU model parameters. The illustration highlights the 8 most informative segments which are sufficient for a reliable calibration.

visual-inertial system, and

- evaluations of the calibration parameter repeatability showing comparable performance to a batch approach.

In this work, we extend with the following contributions:

- a comprehensive review of the state-of-the-art on visual and inertial sensor calibration,
- a study of three different metrics for the selection of informative segments,
- an evaluation of the motion estimation accuracy on motion-capture ground-truth, and
- a comparison against an Extended Kalman Filter (EKF) approach that jointly estimates motion and calibration parameters.

2 Literature Review

Over the past two decades, visual-inertial state estimation has been studied extensively by the research community and many methods and frameworks have been presented. For example, the work of Leutenegger et al. [52] fuses the information of both sensor modalities in a fixed-lag-smoother estimation framework and demonstrates metric pose tracking with an accuracy in the sub-percent range of distance traveled. Many applications on resource-constrained platforms, such as mobile phones, however, use filtering-based approaches which offer pose tracking with similar accuracy at a lower computational cost. An early method of this form is the one from Mourikis and Roumeliotis [69], and more recently also from Bloesch et al. [7], that directly minimizes a photometric error on image patches instead of a geometric re-projection error on point-features. Newer frameworks e.g. from Qin et al. [82] or Schneider et al. [91] also incorporate online localization/loop-closures to further reduce the drift or in certain cases even eliminate it completely.

All these methods require an accurate and up-to-date calibration of all sensor models to achieve good estimation performance. For this reason, a multitude of methods have been developed to calibrate models for the camera, IMU and relative pose between the two sensors. An overview of early methods that calibrate each model independently can be found in [3, 39, 57]. In the remaining of this section we, first, provide an overview of the state of the art in self-calibration of visual-inertial sensor systems and, second, discuss the most relevant *observability-aware* calibration approaches. And finally, we review methods that perform information-theoretic data selection for calibration purposes; which are most related to our approach.

2.1 Marker-based Calibration

The work on self-calibration of visual and inertial sensors is still limited and therefore, we first discuss approaches that rely on external markers such as checkerboard patterns. An approach based on an EKF is presented in [67] that uses a checkerboard pattern as a reference to jointly estimate the relative pose between an IMU and a camera with the pose, velocity, and biases.

Zachariah and Jansson [103] additionally estimate the scale error and misalignment of the inertial axis using a sigma-point Kalman filter.

A parametric method is proposed in [34] describing a batch estimator in continuous-time that represents the pose and bias trajectories using B-splines. Krebs and Rehder [50] extends this work by compensating additional sensing errors in the IMU model; namely measurement scale, axis misalignment, cross-axis sensitivity, the effect of linear accelerations on gyroscope measurements and the orientation between the gyroscope and the accelerometer. A similar model is calibrated by Nikolic et al. [75] where they make use of a non-parametric batch formulation and thus avoid the selection of a basis function for the pose and bias trajectories which might depend on the dynamics of the motion (e.g. over the knot density). The non-parametric and parametric formulation are compared in real-world experiments with the conclusion that the accuracy and precision of both methods are similar [75].

2.2 Marker-less Calibration

In contrast to target-based, self-calibration methods solely rely on natural features to calibrate the sensor models without the need for external markers such as checkerboards. Early work of this from was presented by Kelly and Sukhatme [47] and uses an unscented Kalman filter to jointly estimate pose, bias, velocity, IMU-to-camera relative pose and also the local scene structure. Their real-world experiments demonstrate that the relative pose between a camera and an IMU can be accurately estimated with similar quality to target-based methods. The work of Patron-Perez et al. [80] additionally calibrates the camera intrinsics and uses a continuous-time formulation with a B-splines parameterization. Li et al. [56] go one step further and also include the following calibration parameters into the (non-parametric) EKF-based estimator: time offset between camera and IMU, scale errors and axis misalignment of all inertial axis, linear acceleration effect on the gyroscope measurements (g-sensitivity), camera intrinsics including lens distortion and the rolling-shutter line-delay. A simulation study and real-world experiments indicate that all these quantities can indeed be estimated online solely-based on natural features [56].

2.3 Observability of Model Parameters

All of the discussed calibration methods so far, both target-based and self-calibration methods, rely on sufficient excitation of all sensor models to yield an accurate calibration. Mirzaei and Roumeliotis [67] formally prove that the IMU-to-camera extrinsics are observable in a target-based calibration setting where the observability only depends on sufficient rotational motion. The analysis of Kelly and Sukhatme [47] shows that the IMU-to-camera extrinsics remains observable also for a self-calibration formulation. Further, Li and Mourikis [55] derive the necessary condition for the identifiability of a constant time offset between the IMU and camera measurements.

So far, no observability analysis has been performed for the full joint self-calibration problem that includes the intrinsics of the IMU and camera and also the relative pose between the two sensors. Our experience, however, indicates that 'rich' exciting motion is required to render all parameters observable and usually such calibration datasets are collected by expert intuition. Often, this knowledge is missing when Simultaneous Localization and Mapping (SLAM) sys-

tems are deployed to consumer-market products. For this reason, the (re-)calibration dataset collection process must be automated for true life-long autonomy.

2.4 Active Observability-aware Calibration

Active calibration methods automate the dataset collection by planning and executing trajectories which ensure the observability of the calibration parameters wrt. a specified metric. An early work in this direction for target-based camera calibration is [87]. They present an interactive method that suggests the next view of the target that should be captured such that the quality of the model improves incrementally.

Another active calibration method is presented by Bähnemann et al. [5] to plan informative trajectories using a sampling-based planner to calibrate Micro Aerial Vehicle (MAV) models. The informativeness of a candidate trajectory segment within the planner is approximated by the determinant of the covariance of the calibration parameters which is propagated using an EKF. In a similar setting, Hausman et al. [40] plan informative trajectories to calibrate the model of an Unmanned Aerial Vehicle (UAV) using the local observability Gramian as an information measure. An extension to this work is presented by Preiss et al. [81] where they additionally consider free-space information and dynamic constraints of the vehicle within the planner. The condition number of the *Expanded Empirical Local Observability Gramian* (E^2LOG) is proposed as an information metric. The columns of E^2LOG are scaled using empirical data to balance the contribution of multiple states. A simulation study shows that the method outperforms random motion and also the well-known heuristic, such as the figure-8 or star motion pattern. Further, the study indicates that trajectories minimizing the E^2LOG perform slightly better compared to the minimization of the trace of the covariance matrix but in general yield comparable performance.

2.5 Passive Observability-aware Calibration – Calibration on Informative Segments

In contrast to the class of active calibration methods, passive methods cannot influence the motion and instead identify and collect informative trajectory segments to build a complete calibration dataset over time. The framework of Maye et al. [62] selects a set of the most informative segments using an information gain measure to consequently perform a calibration on the selected data. A truncated-QR solver is used to limit updates to the observable subspace. The generality of this method makes it suitable for a wide range of problems. Unfortunately, the expensive information metric and optimization algorithm prevent its use on resource-constrained platforms. Similarly, Keivan and Sibley [46] maintain a database of the most informative images to calibrate the intrinsic parameters of a camera but use a more efficient entropy-based information metric for the selection. Nobre et al. [76] extend the same framework to calibrate multiple sensors and more recently Nobre et al. [77] also include the relative pose between an IMU and a camera.

In our work, we take a similar approach to [46, 62] but also consider inertial measurements and consequently collect informative segments instead of images. In contrast to the general method of [62], we use an approximation for the visual-inertial use-case and neglect any cross-terms between segments when evaluating their information content. This approximation in-

creases the efficiency at the cost that no loop-closure constraints can be considered. Compared to [77], we assume the calibration parameters to be constant over a single session but additionally calibrate the intrinsic parameters of the IMU using a model similar to [50, 53].

3 Visual and Inertial System

The visual-inertial sensor system considered in this work consists of a global-shutter camera and an IMU. For better readability, the formulation is presented only for a single camera, however, the method has been tested for multiple cameras as well. All sensors are assumed to be rigidly attached to the sensor system. The IMU itself consists of a 3-axis accelerometer and a 3-axis gyroscope. In this work, we assume an accurate temporal synchronization of the IMU and camera measurements and exclude the estimation of the clock offset and skew. However, online estimation of these clock parameters is feasible as shown in [55].

The following subsections introduce the sensor models for the camera and IMU. An overview of all model parameters is shown in Table 7.1 and all relevant coordinate frames of the visual and inertial system in Fig. 7.2.

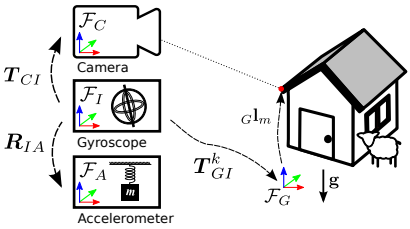


Figure 7.2: Coordinate frames of the visual-inertial sensor system: The camera, 3-DoF gyroscope and 3-DoF accelerometer are all rigidly attached to the sensor system. The frame $\mathcal{F}_C$ denotes the frame of the camera where ${}_C\mathbf{e}_z$ points along the optical axis, ${}_C\mathbf{e}_x$ left-to-right and ${}_C\mathbf{e}_y$ top-down as seen from the image plane. The 6-DoF transformation matrix $\boldsymbol{T}_{CI}$ (extrinsic calibration) relates the IMU $\mathcal{F}_I$ (which is defined to coincide with the frame of the gyroscope) to the frame of the camera $\mathcal{F}_C$. Since the translation ${}_I\mathbf{p}_{IA}$ between the gyroscope and the accelerometer is typically close to zero for single-chip MEMS sensors, we only rotate the accelerometer frame $\mathcal{F}_A$ w.r.t. to the gyroscopes frame $\mathcal{F}_I$ by the rotation matrix $\boldsymbol{R}_{IA}$. The frame $\mathcal{F}_G$ denotes a gravity aligned (${}_G\mathbf{e}_z = -\mathbf{g}$) inertial frame and is used to express the estimated pose of the sensor system $\boldsymbol{T}_{GI}^k$ and the position of the estimated landmarks ${}_G\mathbf{l}_m$.

3.1 Notation and Definitions

A transformation matrix $\boldsymbol{T}_{AB} \in SE(3)$ takes a vector ${}_B\mathbf{p} \in \mathbf{R}^3$ expressed in the frame of reference $\mathcal{F}_B$ into the coordinates of the frame $\mathcal{F}_A$ and can be further partitioned into a rotation

Table 7.1: Model parameters of the visual-inertial sensor system.

Parameter	Symbol	Dim.	Unit
Camera			
focal length	$\mathbf{f}$	$\mathbf{R}^2$	px
principal point	$\mathbf{c}$	$\mathbf{R}^2$	px
distortion	w	$\mathbf{R}$	-
IMU			
axis misalignment (gyro, accel.)	${}_a\mathbf{m}$, ${}_g\mathbf{m}$	$\mathbf{R}^3$,$\mathbf{R}^3$	-
axis scale (gyro, accel.)	${}_a\mathbf{s}$, ${}_g\mathbf{s}$	$\mathbf{R}^3$,$\mathbf{R}^3$	-
rotation $\mathcal{F}_A$ w.r.t. $\mathcal{F}_I$	$\boldsymbol{q}_{AI}$	$SO(3)$	-
Extrinsics			
translation $\mathcal{F}_C$ w.r.t. $\mathcal{F}_I$	${}_C\mathbf{p}_{IC}$	$\mathbf{R}^3$	m
rotation $\mathcal{F}_C$ w.r.t. $\mathcal{F}_I$	$\boldsymbol{q}_{IC}$	$SO(3)$	-

matrix $\boldsymbol{R}_{AB} \in SO(3)$ and a translation vector ${}_A\mathbf{p}_{AB} \in \mathbf{R}^3$ as follows:

$$\begin{bmatrix} {}_A\mathbf{p} \\ 1 \end{bmatrix} = \boldsymbol{T}_{AB} \cdot \begin{bmatrix} {}_B\mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_{AB} & {}_A\mathbf{p}_{AB} \\ \mathbf{0}_{1x3} & 1 \end{bmatrix} \cdot \begin{bmatrix} {}_B\mathbf{p} \\ 1 \end{bmatrix} \tag{7.1}$$

The unit quaternion $\boldsymbol{q}_{AB}$ represents the rotation corresponding to $\boldsymbol{R}_{AB}$ as defined in [98]. The operator $\boldsymbol{T}_{AB}(\cdot)$ is defined to transform a vector in $\mathbf{R}^3$ from $\mathcal{F}_B$ to the frame of reference $\mathcal{F}_A$ as ${}_A\mathbf{p} = \boldsymbol{T}_{AB}\,({}_B\mathbf{p})$ according to 7.1.

3.2 Camera Model

A function $f_p(\cdot)$ models the perspective projection and lens distortion effects of the camera. It maps the m-th 3d landmark ${}_{C_k}\mathbf{l}_m$ onto the image plane of the camera k to yield the 2d image point $\mathbf{p}_{k,m}$ as:

$$\mathbf{p}_{k,m} = f_p\left({}_{C_k}\mathbf{l}_m, \theta_c\right) \tag{7.2}$$

where θ_c denotes the model parameters of the perspective projection function (which we want to calibrate).

In our evaluation setup, we use high-field-of-view cameras as they typically yield more accurate motion estimates [105]. As a consequence the camera records a heavily distorted image of the world. To account for these effects, we augment the pinhole camera model with the field-of-view (FOV) distortion model [18] to obtain the following perspective projection function:

$$\mathbf{p}_{k,m} = f_p({}_C\mathbf{l}_m, \theta_c) = \begin{bmatrix} \beta_r\left(\|\overline{\mathbf{p}}_m\|\right) \cdot f_x \cdot \overline{p}_x + c_x \\ \beta_r\left(\|\overline{\mathbf{p}}_m\|\right) \cdot f_y \cdot \overline{p}_y + c_y \end{bmatrix} \tag{7.3}$$

where $f_{(\cdot)}$ denotes the focal length, $c_{(\cdot)}$ the principal point and $\overline{\mathbf{p}}$ the 2d projection of a 3d landmark ${}_C\mathbf{l}_m$ in normalized image coordinates as:

$$\overline{\mathbf{p}}_m = \frac{1}{{}_C\mathbf{l}_m^z} \cdot \begin{bmatrix} {}_C\mathbf{l}_m^x \\ {}_C\mathbf{l}_m^y \end{bmatrix} \tag{7.4}$$

The function β_r models the (symmetric) distortion effects as a function of the radial distance to the optical center as:

$$\beta_r(r) = \frac{\arctan\left(2 \cdot \tan\left(\frac{w}{2}\right) \cdot r\right)}{w \cdot r} \tag{7.5}$$

with w being the single parameter of the FOV distortion model.

The measurement model for landmark observations expressed in the global frame $\mathcal{F}_G$ (see Fig. 7.2) can be written as:

$$\begin{aligned} \widetilde{\mathbf{p}}_{k,m} &= f_p\left({}_C\mathbf{l}_m, \theta_c\right) + \eta_c \\ &= f_p\left(\boldsymbol{T}_{CI}\left(\boldsymbol{T}_{IG}\left({}_G\mathbf{l}_m\right)\right), \theta_c\right) + \eta_c \end{aligned} \tag{7.6}$$

where $\widetilde{\mathbf{p}}_{k,m}$ denotes the projection of the landmark m onto the image plane of the keyframe k, $\boldsymbol{T}_{IG}^k$ the pose of the sensor system, $\boldsymbol{T}_{CI}$ the relative pose of the camera w.r.t. the IMU and η_c a white Gaussian noise process with zero mean and standard deviation σ_c as $\eta_c \sim \mathcal{N}(\mathbf{0}, \sigma_c^2 \cdot \mathbf{I}_2)$. The full calibration state θ_c of the camera model can be summarized as:

$$\theta_c = \begin{bmatrix} \boldsymbol{q}_{IC}{}^T & {}_C\mathbf{p}_{IC}{}^T & \mathbf{f}^T & \mathbf{c}^T & w \end{bmatrix}^T$$

where the camera-IMU relative pose $\boldsymbol{T}_{CI}$ is split into its rotation part $\boldsymbol{q}_{IC}$ and its translation part ${}_C\mathbf{p}_{IC}$, $\mathbf{f} = \begin{bmatrix} f_x & f_y \end{bmatrix}^T$ is the focal length, $\mathbf{c} = \begin{bmatrix} c_x & c_y \end{bmatrix}^T$ the principal point and w the distortion parameter of the lens distortion model.

3.3 Inertial Model

The IMU considered in this work consists of a (low-cost) MEMS 3-axis accelerometer and a 3-axis gyroscope. As in the work of [50, 53, 75], we include the alignment of the non-orthogonal sensing axis and a correction of the measurement scale into our sensor model. Further, we assume the translation between the accelerometer and gyroscope to be small (single-chip IMU) and only model a rotation between the two sensors (as shown in Fig. 7.2).

Considering these effects, we can write the model for the gyroscope measurements $\tilde{\omega}$ as:

$$\tilde{\omega} = \mathbf{T}_g \cdot_I \omega_{GI} + \mathbf{b}_g + \eta_g \tag{7.7}$$

where ${}_I\omega_{GI}$ denotes the true angular velocity of the system, $\mathbf{T}_a$ a correction matrix accounting for the scale and misalignment of the individual sensor axis (see 7.15), $\mathbf{b}_g$ is a random walk process as:

$$\dot{\mathbf{b}}_g = \eta_{bg} \tag{7.8}$$

with the zero-mean white noise Gaussian processes being defined as

$$\eta_g \sim \mathcal{N}(\mathbf{0}, \sigma_g^2 \cdot \mathbf{I}_3), \tag{7.9}$$

$$\eta_{bg} \sim \mathcal{N}(\mathbf{0}, \sigma_{bg}^2 \cdot \mathbf{I}_3). \tag{7.10}$$

Similarly, the specific force measurements $\tilde{\mathbf{a}}$ of the accelerometer are modeled as:

$$\tilde{\mathbf{a}} = \mathbf{T}_a \cdot \boldsymbol{R}_{AI} \cdot \boldsymbol{R}_{IG}^k \cdot ({}_G\mathbf{a}_{GI} - {}_G\,\mathbf{g}) + \mathbf{b}_a + \eta_a \tag{7.11}$$

where ${}_G\mathbf{a}_{GI}$ is the true acceleration of the sensor system $\mathcal{F}_I$ w.r.t. to the inertial frame $\mathcal{F}_G$, $\boldsymbol{R}_{AI}$ the relative orientation between the gyroscope and accelerometer frame, $\boldsymbol{R}_{IG}^k$ the orientation of the IMU w.r.t. the inertial frame $\mathcal{F}_G$, $\mathbf{T}_a$ is a correction matrix for the scale and misalignment (see 7.15), ${}_G\mathbf{g}$ the gravity acceleration expressed in the inertial frame $\mathcal{F}_G$. The bias process $\mathbf{b}_a$ is defined as a random walk process as:

$$\dot{\mathbf{b}}_a = \eta_{ba} \tag{7.12}$$

with the zero-mean white noise Gaussian processes being defined as:

$$\eta_a \sim \mathcal{N}(\mathbf{0}, \sigma_a^2 \cdot \mathbf{I}_3), \tag{7.13}$$

$$\eta_{ba} \sim \mathcal{N}(\mathbf{0}, \sigma_{ba}^2 \cdot \mathbf{I}_3). \tag{7.14}$$

The noise characteristics of the IMU $\sigma_i = \begin{bmatrix}\sigma_g & \sigma_a & \sigma_{bg} & \sigma_{ba}\end{bmatrix}^T$ are assumed to have been identified beforehand at nominal operating conditions e.g. using the method described in [102]. The correction matrix $\mathbf{T}_g$ and $\mathbf{T}_a$ accounting for the scale and misalignment errors is defined identically for the gyroscope and accelerometer and is partitioned as:

$$\mathbf{T}_{(\cdot)} = \begin{bmatrix} s_{(\cdot)}^x & m_{(\cdot)}^x & m_{(\cdot)}^y \\ 0 & s_{(\cdot)}^y & m_{(\cdot)}^z \\ 0 & 0 & s_{(\cdot)}^z \end{bmatrix} \tag{7.15}$$

where $\mathbf{m}_{(\cdot)}$ denotes the collection of all misalignment and $\mathbf{s}_{(\cdot)}$ all scale factors as:

$$\mathbf{s}_{(\cdot)} = \begin{bmatrix} s_{(\cdot)}^x \\ s_{(\cdot)}^y \\ s_{(\cdot)}^z \end{bmatrix} \quad \mathbf{m}_{(\cdot)} = \begin{bmatrix} m_{(\cdot)}^x \\ m_{(\cdot)}^y \\ m_{(\cdot)}^z \end{bmatrix} \tag{7.16}$$

The full calibration state θ_i of the inertial model can then be summarized as:

$$\theta_i = \begin{bmatrix}\mathbf{s}_g^T & \mathbf{m}_g^T & \mathbf{s}_a^T & \mathbf{m}_a^T & \mathbf{q}_{AI}^T\end{bmatrix}^T \tag{7.17}$$

where $\boldsymbol{q}_{AI}$ describes the rotation of gyroscope frame $\mathcal{F}_G$ w.r.t. to the accelerometer frame $\mathcal{F}_A$ (with the IMU frame $\mathcal{F}_I$ being defined as the gyroscope frame $\mathcal{F}_G$).

4 Visual-Inertial Self-Calibration

In this section, we formulate the self-calibration problem for visual and inertial sensor systems using the sensor models introduced in the previous section. The derived Maximum Likelihood (ML) estimator makes use of all images and inertial measurements within the dataset to yield a *full-batch* solution. The motion of the sensor system and the (sparse) scene structure are jointly estimated with the model parameters to achieve self-calibration without the need for a known calibration target (e.g. a chessboard pattern). The batch estimator will serve as a base to introduce the segment-based calibration which only considers the most informative segments of a trajectory (see Section 5).

4.1 System State and Measurements

The self-calibration formulation jointly estimates all keyframe states $\mathbf{x}_k$, all point landmarks ${}_G\mathbf{l}_m$, the calibration parameters of the camera $\boldsymbol{\theta}_c$ and the IMU $\boldsymbol{\theta}_i$ with the keyframe state $\mathbf{x}_k$ being defined as:

$$\mathbf{x}_k = \begin{bmatrix} \boldsymbol{q}_{GI}^{k\,T} & {}_G\mathbf{p}_{GI}^{k\,T} & {}_G\mathbf{v}_I^{k\,T} & \mathbf{b}_a^{k\,T} & \mathbf{b}_g^{k\,T} \end{bmatrix}^T \tag{7.18}$$

where $\boldsymbol{q}_{GI}{}^k$ and ${}_G\mathbf{p}_{GI}^k$ define the pose of the sensor system at timestep k, ${}_G\mathbf{v}_I^k$ the velocity of the system and $\mathbf{b}_{(\cdot)}^k$ the bias of the gyroscope and accelerometer.

To simplify further notations, we collect all states of the problem in the following vectors:

$$\hat{\mathbf{x}}_{0..K} = \begin{bmatrix} \hat{\mathbf{x}}_0 \\ \vdots \\ \hat{\mathbf{x}}_K \end{bmatrix} \quad {}_G\hat{\mathbf{l}}_{0..M} = \begin{bmatrix} {}_G\hat{\mathbf{l}}_0 \\ \vdots \\ {}_G\hat{\mathbf{l}}_M \end{bmatrix} \quad \hat{\theta} = \begin{bmatrix} \hat{\theta}_c \\ \hat{\theta}_i \end{bmatrix} \tag{7.19}$$

where K is the total number of keyframes and M the number of landmarks. Additionally, the vector $\hat{\boldsymbol{\pi}}_{K,M}$ stacks all estimated states as:

$$\hat{\boldsymbol{\pi}}_{K,M} = \begin{bmatrix} \hat{\mathbf{x}}_{0..K}^T & {}_G\hat{\mathbf{l}}_{0..M}{}^T & \hat{\theta}^T \end{bmatrix}^T \tag{7.20}$$

Further, we define the collection U to contain all IMU measurements and Z all 2d landmark observations of the camera as:

$$\begin{aligned} \mathbf{U} &= \{\mathbf{u}_k | k \in [0, K-1]\} \\ \mathbf{Z} &= \{\mathbf{p}_{k,m} | k \in [0, K], m \in [0, M(k)]\} \end{aligned} \tag{7.21}$$

where $\mathbf{u}_k$ is the set of all accelerometer and gyroscope measurements between the keyframes k and $k+1$ and $\mathbf{p}_{k,m}$ the 2d measurement of the m-th landmark seen from the k-th keyframe and K and M denote the number of keyframes and landmarks respectively.

4.2 State Initialization using visual-inertial odometry (VIO)

A vision front-end tracks sparse point features between consecutive images and rejects potential outliers based on geometrical consistency using a perspective-n-point algorithm in a RANSAC scheme. The resulting feature tracks and the IMU measurements are processed by an EKF which is loosely based on the formulation of [56, 69] but with various extension to increase robustness and accuracy. The filter recursively estimates all keyframe states $\mathbf{x}_{0..K}$ and landmark positions ${}_G\mathbf{l}_{0..M}$. The calibration states are not estimated by this filter except for the camera-to-IMU relative pose (camera extrinsics). However, for the initialization of the calibration problem, we only use the keyframe states (pose, velocity, biases) and the most recent estimate of the camera-to-IMU extrinsics. The landmark states are initialized by triangulation using the poses estimated by the EKF filter.

It is important to note that the filter needs sufficiently good calibration parameters in order to run properly and provide accurate initial estimates. In our experience, it is sufficient for most single-chip IMUs to initialize their intrinsic calibration to a nominal value (unit scale, no misalignment). However, a complete self-calibration may be difficult if no priors are available for the camera intrinsics. In this case, a specialized calibration method should be used beforehand e.g. [84, 104].

4.3 ML-based Self-Calibration Problem

We use the framework of ML estimation to jointly infer the state of all keyframes $\hat{\mathbf{x}}_{0..K}$, landmarks ${}_G\hat{\mathbf{l}}_{0..M}$ and calibration parameters $\hat{\theta}$ using all available measurements U of the IMU and the 2d measurements Z of the point landmarks extracted from the camera images. A factor graph representation of the visual-inertial self-calibration formulation is shown in Fig. 7.3. The problem contains two types of factor: the visual factor $g_{k,m}^{cam}$ models the projection of the landmark m onto the image plane of the keyframe k and the inertial factor g_k^{imu} forms a

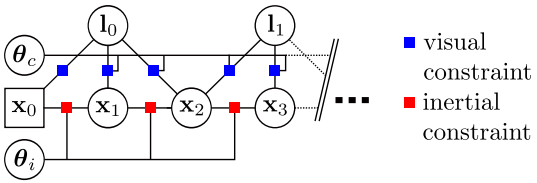


Figure 7.3: Batch calibration problem shown in factor-graph representation: the problem contains keyframe states $\mathbf{x}_k$ (pose, velocity, gyroscope and accelerometer biases), the calibration states for the IMU $\boldsymbol{\theta}_i$ and the camera $\boldsymbol{\theta}_c$ and the landmarks $\mathbf{l}_m$. Two types of factor are used: (red) inertial constraints $g_k^{imu}\left(\mathbf{x}_k, \mathbf{x}_{k+1}, \boldsymbol{\theta}_i, \mathbf{u}_k\right)$ based on the integrated IMU measurements; (blue) landmark reprojection factors $g_{k,m}^{cam}\left(\mathbf{x}_k, \mathbf{l}_m, \mathbf{p}_{k,m}\right)$ modeling the feature observations (measurements of a landmark projection) observed by the camera. Additionally, the unconstrained directions of the first keyframe state, namely the global position ${}_G\mathbf{p}_{GI}^0$ and the rotation around the gravity vector $\boldsymbol{q}_{GI}^0$ (z-axis of frame $\mathcal{F}_G$) are fixed to zero (denoted by the square).

differential constraint between two consecutive keyframe states x_k and x_{k+1} (pose, velocity, bias). The ML estimate $\hat{\boldsymbol{\pi}}_{ML}$ is obtained by a maximization of the corresponding likelihood function $p(\boldsymbol{\pi}|\mathbf{Z}, \mathbf{U})$. When assuming Gaussian noise for all sensor models (see Section 3), the ML solution can be approximated by solving the (non-linear) least-squares problem with the following objective function $S(\boldsymbol{\pi})$:

$$\begin{aligned} S(\boldsymbol{\pi}) = &\sum_{k=0}^{K} \sum_{m}^{M(k)} \mathbf{e}_{k,m}^{cam\,T} \mathbf{W}_{k,m}^{cam} \mathbf{e}_{k,m}^{cam} \\ &+ \sum_{k=0}^{K-1} \mathbf{e}_{k}^{imu\,T} \mathbf{W}_{k}^{imu} \mathbf{e}_{k}^{imu} \end{aligned} \tag{7.22}$$

where K denotes the number of keyframes, $M(k)$ the set of landmarks off from keyframe k, $\mathbf{e}_{k,m}^{cam}$ the reprojection error of the m-th point landmark of observed from the k-th keyframe and $\mathbf{e}_{k}^{imu}$ denotes the inertial constraint error between two consecutive keyframe states k and $k+1$ as a function of integrated IMU measurements. The terms $\mathbf{W}_{k,m}^{cam}$ and $\mathbf{W}_{k}^{imu}$ denote the inverse of the error covariance matrices: keypoint measurement and the integrated IMU measurement covariance respectively. The reprojection error $\mathbf{e}_{k,m}^{cam}$ is defined as:

$$\mathbf{e}_{k,m}^{cam} = \mathbf{p}_{k,m} - \widetilde{\mathbf{p}}_{k,m} \left(\boldsymbol{T}_{CI}, \boldsymbol{T}_{IG,G}^{k}\, \mathbf{l}_m, \theta_c \right) \tag{7.23}$$

where $\mathbf{p}_{k,m}$ is the 2d measurement of the projection of the landmark m into camera k and $\widetilde{\mathbf{p}}_{k,m}$ its prediction as defined in 7.6. The inertial error $\mathbf{e}_{k}^{imu}$ is obtained by integrating the continuous equations of motion using the sensor models described in Section 3.3 and is based on the method described in [56]. The non-linear objective function $S(\boldsymbol{\pi})$ is minimized using numerical optimization methods. In our implementation, we use the Levenberg-Marquardt implementation of the Ceres framework [1].

5 Self-Calibration using Informative Motion Segments

In this section, we propose a method to identify informative segments in a calibration dataset and a modified formulation for estimating calibration parameters based on a set of segments. First, the method can be used to sparsify a dataset and consequently reduce the complexity of the optimization problem. And second, a complete calibration dataset can be built over time by accumulating informative segments from multiple sessions, thus enabling the calibration of even weakly observable parameters by collecting exciting motion that occurs eventually. It is important to note that the proposed method is presented on the use-case of visual-inertial calibration but it can be applied to arbitrary calibration problems.

5.1 Architecture

A high-level overview of the modules and data-flows is shown in Fig. 7.4. The proposed method is intended to be run in parallel to an existing visual-inertial motion estimation system. The

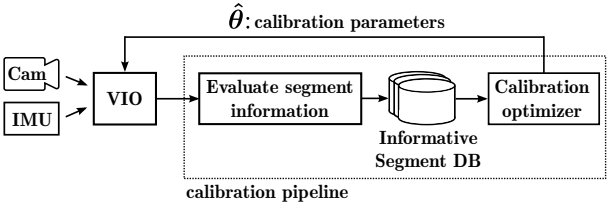


Figure 7.4: High-level overview of the modules and data flows of the proposed method: (1) motion estimates from VIO are used to identify informative motion segments, (2) the most informative segments are maintained in a database for later calibration and (3) an ML-based calibration is triggered once enough data has been collected to update the sensor calibration.

VIO implementation used in this work is described in Section 4.2 but it is important to note that the method is not tied to a particular motion estimation framework. The keyframe and landmarks states estimated by the VIO module are partitioned into segments. In a next step, the information content of each segment w.r.t. the calibration parameters is evaluated using an efficient information theoretic metric. A database maintains the most informative segments of the trajectory and a calibration is triggered once enough data has been collected. This algorithm is summarized in Alg. 3 and explained in more details in the following sections.

Algorithm 3 Self-calibration on informative motion segments.

Input: Initial calibration: $\hat{\boldsymbol{\theta}}_{init}$
Output: Updated calibration: $\hat{\boldsymbol{\theta}}$

Loop
  // Initialize motion segments of size N from VIO output.
  $\mathcal{S}_i \leftarrow \{\}$
  repeat
    $data$ = `WaitForNewSensorData()`
    $\hat{\mathbf{x}}_j, \hat{\mathbf{l}}_j \leftarrow$ `RunVIO(`$data, \hat{\theta}_{init}$`)` // Section 4.2
    $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup (\hat{\mathbf{x}}_j, \hat{\mathbf{l}}_j)$
  until $dim(\mathcal{S}_i) == N$;
  $H(\boldsymbol{\theta}) \leftarrow$ `EvaluateSegmentInformation(`$\mathcal{S}_i$`)` // Section 5.2
  `UpdateDatabase(`$\mathcal{S}_i$, $H(\boldsymbol{\theta})$`)` // Section 5.3
  if `EnoughSegmentsInDatabase()` **then**
    $\mathbf{S}_{info} \leftarrow$ `GetAllSegmentsFromDatabase()`
    $\hat{\boldsymbol{\theta}} \leftarrow$ `RunOptimization(`$\mathbf{S}_{info}$`)` // Section 5.4
    return $\hat{\boldsymbol{\theta}}$
  end
  $i \leftarrow i + 1$
EndLoop

5.2 Evaluating Information Content of Segments

The continuous stream of keyframe $\hat{\mathbf{x}}_k$ (pose, velocity, bias) and landmark states ${}_G\hat{\mathbf{l}}_m$, estimated by the VIO, is partitioned into motion segments. The i-th segment $\mathcal{S}_i$ is made up by the N consecutive keyframes $\hat{\mathcal{X}}_i = \hat{\mathbf{x}}_{(i\cdot N)..((i+1)\cdot N-1)}$ and the set of landmarks $\hat{\mathcal{L}}_i$ observed from this segment.

We propose to use information metrics that only consider the constraints within each segment to evaluate the information content w.r.t. the calibration parameters $\boldsymbol{\theta}$. Using such an information metric which is independent of all other segments makes its evaluation very efficient at the cost of neglecting cross-terms coming from other segments such as loop-closure constraints. However, the neglected constraints can be re-introduced and considered during the calibration. Thus, this assumption only affects the selection of informative segments and potentially leads to a conservative estimate of the actual information but should not bias the calibration results.

To quantify the information content of the i-th segment $\mathcal{S}_i$, we recover the marginal covariance $\boldsymbol{\Sigma}_\theta^{\mathcal{S}_i} = \mathrm{Cov}\left[p(\boldsymbol{\theta}|\mathbf{U}_i, \mathbf{Z}_i)\right]$ of the calibration parameters $\boldsymbol{\theta}$ given all the constraints within the segment. For this, we first approximate the covariance $\boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L}\theta}^{\mathcal{S}_i}$ over all segment states using the *Fisher Information Matrix* as:

$$\boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L}\theta}^{\mathcal{S}_i} = \mathrm{Cov}\left[p(\mathcal{X}_i, \mathcal{L}_i, \boldsymbol{\theta}|\mathbf{U}_i, \mathbf{Z}_i)\right] = (\mathbf{J}_i^T \mathbf{G}_i^{-1} \mathbf{J}_i)^{-1} \tag{7.24}$$

The matrix $\mathbf{J}_i$ represents the stacked Jacobians of all error terms $\mathbf{e}_k$ and $\mathbf{G}_i$ the stacked error covariances $\mathbf{W}_k$ corresponding to the errors terms as:

$$\mathbf{J}_i = \begin{bmatrix} \frac{\partial \mathbf{e}_0}{\partial \boldsymbol{\Pi}_i} \\ \vdots \\ \frac{\partial \mathbf{e}_K^T}{\partial \boldsymbol{\Pi}_i} \end{bmatrix}, \quad \mathbf{G}_i := diag\{\mathbf{W}_0 \ldots, \mathbf{W}_K\} \tag{7.25}$$

where $\boldsymbol{\Pi}_i = [\mathcal{X}_i, \mathcal{L}_i, \boldsymbol{\theta}]$ denotes the collection of all states within the segment i and K the number of errors terms within the segment i. Further, the state ordering is chosen such that the rightmost columns of $\boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L}\theta}^{\mathcal{S}_i}$ correspond to the states of the calibration parameters $\boldsymbol{\theta}$.

A rank-revealing QR decomposition is used to obtain $\mathbf{Q_i}\mathbf{R_i} = \mathbf{L}_i\mathbf{J}_i$ with $\mathbf{G}_i^{-1} = \mathbf{L}_i^T\mathbf{L}_i$ being the Cholesky decomposition of the error covariance matrix. The 7.24 can then be rewritten as

$$\boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L}\theta}^{\mathcal{S}_i} = (\mathbf{R}_i^T\mathbf{R}_i)^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L}}^{\mathcal{S}_i} & \boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L},\theta}^{\mathcal{S}_i} \\ \boldsymbol{\Sigma}_{\mathcal{X}\mathcal{L},\theta}^{\mathcal{S}_i}{}^T & \boldsymbol{\Sigma}_\theta^{\mathcal{S}_i} \end{bmatrix} \tag{7.26}$$

As $\mathbf{R}_i$ is an upper-triangular matrix, we can obtain the marginal covariance $\boldsymbol{\Sigma}_\theta^{\mathcal{S}_i}$ efficiently by back-substitution.

In a next step, we normalize the marginal covariance $\boldsymbol{\Sigma}_\theta^{\mathcal{S}_i}$ to account for different scales of the calibration parameters with:

$$\overline{\boldsymbol{\Sigma}}_\theta^{\mathcal{S}_i} = \mathrm{diag}(\boldsymbol{\sigma}_{ref})^{-1} \cdot \boldsymbol{\Sigma}_\theta^{\mathcal{S}_i} \cdot \mathrm{diag}(\boldsymbol{\sigma}_{ref})^{-1} \tag{7.27}$$

where $\boldsymbol{\sigma}_{ref}$ is the expected standard deviation that has been obtained empirically from a set of segments from various datasets. It is important to note, that $\boldsymbol{\sigma}_{ref}$ depends on the sensor setup (e.g. focal length, dimensions, etc.) and should either be re-evaluated for each setup or a normalization based on nominal calibration parameters should be performed.

We can now define different information metrics based on the normalized marginal covariance $\overline{\boldsymbol{\Sigma}}_{\theta}^{\mathcal{S}_i}$. These metrics will be used to compare segments based on their information content w.r.t. the calibration parameters $\boldsymbol{\theta}$. They are defined such that a lower value corresponds to more information. In this work, we will investigate the three most common information-theoretic metrics from optimal design theory:

A-Optimality

This criterion seeks to minimize the trace of the covariance matrix which results in a minimization of the mean variance of the calibration parameters. The corresponding information metric is defined as:

$$H_{Aopt}^{i} = \operatorname{trace}\left(\overline{\boldsymbol{\Sigma}}_{\theta}^{\mathcal{S}_i}\right) \tag{7.28}$$

D-Optimality

Minimizes the determinant of the covariance matrix which results in a maximization of the differential Shannon information of the calibration parameters.

$$H_{Dopt}^{i} = \det\left(\overline{\boldsymbol{\Sigma}}_{\theta}^{\mathcal{S}_i}\right) \tag{7.29}$$

It is interesting to note that this criterion is equivalent to the minimization of the differential entropy $H_e^i(\boldsymbol{\theta})$ which for Gaussian distributions is defined as:

$$\begin{aligned} H_e^i\left(\boldsymbol{\theta}\right) &= -\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \bar{p}_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \ln \bar{p}_{\boldsymbol{\theta}}(\boldsymbol{\theta})\, d\boldsymbol{\theta} \\ &= \frac{1}{2}\ln\left((2\pi e)^k \cdot \det\left(\overline{\boldsymbol{\Sigma}}_{\theta}^{\mathcal{S}_i}\right)\right) \end{aligned} \tag{7.30}$$

where $\bar{p}_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \bar{p}(\boldsymbol{\theta}|\mathbf{U}_i, \mathbf{Z}_i)$ is the normalized normal distribution of $\boldsymbol{\theta}$ and k the dimension of this distribution.

E-Optimality

This design seeks to minimize the maximal eigenvalue of the covariance matrix with the metric being defined as:

$$H_{Eopt}^{i} = \max\left(\operatorname{eig}\left(\overline{\boldsymbol{\Sigma}}_{\theta}^{\mathcal{S}_i}\right)\right) \tag{7.31}$$

Figure 7.5: The segment calibration problem only includes the most informative segments of the motion trajectory (keyframes) estimated by the VIO (upper graph). A constraint on the bias evolution is introduced where non-informative segments have been removed (red cross) while the pose and velocity remain unconstrained. Additionally, the segments are partitioned such that each partition co-observes less than N landmarks of other partitions. Consequently, the unobservable modes of each partition, namely the rotation around gravity and the global position, are held constant during the optimization for exactly one keyframe of the partition (marked with a square).

5.3 Collection of Informative Segments

We want to maximize the information contained within a fixed-sized budget of segments. For this reason, we maintain a database with a maximum capacity of N segments retaining only the most informative segments of the trajectory. The information metric will be used to decide which segments are retained and which are rejected such that the sum over the information metric of all segments in the database is minimized. Such a decision scheme will ensure that the accumulated information on the calibration parameter $\boldsymbol{\theta}$ is increasing over time while the number of segments remains constant. Therefore, an upper bound on the calibration problem complexity can be guaranteed. However, it is important to note that the sum of information metrics is only a conservative approximation of the total information content for two reasons: First, the information metric is only a scalar and therefore no directional information is available. Second, the information metrics neglect any cross-terms to other segments and thus underestimates the true information.

5.4 Segment Calibration Problem

The segment-based calibration differs from the batch estimator introduced in Section 4 in that it only contains the most informative segments of a (multi-session) dataset. The removal of trajectory segments from the original problem leads to two main challenges.

First, the time difference between two (temporally neighboring) keyframes could become arbitrarily large when non-informative keyframes have been removed in-between. An illustration of such a dataset with a temporal gap due to the keyframe removal is shown in Fig. 7.5 (between keyframe 6/10 and 12/16). In this case, we only constrain the bias evolution between the two neighboring keyframes using a random walk model described in Section 3.3 and no constraints are introduced for the remaining keyframe states (pose, velocity).

Second, the removal of non-informative trajectory segments often creates partitions of keyframes that are neither constrained to other partitions through (sufficient) shared landmark observations nor through inertial constraints. Each of these partitions can be seen as a (nearly) independent calibration problem that only shares the calibration states with other partitions. Assuming non-degenerate motion and sufficient visual constraints, each of these partitions contains the 2 structurally unobservable modes of the visual-inertial optimization problem namely the rotation around the gravity vector (yaw in global frame) and the global position. These modes are eliminated from the optimization by keeping them constant for exactly one keyframe in each of the partitions to achieve efficient convergence of the iterative solvers.

We identify the partitions based on the co-visibility of landmarks and the connectivity through inertial constraints. An overview of the algorithm is shown in Alg. 4. In a first step, all segments that are direct temporal neighbors, and thus connected through inertial constraints, are joined into larger segments (e.g. segment 1 and 2). In a next step, we use a union-find data structure to iteratively partition the joined segments into disjoint sets (partitions) such that the number of co-observed landmarks between the partitions lies below a certain threshold. At this point, all keyframes within a partition are either constrained through inertial measurements or through sufficient landmark co-observations w.r.t. each other. It is important to note that degenerate landmark configurations are still possible using such a heuristic metric. However, an error will only influence the convergence rate of the incremental optimization but should not bias the

Algorithm 4 Partitioning segments on landmark co-visibility

Input: Set of motion segments $\mathbf{S} = \{\mathcal{S}_0, \ldots, \mathcal{S}_K\}$
Input: Max. co-observed landmarks between partitions N
Result: Set of motion segment partitions $\mathcal{P}$

$\mathbf{P} \leftarrow \{\}$
foreach $\mathcal{S}_k \in \mathbf{S}$ **do**
  $\mathbf{C} \leftarrow \{\{\mathcal{S}_k\}\}$
  foreach $p \in \mathbf{P}$ **do**
    if `CountSharedLandmarks` $(p, \mathcal{S}_k) > N$ **then**
      $\mathbf{C} \leftarrow \mathbf{C} \cup \{p\}$
    end
  end
  $p_{\mathcal{C}} \leftarrow$ `MergePartitions`$(\mathbf{C})$
  $\mathbf{P} \leftarrow (\mathbf{P} \setminus \mathbf{C}) \cup \{p_{\mathcal{C}}\}$
end

calibration results.

6 Experimental Setup

This section introduces the experiments, datasets, and hardware used to evaluate the proposed method. The results are discussed in the next section.

6.1 Single-/Multi Session Database

We evaluate the proposed method using two different strategies to maintain informative segments in the database. Each strategy is investigated using a set of multi-session datasets and discussed along a suitable use-case:

Single-session Database: Observability-aware Sparsification of Calibration Datasets

Each session starts with an empty segment database and the N most informative segments from this single session are kept. After each session, a segment-based calibration is performed using all the segments in the database and the calibration parameters are updated for use in the next session. This strategy can be seen as an observability-aware sparsification method for calibration datasets. It is well suited for infrequent and long sessions (e.g. *navigation use-case* with lots of still phases) where batch calibration over the entire dataset would be too expensive and data selection is necessary.

Multi-session Database: Accumulation of Information over Time

The multi-session strategy does not reset the database between sessions and the most informative segments are collected from multiple consecutive sessions. In contrast to the single-session strategy, it is particularly suited for frequent and short sessions; for example in an *AR/VR use-case* where a user performs many short session over a short period of time. It accumulates in-

formation from multiple sessions and thus enables the calibration of weakly observable modes which might not be sufficiently excited in a single session.

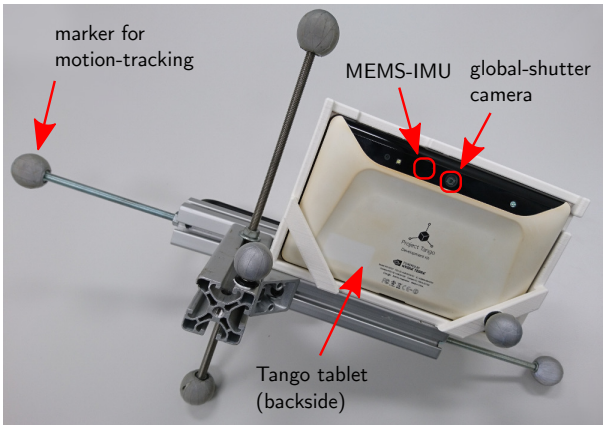


Figure 7.6: The Google Tango tablet used for the dataset collection is equipped with markers for external pose tracking by a Vicon motion-capture system. The tablet contains a sensor suite specifically designed for motion tracking including a high field-of-view camera and a single-chip MEMS IMU.

6.2 Datasets and Hardware

All datasets were recorded using a Google Tango tablet as shown in Fig. 7.6. This device uses a high-field-of-view global shutter camera (10 Hz) and a single-chip MEMS IMU (100 Hz). The measurements of both sensors are time-stamped in hardware on a single clock for an accurate synchronization. Additionally, the sensor rig is equipped with markers for external tracking by a Vicon motion capture system. All datasets were recorded on the same device, in a short period of time and while trying to keep the environmental factors constant (e.g. temperature) to minimize potential variations of the calibration parameters across the datasets and sessions.

We have collected datasets representative for each of the two use-cases introduced in the previous section in different environments (office, class room, and garage). These datasets consist of multiple sessions that will be used to obtain a calibration using the proposed method. Right after recording the calibration datasets, we have collected a batch of 15 evaluation datasets with motion capture ground-truth. These datasets are used to evaluate the motion estimation accuracy that can be achieved using the obtained calibration parameters. An overview of all datasets and their characteristics is shown in Table 7.2 and Fig. 7.7.

While recording the calibration datasets, we tried to achieve the following characteristics representative for the two use-cases:

Table 7.2: Datasets used for the evaluation. All datasets have been recorded using a Google Tango tablet as shown in Fig. 7.6.

dataset	avg. length duration	avg. linear / angular vel.	description
AR/VR use-case:			
office room	23.8 m	0.20 m/s	well-lit, good
(5 sessions)	117.3 s	20.3 deg/s	texture
class room	37.4 m	0.29 m/s	well-lit, open space,
(5 sessions)	122.9 s	29.62 deg/s	good texture
Navigation use-case:			
parking garage	168.4 m	0.57 m/s	dark, low-texture
(3 sessions)	305.0 s	20.51 deg/s	walls, open space
office building	164.8 m	0.55 m/s	well-lit, good
(3 sessions)	295.6 s	23.12 deg/s	texture, corridors
Evaluation datasets:			
Vicon room	59.7 m	0.49 m/s	motion-capture data,
(15 sessions)	114.1 s	42.95 deg/s	well-lit

AR/VR use-case

We collected datasets that mimic an AR/VR use-case to evaluate whether we can accumulate information from multiple-sessions (multi-session database strategy). Characteristic of this use-case, the datasets consists of multiple short sessions restricted to a small indoor space (single room), containing mostly fast rotations, only slow and minor translation and stationary phases. Two datasets have been recorded in a *class* and *office* room each containing 5 sessions that are 2 min long.

Navigation use-case

In contrast to the AR/VR use-case, the navigation sessions contain mostly translation over an area of multiple rooms and only slow rotations but also contain stationary and rotation-only phases. Datasets have been recorded in two locations: *garage* and *office* - each contains 3 sessions with a duration of 5 min. These datasets will be used to evaluate the observability-aware sparsification (single-session database strategy).

6.3 Evaluation Method

For performance evaluation, we calibrate the sensor models on each session of the dataset in temporal order where we use the calibration parameters obtained from the previous session as initial values. The first session uses a nominal calibration consisting of a relative pose between camera and IMU from CAD values, nominal values for the IMU intrinsics (unit scale factors, no axis misalignment) and camera intrinsics.

(a) AR/VR: office room

(b) AR/VR: class room

(c) NAV: parking garage

(d) NAV: office building

Figure 7.7: The four different environments in which the calibration datasets have been recorded. The images were taken by the motion tracking camera of the Tango tablet.

This calibration scheme is performed for all datasets and for both of the database strategies to obtain a set of calibration parameters for each session. The quality of the obtained calibration parameters is then evaluated using the following methods:

Motion estimation performance

As the main objective of our work is to calibrate the sensor system for ego-motion estimation, we use the accuracy of the motion estimation (based on our calibrations) as the main evaluation metric. We run all 15 evaluation datasets for each set of calibration parameters and evaluate the accuracy of the estimated trajectory against the ground-truth from the motion-capture system.

The motion estimation error is obtained by first performing a spatio-temporal alignment of the estimated and the ground-truth trajectory. Second, a relative pose error is computed at each time-step between the two trajectories. To compare different runs, we use the root-mean-square error (RMSE) calculated over all the relative pose errors.

Parameter repeatability

We only evaluate the parameter repeatability over different calibrations of the same device as no ground-truth for the calibration parameters is available. We have recorded all dataset close in time while keeping the environmental conditions (e.g. temperature) similar and avoiding any shocks to minimize potential variations of the calibration parameters between the datasets.

7 Results and Discussion

In this section, we discuss the results of our experiments (Section 6) along the following questions:

- Section 7.1: How accurate are motion estimates based on calibrations derived only from informative segments? How does it compare to the non-sparsified (batch) calibration?
- Section 7.2: Does the sparsified calibration yield similar calibration parameters to the (full) batch problem?
- Section 7.4: Can we accumulate informative segments from multiple sessions and perform a calibration where the individual session would not provide enough excitation for a reliable calibration?
- Section 7.5: How does the proposed method compare against an EKF approach that jointly estimates motion and calibration parameters?
- Section 7.3: How do the three different information metrics compare? Can we outperform random selection of segments?
- Section 7.6: What segments are being selected as informative? What are their properties?
- Section 7.7: How do we select the number of segments to retain in the database?

7.1 Motion Estimation Performance using the Observability-aware Sparsification (Single-session Database)

In this experiment, we use a database of 8 segments (4 seconds each) which leads to a reduction of the sessions size by around 75% in the *AR/VR use-case* and 90% in the *navigation use-case*. To evaluate the observability-aware sparsification, we select the most informative segments for all sessions of a dataset independently. A segment-based calibration is then run over the selected segments to obtain an updated set of calibration parameters for each session. Finally, the VIO motion estimation accuracy is evaluated for each calibration on all of the 15 evaluation datasets as described in Section 6.3. The resulting statistics of the RMSE are shown in Table 7.3 for each dataset. The mean of rotation states corresponds to the rotation angle of the averaged quaternion

Table 7.3: Comparison of the motion estimation accuracy evaluated on VIO estimates when run with different calibration strategies. The errors are shown for calibrations obtained from a sparsified problem (8 segments, 4 seconds each) for three different information metrics and random selection as a baseline. For reference, the errors are given for the batch calibration (no sparsification), the initial calibration and a related EKF-based approach that uses the same VIO estimator but jointly estimates the calibration, the motion and scene structure (similar to [56]). All values show the median and standard deviation of the RMSE over all evaluation datasets using the calibration under investigation.

RMSE on VIO trajectory vs. motion capture ground-truth (translation [cm] / rotation [deg])							
	initial calibration	no sparsification (batch)	sparsified (8 segments, each 4 seconds)				joint EKF
			E-optimality	D-optimality	A-optimality	random	
AR/VR: office room	13.07 ± 9.10 cm	**1.50 ± 0.89 cm**	1.62 ± 0.60 cm	1.76 ± 0.59 cm	1.79 ± 0.62 cm	3.99 ± 2.49 cm	1.86 ± 1.17 cm
	1.18 ± 0.60 deg	0.47 ± 0.26 deg	**0.34 ± 0.12 deg**	0.37 ± 0.13 deg	0.35 ± 0.13 deg	0.64 ± 0.34 deg	0.49 ± 0.27 deg
AR/VR: class room	13.09 ± 9.13 cm	1.41 ± 1.07 cm	1.79 ± 0.75 cm	**1.28 ± 0.54 cm**	1.42 ± 0.57 cm	5.45 ± 5.81 cm	2.44 ± 1.71 cm
	1.17 ± 0.59 deg	0.46 ± 0.25 deg	0.35 ± 0.12 deg	**0.34 ± 0.12 deg**	0.35 ± 0.12 deg	0.77 ± 0.54 deg	0.52 ± 0.32 deg
NAV: parking garage	13.09 ± 9.13 cm	4.66 ± 34.73 cm	1.65 ± 0.56 cm	2.14 ± 1.03 cm	**1.59 ± 0.59** cm	4.97 ± 3.56 cm	3.04 ± 1.81 cm
	1.17 ± 0.59 deg	0.57 ± 0.62 deg	**0.31 ± 0.11** deg	0.38 ± 0.14 deg	**0.31 ± 0.11 deg**	0.73 ± 0.43 deg	0.55 ± 0.29 deg
NAV: office building	13.13 ± 9.17 cm	1.86 ± 1.17 cm	1.68 ± 0.62 cm	1.39 ± 0.49 cm	**1.26 ± 0.45 cm**	2.32 ± 1.18 cm	2.56 ± 1.60 cm
	1.16 ± 0.57 deg	0.51 ± 0.27 deg	0.41 ± 0.14 deg	**0.34 ± 0.12 deg**	0.35 ± 0.12 deg	0.50 ± 0.27 deg	0.60 ± 0.35 deg

as described in [61] and the standard deviation is derived from rotation angles between the samples and the averaged quaternion. For comparison, the same evaluations have been performed for the initial and batch calibration (no sparsification).

The calibrations obtained with the sparsified dataset yield very similar motion estimation performance when compared to full batch calibrations. This indicates that the proposed method can indeed sparsify the calibration problem while retaining the relevant portion of the dataset and still provide a calibration with motion estimation performance close to the non-sparsified problem. It is interesting to note, that the sparsification to a fixed number of segments keeps the calibration problem complexity bounded while the complexity of the batch problem is (potentially) unbounded when used on large datasets with redundant and non-informative sections.

7.2 Repeatability of Estimated Calibration Parameter

As we have no ground-truth for the calibration parameters, we can only evaluate their repeatability across multiple calibrations of the same device. The statistics over all calibration parameters obtained with all sessions of the *class room* datasets are shown in Table 7.4. We used the same sparsification parameters as in Section 7.1 (8 segments, each 4 seconds).

The experiments show that the deviation between the full-batch and sparsified solution remain insignificant in mean and standard deviation even though 75% of the trajectory has been removed. This is a good indication that the sparsified calibration problem is a good approximation to the complete problem.

7.3 Comparison of Information Metrics

In Section 5.2, we have proposed three different information metrics to compare trajectory segments for their information w.r.t. to the calibration parameters. The same evaluation performed for the sparsification use-case (Section 7.1) has been repeated for each of the proposed metrics and, as a baseline, also for calibrations based on randomly selected segments. The motion estimation errors based on these calibration is reported in Table 7.3.

The motion estimation error is around 2-3 times larger when randomly selecting the same amount of data indicating that the proposed metrics successfully identify informative segments for calibration. It is important to note, that this comparison heavily depends on the ratio of informative / non-informative motion in the dataset and therefore this error might be larger when there is less excitation in a given dataset. In general, all three metrics show comparable performance, however, the A-optimality criteria performed slightly better on the *navigation* and the D-optimality on the *AR/VR use-case*.

7.4 Accumulation of Information over Time: Single- vs Multi-session Database

In this section, we evaluate whether the proposed method can accumulate informative segments from multiple consecutive sessions to obtain a better and more consistent calibration than the individual session would yield. This is especially important in scenarios where a single session often would not provide enough excitation for a reliable calibration. The evaluations were performed on the *AR/VR use-case* datasets which consist of multiple short sessions. We use

Table 7.4: Mean and standard deviation of the estimated calibration parameters for the sparsified calibration problem (8 segments, 4 seconds), the batch solution and the final estimate of the joint EKF run on the complete dataset. The statistics have been derived from the calibrations obtained on all session of the *AR/VR use-case* dataset. The joint EKF only estimates the IMU intrinsics and the camera-IMU relative pose, therefore no values are given for the camera intrinsics.

parameter	**proposed method** (sparsified)	**batch** (complete dataset)	**joint EKF** (complete dataset)
$\mathbf{f}$ [px]	255.79 ± 0.60	256.30 ± 0.22	-
	255.68 ± 0.67	256.31 ± 0.27	-
$\mathbf{c}$ [px]	313.63 ± 0.67	313.19 ± 0.63	-
	241.62 ± 1.17	243.16 ± 0.18	-
w [-]	0.9203 ± 0.0009	0.9208 ± 0.0008	-
$\mathbf{s}_g - 1$ [-]	-2.82e-03 ± 1.32e-03	-2.11e-03 ± 2.27e-04	2.39e-03 ± 2.06e-03
	4.33e-03 ± 4.83e-03	4.02e-03 ± 2.70e-04	7.71e-03 ± 3.08e-03
	-1.21e-03 ± 5.18e-04	-1.54e-03 ± 4.18e-04	2.61e-03 ± 3.90e-03
$\mathbf{s}_a - 1$ [-]	-9.70e-03 ± 1.50e-02	-1.85e-02 ± 3.07e-03	-1.64e-02 ± 6.54e-03
	-1.16e-02 ± 1.17e-02	-1.65e-02 ± 1.19e-03	-1.24e-02 ± 5.59e-03
	-1.95e-02 ± 7.38e-03	-1.86e-02 ± 1.48e-03	-1.34e-02 ± 2.43e-03
$\mathbf{m}_g$ [-]	-3.22e-04 ± 1.69e-03	7.36e-04 ± 6.56e-04	1.03e-03 ± 8.78e-04
	2.37e-03 ± 1.95e-03	3.96e-04 ± 2.30e-04	-7.36e-04 ± 1.32e-03
	-6.78e-04 ± 1.60e-03	-4.95e-05 ± 1.17e-03	-9.82e-04 ± 1.77e-03
$\gamma(\mathbf{q}_{GA})$ [deg]	1.897 ± 0.428	1.504 ± 0.010	1.368 ± 0.150
$\mathbf{m}_a$ [-]	2.11e-02 ± 1.11e-02	1.35e-02 ± 1.54e-03	1.68e-02 ± 5.05e-03
	-3.68e-02 ± 1.11e-02	-2.78e-02 ± 2.59e-03	-2.76e-02 ± 6.78e-03
	-7.93e-03 ± 9.30e-03	-3.19e-03 ± 1.21e-03	-7.92e-04 ± 2.99e-03
${}_C\mathbf{p}_{IC}$ [m]	1.06e-03 ± 4.01e-03	4.93e-03 ± 2.33e-03	5.43e-03 ± 3.68e-03
	4.62e-03 ± 1.86e-02	7.05e-04 ± 2.17e-03	4.09e-04 ± 2.85e-03
	-1.48e-02 ± 1.12e-02	-6.09e-03 ± 4.08e-03	-1.19e-02 ± 6.77e-03
$\gamma(\mathbf{q}_{IC})$ [deg]	1.174 ± 0.133	1.065 ± 0.071	0.753 ± 0.069

the A-optimality criteria to select the most informative segments of each sessions and maintain them in the database (8 segments, 4 seconds). In contrast to the sparsification use-case from Section 7.1, the database is not reset between the sessions. In other words, the database will collect the N most informative segments from the first up to the current session. After each session, a calibration is triggered using all segments of the database. These calibrations are then used to evaluate the motion estimation error on all 15 evaluation datasets. The results are shown in Fig. 7.8 for the *class room* dataset.

The evaluation shows that the motion estimation error decreases as the number of sessions increases (from which informative segments have been selected). Further, the motion estimation error is smaller when compared to calibrations based on the most informative segments from individual sessions. After around 2 sessions the estimation performance is close to what would be achieved using a batch calibration. This indicates that the proposed method can accumulate information from multiple sessions while the number of segments in the database remains constant. It can therefore provide a reliable calibration when a single session would not provide enough excitation.

7.5 Comparison vs. joint EKF

In this section, we compare the proposed method against an EKF filter that jointly estimates the motion, scene structure and the calibration parameters (similar to [56]). In our implementation, we only estimate the IMU intrinsics and the relative pose between the camera and IMU. The camera intrinsics are not estimated and set to parameters obtained with a batch calibration on the same dataset.

We evaluated the motion estimation errors on all datasets and report the results in Table 7.3. The resulting calibration parameters are compared to the proposed method and batch solution in Table 7.4. The evaluations show a position error that is up to 2 times larger compared to calibrations obtained with the proposed method or a batch calibration. When looking at the state evolution of e.g. the misalignment factors, as shown in Fig. 7.9 for one of the datasets, it can be seen that it converges roughly to the batch estimate but does not remain stable over time. We see this as an indication that the local scope of the EKF is not able to infer weakly observable states properly and thus a segment-based (sliding-window) approach is beneficial in providing a stable and consistent solution over time.

7.6 Selected Informative Segments

In this section, we investigate the motion that is being selected as informative by the proposed method. Fig. 7.10 shows the 8 most informative segments that have been selected in one of the session of the *navigation use-case*. We only show the first minute of the session as otherwise the trajectory would start to overlap. It can be seen that the information metric correlates with changes in linear and rotational velocity and therefore mostly segments containing turns have been selected while straight segments have been found to be less informative. This experiment seems to confirm the intuition that segments with larger accelerations and rotational velocities are more informative for calibration.

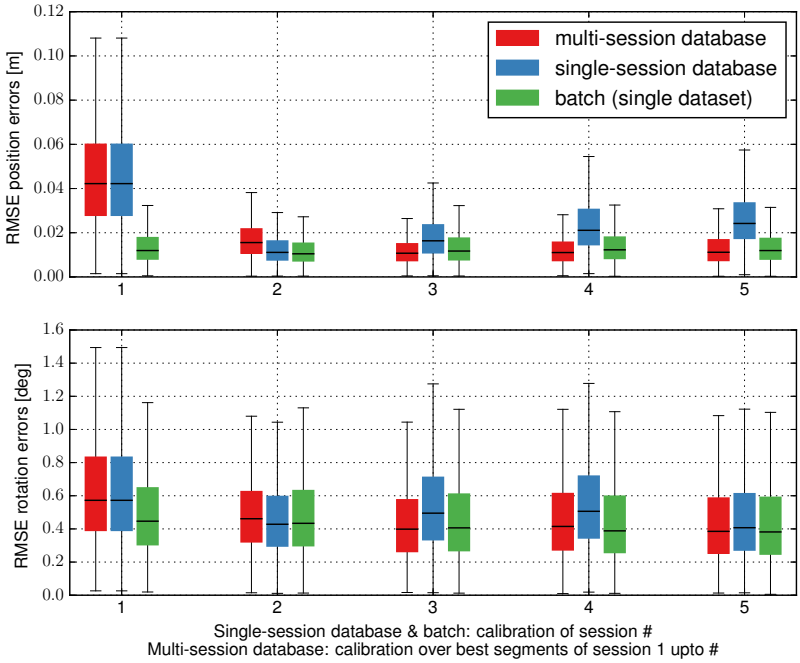


Figure 7.8: Comparing the VIO motion estimation RMSE for calibrations obtained with two different database strategies. A fixed number of the most informative segments (8 segments each 4 seconds) have been collected either: (a) incrementally over all datasets (multi-session: Section 6.1), or (b) only from a single dataset (single-session: Section 6.1). The motion estimation errors have been evaluated for all obtained calibrations based on these segments. For example, the calibration of session 3 (x=3) and method (a), in red, is based on the 8 most informative segments from the sessions 1-3 and for method (b), in blue, on the 8 most informative segments from session 3 alone. The batch solution (green) uses all segments of a single dataset.

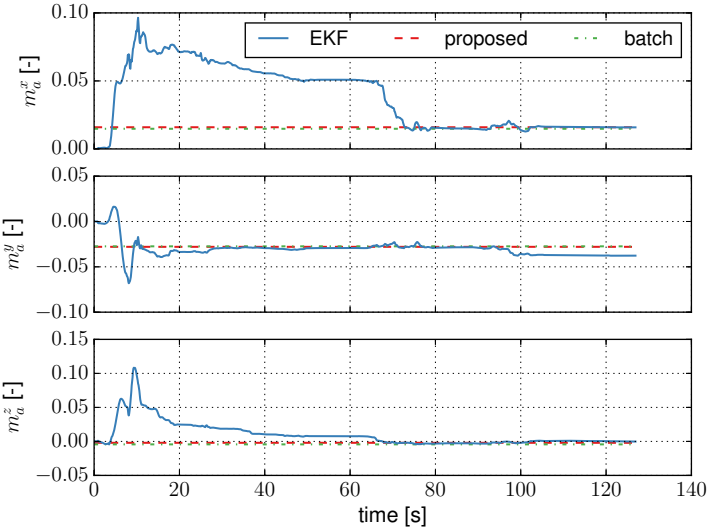


Figure 7.9: Misalignment of the gyroscope axis $\boldsymbol{m}_g$ estimated by the EKF on one of the sessions in the *class room* dataset. The EKF jointly estimate the motion, structure and calibration parameters in a single filter. For comparison, the estimates obtained with the proposed method and the batch estimator are shown.

7.7 Influence of Database Size on the Calibration Quality

In this experiment, we investigate the effect of the database size on the calibration quality to find the minimum amount of data required for a reliable calibration. We sparsify all sessions of all datasets repeatably to retain 1 to 15 of the most informative segments. A segment-based calibration is then run on each of the sparsified datasets and the motion estimation error is evaluated on all evaluation datasets. The segment duration was chosen as 4 seconds from geometrical considerations such that segments span a sufficiently large distance for landmark triangulation with the assumption that the system moves at a steady walking speed. The median of the RMSE over all evaluation datasets is shown in Fig. 7.11.

The motion estimation error seems to stabilize when using more than $7-8$ segments. Based on these experiments, we have selected a database size of 8 segments as a reasonable trade-off between calibration complexity and quality and used this value for all the evaluations in this work. It is important to note, that the amount of data required for a reliable calibration depends on the sensor models, the expected motion and the environment and a re-evaluation might become necessary if these parameters change. In future work, we plan to investigate methods to determine the information content of the database directly to avoid a selection of

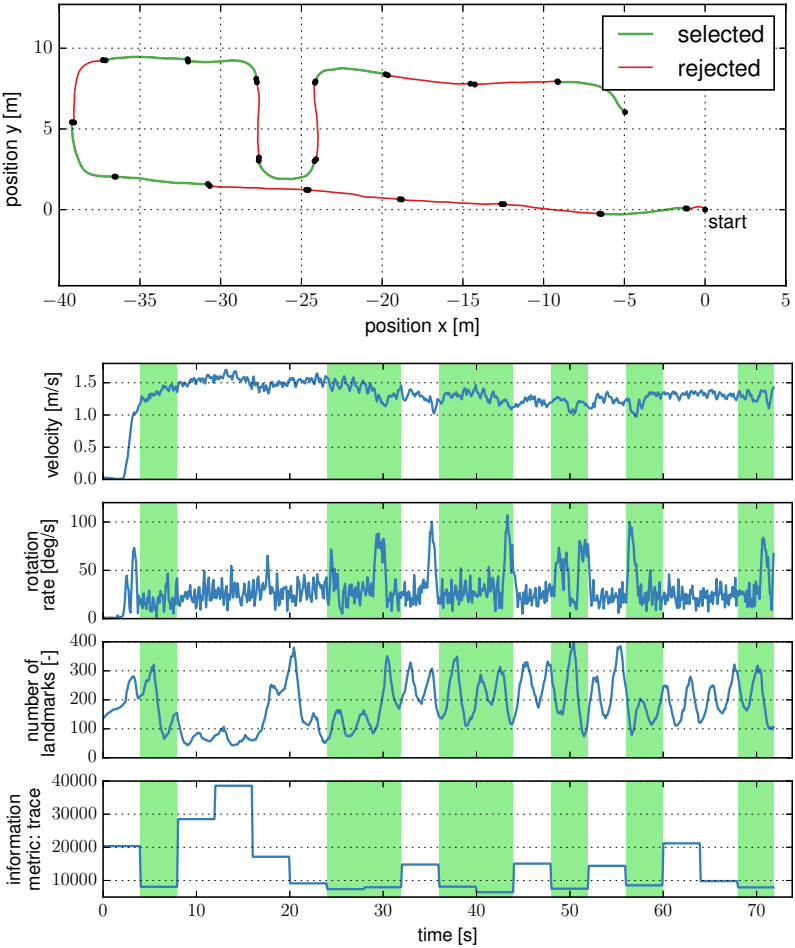


Figure 7.10: The 8 most informative segments identified using the A-optimality criteria in one of the sessions of the *navigation use-case* (a lower value indicates more information). The metric correlates with changes in the linear and rotational velocity and therefore mostly segments during turns have been selected whereas the straight segments were found to be less informative.

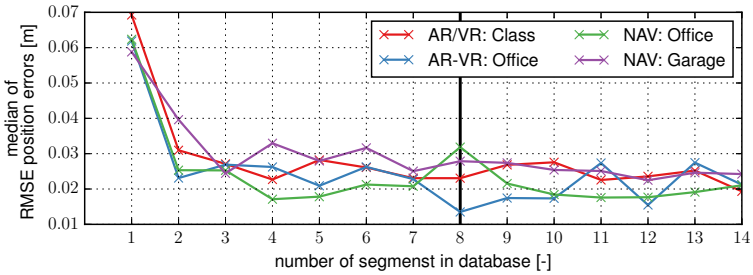


Figure 7.11: Median of the motion estimation error for different levels of calibration datasets sparsification. The error seems to stabilize when using more than $7-8$ segments and we found that 8 segments provides a reasonable trade-off between complexity and quality.

Table 7.5: Evaluation of the run-time for the proposed method, batch estimator and joint-EKF obtained while running the experiments of Section 7.1. The run-time of the batch calibration is unbounded as the calibration dataset increase. The run-time of the proposed method, however, only depends on the number of collected informative segments and therefore has an upper bound.

	proposed method	**batch**	**joint EKF**
VIO (each image)	0.003 s	-	0.003 s
Data selection (each segment)	0.156 s	-	-
Calibration (each dataset)	12.050 s	27.028 s	-

this parameter.

7.8 Run-time

Table 7.5 reports the measured run-times of the proposed method and the batch calibration for the experiments of Section 7.1. Both optimizations use the same number of steps and the same initial conditions.

It is important to note, that the complexity and thus run-time of the batch method is unbounded when the duration of the sessions increase. The run-time of the proposed method, however, remains constant as we only include a constant amount of informative data. This property makes the proposed method well-suited for systems performing long sessions.

8 Conclusion

We have proposed an efficient self-calibration method for visual and inertial sensors which runs in parallel to an existing motion estimation framework. In a background process, an information-theoretic metric is used to quantify the information content of motion segments and a fixed number of the most informative are maintained in a database. Once enough data has been collected, a segment-based calibration is triggered to update the calibration parameters. With this method, we are able to collect exciting motion in a background process and provide reliable calibration with the assumption that such motion occurs eventually - making this method well-suited for consumer devices where the users often do not know how to excite the system properly.

An evaluation on motion capture ground-truth shows that the calibrations obtained with the proposed method achieve comparable motion estimation performance to full batch calibrations. However, we can limit the computational complexity by only considering the most informative part of a dataset and thus enable calibration even on long sessions and resource-constrained platforms where a full-batch calibration would be unfeasible. Further, our evaluations show that we can not only sparsify single-session datasets but also accumulate information from multiple sessions and thus perform reliable calibrations when a single-session would not provide enough excitation. The comparison of three information metrics indicates that A-optimality could be selected for navigation purposes while D-optimality looks like a good compromise for AR/VR applications.

In future work, we would like to investigate methods to dynamically determine the segment boundaries instead of using a fixed segment length and also account for temporal variations in the calibration parameters by detecting and removing outdated segments from a database.

Acknowledgements

We would like to thank Konstantine Tsotsos, Michael Burri and Igor Gilitschenski for the valuable discussions and inputs. This work was partially funded by Google's Project Tango.

Bibliography

[1] S. Agarwal, K. Mierle, and Others. Ceres solver. http://ceres-solver.org, 2019. [Online; accessed 29-Jan-2019].

[2] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 510–517, June 2012.

[3] J. Alves, J. Lobo, and J. Dias. Camera-inertial sensor modelling and alignment for visual navigation. *Machine Intelligence and Robotic Control*, 5(3):103–112, 2003.

[4] A. Babenko and V. Lempitsky. The inverted multi-index. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 3069–3076, June 2012.

[5] R. Bähnemann, M. Burri, E. Galceran, R. Siegwart, and J. Nieto. Sampling-based motion planning for active multirotor system identification. In *IEEE Int. Conf. on Robotics and Automation*, pages 3931–3938, 2017.

[6] F. Blochliger, M. Fehr, M. Dymczyk, T. Schneider, and R. Siegwart. Topomap: Topological mapping and navigation based on visual slam maps. *IEEE Int. Conf. on Robotics and Automation*, pages 1–9, May 2018.

[7] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart. Robust visual inertial odometry using a direct EKF-based approach. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 298–304, 2015.

[8] M. Bosse, P. Newman, J. Leonard, and S. Teller. Simultaneous Localization and Map Building in Large-Scale Cyclic Environments Using the Atlas Framework. *The Int. Journal of Robotics Research*, 2004.

[9] J.-Y. Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 2001.

[10] M. Burri, H. Oleynikova, M. W. Achtelik, and R. Siegwart. Real-time visual-inertial mapping, re-localization and planning onboard mavs in unknown environments. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1872–1878, Sep. 2015.

[11] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The euroc micro aerial vehicle datasets. *IJRR*, 35(10):1157–1163, Sept. 2016.

[12] M. Bürki, I. Gilitschenski, E. Stumm, R. Siegwart, and J. Nieto. Appearance-based landmark selection for efficient long-term visual localization. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 4137–4143, Oct 2016.

[13] M. Bürki, M. Dymczyk, I. Gilitschenski, C. Cadena, R. Siegwart, and J. Nieto. Map management for efficient long-term visual localization in outdoor environments. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 682–688, June 2018.

[14] W. Churchill and P. Newman. Experience-based navigation for long-term localisation. *The Int. Journal of Robotics Research*, 2013.

[15] T. Cieslewski, S. Lynen, M. Dymczyk, S. Magnenat, and R. Siegwart. Map api - scalable decentralized map building for robots. In *IEEE Int. Conf. on Robotics and Automation*, pages 6241–6247, May 2015.

[16] M. Cummins and P. Newman. Appearance-only slam at large scale with fab-map 2.0. *The Int. Journal of Robotics Research*, 30(9):1100–1123, 2011.

[17] F. Dellaert. Factor Graphs and GTSAM: A Hands-on Introduction. Technical report, Georgia Institute of Technology, 2012.

[18] F. Devernay and O. Faugeras. Straight lines have to be straight. *Machine vision and applications*, 13(1):14–24, 2001.

[19] M. Dymczyk, S. Lynen, M. Bosse, and R. Siegwart. Keep it brief: Scalable creation of compressed localization maps. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2536–2542, Sep. 2015.

[20] M. Dymczyk, S. Lynen, T. Cieslewski, M. Bosse, R. Siegwart, and P. Furgale. The gist of maps - summarizing experience for lifelong localization. In *IEEE Int. Conf. on Robotics and Automation*, pages 2767–2773, May 2015.

[21] M. Dymczyk, I. Gilitschenski, R. Siegwart, and E. Stumm. Map summarization for tractable lifelong mapping. In *RSS Workshop*, 2016.

[22] M. Dymczyk, T. Schneider, I. Gilitschenski, R. Siegwart, and E. Stumm. Erasing bad memories: Agent-side summarization for long-term mapping. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 4572–4579, Oct 2016.

[23] M. Dymczyk, M. Fehr, T. Schneider, and R. Siegwart. Long-term large-scale mapping and localization using maplab. *Workshop International Conference on Robotics and Automation (ICRA)*, 2018.

[24] J. Elseberg, S. Magnenat, R. Siegwart, and A. Nüchter. Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics (JOSER)*, pages 2–12, 2012.

[25] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular slam. In *European Conf. on Computer Vision*, pages 834–849, Cham, 2014. Springer.

[26] C. Estrada, J. Neira, and J. D. Tardos. Hierarchical slam: real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596, Aug 2005.

[27] P. Fankhauser, M. Bloesch, P. Krüsi, R. Diethelm, M. Wermelinger, T. Schneider, M. Dymczyk, M. Hutter, and R. Siegwart. Collaborative navigation for flying and walking robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2859–2866, Oct 2016.

[28] M. Fehr, M. Dymczyk, S. Lynen, and R. Siegwart. Reshaping our model of the world over time. In *IEEE Int. Conf. on Robotics and Automation*, pages 2449–2455, May 2016.

[29] M. Fehr, T. Schneider, M. Dymczyk, J. Sturm, and R. Siegwart. Visual-inertial teach and repeat for aerial inspection. *Workshop International Conference on Robotics and Automation (ICRA)*, 2018.

[30] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza. Collaborative monocular slam with multiple micro aerial vehicles. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2013.

[31] C. Forster, M. Pizzoli, and D. Scaramuzza. Air-ground localization and map augmentation using monocular dense reconstruction. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3971–3978, Nov 2013.

[32] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *IEEE Int. Conf. on Robotics and Automation*, pages 15–22, May 2014.

[33] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Robotics: Science and Systems*, 2015.

[34] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1280–1286, 2013.

[35] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1434–1441, June 2010.

[36] M. Gehrig, E. Stumm, T. Hinzmann, and R. Siegwart. Visual place recognition with probabilistic voting. In *IEEE Int. Conf. on Robotics and Automation*, pages 3192–3199, May 2017.

[37] G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010. [Online; accessed 29-Jan-2019].

[38] C. Guo, K. Sartipi, R. DuToit, G. Georgiou, R. Li, J. O'Leary, E. Nerurkar, J. Hesch, and S. Roumeliotis. Large-scale cooperative 3d visual-inertial mapping in a manhattan world. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1071–1078, 2015.

[39] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge, UK; New York, 2003.

[40] K. Hausman, J. Preiss, G. S. Sukhatme, and S. Weiss. Observability-aware trajectory optimization for self-calibration with application to uavs. *IEEE Robotics and Automation Letters*, 2017.

[41] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis. Camera-imu-based localization: Observability analysis and consistency improvement. *The Int. Journal of Robotics Research*, 33(1):182–201, 2014.

[42] T. Hinzmann, T. Schneider, M. Dymczyk, A. Melzer, T. Mantel, R. Siegwart, and I. Gilitschenski. Robust map generation for fixed-wing uavs with low-cost highly-oblique monocular cameras. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3261–3268, Oct 2016.

[43] T. Hinzmann, T. Schneider, M. Dymczyk, A. Schaffner, S. Lynen, R. Siegwart, and I. Gilitschenski. Monocular visual-inertial SLAM for fixed-wing UAVs using sliding window based nonlinear optimization. In *IEEE International Symposium On Visual Computing*, pages 569–581, Cham, 2016. Springer.

[44] C. Hughes, P. Denny, M. Glavin, and E. Jones. Equidistant fish-eye calibration and rectification by vanishing point extraction. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(12):2289–2296, 2010.

[45] M. Hutter, C. Gehring, M. Bloesch, M. A. Hoepflinger, C. D. Remy, and R. Siegwart. StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion. In *CLAWAR*, 2012.

[46] N. Keivan and G. Sibley. Constant-time monocular self-calibration. In *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 1590–1595, Dec 2014.

[47] J. Kelly and G. S. Sukhatme. Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The Int. Journal of Robotics Research*, 30(1):56–79, 2011.

[48] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *IEEE Int. Symp. On Mixed and Augmented Reality*, pages 225–234, 2007.

[49] D. G. Kottas, K. J. Wu, S. Roumeliotis, et al. Detecting and dealing with hovering maneuvers in vision-aided inertial navigation systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3172–3179, Nov 2013.

[50] C. Krebs and J. Rehder. Generic imu-camera calibration algorithm. *Autonomous Systems Lab, ETH Zurich, Tech. Rep*, 2012.

[51] S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Int. Conf. on Computer Vision*, pages 2548–2555, Nov 2011.

[52] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The Int. Journal of Robotics Research*, 34(3):314–334, 2015.

[53] M. Li and A. I. Mourikis. High-precision, consistent EKF-based visual–inertial odometry. *The Int. Journal of Robotics Research*, 32(6):690–711, 2013.

[54] M. Li and A. I. Mourikis. Optimization-based estimator design for vision-aided inertial navigation. *Robotics: Science and Systems*, 2013.

[55] M. Li and A. I. Mourikis. Online temporal calibration for camera–imu systems: Theory and algorithms. *The Int. Journal of Robotics Research*, 33(7):947–964, 2014.

[56] M. Li, H. Yu, X. Zheng, and A. I. Mourikis. High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation. In *IEEE Int. Conf. on Robotics and Automation*, pages 409–416, 2014.

[57] J. Lobo and J. Dias. Relative pose calibration between visual and inertial sensors. *The Int. Journal of Robotics Research*, 26(6):561–575, 2007.

[58] S. Lynen, M. Bosse, P. Furgale, and R. Siegwart. Placeless place-recognition. In *Proceedings of International Conference on 3D Vision (3DV)*, volume 1, pages 303–310, Dec 2014.

[59] S. Lynen, T. Sattler, M. Bosse, J. Hesch, M. Pollefeys, and R. Siegwart. Get out of my lab: Large-scale, real-time visual-inertial localization. In *Robotics: Science and Systems*, 2015.

[60] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *European Conf. on Computer Vision*, pages 183–196. Springer, Berlin, Heidelberg, 2010.

[61] F. L. Markley, Y. Cheng, J. L. Crassidis, and Y. Oshman. Averaging quaternions. *Journal of Guidance, Control, and Dynamics*, 30(4):1193–1197, 2007.

[62] J. Maye, P. Furgale, and R. Siegwart. Self-supervised calibration for robotic systems. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 473–480, 2013.

[63] J. McDonald, M. Kaess, C. Cadena, J. Neira, and J. Leonard. Real-time 6-DOF multi-session visual SLAM over large-scale environments. *Robotics and Autonomous Systems*, 61(10):1144 – 1158, 2013.

[64] H. Merzić, E. Stumm, M. Dymczyk, R. Siegwart, and I. Gilitschenski. Map quality evaluation for visual localization. In *IEEE Int. Conf. on Robotics and Automation*, pages 3200–3206, May 2017.

[65] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, et al. Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, 2012.

[66] S. Middelberg, T. Sattler, O. Untzelmann, and L. Kobbelt. Scalable 6-dof localization on mobile devices. *European Conf. on Computer Vision*, pages 268–283, 2014.

[67] F. M. Mirzaei and S. I. Roumeliotis. A Kalman filter-based algorithm for imu-camera calibration: Observability analysis and performance evaluation. *IEEE Transactions on Robotics*, 24(5):1143–1156, 2008.

[68] P. Moulon, P. Monasse, R. Perrot, and R. Marlet. Openmvg: Open multiple view geometry. In *Reproducible Research in Pattern Recognition*, volume 10214, Cham, 2016. Springer.

[69] A. I. Mourikis and S. I. Roumeliotis. A multi-state constraint kalman filter for vision-aided inertial navigation. In *IEEE Int. Conf. on Robotics and Automation*, pages 3565–3572, 2007.

[70] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, Oct 2017.

[71] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[72] P. Mühlfellner, M. Bürki, M. Bosse, W. Derendarz, R. Philippsen, and P. Furgale. Summary maps for lifelong visual localization. *Journal of Field Robotics*, 33(5):561–590, 2016.

[73] E. D. Nerurkar, K. J. Wu, and S. I. Roumeliotis. C-klam: Constrained keyframe-based localization and mapping. In *IEEE Int. Conf. on Robotics and Automation*, pages 3638–3643, 2014.

[74] J. Nikolic, J. Rehder, M. Burri, P. Gohl, S. Leutenegger, P. Furgale, and R. Siegwart. A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM. In *IEEE Int. Conf. on Robotics and Automation*, pages 431–437, 2014.

[75] J. Nikolic, M. Burri, I. Gilitschenski, J. Nieto, and R. Siegwart. Non-parametric extrinsic and intrinsic calibration of visual-inertial sensor systems. *IEEE Sensors Journal*, 16(13): 5433–5443, 2016.

[76] F. Nobre, C. R. Heckman, and G. T. Sibley. Multi-sensor slam with online self-calibration and change detection. In *Proceedings of the International Symposium on Experimental Robotics (ISER)*, pages 764–774. Springer, 2016.

[77] F. Nobre, M. Kasper, and C. Heckman. Drift-correcting self-calibration for visual-inertial slam. In *IEEE Int. Conf. on Robotics and Automation*, pages 6525–6532, 2017.

[78] H. Oleynikova, M. Burri, S. Lynen, and R. Siegwart. Real-time visual-inertial localization for aerial and ground robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 3079–3085, Sep. 2015.

[79] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1366–1373, Sep. 2017.

[80] A. Patron-Perez, S. Lovegrove, and G. Sibley. A spline-based trajectory representation for sensor fusion and rolling shutter cameras. *Int. Journal of Computer Vision*, 113(3): 208–219, 2015.

[81] J. A. Preiss, K. Hausman, G. S. Sukhatme, and S. Weiss. Trajectory optimization for self-calibration and navigation. In *Robotics: Science and Systems*, 2017.

[82] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[83] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *IEEE Int. Conf. on Robotics and Automation*, 2009.

[84] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart. Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. In *IEEE Int. Conf. on Robotics and Automation*, 2016.

[85] J. Rehder, J. Nikolic, T. Schneider, and R. Siegwart. A direct formulation for camera calibration. In *IEEE Int. Conf. on Robotics and Automation*, pages 6479–6486, 2017.

[86] L. Riazuelo, J. Civera, and J. Montiel. C2tam: A cloud framework for cooperative tracking and mapping. *Robotics and Autonomous Systems*, 62(4):401–413, 2014.

[87] A. Richardson, J. Strom, and E. Olson. Aprilcal: Assisted and repeatable camera calibration. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1814–1821, 2013.

[88] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Int. Conf. on Computer Vision*, pages 667–674, Nov 2011.

[89] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *European Conf. on Computer Vision*, pages 752–765, Berlin, Heidelberg, 2012. Springer.

[90] T. Schneider, M. Li, M. Burri, J. Nieto, R. Siegwart, and I. Gilitschenski. Visual-inertial self-calibration on informative motion segments. In *IEEE Int. Conf. on Robotics and Automation*, pages 6487–6494, 2017.

[91] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart. maplab: An open framework for research in visual-inertial mapping and localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425, July 2018.

[92] T. Schneider, M. Li, C. Cadena, J. Nieto, and R. Siegwart. Observability-aware self-calibration of visual and inertial sensors for ego-motion estimation. *IEEE Sensors Journal*, 19(10):3846–3860, May 2019.

[93] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3):835–846, July 2006.

[94] T. J. Steiner, G. Huang, and J. J. Leonard. Location utility-based map reduction. In *IEEE Int. Conf. on Robotics and Automation*, pages 479–486, May 2015.

[95] N. Sünderhauf and P. Protzel. Switchable constraints for robust pose graph SLAM. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1879–1884, Oct 2012.

[96] C. Sweeney. Theia multiview geometry library: Tutorial & reference. `http://www.theia-sfm.org`, 2019. [Online; accessed 29-Jan-2019].

[97] L. Traffelet, T. Eppenberger, A. Millane, T. Schneider, and R. Siegwart. Target-based calibration of underwater camera housing parameters. In *IEEE Int. Symposium on Safety, Security, and Rescue Robotics*, pages 201–206, Oct 2016.

[98] N. Trawny and S. I. Roumeliotis. Indirect kalman filter for 3d attitude estimation. *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, 2, 2005.

[99] F. Tschopp, T. Schneider, A. W. Palmer, N. Nourani-Vatani, C. Cadena, R. Siegwart, and J. Nieto. Experimental comparison of visual-aided odometry methods for rail vehicles. *IEEE Robotics and Automation Letters*, 2019. (accepted for publication).

[100] J. Ventura, C. Arth, G. Reitmayr, and D. Schmalstieg. Global Localization from Monocular SLAM on a Mobile Phone. *IEEE Transactions on Visualization and Computer Graphics*, 20(4):531–539, 2014.

[101] T. A. Vidal-Calleja, C. Berger, J. Solà, and S. Lacroix. Large scale multiple robot visual mapping with heterogeneous landmarks in semi-structured terrain. *RAS*, 59(9):654 – 674, 2011.

[102] O. J. Woodman. An introduction to inertial navigation. Technical Report UCAM-CL-TR-696, University of Cambridge, Computer Laboratory, 2007.

[103] D. Zachariah and M. Jansson. Joint calibration of an inertial measurement unit and coordinate transformation parameters using a monocular camera. In *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–7, 2010.

[104] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

[105] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza. Benefit of large field-of-view cameras for visual odometry. In *IEEE Int. Conf. on Robotics and Automation*, pages 801–808, 2016.

Curriculum Vitae

Thomas Schneider
born August 25, 1986
citizen of Rüthi SG, Switzerland

PROJECTS

2015–2019 *Maplab: visual-inertial mapping and localization framework* at ETH Zurich, Switzerland
Development and implementation of visual-inertial state estimation framework. http://www.github.com/ethz-asl/maplab

2014–2019 *Kalibr: open-source visual-inertial calibration framework* at ETH Zurich and Skybotix AG, Switzerland
Open-source calibration tools for visual and inertial sensors within a continuous-time batch estimation framework. http://www.github.com/ethz-asl/kalibr

2012 *Master thesis* at I3S, University of Nice, France
Fault-tolerant control allocation for multirotor helicopters using parametric programming

2010 *Bachelor thesis* at Autonomous Systems Lab, ETH Zurich
Developed a GNSS-IMU state estimation algorithm for a aerial vehicles.

2008–2009 *DisneyCopter — a flying entertainment robot* at ETH Zurich, Switzerland
Development of a flying entertainment robot; funded by Walt Disney Imagineering, USA.

RESEARCH AND WORK EXPERIENCE

2014–2019 *PhD Candidate* at Autonomous Systems Lab, ETH Zurich, Switzerland
Doctoral studies at the Autonomous Systems Lab; Supervised by Prof. Roland Siegwart and Prof. Stephan Weiss

2016 *Visiting Researcher* at Google Inc., USA
Visiting Researcher at Google Inc., Mountain View, USA; Development of a life-long calibration pipeline for visual-inertial sensors in collboration with Google Project Tango.

2012–2014 *Research and development engineer* at Skybotix AG (acquired by GoPro), Zurich, Switzerland
Developing the embedded firmware of a visual-inertial sensor init and the low and high-level control stack of the hexacopter platform Skybotix Flybox.

2010–2011 *Teaching assistant* at Institute for Dynamic Systems and Control, ETH Zurich
Teaching Embedded Control System course in collaboration with the Systems Laboratory at the University of Michigan, USA.

2009 *IAESTE internship* at Universitario da FEI, Brasil
Dynamic compensation of a piezo-electric force sensor.

EDUCATION

2010–2012 *MSc. in Mechanical Engineering* at ETH Zurich, Switzerland
Focus on state estimation and renewable energy technologies.

2006–2009 *BSc. in Mechanical Engineering* at ETH Zurich, Switzerland
Focus on embedded systems, system modelling and control.

1999–2005 *High School* at Kantonsschule Zug, Switzerland
Swiss Matura with focus on chemistry and biology.