UNIVERSITY OF BREMEN
INSTITUTE FOR
ARTIFICIAL INTELLIGENCE

Collaborative Localization and Mapping for Autonomous Planetary Exploration

Distributed Stereo Vision-Based 6D SLAM in GNSS-Denied Environments

Martin Johannes Schuster

Vollständiger Abdruck der vom Fachbereich 3 (Mathematik und Informatik) der Universität Bremen zur Erlangung des akademischen Grades eines

Doktor der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

1. Gutachter:	Prof. Michael Beetz, PhD *Universität Bremen*
2. Gutachter:	PD Dr. habil. Rudolph Triebel *Technische Universität München*
Weitere Prüfer:	Prof. Dr. Kerstin Schill *Universität Bremen* Prof. Dr. Rolf Drechsler *Universität Bremen*

Die Dissertation wurde am 28.05.2019 bei der Universität Bremen eingereicht und durch den Prüfungsausschuss am 21.08.2019 angenommen.

Abstract

Mobile robots are a crucial element of present and future scientific missions to explore the surfaces of foreign celestial bodies such as Moon and Mars. The deployment of teams of robots allows to improve efficiency and robustness in such challenging environments. As long communication round-trip times to Earth render the teleoperation of robotic systems inefficient to impossible, on-board autonomy is a key to success. The robots operate in Global Navigation Satellite System (GNSS)-denied environments and thus have to rely on space-suitable on-board sensors such as stereo camera systems. They need to be able to localize themselves online, to model their surroundings, as well as to share information about the environment and their position therein. These capabilities constitute the basis for the local autonomy of each system as well as for any coordinated joint action within the team, such as collaborative autonomous exploration.

In this thesis, we present a novel approach for stereo vision-based on-board and online Simultaneous Localization and Mapping (SLAM) for multi-robot teams given the challenges imposed by planetary exploration missions. We combine distributed local and decentralized global estimation methods to get the best of both worlds: A local reference filter on each robot provides real-time local state estimates required for robot control and fast reactive behaviors. We designed a novel graph topology to incorporate these state estimates into an online incremental graph optimization to compute global pose and map estimates that serve as input to higher-level autonomy functions. In order to model the 3D geometry of the environment, we generate dense 3D point cloud and probabilistic voxel-grid maps from noisy stereo data. We distribute the computational load and reduce the required communication bandwidth between robots by locally aggregating high-bandwidth vision data into partial maps that are then exchanged between robots and composed into global models of the environment. We developed methods for intra- and inter-robot map matching to recognize previously visited locations in semi- and unstructured environments based on their estimated local geometry, which is mostly invariant to light conditions as well as different sensors and viewpoints in heterogeneous multi-robot teams. A decoupling

of observable and unobservable states in the local filter allows us to introduce a novel optimization: Enforcing all submaps to be gravity-aligned, we can reduce the dimensionality of the map matching from 6D to 4D. In addition to map matches, the robots use visual fiducial markers to detect each other. In this context, we present a novel method for modeling the errors of the loop closure transformations that are estimated from these detections.

We demonstrate the robustness of our methods by integrating them on a total of five different ground-based and aerial mobile robots that were deployed in a total of 31 real-world experiments for quantitative evaluations in semi- and unstructured indoor and outdoor settings. In addition, we validated our SLAM framework through several different demonstrations at four public events in Moon and Mars-like environments. These include, among others, autonomous multi-robot exploration tests at a Moon-analogue site on top of the volcano Mt. Etna, Italy, as well as the collaborative mapping of a Mars-like environment with a heterogeneous robotic team of flying and driving robots in more than 35 public demonstration runs.

Zusammenfassung

Mobile Roboter stellen eine wesentliche Technologie für gegenwärtige und zukünftige wissenschaftliche Missionen zur Exploration der Oberflächen fremder Himmelskörper, wie dem Mond oder Mars, dar. Die Entsendung von Roboterteams erlaubt es, die Effizienz und Robustheit in solch schwierigen Umgebungen zu erhöhen. Ein Schlüssel zum Erfolg sind lokale Autonomiefunktionen der robotischen Systeme, da lange Signallaufzeiten zur Erde eine teleoperierte Steuerung ineffizient oder gar unmöglich machen. Die Roboter operieren in Umgebungen, in denen globale, satellitengestützte Positionierungssysteme nicht verfügbar sind und können sich daher nur auf ihre eigene weltraumtaugliche Sensorik, wie z. B. mitgeführte Stereokamerasysteme, verlassen. Sie müssen in der Lage sein, sich selbst schritthaltend zu lokalisieren, ihr Umfeld zu modellieren, sowie Informationen über die Umgebung und ihre Position darin mit anderen Robotern zu teilen. Diese Fähigkeiten stellen die Grundlage sowohl für die lokale Autonomie jedes einzelnen Systems dar, als auch für jegliche koordinierte Zusammenarbeit im Team, wie beispielsweise eine gemeinschaftliche autonome Exploration.

In dieser Dissertation werden neuartige Methoden zur simultanen Lokalisierung und Kartenerstellung (SLAM) für Teams stereokamerabasierter Roboter in Anbetracht der Herausforderungen planetarer Explorationsmissionen vorgestellt. Alle Roboter berechnen dabei ihre Schätzungen schritthaltend auf ihren jeweiligen Bordcomputern. Verteilte lokale und dezentralisierte globale Schätzmethoden werden kombiniert, um das Beste aus beiden Welten zu vereinen: Ein Lokalreferenzfilter auf jedem Roboter stellt lokale Zustandsschätzungen, welche für die Roboterregelung und schnelle reaktive Verhalten benötigt werden, in Echtzeit zu Verfügung. Eine neuartige Graphtopologie wurde entworfen, um diese Zustandsschätzungen in einer schritthaltenden, inkrementellen Graphoptimierung zu verbinden und dadurch globale Schätzungen von Posen und Karten zu berechnen, welche als Basis für die Autonomiefunktionen der Roboter dienen. Zur Modellierung der 3D Geometrie der Umgebung werden aus verrauschten Stereodaten dichte 3D Punktwolken und probabilistische Voxelgitterkarten erstellt. Kameradaten mit hoher Bandbreite werden lokal zu Teilkarten aggregiert um die Rechenlast auf die einzelnen Roboter zu verteilen. Zudem wird durch einen Austausch dieser kompakten Kartenrepräsentation zwischen den Robotern die benötigte Kommunikationsbandbreite reduziert. Die Teilkarten werden dann auf den jeweiligen Robotern zu einem globalen Umgebungsmodell zusammengefügt. Es wurden Methoden für den Abgleich von Karten eines als auch mehrerer Roboter entwickelt um zuvor besuchte Orte in semi- und unstrukturierten Umgebungen basierend auf dem geschätzten Modell ihrer lokalen Geometrie wiederzuerkennen. Dieses Modell ist größtenteils invariant in Bezug auf unterschiedliche Belichtungsbedingungen als auch unterschiedliche Sensoren, Standpunkte und Blickwinkel verschiedener Roboter in heterogenen Teams. Eine Entkopplung beobachtbarer und unbeobachtbarer Zustände im lokalen Filter erlaubt die Einführung einer neuartigen Optimierung: Eine Ausrichtung aller Teilkarten am Schwerkraftvektor ermöglicht es, die Dimensionalität des Kartenabgleichs von 6D auf 4D zu reduzieren. Zusätzlich zum Kartenabgleich werden Marker als optische Referenzpunkte zur gegenseitigen Detektion der Roboter verwendet. In diesem Kontext wird eine neuartige Methode zur Modellierung der Fehler der daraus geschätzten Transformationen vorgestellt.

Die Robustheit der vorgestellten Methoden wird durch ihre Integration auf fünf verschiedenen fahrenden und fliegenden Robotersystemen demonstriert, welche in 31 Experimenten zur quantitativen Evaluierung in realen semi- und unstrukturierten Innen- und Außenumgebungen eingesetzt wurden. Zusätzlich wurde das Framework zur Lokalisierung und Kartenerstellung bei mehreren verschiedenen öffentlichen Veranstaltungen in mond- und marsähnlichen Umgebungen validiert. Diese umfassen unter anderem autonome Explorationstests mit mehreren Robotern in einer mondanalogen Testumgebung auf dem Vulkan Ätna in Italien, sowie die kollaborative Erstellung der Karte einer marsähnlichen Umgebung mit einem heterogenen Team aus fliegenden und fahrenden Robotern in mehr als 35 öffentlichen Vorführungen.

Acknowledgments

This thesis is the result of more than six years of research conducted between 2012 and 2019 at the Institute of Robotics and Mechatronics (RM) at the German Aerospace Center (DLR) in Oberpfaffenhofen, Germany. I would like to thank all the people who have supported me throughout these years during my research and the writing of this thesis. First, I would like to thank my advisor Prof. Michael Beetz for his vision, inspiration, and advice, as well as for the opportunity to pursue my PhD thesis at the Institute for Artificial Intelligence (IAI) of the University of Bremen as an external PhD student. He sparked my interest in artificial intelligence and robotics already during my studies at TUM and from there on supported many of my research endeavors throughout the years, for which I am very grateful. I would like to thank PD Dr. habil. Rudolph Triebel for providing guidance and support as my second advisor. I would also like to thank him, Dr. Michael Suppa, and Prof. Alin Albu-Schäffer for giving me the opportunity to work at the DLR Institute of Robotics and Mechatronics (RM), one of the most inspiring research environments, and for supporting my work there.

Many thanks go to Dr. Tim Bodenmüller for supervising my thesis work at DLR, providing constant encouragement, guidance, and extensive proofreading. In addition, I owe lot of thanks to Dr. Armin Wedler, Dr. Heiko Hirschmüller, and Dr. Wolfgang Stürzl for their support of my work and research over the years. Moreover, thanks go to Mallikarjuna Vayugundla, Dr. Wolfgang Stürzl, Marcus Müller, and Dr. Korbinian Schmid for proofreading this thesis. Developing and operating complex robot systems is always a great team effort. Thus, some of the work presented in this thesis was conducted in close collaboration with my colleagues from the mobile robotics group and its planetary exploration team of the Institute of Robotics and Mechatronics (RM) at DLR. In particular, I would like to thank Christoph Brand for the fruitful collaboration on several aspects of the SLAM framework, Dr. Korbinian Schmid and Felix Rueß for their work on the local reference filter, Mallikarjuna Vayugundla and Marcus Müller for their support with the robots and experiments, Hannah Lehner for providing exploration experiment data, Matthias Hellerer for his work on the simulator, Dr. Stefano Seriani for the design of the marker mounting brackets, as well as Dr. Michal Smisek and Florian Steidle for their support with robot calibration. Furthermore, I would like to thank Dr. Simon Kriegel, Dr. Zoltán-Csaba Márton, Philipp Lutz, Sebastian Brunner, Peter Lehner, Kristin Bussmann, Iris Grixa, Rico Belder, Michael Panzirsch, Dr. Annett Stelzer, Dr. Teodor Tomić, and Dr. Elmar Mair, as well as all of my colleagues named above and many more from our institute for either their work on the robot systems, many valuable and insightful discussions, or, in many cases, both. It has been a fun and rewarding experience working together.

I would like to thank the members of the mobile robotics group as well as the SpaceBotCamp, ROBEX and ARCHES project teams for their great support in preparing and conducting many of the experiments and demonstrations presented in this thesis. The photo on the front cover, showing one of our ROBEX multi-robot experiments on Mt. Etna, Sicily, Italy, is a courtesy of Esther Horvath. Over the years, I have had the pleasure of working with several excellent students who assisted me in implementing and testing some of the concepts presented in this thesis. In particular, my thanks go to Sebastian Brunner, Sebastian Vetter, Philip Heijkoop, and Matthias Holoch. Furthermore, I would like to thank all the people who keep things running at the institute: The IT system administrators, the workshop teams, and our service team.

Last but not least, I would like to thank my family and friends for their constant support and encouragement. My parents Martha and Hans-Jürgen Schuster have provided me with their unconditional support throughout my life, believing in my strengths and exciting my curiosity, for which I am deeply grateful. Furthermore, I would like to thank my aunt Inge Axtmann for her support and English language proofreading. Finally, I would like to thank my girlfriend Alice Wacker for her great encouragement, support, and patience while I was writing this thesis.

May 2019,

Martin J. Schuster

This work has received funding from the Helmholtz Association, project alliance ROBEX (contract number HA-304) and project ARCHES (contract number ZT-0033), and was supported by the DFG Collaborative Research Center 1320, EASE.

Contents

List of Figures

List of Tables

Chapter one

Introduction

The exploration of foreign moons and planets is an important present and future application for mobile robots. Compared to Earth, their surfaces typically feature extreme environmental conditions like a high variation in temperature, lack of a proper atmosphere, fine-grained dust, and radiation. These, in addition to a long journey in space, make them hard to access and difficult to reach for humans. The *2018 Global Exploration Roadmap* (ISECG, 2018), a joint publication of fourteen major space agencies that form the *International Space Exploration Coordination Group*, states:

> *"Robotic missions accomplish world-class science while also serving as our scouts and proxies, venturing first into hostile environments to gather critical information that makes human exploration safer. [...] In this global vision, robotic missions precede human explorers to the Moon, near-Earth asteroids, and Mars in order to unveil many of their secrets, characterise their environments, and identify risks and potential resources."* — *(ISECG, 2018)*

A famous example of the successful deployment of a mobile robot during a scientific exploration mission is the *Curiosity* rover, which landed on Mars in 2012 in order to examine the possibility of microbiological life there (JPL NASA, 2013). The deployment of such a single huge and complex system, which the car-sized Curiosity rover with its 17 cameras and other instruments doubtlessly is (Neith, 2012), however, creates many single points of failure for the costly mission. As a consequence, Curiosity is moving very slowly and carefully in order to avoid getting stuck, as, for example, the Mars rover *Spirit* did in 2009 (Wolchover, 2011).

We believe one of the next big steps will be the development of multi-robot systems to cooperatively explore and map the surface of foreign celestial bodies as a crucial part of future scientific missions. Teams of robots can avoid the aforementioned single points of failure, improve efficiency through parallelization, robustness through redundancy, and can benefit from complementary capabilities in heterogeneous robot teams. Typically, not one type of robot exists that is perfectly well suited for all kinds of tasks involved in a complex mission. Each robot is specialized for a certain range of jobs and thus has its own specific strengths and weaknesses reflected in its particular actuators, sensors, and computational resources. In heterogeneous teams, additional benefits can emerge from the collaboration of different robots when they complement each other in order to gain a greater advantage than just by adding up their sensorial and computational capacities. A single robot, for example, might lack the capabilities to both localize and reach as well as to enter and explore scientifically interesting locations like caves or lava tubes. In contrast, the robots in a heterogeneous team can combine their complementary capabilities to locate, access, and map such challenging areas. For example, aerial robots can create large-scale maps during scouting trips, while ground-based systems use these to plan their target locations and then enhance them locally by adding details from close-range observations. The deployment of teams of robots can thus be the key to success for lunar and planetary surface exploration in the context of future scientific missions to search for signs of life, resources and potentially habitable areas on the Moon or Mars. Furthermore, similar technologies can also be crucial for terrestrial applications in environments dangerous for humans to access, like, for example, search and rescue missions at the sites of nuclear power plants that have been partially destroyed by disaster events.

In order to autonomously explore the surface of foreign planets and moons with heterogeneous robotic teams, the robots forming such teams need to be able to localize themselves, model their surroundings, as well as share information about the environment and their position therein. These capabilities constitute the basis for the local autonomy of each system as well as for any coordinated joint action within the team, such as a collaborative autonomous exploration. The goal of this thesis is to contribute to the development of such multi-robot systems by developing, evaluating, and demonstrating novel methods for on-board and online Simultaneous Localization and Mapping (SLAM) for heterogeneous multi-robot teams given the challenges and restrictions imposed by planetary exploration missions.

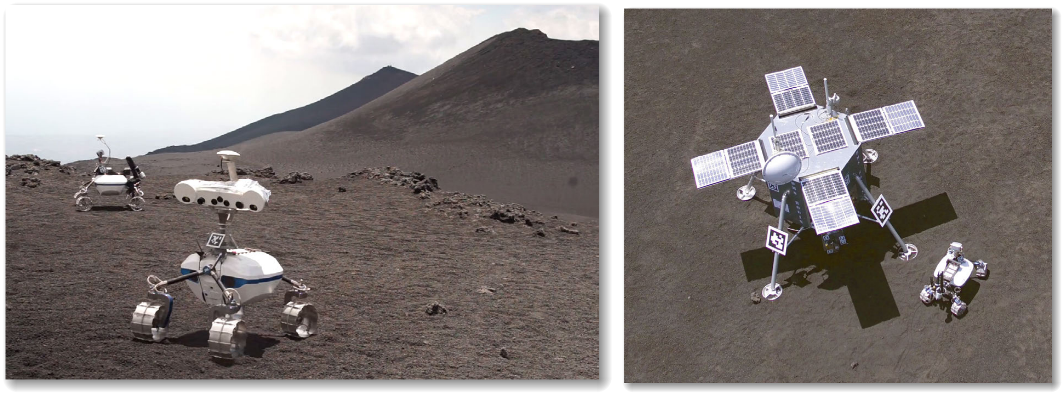


Figure 1.1: *Impressions from experiments with two of our Lightweight Rover Units (LRUs) that we conducted in 2017 as part of the Robotic Exploration of Extreme Environments (ROBEX) project at a Moon-analogue test site on the volcano Mt. Etna, Sicily, Italy*

1.1 Planetary Exploration Scenario

Let us consider a mission to the surface of Earth's moon with the key objective to explore sheltered areas like craters or subterranean caves that are less influenced by the Moon's harsh environmental conditions. According to the Global Exploration Roadmap (ISECG, 2018), scientifically "[i]nteresting locations include the lunar poles (both north and south), volcanic deposits, impact craters and basins, and lava tubes or pits." Offering a more stable temperature, lower radiation levels and protection from micrometeorite bombardment, sheltered areas are promising locations to obtain valuable rock samples that have not been space-weathered, as well as locations to set up infrastructure for future human habitation. Furthermore, caves on the Moon can be seen as analogue environments to caves on Mars, which are promising areas to search for potential signs of alien life and which, in the remote future, might provide convenient access to resources like recently discovered subterranean deposits of water ice (Brown et al., 2016). Featuring water ice in its polar regions, also the Moon itself offers opportunities for In-Situ Resource Utilization (ISRU), with robotic prospecting and ISRU demonstration missions being planned by Roscosmos, NASA, ESA, and JAXA within the next decade (ISECG, 2018). In the remainder of this section, we envision a future robotic lunar exploration scenario that might follow such first prospecting missions. Based on this scenario, we highlight and discuss the challenges and our contributions regarding localization and mapping for multi-robot autonomous planetary exploration.

Assume a lander spacecraft has transported several different robots to the surface of the Moon. In Figure 1.1, we give an impression of our experiment campaign at a

Moon-analogue test site. Next to a lander mockup, it shows two rover prototypes, which we have developed in the planetary exploration team of the Institute of Robotics and Mechatronics (RM) at the German Aerospace Center (DLR). After landing on the Moon's surface, the lander itself acts as a base station, allowing the robots to recharge or refuel, collect tools or infrastructure elements, and provide heavy equipment to analyze rock samples. As the first step after landing, flying robots are deployed to provide an overview of the area around the landing site. While future rotor-based robotic vehicles might be able to fly on planets and moons with a sufficiently dense atmosphere, such as Mars (Huber, 2016) or Saturn's moon Titan (Lorenz et al., 2017), flying robots to be deployed on Earth's moon require thruster-based propulsion. Such lightweight robots are nonetheless envisioned to be able to travel distances of up to 5 km (Thangavelautham et al., 2014), sufficient to gain an overview of the areas accessible by slower rovers that can carry heavier payloads. They create a 3D map of the terrain that gives valuable information on where to take measurements and which areas to explore next with ground-based robots, allowing to locate interesting features like, for example, the entrances to craters and caves. In a next step, rovers are sent out to collaboratively explore and map their surroundings. For navigation, they classify the terrain according to its traversability and autonomously avoid obstacles therein. They can use the previously created aerial maps to support their long-range path planning through difficult terrain and enhance these 3D environment models with further details from their close-range sensors. After identifying suitable target locations, they fetch sensor boxes from the lander and place them in an array formation in order to gain insights into subterranean structures, as proposed in the Robotic Exploration of Extreme Environments (ROBEX) project (Wedler et al., 2017). Such sensor networks allow to gain novel scientific insights into the structure of the lunar crust and mantle, similar to the seismometers placed during the Apollo 12 - 16 missions, which had been operational until 1977 to detect moonquakes and meteoroid impacts (Nakamura et al., 1982). Furthermore, measurements from seismic sensors or ground-penetrating radar can be used to survey subterranean lava tubes. Such natural lunar caves of up to 50 km length have recently been discovered next to *Marius Hills Hole*, a skylight of at least 15 m height from floor to ceiling that potentially leads to an intact lava tube (Kaku et al., 2017).

Once a cave suitable for close-up exploration has been identified, larger transport rovers carry smaller, but more agile, robots to its entrance. These can, for example, be thruster-based flying robots such as *Extreme Access Flyers* (Siceloff, 2015) or *PitBots* (Thangavelautham et al., 2014). The latter is a concept for small 3 kg spherical systems equipped with stereo camera systems for 3D mapping. They perform series of short one-to-two-minute hops to pass steep slopes and difficult obstacles that can be expected at the entrance of a lava tube. Once the transport rovers have reached their destination, they deploy the smaller systems at the entrance of the cave, which is not accessible for larger robots. Therein, the smaller systems are cut off from any external means of global localization. Exploring previously unknown terrain, they can solely rely on their on-board sensors. In addition, as individual robots might experience communication losses to their team, they require full on-board autonomy to fulfill their mission. In order to navigate in three-dimensional cave structures, they thus need to create a 3D model of their environment online and on-board. Exchanging data with each other whenever they are within communication range, these robots collaboratively explore and map the cave, use this map to select areas of scientific interest, conduct close-range inspections and take samples, utilizing their different sensor setups and manipulation capabilities respectively. After they have returned on their own, they are transported back to the lander for recharging and for the analysis of samples. Whereas the mobile robots build models of their environment online to serve as the basis for their autonomous capabilities, offline and off-board data processing at the lander as well as off-site processing on Earth allow for further in-depth analyses of the robots' findings after their return.

The deployment of a multi-robot team in this scenario allows the robots to combine their complementary capabilities to reach places, like the interior of lava tubes, that otherwise would remain inaccessible. Redundant capabilities in the team increase the robustness of such high-risk missions by avoiding single points of failure. In addition, tasks like the exploration and mapping of the environment can be sped up through parallelization, improving efficiency and the potential scientific output.

1.2 Goal and Objectives

Localization and mapping capabilities are crucial for an autonomous robotic team that explores previously unknown areas on planetary surfaces, as we sketched out in the exploration scenario of the previous section. We agree with Olson et al. (2013) and many other robotics researchers in believing that "[a] good state estimate, in the form of a map, is the most critical piece of information for a team of robots – and the most difficult to obtain". Maps and localization estimates constitute the basis for the local autonomy of each mobile robot as well as for any coordinated collaborative multi-robot action. Geometric models of the environment allow robots to navigate, locate objects of interest therein, annotate them semantically and ultimately use them to gain novel scientific insights, directly from the models themselves as well as indirectly by facilitating and supporting scientific missions. Up-to-date map and localization estimates on board and for all robots allow them to plan and coordinate their actions based on environment models that incorporate their most recent sensor data, reflecting the progress of the robotic team during exploration missions.

The goal of this thesis is the development of novel methods for on-board and online Simultaneous Localization and Mapping (SLAM) for heterogeneous multi-robot teams given the challenges and restrictions imposed by planetary exploration missions.[1] This includes the evaluation and demonstration of our methods on real exploration robot prototypes both in controlled lab settings as well as in realistic, space-like environments. From this overall goal, we derived as objectives the development, improvement and combination of methods to provide the following capabilities to planetary exploration robots:

- *Collaborative online creation of a 3D map of the environment* for exploration planning, inspection, and as a basis for future semantic annotations
- *Global localization of all robots in that map* to allow for multi-robot coordination
- *Fast local localization and mapping* for navigation and obstacle avoidance

These objectives are complicated by a number of additional challenges imposed by planetary exploration missions: Communication links are typically limited in bandwidth and unreliable, i. e., communication losses to and between the robots are to be expected. Therefore, all robots need to be able to act autonomously, relying solely on their on-board sensors and on-board computation. Hazardous environmental conditions, the mechanical stress of space flight, as well as constraints on size, weight, and power consumption further limit the selection of sensors available for small to medium-sized robots. Camera-based perception systems are well suited and real-world tested to fulfill these constraints, their noise characteristics, however, pose additional challenges on any localization and mapping systems based on their data. We will discuss these and further challenges in more detail in Section 2.1.

[1]The development of novel methods for plan and action generation based on models of the robots' environment, such as autonomous exploration planning, lies mostly beyond the scope of this thesis and is part of recent and ongoing research by the planetary exploration team at DLR-RMC.

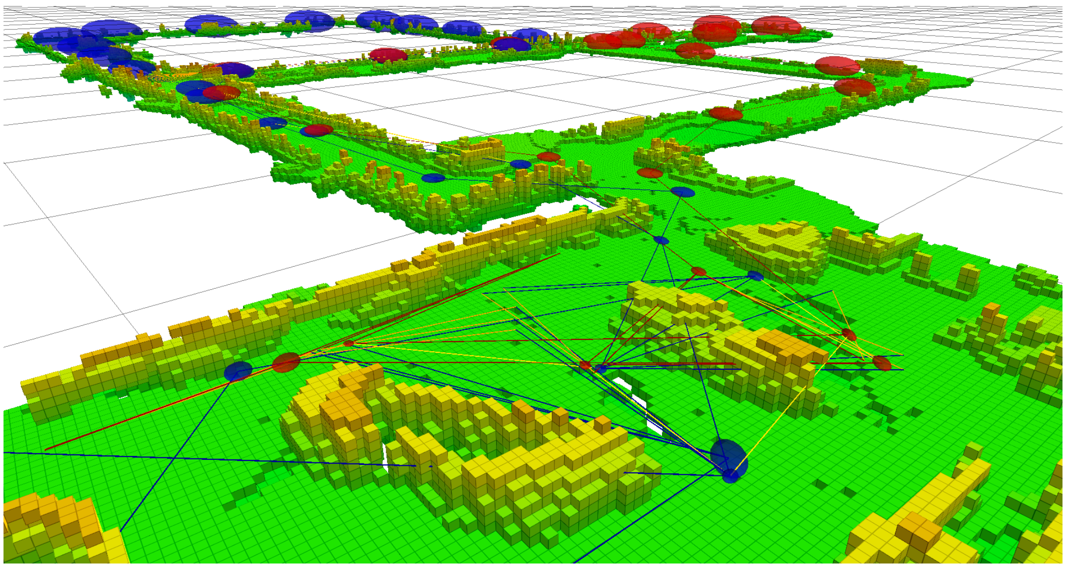


Figure 1.2: *Multi-robot 3D probabilistic voxel-grid map (height-colored, resolution:* 10 cm*, grid size:* 5 m*) and SLAM graph with pose covariance ellipsoids, created by our two Lightweight Rover Units LRU1 (blue) and LRU2 (red). We provide a detailed description of the experiments and its results in Section 8.3.2 and Figure 8.24.*

1.3 Approach

In order to approach our goal and fulfill the objectives given the aforementioned challenges, we designed a framework for 6D local and global Simultaneous Localization and Mapping (SLAM) for heterogeneous multi-robot teams of planetary exploration robots. To illustrate our results, we present a joint 3D probabilistic voxel-grid map in Figure 1.2, which was created by two Lightweight Rover Units (LRUs) (Schuster et al., 2016, 2017) equipped with stereo camera systems.

A major design goal for our system is a loose coupling of fast local and online global estimation to support both local robot autonomy, including fast obstacle avoidance, as well as collaborative behavior, such as the exploration of unknown terrain based on a joint model. In order to deal with limited and unreliable communication links, the robots share their computational workload and create joint maps when linked via low-bandwidth connections. For this, we designed a distributed system that builds upon the local aggregation of high-frequency and high-bandwidth sensor data for dense 3D multi-robot mapping. Its central aspects are:

- *Modular multi-robot 6D localization and mapping architecture* to allow an adaptation to the perceptive and computational capabilities of each robot in a heterogeneous team

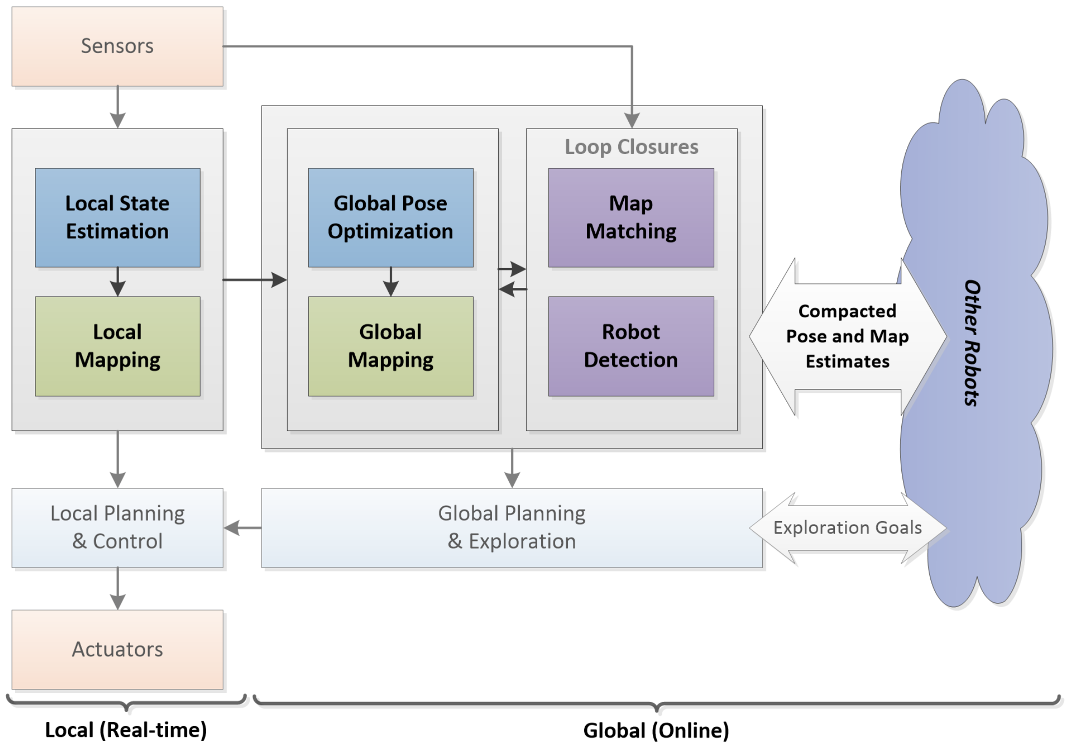


Figure 1.3: *System architecture overview with focus on our on-board localization and mapping modules*

- *Combination of distributed local and decentralized global estimation* to distribute the computational load and achieve robustness w. r. t. communication losses by eliminating single points of failure
- *Stereo vision-based online dense 3D map creation and exchange of aggregated data* to create a joint environment model from the data perceived by space-suitable sensors and exchange it via low-bandwidth connections
- *Online inter- and intra-robot loop closure generation* to compose and optimize a joint map and localize all robots therein

In Figure 1.3, we present a high-level overview of our system architecture. The components visualized in the block diagram run simultaneously on-board all robots. We focus on stereo cameras as our main exteroceptive sensor to satisfy the requirement for a space-qualifiable sensor setup. They allow the computation of dense 3D data under varying light conditions in both indoor and outdoor environments. As proprioceptive sensor data, we gather estimates from an Inertial Measurement Unit (IMU) and, where available, wheel odometry. We fuse the local sensor data on each robot for real-time local state estimation (Schmid et al., 2014a), 3D environment mapping, and stereo error-adaptive obstacle classification, which we first introduced in our conference paper by Brand et al. (2014). The results from these components can directly be used for control and local planning, for example to realize a fast obstacle avoidance. These local estimates for each robot's pose and map represent an aggregation of the raw sensor data. This lowers the complexity of subsequent processing steps. In addition, we can distribute the workload of the computationally expensive processing of dense 3D data. Exchanging data between robots at this higher level of abstraction allows for a more efficient transmission, with lower bandwidth and lower frequency, compared to sending the raw measurement data.

To compute global pose and map estimates, we integrate the aggregated local estimates into a global optimization framework. The usage of this preprocessed data allows for an incremental online joint optimization taking into account all the aggregated data that are available from all robots within communication range. For this, we combine keyframe-based local reference filters (Schmid et al., 2014b) and multi-robot graph SLAM with incremental optimization, an architecture that we introduced in our conference and journal publications by Schuster et al. (2015, 2018). We designed a novel multi-robot SLAM graph topology that allows the better integration of the filter results according to their estimated uncertainty and independence assumptions, leading to more accurate estimates. The decoupled integration of these local and global methods achieves the best of both worlds: Local reference filters on each robot provide real-time, long-term stable state estimates that are required for stabilization, control and fast obstacle avoidance, whereas online graph optimization provides global multi-robot pose and map estimates needed for cooperative planning. Furthermore, it enables a distributed integration of high-frequency and high-bandwidth measurements and allows each robot independently to estimate its own pose and map on-board and online at all times. Thus, we can distribute significant parts of the computational workload, avoid single points of failure and gain robustness w. r. t. interrupted communication and failures of individual robots. The resulting, collaboratively built environment model can then be used for global planning, for example to compute goal locations for each robot in a multi-robot exploration algorithm (Lehner et al., 2017).

For global pose and map optimization, we consider two different types of loop closure constraints. The first type stems from place recognition for the re-localization of the robots in their environment model. Sensor data association for loop closure generation is particularly challenging for stereo vision-based systems due to the typically narrow angle of view of their cameras compared to laser scanners. The integration of multiple measurements into local maps with limited drift, so-called submaps, allows us to tackle this challenge: In our publications by Brand et al. (2015) and Schuster et al. (2018), we introduced a novel approach to select and match submaps, computing an estimate of their relative transformation as well as of its uncertainty. We compute these intra- and inter-robot constraints by matching the geometric 3D information, thus allowing us to match submaps acquired under varying light conditions and by robots with different camera systems and viewpoints.

The second type of loop closures stems from the ability of our robots to detect and localize each other when they are within detection range and line of sight. We base this localization on artificial visual markers attached to the robots, a solution viable in a space exploration setting, where more sophisticated model-fitting methods could result in a waste of scarce computational resources. For the integration of such inter-robot loop closure constraints into global optimization, their estimation uncertainty should be known. This is particularly important in the case of far-away detections that are prone to significant errors. We therefore propose a method to approximate the worst-case uncertainty of particular measurements through a combination of camera-dependent precomputed error values and an online error propagation. Thus, all loop closure constraints are accompanied by Gaussian uncertainty estimates, which we consider during graph optimization.

We designed the architecture of our localization and mapping framework taking into account the functional and non-functional challenges posed by space exploration scenarios. Its modularity allows a decoupling of two types of components: first, modules that are required to run in real-time and provide system-critical data, like input to controllers or local maps for a sufficiently fast obstacle avoidance. Second, less critical modules that run online, but for which unexpected delays or failures would not endanger the robot hardware. As all computation is decentralized, online, and on-board, each individual robot has sufficient information to act as an autonomous agent in case of communication losses or the failure of its peers.

1.4 Contributions

In this thesis, we present a framework for online multi-robot localization and mapping with a focus on the integration of fast local methods on each robot for distributed computation and decentralized online global pose and map estimation. Our main contributions can be clustered and summarized as follows:

Combination of Local and Global Estimation

- We propose a *modular localization and mapping architecture* that builds on modules for distributed and decentralized computation. Online pose and map estimates are computed on board of each robot, representing the basis for local robot autonomy that can achieve robustness w. r. t. failures of communication and individual robots.
- We achieve a *fast local state estimation and online global optimization* by developing a loosely coupled system that combines real-time filters with incremental graph SLAM. Running the graph optimization at the level of aggregated high-frequency measurements allows us to perform fast optimization steps on a small graph, thus reducing the computational load compared to more tightly coupled systems.
- We improved the *accuracy of global estimation* through a *novel SLAM graph topology* for the decoupled integration of local filter estimates. We construct the SLAM graph to better represent the dependency assumptions of the filter in the underlying probabilistic framework, in particular compared to graph topologies typically used for sequential odometry measurements.

Online Dense Map Creation and Exchange

- We present an *error-adaptive obstacle classification* on depth images in order to cope with the quadratic growth of the distance-dependent error of stereo camera systems. Taking the camera viewpoint into account, we can detect obstacles that are only indirectly observable, such as steep cliffs leading downwards in front of the robot. In addition to supporting fast obstacle avoidance, the classification results provide *valuable information for place recognition*.
- We *distribute the computational load* and *reduce the required communication bandwidth* between robots by aggregating high-bandwidth vision data into local maps on each robot. Thereby, we partition the shared environment model into so-called submaps based on their inherent uncertainties. Any updates resulting from global optimization only modify the relative transformations between the submaps, not the submap content itself. Thus, to compose a full global map, each submap needs to be transmitted only once to each robot, saving bandwidth compared to an exchange of ever-changing full single-robot maps.

Online Intra- and Inter-Robot Loop Closure Generation

- We *recognize previously visited locations in semi- and unstructured environments* based on the local geometry represented in the submaps of one or multiple robots. Using the 3D information, we can match submaps created by robots with different camera setups. In particular, we leverage the properties of local reference filters to separate observable and unobservable system states. This allows us to reduce the dimensionality of the submap matching problem from 6D to 4D, leading to *more loop closures* that can increase the robustness of the global estimation compared to a 6D matching approach at no additional cost.
- We *improve the localization and mapping accuracy* in multi-robot setups by estimating the uncertainty of visual robot detections. For this, we model the uncertainties of marker-based detections and simulate their propagation. An estimation of the distance- and view angle-dependent measurement noise allows a more accurate representation of marker detection results in our SLAM graph, leading to an improved global optimization result.

We integrated our methods on several single- and multi-robot systems and validated our approach through series of experiments that map to the key challenges regarding the collaborative modeling of planetary surfaces with autonomous robots.

Experimental Validation

- For our *experimental evaluation*, we analyzed data from 40 simulated and 31 real single- and multi-robot experiments, featuring three different robots in areas of up to 3000 m^2 (bounding box). We demonstrate our SLAM framework and evaluate the impact of our key contributions. Our novel graph topology, for example, leads to an improvement in localization accuracy of on average 15 %. An analysis of the computational resources shows that we can significantly reduce bandwidth requirements (58 KB/s instead of 38.75 MB/s) through the aggregation of data and are able to compute fast local state and online pose and map estimates in a distributed fashion.
- We *demonstrated the applicability of our methods* to robots operating in space-like environments at four well-attended public events: at the *Innovation and Leadership in Aerospace (ILA)* trade fair, at the *SpaceBotCamp* robotics competition, at a Moon-analogue test site on the volcano Mt. Etna (Sicily, Italy) as part of the *ROBEX* project, as well as at the *International Astronautical Congress (IAC)*. At IAC, our heterogeneous robotic team of flying and driving robots collaboratively mapped a Mars-like environment in more than 35 demonstrations.

1.5 Publications

We have already published several aspects of this thesis in articles in international journals and conferences, which are referred to in the respective sections. In this thesis, I have reused parts of the text and figures created by myself from some of these publications. The following lists contain those of my prior publications that are relevant in the context of this thesis. For a complete list, see Appendix A.

Publications on the key aspects of this thesis:

- M. J. Schuster, K. Schmid, C. Brand, and M. Beetz. Distributed stereo vision-based 6d localization and mapping for multi-robot teams. *Journal of Field Robotics (JFR)*, 2018. doi: 10.1002/rob.21812. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21812`
- M. J. Schuster, S. G. Brunner, K. Bussmann, S. Büttner, A. Dömel, M. Hellerer, H. Lehner, P. Lehner, O. Porges, J. Reill, S. Riedel, M. Vayugundla, B. Vodermayer, T. Bodenmüller, C. Brand, W. Friedl, I. Grixa, H. Hirschmüller, M. Kaßecker, Z.-C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler. Towards Autonomous Planetary Exploration: The Lightweight Rover Unit (LRU), its Success in the SpaceBotCamp Challenge, and Beyond. *Journal of Intelligent & Robotic Systems (JINT)*, Nov. 2017. ISSN 1573-0409. doi: 10.1007/s10846-017-0680-9. URL `https://doi.org/10.1007/s10846-017-0680-9`
- M. J. Schuster, C. Brand, S. G. Brunner, P. Lehner, J. Reill, S. Riedel, T. Bodenmüller, K. Bussmann, S. Büttner, A. Dömel, W. Friedl, I. Grixa, M. Hellerer, H. Hirschmüller, M. Kassecker, Z.-C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler. The LRU Rover for Autonomous Planetary Exploration and its Success in the SpaceBotCamp Challenge. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Bragança, Portugal, 2016. doi: 10.1109/ICARSC.2016.62
- M. J. Schuster, C. Brand, H. Hirschmüller, M. Suppa, and M. Beetz. Multi-Robot 6D Graph SLAM Connecting Decoupled Local Reference Filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015. doi: 10.1109/IROS.2015.7354094
- C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa. Submap Matching for Stereo-Vision Based Indoor/Outdoor SLAM. In *IEEE/RSJ International*

Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 2015. doi: 10.1109/IROS.2015.7354182

- C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa. Stereo-Vision Based Obstacle Mapping for Indoor / Outdoor SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, Illinois, USA, 2014. doi: 10.1109/IROS.2014.6942805 *(The first two authors assert equal contribution and joint first authorship.)*

Publications related to the topic of this thesis:

- M. G. Müller, F. Steidle, M. J. Schuster, P. Lutz, M. Maier, S. Stoneman, T. Tomić, and W. Stürzl. Robust Visual-Inertial State Estimation with Multiple Odometries and Efficient Mapping on an MAV with Ultra-Wide FOV Stereo Vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. doi: 10.1109/IROS.2018.8594117

- A. Wedler, M. Wilde, J. Reill, M. J. Schuster, M. Vayugundla, S. G. Brunner, K. Bussmann, A. Dömel, M. Drauschke, H. Gmeiner, H. Lehner, P. Lehner, M. G. Müller, W. Stürzl, R. Triebel, B. Vodermayer, A. Börner, R. Krenn, A. Dammann, U.-C. Fiebig, E. Staudinger, F. Wenzköfer, S. Flögel, S. Sommer, T. Asfour, M. Flad, S. Hohmann, M. Brandauer, and A. O. Albu-Schäffer. From single autonomous robots toward cooperative robotic interactions for future planetary exploration missions. In *69th International Astronautical Congress (IAC)*, Bremen, Germany, Oct. 2018b

- S. G. Brunner, P. Lehner, M. J. Schuster, S. D. Riedel, R. Belder, D. Leidner, A. Wedler, M. Beetz, and F. Stulp. Design, Execution and Post-Mortem Analysis of Prolonged Autonomous Robot Operations. *IEEE Robotics and Automation Letters*, 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2794580

- M. Vayugundla, F. Steidle, M. Smisek, M. J. Schuster, K. Bussmann, and A. Wedler. Datasets of Long Range Navigation Experiments in a Moon Analogue Environment on Mount Etna. In *International Symposium on Robotics (ISR)*, 2018

- M. Panzirsch, H. Singh, M. Stelzer, M. J. Schuster, C. Ott, and M. Ferre. Extended Predictive Model-Mediated Teleoperation of Mobile Robots through Multilateral Control. In *IEEE Intelligent Vehicles Symposium (IV)*, 2018. doi: 10.1109/IVS.2018.8500578

- A. Wedler, M. Vayugundla, H. Lehner, P. Lehner, M. J. Schuster, S. G. Brunner, W. Stürzl, A. Dömel, H. Gmeiner, B. Vodermayer, B. Rebele, I. L. Grixa, K. Bussmann, J. Reill, B. Willberg, A. Maier, P. Meusel, F. Steidle, M. Smisek, M. Hellerer, M. Knapmeyer, F. Sohl, A. Heffels, L. Witte, C. Lange, R. Rosta, N. Toth, S. Voelk, A. Kimpe, P. Kyr, and M. Wilde. First Results of the ROBEX Analogue Mission Campaign: Robotic Deployment of Seismic Networks for Future Lunar Missions. In *International Astronautical Congress (IAC)*, 2017
- H. Lehner, M. J. Schuster, T. Bodenmüller, and S. Kriegel. Exploration with Active Loop Closing: A Trade-off between Exploration Efficiency and Map Quality. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017. doi: 10.1109/IROS.2017.8206521
- M. Hellerer, M. J. Schuster, and R. Lichtenheldt. Software-in-the-loop Simulation of a Planetary Rover. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2016
- A. Wedler, B. Rebele, J. Reill, M. Suppa, H. Hirschmüller, C. Brand, M. Schuster, B. Vodermayer, H. Gmeiner, A. Maier, B. Willberg, K. Bussmann, F. Wappler, M. Hellerer, and R. Lichtenheldt. LRU - Lightweight Rover Unit. In *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Noordwijk, Netherlands, 2015
- A. Wedler, A. Maier, J. Reill, C. Brand, H. Hirschmüller, M. J. Schuster, M. Suppa, A. Beyer, N. Y. Lii, M. Maier, H.-J. Sedlmayr, and R. Haarmann. Pan/Tilt-Unit as a Perception Module for Extra-Terrestrial Vehicle and Landing Systems. In *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Noordwijk, Netherlands, 2013

1.6 Outline and Reader's Guide

In Figure 1.4 we give an overview of the organization of this thesis. Its chapters can be clustered into three major parts: In the first part, Chapter 1 to Chapter 3, we introduce the topic and its challenges, describe basic concepts and methods, and present the architecture of our approach. The second part, Chapter 4 to Chapter 6, consists of three chapters, in which we present our methods and main technical contributions. In the third part, Chapter 7 to Chapter 10, we discuss the implementation of our methods on different robot systems, present quantitative and qualitative evaluations,

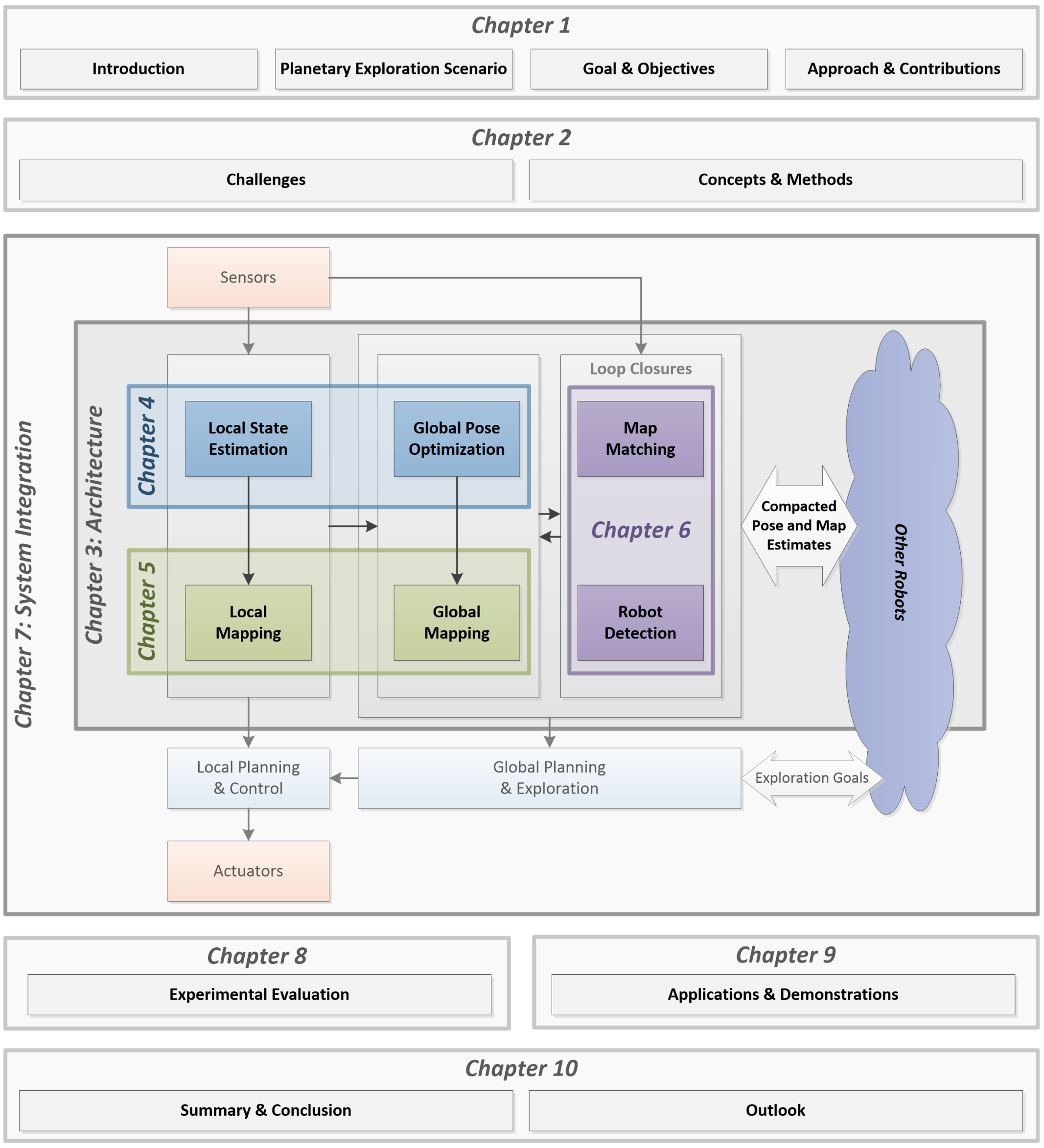


Figure 1.4: *Thesis outline*

and conclude the thesis. The individual chapters each contain discussions of related work as well as cross-references to other parts. Although this thesis is organized to be read in a linear fashion, its chapters are mostly self-contained so that they can also be read individually. The remaining chapters are organized as follows:

In **Chapter 2**, we first identify the challenges faced by heterogeneous teams of autonomous robots tasked to map planetary surfaces. We then introduce the general concepts and methods that form the basis of our approach towards these challenges.

In **Chapter 3**, we introduce our modular multi-robot SLAM architecture, discussing its software components and their interconnections, as well as the aspects of distributed computation and inter-robot data exchange.

In **Chapter 4**, we present our approach for on-board local and global estimation based on a combination of filter and graph SLAM methods. We first summarize the concept of local reference filters and then present our novel method to integrate the state estimates from local filters running on multiple robots in a SLAM graph for global pose estimation.

In **Chapter 5**, we introduce our stereo-vision based 3D mapping system. We first present methods for local terrain classification and the aggregation of high-bandwidth dense depth data into partial maps. We then describe the exchange of these maps between robots and their composition into a global environment model.

In **Chapter 6**, we present our approach to the multi-robot data association problem. It consists of two major techniques to generate intra- and inter-robot loop closures: first, the pairwise matching of partial maps between one or multiple robots and second, the detection and pose estimation of other robots based on visual markers, with focus on an estimation of the respective measurement uncertainties.

In **Chapter 7**, we describe how we integrated our localization and mapping methods on several different robot platforms in order to conduct the experimental evaluations and demonstrations presented in the following chapters.

In **Chapter 8**, we present evaluations of our contributions to methods for multi-robot localization and mapping. They include experiments on loop closure generation, single-robot 6D localization with dense 3D mapping, as well as our full collaborative multi-robot SLAM framework. We analyze and discuss the required computational resources and map our evaluations to the challenges of autonomous planetary exploration, which we identified in Chapter 2.

In **Chapter 9**, we present applications of our methods for robot localization and mapping, validating our approach in relevant Moon- and Mars-like environments. For this, we successfully conducted several public demonstrations with autonomous robots operating in environments that simulate the terrains and visual appearances of lunar and planetary surfaces, including experiments at a Moon-analogue test site as well as the deployment of a heterogeneous team of aerial and ground-based robots.

In **Chapter 10**, we conclude this thesis with a summary and a discussion of open challenges that represent topics for future work.

Chapter two

Autonomous Mapping of Planetary Surfaces

In the first part of this chapter, Section 2.1, we identify the challenges faced by heterogeneous teams of autonomous robots tasked to map planetary surfaces. In the second part, Section 2.2, we introduce the general concepts and methods that form the basis of our novel approach towards these challenges.

2.1 Challenges

Multi-robot systems can benefit from the collaboration within heterogeneous robotic teams.[1] They, however, face a number of challenges to operate efficiently in planetary exploration scenarios. In this context, on-board autonomy is a key capability for each robot. Communication round-trip times range from three to tens of seconds between Moon and Earth and from eight to forty minutes to Mars (ISECG, 2018; Mankins, 1987). Such high latencies render direct teleoperation approaches impractical to impossible and low-level semi-autonomous behavior with frequent human intervention inefficient. A high level of autonomy can speed up a mission and thus lead to a better usage of the available resources and a higher scientific output, in particular in

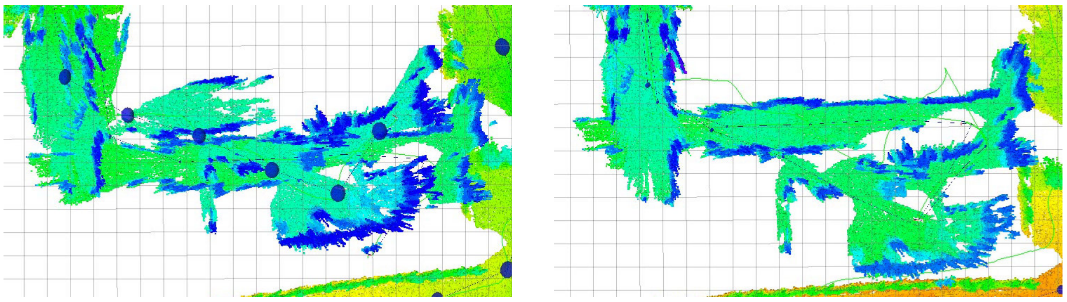
[1]We omit a discussion of robotic manipulation capabilities here. While they are important for planetary exploration missions that involve, for example, the acquisition and analysis of samples, the deployment of instruments, or the assembly of structures, the topic lies beyond the scope of this thesis.

the light of typically limited system lifetimes. Furthermore, communication losses to and between the robots might occur, for example through shadowing effects inside underground caves in the aforementioned scenario. Thus, each system entering such areas should at least be able to operate at a level of autonomy that allows it to return into communication range on its own. This includes a sufficiently fast on-board estimation of its position, of a map of its surroundings and potentially dangerous obstacles therein. In a multi-robot team, the overall mission goals need to be broken down into smaller tasks, which then are distributed to the individual robots, taking their estimated position in a global map, their specific capabilities, as well as dependencies between subtasks into account. This is necessary to fully exploit the potential benefits that can be achieved through parallelization and gained from the complementary capabilities of the robots. In our scenario, regarding a mission on a moon or foreign planet, the exploration of the previously unknown environment plays a central role. In order to solve this task efficiently with multiple robots, it is important to coordinate their actions based on a collaboratively created joint map. This includes planning for the target locations for each robot w. r. t. the maximization of the information while taking into account the execution costs, the restricted resources and the capabilities of the robot systems. Additional trade-offs exist between exploration speed and the quality of the resulting environment model.

In this thesis, we focus on the challenges for collaborative Simultaneous Localization and Mapping (SLAM), as it constitutes the basis for autonomous exploration and is also crucial for most other types of coordinated joint actions with a team of robots. We have identified the following functional and non-functional challenges:

2.1.1 Functional Challenges

- *Robot Localization*
  Each robot requires at all times an up-to-date localization estimate with respect to its environment to navigate and plan therein. In order to allow for meaningful collaboration, agents in a multi-robot team, in addition, require up-to-date pose estimates of the other robots they work with, set in relation to their own coordinate frames and environment models. On Earth, Global Navigation Satellite Systems (GNSSs) combined with dense radio networks provide at least a rough global absolute localization in many places. In contrast, when exploring foreign planets, initially no such infrastructure is available. While it might be possible to set up absolute positioning systems in specific places, most such methods rely on multiple direct line-of-sight connections that



(a) Incremental localization only **(b)** With global optimization

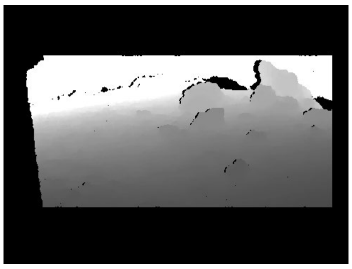
Figure 2.1: *Example of the impact of global optimization on the quality of self-localization: height-colored 3D point cloud maps of an indoor scenario created (a) without and (b) with optimization on a loop closure. While in (a), artifacts due to the drift of incremental localization deteriorate the point cloud, (b) shows a consistent map. See Section 8.2 and Figure 8.16 for details on the experiment.*

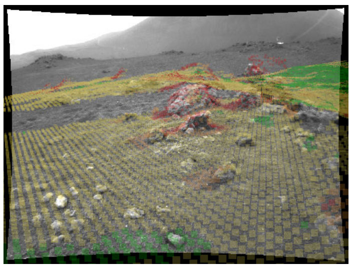
are not available in typical scientifically interesting spots located, for example, close to crater walls or inside caves. Thus, robots operating in such areas are required to maintain local position and orientation estimates solely based on their on-board sensors. Accumulated measurement and model errors, however, lead to drift for an incremental localization with on-board sensors. It is therefore important for a robot to re-localize itself w. r. t. other robots as well as w. r. t. already known parts of the environment. Drift can then be reduced by computing globally optimized localization estimates on such loop closures, as illustrated in Figure 2.1. In a multi-robot team, each robot should be able to further improve its localization estimate by exploiting the information gathered from other robots.

- *Local Mapping and Obstacle Classification*
  A robot requires local metric maps that can be computed and updated online at a sufficiently high frame rate to perform local path planning and fast obstacle avoidance. In order to move and operate in previously unknown environments, it needs to estimate the traversability of its surrounding terrain and identify obstacles within it. This is particularly challenging when using sensor data that exhibits significant noise, such as depth data from stereo reconstruction, as illustrated in Figure 2.2.

- *Robot Detection*
  In order to compute joint localization estimates for all robotic team members, as discussed above, the robots need to set their local sensor data into relation w. r. t. each other. Relative transformations between pairs of robots can be directly estimated through *direct encounters*, i. e., detections of the robots stemming from line-of-sight observations (Saeedi et al., 2016). As the robots

(a) Left camera image


(b) Depth image


(c) Obstacle classification

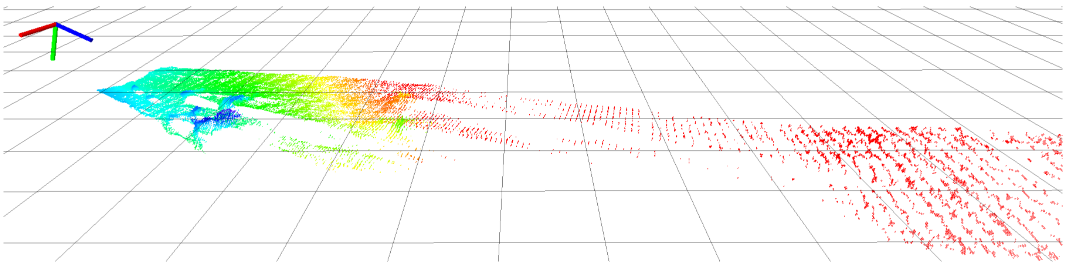
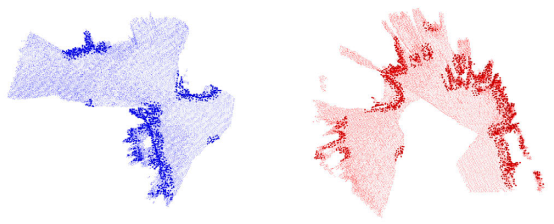

(d) Point cloud (unfiltered, height-colored) generated from the depth image

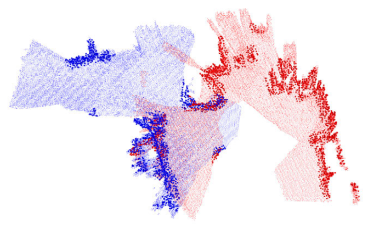
Figure 2.2: *Impression of the LRU's stereo vision-based perception of the environment during our experiments at a Moon-analogue site on Mt. Etna. While light and texture in this setting present ideal conditions for stereo reconstruction, the resulting depth data nonetheless is noisy and in parts fragmentary. See Chapter 5 for details on our obstacle classification and 3D mapping.*

can communicate with each other, it may be sufficient if only some of the robots are able to detect their team mates and share that information appropriately. In the context of planetary exploration, artificial visual markers provide a viable and well-established approach. An estimation of the detection uncertainty and its projection into 6D space for the subsequent pose estimation, however, remains an open challenge. For visual detections, the camera-dependent error characteristics are non-linear w. r. t. distance and angle of view. Furthermore, image noise can cause ambiguities during the estimation of the pose of an object, leading to significant errors. Thus, in order to use these measurements and fuse them with other robot state estimates, it is important to have an estimation of their quality, i. e., their inherent uncertainties.

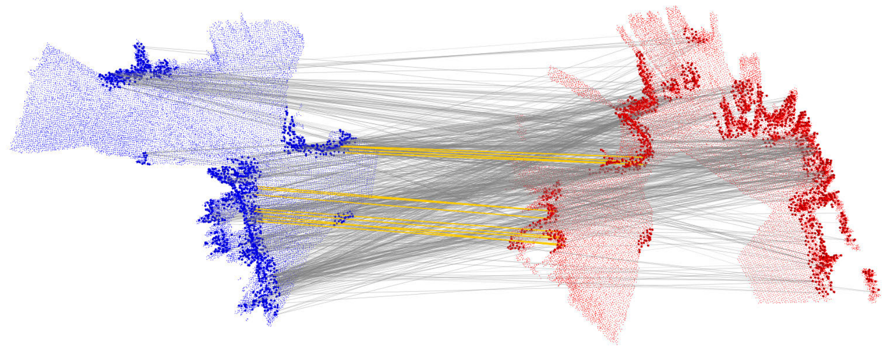
- *Place Recognition*
  In order to re-localize itself and refine its map, a robot needs to be able to recognize places that it visited before. In a multi-robot setup, this extends to the recognition of places from other robots' partial maps. Such *indirect encounters* provide additional transformations between the estimates of different robots (Saeedi et al., 2016). The recognition of places in a robot's own map as well as in maps generated by other robots is required to generate intra- and inter-



(a) *Partial point cloud maps*



(b) *Alignment after map matching*



(c) *Correspondences (grey: all, orange: used) based on geometric 3D features*

Figure 2.3: *Place recognition through the alignment of 3D partial maps based on their geometry: example of a challenging match between partial maps created by two of our rovers. See Section 6.1 and Figure 6.3 for details on our map matching method.*

robot loop closure constraints for global single- and multi-robot localization and mapping. A reliable recognition of places based on noisy sensor data, however, is a challenging task (Saeedi et al., 2016; Lowry et al., 2016). This is particularly true under varying light conditions and in heterogeneous multi-robot teams, in which the individual robots have different sensor setups and different points of view on the environment. In Figure 2.3 we illustrate this challenge with an example of two point cloud-based maps generated by two of our rovers equipped with narrow-angle stereo cameras for navigation.

- *Collaborative Global Mapping*

  The robots within a multi-robot team need to build a joint global map as a shared model of the environment that can, for example, be used for multi-robot exploration goal planning. For this, they are required to share their local partial maps and set them into meaningful relation w. r. t. each other. Once the relative transformations between partial maps are known, they can be merged into a global map. In order to model structures with overhangs or ceilings like, for example, the lavatube caves discussed in the exploration scenario, the robots need to create a three-dimensional map of their environment. In addition, knowledge about already observed and still unknown space is necessary to drive a goal-oriented autonomous exploration. The exchange of partial maps as well as their composition to a collaboratively-built joint global map poses many challenges, particularly in the light of constrained resources and limited communication. Many of them are non-functional challenges, which we will discuss in the following section.

2.1.2 Non-Functional Challenges

- *Space-Qualifiable Sensor Setup*
  Planetary exploration missions pose certain hardware constraints on the components of autonomous robots. Minimal weight and power consumption are crucial for space flight and operations, additional constraints for space-qualifiable hardware include resistance to heat, vibrations, radiation, dust, etc. These requirements limit the choice of sensors available for localization and mapping in comparison to terrestrial applications.[2] Thus, in recent and ongoing planetary exploration missions like the *Mars Exploration Rover Mission (MER)* (Maimone et al., 2006) and the *Mars Science Laboratory (MSL)* (Grotzinger et al., 2012), cameras are preferred to Light Detection and Ranging (LIDAR) systems, which typically are heavier, consume more power and are hard to qualify for space missions. This, however, means that localization and mapping algorithms need to deal with the sensor-specific noise characteristics. Stereo cameras, for example, are suitable for space applications but typically have higher noise levels and allow a less dense depth reconstruction than LIDAR systems, which are often employed for robot navigation on Earth as they permit high-precision measurements within a long range of distances. The restrictions on the selection of sensors thus make SLAM more challenging, requiring the algorithms to take high measurement uncertainties into account.
- *Handling of Uncertainties*
  There are three different sources of uncertainty in a robot's estimate: sensor noise, model errors and limited numerical precision. First, the readings of all typical sensors in our context like IMUs and stereo camera systems exhibit significant levels of noise. Second, every measurement, update or environment

[2]While requirements regarding possible space qualification also exist for all other parts of planetary exploration robots, including their computing hardware, we focus on the sensors relevant for localization and mapping. Our research concerns prototypes and algorithms at a proof-of-concept stage and thus benefits from short development iterations facilitated by the use of terrestrial computing hardware. It is a topic for future engineering to devise implementations for space-qualified processors like radiation-hardened Field Programmable Gate Arrays (FPGAs).

model is a simplification of the real world. Thus, many processing steps do make simplifying assumptions. Third, all of them are in any case limited with regard to the numerical precision of their computation. Thus, none of the robots' pose and map estimates will be exact. As there is no way to prevent being off the mark, the best we can do is to estimate the uncertainty of our knowledge. This is true of the localization as well as of most information in our model of the environment. So it is important to perform our computations within a probabilistic framework. This includes the exchange of information between the robots, which needs to be accompanied by uncertainty estimates. It, however, is a big challenge to handle these uncertainties in an efficient way, many steps requiring a trade-off between accuracy, computational efficiency, and communication bandwidth.

- *Online and Real-time Algorithms*
  In order to autonomously navigate and avoid obstacles in previously unknown terrain, each robot needs to have up-to-date pose and map estimates available at all times. For multi-robot coordination, this extends to estimates for all other robots within communication range, leading to a requirement for an online computation of local and global SLAM estimates. In our context, "online" refers to an estimation "at frame rate", i. e., sufficiently fast for a mobile robot to react on new visual input without pausing its autonomous operation. A subset of the algorithms in addition has to satisfy real-time requirements, in particular when providing input to high-frequency control loops on highly dynamic systems. An example is the real-time local state estimation required for the control and stabilization of flying systems like Unmanned Aerial Vehicles (UAVs).
- *Fault Tolerance*
  In planetary exploration missions, communication losses to Earth as well as between the individual robots within a multi-robot system are to be expected. Thus, the robots should not rely on the availability of communication links, neither between each other nor to some central node or operator. Furthermore, robots operating in challenging environments far away from any possibility of direct human intervention are always at risk of serious hardware and software failures. These can lead to erroneous sensor measurements and estimates, or, in the worst case, the loss of individual robots. In order to gain robustness at the level of a robotic team, such failures must not endanger the whole mission.
- *On-board Computation*
  In order to achieve robustness w. r. t. communication losses and failures of individual robots, as discussed above, the computation of all crucial algorithms needs to be decentralized. Thus, each robot needs to be able to run all algo-

rithms critical to its local autonomy on its on-board computation hardware. This avoids single points of failure and, in case of broken communication links, allows each robot to continue the mission or at least to return within communication range on its own.

- *Low-Bandwidth Communication*
  Communication links between the robots of a multi-robot team as well as between the robots and an operator at a ground station are limited in bandwidth. Low-bandwidth communication typically requires less energy, a precious resource on planetary exploration missions. This raises the challenge of compressing or locally aggregating high-frequency and high-bandwidth data that need to be exchanged between the robots.
- *Distributed Computation*
  While decentralized on-board computation allows for local autonomy, simultaneously processing all the data on all the robots would result in a major computational overhead. Furthermore, as discussed above, a transfer of the raw sensor data is infeasible over low-bandwidth communication channels. Thus, a distributed system is needed to efficiently process high-frequency and high-bandwidth sensor data. Furthermore, it allows to satisfy the requirements posed by real-time algorithms as raw measurement data can be processed on each robot locally, close to its sensors.

2.2 Concepts and Methods

In this section, we introduce the general concepts and methods that we will use throughout the thesis to describe and discuss our novel approach to multi-robot Simultaneous Localization and Mapping (SLAM) for autonomous planetary exploration. For a more detailed discussion of the individual aspects of existing SLAM systems and their relation to our approach, we refer the reader to the “Related Work and Discussion” sections at the end of the respective chapters of this thesis.

Simultaneous Localization and Mapping (SLAM) refers to the concurrent estimation of the state of a moving robot system and the construction of a map as a model of its environment. In the case of *3D localization*, typically used for ground-based robots operating in mostly planar environments such as indoor rooms and hallways, the state consists (at least) of the robot’s x and y coordinate and its heading (yaw) angle. For *6D localization*, required for aerial robots as well as ground-based systems operating in geometrically complex environments and rough terrain, the state is extended to contain estimates of the full x, y, z position and $roll, pitch, yaw$ orientation angles. Most metric maps created by SLAM systems are either 2D, i. e., planar and representing the footprint of the objects in the environment at or up to a certain height, 2.5D, i. e., digital elevation maps modeling the height above ground, or 3D, allowing to model complex geometry such as caves, overhangs or objects with arbitrary shapes. We focus on *online* SLAM that, in contrast to *offline* (batch) methods, runs at a frame rate sufficient for a robot to continuously act on the results without significant delays caused by the computation of up-to-date estimates. An application example is a robot planning its path and avoiding obstacles based on its latest map and localization estimates therein. Online SLAM is often realized via incremental methods that provide up-to-date estimates by sequentially adding new sensor data to the estimation as it comes in. Many online methods thus only compute estimates for the latest robot pose and map, whereas batch methods typically solve the full SLAM problem, i. e., estimating the complete robot trajectory (Thrun et al., 2005).

We further distinguish between *local* and *global estimation*. Local estimation either neglects or aggregates, i. e., merges, data beyond a certain time horizon, resulting in a loss of information. While this allows to limit the computational effort required for each computation step, estimates for unobservable parts of the state will be subject to potentially unbounded drift over time or traveled distance. In planetary exploration scenarios, we assume that absolute measurements of a robot's position and its *yaw* angle w. r. t. a global reference frame are not available, thus rendering these parts of its state unobservable.[3] Regarding the environment model, local mapping refers to the creation of a map of limited size that typically features the immediate surroundings of the robot. In contrast, global estimation takes all available measurements into account, either in their original or aggregated forms. The drift from local estimation is reduced either by absolute measurements or by long-term loop closures, i. e., measurements or estimates that reference back to previously acquired information. An example is the re-localization through the recognition of previously visited places that have been added at earlier points in time to the global environment model. In global estimation, the map as a model for the environment is anchored in a fixed coordinate frame[4] and contains all previously visited areas. For multi-robot teams, global estimation further combines the data and environment model generated by all robots participating in the collaborative effort. The drawback

[3]While compasses and GNSS receivers can provide such measurements in outdoor environments on Earth, moons or foreign planets typically neither feature homogeneous and well-known magnetic fields nor do they have GNSS infrastructure available.

[4]A global map frame is typically set as either the initial pose of the robot or relative to some external reference coordinate system.

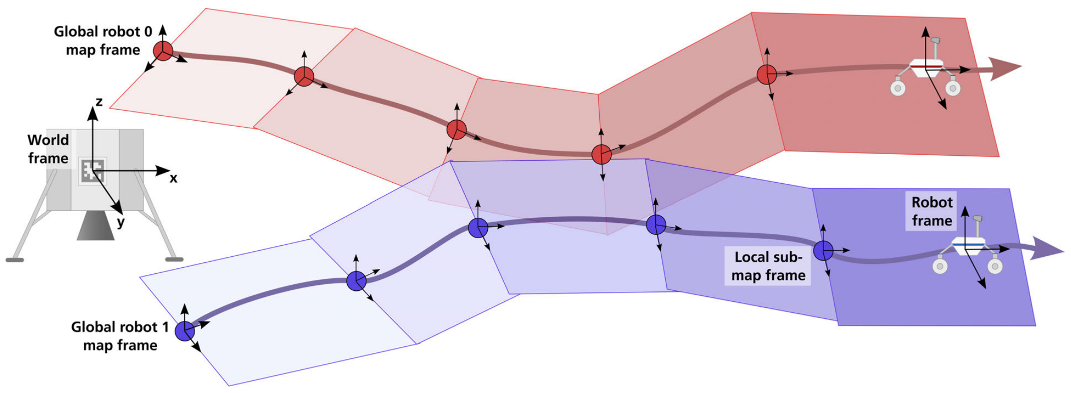


***Figure 2.4:** Sketch of the coordinate frames relevant in local and global estimation and of the global map that is partitioned into the local submaps created by two robots*

of global estimation is its potentially unbounded computational effort that typically grows with the size of the environment model or the runtime of the system.

In our approach, we combine local and global methods to get the best from both worlds. Local estimation gives fast estimates required for reactive behavior while global estimation provides larger-scale multi-robot environment models and a localization of all agents therein. In Figure 2.4, we sketch the relevant coordinate frames in a multi-robot scenario: The *robot frames* define the base coordinate systems of the robots themselves and are estimated for each system in a local reference frame, i. e., the *submap frame* defined by the robot's latest partial map. All of these local frames are defined w. r. t. the respective robot's global *map frame*.[5] When collaboratively creating maps, the robots estimate the frames of all other robots as well as of all partial maps with respect to their own global map frame. In case the robots need to reference themselves globally with respect to the external world, an optional *world frame* can be introduced, for example defined by a static artificial landmark, such as a lander in planetary exploration scenarios. We refer to Chapter 4 for details on our combination of local and global estimation methods.

There exists a large body of related work on SLAM. For a general overview of the basic approaches, see Thrun et al. (2005), Durrant-Whyte and Bailey (2006), and Bailey and Durrant-Whyte (2006). The more recent work of Cadena et al. (2016) gives an introduction to modern SLAM approaches, highlighting current challenges with focus on single-robot systems. Saeedi et al. (2016) introduce additional concepts regarding multi-robot SLAM and give an overview of state-of-the-art systems.

Most SLAM frameworks can be structured into two parts, a *front-end* and a *back-*

[5]The global map frame is defined for each robot independently of the others as robots might start out without any initial knowledge about their relative positions w. r. t. to each other. Thus it would be impossible to enforce a joint global frame at all times.

end (Grisetti et al., 2010; Cadena et al., 2016). The front-end is concerned with data association, i. e., abstracting sensor data into models suitable for estimation. This means connecting measurements and the estimated environment model, for example, by recognizing landmarks, other robots, as well as previously visited places. This includes all methods to generate short- and long-term loop closures, such as the extraction, identification, and tracking of features, markers, and other types of landmarks. The task of the back-end of a SLAM framework is to perform inference on the data generated by the front-end, i. e., the estimation of pose, trajectory, and map hypotheses. This can be formulated as a, typically non-linear, optimization problem. In case of Bayesian single-hypothesis methods, such as graph optimization (described below), the outcome is often a Maximum a Posteriori Probability (MAP) estimate of the robots' poses and map.

In the following Section 2.2.1, we first discuss different map representations, as they determine the choice of methods for the other parts of the system. In the subsequent Section 2.2.2, we describe the data association challenge with focus on vision-based loop closure methods suitable for our planetary exploration robots, which are all equipped with cameras as their exteroceptive sensors. We then introduce three major filter and optimization techniques in Section 2.2.3, which can be employed as part of the SLAM back-end. In Section 2.2.4, we discuss concepts for data distribution and exchange in multi-robot systems. After introducing the basic concepts in this chapter, we present our multi-robot SLAM architecture and compare it to related approaches in Chapter 3.

2.2.1 Map Representations

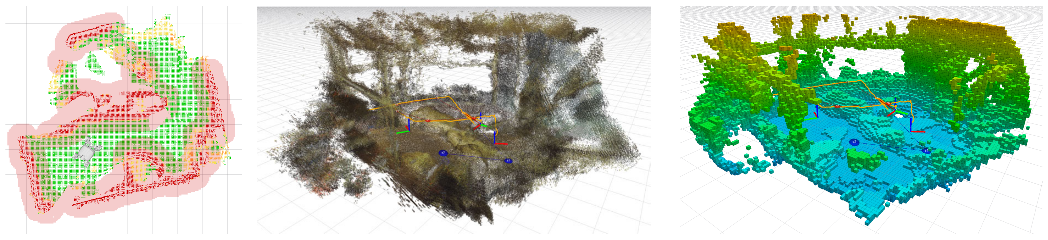
In general, maps created by mobile robots are sources of information that enable or support autonomous robots as well as human robot operators to fulfill their tasks. They constitute models of the environment that can, among others, be used for navigation and planning. Robots exploring previously unknown areas use maps, for example, to identify target locations and to find a safe way back. As map data can be represented in many different ways, particular representations are chosen as a trade-off based on their characteristics regarding specific applications.

Robotic maps can be categorized into one of three broad types: *metric*, *topological* and *semantic*. Metric maps represent the geometry of the environment and the coordinates of objects therein based on a metric space with a distance function that maps pairs of coordinates to $\mathbb{R}$. An example would be the floor plan of a building that is drawn up to scale. Topological maps represent only relationships between points of interest, but, in general, lack scale, distance, and direction information as part of their representation. This network of relationships is typically visualized by a graph structure with nodes representing objects or points of interest and edges their relationships. Examples would be subway maps, as they represent the relationships, i. e., connections, between stations, but not their exact location in the real world. Semantic maps add a semantic layer to either metric, topological, or hybrid representations. They add attributes to elements (pixels, voxels, clusters, nodes, relationships, etc.) of a map, which have a certain meaning grounded in the real world. Examples are the classification of rooms, known objects, or different types of terrain in a map, but also the annotation of object properties like a door being open, an artificial light source being "active", etc. Such attributes are often task-oriented w. r. t. to a robot's skills or mission, for example defining the reachability of certain locations or the types of manipulations that can be applied to certain objects.

The concept of *submapping* describes the partitioning of a map into a number of partial maps, so-called *submaps*, as sketched in Figure 2.4. It can be applied to any of the aforementioned map types. In addition to the partial maps themselves, a submapping-based framework needs to keep track of the relations between them or w. r. t. a central node. For example regarding metric maps, these relations can be represented by relative metric transformations and a central node by a global coordinate frame. Submapping has several benefits: In online mapping, it allows many algorithms to restrict the amount of data that needs to be processed and held in working memory to a single or a limited set of submaps. This can be a prerequisite for them to scale to large models. In addition, submaps and their relations can be applied in a hierarchical approach to integrate local partial maps into a global model. Furthermore, they can be used for place recognition as well as as a unit to be shared between robots in collaborative mapping frameworks.

In this thesis, we focus on the creation of dense metric maps to support autonomous robotic exploration in our planetary mission scenario. Dense metric maps enable path planning and obstacle avoidance and thus form the basis for autonomous navigation in previously unknown unstructured environments. They are also used for single- and multi-robot re-localization and can guide autonomous exploration planning. In addition, regarding previously unknown areas on foreign celestial bodies, the creation of metric maps as models of the geometry of the environment can be a goal in itself. Such maps can, for example, be used for visualization, to geolocate scientific measurements or sample acquisitions or, in future work, be enriched with topological and semantic annotations.

There are a multitude of different dense metric map representations such as 2D



(a) *Grid map* ***(b)*** *Point cloud* ***(c)*** *Voxel map*

Figure 2.5: *Examples of metric map types: top-down 2.5D grid map with terrain classification results for local path planning (a), colored 3D point cloud map of complex 3D environment (b), and probabilistic 3D voxel grid representation of the same scene (c). All three maps were created online and on board our robots during multi-robot demonstrations at the International Astronautical Congress (IAC) 2018, described in detail in Section 9.4.*

grid maps, 2.5D height maps, 3D voxel grid maps, polygon- or triangle mesh-based maps, surface maps, point cloud or surfel (points with orientation) maps, etc., as well as various combinations thereof as multi-layer and multi-resolution maps (Dellaert and Kaess, 2017; Triebel et al., 2006; Oberländer et al., 2014). All of these map types provide different trade-offs between various factors like, for example, information content, accuracy, representation of complex geometries, representation of uncertainty, computational efficiency, memory requirements, and visualization possibilities.[6] The choice of a particular representation thus depends on its application.

Our envisioned planetary exploration scenario includes the mapping of complex geometrical structures in the environment, such as caves and different types of rock formations. For our mapping framework, we thus decided to use a combination of 2.5D *grid maps* for obstacle avoidance with ground-based robots and dense 3D *point clouds* as well as *probabilistic voxel maps* to describe the 3D geometry of the environment. We show examples of environment models created by our mapping system using these three types of metric maps in Figure 2.5. In the following paragraphs, we give a basic introduction to these map representations and discuss their application in our framework in detail in Chapter 5.

2D Grid Maps

Grid maps discretize the environment model into 2D cells of a fixed, predefined resolution. Each cell can represent one or multiple values like occupancy, height (in case of 2.5D elevation maps), or further semantic annotations, for example, to designate objects or to classify terrain by its type or traversability. These values can also represent probabilities on binary variables or distributions over multiple classes

[6]In Section 5.3.1, we discuss several different grid-based map types for terrain classification and obstacle avoidance.

or continuous variables. In such cases, grid maps typically assume a probabilistic independence between their individual cells.

3D Point Clouds

In contrast to grid maps that follow a predefined discretization of the space, 3D point clouds describe sets of points at arbitrary coordinates (x, y, z). Each point can be augmented by additional information, such as color or semantic annotations like object classifications. Point clouds are fast to generate and aggregate from depth data and easy to post-process, for example for 3D visualizations or re-localization via place recognition based on the geometry of the environment. They do not have a fixed resolution, but post-processing filters can be used to reduce their density by removing or aggregating points in local neighborhoods. As drawbacks, point clouds neither model free nor unknown space and sensor noise cannot be dealt with directly (e. g., via probabilistic aggregations), but only indirectly by detecting and filtering spurious points as a post-processing step.

Probabilistic 3D Voxel Maps

While point clouds are suitable to represent surface geometry, voxel maps, i. e., three-dimensional grid maps, allow the representation of volumetric models. Similar to points in point clouds, voxels in general can be used to spatially locate arbitrary data. A typical application in robotic mapping are occupancy maps, which we describe based on the "OctoMap" framework for probabilistic voxel-grid mapping introduced by Wurm et al. (2010) and Hornung et al. (2013). Such maps allow an explicit distinction between *occupied*, *free* and *unknown space*.[7] This is important for robotic path planning, as unknown space either needs to be avoided or traversed more carefully than known areas. Furthermore, knowledge about known and unknown areas is required to guide autonomous exploration in order to determine goal locations to be explored next. A probabilistic modeling of the boolean occupancy property per voxel allows to deal with sensor noise, reflections, and dynamic obstacles by integrating multiple measurements as well as with changing environments by updating the model of previously mapped areas with new measurement data. It further allows the integration of data from multiple sensors of one or multiple robots according to their respective sensor models.

[7]The "OctoMap" framework by Wurm et al. (2010) and Hornung et al. (2013) does not distinguish between unknown (not yet measured) and uncertain (conflicting measurements) parts of a map. While this distinction might become important in exploration scenarios with complementary sensors or significant sensor noise, e. g., due to reflecting surfaces, it would result in more complex models requiring more memory and a higher computational effort.

2.2.2 Data Association

Data association describes the matching of sensor measurements with each other or with a model of the environment. This requires finding suitable abstractions and aggregations of the raw sensor data. Data associations can be sequential, for example, matching image features frame by frame in visual odometry systems, or span longer periods of time, then referencing back to previously acquired parts of the environment model. The latter are referred to as *loop closures* as they create loops in a graph-based visualization of the estimation problem, with state and model variables being represented by nodes and measurement constraints by edges in the graph. Loop closures are crucial for any kind of global estimation as they allow to significantly reduce the estimation error of the current state w. r. t. states estimated prior to the loop. For example in localization that lacks global measurements such as GNSS fixes, loop closures can reduce the drift accumulated by sequential estimation over long trajectories. In multi-robot systems, we distinguish between *intra-robot* and *inter-robot* loop closures. The former only use information acquired by a single robot whereas the latter connect the estimates of two or more robots.

Loop closures can be generated from the data of a wide variety of exteroceptive sensors with different modalities, such as the perception of visual, acoustic, or tactile cues or the reception and timing of radio signals. In order to focus on our space exploration scenario and its robots with camera-based perception, we limit the following introduction to suitable vision-based methods. *Visual place recognition* denotes the general method for re-localization based on a model, i. e., map, of the environment. For an overview on the topic, we refer to a recent survey of state-of-the-art methods by Lowry et al. (2016). A particular challenge for visual place recognition is to decide whether measurements refer to a new part or a previously observed part of the environment and, in the latter case, to identify that part in the estimated model. Many approaches build on the *identification*, *description*, and *matching* of locally or globally identifiable *landmarks* on different levels of abstractions. These can, for example, be features from individual images, features extracted from the local 2D or 3D geometry, or, on a semantic level, objects that are being recognized and located. While the identification of such landmarks can be sufficient for purely topological localization and mapping, full metric SLAM approaches require further information by measuring metric quantities, i. e., distances in a metric space, between a robot's pose and the landmarks. That can, for example, be range measurements (distance on the positions), bearing measurements (distance on the orientations), or combinations thereof on different axes, leading to position or full pose measurements.

In addition to matching the most recent sensor data against an a-prior given or online estimated model, overlapping parts of a model that have been created either at different points in time or by different robots can be set into relation to each other. In the context of mapping, this is referred to as *map matching*. Based on the aforementioned concept of submapping, i. e., splitting a global model (the map) into a number of local partial models (the submaps), *submap matching* denotes the matching of these partial models against each other.

An important challenge in the identification of places arises from so-called *perceptual aliasing*, describing the problem that different places in the environment might look very similar for certain types of sensors. This leads to ambiguities in the landmark matching process and can thus result in false positive matches. An example would be man-made structures such as long hallways that visually look almost exactly the same at each corner. Sufficiently large partial models created via the aggregation of sensor data can reduce the amount of perceptual aliasing as submaps are more descriptive and therefore less ambiguous to match than individual measurements. They can also lower the dependence of the matching on the similarity of the position and viewpoint in comparison to earlier measurements.

In multi-robot systems, we distinguish between *indirect encounters*, i. e., inter-robot loop closures based on place recognition, and *direct encounters*, i. e., loop closures based on direct line-of-sight measurements between multiple robots (Saeedi et al., 2016). Direct encounters for camera-based systems rely on the visual identification of other robots. This task can be supported by artificial markers attached to the robots that allow an easier identification and disambiguation in order to obtain more accurate relative position or pose estimates.

In general, we need to distinguish between *single-* and *multi-hypothesis* data association (Bailey and Durrant-Whyte, 2006). Estimating and keeping track of multiple hypotheses allows to deal with high uncertainties and potential error sources, such as perceptual aliasing, by resolving possible ambiguities at future points in time once more information becomes available. It, however, comes with added complexity, computational effort, and memory requirements, in particular as it typically means to keep and update multiple maps in parallel. Therefore, many state-of-the-art approaches rely on single-hypothesis estimates. Erroneous data associations, however, can lead to significant pose and map errors, in particular in online SLAM methods, for which any consensus-based outlier filtering is only possible up to the current point in time. If multiple hypotheses can be used mainly depends on the choice of the SLAM back-end for optimization, which we will discuss in the next section.

We refer to Chapter 6 for detailed information on how we apply selected data association techniques for global multi-robot localization and mapping. In particular, we create intra- and inter-robot loop closures using submap matching (Section 6.1) and robot detections (Section 6.2).

2.2.3 State Estimation

State estimation techniques allow a robot to create a model of its environment and localize itself therein based on noisy and uncertain sensor data. The robot state typically comprises its position and orientation w. r. t. a map of its environment in order to enable robot navigation and planning. It can contain additional variables, such as the robot's velocity, required for the control of highly dynamic systems, or sensor biases and calibrations, in particular when they change over the runtime of a system and thus need to be estimated online. The selection of state variables as well as the requirements on the temporal resolution and estimation accuracy are system- and application-dependent. Filter and optimization techniques are used to estimate the state of the robot and a map of its environment. The methods differ with regard to various aspects such as their modularity, simplifying assumptions, computational complexity, accuracy, the consideration of multiple hypotheses, and the estimation of uncertainties. In this thesis, we combine different techniques to benefit from both fast but local filter-based estimation and slower but global optimization-based methods, as we will discuss in detail in Chapter 4.

In its probabilistic formulation, SLAM is concerned with simultaneously estimating the posterior over a robot's trajectory along with its map. Let $\boldsymbol{m}$ denote the robot's map, $\boldsymbol{x}_{1:t} = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$ its trajectory, $z_{1:t} = z_1, \ldots, z_t$ the robot's observations and $\boldsymbol{u}_{1:t} = \boldsymbol{u}_1, \ldots, \boldsymbol{u}_t$ its control inputs.[8] According to Thrun et al. (2005), we can distinguish between full SLAM that estimates the posterior for the full robot trajectory

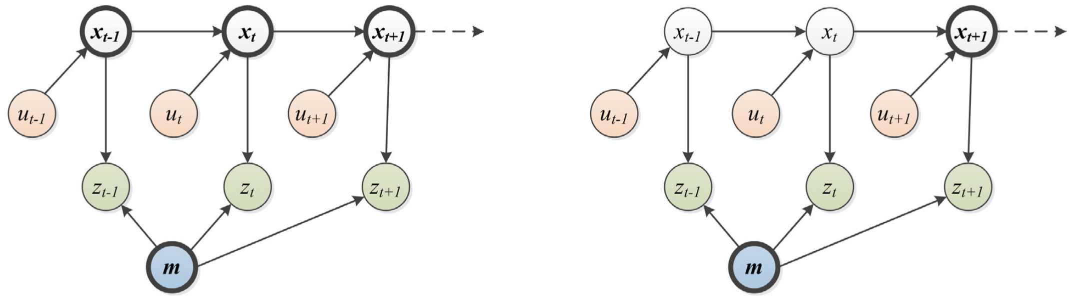
$$P(\boldsymbol{x}_{1:t}, \boldsymbol{m} | z_{1:t}, \boldsymbol{u}_{1:t-1}) \tag{2.1}$$

and online SLAM that estimates the robot's latest pose $\boldsymbol{x}_t$ at time t along with its map:

$$P(\boldsymbol{x}_t, \boldsymbol{m} | z_{1:t}, \boldsymbol{u}_{1:t-1}) \tag{2.2}$$

In Figure 2.6, we show a visualization of the probabilistic structure of both variants in a graphical model. Online SLAM results from integrating out the robot's past poses

[8]The control inputs $\boldsymbol{u}_{1:t}$ are not always available and then can either be omitted or replaced by inertial measurements (as, for example, in our case of a local reference filter discussed in Section 4.1).



(a) *Graphical model for full (batch) SLAM* ***(b)*** *Graphical model for online SLAM*

Figure 2.6: *Graphical model of the SLAM problem similar to the visualization by Thrun et al. (2005). Bold nodes denote the variables estimated in its full (a) and online (b) formulations.*

from the full SLAM problem stated in Equation 2.1:

$$P(\boldsymbol{x}_t, \boldsymbol{m} | z_{1:t}, \boldsymbol{u}_{1:t-1}) = \int \cdots \int P(\boldsymbol{x}_{1:t}, \boldsymbol{m} | z_{1:t}, \boldsymbol{u}_{1:t-1}) \, d\boldsymbol{x}_1 \; \ldots \; d\boldsymbol{x}_{t-1} \tag{2.3}$$

This is typically done incrementally, i. e., one step at a time based on the availability of new data.

For cases that go beyond toy examples, the computation of the full posterior usually is computationally infeasible. This is due to the high dimensionality of the problem, i. e., the large number of variables. Thus, in practice, approximative techniques are used to compute SLAM estimates. These either only estimate the most likely hypothesis as a point estimate of the distribution, e. g., in case of Maximum a Posteriori Probability (MAP) estimation, or track a limited number of different trajectory and map hypotheses.

Most SLAM approaches can be classified into one of three major techniques: Kalman-based filters like Extended Kalman Filters (EKFs), Rao-Blackwellized Particle Filters (RBPFs), and graph optimization approaches. In the following paragraphs, we give a brief introduction to each of them and discuss their suitability for multi-robot SLAM. We refer to the introductory works by Thrun et al. (2005), Durrant-Whyte and Bailey (2006), Bailey and Durrant-Whyte (2006) and Cadena et al. (2016) for more details on the individual and related techniques.

Extended Kalman Filter (EKF)

One of the earliest and most widely used techniques for robot state estimation and feature-based SLAM systems are Gaussian filter algorithms such as the Kalman Filter (KF) and its derivatives (Thrun et al., 2005). They implement a belief propagation for continuous states based on the Markov assumption, i. e., the assumption that future states only depend on the current state and inputs and are independent of any events that happened before. This property allows small and bounded computation times for each step when employed for incremental online estimation. In basic Gaussian filters, the posterior probability is modeled as a multivariate Gaussian. They thus are only well-suited for unimodal distributions as they only estimate a single most likely hypothesis and its uncertainty. They further assume white Gaussian noise for both the robot's motion as well as for its sensor models. While Kalman Filters (KFs) compute the exact Bayesian estimate for linear systems, Extended Kalman Filters (EKFs) use approximations to handle nonlinear problems, such as any robot pose estimations that involve rotations. The robot state estimate is updated via nonlinear prediction and measurement functions g and h:

$$\boldsymbol{x}_t = g(\boldsymbol{u}_t, \boldsymbol{x}_{t-1}) + \boldsymbol{\epsilon}_t \tag{2.4}$$

$$z_t = h(\boldsymbol{x}_t) + \boldsymbol{\delta}_t \tag{2.5}$$

with $\boldsymbol{\epsilon}_t$ describing the process noise and $\boldsymbol{\delta}_t$ the measurement noise. During the prediction and update steps of EKFs, these functions are typically linearized at the latest estimated mean of the state $\hat{\boldsymbol{x}}_t$ via first order Taylor expansion. As the errors introduced by linearization rise with the amount of posterior variance, EKFs are better suited for estimation problems with low uncertainty. Further variants of Kalman-based filters with different characteristics exist that either employ a different parametrization of the Gaussians (e. g., Information Filter) or use different approximations of the nonlinear propagation steps (e. g., via linearization or unscented transformations). They, however, share most of the aforementioned major properties and limitations.

Bounded computation times on each prediction and update step allow Kalman-based filters to satisfy real-time requirements posed by control loops of highly dynamic systems such as, for example, Micro Aerial Vehicles (MAVs). As they have a limited look-back due to their Markov assumption, they are better suited for local than global state estimation. In our SLAM framework, we thus employ EKFs as our local reference inertial navigation filters running locally on each robot, which we describe in detail in Section 4.1. For the global optimization problem, we, however, regard the technique of EKFs as less suited. As EKF SLAM models landmark-based maps via multivariate Gaussians, they typically imply a computational effort and memory requirement that grow quadratically with the number of landmarks (Durrant-Whyte and Bailey, 2006). Furthermore, globally unbounded uncertainty on unobservable state variables can lead to large approximation errors due to the filter's linearizations.

Rao-Blackwellized Particle Filter (RBPF)

Particle Filters (PFs) are non-parametric Bayes filters that model their posterior distribution by sampling a set of weighted, discrete particles. Each particle represents a state hypothesis, the set of all weighted particles approximates the posterior distribution $P(\boldsymbol{x}_{1:t}, \boldsymbol{m}|z_{1:t}, \boldsymbol{u}_{1:t})$. In contrast to EKFs, PFs thus can model arbitrary, also multi-modal, distributions. The computational challenge lies in the required number of particles to adequately represent complex distributions. While techniques such as importance resampling help to mitigate this problem, it is still relevant, in particular for high-dimensional state spaces.

A variant of PFs suitable for SLAM, in particular with 2D occupancy grid-based map representations, are Rao-Blackwellized Particle Filters (RBPFs), as they allow for an efficient computation with small numbers of particles (Hähnel et al., 2003; Grisetti et al., 2007). They reduce the dimensionality of the PF state space by separating the estimation of poses and maps. Their key idea is the factorization of the joint posterior probability $P(\boldsymbol{x}_{1:t}, \boldsymbol{m}|z_{1:t}, \boldsymbol{u}_{1:t-1})$ of the robot trajectory $\boldsymbol{x}_{1:t} = \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$ and the map $\boldsymbol{m}$ into a distribution over potential trajectories and their respective distributions of potential maps:

$$P(\boldsymbol{x}_{1:t}, \boldsymbol{m}|z_{1:t}, \boldsymbol{u}_{1:t-1}) = P(\boldsymbol{x}_{1:t}|z_{1:t}, \boldsymbol{u}_{1:t-1}) \cdot P(\boldsymbol{m}|\boldsymbol{x}_{1:t}, z_{1:t}) \tag{2.6}$$

The trajectory hypotheses are tracked by a Particle Filter (PF), while the environment is modeled by a probabilistic grid map for each trajectory hypothesis. Assuming the independence of all N grid map cells $\boldsymbol{m}_i$ with $i \in \{1, \ldots, N\}$, i. e.,

$$P(\boldsymbol{m}|\boldsymbol{x}_{1:t}, z_{1:t}) = \prod_{i=1}^{N} P(\boldsymbol{m}_i|\boldsymbol{x}_{1:t}, z_{1:t}) \tag{2.7}$$

allows a straightforward computation of the map hypothesis for each particle by iteratively integrating measurements along its estimated trajectory. *FastSLAM* (Hähnel et al., 2003) and its successors *GMapping* (Grisetti et al., 2007) are widely-used RBPF implementations for 2D occupancy grid map-based SLAM. The GMapping algorithm significantly outperforms a naïve RBPF implementation by applying adaptive resampling to mitigate particle depletion and utilizing a scan matcher to improve the proposal distribution. This allows it to estimate 2D maps with a relatively small number of particles, making it suitable for online and on-board planar localization and mapping applications. While these algorithms were initially designed for LIDAR-based systems, we recently demonstrated their applicability to stereo vision data with appropriate preprocessing (Brand et al., 2014). We employed this RBPF-based

SLAM for single-robot 2D occupancy mapping in early demonstrations (Section 9.1), in order to evaluate the suitability of our obstacle classification algorithm for SLAM (Section 8.1.1), and as a benchmark to evaluate the accuracy of our novel 6D SLAM framework (Section 8.2.1).

Extending RBPFs from three to six degrees of freedom, however, requires their number of particles to grow exponentially with the size of the state space to avoid weight collapse (Quang et al., 2010), leading to very challenging computational and memory requirements for 6D SLAM (Welle et al., 2010). Similar issues apply for multi-robot SLAM. While multi-robot RBPF variants exist for planar mapping, they raise additional challenges and restrictions like requiring rendezvous events for inter-robot exchange of information (Carlone et al., 2010) or assuming all robots to move at approximately the same speed in order to avoid particle depletion problems (Howard, 2006). Thus, we regard RBPFs as unsuitable for global multi-robot 6D estimation and focus on the third method, graph SLAM, which we will introduce in the following paragraph.

Graph Optimization

Graph optimization is a technique to model estimation problems as a set of variables and constraints affecting them. The resulting, typically sparse, structure can be described and visualized by a graph model. Factor graphs are suitable to represent such models, in particular when they contain non-binary factors[9] (Dellaert and Kaess, 2017; Cadena et al., 2016). While factor graphs can be applied to a variety of robotic estimation problems, in this work we consider pose graph optimization for SLAM, in which the variables represent 6D poses and the constraints measurements relating to them: Robot poses and landmarks are modeled as nodes in an undirected graph that can be constructed in an incremental fashion at runtime. These nodes are connected via measurement constraints, represented as edges that are weighted according to their respective Gaussian uncertainty estimates. The typically sparse structure of the graph reflects the (in)dependencies of the underlying optimization problem. In Figure 2.7, we give a toy example of a small SLAM graph with robot and landmark poses as variable nodes. A Maximum a Posteriori Probability (MAP) estimation of the variables is computed on loop closures through iterative least-squares error minimization. As the optimizer has access to all measurement constraints, it can apply re-linearizations during its iterative computation, thereby typically achieving a higher accuracy than KF-based approaches. Similar to EKFs, graph SLAM only estimates a single best hypothesis. As graph optimization thus is highly sensitive

[9]Examples for non-binary factors would be unary priors or *n*-ary calibration factors that affect *n* variables.

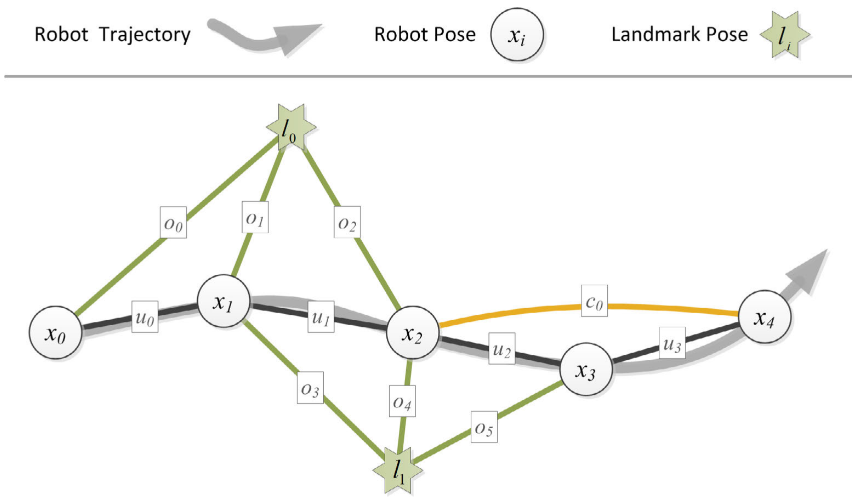


Figure 2.7: *Example of SLAM graph with robot poses and landmarks as variable nodes. Subsequent poses are sampled along the robot trajectory and connected via control or odometry estimates $\boldsymbol{u}_i$. Additional constraints are given by landmark observations $\boldsymbol{o}_i$ as well as additional loop closure constraints $\boldsymbol{c}_i$, which connect non-subsequent robot poses.*

to overconfident erroneous measurements, e. g., from incorrect data associations, robust estimation methods have been developed to reduce the influence of outliers (Latif et al., 2014; Agarwal et al., 2013).

Graph SLAM started out with batch optimization methods (Kümmerle et al., 2011a) for offline use. Recent advances in sparse non-linear optimization, together with incremental approaches such as *iSAM2* (Kaess et al., 2012), however, enabled its application for online and on-board robot localization and mapping. All of these batch and incremental methods solve the full SLAM problem by jointly optimizing for all variable nodes. While modern incremental graph construction and optimization methods can efficiently deal with thousands of nodes and constraints (Dellaert and Kaess, 2017), scaling remains an open issue. The worst-case computational effort on loop closures grows with the number of nodes and constraints and thus typically with the distance traveled by a robot. This challenge can, for example, be approached by constraining the optimization to local regions (Mei et al., 2010), by removing nodes through marginalization (Williams et al., 2014), or, as in our case, by limiting the growth rate of the SLAM graph via a hierarchical estimation framework.

For global 6D multi-robot joint localization and mapping, we consider graph SLAM to be the most promising technique. It allows a straightforward integration of inter-robot measurement constraints between intra-robot subgraphs while keeping the computational complexity at manageable level (Ahmad et al., 2013). We apply 6D pose graph optimization as a global technique at the core of our hierarchical multi-robot SLAM framework, combining the estimates of EKFs running on different robots into a joint graph optimization. In Section 4.2.1, we provide a formalization of graph SLAM based on factor graphs to serve as an introduction to our novel graph topology that we present in Section 4.2.2.

2.2.4 Multi-Robot Data Distribution and Exchange

Data Distribution

An important aspect regarding SLAM for multi-robot teams is the distribution of data processing tasks among the individual agents. We can distinguish between two orthogonal pairs of concepts (Saeedi et al., 2016): Data processing is either *centralized* or *decentralized* and either *distributed* or *non-distributed*. In centralized systems, computations are performed on board a predetermined single robot or a central ground station and the results are then fed back to the other agents. In contrast, in a decentralized system, the processing is done simultaneously by multiple agents. Centralized systems typically can achieve a high efficiency regarding the computational effort and, in case of a ground station, shift complex computations from resource-constrained robots to powerful off-board computation nodes. They, however, create single points of failure and can require large amounts of data to be transferred to and from the central node, in particular when processing high-bandwidth sensor data. Distributed computation means that a task is divided into sub-tasks that are distributed among and processed by the individual robots, which can be coordinated in a centralized or decentralized fashion. In contrast, non-distributed computations are either performed only by a single agent (in centralized systems) or duplicated on multiple agents (in decentralized systems). Distributed data processing is a concept to reduce the computational effort of the individual nodes at the expense of either data aggregation and the resulting loss of information or the need for complex algorithms to achieve convergence and consistency in distributed estimations. In practice, hybrid approaches are used to combine different choices for individual computation steps by making trade-offs based on their particular requirements and properties. We refer to Chapter 3 for a discussion of our hybrid, i. e., decentralized and partially distributed, localization and mapping architecture.

Data Exchange

Closely related to the distribution of data processing is the exchange of data in a multi-robot system. Data can be exchanged on different levels of abstraction: as *raw* *sensor measurements* (e. g., stereo camera images), as a *filtered and preprocessed set of measurements* (e. g., depth images for selected keyframes) or as *aggregated estimates* (e. g., robot poses and partial maps). Filtering and aggregating data can significantly reduce the frequency and size of data transfers compared to the exchange of raw measurements, however, at the cost of a loss of information during such preprocessing steps. Considering these trade-offs is particularly important in the context of planetary exploration, as communication links between systems are restricted in bandwidth due to hardware limitations or their energy consumption, typically rendering a transfer of all raw data infeasible.

Chapter three

Distributed Multi-Robot Localization and Mapping Architecture

In this chapter, we describe our system architecture with focus on our multi-robot localization and mapping framework, which we first published in a conference paper and journal article by Schuster et al. (2015, 2018). In Section 3.1, we start by giving an overview of its software components and their connections to the rest of the system. In the following Section 3.2, we discuss the architecture and interconnections of the key components in more detail and focus on the aspects of distributed computation and data exchange in Section 3.3. In the final section of this chapter, Section 3.4, we discuss our architecture in comparison to related work.

3.1 System Architecture Overview

In the block diagram in Figure 3.1, we present the key components of our multi-robot localization and mapping framework. We indicate their connections to the robots' sensor data on the input side as well as to the robots' control, planning, and exploration components on the output side.

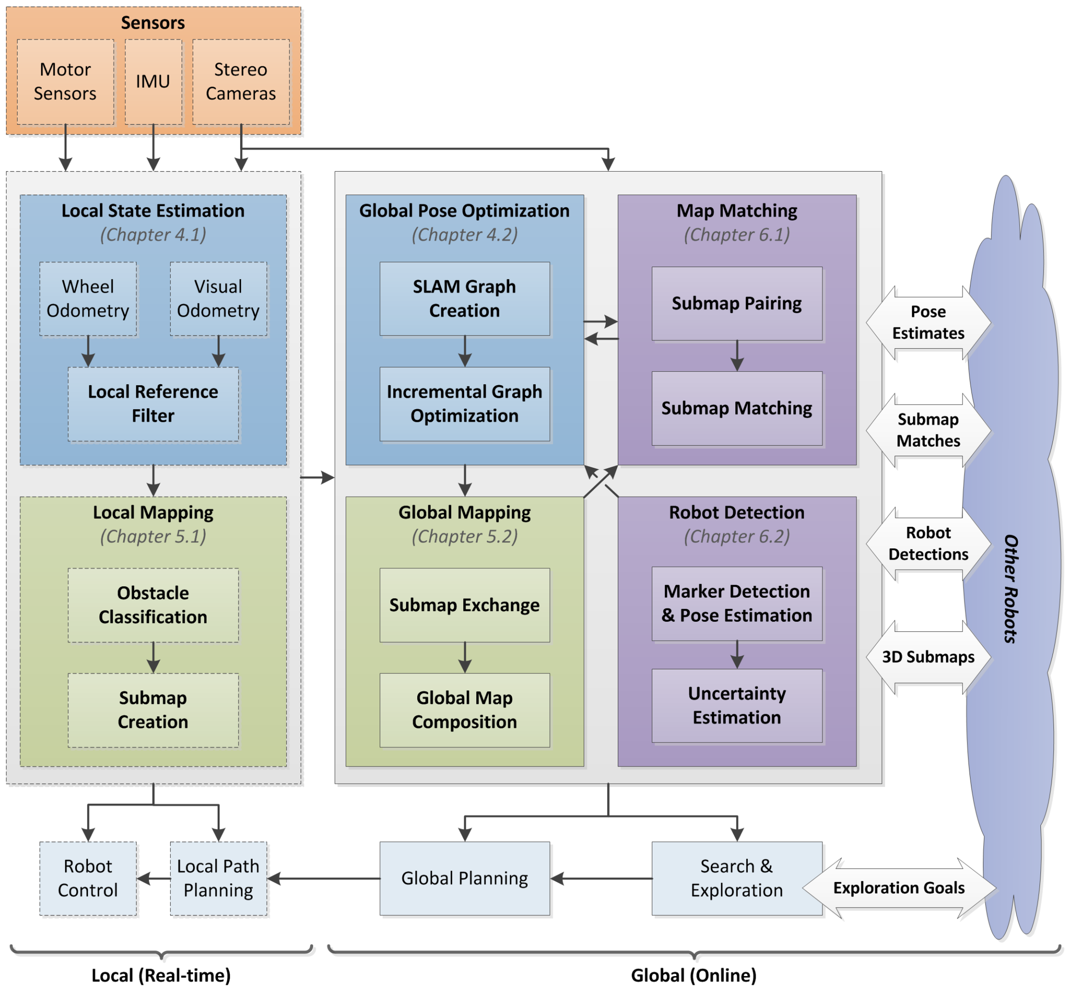


Figure 3.1: *System architecture overview with focus on our multi-robot mapping modules, their interactions, as well as their integration with other components. Modules surrounded by a dashed line are critical for local navigation to run in real-time, whereas the others may run at a slower rate or could even fail without endangering the system.*

The types of input to our localization and mapping components are threefold. The first comes from a stereo camera system that provides color- or greyscale images as well as noisy but mostly dense depth values computed via stereo matching. We provide information on the particular camera setups on our robots as well as their FPGA-based stereo reconstruction in Chapter 7. The second type of input are accelerometer and gyroscope data from an Inertial Measurement Unit (IMU). The third are motor position measurements to determine the pose of a pan-tilt sensor head as well as to compute wheel odometry estimates on some of our ground-based robots. In our localization and mapping pipeline, we compute local and global pose and map estimates based on this data on board each robot, as we describe in more detail in the next section.

The resulting high-frequency local state estimates contain position and velocity data, satisfying real-time properties. This makes them suitable for local planning and rover control, but also for the more challenging stabilization of highly dynamic systems like MAVs (Schmid et al., 2014a). Combining these estimates with local maps allows the realization of local path planning with fast obstacle avoidance. We successfully employed these, for example, for autonomous waypoint-based rover navigation in rough terrain at the *2015 SpaceBotCamp Challenge*, which we present in Section 9.2. While our global multi-robot estimation runs at a slower rate than the local estimation, it provides online estimates suitable for global path and exploration planning. In a multi-robot context, the availability of joint map and pose estimates of all robots constitutes the foundation for coordinated cooperative action like joint exploration of a previously unknown environment. We successfully used the resulting global map for autonomous, information gain-based exploration, and demonstrated it in single-robot experiments (Lehner et al., 2017) as well as in a multi-robot setup with two rovers operating at a Moon-analogue site, which we present in Section 9.3.

3.2 Localization and Mapping Components

Our localization and mapping architecture consists of two major parts, one for real-time local estimation (left part of Figure 3.1), and the other for online global estimation (right part of Figure 3.1). Local estimation on each robot is centered around the aggregation of high-frequency and high-bandwidth measurements under consideration of their respective measurement noise. For local state estimation, we integrate IMU data with a keyframe-based visual stereo odometry (Hirschmüller et al., 2002) and wheel odometry estimates, where available. The latter is computed on some of our rovers by aggregating the position sensor measurements of the driving and steering motors into velocity estimates for the whole rover body, taking into account possible slip in soft terrain (Bussmann et al., 2018). Wheel odometry and visual odometry complement each other well due to their different performance characteristics in different environments. On the one hand, wheel odometry outperforms visual odometry in low-slip areas, such as man-made hallways, as these tend to be untextured and thus are often difficult for visual approaches. On the other hand, visual odometry often performs better in high-slip environments, such as rough terrain outdoor areas, which typically feature almost randomized natural textures, thereby providing ideal conditions for tracking visual features. Furthermore, IMU measurements make the direction of the gravity vector observable and can help to bridge short gaps in visual or wheel odometry data that might occur due to a lack of features or high-slip events. To benefit from the three different sensing modalities, all measurements and estimates are fused in a local reference filter that takes the respective measurement uncertainties into account. It has been developed by Schmid et al. (2014b) as a keyframe-based Extended Kalman Filter (EKF) with time-delay compensation for real-time robust local state estimation. We provide more details on the local reference filter and its integration with our global estimation methods in Section 4.1. On our rovers operating in rough terrain, we perform a stereo-error adaptive local obstacle and terrain classification directly on the depth images computed via stereo matching. This allows us to take viewpoint and view distance-dependent information into account, which is lost later on due to the aggregation of the dense 3D data. We present more information on this component in Section 5.1.1. We aggregate the obstacle classification results together with dense depth data along the trajectories estimated by the local reference filter, thereby creating so-called submaps of limited size and uncertainty. These local aggregations constitute a compacted representation of high-bandwidth 3D information, as we describe in detail in Section 5.1.2 and 5.1.3.

We split our algorithms for global multi-robot estimation into four major components, visualized by the four blocks in the right part of Figure 3.1: global pose optimization, global mapping, map matching, and robot detection. First, we construct a multi-robot SLAM graph for global optimization that integrates the low-frequency estimates of the other localization components, as we describe in detail in Section 4.2. We apply incremental methods for online optimization of the 6D poses of all submaps and are thus able to compute an up-to-date global 3D map and to estimate the 6D poses for all participating robots online and on board each robot. Second, we exchange submaps between robots in a multi-robot team, as described in Section 5.2.1. Containing aggregated and thus compacted dense probabilistic 3D information, they represent a suitable unit for the transfer over low-bandwidth wireless network links. The set of submaps created in a distributed fashion by all robots partitions the full multi-robot map. Based on the latest global estimates obtained from the SLAM graph optimization, we compose these partial maps to an up-to-date joint 3D model of the environment, as discussed in Section 5.2.2. Third, we match pairs of submaps created by the same or by different robots. This is a two-step process that starts by finding promising pairs of maps to match in order to exclude false hypotheses early on and keep the matching process computationally feasible, as we will discuss in Section 6.1.1. We then compute an estimate for the relative transformation between the two submaps in each pair, as presented in Section 6.1.2. The resulting transformation serves as an intra- or inter-robot loop closure constraint in our SLAM graph for global optimization. Fourth, we use the camera images directly to generate further inter-robot loop closures by detecting other robots based on planar AprilTag markers and use these detections to estimate their relative 6D poses (Olson, 2011). We describe this process in Section 6.2, with emphasis on our novel estimation of the measurement uncertainties. These are crucial in order to exclude potential outliers and weight all measurements according to their assumed accuracy during global optimization.

3.3 Distributed Computation and Data Exchange

All localization and mapping components, as shown in Figure 3.1, are executed on board of all the robots within a multi-robot team in a decentralized or distributed fashion. This architecture has several benefits compared to a centralized approach and thus allows us to tackle the non-functional challenges discussed in Section 2.1.

First, it ensures the online availability of an up-to-date pose estimate and map on each robot at all times, increasing the robustness in particular in the light of possible communication losses or failures of individual robots within a multi-robot team. Pose estimates and aggregated submap information are exchanged between robots whenever communication links are established. Each robot then computes its own maximum likelihood estimate of all robots' poses and of the joint map given its available measurement data. As we do not assume any initial knowledge about the robots' starting positions, we do not introduce any shared global coordinate frame. Nevertheless, each robot estimates the poses of all other robots in its team and includes their submap data within its own model of the environment as soon as connections between the robots can be made in the SLAM graph. These can, for example, be created through the matching of submaps, through visual robot detections, or through the identification of the same globally identifiable landmark.

Second, our architecture distributes computationally challenging tasks among the individual robots. All high-frequency measurements, like IMU data, and high-bandwidth sensor data, like camera images, are processed and aggregated locally on each robot. Besides local state estimation and map creation, we also distribute the task of pairwise map matching between the participating robots, checking each pair only once. Thus, solely the tasks of global graph optimization and global map composition are executed repeatedly – it would be possible to perform them only once in a centralized setup. However, we chose to compute them in a decentralized fashion on each robot separately to achieve the aforementioned robustness w. r. t. communication losses. As our SLAM graph is constructed from low-frequency estimates, it is typically small in size and thus cheap and fast to optimize, not introducing any relevant computational overhead. While the composition of a global map is an expensive operation, it could be limited in future work to on-demand processing of regions of interest. The decision whether to compute simultaneously on several robots or to query certain robots for it comes down to a trade-off between computational effort and communication bandwidth that might need to be re-evaluated dependent on the application and target hardware.

Third, due to the local aggregation of data on the individual robots, they are able to communicate over low-bandwidth channels compared to sharing the raw data, as would be required in a simple centralized mapping approach. In our setup, the robots exchange the following information:

- *Robot Pose Estimates*: local pose estimates of the robots
- *Submap Matches*, *Robot Detections*, and *Global Landmark Detections*: loop closure constraints for the SLAM graph; in addition, information about submap match attempts to ensure that each pair is attempted to match at most once
- *3D Submaps*: poses of the submaps' origins and their aggregated dense 3D data, filtered and compacted during aggregation

We provide a short comparison of the bandwidth required for the raw data and the compacted submap data for one of our experiments in Section 8.4.

Furthermore, the modularity of the localization and mapping system gives us the option to run only some of its components on resource-constrained systems like, for example, MAVs. In case the creation of a 3D map might not be feasible due to the sensor setup or limited computational power, such robots can still run the filter for local and the graph optimization for global pose estimation. By exchanging measurement data like robot detections and filter estimates, they can still both contribute to and benefit from the joint localization with other robots in a heterogeneous team.

3.4 Related Work and Discussion

Our multi-robot localization and mapping architecture is the result of many design decisions that were driven by the challenges of autonomous planetary exploration identified in Section 2.1. Most of them involve trade-offs between different benefits and system characteristics. In the following paragraphs, we discuss our choices in comparison to similar robotic single- and multi-agent systems.

Like Hidalgo-Carrió et al. (2018) and Wettergreen et al. (2008), we solely consider sensors as input to our SLAM system that are similar to those available on real space exploration robots operating on Mars (Maimone et al., 2006; Grotzinger et al., 2012). Thus, we focus our research on stereo camera-based localization and mapping methods. In contrast, many other space rover prototypes built for terrestrial tests employ either rotating LIDARs (Schwarz et al., 2016; Schwendner et al., 2014b), active RGBD cameras (Avsar et al., 2014; Sünderhauf et al., 2014), or a combination of both for 3D mapping and localization. While LIDARs provide high-precision measurements and active RGBD cameras gather dense data on untextured surfaces, the former are – at least until now – not suitable for mobile robots sent to space, whereas the latter face issues in bright sunlight and thus typically are constraint to indoor usage. Our focus on stereo vision-based systems, however, leads to additional challenges as we need to deal with high sensor noise and, compared to LIDARs, typically smaller fields of view. Despite these challenges, stereo camera setups provide dense depth data that is required for many crucial tasks such as terrain classification, navigation, exploration, modeling of the robots' surroundings for visual inspection, as well as for future automatic segmentation and semantic annotation.

Similar to Reid and Bräunl (2011), Olson et al. (2013), Schwendner et al. (2014a,b), and Schwarz et al. (2016), we aggregate data first into local maps that are then used to compose a global map. In contrast to Schwarz et al. (2016), we use local maps directly for path planning, i. e., obstacle avoidance, as they can be updated at a higher frequency. Thereby we can keep safety-critical functions, such as obstacle avoidance, independent of delays, noise, or potential errors in a global map. The latter is used for global planning tasks, such as exploration goal selection, which are sufficient to run at lower frequencies. For an in-depth discussion of our combination of local filter and global graph optimization methods, we refer to Chapter 4. While teleoperation or shared autonomy approaches allow to transfer some computation off-board the robots to a ground control center, we aim for fully autonomous systems. Our hierarchical approach to localization and mapping enables online and on-board computation for local robot autonomy even on resource-constrained systems such as MAVs, which we demonstrate with a heterogeneous multi-robot team in Section 9.4.

Centralized multi-robot SLAM approaches, such as the 2D multi-robot mapping frameworks presented by Olson et al. (2013) and Nagatani et al. (2011), have a lower complexity than distributed systems and oftentimes can harness the power of an off-board ground control station for computationally expensive tasks such as map alignment and merging. For our planetary exploration application scenario, however, a decentralized approach is required in order to enable local autonomy for each robot. It allows the system to work in the light of communication outages to and between the robots as well as to deal with potential failures of individual robots. Our multi-robot SLAM framework is decentralized and in large parts distributed. In this regard, our architecture is similar to the system developed for 2D mapping by Reid and Bräunl (2011). SLAM graph creation and optimization as well as global map composition are the only non-distributed components. In our framework, graph optimization is computationally lightweight as our combination of local and global estimation leads to small and sparse graphs, as we discuss in more detail in Chapter 4. In contrast, global map composition is an expensive operation. However, distributing it would require to transfer additional map data, i. e., already merged sets of submaps representing parts of a globally aligned map. Instead, we choose to merge the submaps on each robot, trading the higher communication bandwidth requirements as well as the increased system complexity of a fully distributed approach against the additional computational effort.

So far, we do not enforce map consistency in our decentralized estimation. Thus, in case of incomplete communication, the robots' pose and map estimates can differ depending on the respective measurements each robot has available. Methods to enforce consistency on encounters in a distributed system would need a master robot (e. g., dynamically determined via a leader selection algorithm) or a static master node (e. g., the central ground station proposed by Reid and Bräunl (2011)) that can correct the results in case of significant deviations.

While our decentralized and distributed architecture does not require any central node, in space exploration, static infrastructure such as a lander on the planetary surface or a ground control center on Earth could, at least temporarily, provide additional computation power. In a heterogeneous multi-robot setup, a larger rover could also fill that role for its smaller teammates at certain times, e. g., whenever sufficient energy and communication bandwidth are available. Our modular system architecture would allow to easily add further computation nodes to support the computationally expensive map matching operations, which could also run off-board or even off-site. As submaps are aggregated 3D representations, the communication bandwidth to outsource these computations would be significantly lower than what would be required to transfer raw sensor data. Throughout this thesis, we will further discuss the design choices and trade-offs regarding the individual methods that form our SLAM framework in their respective chapters.

Chapter four

Combination of Local and Global Estimation

We combine local and global estimation methods for multi-robot SLAM to get the best of both worlds: We locally fuse high-frequency measurements on each robot to compute real-time local state estimates required for robot control and fast reactive behaviors such as obstacle avoidance. Slower but online global optimization provides pose and map estimates as input to higher-level autonomy functions such as path and exploration planning as well as multi-robot coordination. Unobservable states in local estimation, such as a robot's position or heading angle, are subject to potentially unbounded drift. In contrast, global estimation can correct these accumulated errors through optimization on loop closures in order to gain consistent estimates for longer trajectories. This includes the combined estimation of the poses of all robots in a multi-robot system and the collaborative generation of a global map. In multi-robot teams, our loose coupling of local and global methods allows the distributed local processing of high-frequency measurements as well as the decentralized online computation of global pose and map estimates on all robots.

First, in Section 4.1, we summarize the concept of local reference filters, an existing method for local state estimation that constitutes the basis for our work. We then present our approach for multi-robot global pose estimation in Section 4.2. Therein, we combine local filter estimates of multiple robots through a novel SLAM graph topology. We close the chapter with a discussion of our method in the context of related work in Section 4.3.

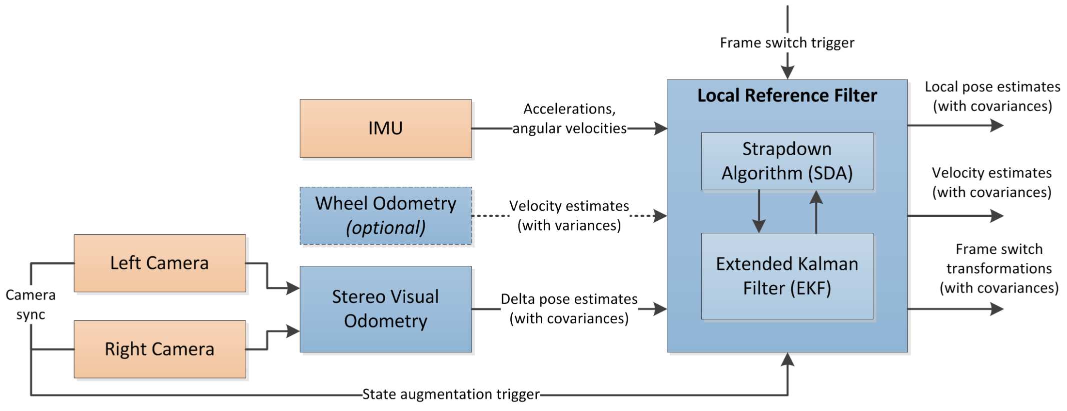

Frame switch trigger
Local Reference Filter
IMU
Accelerations, angular velocities
Strapdown Algorithm (SDA)
Wheel Odometry (optional)
Velocity estimates (with variances)
Extended Kalman Filter (EKF)
Left Camera
Right Camera
Camera sync
Stereo Visual Odometry
Delta pose estimates (with covariances)
State augmentation trigger
Local pose estimates (with covariances)
Velocity estimates (with covariances)
Frame switch transformations (with covariances)

Figure 4.1: *Schematic of the Vision-Aided Inertial Navigation System (VINS) for locale state estimation. It is based on an Extended Kalman Filter (EKF) that fuses the inputs from visual odometry, an IMU, and, where available, wheel odometry.*

4.1 Local Reference Filter

In this section, we first give a short overview of the local reference filter, a Vision-Aided Inertial Navigation System (VINS) that we employ for local state estimation. We then focus on its ability to switch its local frame of reference, a feature that is relevant for our overall system architecture as well as for the integration of the filter with our global graph-based optimization.

4.1.1 Vision-Based Keyframe Inertial Navigation

The local reference filter is a VINS developed by Schmid et al. (2014b). It runs on each robot to fuse high-frequency measurements for real-time local state estimation. An implementation as an *Extended Kalman Filter (EKF)* allows it to meet hard real-time constraints imposed by robot control algorithms.[1] We show a block diagram of the local reference filter with its inputs and outputs in Figure 4.1. As indicated in our system architecture in Chapter 3, we use the filter to fuse the following types of measurement data:

- High-frequency three-axis accelerometer and gyroscope measurements from an Inertial Measurement Unit (IMU)

[1]See also Section 2.2.3 for a short general discussion of EKFs.

- Delta position and orientation estimates from a keyframe-based stereo odometry system (Hirschmüller et al., 2002)
- Wheel odometry estimates represented as linear x, y velocities in the robot's body frame (only available on some of our rovers, see Bussmann et al., 2018)

The VINS uses a Strap-Down Algorithm (SDA) to integrate high-frequency acceleration and angular rate measurements from the IMU into the direct system state $\boldsymbol{x}$ (Schmid et al., 2014a). It includes the position $\boldsymbol{p}$, velocity $\boldsymbol{v}$, and orientation quaternion $\boldsymbol{q}$ of the IMU frame relative to the filter's current local navigation frame, as well as the biases of the IMU's accelerometers $\boldsymbol{b}_a$ and gyroscopes $\boldsymbol{b}_\omega$:

$$\boldsymbol{x} = \left(\boldsymbol{p}^T,\ \boldsymbol{v}^T,\ \boldsymbol{q}^T,\ \boldsymbol{b}_a^T,\ \boldsymbol{b}_\omega^T\right)^T \in \mathbb{R}^{16} \tag{4.1}$$

As the used EKF is implemented as an error-state filter, it estimates the errors of the direct state in the indirect state $\boldsymbol{\delta}$. Assuming small angular errors between filter updates, the attitude errors in the indirect state can be minimally parameterized as a three-dimensional orientation vector $\delta\boldsymbol{\sigma}^T$. The indirect state is defined as:

$$\boldsymbol{\delta} = \left(\delta\boldsymbol{p}^T,\ \delta\boldsymbol{v}^T,\ \delta\boldsymbol{\sigma}^T,\ \delta\boldsymbol{b}_a^T,\ \delta\boldsymbol{b}_\omega^T\right)^T \in \mathbb{R}^{15} \tag{4.2}$$

In order to estimate the error uncertainties, the local reference filter employs a linearized, continuous-time error transition model within its EKF prediction step, as described by Schmid et al. (2014a). The processing of delta pose estimates from visual odometry, the time-delay compensation for delayed measurements, as well as the switching of the filter's frame of reference require state cloning and marginalization. Both steps can be included into the prediction equation of a Kalman Filter (KF). State augmentation is the general form of state cloning and can be written by introducing a state augmentation function that builds the new state vector $\bar{\boldsymbol{x}}_t$ at time step t as

$$\bar{\boldsymbol{x}}_t = \left(\boldsymbol{x}_t,\ \boldsymbol{x}_{aug}\right)^T \tag{4.3}$$

The new state consists of the current state $\boldsymbol{x}_t$ and the newly augmented state $\boldsymbol{x}_{aug}$, which in our case is a clone of the latest pose of the robot that is estimated as part of the filter's main state.

The filter needs to integrate delta poses estimated by the keyframe-based visual odometry. These delta poses refer to relative transformations between a past keyframe and the latest camera image. They are accompanied by uncertainty estimates stemming from propagated measurement noise. The visual odometry algorithm internally keeps a history of $n = 5$ keyframes to improve precision and allow for locally drift-free estimation. This is based on the assumption that in a small area, features stay visible for some time (Schmid et al., 2012). Each time a new pair of stereo images is taken, a state augmentation trigger is sent to the local reference filter, as indicated in Figure 4.1. On receiving the trigger, the filter adds the robot pose as an augmentation to the system's state at the exact time of image capture. The filter then can later on process the delta pose estimates received from the visual odometry module by referencing the corresponding augmented states. Using these state augmentations, it can also compensate for measurement delays as introduced by typical vision pipelines. These are, for example, approx. 250 ms for the FPGA-based implementation of Semi-Global Matching (SGM) followed by the computation of visual odometry estimates on the aerial and ground-based robots that we used in our experiments.[2] This delay compensation enables the filter to always provide up-to-date state estimates at up to IMU frequency for robot control, instead of having to wait for the slowest measurement to arrive. We refer to the work by Schmid et al. (2012, 2014a,b) for more details on the filter internals and focus on switching the filter's frame of reference in the next section.

4.1.2 Frame of Reference Switching

The local reference filter is able to switch its frame of reference on an external trigger at arbitrary points in time. In this section, we motivate regular switches into the robot's current pose, describe their effects and discuss the benefits for the filter's consistency as well as for the overall SLAM system. For an introduction of the frame switching formulated via state augmentation and marginalization, we refer to the work by Schmid et al. (2014b). We provide further details on switching the frame of reference into the robot's current pose in our journal article by Schuster et al. (2018).

It has been shown that some state variables of a VINS, i. e., the position (x, y, z) and the *yaw* angle around the gravity vector, are unobservable (Weiss, 2012). In contrast to observable states, their errors and the corresponding uncertainty estimates can rise unbounded. This can lead to numerical issues, to larger linearization errors, as well as to the violation of small-angle approximations in an EKF-based VINS. Filter-based global SLAM systems have been shown to be inconsistent. Bailey et al. (2006) in particular relate significant inconsistencies to rising errors of the *yaw* (heading) angle of a vehicle. These findings provide a motivation to split state estimation by observability: We employ the local reference filter for a long-term estimation

[2]See Chapter 7 for our robot hardware platforms and their common system architecture.

of only the observable modes, as convergence to their real values can be expected. Unobservable modes, on the contrary, are only locally estimated within the filter and then globally optimized in our SLAM graph, as described in Section 4.2.

In order to achieve this split, we regularly switch the filter's frame of reference. A frame of reference is defined by its 3D position, its *yaw* angle and the respective uncertainties relative to the 6D pose of the robot. A frame switch in the filter transforms all state variables into the respective new frame of reference. This, by definition, resets the estimated uncertainties of the position and the *yaw* angle to zero. All other state variables and their uncertainty estimates, i. e., the IMU biases, velocities and augmented keyframe states, are kept or transformed into the new frame of reference accordingly. We refer to the work by Schmid et al. (2014b) for details on the realization of the frame switching within the filter. The authors demonstrated a long-term stable, real-time state estimation using the local reference filter for unobservable systems in simulated and real-world experiments. By periodically switching the local frame of reference, the globally unbounded uncertainty for unobservable states became locally bounded.

In the context of our SLAM framework, we always define the local reference to coincide with the origin of a submap in our mapping components, as sketched in Figure 2.4. Each switch of the frame of reference in the filter is actively triggered by the mapping system based on map uncertainty and size considerations, as we describe in Section 5.1.3. Whenever the mapping system creates a new submap, it triggers the filter to switch its frame of reference into the non-observable part of its current estimate for the robot pose, i. e., its (x, y, z) position and *yaw* angle estimate. The *roll* and *pitch* angles are observable within the filter due to the accelerometer measurements of the IMU. They are thus excluded from the switch, which keeps each local navigation frame as well as all submap origins aligned to the gravity vector.

In the following paragraphs, we summarize the benefits from these periodical switches of the filter's frame of reference:

- *Filter Consistency*
  The local estimation leads to a limitation of linearization errors in the filter that otherwise could rise potentially unbounded for unobservable states. Furthermore, unbounded error states could lead to violations of small-angle approximations used in the EKF. Global estimates of the robot position are then estimated during the graph optimization process, see Section 4.2, which can employ re-linearization to improve global consistency.

- *Numerical Stability*
  Frame switches reset otherwise unbounded covariance estimates for unobserv-

able states in the filter. This ensures long-term numerical stability of the filter implementation.

- *Integration with SLAM System*
  The local reference frames of the filter define the local frames for our submapping system. This leads to a clean separation of local and global estimation for both localization and mapping. Furthermore, switching the filter's frame of reference allows for a more accurate integration of the filter output into the SLAM graph according to its uncertainty estimates, as we discuss in Section 4.2.2.

- *Submap Matching*
  The aforementioned partial frame switches ensure all submap origins to be aligned to the gravity vector. We can exploit this property during submap matching to reduce the dimensionality of the matching problem from six to four degrees of freedom, as presented in Section 6.1.

4.2 Multi-Robot Graph SLAM Connecting Decoupled Filters

In this section, we present our methods to combine the estimates of the local reference filters on each robot into a multi-robot SLAM graph for global optimization, which we first described in our conference and journal publications by Schuster et al. (2015, 2018). We will start by introducing a formalization of graph optimization in the context of robot pose estimation. We will then characterize the types of measurement constraints relevant for our SLAM system, present our contributions to the multi-robot SLAM graph construction, and describe the optimization back-end.

4.2.1 Graph Optimization for 6D SLAM

In Section 2.2.3, we gave a brief general introduction to graph optimization approaches. In this section, we formalize the SLAM problem for 6D pose graph optimization in a graphical model. For this, we employ factor graphs (Kschischang et al., 2001), following the formalization used by Kaess et al. (2012) and Dellaert and Kaess (2017). A factor graph is a bipartite graph $G = (\mathcal{F}, \Theta, \mathcal{E})$ with factor nodes $\boldsymbol{f}_i \in \mathcal{F}$, variable nodes $\boldsymbol{\theta}_i \in \Theta$, and edges $\boldsymbol{e}_{i,j} \in \mathcal{E}$ representing dependencies as undirected connections between nodes of the two different types. Such a graph defines the factorization of a function over the variable nodes $f(\Theta) = \prod_i f_i(\Theta_i)$ with Θ_i denoting the set of variables adjacent to the factor f_i. In our SLAM context, the variable nodes $\boldsymbol{\theta}_i$ represent the 6D poses of interest (robot poses, submap origins, landmarks). The factors f_i represent constraints given by measurements or "virtual measurements", i. e., estimates from other modules like our local reference filter.

The goal of graph optimization is to find an assignment of variables Θ^* that maximizes this function, i. e., $\Theta^* = \arg\max_\Theta f(\Theta)$. Assuming a measurement model with zero-mean Gaussian noise, as is standard in SLAM literature, the factors can be seen as cost functions on the variables Θ_i with

$$f_i(\Theta_i) \propto \exp\left(-\frac{1}{2}||\boldsymbol{h}_i(\Theta_i) - \boldsymbol{z}_i||^2_{\Sigma_i}\right) \tag{4.4}$$

Hereby, the measurement functions are denoted $\boldsymbol{h}_i$, the measurements $\boldsymbol{z}_i$, and their measurement noise covariance matrices Σ_i. We use $||\boldsymbol{e}_i||^2_{\Sigma_i} \triangleq \boldsymbol{e}_i^T \Sigma_i^{-1} \boldsymbol{e}_i$ as a notation for the squared Mahalanobis distance with covariance matrix Σ_i. The maximization problem can thus be formulated as a non-linear least squares minimization problem:

$$\arg\min_\Theta(-\log f(\Theta)) = \arg\min_\Theta \frac{1}{2}\sum_i ||\boldsymbol{h}_i(\Theta_i) - \boldsymbol{z}_i||^2_{\Sigma_i} \tag{4.5}$$

For SLAM problems, e. g., when considering 6D pose compositions, the measurement functions $\boldsymbol{h}_i$ are mostly non-linear. Optimization methods thus typically approach the minimum by solving iterations of linear approximations. In general, factors f_i can connect an arbitrary number of variable nodes $\boldsymbol{\theta}_i$. In this thesis, we only refer to binary measurement factors, except for a single unary prior factor per graph.

As discussed in Section 2.2 for SLAM systems in general, graph SLAM approaches can be structured into two parts, a front-end and a back-end. While the back-end deals with the optimization problem, i. e., the aforementioned minimization of a non-linear quadratic error function, the front-end is concerned with the construction of the graph. This includes solving the data association problem as well as asserting dependencies between variable nodes by deciding on a graph topology (Grisetti et al., 2010). In the following sections, we will characterize the measurement constraints relevant for our SLAM system, present our contributions to the multi-robot SLAM graph construction and describe the optimization back-end.

4.2.2 SLAM Front-End: Multi-Robot Graph Topology

We consider a multi-robot setup with R robots and include the following six-dimensional variable nodes in the SLAM graph:

- *Robot Poses* $\{\boldsymbol{x}_i^r\}$
  $\boldsymbol{x}_i^r$ represents the pose of robot $r \in \{0, \ldots, R-1\}$ at time t_i. We sample the robot poses sparsely by only including them once they are connected to another robot's pose or a landmark via a measurement edge.

- *Submap Poses* $\{\boldsymbol{s}_i^r\}$
  $\boldsymbol{s}_i^r$ represents the pose of the origin of the i^{th} submap of robot $r \in \{0, \ldots, R-1\}$. It is equal to the corresponding local navigation frame of robot r in its local reference filter.

- *Landmark Poses* $\{\boldsymbol{l}_i\}$ *(optional)*
  $\boldsymbol{l}_i$ represents the pose of the i^{th} landmark. All of these landmarks are assumed to be static, robot-independent, and globally identifiable.

Each of them represents a 6D pose $(x, y, z, roll, pitch, yaw) \in \mathbb{R}^6$ with Gaussian uncertainty. On each robot r that runs a graph optimization module, its first submap pose $\boldsymbol{s}_0^r$ is connected to an unary prior factor that defines its map origin, which we arbitrarily chose to be located at zero. We decided against an explicit representation of visual odometry keyframes in the SLAM graph as a design trade-off to ensure a limited growth rate of the graph's size while allowing our local reference filter to internally use arbitrary techniques to integrate such high-frequency measurements. This allows for fast optimization steps on loop closures in the graph. We can always compute an online pose estimate $\tilde{\boldsymbol{x}}_t^r$ for each robot r w. r. t. the global map origin at the latest filter timestep t by combining the output of the local reference filter and our SLAM graph.[3] This simply means chaining the pose of the respective latest submap origin $\boldsymbol{s}_j^r$ (time of submap creation: t_j^r), as estimated through graph optimization, with the latest filter estimate of the robot's pose $\tilde{\boldsymbol{v}}_t^r$ at time t:

$$\tilde{\boldsymbol{x}}_t^r = \boldsymbol{s}_j^r \oplus \tilde{\boldsymbol{v}}_t^r \quad \text{with} \quad t_j^r \le t < t_{j+1}^r \tag{4.6}$$

[3]Note that, in contrast to the description of the full VINS state in Section 4.1.1, in this and the following sections, $\boldsymbol{x}$ and $\boldsymbol{v}$ refer to the global robot poses in the SLAM graph and the local robot pose estimates of the filter respectively.

Integration of Observations

We represent three different types of observations as factors connecting the variable nodes in the SLAM graph:

- *Robot Detections* $\{\boldsymbol{d}_i\}$
  $\boldsymbol{d}_i$ represents the transformation between the poses of two different robots r_0 and r_1. These can be determined by visually detecting r_0 from r_1 (or vice versa) and estimating its 6D pose. In our experiments, we attached planar visual markers to the robots in our multi-robot team and detect them in the other robots' camera images. Having detected the marker, we can estimate the 6D transformation between the camera and center of the marker. The quality of these estimates, however, highly varies, in particular depending on view distance and view angle. We therefore perform a worst-case error approximation depending on distance, view angle and camera parameters that we describe in detail in Section 6.2. This allows us to avoid overconfidence in measurements that could degrade the results of graph optimization. Assuming the transformations $\boldsymbol{T}_c^{r_0}$ between r_0 and its camera as well as $\boldsymbol{T}_m^{r_1}$ between r_1 and its marker to be known through calibration, we can simply chain these with the detector's estimate $\hat{\boldsymbol{T}}_m^c$ to compute the relative transformation $\boldsymbol{d}_i$:

  $$\boldsymbol{d}_i = \hat{\boldsymbol{T}}_{r_1}^{r_0} = \boldsymbol{T}_c^{r_0} \oplus \hat{\boldsymbol{T}}_m^c \oplus \left(\boldsymbol{T}_m^{r_1}\right)^{-1} \tag{4.7}$$

  In order to add $\boldsymbol{d}_i$ as well as its adjacent nodes $\boldsymbol{x}_i^{r_0}$ and $\boldsymbol{x}_j^{r_1}$ to the graph, we require pose estimates from the filters of both robots at the point in time $t_i = t_j$ of the detection. Each robot therefore buffers its filter estimates in order to recover this information in case of communication delays or interruptions. The memory requirements of this buffer, implemented as a hashmap for constant-time lookup, are negligible compared to the dense 3D map data.

- *Landmark Observations* $\{\boldsymbol{o}_i^r\}$
  $\boldsymbol{o}_i^r$ represents the transformation between a robot r and a robot-independent static and globally identifiable landmark. Each landmark defines an external global coordinate frame, for example represented via an artificial visual marker similar to the ones used for robot detections. In space scenarios, static infrastructure elements, such as a stationary lander (Wedler et al., 2017), can be used as landmarks in order to improve localization and provide transformations between multiple robots. However, during the exploration of previously unknown natural environments, such artificial landmarks are typically not available. We therefore did not utilize any such static landmarks in any of our multi-robot

exploration and mapping experiments presented in this thesis.

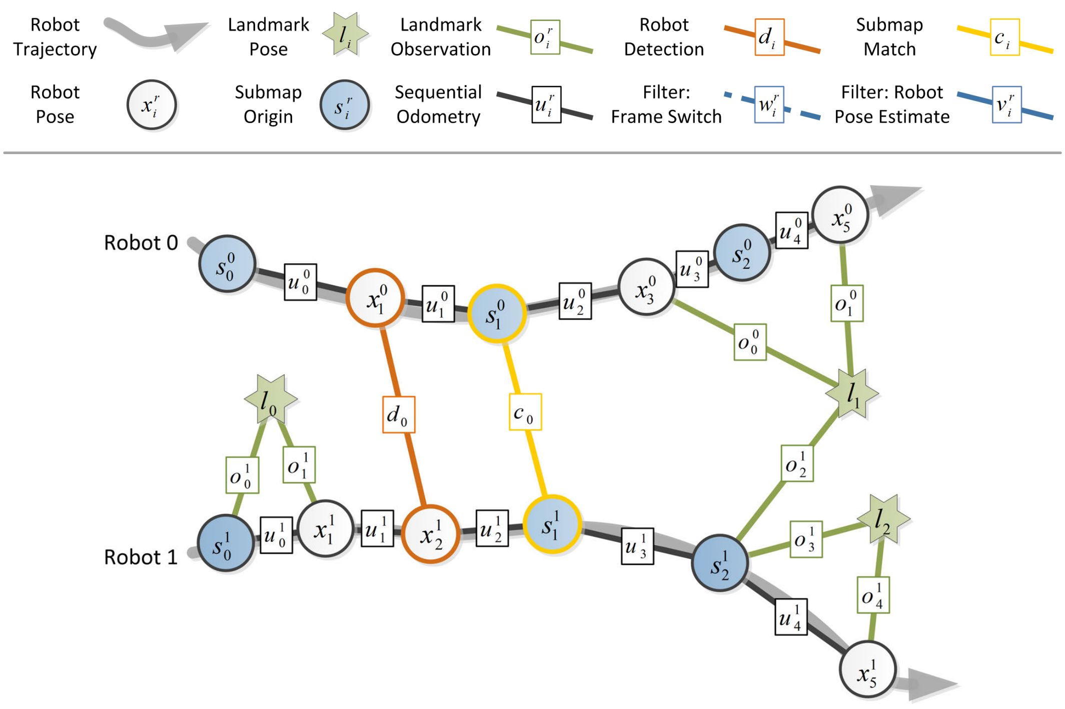
- *Submap Matches* $\{\boldsymbol{c}_i\}$
  $\boldsymbol{c}_i$ represents the transformation between the origins of two submaps, which is the result of our submap matching process, described in detail in Section 6.1. Submap matches can be computed between maps created by a single or by multiple different robots and thus serve as intra- as well as inter-robot loop closure constraints.

All these measured and estimated transformations are added to the graph as binary factors representing six-dimensional constraints that connect exactly two nodes each. They all are accompanied by a Gaussian uncertainty estimate. It would be straightforward to add further types of measurements: Global position information from GNSS or the matching of aerial images (Kümmerle et al., 2011b) could, for example, be represented by additional unary factors. While we are able to restrict the submap matching itself to 4D, the graph SLAM still needs to work on 6D poses, as some types of observations like landmark or robot detections are measured in 6D. Furthermore, we intentionally do not introduce hard constraints on the *roll* and *pitch* angles of the submap origins but integrate them with the, typically low, variance estimated by our local reference filter. This allows the graph optimization to compensate for errors in this estimation w. r. t. other types of measurements.

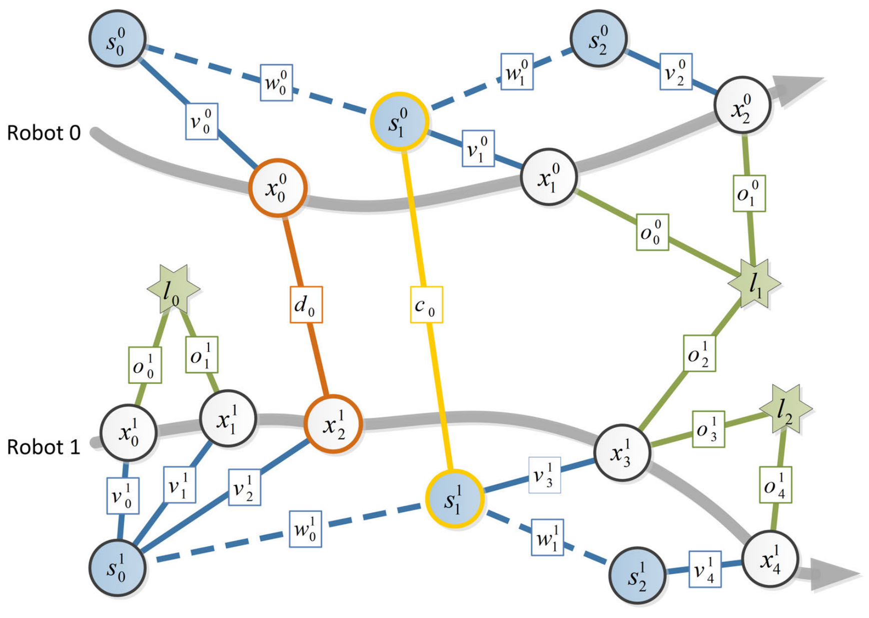
In addition to the observations discussed above, we integrate the 6D pose estimates of our local reference filters to connect submap and robot poses as well as subsequent submaps. The adequate integration of filter estimates into a SLAM graph requires an adaption of the graph topology that differs from the one typically used for sequential odometry measurements. In the following sections, we present and discuss these changes in a comparison of two different graph topologies that we outline in Figure 4.2.

Graph with Sequential Odometry Measurements

The graph topology typically found in SLAM literature sequentially connects the robot poses through odometry-like measurements $\boldsymbol{u}_i^r$ between $\boldsymbol{x}_i^r$ and $\boldsymbol{x}_{i+1}^r$, as shown in Figure 4.2(a). The set of submap origins thus is a subset of the set of robot poses. This graph topology builds on the assumption that the incremental robot ego motion estimates are independent from each other and from any prior states. For most pure odometry measurements like wheel odometry, simple visual odometry without keyframes in 2D images or 3D data through sequential scan matching, this assumption constitutes a reasonable approximation. Dependencies to prior estimates


Robot Trajectory
Landmark Pose
l_i
Landmark Observation
o_i^r
Robot Detection
d_i
Submap Match
c_i
Robot Pose
x_i^r
Submap Origin
s_i^r
Sequential Odometry
u_i^r
Filter: Frame Switch
w_i^r
Filter: Robot Pose Estimate
v_i^r
Robot 0
Robot 1
s_0^0 u_0^0 x_1^0 u_1^0 s_1^0 u_2^0 x_3^0 u_3^0 s_2^0 u_4^0 x_5^0
o_1^0 o_0^0 l_1 d_0 c_0 l_0
o_0^1 o_1^1 o_2^1
s_0^1 u_0^1 x_1^1 u_1^1 x_2^1 u_2^1 s_1^1 u_3^1 s_2^1 o_3^1 l_2 o_4^1 u_4^1 x_5^1

(a) *Graph topology for sequential odometry measurements with submap origins at certain robot poses*


Robot 0
Robot 1
s_0^0 w_0^0 v_0^0 x_0^0 s_1^0 w_1^0 s_2^0 v_2^0 x_2^0 v_1^0 x_1^0
o_1^0 o_0^0 l_1 d_0 c_0 l_0
o_0^1 o_1^1 o_2^1
x_0^1 x_1^1 x_2^1 x_3^1 o_3^1 l_2 o_4^1
v_0^1 v_1^1 v_2^1 w_0^1 s_0^1 s_1^1 v_3^1 w_1^1 s_2^1 v_4^1 x_4^1

(b) *Novel graph topology to connect local reference filter estimates with submap origins $\mathbf{s}_i^r$ (local reference frames) separated from robot poses $\mathbf{x}_i^r$*

Figure 4.2: *Comparison of SLAM graph topologies with robot detections $\mathbf{d}_i$, submap matches $\mathbf{c}_i$, and landmarks observations $\mathbf{o}_i^r$ as inter-robot loop closure constraints*

exist only indirectly through the robot's environment, for example, by repeatedly observing parts of the same scene in case of visual odometry, and are thus hard to quantify. In contrast, this assumption is violated when integrating estimates of a keyframe-based visual odometry and filter, as we do in our local reference filter. The filter estimates can depend on each other through filter-internal states like the augmentations made on keyframes. In our early work (Brand et al., 2015), we ignored these dependencies by approximating sequential odometry measurements through the computation of delta poses from subsequent filter estimates of the robots' poses as

$$\boldsymbol{u}_i^r = \boldsymbol{x}_{i+1}^r \ominus \boldsymbol{x}_i^r \tag{4.8}$$

and used an approximation for their Gaussian measurement uncertainty as

$$\Sigma_{\boldsymbol{u}_i^r} = \max\left(\Sigma_{\boldsymbol{x}_{i+1}^r} - \Sigma_{\boldsymbol{x}_i^r},\ \boldsymbol{I}_6 \cdot 10^{-10}\right) \tag{4.9}$$

with $\boldsymbol{I}_6$ denoting the 6×6 identity square matrix. Thereby, we enforce the resulting covariance matrices $\Sigma_{\boldsymbol{u}_i^r}$ to be non-negative and above an, experimentally determined, threshold to ensure numerical stability during graph optimization. This rough approximation, however, neglects the aforementioned state dependencies. In the following paragraphs, we therefore introduce an adaption of the graph topology that allows a more suitable integration of local reference filter estimates.

Graph with Local Reference Filter Estimates

In Figure 4.2(b), we sketched our novel graph topology, which we first presented in our conference paper by Schuster et al. (2015). When constructing the graph, we replace the aforementioned approximation of sequential odometry measurements $\boldsymbol{u}_i^r$ with two types of estimates that are directly computed by our local reference filter:

- *Frame Switch Transformations* $\{\boldsymbol{w}_i^r\}$
  $\boldsymbol{w}_i^r$ represents the transformation between the poses of two consecutive submap origins $\boldsymbol{s}_i^r$ and $\boldsymbol{s}_{i+1}^r$. It refers to a switch of the frame of reference in our local reference filter.

- *Robot Pose Estimates* $\{\boldsymbol{v}_i^r\}$
  $\boldsymbol{v}_i^r$ represents the transformation between a submap origin $\boldsymbol{s}_j^r$ and a robot pose $\boldsymbol{x}_i^r$ that is estimated by the filter w. r. t. the local reference frame anchored in $\boldsymbol{s}_j^r$.

The local reference frames and hence the 6D submap origins $\boldsymbol{s}_i^r$ are aligned w. r. t. to the gravity vector, as we discussed in Section 4.1. Thus, with this graph topology, they can be dissimilar from the robot poses $\boldsymbol{x}_i^r$ w. r. t. to the *roll* and *pitch* angles, as we indicated in Figure 4.2(b) by drawing their nodes separated. Compared to the graph topology for sequential odometry measurements, as presented in the previous section, the direct integration of the estimates computed by the local reference filter allows a better representation of the underlying probabilistic structure. Within a local reference frame ($\widehat{=}$ submap), we thus do not introduce any additional independence assumptions. Furthermore, pose estimates and delayed measurements for each robot can be added at any point in time without requiring additional methods to remove constraints from the graph or to avoid double-counting of information, such as anti-factors (Cunningham et al., 2013). Our graph topology thus allows a straightforward inclusion of delayed measurements like those received from other robots in case of delayed or interrupted communication.

Integrating the frame switch transformations $\boldsymbol{w}_i^r$ into the graph, we assume independence between the subsequent submap origins $\boldsymbol{s}_i^r$ and $\boldsymbol{s}_{i+1}^r$. This is an approximation, as during frame switches, several filter-internal states (e. g., velocities, IMU biases, and visual odometry keyframes) are transferred across submaps. Correlations between submaps could be explicitly considered in the graph optimization by, for example, creating conditionally independent local maps as described by Piniés and Tardós (2008). This, however, would require additional nodes to be added to the graph that represent all common states between submaps. These nodes thus would expose filter-internal and robot-specific states, such as velocities, IMU biases, and visual odometry keyframe augmentations, to the graph SLAM. In the design of our system, we instead focus on an explicit decoupling of the local filter and global graph optimization components in favor of a modular multi-robot system architecture. Based on observability considerations, we define the interface between filter and graph on and between all robots at a pure pose level. Therefore, in our approach, the internal states of each individual robot are not exposed and do not need to be transferred to other robots. A change in sensing modalities on one system thus does not require any changes at the SLAM graph level on any of the robots. This is particularly important in heterogeneous multi-robot setups, in which different robots integrate different types of high-frequency sensor measurements in their local filters. These design decisions are a trade-off between modularity, computational efficiency, and the integration of all available information in a smoothing process. Our explicit decoupling of filter and graph SLAM gives us greater design freedom for both sub-systems compared to tightly coupled solutions such as, for example, *Concurrent Filtering and Smoothing* (Williams et al., 2014).

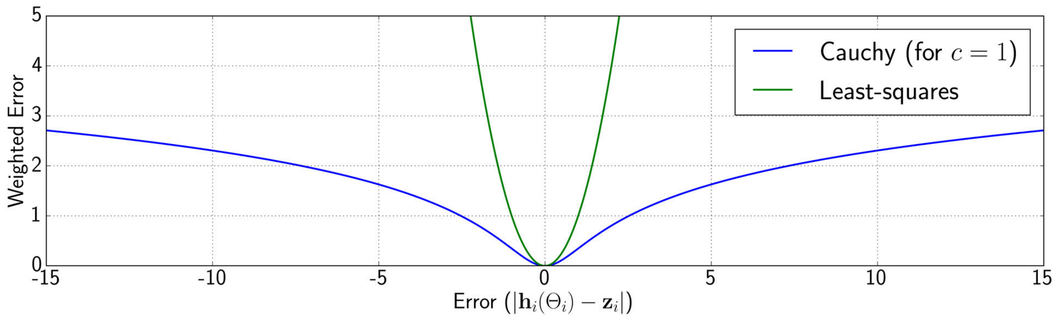


Figure 4.3: *Exemplary comparison of quadratic and Cauchy error functions*

4.2.3 SLAM Back-End: Incremental Optimization

As discussed in Section 2.2.3, the computational cost of batch graph optimization grows with the size of the graph and is therefore not suitable for online global optimization. Thus, we decided to utilize the incremental *iSAM2* optimizer (Kaess et al., 2012), an implementation of which is available as open source within the *GTSAM 3.2.1* library (Dellaert, 2015). The key idea of iSAM2 for efficient incremental optimization is the conversion of the factor graph to a Bayes net and further to a Bayes tree. This data structure allows to add new variables and factors while keeping those sub-trees unchanged that are not affected by local loop closures. In our system, it thus allows for fast average optimization steps when adding new measurements and filter estimates, while slower and computationally more demanding optimization steps are limited to the infrequent occurrences of large loop closures.

Overconfident, erroneous loop closure constraints can corrupt the entire graph optimization result. Robust SLAM back-ends mitigate this risk, for example by applying a dynamic scaling to the respective measurement covariances in order to reduce the influence of outliers (Agarwal et al., 2013; Latif et al., 2014). We replace the quadratic error term in Equation 4.5 with a robust error function for the integration of our landmark and robot detections as well as submap match estimates. In particular, we employ the GTSAM implementation of M-Estimators and chose the Cauchy error function, as it is suitable to suppress outliers with large errors (Lee et al., 2013):

$$\rho(x) = \frac{c^2}{2} \log\left(1 + \left(\frac{x}{c}\right)^2\right) \tag{4.10}$$

In Figure 4.3, we show a comparison of the Cauchy function to a standard least-squares error function. Applying the Cauchy function, the optimization problem can be formulated as an iterative re-weighted least-squares minimization with the weights in the k^{th} iteration being computed as

$$\boldsymbol{w}^k = \frac{c^2}{c^2 + \left\|\boldsymbol{h}_i\left(\Theta_i^{k-1}\right) - \boldsymbol{z}_i\right\|_{\Sigma_i}^2} \qquad (4.11)$$

where the value of the constant c determines the range of the Mahalanobis distances (Mahalanobis, 1936) to be still considered as inliers. This error function allows us to gain robustness by mitigating the influence of large outliers that can originate from incorrect data associations as well as measurement or estimation errors.

4.3 Related Work and Discussion

We present an experimental evaluation comparing the two SLAM graph topologies in Section 8.2.2. Our novel integration of local references filter estimates according to their dependencies and uncertainties improved the 3D self-localization accuracy by, on average, 15 %. In Section 8.3, we evaluate the accuracy of our full multi-robot SLAM system in a total of seven further experiments in areas of up to 57 m×53 m. Our system achieved average errors below 0.5 % of the robots' respective total trajectories w. r. t. partial ground truth and its separation of local and global methods led to small SLAM graphs with less than 180 nodes and 250 factors in all of our experiments.

In the remainder of this section, we first discuss similarities and differences of our method compared to other recent approaches on the combination of local and global methods for online SLAM. In the second part, we focus on related work with regard to multi-robot graph construction and optimization.

4.3.1 Combination of Local and Global Methods

We integrate filter and graph SLAM methods to fulfill both local real-time requirements as well as to compute global multi-robot estimates. Similarly, Leishman et al. (2013) and Mohanarajah et al. (2015) also present SLAM systems that combine keyframe-based approaches with a pose graph for global localization. In contrast to our work, they explicitly represent all (Red, Green, Blue, Depth) (RGBD) keyframes as individual nodes in their graph. We decouple our SLAM graph from such low-level states and trigger the creation of local reference frames as new graph nodes from our higher-level mapping modules instead. Thereby, we can keep the size of the graph independent of the number of keyframes from lower-level vision modules, which is typically orders of magnitude larger than the number of submaps required in our system. Furthermore, our local data aggregation into submaps allows to keep more 3D information about the environment than a sparse sampling of RGBD keyframes.

Similar to our approach, Vidal-Calleja et al. (2011) designed a hierarchical system of EKFs at the level of local submaps and an adjacency graph for smoothing by incorporating loop closures at the global, multi-robot level. However, at the graph level, they do not distinguish between robot poses and submap poses. Thus, they would not be able to cope with partial frame switches, as proposed in our system. Splitting the filter state into its observable and unobservable modes, we can define all submaps to be gravity-aligned. This allows for an improved submap matching, as described in Section 6.1. It, however, requires a separation of the set of robot poses from the set of submap origins. Instead of this, Vidal-Calleja et al. (2011) define a new local reference for each submap at the robot pose by starting a new local EKF that resets the full robot pose and covariance estimates to zero. This requires that they start a new submap each time a robot pose is added at the graph level in order to connect subgraphs via loop closure constraints, i. e., on each multi-robot rendezvous, submap match, or Global Positioning System (GPS) fix. Depending on the respective measurement frequencies, this could lead to small submaps of little use or would require an upper limit on the number of loop closures within a certain time frame or distance. While we also prefer low-frequency measurements for integration at the graph level, our approach does not impose the same limitations. Our novel graph topology for the integration of filter estimates, as presented in Figure 4.2(b), allows a separation of submap origins and filter estimates. We add relative robot poses, estimated by the filter in its local reference frame, i. e., in the current submap origin, as additional nodes in the graph whenever required in order to integrate a measurement constraint between the robot and some other node. Thus, our system can trigger frame switches and the creation of new submaps independently of loop closure events and solely base these decisions on considerations w. r. t. the accumulated uncertainty and map size, as discussed in Section 5.1.3.

Williams et al. (2014) as well as Emter and Petereit (2018) also combine filter- and graph optimization to satisfy the real-time requirements imposed by robot control as well as the need for online global pose and map estimates. In contrast to our loosely coupled approach, Williams et al. (2014) combine filter and graph SLAM by splitting a graph containing all measurements into a real-time filter part for the most recent data and a slower smoother part for past states, both running in parallel. While they are able to recover the solution of full batch optimization, they require a tight coupling between filter and smoother, working on the same types of sensor data and exchanging state information in both directions. In contrast, we propose to process sensor information at different levels of abstraction. By solely adding aggregated pose information to the graph, we can keep it small, allowing for fast, online global optimization. As we combine the results from our real-time filter and online graph optimization, it is sufficient to fuse all high-frequency measurements only once, i. e., aggregating them in the EKF. In contrast, Emter and Petereit (2018) process them at least twice[4], once in their real-time EKF and then again in the full graph optimization. In addition, our loose coupling of filter and graph SLAM gives more design freedom for both subsystems, as the graph level does not need any information about internal states of the filter, nor about the data from lower-level components. Thus, it would be possible to add additional sensor input like, for example, barometer height measurements to the filter or even to completely replace the visual odometry pipeline without making any changes to the graph creation and optimization. Furthermore, in contrast to Williams et al. (2014) as well as Emter and Petereit (2018), we do not feed back loop closure results into our filter, allowing it to generate smooth local state estimates required for robot control.

4.3.2 Multi-Robot Graph Optimization

Focusing on the aspects of multi-robot graph creation, exchange, and optimization, we consider several existing approaches to discuss in comparison to our system with respect to major design decisions and their respective trade-offs.

In the hierarchical SLAM system proposed by Vidal-Calleja et al. (2011), only the smallest cycle in the graph is used for optimization. Furthermore, in the distributed version of their algorithm, they restrict the enforcement of loop closure constraints to situations in which all robots in a multi-robot system can close the same loops in order to ensure compatible graphs. In contrast, our system employs the iSAM2 optimizer to perform online incremental optimization approximating the Maximum a Posteriori Probability (MAP) solution by taking all loop closure constraints into account. Also in cases of limited communication between robots, we add all loop closure constraints as soon as they are available to the graph optimization of each robot in order to compute a global estimate that is as up-to-date as possible and includes all available information. As incremental optimization can be sensitive

[4]Emter and Petereit (2018) employ an additional catchup-EKF to bridge the time delay caused by graph optimization when propagating estimates back to the online EKF. Thus, some measurements need to be processed even thrice.

to the order in which new constraints are added and alternated with intermediate optimization steps, we cannot guarantee to get exactly the same results on all robot systems. While in our experiments we did not observe any significant deviations, it would be possible to mitigate potential long-term inconsistencies by performing slower batch optimization steps on the full graph from time to time. They would have the added benefit of allowing robust error functions to better suppress outliers that had been inserted early on during graph creation by performing global consistency checks that are independent of the order of measurements.

Similar to Kim et al. (2010), we do not make any assumptions about the synchronization of encounters in order to add constraints between the partial graphs of different robots, except for synchronized system clocks. Thus, multi-robot estimates like submap matches can connect nodes that have been created at arbitrary times. This is also true in case of direct encounters, i. e., robot observations, that happened, for example, during a communication outage. As robot poses x_i^r of the same robot r are not directly connected to each other, but only to their respective submap origins, our graph topology allows to add them at any later point in time without the need to remove any previous edges from the graph.

In order to specify frame-of-reference constraints between the subgraphs of multiple robots, Kim et al. (2010) introduce *anchor nodes*. This allows them to locally optimize unconnected subgraphs for each robot and connect these once the first respective inter-robot measurement is available. Lázaro et al. (2013) and Cunningham et al. (2013) go one step further and exchange condensed graphs between robots, the latter explicitly removing double-counted information by introducing anti-factors. Double counting cannot occur in our proposed system as each robot adds all estimates and measurements to its own graph only once. In contrast to all three approaches, we do not expect significant benefits from early optimization of unconnected sub-graphs (Kim et al., 2010) or an exchange of optimized partial graphs (Lázaro et al., 2013; Cunningham et al., 2013) for our method as its combination of local reference filter estimates sampled at a low frequency leads to a small joint graph in the first place.

Chapter five

Dense 3D Map Creation and Exchange

In our mapping system, we distinguish between the creation of local and global models of the environment as they need to satisfy different requirements regarding update rates, accuracy, and map content. The local aggregation and global composition of dense 3D map data are based on the outcome of each robot's respective local and global pose estimation modules that we introduced in Chapter 4.

Thus, we divided this chapter into two parts: The first part, Section 5.1, is concerned with the local terrain classification and aggregation of high-bandwidth dense 3D data. In order to create a joint multi-robot map online, we follow a distributed approach by splitting the maps of each robot into submaps of limited size. In the second part, Section 5.2, we describe the exchange of these submaps between robots and their composition into a global map. The creation of submaps does not only permit efficient sharing of map data between robots over communication links with limited bandwidth, but also allows to generate global intra- and inter-robot loop closures through a pairwise matching of these partial maps, which we present in Section 6.1.

5.1 Local Mapping

The local part of our mapping pipeline runs on each robot and only processes the high-bandwidth data generated by its own stereo vision system. We start this section with a terrain classification regarding obstacles and traversable areas that can be directly used for local path planning and obstacle avoidance of ground-based robots like planetary exploration rovers. We then present our definition of submaps and how we use them to partition the robot's environment model.

5.1.1 Obstacle Classification

The identification of untraversable areas in their surroundings allows robots not only to avoid these, but also to use formations of such obstacles as geometric landmarks to support place recognition. We exploit this for loop closure detection, as we discuss in Section 6.1. In the following paragraphs, we describe how we locally identify obstacles directly in the robots' depth images, which is part of the work that we first presented in our conference paper by Brand et al. (2014).[1] The first step of our processing pipeline is a detection of negative edges, such as steep cliffs, directly in the depth data. It is followed by a stereo error-adaptive terrain classification based on local 2.5D maps.

Negative Edge Detection

The obstacle avoidance of any ground-based mobile robot system needs to consider negative edges, i. e., steep, downward-facing slopes like cliffs or vertical cave entrances, since moving beyond them could lead to serious damage. Such geometrical features can be detected directly in the depth images by searching for discontinuities in the depth values of local neighborhoods in image coordinates. This detection of steep negative slopes needs to be performed directly on the depth images. The information that is required for this classification will be lost after subsequent processing steps, such as the conversion of the depth images into point clouds or local 2.5D maps. Thus, we introduce it as the first step of our processing pipeline and separate it from the detection of "normal" steps and slopes in local maps, as described below.

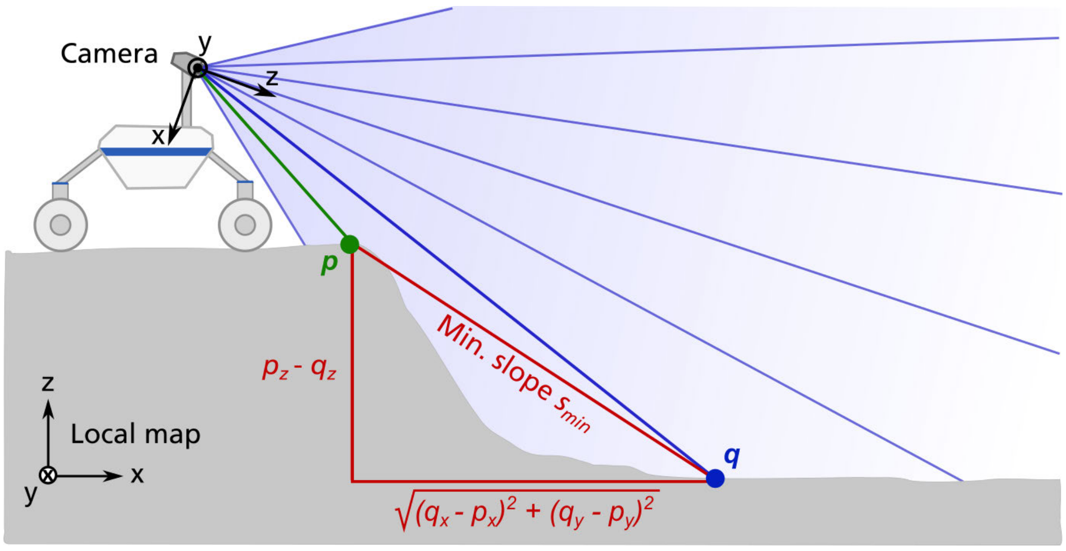
[1]The obstacle and terrain classification algorithm, as first published by Brand et al. (2014), has been developed by Christoph Brand and me in close collaboration with my primary focus on its application for loop closure generation in SLAM systems.

Considering a robot standing on top of a high cliff, the lower ground might lie beyond its local map or even beyond the valid range for stereo reconstruction. In the latter case, both cameras observe the same point at the same image coordinates. This leads to a disparity value of zero, which equals a depth estimate at infinity. In the representation of a depth image, this information is still available and can be distinguished from missing or invalid data points. However, it cannot be preserved during the transformation of depth data from the camera coordinate frame into an elevation-based grid map. Thus, while the existence of a steep cliff can be deducted from a depth image and its known viewpoint, this information will be lost later on.

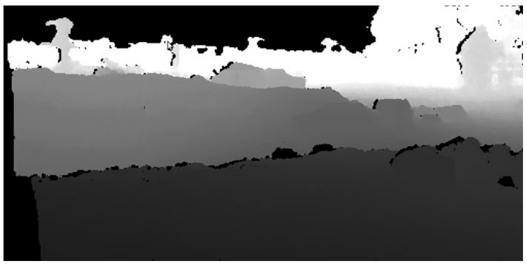
In Figure 5.1(a), we sketch out an example of a mobile robot driving towards the edge of a steep cliff. We visualize the pixels of a column of its depth image by beams originating in the camera center. In this scenario, the robot cannot observe the slope of the cliff itself but can only see areas above and below it. In the depth image, there will be a significant difference in depth values between the respective neighboring pixels representing these areas. In order to detect this discontinuity, it is sufficient to iterate column-wise, from top to bottom, over all pixels in the depth image. This is based on the assumption that the *roll* angle of the robot is small, i. e., that the robot is not heavily tilted sideways. In comparison to well-established edge detection algorithms, such as the edge detector by Canny (1986), our method is computationally less complex: Each pixel only needs to be compared with its direct upper neighbor, which was accessed in the previous iteration. If their depth values are above a threshold (0.2 m for our rovers), we assume the point corresponding to the upper pixel to be located at the top of a potential cliff and the one corresponding to the lower pixel at its bottom. In order to check the slope, we transform both points into a gravity-aligned local map frame with its z-axis pointing upwards. Representing them as the vectors $\boldsymbol{p} = (p_x, p_y, p_z)$ and $\boldsymbol{q} = (q_x, q_y, q_z)$, we can compute the minimal slope that is possible to lie between them as the gradient

$$s_{min} = \frac{p_z - q_z}{\sqrt{(q_x - p_x)^2 + (q_y - p_y)^2}} \tag{5.1}$$

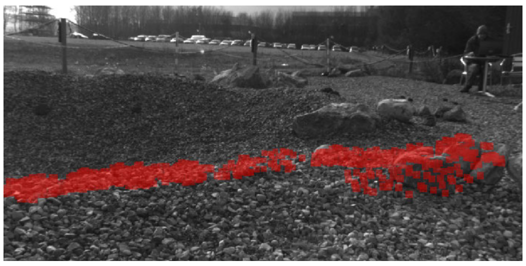
This is the slope of a hypothetical ray connecting the areas above and below the cliff. Therefore, the real geometry of the cliff must either have a slope equal to s_{min} or contain at least some part with a slope steeper than that. Checking s_{min} against a robot-specific maximum-slope threshold defined by a robot's locomotion capabilities thus allows it to classify the cliff as an untraversable obstacle. In Figure 5.1(b), we present a depth image taken by a rover in our outdoor testbed. The robot's stereo cameras are looking right over the edge of a crater, which is clearly visible in the depth image as a sudden change in grey values. This indicates a jump in depth



(a) *Schematic*



(b) *Depth image*



(c) *Camera image*

Figure 5.1: *Negative edge detection at a cliff not visible to the robot (a). Computing the minimal possible slope s_{min} between the two observable points $\boldsymbol{p}$ and $\boldsymbol{q}$, located on opposite sides of the cliff, we can decide whether it is too steep to be traversable. We detect negative edges as discontinuities in the depth image (b); areas classified as untraversable are highlighted in red in the respective part of the camera view (c).*

values along the z-axis of the camera and thus a negative edge. In Figure 5.1(c), we visualize the resulting classification of the crater edge as an untraversable obstacle by red points that we projected into the greyscale image of the robot's camera.

Stereo Error-Adaptive Classification

Sensor noise, calibration errors, as well as errors from stereo reconstruction itself lead to uncertainties and propagated errors in the resulting 3D data. This can affect the classification of obstacles and thus needs to be taken into account in the following processing steps. Assuming independent white Gaussian noise at pixel-level in each dimension of the image coordinates for the left and right camera images of a stereo pair, the propagated error for a 3D point $\boldsymbol{p} = (p_x,\ p_y,\ p_z)$ reconstructed in the camera

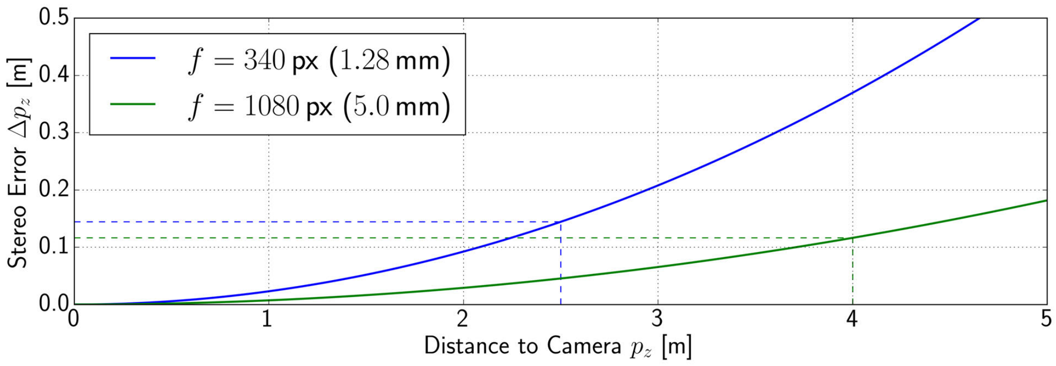


Figure 5.2: *Propagated stereo reconstruction error Δp_z for points at the center of the image ($p_x = 0, p_y = 0$) for narrow- and wide-angle camera lenses used on our robots. The dotted lines indicate maximum view distances that we used in some of our experiments to limit the error below 0.15 m.*

coordinate frame (z-axis pointing into the image plane) from an image-based error Δp can be computed as outlined by Hirschmüller (2003):

$$\Delta p_x = \Delta p \, \frac{p_z}{f\, b} \sqrt{(b - p_x)^2 + p_x^2} \tag{5.2a}$$

$$\Delta p_y = \Delta p \, \frac{p_z}{f\, b} \sqrt{2\, p_y^2 + \frac{b^2}{2}} \tag{5.2b}$$

$$\Delta p_z = \Delta p \, \frac{p_z^2}{f\, b} \sqrt{2} \tag{5.2c}$$

In these equations, f denotes the focal length of the camera in pixels and b the length of the stereo basis. For a point with $p_x = p_y = 0$, its distance to the camera is equal to p_z. It is important to note that, while in Equation 5.2, the errors Δp_x and Δp_y depend linearly on p_z, the error Δp_z scales quadratically with the distance to the camera. In Figure 5.2, we plot Δp_z for the different narrow- and wide-angle stereo camera setups of the robots that we used for the evaluation of our algorithms, see Section 7.1. All of them have a baseline of $b = 9$ cm. We assume pixel errors of $\Delta p = 0.5$ px as we obtained a Root-Mean-Square Error (RMSE) of up to 0.46 px during camera calibrations. The error in the outer parts of the image, however, might be larger. The plots in Figure 5.2 highlight the importance of the consideration of the uncertainty in depth measurements when processing stereo data. In order to be able to build maps that are sufficiently accurate for robot navigation, for all subsequent processing, we filter out 3D points with Δp_z above a threshold that depends on the camera system and the required accuracy. In our experiments, we achieved good results with values in the range of [2.5 m, 5 m] for our narrow- and wide-angle cameras. The near-range data below this threshold is still sufficient for navigation of a relatively slow-moving system like a space exploration rover, including obstacle avoidance, map building and the computation of stereo visual odometry. In general, choosing an appropriate focal length and baseline for the navigation stereo cameras of a robot is a design decision involving trade-offs that constrain its workspace and capabilities in either the near- or far-range. For our rover systems, even after omitting the far-range data, the stereo error, however, can still be large enough to affect obstacle classification. Thus, we require an adaptive treatment of the reconstructed 3D points based on their location relative to the camera and the parameters of the stereo system.

We first transform the depth image into a 2.5D height map as a gravity-aligned top-down view of the local scene. In our experiments, we used a resolution of 0.05 m. Each of the grid cells contains the mean height value of all points falling into it. The maximum difference in height between cells in a local neighborhood gives a good indication whether a cell is traversable or not. However, it is affected by the aforementioned errors from stereo reconstruction. Thus, we take this uncertainty into account when detecting untraversable steps in the environment. In order to estimate the mean stereo error, we consider two 3D points $\boldsymbol{p} = \left(p_x,\ p_y,\ p_z\right)$ and $\boldsymbol{q} = \left(q_x,\ q_y,\ q_z\right)$ in the camera frame and their Euclidean distance:

$$d = \sqrt{(p_x - q_x)^2 + \left(p_y - q_y\right)^2 + (p_z - q_z)^2} \tag{5.3}$$

According to Hirschmüller et al. (2002), the mean error in the distance between the points Δd, given the focal length f in pixels and the stereo basis b in meters, can be computed from the propagated mean pixel error Δp as follows:

$$\Delta d = \frac{\Delta p}{d\,f\,b}\sqrt{q_z^2\,(A + B + C) + p_z^2\,(C + D + E)} \tag{5.4a}$$

$$A = \left((q_x - p_x)\,(t - q_x) - \left(q_y - p_y\right)\,q_y + (q_z - p_z)\,q_z\right)^2 \tag{5.4b}$$

$$B = \left((q_x - p_x)\,q_x + \left(q_y - p_y\right)\,q_y + (q_z - p_z)\,q_z\right)^2 \tag{5.4c}$$

$$C = \frac{1}{2}\,\left(b\,\left(q_y - p_y\right)\right)^2 \tag{5.4d}$$

$$D = \left((q_x - p_x)\,(b - p_x) - \left(q_y - p_y\right)\,p_y - (q_z - p_z)\,p_z\right)^2 \tag{5.4e}$$

$$E = \left((q_x - p_x)\,p_x + \left(q_y - p_y\right)\,p_y + (q_z - p_z)\,p_z\right)^2 \tag{5.4f}$$

Around each grid cell of the height map, we select a small square neighborhood with an empirically determined edge length of three to six cells, i. e., 0.15 m to 0.3 m at a resolution of 0.05 m. Therein, we compute the maximal difference in height between cells h_{diff} and check it against a stereo error-adaptive threshold, as sketched out in Figure 5.3. All cells for which $h_{diff} > h_{max} + \Delta d$ are then classified

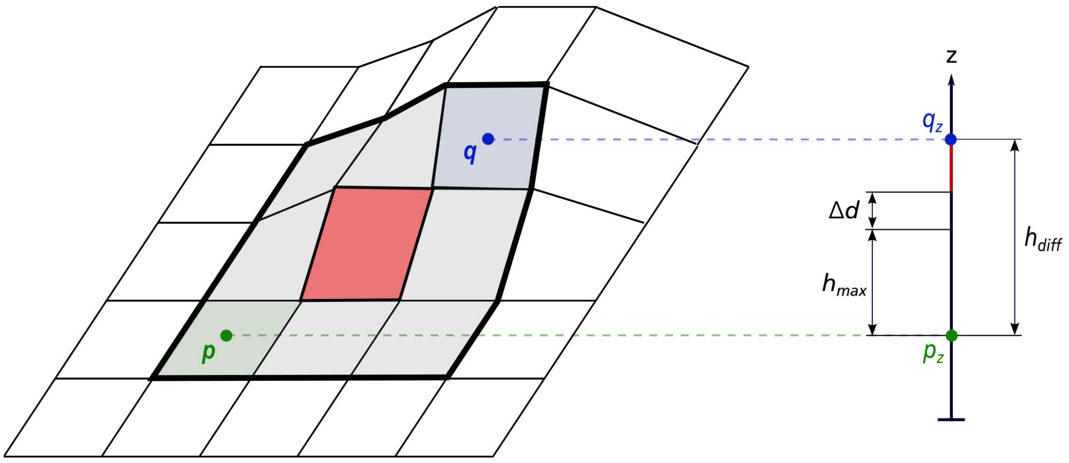
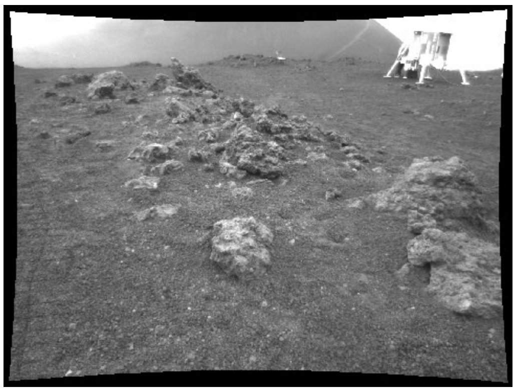


Figure 5.3: *Schematic of stereo error-adaptive obstacle classification for grid cells based on the maximum height difference of neighboring cells within a window of limited size. In the exemplary case of the center cell, this difference exceeds the adaptive threshold. It thus is classified as untraversable (red).*

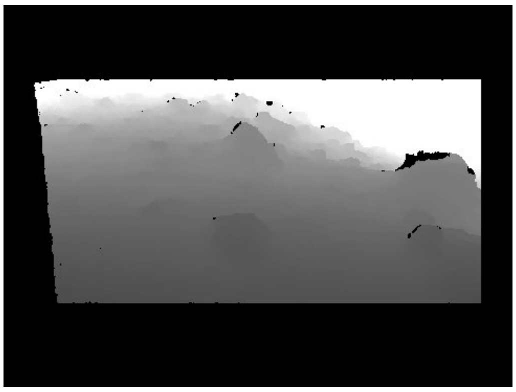
as obstacles. This threshold consists of a constant and a variable part: The constant part h_{max} represents the maximal traversable step height and is determined by the locomotion capabilities of the robot system. We compute the variable part Δd by applying Equation 5.4 to the center points of the two cells with the maximum height difference, visualized as $\boldsymbol{p}$ and $\boldsymbol{q}$ in Figure 5.3. It is important to note that we first have to transform the two points back into the camera coordinate frame. This error-adaptive part of the threshold allows us to account for the uncertainty of stereo estimation w. r. t. to the difference in height between the cells. Neglecting it would lead to false positives depending on the position of potential obstacles relative to the robot's camera: The robot would hallucinate obstacles due to stereo estimation errors, in particular in areas further away from the camera. While probabilistic sensor and environment models could be applied to compute more accurate estimates, they would also introduce additional computational complexity and thus potentially delay the avoidance of obstacles based on this classification.

Local Obstacle Maps

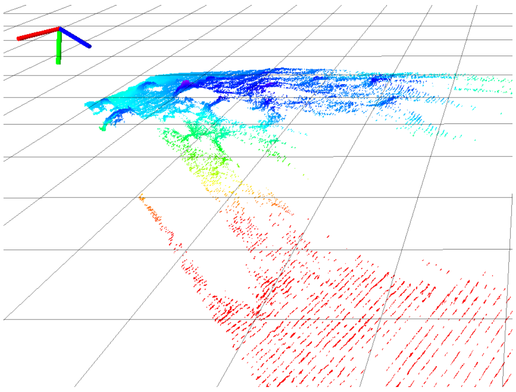
Both steps of the obstacle classification, the negative edge detection and the stereo error-adaptive step detection, depend on knowledge of the viewpoints of the robot's camera, information that is lost during later processing steps that aggregate the depth data into maps. Thus, it is necessary to perform the obstacle classification at the beginning of the mapping pipeline. In addition, this allows to run the local obstacle mapping at a higher frequency than subsequent steps for global mapping, allowing fast updates even on resource-constrained systems. For obstacle avoidance, we



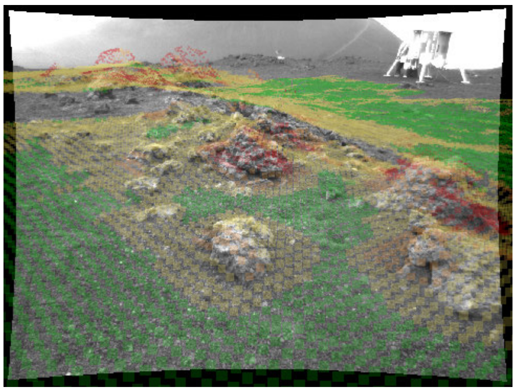
(a) Greyscale image from left camera



(b) Depth image from stereo reconstruction



(c) Height-colored point cloud (unfiltered) and coordinate frame of the camera pointing in direction of the blue z-axis



(d) Color-coded local obstacle map classifying the terrain from easy (green) to untraversable obstacles (red)

***Figure 5.4:** Obstacle classification: example of input data and resulting local obstacle map from one of our experiments at a Moon-analogue site on Mt. Etna (see Section 9.3)*

aggregate the terrain classification results from subsequent depth images by applying a temporal low-pass filtering. For this, we add the data to a rolling top-down 2.5D map, i. e., a grid map of fixed maximum size that is always centered around the robot in order to model its surroundings. This map encodes a terrain traversability value for all cells, which is computed as $\frac{h_{diff}}{h_{max}+\Delta d} \in [0, +\infty)$ based on the step classification described above, with values greater than one indicating untraversable obstacles. The resulting grid map can directly be used to locally plan obstacle-free paths for the robot. In Figure 5.4, we give an example of such a local obstacle map, together with its input camera and depth data from stereo reconstruction. In our experiments, we used a spatial map resolution of 0.05 m, which we empirically determined to be sufficient for obstacle avoidance with our rovers. We provide a discussion of our local obstacle mapping and set it in relation to other approaches from literature at the end of this chapter in Section 5.3.1.

5.1.2 Submap Generation

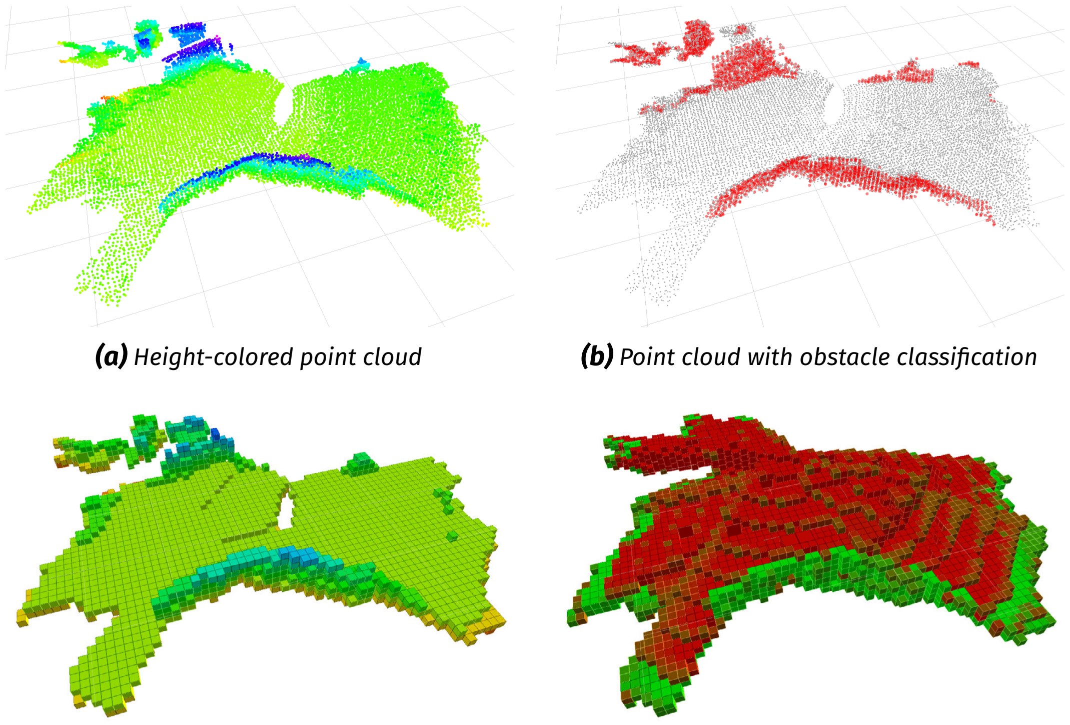
While local rolling maps are useful for local obstacle avoidance of individual robots, our goal is to create a globally optimized 3D map of the environment to facilitate multi-robot planning, for example to realize a collaborative exploration of previously unknown areas. In addition, environment models that capture its 3D geometry are crucial for data association between different types of robots. As stated by Vidal-Calleja et al. (2011), in the context of multi-robot mapping in heterogeneous teams, "[...] the 3D geometry of the environment is the only intrinsic environment characteristic on which one can rely to tackle data association, as it can cope with the fact that the sensors or viewpoints can be very different among the different robots".

We follow a submapping approach in order to handle the large quantities of sensor data, i. e., dense 3D information, that need to be processed and transferred between robots. We aggregate this high-bandwidth sensor data into so-called submaps, i. e., partial maps of limited size. We define a submap as a local reference frame, i. e., the pose of its origin, and associated 3D data structures. It is important to note that for each new submap, we trigger a frame switch in our local reference filter. Thus, the origin of each submap by definition coincides with a respective local reference frame in the filter. We then add these submap origins to our SLAM graph for global optimization, as described in Section 4.2. In order to create 3D submaps, we aggregate the depth data and the aforementioned obstacle classification results along the trajectories estimated by our local reference filter. Each submap has two application-dependent 3D representations with complementary characteristics: a point cloud and a probabilistic voxel space, both of which are shown in Figure 5.5.

3D Point Cloud

The first representation is a set of points with (x, y, z) coordinates, (r, g, b) color information and a binary obstacle classification for each point. We limit the density of the point cloud in order to keep the amount of data manageable and compact for the exchange between robots. In our experiments, we enforced a maximum spatial resolution of $r = 5\,\text{cm}$, as we will explain below.

For the iterative aggregation of new stereo data, we first convert each depth image into a point cloud and, similar to the obstacle classification discussed in the previous section, discard all points that are too far away from the camera and thus are potentially affected by large stereo reconstruction errors. We then apply a voxel-grid filter with grid size r. It limits the density of the input data in order to speed up the subsequent processing steps. For each non-empty voxel, it computes the centroid of



(a) *Height-colored point cloud*

(b) *Point cloud with obstacle classification*

(c) *3D voxel grid map: height-colored known and occupied voxels*

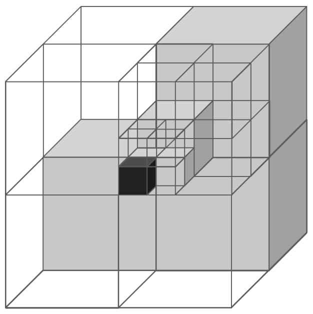
(d) *3D voxel grid map: occupancy probability for known voxels from free (red) to occupied (green)*

Figure 5.5: *Submap representation: visualizations of point cloud (a),(b) and probabilistic voxel space (c),(d) data structures attached to each submap*

the set of all points that fall into it, i. e., their arithmetic mean position.[2] This results in a more accurate representation of surfaces compared to simply using the voxel center coordinates, which are defined by the fixed 3D grid. We classify the resulting point as an obstacle if at least one of its input points was classified as such. The filtered points are then added to the set of points of the submap. Afterwards, we apply another voxel-grid filter to the point cloud of the submap itself to reduce its final density to the desired maximum resolution r.

As point clouds are simple representations of object surfaces that are computationally cheap to create, we can use a higher spatial resolution r than for the more complex probabilistic voxel spaces described below. We use the point cloud representations for map matching, see Section 6.1, and for map visualization during our experiments.

[2]We use the voxel-grid filter implementation from the open source Point Cloud Library (PCL) 1.7.2 library (Rusu and Cousins, 2011).


(a) *Octree voxel map*

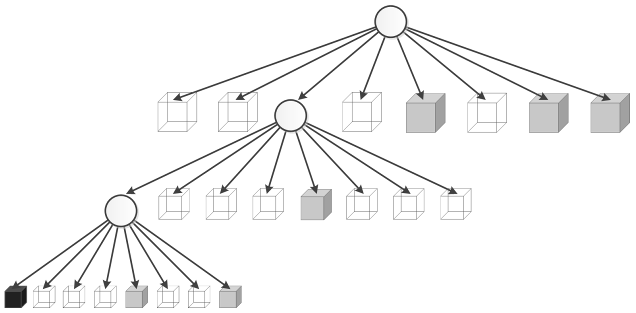

(b) *Octree graph*

Figure 5.6: *Octree representation: example of volumetric voxel map (a) and corresponding tree structure (b). White voxels depict free, black voxels occupied, and transparent voxels unknown space.*

3D Probabilistic Voxel Space

The second representation is a 3D voxel grid that contains probabilistic volumetric information. This type of environment model has a higher complexity and is computationally more expensive than a simple point cloud representation. In our experiments, we thus chose a coarser resolution of $r = 10\,\text{cm}$ as a trade-off between computational load and required precision.

For the fast and memory-efficient creation of a probabilistic voxel space representation, we employ the freely available open-source "OctoMap" framework by Wurm et al. (2010) and Hornung et al. (2013). In the following paragraphs, we will briefly summarize its key ideas. It is based on an *octree* representation, in which cubic volumes, the voxels, are recursively subdivided until a minimum voxel size, i. e., maximum map resolution, is reached. Each voxel is split into eight sub-volumes that are represented by its children in the octree notation, as shown in Figure 5.6. This allows for a memory-efficient storage and compact serialization for the transfer of maps between robots. Furthermore, the hierarchical data structure enables queries to the map at multiple resolutions.

In octree-based occupancy grid mapping, the probabilities $P(\boldsymbol{n}|z_{1:t})$ for each leaf node $\boldsymbol{n}$ to be occupied are assumed to be independent from each other given a series of t sensor measurements $z_{1:t}$. Hornung et al. (2013) estimate them according to

$$P(\boldsymbol{n}|z_{1:t}) = \left[1 + \underbrace{\frac{1 - P(\boldsymbol{n}|z_t)}{P(\boldsymbol{n}|z_t)}}_{\text{measurement}} \cdot \underbrace{\frac{1 - P(\boldsymbol{n}|z_{1:t-1})}{P(\boldsymbol{n}|z_{1:t-1})}}_{\text{recursive term}} \cdot \underbrace{\frac{P(\boldsymbol{n})}{1 - P(\boldsymbol{n})}}_{\text{prior}}\right]^{-1} \tag{5.5}$$

This update formula contains a term for the sensor model $P(\boldsymbol{n}|z_t)$, specifying the probability that node $\boldsymbol{n}$ is occupied given the measurement z_t, a recursive update term referring to the previous estimate $P(\boldsymbol{n}|z_{1:t-1})$ as well as a prior probability $P(\boldsymbol{n})$. Using the log-odds notation $L(\boldsymbol{n})$ defined as

$$L(\boldsymbol{n}) = \log\left[\frac{P(\boldsymbol{n})}{1 - P(\boldsymbol{n})}\right] \tag{5.6}$$

and assuming a uniform prior $P(\boldsymbol{n}) = 0.5$ (value for “unknown”, i. e., asserting the absence of prior knowledge), Equation 5.5 can be re-written as

$$L(\boldsymbol{n}|z_{1:t}) = L(\boldsymbol{n}|z_{1:t-1}) + L(\boldsymbol{n}|z_t) \tag{5.7}$$

This formulation replaces multiplications by additions and thus allows for computationally efficient iterative update steps without any loss of information, as the probabilities can be recovered from the log-odds. When using a voxel-grid map for visualization or robot navigation, thresholds can be applied to map the probabilities to the three distinct states *free*, *occupied*, and *unknown/uncertain*.

New data is added via a beam-based sensor model: For each pixel in the stereo depth map, a ray is cast between the camera and its measurement endpoint. All voxels along this ray are updated to be *free* and the endpoint to be *occupied*, using the following inverse sensor model given by Hornung et al. (2013):

$$L(\boldsymbol{n}|z_t) = \begin{cases} 0.85 \;\; (\widehat{=} \text{ probability of } 0.7) \text{ for updates as } \textit{occupied} \\ -0.4 \;\; (\widehat{=} \text{ probabilitiy of } 0.4) \text{ for updates as } \textit{free} \end{cases} \tag{5.8}$$

For each depth map, a node gets updated at most once, with a preference for the *occupied* state over the *free* state. This avoids holes in the environment model due to conflicting updates from neighboring pixels in the same depth image.

In order to ensure the adaptability of the environment model regarding dynamic objects, changes in the environment, and the correction of erroneous measurements, the log-odd values for each voxel are clamped. We use lower and upper bounds of $l_{min} = -2$ and $l_{max} = 3.5$ (corresponding to probabilities of 0.12 and 0.97), which have been experimentally determined by Hornung et al. (2013) for the mapping of mostly static environments. These bound the confidence in individual parts of the map. Assuming a fixed sensor model, they define a maximum number of updates that are required to change the state of a voxel between *free* and *occupied*. Thus, Hornung et al. (2013) replace Equation 5.7 by

$$L(\boldsymbol{n}|z_{1:t}) = \max(\,\min(\,L(\boldsymbol{n}|z_{1:t-1}) + L(\boldsymbol{n}|z_t),\; l_{max}),\; l_{min}) \tag{5.9}$$

Furthermore, the clamping enables a better compression of the map through an extended pruning of inner nodes. Pruning with regard to nodes with boolean or discrete variables denotes the removal of all children of a node if they have the same state, leading to a more compact tree representation. In the probabilistic model, the aforementioned clamping values define "stable nodes" with high confidence in their free or occupied state that, if occurring adjacent in suitable neighborhoods, can be compacted via pruning. The combination of clamping and pruning thus leads to lossy compression of the probabilistic map.

We refer to the work by Hornung et al. (2013) for further in-depth information on octree-based probabilistic voxel maps. On our robots, we use probabilistic voxel maps for autonomous exploration based on expected information gain, as we demonstrated in experiments discussed in Section 9.3. This volumetric representation can also serve as valuable input for further applications, such as the navigation of MAVs as well as for whole-body path planning of mobile robots with manipulator arms. All of these applications require an explicit distinction between occupied, free, and unknown space, which is not available in simple point cloud representations.

5.1.3 Submap Partitioning

We partition the global multi-robot map into partial maps by creating a series of submaps for each robot. This partitioning is based on two criteria that we apply to start a new submap. First, when aggregating the dense 3D data of a robot along its trajectory, we assume that its filter estimates are locally sufficiently accurate. In order to satisfy this assumption, we create a new submap once the filter's estimated uncertainty grows above map resolution, which on our robots is 0.1 m for the probabilistic voxel maps. As our first criterion, we therefore apply a threshold of 0.1 m on the estimated standard deviation of a robot's position. This not only ensures a limited drift within each submap, but, by triggering a frame switch, also limits the accumulated error in the local reference filter. The periodical definition of small-scale submaps is an important measure to improve long-term filter consistency, as recommended by Bailey et al. (2006) and discussed in Section 4.1.2. As our second criterion, we employ an empirically determined threshold on the accumulated driven distance (3.5 m in early and 7 m in later experiments, see Chapter 8), thereby restricting the size of the individual submaps in order to limit their memory and processing time requirements during exchange and matching. In Figure 5.7 we show an example of the partitioning of one of our rover's maps into several submaps. As the localization

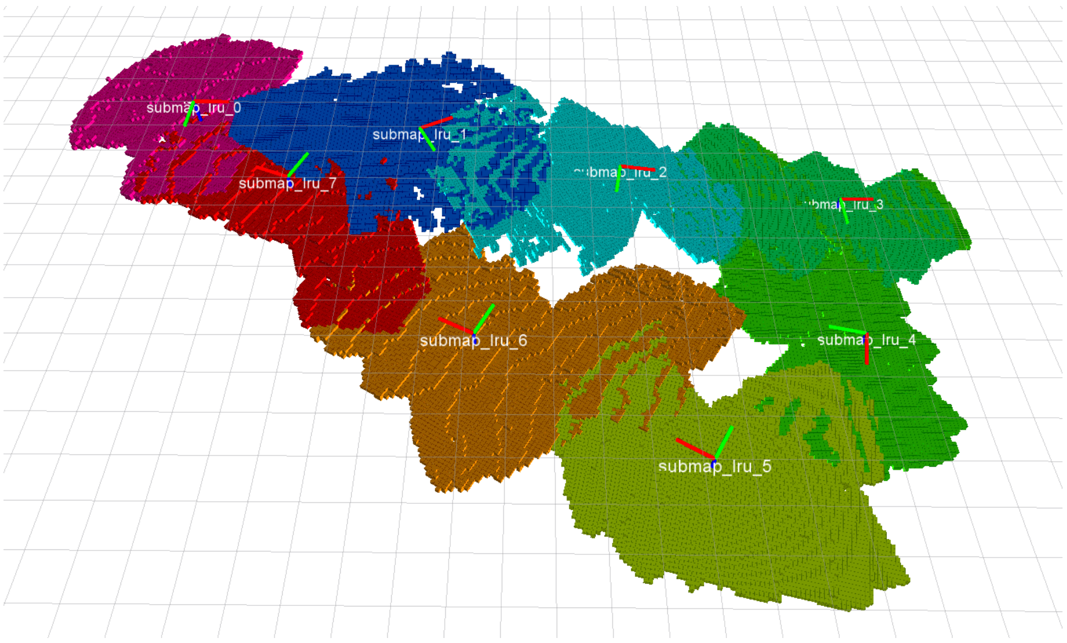


Figure 5.7: *Submap partitioning: Different colors visualize the voxel map representation of different submaps, the coordinate frames indicate their respective origins. The map was created from a dataset gathered by LRU operating in a Moon-analogue environment on Mt. Etna, Italy (see also Section 9.3).*

error within a submap is limited by the uncertainty-based threshold, we perform global corrections only between the submaps of one or multiple robots.

5.2 Global Mapping

In order to create a global, multi-robot map, we exchange the partial submaps created by the individual robots (Section 5.2.1) and compose them into a globally optimized, joint 3D map (Section 5.2.2).

5.2.1 Submap Exchange

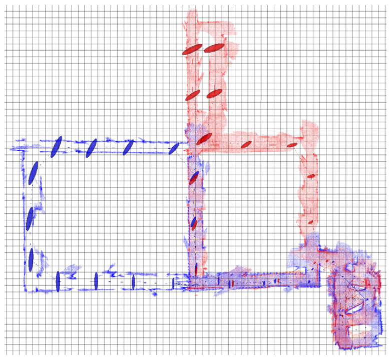
Submapping allows to reduce the high-bandwidth 3D data by aggregating it locally into partial maps of limited size. Exchanging only this compacted data between robots in a multi-robot system can save memory and bandwidth compared to keyframe-based approaches (e. g., see Leishman et al., 2013; Mohanarajah et al., 2015) that keep and potentially exchange the full 3D information at a higher sampling rate. It thus allows an efficient data exchange in the light of limited communication bandwidth.

In our systems, submaps created by an individual robot are broadcasted to all other agents once they are "finished", i. e., once the robot's next subsequent submap has been started. Thus, each submap is transferred to each robot only once. Thereby, we can avoid the network load of sending regular updates of the most recent, constantly changing, partial map. This, however, means that the most recent map data from other robots is delayed until their respective submaps are finished and transferred. In our experience, this map sharing strategy proved to be sufficient for applications such as autonomous multi-robot exploration. Typically, the parts of a robot's environment model that are most time-critical for its on-board and online decision making are its immediate surroundings. They can be perceived by its own sensors and are modeled in its own, up-to-date current submap. Nonetheless, data exchange between robots happens regularly during robot movement due to the limits on the uncertainty and accumulated driving distance for each submap, as we detailed in the previous section.

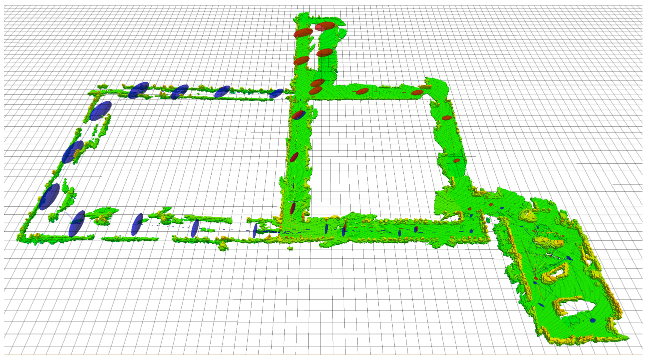
5.2.2 Global Map Composition

In order to generate joint global maps, as visualized in Figure 5.8, we merge the information of all of our submaps based on the latest graph SLAM estimates for their respective origins. This can be done either periodically, e. g., for up-to-date visualizations, or only on-demand, e. g., triggered by requests from global path and exploration planning components. The merging step requires a significant computational effort that grows with the number of submaps, as we discuss in our evaluation of the required computational resources in Section 8.4. We therefore cache the combined map and invalidate this cache only on significant changes of the respective estimated submap origins, i. e., after significant new loop closures. We exclude the latest submap of the robot performing these computations from the caching. This latest submap is still unfinished and growing, in the sense that it is constantly being modified by aggregating the most recent sensor data. Excluding this map from the cache of the same robot and merging its latest estimate into the global map allows each robot to compute a global environment model on board that is always up-to-date regarding its own sensor data.

We merge the point cloud representations by concatenating the sets of points and using a voxel-grid filter to reduce the spatial density in overlapping parts back to their



(a) *Global 3D map in its point cloud representation (colored by robot)*



(b) *Global 3D map in its probabilistic voxel grid representation (colored by height)*

Figure 5.8: *Global multi-robot map composed by merging 53 submaps created by two LRU rovers. The submap origins are indicated by their positional uncertainty ellipsoids, colored in red and blue to indicate the two robots. We describe the respective experiment and its results in detail in Section 8.3.2.*

original resolution of, in our experiments, 5 cm. For the probabilistic voxel space representation, we employ the merging algorithm developed by Jessup et al. (2015) to first align the octree grids using trilinear interpolation and then compute the sum of the log odds for each voxel, akin to the method described by Saeedi et al. (2016) for 2D grid maps. Merging two voxels by adding up their log odds and clamping the result is similar to a measurement update, as defined by Equation 5.9.[3] Using these methods, we merge all submaps in a sequential fashion into a growing global map.

5.3 Related Work and Discussion

In this section, we first discuss our local obstacle mapping in relation to other approaches from literature. We then focus on the combination of local and global mapping to create dense 3D maps.

5.3.1 Obstacle Mapping

We used the obstacle maps for navigation of our rovers in rough terrain throughout the experiments presented in Chapter 8 and 9. In addition to autonomous obstacle

[3]For more details on the algorithm to merge octree-based probabilistic voxel spaces, we refer to the master's thesis by Jessup (2013).

avoidance, we successfully employed these maps for assisted robot control via a force-feedback joystick as part of the *Kontur-2* project (Panzirsch et al., 2018): A smoothed version of the map is used to compute a virtual force that directs an operator away from obstacles in the robot's assumed path. This allows to safely steer a rover over delayed communication channels with round-trip times of up to 800 ms.

Furthermore, obstacle classification results provide valuable information for loop closure generation by identifying distinctive 3D geometry in the environment. In Section 6.1, we describe their application in our map matching algorithm. We present evaluation results showing the impact of our negative step detection and stereo error-adaptive classification for re-localization in Section 8.1.1. Our rovers achieved final position deviations below 0.08 % of their full trajectories for both narrow- and wide-angle camera setups with an RBPF-based SLAM system for 2D mapping that we used for these early experiments prior to the design of our full 6D SLAM framework.

In literature, many different types of maps have been introduced in order to facilitate path planning and local obstacle avoidance for ground-based robots. Most of them are based on some kind of grids with statistically independent cells. Elevation maps (Kweon and Kanade, 1992) have been proposed to model uneven terrain. They were extended by Triebel et al. (2006) to represent multiple layers per cell as well as vertical objects and by Oberländer et al. (2014) to support multiple levels of detail. The latter encode additional surface properties, including the variance of its points as an indicator for roughness. Similarly, Marks et al. (2009) argue that elevation maps alone are unsuitable to capture the notion of traversability in areas, in which the variance in height within a cell is more relevant than the differences between neighboring cells, for example in outdoor scenarios when traversing tall grass. They thus propose a map where each cell represents a distribution over elevation variances instead of mean elevations and demonstrate its applicability to rough terrain scenarios. While all of these types of elevation-based maps share the desirable property of being robot-agnostic, they cannot be used to model obstacles like negative edges, such as steep cliffs viewed from above, as discussed in Section 5.1.1. When, for example, a cliff itself is not observable, there is no elevation or elevation variance value to be encoded in the cells of such a map. Thus, we follow a different approach by encoding obstacle classifications directly into local maps used for obstacle avoidance. While this requires to encode values that are specific to the locomotion capabilities of particular robot systems, it allows us to represent obstacles like the aforementioned negative edges. In addition, the camera viewpoint and thus aspects like the stereo reconstruction error can be taken into account during map creation. Stelzer et al. (2012) propose further criteria for the classification of slope and roughness, some of which we adapted in our previous work (Brand et al., 2014).

They can be important for obstacle avoidance, in particular for complex locomotion systems like a six-legged crawler (Stelzer et al., 2012; Stelzer, 2016). Their discussion, however, would go beyond the scope of this work, which focuses on the utility of binary obstacle classifications to support loop closure detection for global mapping. In order to limit the stereo reconstruction error to their local map resolution, Stelzer (2016) severely limits the maximum distance of valid measurements. In contrast to her approach, our obstacle classification takes the potential error from stereo reconstruction into account. Therefore, we are able to support larger view distances, as our adaptive threshold allows a robot to navigate close to obstacles despite possible imprecisions in the far range of its local terrain classification map. In general, encoding information about traversability of terrain into maps has the additional advantage that the traversability only needs to be updated locally on map changes and is not required to be re-evaluated in each path planning step, as it would be the case for simple elevation-based maps. Despite having robot-specific data in the local maps, our submaps that are shared between robots also contain robot-agnostic 3D point cloud and probabilistic 3D voxel-grid representations. These are sufficient for global exploration goal planning, as we demonstrated in our multi-robot exploration experiments presented in Section 9.3.

5.3.2 Local and Global Mapping

A distinction between local and global maps allows to satisfy the different requirements posed by their respective applications. Local maps typically require high update rates to ensure a sufficiently fast obstacle avoidance based on an interpretation of a robot's immediate surroundings. Global maps, in contrast, are used for long-range navigation, long-term planning, and multi-robot coordination, tasks that are often executed online but at lower rates. Stelzer (2016) combines local terrain classification maps with a sparse global topological model for long-range route following. While we make a similar general distinction, our global maps serve a different purpose: For the autonomous exploration and modeling of previously unknown areas, robots require a dense metric representation of the environment that allows to explicitly represent the still unknown space. However, in future work, an additional topological layer similar to Stelzer (2016) could be used for a resource-efficient navigation between workspaces or other areas of interest in scenarios where it is not necessary to build and store 3D environment models for the paths between them.

The creation of submaps is a widely-used technique to locally aggregate high-bandwidth or high-frequency sensor data into partial maps of limited size (Reid and Bräunl, 2011; Vidal-Calleja et al., 2011). This approach allows to keep more information through the aggregation of sensor data than key-frame based mapping techniques (Leishman et al., 2013; Mohanarajah et al., 2015), which apply sparse temporal sampling on, for example, high-bandwidth depth data. The local aggregation into partial maps not only reduces the memory and computational requirements of further processing and optimization steps but is particularly important to allow an efficient exchange of environment model data in a multi-robot system. In our experimental evaluation presented in Section 8.4, we demonstrate a significant reduction of the bandwidth required to share aggregated submaps between robots (58 KB/s) compared to transferring raw navigation camera image data (38.75 MB/s). The 3D data itself is currently only filtered and compacted during submap generation as part of the aggregation process and then serialized for transfer. If necessary, additional lossless or lossy compressions of the data would allow to further reduce the required communication bandwidth. In addition, the exchange of submaps as well as their composition into a global map could be limited to an application-dependent area of interest, as proposed by Koch and Lacroix (2016).

Similar to the 2D and 2.5D maps discussed above, the choice of a suitable representation of 3D data in submaps always involves an application-dependent trade-off between the type of information being represented, map accuracy, and the required computational resources. As 3D data is typically challenging w. r. t. memory and processing demands, the latter aspect becomes important in order to allow for on-board and online creation of maps at a frequency that is sufficient for the autonomous operation of robot systems. For our use case, the autonomous exploration with mobile robots, we decided for a combination of 3D point clouds and a probabilistic voxel space. In order to allow sufficiently fast online updates, our voxel space uses a simplified sensor model by casting a ray into 3D space that is assumed to hit at most a single occupied voxel. In particular for stereo camera systems, more complex sensor models could increase map accuracy, for example by taking the pixel-wise uncertainty of each depth measurement in 3D space into account. They, however, are computationally more demanding and, while useful for applications like the creation of precise 3D models of small-scale scenes, not necessarily required to support the autonomous robotic exploration of large areas, in particular as we already consider the stereo error during our local terrain classification used for short-range navigation.

In order to create a global map, we sequentially merge the point cloud and probabilistic voxel space representations of all submaps. A clamping of the voxel probabilities, as implemented in the "OctoMap" library (Hornung et al., 2013) and discussed in Section 5.1.2, can make the results dependent on the merge order. Furthermore, the merge algorithm neither takes the age of the data nor the number of raw sensor observations for each voxel into account. Thus, for an application in semi-static or dynamic environments, it might need to be extended by introducing weights in order to give a higher influence to more recent observations. Additional improvements on the merge results might be gained by adding a relative entropy filter, as proposed by Yue et al. (2016), that decides for each of voxel to either trust both or only one of the input maps based on the Kullback-Leibler (KL) divergence of their occupancy probability distributions. In future work, submaps could be merged after successful matching, similar to the work by Mohanarajah et al. (2015), and deprecated submaps removed when the same area has been exhaustively mapped again more recently.

Chapter **six**

Online Loop Closure Generation

In this chapter, we present our approach to the multi-robot data association problem as part of the front-end of our SLAM framework. It consists of two major techniques to generate intra- and inter-robot loop closures in a multi-robot SLAM graph: First, the pairwise matching of submaps between one or multiple robots to estimate their relative transformation (Section 6.1); Second, the detection and pose estimation of other robots based on visual markers, with focus on an estimation of the respective measurement uncertainty (Section 6.2).

6.1 Submap Matching

As argued by Vidal-Calleja et al. (2011), the 3D geometry of the environment is its only intrinsic characteristic that can be relied on for data association under the assumption of different sensors or viewpoints among robots in a heterogeneous multi-robot team. Following this line of reasoning, we perform a pairwise matching of the 3D structure of submaps in order to compute relative transformations accompanied by uncertainty estimates between their origins and thus generate intra- as well as inter-robot loop closure constraints.

This section is based on our journal article by Schuster et al. (2018), in which we improved and extended upon our map matching concepts that we introduced in our conference paper by Brand et al. (2015) and first applied to multi-robot systems in our publication by Schuster et al. (2015).[1]

In particular, we improved our matching method with the goal of maximizing the resulting number of loop closure constraints while maintaining their level of accuracy. Our submap matching works on aggregated stereo depth data from noisy sensors with limited field of view and can thus generate only small numbers of loop closure hypotheses compared to, for example, image feature-based systems. While our matcher is designed to minimize the number of erroneous data associations, they nonetheless are impossible to rule out. Thus, increasing the total number of loop closure constraints is important for the graph SLAM to be able to filter such false positives as outliers, or at least compensate for their influence in order to increase the robustness of global estimation. To achieve this, we exploit the properties of partial frame switches in our local reference filters to reduce the dimensionality of the matching problem. As described in Section 4.1, we switch the local frame of reference in the filter only with respect to the unobservable states x, y, z and yaw. Thus, we can define the origins of all submaps to be aligned to the observable gravity vector, i. e., for all of them $roll = 0$ and $pitch = 0$. This does neither restrict the content of submaps, being able to represent arbitrary 3D geometries, nor the applicability of our system to aerial robots, as the observable states of each robot, including its $roll$ and $pitch$ angles, are continuously estimated within the filter. We, however, are able to reduce the number of dimensions of the map matching optimization from six to four by solely estimating a relative transformation for the four remaining, unobservable degrees of freedom. Constraining the matcher's optimization steps early on to valid hypotheses w. r. t. $roll$ and $pitch$ leads to an increased number of valid matches, as we discuss in Section 6.1.2 and demonstrate in our experimental evaluation in Section 8.1.2.

In Figure 6.1, we present the architecture of our submap matching process. It consists of two parallel threads, one to filter the input data and generate pairs of potentially matching submaps, and another to perform the matching itself. As the processing of different pairs of submaps is independent from each other, it would be easy to further parallelize this process for multi- and many-core architectures. We already implemented a similar type of parallelization across machines by executing matcher processes on multiple robots in parallel. They share information about their results in order to avoid duplicate work as well as double-counting of successful matches. In general, we run the matcher itself as a background process with low priority that can

[1]The first version of the map matching algorithm itself, as published by Brand et al. (2015), has been developed by Christoph Brand in close collaboration with me and later on revised and extended by me to the algorithm presented in the article by Schuster et al. (2018) and here.

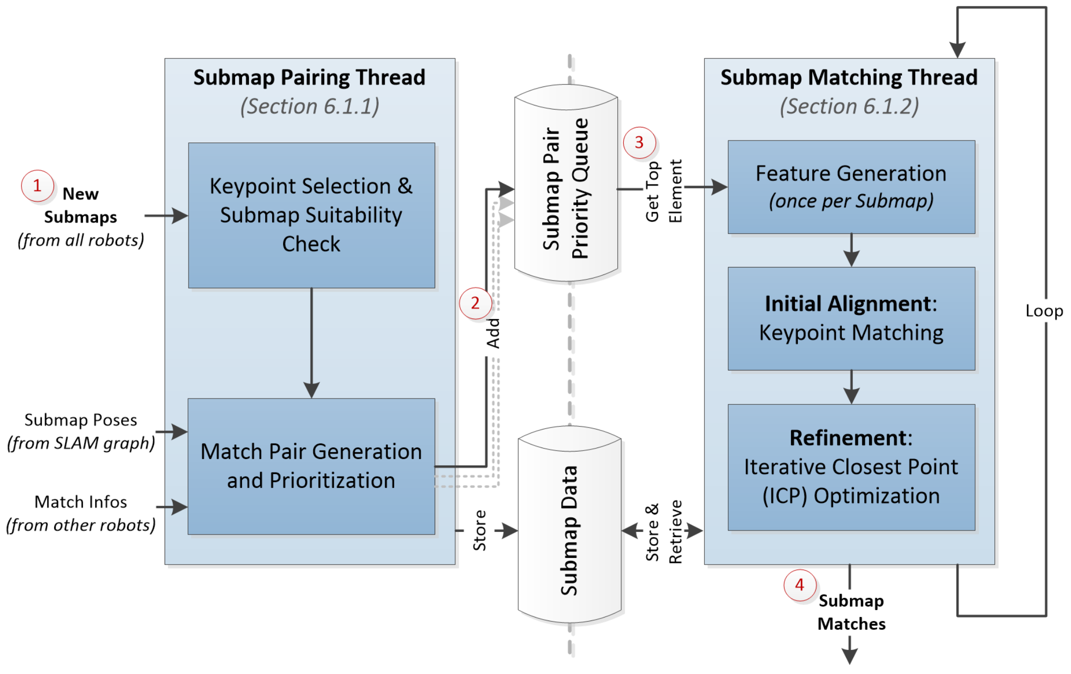


Figure 6.1: *Submap matcher architecture with two threads for parallel execution of pairing and matching (numbers 1-4 indicate typical order of main data flow)*

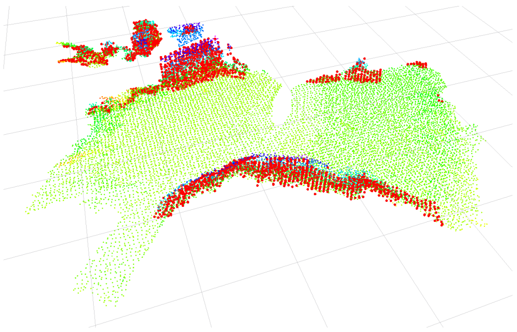
acquire all free processing resources on demand. We will explain the individual steps of pairing and matching in the following sections.

6.1.1 Pairing of Submaps

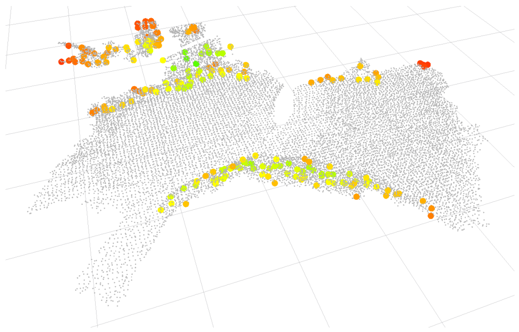
The submap pairing thread, visualized in the left part of Figure 6.1, receives the newly finished submaps from all the robots, the latest submap pose estimates from graph optimization, as well as information about matches that have been checked by submap matcher processes running on other robots in a multi-robot system. For each submap, we first precompute data structures required for the matching itself, such as keypoint data, and filter out unsuitable submaps. Then we use the available data to pair submaps by selecting and ranking candidates that could possibly match.

Keypoint Selection

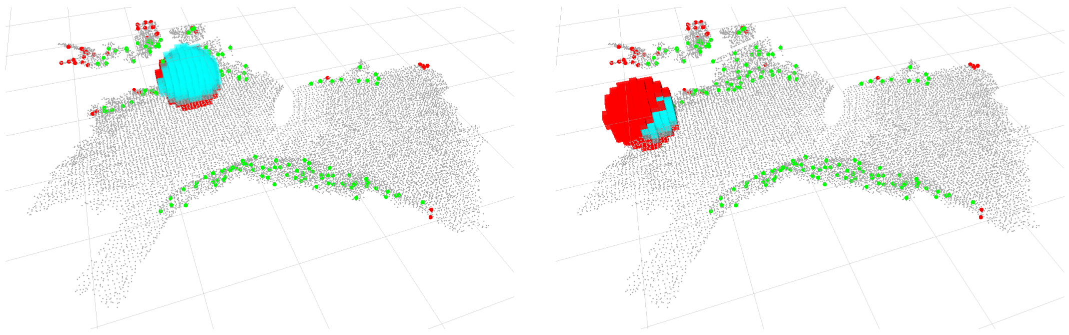
For 3D feature matching, we first select suitable keypoints based on our precomputed obstacle classification in the point clouds, as shown in Figure 6.2(a). The environmental features encoded in our binary obstacle classification typically represent distinctive geometrical landmarks that provide valuable information to support



(a) *Height-colored point cloud with binary obstacle classification (red)*



(b) *Keypoints ranked from good (green) to bad (red)*



(c) *Keypoints classified as inliers (green) and outliers (red) for a neighborhood radius of* 0.5 m *and a keypoint filtering threshold of* 0.7. *Blue and red voxels denote known and unknown space respectively, showing an example of the discretized spherical neighborhood of an inlier (left) and a filtered-out outlier keypoint (right).*

Figure 6.2: *Visualization of the keypoint filtering process for the submap depicted in Figure 5.5: point cloud with obstacle points (a) and sub-sampled keypoints that are ranked (b) and filtered (c) according to the amount of knowledge about their local neighborhood*

data association when recognizing previously visited places.[2] In order to reduce the computational effort of the subsequent processing steps, we do not consider all the obstacle points, but use them to compute a reduced set of keypoints. For this, we apply a 3D voxel-grid filter with a voxel size of 0.1 m. It is similar to the type of filter used during submap generation, which we described in Section 5.1.2, but uses a coarser resolution. In Figure 6.2, we present an exemplary submap with its obstacle points as well as the keypoints computed from them.

Next, we filter out keypoints that are located too close to unknown regions. As feature vectors describe local spherical neighborhoods around the keypoints (see below for details), including an excessive amount of unknown space will lead to incorrect and misleading descriptors that are unsuitable for matching. Whereas the keypoint descriptors themselves are defined on the point cloud of each submap, for

[2]We demonstrated the suitability of our obstacle classification for loop closure generation in our evaluations presented in Section 8.1.1.

keypoint filtering, we exploit the information available in its probabilistic voxel space representation: It encodes an explicit distinction between known and unknown space. Around each keypoint, we discretize the local spherical neighborhood in the voxel space with a radius equal to the keypoint descriptor radius used later on. We then apply a threshold of 0.8 on the ratio of known to unknown voxels in that space to filter out keypoints that misleadingly would encode large amounts of unknown space as free space. In Figure 6.2, we visualized the steps of the keypoint filtering process. The keypoint filtering based on the probabilistic voxel map is an improvement of our algorithm going beyond the work presented by Schuster et al. (2018). While first preliminary tests indicate a positive impact on match accuracy, it has not yet been implemented for most of the experimental evaluations presented in this thesis.

The keypoints for each submap are independent of all other submaps. Thus, we compute them just once and store them together with the submap data for later use during the pairwise matching, as we indicated in Figure 6.1

Submap Suitability Check

We determine the suitability of each submap for map matching by checking its size and descriptiveness. As a first step, prior to the keypoint computation described above, we filter out submaps as being too small or uninformative if their point clouds contain less than 1000 points in total or less than 100 obstacle points.

After selecting keypoints, we check their distribution as a measure for the descriptiveness of a submap. As we argued in Section 5.1.1, arrangements of obstacles represent informative and unambiguous geometrical features in both indoor and outdoor environments, with the exception of very self-similar man-made structures. We aim to distinguish these valuable arrangements of obstacles from simple structures like straight walls or ambiguous ones like single stones. For this, we check the distribution of the keypoints, which are located on the obstacles. We fit a 3D line model to the set of keypoints using Random Sample Consensus (RANSAC) and count the number of points that are sufficiently far away from the fitted line in the xy-plane.[3] If the count is below a threshold, we remove the submap from the matching process, as its distribution of keypoints might lead to ambiguous matching results w. r. t. one or several degrees of freedom. As empirically determined thresholds, we require at least 10 keypoints to be farther away than 0.5 m to match maps created by our rover systems. By this, we replace the heuristic based on bounding boxes in the xy-plane described by Brand et al. (2015) with a more general approach. These suitability

[3]Performing checks in the xy-plane is a heuristic suitable for ground-based robots or aerial robots flying at low altitudes so that they can map the ground. It could be adapted or removed when mapping and matching vertical structures.

checks allow us to dismiss too small or ambiguous submaps before even including them into the matching process.

Match Pair Generation and Prioritization

In the submap pairing thread, we select potentially matching pairs of submaps. As for n submaps there are $\frac{n!}{2(n-2)!}$ pairwise combinations, for large n it would be infeasible to try to match all of them. In addition, a pre-selection of potential matches allows us to reduce the number of false positives by performing sanity checks against the current SLAM estimates early on.

Our central criterion to determine if two submaps s_i and s_j can match is based on the overlap of their axis-aligned xy-bounding boxes, given the most recent graph optimization estimates for the poses of their origins. We limit these bounding boxes to include only points classified as obstacles, as these define the keypoints for geometric feature matching. $overlap_t$ denotes the amount of overlap of the two bounding boxes in the respective dimension $t \in \{x, y\}$. When computing a value $overlap'_{x,y}$ for the potentially overlapping two-dimensional area of the submaps, we account for the uncertainties in the submap poses as follows:

$$overlap'_t = overlap_t + 2 \cdot \Delta\sigma_t \quad \text{for } t \in \{x, y\} \tag{6.1}$$

$$overlap'_{x,y} = overlap'_x \cdot overlap'_y \tag{6.2}$$

Thereby $\Delta\sigma_t$ denotes the relative positional uncertainty between the poses of two submap origins in terms of standard deviations in the respective dimension $t \in \{x, y, z\}$. It can be obtained from the SLAM graph by expressing the uncertainty of one of the two submap origins in the pose of the other. While we plan to implement this in the future, we used a rough approximation for $\Delta\sigma_t$ in the experiments presented in this thesis: $\Delta\sigma_t \approx \sqrt{|\Sigma^{s_i}_{t,t} - \Sigma^{s_j}_{t,t}|}$ with $\Sigma^{s_i}_{t,t}$ and $\Sigma^{s_j}_{t,t}$ referring to the variances in the dimension $t \in \{x, y, z\}$ of the poses of the origins of s_i and s_j as estimated by our graph SLAM. Although this heuristic is fast and easy to compute, we are aware that it fails in certain cases of well-connected and multi-robot graphs. The resulting approximation errors, however, can in the worst case lead to an over-filtering of potential matches.

We require $overlap'_{x,y} > 2\,\mathrm{m}^2$ in order for a pair of submaps to be added to the match queue. In addition, we exclude pairs consisting of successive submaps from the same robot r, i. e., $\left(s^r_i,\ s^r_{i+1}\right)$, as the error of the filter estimates between their origins also affects the accumulated 3D data of s_i. Thus, we would not expect any significant improvement from such a match in comparison to the transformation that is estimated by the local reference filter and added to the SLAM graph anyway. Furthermore, we remove pairs of submaps for which the estimated pose uncertainty is below the precision of the matcher. We approximate the latter by the stereo error, discussed in Section 5.1.1, which grows quadratically with the distance to the cameras. As shown in Figure 5.2, it is significantly large for our camera setups. According to Equation 5.2c, the distance error for a 3D point centered in the camera image is 0.12 m at a view distance of $p_z = 4\,\mathrm{m}$ for the LRU's stereo camera setup with its focal length of $f = 1080\,\mathrm{px}$, baseline of $b = 0.09\,\mathrm{m}$ and a pixel error of $\Delta p = 0.5\,\mathrm{px}$. By checking the following constraint, we compare the approximated mean translational standard deviation of the relative transformation estimate between the submaps to the propagated stereo error as stated in Equation 5.2c:

$$\frac{\sum_{t \in \{x,y,z\}} \Delta\sigma_t}{3} < \Delta p \, \frac{p_z^2}{f\, b} \sqrt{2} \tag{6.3}$$

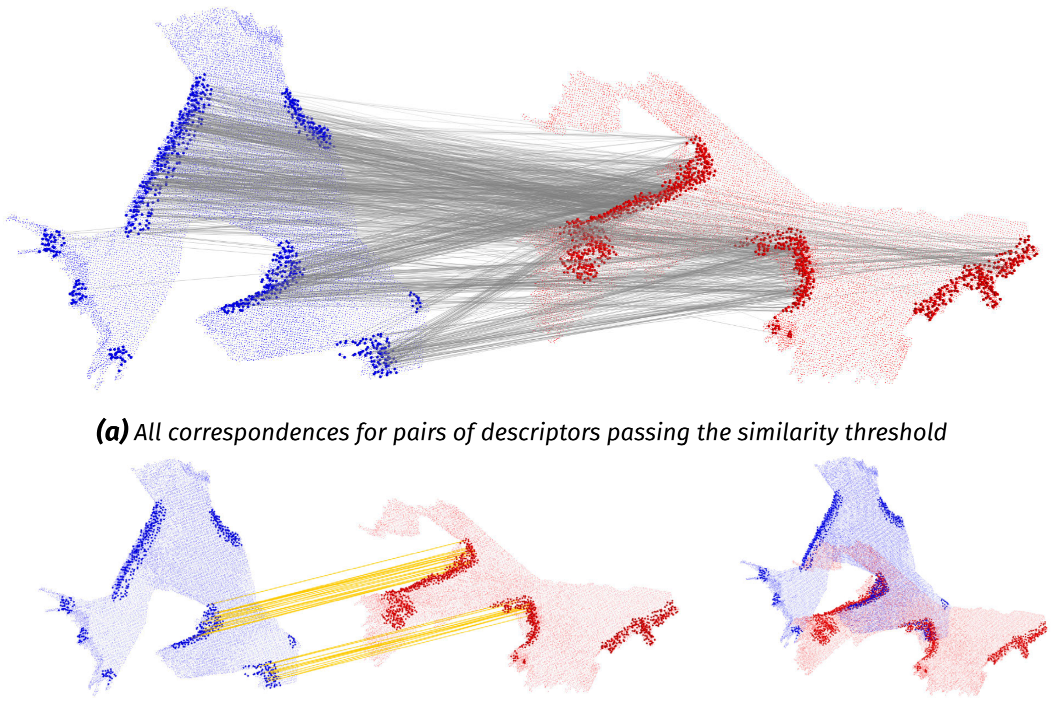
We discard each pair of submaps that fulfills the constraint stated in Equation 6.3, as we cannot expect its match to significantly improve on the current estimate for the relative transformation between its submaps. After these checks, we add the remaining pairs of submaps to a priority working queue, prioritizing them in order to try the most promising pairs first. For this, we rank them according to:

$$score = \alpha \cdot overlap'_{x,y} + \sum_{t \in \{x,y,z\}} \Delta\sigma_t \tag{6.4}$$

The score consists of two parts, the first being a heuristic for the expected probability to match and the second for the expected impact of the match on global optimization. In our multi-robot experiments, we weighed them with an empirically determined $\alpha = 0.125$ to make a trade-off between the two criteria.

6.1.2 Pairwise Matching of Submaps

The submap matching thread, visualized in the right part of Figure 6.1, continuously loops through all pairs of submaps queued for matching. We based the pairwise matching between submaps primarily on their point cloud information and follow a two-step approach: First, we compute potential initial alignments based on 3D feature matching and select the best model. Second, we perform an Iterative Closest Point (ICP) step for refinement and match uncertainty estimation. In Figure 6.3, we show a submap match from our multi-robot experiment described in Section 8.3.2,



(a) *All correspondences for pairs of descriptors passing the similarity threshold*

(b) *Filtered correspondences after Hough3D voting for the initial alignment between the two submaps*

(c) *Final transformation after refinement*

Figure 6.3: *Submap match between maps from LRU1 (blue) and LRU2 (red) from our multi-robot experiment described in Section 8.3.2. The larger points indicate keypoints located on obstacles (in this case large artificial rocks, similar to those depicted in Figure 8.7(b)), used to compute correspondences for initial alignment.*

depicting the keypoints and correspondences used during initial alignment as well as the final transformation between the submaps after refinement. In the remainder of this section, we will explain the individual matching steps in detail.

Feature Generation

As the first step of the submap matching process, we retrieve the top element of the submap pair priority queue. We compute 3D features for the keypoints of both submaps and store them for use in future additional match attempts that include either one of the two submaps. Delaying the feature computation to the matching thread avoids unnecessary computational effort in case a submap is never included in any match attempt.

In order to characterize geometric features in the environment, we decided to employ Color-SHOT (CSHOT) (Tombari et al., 2011) feature descriptors, an extension to Signature of Histograms of Orientations (SHOT) (Tombari et al., 2010) descriptors, as they are robust w. r. t. to noise and clutter. They characterize the local spherical neighborhood around a keypoint based on a signature of histograms. A local 3D reference frame for each keypoint makes the descriptor invariant to rotations of the input data. First, the spherical neighborhood of a keypoint is discretized into a set of volumes based on an isotropic spherical grid that is aligned with the local reference frame. For each of these volumes, a histogram of the angles of its point normals w. r. t. the local reference frame is computed. The descriptor then is represented by a concatenation, i. e., signature, of these histograms. It is normalized to have a L^2 norm equal to 1. In our experiments, we used an empirically determined radius of $r = 0.2\,\mathrm{m}$ for the computation of the point normals, of the local reference frames, as well as of the descriptor histograms themselves. We selected this value in order to base the descriptors on a sufficiently large volume in relation to the expected noise and reconstruction errors from our stereo camera systems. The CSHOT extension additionally includes a signature of histograms for texture information, which is available on our robots equipped with stereo cameras.[4] However, on our rovers, we solely employed greyscale cameras, whereas the CSHOT feature matching might also benefit from color information. It is a topic for future work to analyze the effect of the texture information, in particular in heterogeneous multi-robot settings involving different camera setups.

Initial Alignment: Keypoint Matching

We compute correspondences between the keypoints in both submaps by comparing their CSHOT descriptors. For each feature in one submap, we select the three most similar features from the other submap and vice versa, filtering duplicates and pairs of feature descriptors that do not pass a similarity threshold of 0.5 in their range of valid values from $[0, 1]$. The feature descriptors are compared based on their L^2 distance. As we only consider a small number of best matches, our algorithm is not very sensitive to the similarity threshold, which we selected as a conservative choice from the interval of $[0.4, 0.8]$ that we empirically determined to yield good results. Although we employ k-d trees to speed up this high-dimensional search, it remains one of the computationally most expensive steps in our pipeline.

We then cluster the correspondences into match transformations through Hough3D voting (Tombari and Di Stefano, 2010), which allows multiple hypotheses to be found and has been shown to be effective for stereo setups providing noisy 3D data. Due to the asymmetry introduced by its separation of translation and rotation, we compute the Hough3D voting two times for each pair of submaps $(s_i,\ s_j)$, from s_i to s_j and vice versa. This allows us to obtain more hypotheses and make them

[4]In our experiments, we used CSHOT features with a total descriptor length of 1344, as provided by the open source Point Cloud Library (PCL) 1.7.2 library (Rusu and Cousins, 2011).

independent of the order of submaps in the pair. Basically, each correspondence votes for a translation hypothesis in the 3D Hough space. After this clustering step, a RANSAC registration method is used on each cluster in order to determine the most likely transformation, including a 3D orientation, associated with it. Compared to a closed-form solution using Singular Value Decomposition (SVD) (Umeyama, 1991), RANSAC is more robust to large outliers that are to be expected to occur due to noise and potential symmetries in the point cloud data of the submaps.

As mentioned before, the *roll* and *pitch* angles of the coordinate frames of both submap origins are zero. As both angles are well observable in our local reference filter, we expect the size of their estimation errors to be negligible, in particular compared to the accuracy of the map matching that is limited by the noise and resolution of our stereo-based point cloud data. Thus, we do not need to take their uncertainty estimates into account for the map matching pipeline and can limit the RANSAC optimization to x, y, z and the *yaw* angle. For this, we adapted the Hough3D implementation from the open source Point Cloud Library (PCL) 1.7.2 (Rusu and Cousins, 2011), in particular replacing its RANSAC step that uses a SVD to compute transformation samples from three points in each RANSAC iteration. Instead, we randomly select two points $\boldsymbol{p}_i$ and $\boldsymbol{q}_i$ from the cluster of each of the two submaps $i \in \{0, 1\}$. We then use the direction of the gravity vector $\boldsymbol{z}$ as an additional constraint to define a transformation between the two pairs of points, thereby enforcing $roll = pitch = 0$:

$$\left.\begin{aligned} \boldsymbol{z} &= (0,0,1) \\ \Delta\boldsymbol{p}_i &= \boldsymbol{q}_i - \boldsymbol{p}_i \\ \Delta\boldsymbol{p}'_i &= \Delta\boldsymbol{p}_i - \left(\Delta\boldsymbol{p}_i \cdot \boldsymbol{z}\right) \cdot \boldsymbol{z} \\ \boldsymbol{R}_i &= \left(\left(\Delta\boldsymbol{p}'_i\right)^T,\ \left(\Delta\boldsymbol{p}'_i \times \boldsymbol{z}\right)^T,\ \boldsymbol{z}^T\right)^T \end{aligned}\right\} \text{for } i \in \{0,1\} \tag{6.5}$$

We compute the vector $\Delta\boldsymbol{p}_i$ between the points and then adapt it to $\Delta\boldsymbol{p}'_i$ so that it is orthogonal to $\boldsymbol{z}$. The cross product of $\Delta\boldsymbol{p}'_i$ and $\boldsymbol{z}$ gives us a third orthonormal vector, defining the rotation matrix $\boldsymbol{R}_i$. With this and the mean of the points of each submap, $\bar{\boldsymbol{p}}_i = \left(\boldsymbol{p}_i + \boldsymbol{q}_i\right) / 2$ for $i \in \{0, 1\}$, we compute the transformation hypothesis between the submaps with rotation matrix $\boldsymbol{R}$ and translation vector $\boldsymbol{t}$:

$$\boldsymbol{R} = \boldsymbol{R}_0 \cdot (\boldsymbol{R}_1)^T \tag{6.6}$$

$$\boldsymbol{t} = \bar{\boldsymbol{p}}_0 - \boldsymbol{R} \cdot \bar{\boldsymbol{p}}_1 \tag{6.7}$$

The next step is to filter out all hypotheses that exceed the $2\Delta\sigma_t$ bounds of the approximated uncertainty between both submaps in any dimension $t \in \{x, y, z\}$. In addition, for each submap, we check the distribution of matching keypoints for each hypothesis separately by fitting a line model, similar to the submap suitability check described in Section 6.1.1. Thereby, we dismiss matches based on features that are not well distributed and thus might be ambiguous w. r. t. one or more degrees of freedom. From the remaining hypotheses, we select the one with the largest number of correspondences for the subsequent refinement step.

Refinement: ICP Optimization

Next, we perform an ICP optimization on the 3D point clouds of the two submaps. As this method can be sensitive to local minima (Mendes et al., 2016), it requires a close-enough initial alignment, which we gain from the previous steps. For the ICP, we use the full point cloud, as it has a higher resolution than the keypoints and includes non-obstacle parts of the map, like traversable terrain, that can give valuable information, in particular for a correct alignment of the ground planes. While such areas lack robust 3D features for initial alignment, they work well for an ICP.

We also restrict the refinement step to 4D by removing the *roll* and *pitch* angles from the optimization problem. For this, we adapted the Levenberg-Marquardt-based ICP algorithm from the open source PCL 1.7.2 library (Rusu and Cousins, 2011) to restrict *roll* and *pitch* of its internal transformation estimates to zero, before using them to evaluate the cost function in its Levenberg-Marquardt optimization steps. Similar to our adapted Hough3D voting, the early incorporation of prior knowledge to reduce the dimensionality of the optimization directs the optimizer towards the correct solution. The alternative, full 6D steps as used in our previous work (Brand et al., 2015), runs the risk of first converging to implausible solutions that afterwards get eliminated in post-processing by applying restrictions that have been unknown to the RANSAC and ICP algorithms themselves. We demonstrate in our experimental evaluation in Section 8.1.2 that our 4D matching thus yields a larger number of map matches, i. e., loop closures, after applying the same plausibility checks to filter potentially erroneous data associations.

As a final step after the ICP, we once again check whether the resulting transformation is within the approximated $2\Delta\sigma_t$ error bounds of the latest SLAM estimates in all dimensions $t \in \{x, y, z\}$. In order to weight the map match constraints against each other and against other estimates, our graph-based global optimization requires an uncertainty measure for the map match transformations. We approximate this based on the Root-Mean-Square Error (RMSE) in point-to-point differences computed during the ICP in the final alignment step. In Figure 6.4 we present a sketch showing integration of a match of two overlapping submaps into the SLAM graph.

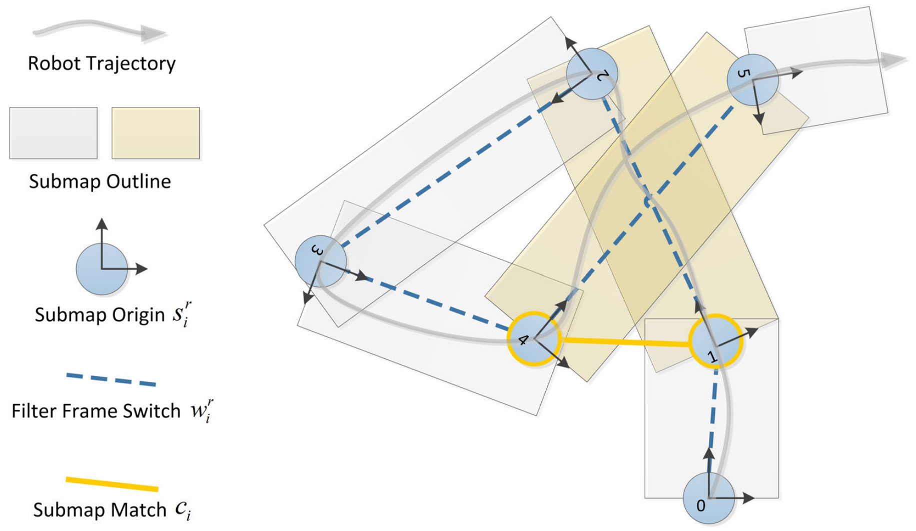


Figure 6.4: *Schematic of SLAM graph with submap origins and submap bounding boxes. The overlapping highlighted rectangles represent submaps that match, resulting in a loop closure constraint that is added to the graph.*

6.1.3 Related Work and Discussion

Intra- and inter-robot loop closures constitute the basis for global optimization in multi-robot SLAM. Thus, many different approaches exist in literature, using various types of sensor modalities like vision, sound, radio waves, etc. As all of our robot systems for planetary exploration are equipped with stereo cameras, we restrict the discussion in this work to vision-based methods. Existing approaches for loop closure generation in visual SLAM include the matching of image features as identifiable landmarks (Endres et al., 2012), the matching of visual key-frames (Leishman et al., 2013), and the registration of depth data through Iterative Closest Point (ICP) techniques (Newcombe et al., 2011).

As discussed in Chapter 5, the creation and exchange of submaps has the advantage of allowing an efficient exchange of data between roots in a multi-robot system with low-bandwidth communication channels. Submap matching uses the intrinsic information of these submaps for multi-robot data association. As we discuss in Section 8.2.1, by matching 3D submaps, our 6D SLAM framework significantly outperformed our preceding 3D SLAM system that used a 2D scan matching approach to generate loop closures (Brand et al., 2014). In a direct comparison, we could demonstrate an improvement on the mean 2D position error of 27 % and 50 % in outdoor and indoor experiments respectively. Furthermore, our experimental evaluation in Section 8.1.2 shows that our novel reduction of the dimensionality of the matching problem from 6D to 4D increases the number of successful matches by 40 % on average. In previous related work, submap matching has been successfully employed for 2D (Reid and Bräunl, 2011) and 2.5D maps (Forster et al., 2013) using a brute-force correlation search. This, however, requires the computational resources of a Graphics Processing Unit (GPU), even when limiting the search space to a discretization of $\mathbb{R}^3$ or $\mathbb{R}^2$ respectively. In contrast, we propose a purely Central Processing Unit (CPU)-based submap matching algorithm for 3D maps that runs independently of the availability of GPUs on the robots.

For 3D mapping, Labbé and Michaud (2014) as well as Mohanarajah et al. (2015) employ graph SLAM with Speeded Up Robust Features (SURF) features for loop closure generation. Such image-based features, however, are not robust to changes in viewpoint and illumination and thus not well suited for heterogeneous multi-robot teams (Forster et al., 2013). ICP algorithms work well for frame-to-frame registration based on 3D geometry (Newcombe et al., 2011; Nüchter et al., 2007). They can be employed for map matching when a close initial estimate is available (Mendes et al., 2016), e. g., through known relative start positions in a multi-robot scenario (Michael et al., 2012). As ICP optimization easily becomes trapped in local minima, it is less suitable as a first step for long-range global loop closure detection, in particular on noisy stereo data. It, however, can improve precision as a refinement step if provided with a good initial alignment. For this, 3D feature descriptors (Li and Guskov, 2005) have become popular in order to find correspondences between point clouds, a central challenge being the selection and description of robust geometric features (Yousif et al., 2014). In our SLAM system, we employ local obstacle maps for keypoint pre-selection, as in our publication by Brand et al. (2014), we have shown that they yield robust geometric landmarks for both indoor and outdoor scenarios. From the existing multitude of 3D feature descriptors, we chose CSHOT (Tombari et al., 2011) as it exhibits a good trade-off between performance and computational complexity (Alexandre, 2012).

It is important to note that our submap matching method for rough-terrain outdoor scenarios does not make any assumptions about specific (man-made) structures in the environment like the existence of straight lines, as required by other mapping methods designed for structured indoor (Carpin, 2008) or semi-structured outdoor environments (Vidal-Calleja et al., 2011). The only exception regarding our map matcher is a requirement for sufficiently discriminate 3D geometry that can be classified as obstacles. This has the advantage of allowing our system to use a pre-selection of suitable 3D structures for feature matching that is already available, as it needs to be computed for local obstacle avoidance anyway. The downsides, however, are that it can lead to a lack of features in rough but traversable terrain and that the obstacle classification is specific to the maneuverability of a particular type of robot. The latter can be an issue for matching maps between systems in heterogeneous multi-robot teams, in particular those that feature diverse types of locomotion capabilities like driving, crawling, and flying. Thus, we propose to look into alternative robot-independent keypoint selection criteria as an important topic for future research.

For map matching, we assume to have at least some rough initial pose and pose uncertainty estimate between the submaps to be matched. In multi-robot setups, these stem from visual robot detections or the observation of the same artificial landmark, both of which can connect the measurements and estimates from different robots in the SLAM graph. This could be extended by integrating additional sensors with different modalities like, for example, Ultra-Wide Band (UWB) radio devices that use time-of-flight measurements to provide distance estimates between robots. The resulting rough hypotheses of inter-robot transformations can then serve as initial guesses for the inter-robot map matching process. Note that an initial transformation is not required to be known beforehand when deploying a multi-robot system. The inter-robot map matching is simply delayed until a connection between the estimates of the robots is first available in the SLAM graph and can from there on also process the previously acquired data. Furthermore, due to our multi-step pipeline, the initial guess is not required to be as precise as in methods that merge multi-robot maps with ICP techniques alone (see, e. g., Michael et al., 2012). We considered to match all possible pairs of submaps between robots to loosen this restriction, but this would mean a high computational effort of performing up to $n \times m$ matches for only two robots with n and m submaps respectively, not scaling well to larger setups. In addition, we do not only restrict the input set for the map matcher but also base several consistency checks on the uncertainty bounds of the initial estimate that serve as outlier filters for the matcher result. Dong et al. (2015) approach the initial multi-robot pose estimation for 2D LIDAR maps by generating a large set of inter-robot transformation hypotheses through scan matches that they then classify into a set of inliers and outliers. This is based on the observation that inliers form clusters in the space of transformations while outliers are scattered, similar to the assumption underlying the Hough3D clustering in our system. While this might be an interesting approach for future work, a key difference to our 3D map matcher is that generating hypotheses, i. e., matches, in our case is computationally more expensive than for 2D laser scans. In addition, our aggregation into submaps leads to a significantly lower number of potential hypotheses we could generate, which might not be sufficient for a clustering-based approach.

6.2 Marker-Based Robot Detection

Visual fiducial marks on some or all robots in a multi-robot team allow them to detect each other in their camera images and to estimate the respective relative transformations in 3D space. These estimates then serve as loop closure constraints for global optimization. As we focus on camera-based navigation, all of our robots are already equipped with the appropriate sensors, i. e., cameras, anyway. While many sophisticated methods for general object detection and localization in camera and depth data exist, marker-based techniques typically are more robust, have a higher accuracy, and require less computational resources. These considerations make them a suitable choice in the context of planetary exploration missions, where all robots and other man-made structures are designed for particular missions and can thus be adapted accordingly beforehand. Furthermore, these detection and estimation methods are not limited to inter-robot loop closures. They also allow artificial structures like, for example, a lander or other infrastructure elements brought onto the surface of foreign planets, to serve as globally identifiable landmarks. Marker detections can then be used to identify and localize these objects, but also to improve single- and multi-robot localization through additional loop closure constraints. Visual fiducial marks have already been successfully used on real planetary exploration rovers since, at least, 2004, when the two rovers *Spirit* and *Opportunity* of the Mars Exploration Rover (MER) mission landed on Mars. Fiducials were placed on the lander and deck of each rover to check the geometric calibration of their panoramic cameras (Bell et al., 2003). The *Curiosity* rover of the subsequent Mars Science Laboratory (MSL) mission (landed in 2012) and the upcoming *ExoMars* rover (planned for 2020) feature visual fiducial marks as well, in order to support the calibration of cameras and various other rover mechanisms (Peters, 2016; Coates et al., 2017).

While there are well-established approaches for detecting and estimating the 6D pose of artificial markers, most methods do not provide an overall quality measure for their estimations. This, however, is crucial for the application in a SLAM system. The marker-based pose estimates need to be integrated as loop closure constraints according to their respective uncertainty, as overconfident measurements and false positives might degrade the results or even break the optimization process.

This section is based on data acquired and first analyzed as part of the master thesis of Vetter (2015), which was supervised by Christoph Brand and me. It features a rewritten, updated description and extended discussion of our findings. First, we summarize the marker detection process itself and discuss the potential sources for errors in the estimation pipeline (Section 6.2.1). We then introduce our empirical analysis of real-world estimation errors (Section 6.2.2) and a novel approach to model these (Section 6.2.3). Finally, we compare our model with the experimental data for validation, discuss its strengths and limitations, and identify aspects that could benefit from future improvements (Section 6.2.3).

6.2.1 Vision-Based Robot Detection and 6D Pose Estimation

A wide range of different active and passive visual marker systems has been developed and used to support localization in robotics applications, with several approaches originally coming from research in the field of augmented reality. A review and comparison would go beyond the scope of this thesis; for a discussion and comparison of several planar marker and detector systems see the publications of Olson (2011) and Wang and Olson (2016). The choice of a suitable marker system always involves an application-dependent trade-off between marker shape and size, detectability, computational effort, robustness and accuracy of detection and pose estimation, and the amount of payload that can be encoded into it.

For our rover prototype, we decided to use planar markers, which are easy to manufacture and can be detected with regular cameras. We selected the *AprilTag* visual fiducial system to define the markers and detectors, as it allows more accurate and robust detections of planar markers than several of its predecessor systems (Olson, 2011). Several implementations of AprilTag detectors are available as open source (Olson, 2016; Kaess, 2013). The markers themselves can be generated in adaptable complexities to carry a numeric payload that is sufficient to identify individual markers and robots in a multi-robot system. For real space missions, given the aforementioned trade-off, other marker shapes and detection systems might, however, be better suited. Thus, we aimed to design our methods for error modeling and estimation to be sufficiently general to allow a future adaption to other visual marker detection systems if needed.

The AprilTag Visual Fiducial System

The AprilTag markers and detectors (Olson, 2011) have been designed with focus on accuracy and robustness w. r. t. partial occlusions and changing light conditions. Their characteristics as well as the aforementioned public availability of open source implementations make them a popular choice in robotics research. Knowing the size of a marker, they allow the estimation of its full 6D pose from a single image. The markers themselves are simple black-and-white patterns that can easily be printed

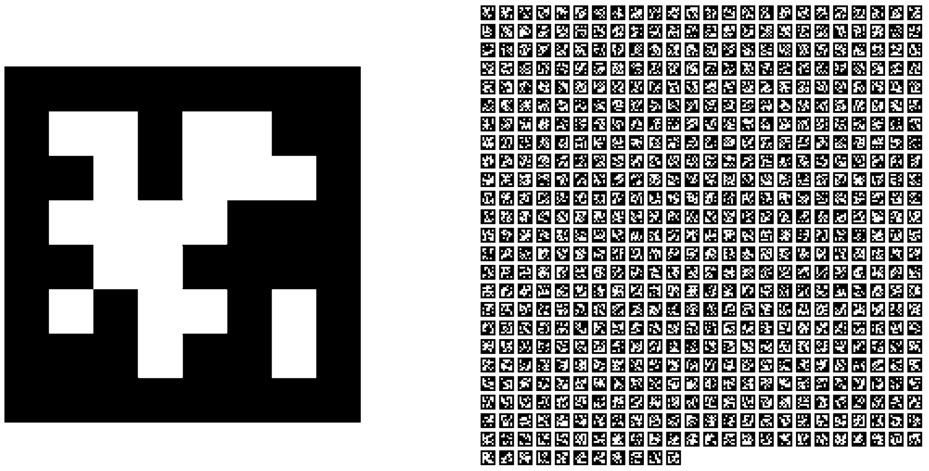


Figure 6.5: *AprilTag marker with payload "1" from the default marker family "36h11" (left) and mosaic showing all 587 valid tags of that family (right)*

onto planar surfaces. Each of them encodes a numeric id from a predefined set of valid values. A modified lexicode-based coding scheme allows to generate code families with different complexities and sets of codewords, defining their payload, identifiability, and error correction characteristics. Valid markers are required to be robust w. r. t. rotation and false positives arising from structures appearing in natural environments. The latter is achieved by rejecting codes that result in simple geometric patterns. In Figure 6.5, we give an example of a marker from the "36h11" default marker family, which uses a 36 bit encoding with a minimum Hamming distance of 11 between the code words. We also experimented with marker families of a lower complexity (e. g., "16h5") to increase the maximum view distance for detections. They, however, lead to a larger number of false positive detections by confusing parts of randomly textured floors with far-away markers. Thus, we decided to stay with the default "36h11" markers for all of our experiments.

In this paragraph, we provide a short summary of the marker detection and decoding developed by Olson (2011). The estimation process can be split into three major steps: the detection of a marker, the decoding of its payload, and the computation of a hypothesis for its 6D pose. The detection starts with a clustering of gradients in the image into components, which are then used for a least-squares fitting of line segments. Based on these, the next step searches for four-sided regions as candidates for marker detections. Their corner points are computed at sub-pixel accuracy by intersecting the line segments. Using a homography matrix, the tag-relative coordinates of the payload fields are transformed into image coordinates. The payload itself is then decoded based on spatially varying, local models for the intensity values of black and white pixels to achieve robustness w. r. t. changing light conditions.

Once a valid payload could be decoded, or reconstructed using error-correcting codes with sufficiently large Hamming distances, the 6D pose of the marker in the camera coordinate frame is computed. This requires solving the Perspective-*n*-Point (P*n*P) problem of finding a transformation between a set of 2D image points, i. e., the corner points of the marker, and their respective 3D points. We used the open source implementation of the iterative Levenberg-Marquardt optimization provided by the *OpenCV 2.4* library (OpenCV-Team, 2018) to estimate this transformation, given the camera parameters and the transformations between the points in 3D provided by the definition of the markers and their known sizes.

Error Sources in the Estimation Pipeline

The estimation pipeline is influenced by a multitude of potential error sources, both in the physical world as well as in the algorithms themselves. In Figure 6.6, we visualize the overall process from marker creation to 6D pose estimation, identify the various sources for errors therein and discuss them in the following paragraphs.

The marker detection and pose estimation are based on the assumptions of a perfect physical marker with known size. These can be violated through inexact printing, an uneven target material surface, and errors in the measurement of the printed marker's size. The projection of an image of the physical marker through a set of lenses onto the sensor of a camera is subject to aberrations like lens distortions, blur in defocused areas, etc. Furthermore, the conversion of the photons reaching the camera sensor into digital images is subject to noise, discretization effects, and possible over- or underexposures. All of these errors are propagated into the digital image or lead to deviations of the ideal model of the marker and thus can influence and degrade the result of the detection and pose estimation.

Certain types of errors can be compensated up to some degree through a good camera calibration that, for example, can be used to remove the most prominent types of radial image distortions during rectification. This process, however, introduces additional errors through the interpolation of pixel values that is necessary to fit the pixel grid of the image after correcting transformations. We obtain our camera model as well as an estimate of its projection error in pixel space from our camera calibrations that we carried out with the *DLR CalDe* and *DLR CalLab* toolboxes (Strobl and Hirzinger, 2008). Errors in the camera model influence all processing steps from image rectification to the marker pose estimation. Based on noisy and error-prone image data as well as approximations and heuristics in the detection algorithm, the marker detection results, i. e., the corner points of the marker, will be imprecise. This noisy data is propagated through the P*n*P solver to estimate a 6D pose in the camera

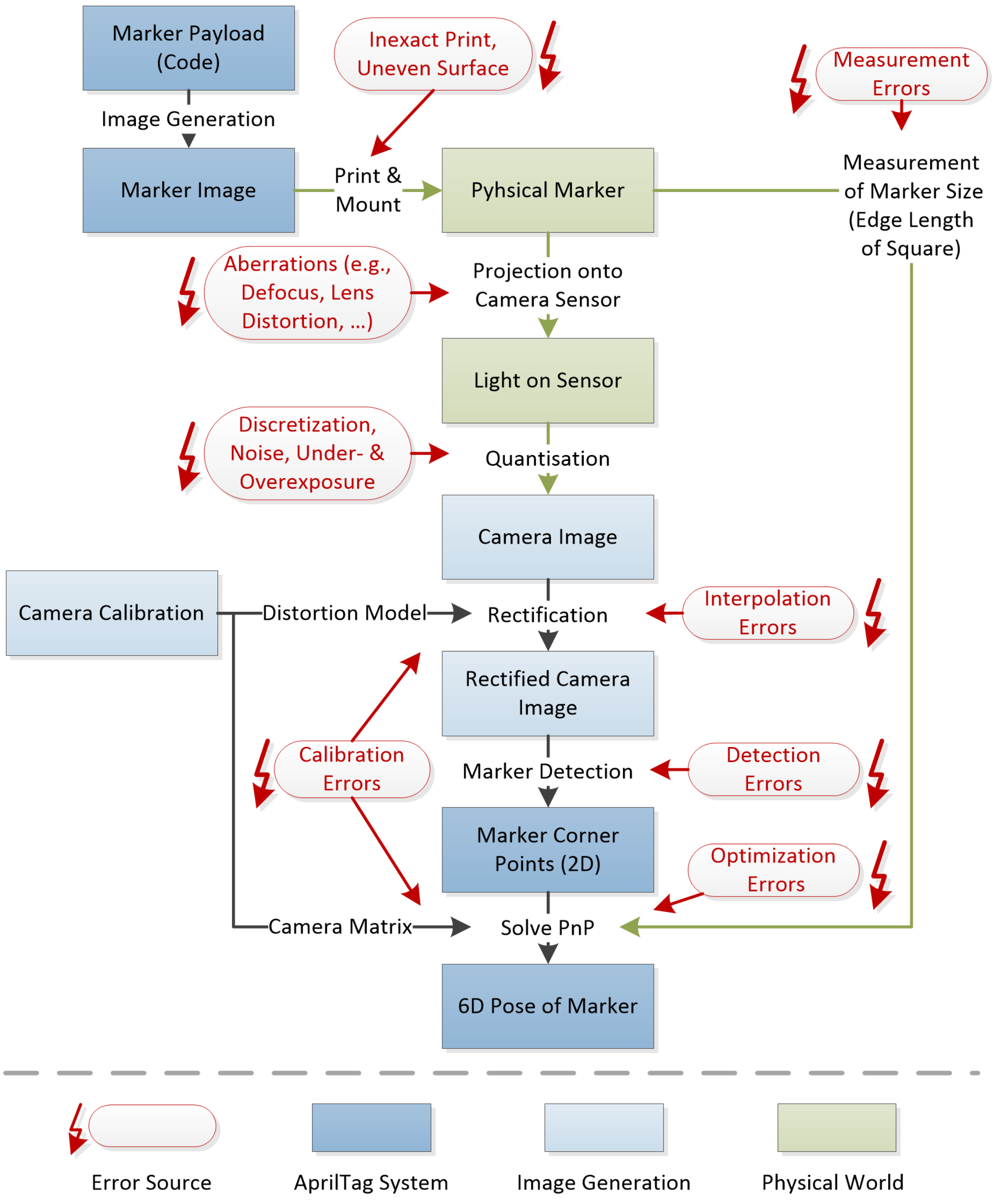


Figure 6.6: *Overview of marker detection and pose estimation process with its different types of potential error sources*

coordinate frame. For planar targets, noise in the corners can lead to ambiguities in pose estimation due to multiple local minima that can be resolved only up to some degree (Schweighofer and Pinz, 2006).

6.2.2 Analysis of Real-World Effective Estimation Errors

It is impossible to model all of the various potential error sources and estimate or even correct for their effects individually. Thus, we analyze their effective error, i. e., their combined effect on the accuracy of pose estimation, by looking at empirical findings from real-world marker detection experiments.

Data Acquisition

According to both Olson (2011) and our experience, the critical parameters influencing the quality of estimations are the marker's distance to the camera d and the rotation angle Φ between its normal vector and the vector pointing towards the camera. Thus, we designed an experimental setup for automatic data acquisition that allows us to vary both parameters in order to record the results along with ground truth transformations for sufficiently many configurations. It consists of one static greyscale camera (Guppy F-080B (Allied Vision Technologies GmbH, 2019a) with 1/3" chip size, resolution of 1032×778 px, and a $f = 5$ mm narrow-angle lens) as well as a set of AprilTag markers that we attached to a Pioneer 3-DX (P3DX) mobile robotic platform, see Section 7.1 for more details. As shown in Figure 6.7, both the static camera and the mobile robot are equipped with retro-reflective tracking targets. These are tracked by the ceiling-mounted cameras of a stationary *ARTTRACK2* system (Advanced Realtime Tracking GmbH, 2014), which we describe in Section 7.4. It allows us to capture ground-truth transformations between the tracking targets within its limited workspace of approx. 12 m^2. Due to this restriction, we used three markers with different sizes of 12 cm, 6 cm, and 3.5 cm in order to gather data within the full spectrum between close-range detections with the tag filling most of the image up to far-range detections that are at the limits of detectability. We compute the ground truth transformation $\boldsymbol{T}_m^c$ for the estimate $\hat{\boldsymbol{T}}_m^c$ of the 6D transformation between camera and marker frame by forming an alternative chain of transformations through the tracking system, as depicted in Figure 6.8:

$$\boldsymbol{T}_m^c \triangleq \boldsymbol{T}_{tc}^c \oplus \boldsymbol{T}_{to}^{tc} \oplus \boldsymbol{T}_{tm}^{to} \oplus \boldsymbol{T}_m^{tm} = \left(\boldsymbol{T}_c^{tc}\right)^{-1} \oplus \left(\boldsymbol{T}_{tc}^{to}\right)^{-1} \oplus \boldsymbol{T}_{tm}^{to} \oplus \boldsymbol{T}_m^{tm} \tag{6.8}$$

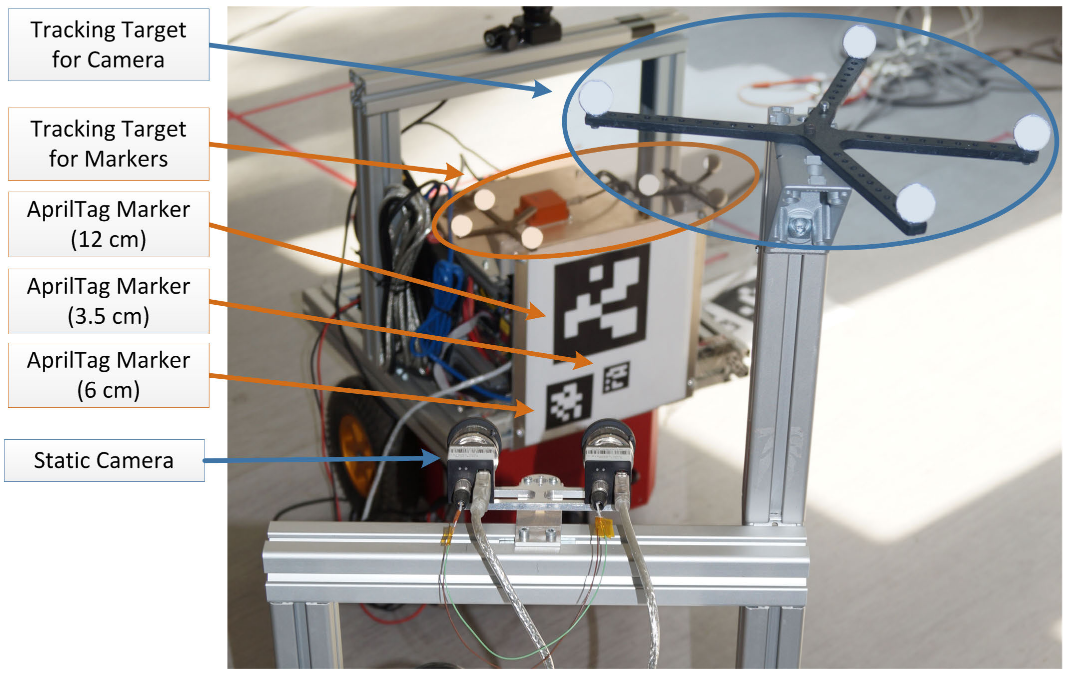


Figure 6.7: *Experimental setup to gather ground truth data with a static camera and markers attached to a mobile robot, both being localized through an external tracking system*

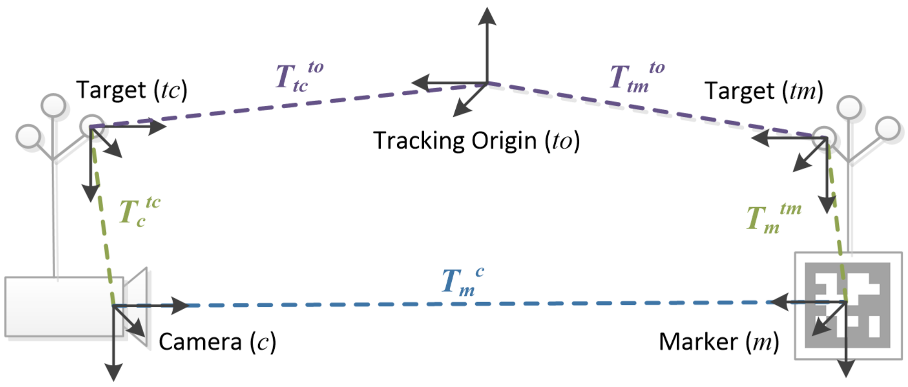


Figure 6.8: *Transformations in experimental setup to gather ground truth data for marker detection error analysis and model validation*

Hereby, $\boldsymbol{T}_{tc}^{to}$ and $\boldsymbol{T}_{tm}^{to}$ denote the transformations between the tracking system's (arbitrary) origin and the tracking targets attached to camera and moving marker respectively. $\boldsymbol{T}_{c}^{tc}$ and $\boldsymbol{T}_{m}^{tm}$ refer to the static transformations between the respective tracking targets and the coordinate frames of camera and marker themselves. We calibrated $\boldsymbol{T}_{c}^{tc}$ with methods for hand-eye calibration (Strobl and Hirzinger, 2006), acquiring data by moving the camera system in front of a checkerboard grid. For the calibration of $\boldsymbol{T}_{m}^{tm}$, we used an additional tracking target that was temporarily placed at the center of the marker. Despite all calibration efforts to get precise ground truth, chaining their uncertainties can still lead to errors in the ground truth transformations, in particular when angular errors are propagated over long distances. As our analysis thus can only give meaningful results for large-enough errors, we focus on effects above sub-centimeter and sub-degree level for translation and rotation respectively. In any case, these large deviations of marker pose estimates w. r. t. ground truth are the relevant ones that can have a significant impact on the overall optimization results when feeding them into a SLAM system.

We sampled the two-dimensional parameter space defined by d and Φ by moving and rotating the mobile robot at predefined poses within the field of view of the static camera, always keeping the tags approximately centered within the camera image. Thereby, the positioning accuracy of the robot platform itself w. r. t. its target pose is not relevant, as we could measure its final pose through the tracking system and use this value as ground truth. We acquired data for $d \in [0.6\,\mathrm{m},\ 3.1\,\mathrm{m}]$ with step sizes of 5 cm and 10 cm in the near- and far-range respectively as well as for $\Phi \in [-80°,\ +80°]$ with a step size of 5° at a total of 197 poses, as plotted in Figure 6.9. We increased the sampling density in the far range where small changes in parameters have a greater influence on the detectability and estimation errors. At each pose, we took approx. 850 images and averaged over more than 3000 measurements of the tracking targets. The drift of the static tracking transformation $\boldsymbol{T}_{tc}^{to}$ and the deviations of the ground truth transformation $\boldsymbol{T}_{m}^{c}$ from its mean value were in all cases below 3 mm and 0.3° for the individual degrees of freedom of translation and rotation respectively.

Data Analysis

In this section, we compare the marker pose estimates to ground truth for varying d and Φ. In Figure 6.10, we give an impression of the behavior of the estimation error in the camera frame for markers of 6 cm size at different distances. The z coordinate equals the distance d to the camera, whereas the tilt angle Φ results from the combination of $roll$ and $pitch$. As expected, the errors in x, y and yaw, representing translations in the marker plane and rotations around its normal, are in

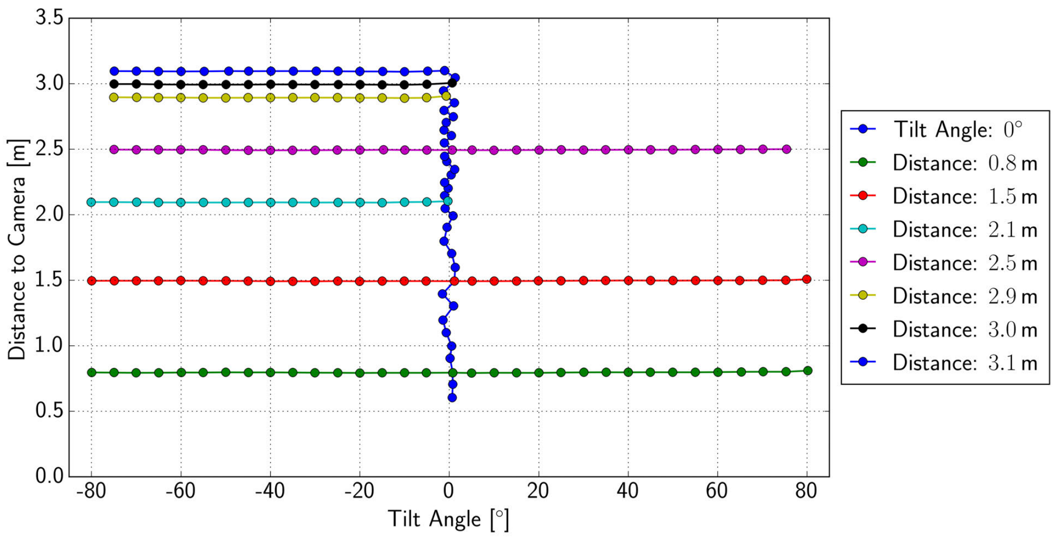
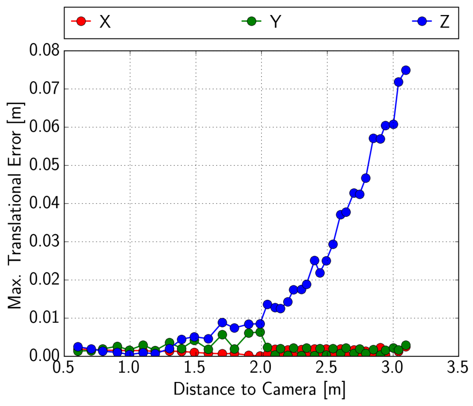
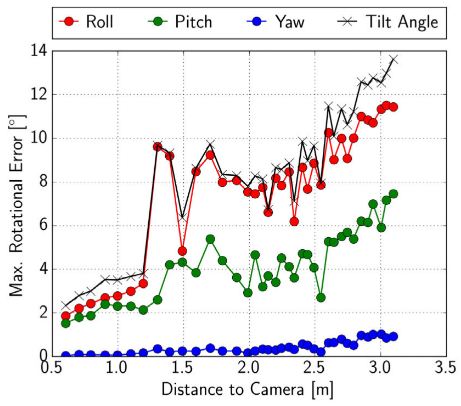


Figure 6.9: *Measured reference poses for which we acquired marker pose estimation data together with ground truth*

most cases negligibly small, often even below the level of measurement noise for our ground truth. Figure 6.10 shows a relevant increase of the error in z, *roll* and *pitch* with increasing distance to the camera. Regarding the errors w. r. t. variations in Φ at a fixed distance of 1.5 m, as shown in Figure 6.11, translation errors are mostly below 1 cm whereas errors in *roll* and *pitch* increase around a tilt angle of 0° to values of almost 10°. At a distance of 2.5 m, as shown in Figure 6.12, in particular the rotational errors reach very large values for a certain range of tilt angles. They are most likely caused by pose ambiguities in the step of solving the Perspective-n-Point (PnP) problem. As Schweighofer and Pinz (2006) analyzed, for planar markers, the cost functions used for optimization exhibit two minima both in image and object space for certain configurations, the similarity of their values increasing with increased distance and tilt angle. Choosing the wrong minimum would cause an incorrect estimation of approx. $-\Phi$ instead of Φ in case of rotation around a single axis, as the effect of perspective projection decreases with increasing distance d between camera and marker (degrading to an orthographic projection in the limit case of $d = \infty$). This leads to an estimation error of approx. 2Φ, which is exactly what we could observe in our data. We will treat these as a separate, special type of errors in our discussion and model. While they dominate the plot in Figure 6.12, we illustrate the other effects in Figure 6.13 ($d = 2.5$ m). For this, we compensated for the errors caused by pose ambiguities by approximating erroneous estimates $\hat{\Phi}$ with $-\hat{\Phi}$ and in addition filtered out any remaining large outliers. The resulting error plots show a similar behavior as those depicted in Figure 6.11 ($d = 1.5$ m), i. e., having peaks around a tilt angle of 0°.

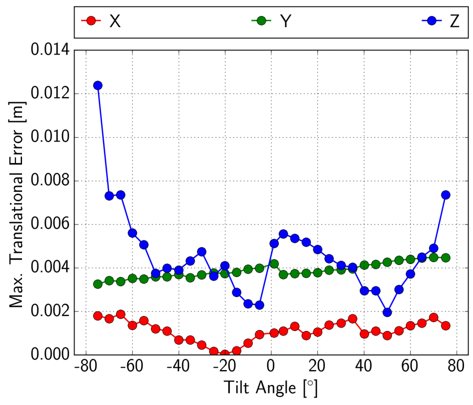


(a) *Translational errors*

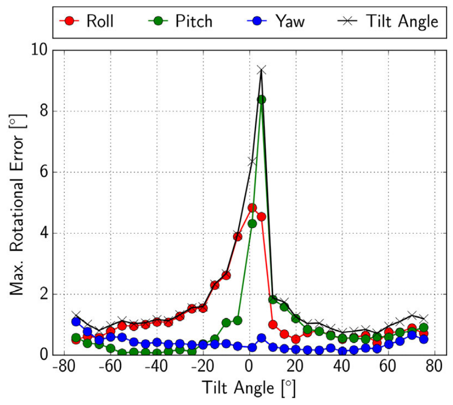


(b) *Rotational errors*

Figure 6.10: *Maximum errors for tag detection experiments at different distances to the camera d and a view angle of $\Phi = 0°$ (marker size: 6 cm)*

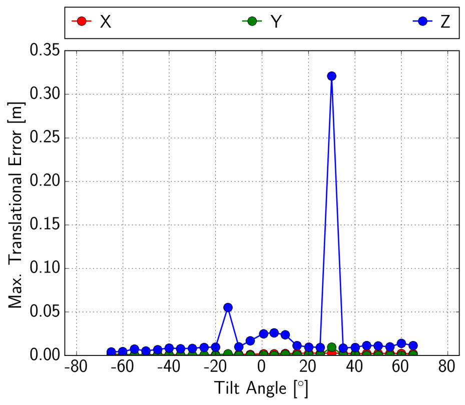


(a) *Translational errors*

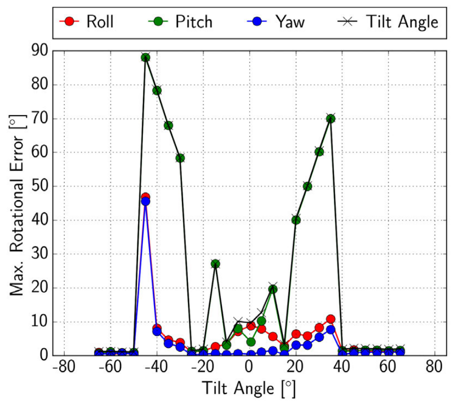


(b) *Rotational errors*

Figure 6.11: *Maximum errors for tag detection experiments at $d = 1.5$ m and different view angles Φ (marker size: 6 cm)*

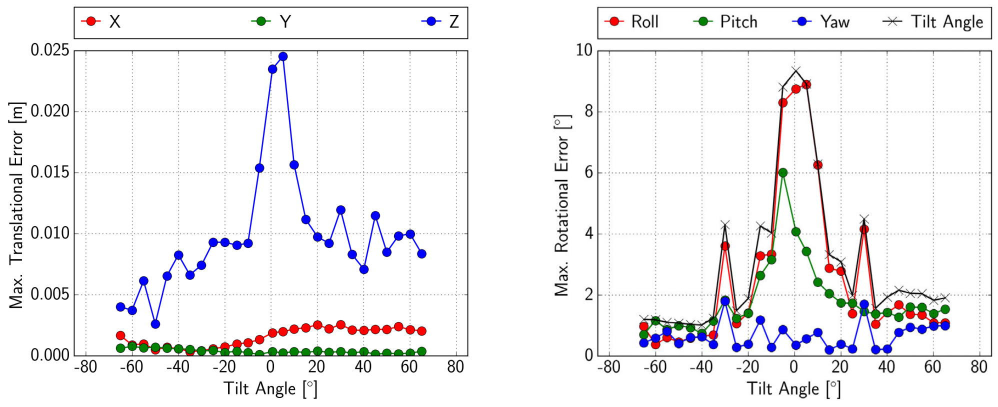


(a) *Translational errors*



(b) *Rotational errors*

Figure 6.12: *Maximum errors for tag detection experiments at $d = 2.5$ m and different view angles Φ (marker size: 6 cm)*



(a) *Translational errors* ***(b)*** *Rotational errors*

Figure 6.13: *Maximum errors for tag detection experiments at $d = 2.5$ m and different view angles Φ (marker size: 6 cm), similar to Figure 6.12, but with outliers and large rotational errors due to PnP pose ambiguities filtered out*

Schweighofer and Pinz (2006) propose a PnP algorithm to identify both minima of the cost function and select the one with the lower value during optimization. While integrating this approach into our system is an interesting topic for future work, we do not expect it to completely resolve the issue. Under heavy image noise and in particular for large distances, the correct minimum does not always have the lowest cost value, making it impossible to guess the correct solution without further information. Furthermore, most theoretical considerations are made under the assumption of Gaussian image noise. As we discussed earlier in Section 6.2.1, many factors influence the quality of the estimation, such as the camera calibration as well as errors in printing and measuring the marker. Thus, it is reasonable to assume that their combined effect can only roughly be approximated by Gaussian noise and that certain error sources can, for example, cause non-Gaussian biases, posing additional challenges to an identification of the correct PnP solution.

In summary, we observed three major effects in our data: first, a dependence of the error on the distance between camera and marker that is non-linear at least w. r. t. the translational error. Second, an increase of rotational errors for tilt angles around $\Phi = 0°$. Third, large rotational errors that are most likely caused by the aforementioned pose ambiguities. All of these effects could be observed in our results for the three different tag sizes and the additional robot poses indicated in Figure 6.9. However, occasional additional large outliers occurred, in particular when approaching the limits of detectability for the smaller tag sizes. Some of them might be correlated to the pose ambiguities and measurements off the center of the camera. A thorough analysis of these, however, remains a topic for future work.

6.2.3 Modeling of the Estimation Uncertainty

As discussed above, the marker pose estimation is subject to a large number of different error sources, leading to several overlain effects in the results. In order to model the marker pose estimation uncertainty, we designed a pipeline that allows us to simulate many of the relevant effects using the actual marker detector and P*n*P solver. This makes our general approach independent of the underlying detection and P*n*P algorithms and their respective implementations, which thus can easily be replaced by improved versions over time.

Separating the large rotational errors caused by the aforementioned pose ambiguities from most of the other effects, we arrived at a two-step approach: The first step is based on a precomputed lookup table that serves as a model for pose estimation uncertainties. The second step samples variations of the input values to the P*n*P solver close to the actual estimate in order to identify areas of instability that can lead to large errors caused by the aforementioned pose ambiguities. We present a sketch of the architecture of our offline precomputation and online error estimation in Figure 6.14 and describe the individual processing steps in the following paragraphs.

Precomputation of Error Model

Our error model is based on a lookup table that we precompute offline by simulating a large number of marker detections, as shown in the left part of Figure 6.14. The lookup table contains the maximum detection errors for marker poses sampled along a two-dimensional grid with varying distance d and angle Φ w. r. t. the camera. We chose their respective minimum and maximum values to cover the full space of detectable poses. For each of the marker poses, we simulate the rectified camera image by rendering a marker with the freely available ray tracing software *POVRay* (Persistence of Vision Raytracer Pty. Ltd., 2008) and then apply a manually tuned Gaussian blur to the resulting image to approximate the smoothing in defocused real images. Thus, each precomputed lookup table depends on the camera's resolution, its focal length, as well as the marker type and size. In Figure 6.15, we present an example of a real camera image and a rendering of the three visible tags at the same respective poses.

We then run the AprilTag detector to detect the marker and estimate the coordinates of its four corner points in the 2D image. In order to simulate the influence of additional error sources like camera calibration and image rectification, we sample additional detections with distorted pixel values for the detected corner points. We base this on the pixel error Δp of the rectified camera image estimated during camera

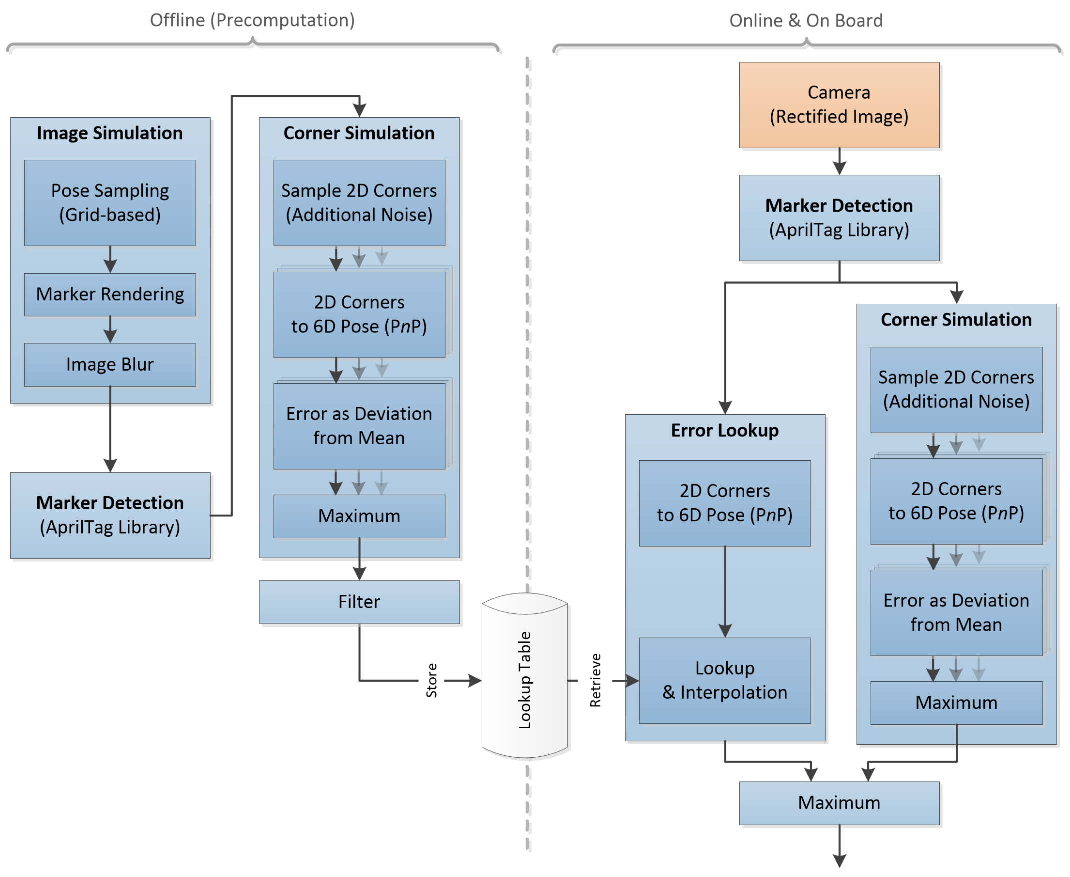
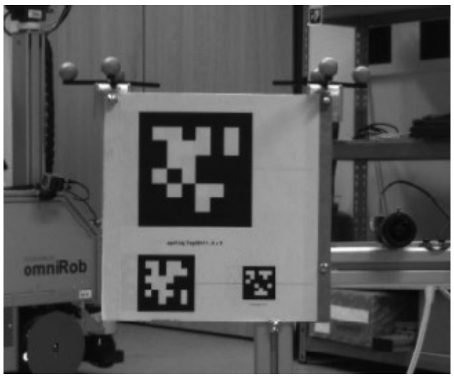
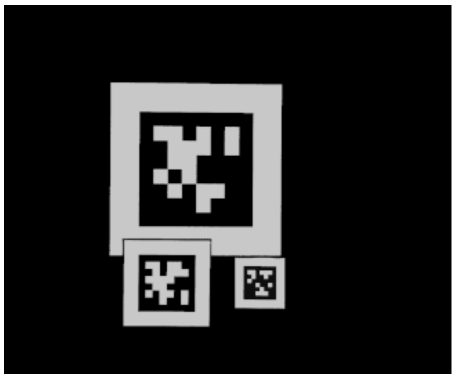

Offline (Precomputation)
Online & On Board
Image Simulation
Pose Sampling (Grid-based)
Marker Rendering
Image Blur
Marker Detection (AprilTag Library)
Corner Simulation
Sample 2D Corners (Additional Noise)
2D Corners to 6D Pose (PnP)
Error as Deviation from Mean
Maximum
Filter
Store
Lookup Table
Retrieve
Camera (Rectified Image)
Marker Detection (AprilTag Library)
Error Lookup
2D Corners to 6D Pose (PnP)
Lookup & Interpolation
Corner Simulation
Sample 2D Corners (Additional Noise)
2D Corners to 6D Pose (PnP)
Error as Deviation from Mean
Maximum
Maximum

Figure 6.14: *Overview of our two-step approach to model marker detection errors*



(a) *Camera image*



(b) *Rendering with Gaussian blur*

Figure 6.15: *Visual comparison of real camera image and rendered tags at the same poses*

calibration (see also Section 5.1.1). With the four corner points being represented by the eight coordinate values $(q_0, \ldots, q_7) \in \mathbb{R}^8$, we generate a set $\mathcal{S}$ of $|\mathcal{S}| = 2^8 = 256$ samples by taking the Cartesian product of the variations to each coordinate value:

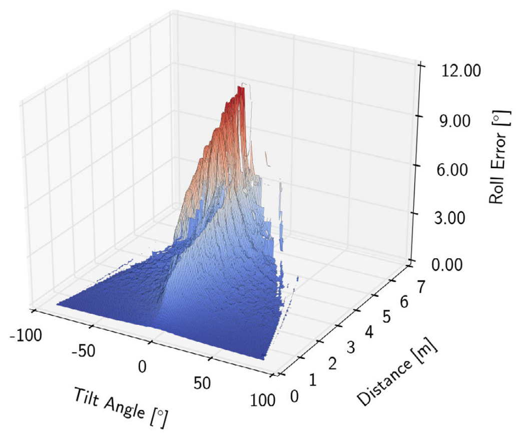
$$\mathcal{S} = \prod_{i=0}^{7} \{q_i + \Delta p,\ q_i - \Delta p\} \tag{6.9}$$

We then pass the original as well as all sampled vectors of corner coordinates to the P*n*P algorithm, compute the errors of the resulting 6D poses w. r. t. the known ground truth and write their maximum values into the lookup table.

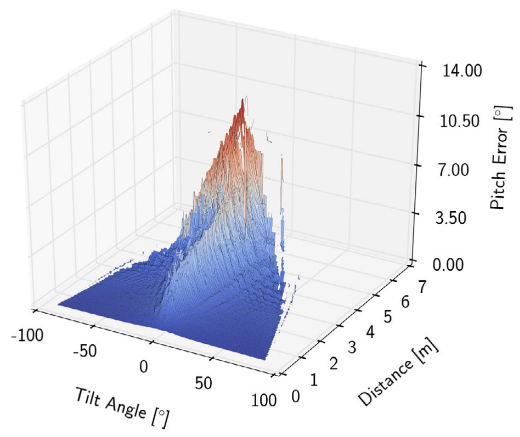
As post-processing steps, we compensate for large rotational errors given the known ground truth values similar to Section 6.2.2 and remove outliers by applying a median filter to the lookup table. This eliminates large untypical error values that are likely to appear only in very specific configurations. They would be hard to model in a lookup table of limited resolution that is based on an approximative simulation of a multitude of potential error sources influencing the results. Thus, we handle these effects separately in the second step of our pipeline during online error simulation based on the actual detection that we describe at the end of this section. In Figure 6.16, we present a visualization of the precomputed lookup table for $200 \times 200 = 40000$ poses for a marker of size 6 cm, equidistantly sampled from $\Phi \in [-89°,\ 89°]$ and $d \in [0.2\,\mathrm{m},\ 7.0\,\mathrm{m}]$. Missing values indicate poses for which the rendered AprilTag marker could not be detected. As a final post-processing step, they are filled using two simple extrapolation steps. First, each row, i. e., discretized distance value, is filled with the error value at the border of the valid region. Second, for each column, i. e., discretized tilt angle value, the maximum of the final 10% of valid error values is used to fill its remainder. However, this simple extrapolation technique tends to overestimate the errors for large angles and to underestimate them for large distances. Thus, replacing it with more complex extrapolation methods, such as quadratic curve fitting, could give more accurate results in the border regions of the lookup table. Filling the missing values in the lookup table already during precomputation allows to reduce the computational effort of the online lookup at runtime.

Online Lookup and Error Simulation

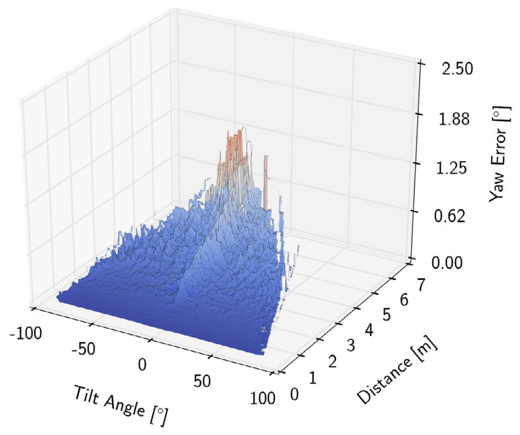
At system runtime, we use a combination of two different error approximation heuristics to compute an estimate for the uncertainties associated with the 6D pose of an individual marker detection. We give an overview of this online computation in the right part of Figure 6.14. The first step is a lookup of the precomputed error values



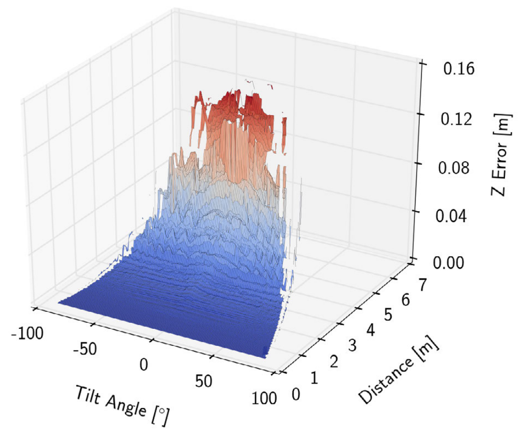
***(a)** Error in roll*



***(b)** Error in pitch*



***(c)** Error in yaw*



***(d)** Error in z*

***Figure 6.16:** Precomputed lookup table generated from simulated marker detections (outliers filtered, marker size: 6 cm). We omitted plots for the errors along the x and y axes as they are all below 0.001 m and thus negligible for our application.*

for the estimated distance and tilt angle from the error model that we generated offline beforehand. We use bilinear interpolation to retrieve error values from the two-dimensional lookup table for inputs that differ from the discretized indices of the precomputed values. This gives us the base error that is to be expected at a certain input pose, neglecting any effects caused by outliers due to pose ambiguities, which are dealt with in the next step.

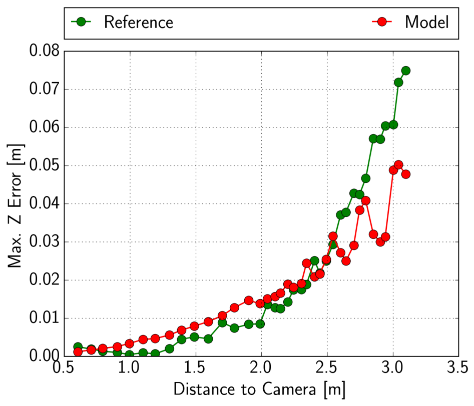
In a second step, we consider noise on the estimated marker corners that gets propagated through the PnP optimization. For this, we sample values for 256 additional sets of marker corner points as input to the PnP solver, similar to the sampling done during error model precomputation described in Equation 6.9. We base their values on the actual corners estimated by the marker detector and thus are able to identify areas of instability of the P*n*P optimization, i. e., input values for which small variations can cause large changes and thus large errors in the outcome. As at this point no ground truth is available, we take the maximum deviation from the arithmetic mean of the actual detection and all 256 additional samples as an approximation of the error. This allows us to account for large errors due to instabilities of the P*n*P solver around a particular pose. We can compute this sampling-based step online for each marker detection as its computational overhead is negligible compared to the image processing required for marker detection.

To approximate the maximum error for a particular detection, we take the maximum value returned by the two estimation methods. As the integration of marker pose estimates in our graph optimization-based SLAM framework (presented in Section 4.2) requires Gaussian uncertainty estimates, we take the maximum error values as rough approximations for the standard deviations of the individual six degrees of freedom.

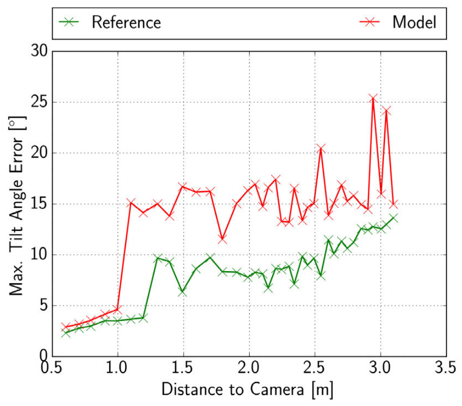
6.2.4 Model Validation and Discussion

The goal of our error model is to provide an estimation of the pose uncertainty that is conservative in the sense that it provides an approximation for worst case detections and thus rather overestimates the real error. While underestimated uncertainties would risk to degrade the overall results of our graph-based global optimization on erroneous measurements, an overestimation of the error can at most lead to a reduction of potential improvements. In order to validate our error model, we compare it to the experimental data acquired for our error analysis presented in Section 6.2.2. In Figure 6.17 to 6.22, we present exemplary data for our 6 cm and 3.5 cm markers for $\Phi = 0$ at varying distances as well as for $d \in \{1.5\,\text{m},\ 2.5\text{m}\}$ and varying tilt angles. We omitted the error curves for the 12 cm markers, as they exhibit in large parts similar behavior and are less relevant in our evaluation due to the better detectability and thus lower error values for the 12 cm markers within the maximum distance of 3.1 m, for which we could acquire ground truth during our experimental evaluation.

The plots show a comparison of the outcome of our two-step error model simulation, presented in Section 6.2.3, to the real-world marker detections that we acquired during our data analysis described in Section 6.2.2. Our model is able to capture the two main characteristics of the error behavior: its dependence on the distance to the camera and an approximation of the pose ambiguity errors. As can be observed in Figure 6.17 and Figure 6.20, the translational error, however, is underestimated

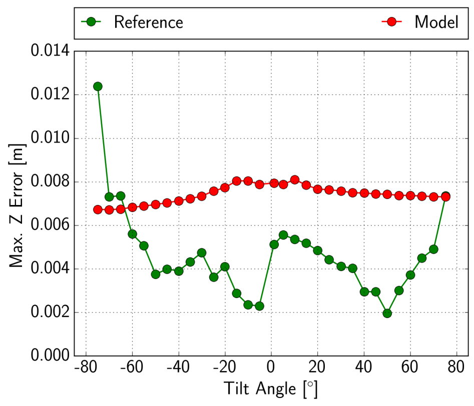


(a) *Translational errors*

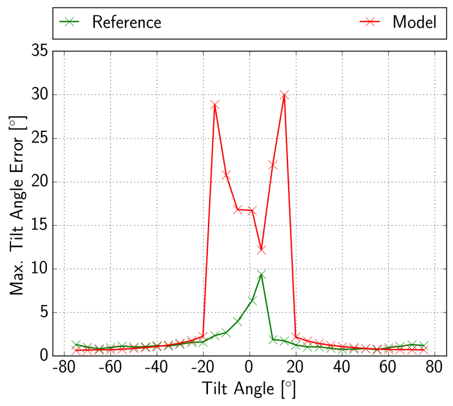


(b) *Rotational errors*

Figure 6.17: *Comparison of the maximum marker pose estimation errors predicted by our model with those from experiments at varying distances* d *(*$\Phi = 0$*, marker size:* 6 *cm)*

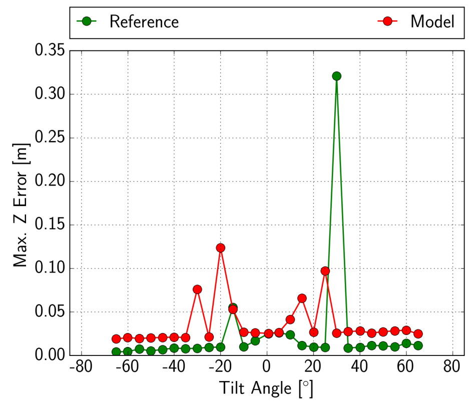


(a) *Translational errors*

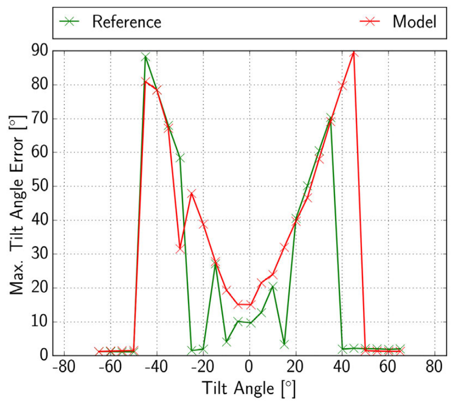


(b) *Rotational errors*

Figure 6.18: *Comparison of the maximum marker pose estimation errors predicted by our model with those from experiments at varying tilt angles* Φ *(*$d = 1.5$ *m, marker size:* 6 *cm)*

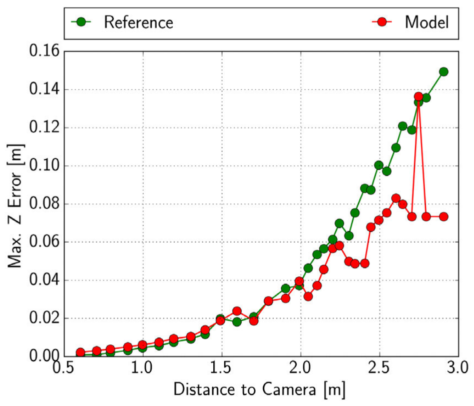


(a) *Translational errors*

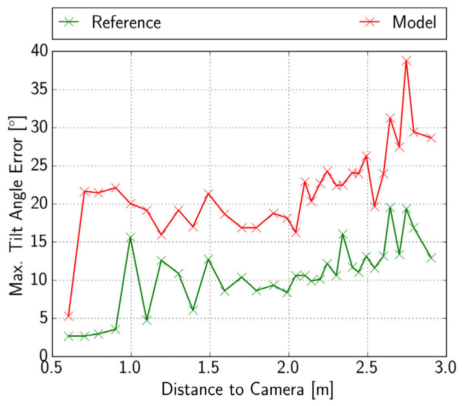


(b) *Rotational errors*

Figure 6.19: *Comparison of the maximum marker pose estimation errors predicted by our model with those from experiments at varying tilt angles* Φ *(*$d = 2.5$ *m, marker size:* 6 *cm)*

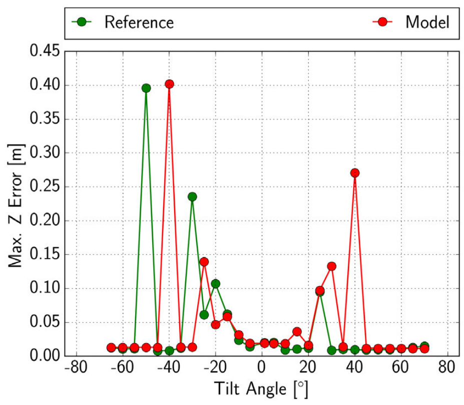


***(a)** Translational errors*

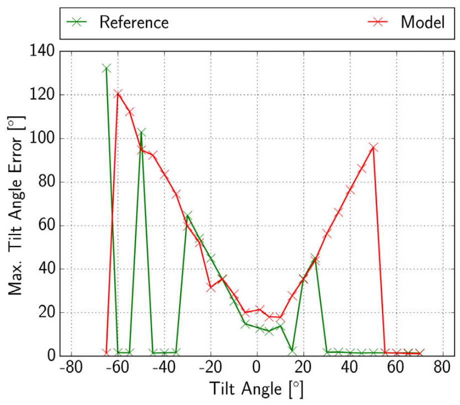


***(b)** Rotational errors*

***Figure 6.20:** Comparison of the maximum marker pose estimation errors predicted by our model with those from experiments at varying distances* d *(*$\Phi = 0$*, marker size:* 3.5 cm*)*

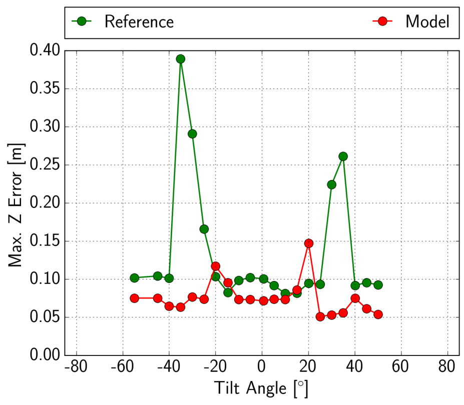


***(a)** Translational errors*

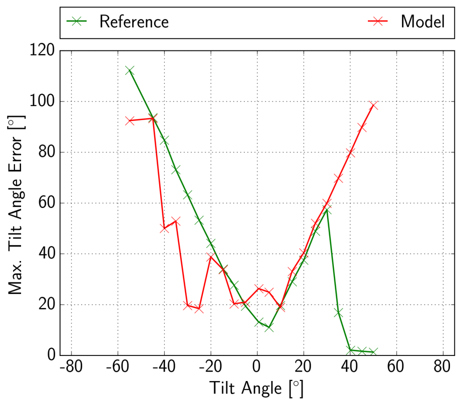


***(b)** Rotational errors*

***Figure 6.21:** Comparison of the maximum marker pose estimation errors predicted by our model with those from experiments at varying tilt angles* Φ *(*$d = 1.5$ m*, marker size:* 3.5 cm*)*



***(a)** Translational errors*



***(b)** Rotational errors*

***Figure 6.22:** Comparison of the maximum marker pose estimation errors predicted by our model with those from experiments at varying tilt angles* Φ *(*$d = 2.5$ m*, marker size:* 3.5 cm*)*

for large distances. Some of these differences are artifacts of our simple inter- and extrapolation method to fill missing values at large distances to the camera, for which markers still could be detected on real images but not on the simulated ones. One potential cause for this might be our approximation of the influence of a camera's limited depth of field by adding Gaussian blur to the simulated images. This is a rough but easy-to-compute approximation for the defocus aberration of real cameras that could be replaced by a more complex and realistic image rendering in future work. This difference between our model and real-world error sources might also cause the individual outliers at large distances and certain angles visible in Figure 6.19(a) and Figure 6.22(a). These and occasional further deviations between our model and the reference measurements might be eliminated by adding additional error sources and variations to our simulation, for example, camera calibration errors, small random changes in the marker poses for the precomputed images, additional artificial image noise, and the sampling of different angles around all three rotational axes as well of off-center positions for the marker in the images. While our current modeling approach leaves room for future improvements, it is sufficiently accurate to be of practical use for our multi-robot SLAM application. It allows an identification of potentially ambiguous detections and, integrated into our overall SLAM system, outperforms simple constant, linear and quadratic error models, as we show in our experimental evaluation in Section 8.1.3. In addition to achieving an increased overall accuracy, our error model allows the integration of far-away marker detections into the SLAM graph with a significantly lower risk of underestimating large errors that might deteriorate the overall global optimization results. While we are aware that in particular pose ambiguity errors are highly non-Gaussian, a more accurate representation of the error distribution would require a different, non-standard graph optimization approach.

We based our error model on a simulation of marker detections and a sampling-based propagation through the P*n*P algorithm that we validated against experimental data. This allows our general method to be independent of the actual marker detection algorithm and its implementations, in particular compared to analytical approaches based on finding a closed-form solution for the Jacobian that describes the propagation of image errors through the processing pipeline.[5] As the experimental validation of our model as well as some of its parameters are specific to a particular camera system, it is advisable to perform validation experiments for each new camera system or, as future work, analyze how well our model generalizes across cameras. To reduce this effort, instead of creating a large lookup table, it might be possible to find

[5]As most P*n*P techniques make use of iterative optimization algorithms, defining a Jacobian for them might be a non-trivial task that provides a model specific to a particular implementation of a detection and estimation pipeline.

a precomputed model based on fewer parameters that is able to represent all major effects and can be tuned and validated with the data from a small set of well-defined real-world experiments. Further improvements might be gained by integrating the robust P*n*P approach developed by Schweighofer and Pinz (2006) and evaluating how it compares to our sampling-based online technique to identify potentially ambiguous poses and their maximum errors. The influence of partial occlusions and varying light conditions constitute further topics for future analysis and potential aspects to include into an error model. Detecting multiple markers attached at different orientations to a 3D structure might mitigate the problem of pose ambiguities and greatly improve the overall estimation accuracy, however, requiring sufficient space and visibility. A tracking over time of either moving markers or static markers from moving platforms is a further option for future improvements in the context of mobile robotics. This, however, requires a certain minimum detection frequency, for which the computational resources might not always be available, in particular in space exploration scenarios.

Chapter seven

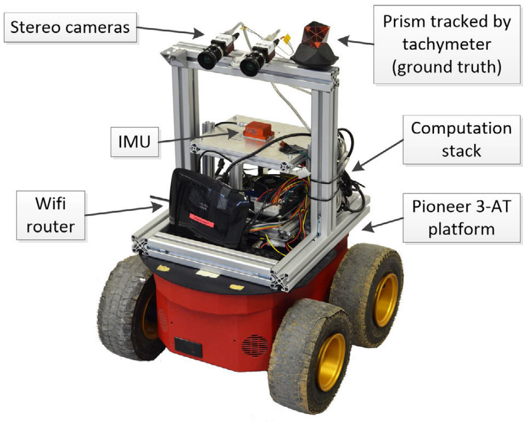
Robot System Integration

In this chapter, we describe how we integrated our methods on several different robot platforms for the evaluations and demonstrations presented in Chapter 8 and Chapter 9. We start by introducing the robot hardware platforms in Section 7.1, focusing on their sensor equipment and computation stacks. In Section 7.2, we then present the robots' system architecture and the integration of our localization and mapping modules therein. Afterwards, in Section 7.3, we introduce a high-fidelity simulation environment for LRU that we used during the development, integration, and testing of our navigation and mapping methods. In Section 7.4, we describe the techniques that we employed to acquire trajectory ground truth data for our quantitative evaluations.

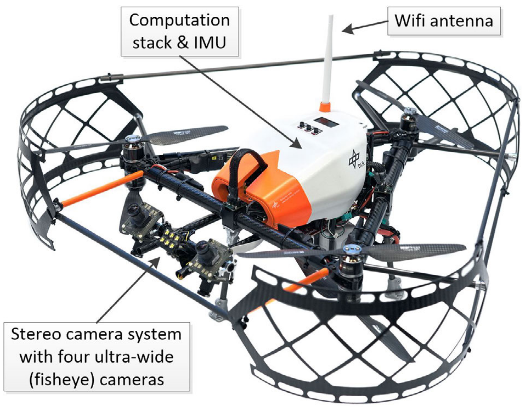
7.1 Robot Hardware Platforms

In order to conduct evaluations and demonstrations, we integrated our localization and mapping framework onto the different robot platforms shown in Figure 7.1. All robots feature a stereo camera-based vision system as well as an IMU as their primary sensor setup.

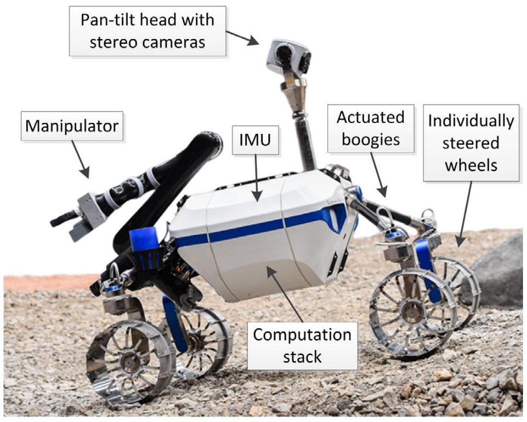
- *Pioneer Platforms (P3AT, P3DX)* (see Figure 7.1(a))
  The Pioneer 3-AT (P3AT) (for indoor and outdoor use) and Pioneer 3-DX (P3DX) (indoor only) are commercial robotic platforms that are widely used



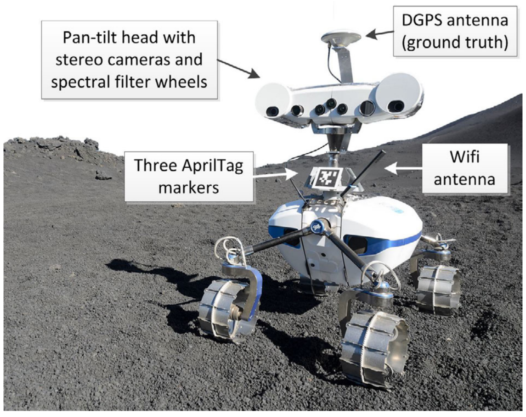
(a) *Rover Pioneer 3-AT (P3AT)*



(b) *Multicopter Ardea*


Pan-tilt head with stereo cameras
Manipulator
IMU
Actuated boogies
Individually steered wheels
Computation stack

(c) *Planetary exploration rover LRU2 with a robotic manipulator arm*


Pan-tilt head with stereo cameras and spectral filter wheels
DGPS antenna (ground truth)
Three AprilTag markers
Wifi antenna

(d) *Planetary exploration rover LRU1 with scientific spectral cameras*

Figure 7.1: *Robot platforms for localization and mapping*

in research (Adept Technology Inc, 2011a,b). They feature skid-steer (P3AT) and differential drive (P3DX) locomotion. We did not compute any wheel odometry for these platforms as we primarily used the outdoor-capable P3AT and would expect its skid-steer locomotion to cause significant inaccuracies. In order to test a heterogeneous sensor setup in a multi-robot team, we equipped P3AT with a narrow-angle stereo system (focal length $f = 5\,\text{mm}$, baseline $b = 9\,\text{cm}$, Guppy F-080B cameras (Allied Vision Technologies GmbH, 2019a) with 1/3" chip size, resolution: 1032 px×778 px, horizontal stereo field of view: 50.6 °) and P3DX with a wide-angle setup ($f = 1.28\,\text{mm}$, $b = 9\,\text{cm}$, Guppy PRO F-125B cameras (Allied Vision Technologies GmbH, 2019b) with 1/3" chip size, resolution: 1292 px × 964 px, horizontal stereo field of view: 112.8 °). For a number of outdoor evaluations to compare narrow- and wide-angle datasets, we temporarily switched the cameras of P3AT to the wide-angle setup.

- *Lightweight Rover Unit (LRU)* (see Figure 7.1(c),(d))
  The Lightweight Rover Units (LRUs) are two planetary exploration rover prototypes that have been developed at the Institute of Robotics and Mechatronics (RM) of the German Aerospace Center (DLR). Each of these small rough-terrain rovers, LRU1 and LRU2, weighs approx. 40 kg, has four individually powered and steered wheels and can operate up to approx. 1.5 h before changing or recharging their two 208 Wh Li-Ion batteries. We provide a more detailed description of the LRU platform in our journal article by Schuster et al. (2017). For navigation, both rovers are equipped with greyscale narrow-angle stereo cameras (f = 5 mm, b = 9 cm, Guppy PRO F-125B cameras (Allied Vision Technologies GmbH, 2019b) with 1/3" chip size, resolution: 1292 px × 964 px, horizontal stereo field of view: 50.6 °). The rovers' sensor heads can be moved via a pan-tilt mechanism that allows them to pan the cameras by ±180 ° and to tilt them by ±90 ° (Wedler et al., 2013). A third, central camera in the sensor head can be adapted to mission-dependent tasks, for instance, by adding a color sensor or a zoom lens. Both robots allow the computation of wheel odometry (Bussmann et al., 2018), which we used for navigation as input to the local reference filter in some, but not all, of our experiments. LRU2 is equipped with a robotic arm for sampling, object manipulation, as well as the deployment of infrastructure elements and measuring devices. LRU1, in contrast, can be equipped with an optional platform[1] to transport Ardea, our MAV that we describe below. In addition, LRU1 features additional scientific cameras with spectral filter wheels to differentiate and analyze, for instance, different types of rock. For robot detections in multi-robot scenarios, we equipped each LRU with three tilted AprilTag markers around its camera mast to ensure visibility from all sides as well as from above.

- *Multicopter Ardea* (see Figure 7.1(b))
  Ardea is a hexacopter, i. e., a multicopter with six rotors, which has been developed at the Institute of Robotics and Mechatronics (RM) of the German Aerospace Center (DLR). Its triangular shape allows an unblocked field of view for its four ultra wide-angle (fisheye) cameras (f = 2.1 mm, VRmS-16 cameras (VRmagic Imaging GmbH, 2018) with 1/3" chip size, resolution: 1280 px×860 px) while minimizing the robot's footprint. The four synchronized cameras are arranged in two stereo setups (b = 12 cm) with the optical axes of the lower two cameras being tilted at −60 ° w. r. t. the horizon, and those of the upper two cameras at +60 °. This leads to a combined 80 ° horizontal and 240 ° vertical field of view, allowing Ardea to observe the ground as well as the

[1]The platform was not attached in the image in Figure 7.1(d) but can be seen in Figure 9.19.

ceiling or sky simultaneously at all times. For stereo processing, each image of a ultra wide-angle camera is split and remapped into two virtual pinhole cameras with a resolution of 666 px × 506 px using bilinear interpolation. We refer to our conference paper by Müller et al. (2018) for more details on Ardea and its vision system.

All robots feature similar On-Board Computers (OBCs) (rovers: quadcore Intel Core i7 at 2.7 GHz, Ardea: dualcore Intel Core i7 at 3.0 GHz) and a Spartan 6 LX75 FPGA extension board to perform dense stereo matching with a resolution of 1024 px × 508 px at a frame rate of up to 14.6 Hz. The effective camera frame rates varied across the robots between 8 Hz (Ardea) and 14.6 Hz (LRU2) due to bandwidth limitations on their buses in certain configurations. The effective field of view for the stereo processing is limited by the maximum computable image size of the FPGA implementation. On the rovers, we compute stereo matches only for the central part of the higher-resolution camera images, leading to the effective fields of view stated above (50.6 ° for the narrow-angle and 112.8 ° for the wide-angle systems). For Ardea's four-camera setup, the input images are aligned in a 2 × 2 layout and scaled by a factor of 0.5 in order to compute lower-resolution stereo data for its full field of view. The rovers are equipped with a Xsens MTi-10 IMU (Xsens, 2019), Ardea with a smaller and more lightweight Analog Devices ADIS16407 IMU (Analog Devices Inc, 2011).

We conducted our early evaluations with the Pioneer platform and later switched to the LRU rovers. During our final demonstration (see Section 9.4), we deployed the multicopter Ardea together with LRU to form a heterogeneous team of aerial and ground-based robots.

7.2 Robot System Architecture

In Figure 7.2, we present an overview of the IT architecture of our robots, focusing exemplarily on LRU2. The system architecture, as well as the computation stacks, are very similar between all of our robots, with minor differences depending on the respective platform, additional hardware modules and their bus systems. Considering a space exploration scenario, Figure 7.2 does not only include the robots but also a lander, a ground station, as well as the communication links in between. The ground station consists of computers to monitor and control the robot team from Earth over a communication link to the on-site lander. The role of the lander in our setup is

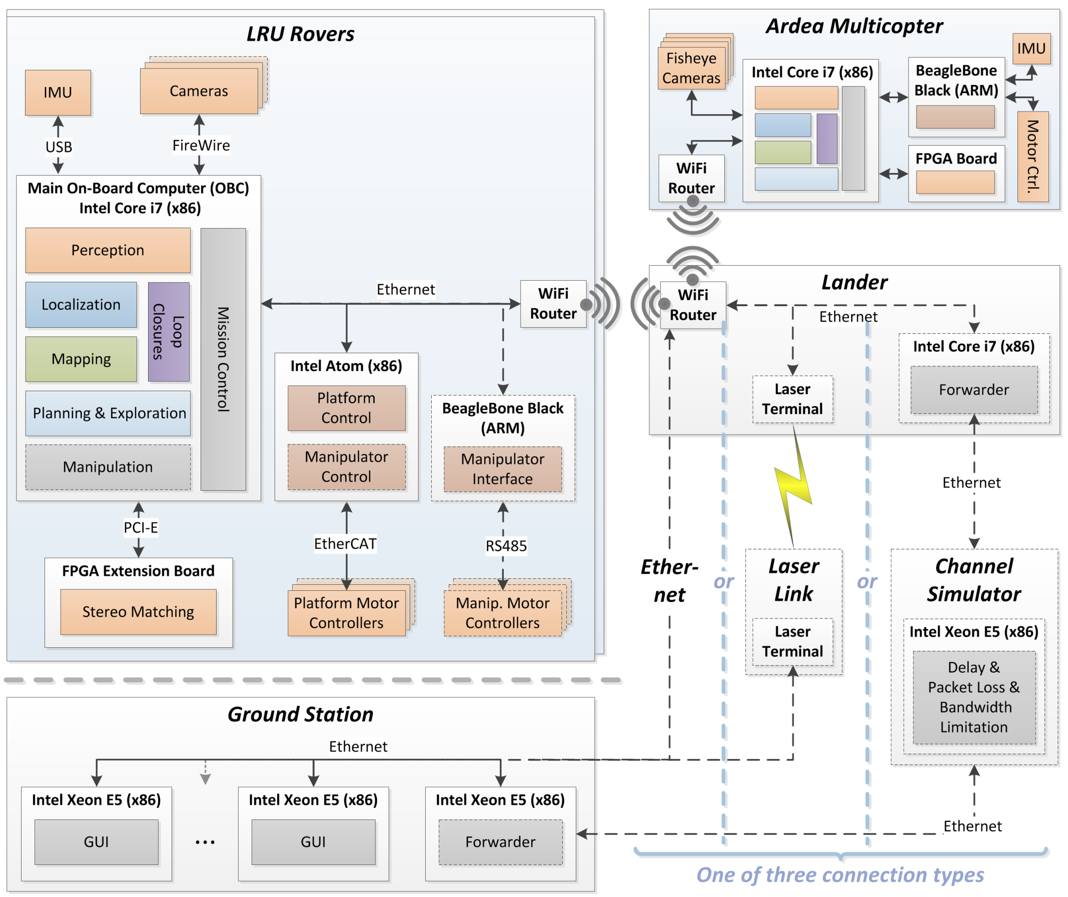
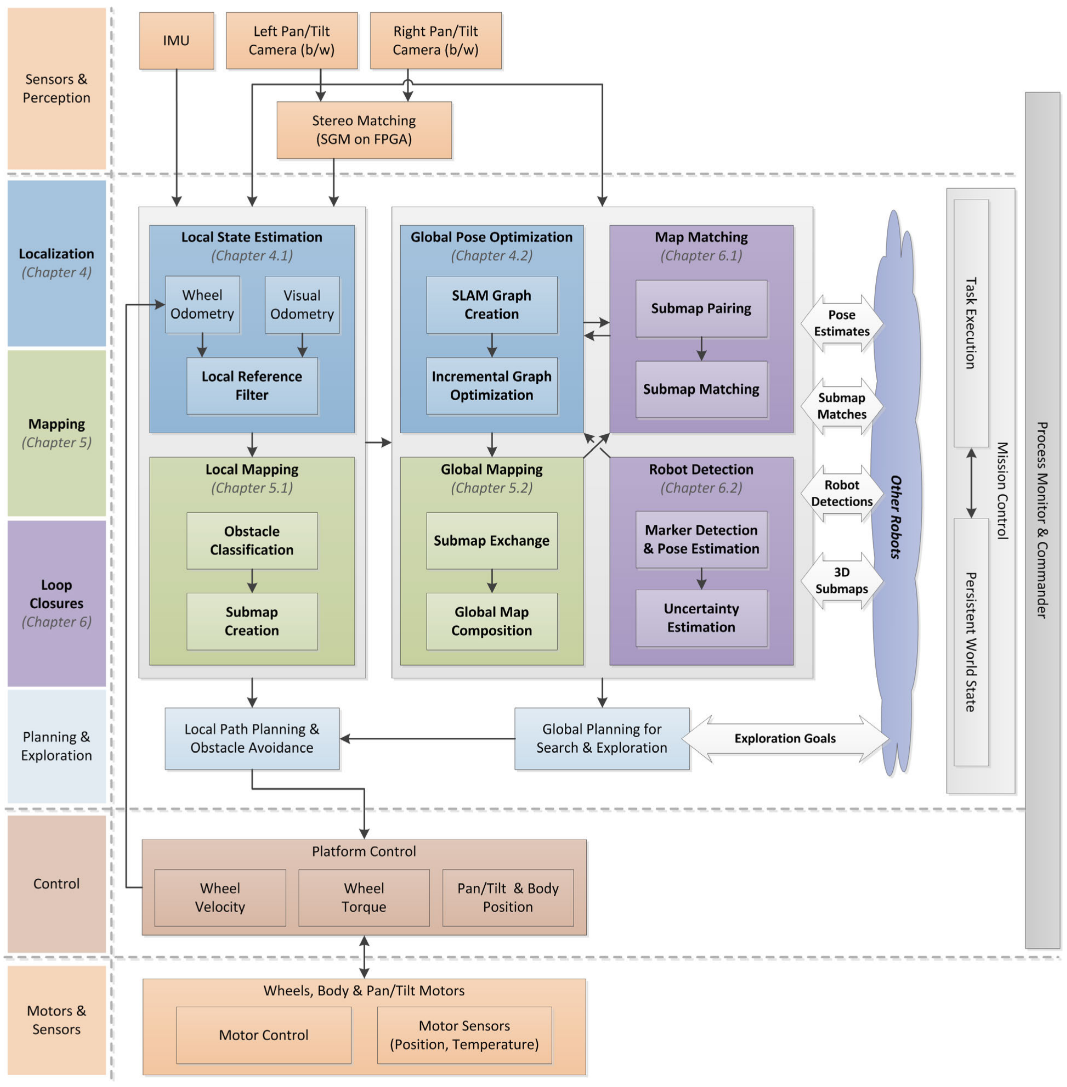


Figure 7.2: *IT architecture of autonomous robots, lander, and ground station. Dashed lines indicate optional components, such as those for robotic manipulation available only on LRU2. The right part shows three alternative types of communication links between lander and ground station.*

limited to converting and forwarding the communication from the rover in a format appropriate for the transportation over one of the following three different link types shown in the right part of Figure 7.2. First, for the quantitative evaluations of our on-board localization and mapping methods, we excluded the additional long-range communication challenges between a moon or planet and Earth, and thus simply connected lander and ground station via Ethernet. Second, at our demonstration at the IAC 2018 (see Section 9.4), this connection was replaced by a laser link between two optical terminals – a technology developed by the Institute of Communications and Navigation (KN) of the German Aerospace Center (DLR) to be deployed in future space missions for long-range and high-bandwidth communication (Calvo et al., 2019). Third, we demonstrated our system under further communication challenges by sending all data through a channel simulator that we employed to simulate the delayed, constrained, and lossy communication link between a ground station on Earth and robots residing far away on a planet or moon. At the SpaceBotCamp 2015 (see Section 9.2), it simulated a Moon-to-Earth connection with a 4 s round-trip delay, limited bandwidth, long periods of one-way communication, as well as total outages.

In order to build a truly autonomous system that can cope with such challenges, we perform all required computation on board the robots. Each robot features a powerful OBC for all the tasks that either process high-bandwidth data, like our vision-based perception, navigation, and search & exploration pipeline, or that pose high computational demands, for instance, manipulation planning. In addition, we execute low-frequency tasks, such as high-level mission planning, on the main computer. On each of our robots, the computationally intensive stereo matching runs on a FPGA extension board connected via PCI Express (PCI-E). The high-frequency, real-time control loops of platform and, on LRU2, manipulator controllers, run on an Intel Atom (LRUs) or BeagleBone Black (Ardea) computer board in order to separate them from the i/o-intensive image processing pipeline, thus allowing to satisfy the controllers' real-time requirements. The high-bandwidth connections between sensors and computer boards are realized via Universal Serial Bus (USB), FireWire and Ethernet, while the motor controllers are addressed via different bus systems suitable for deterministic real-time communication, such as Controller Area Network (CAN), Ethernet for Control Automation Technology (EtherCAT) or RS485.

In Figure 7.3 we present the system architecture of our robots. Its focus is on the integration of our localization and mapping modules with the robots' other software components as well as their sensors and actuators. While we based Figure 7.3 on the components relevant to the navigation capabilities of the LRU rovers, the overall architecture as well as the software integration of our SLAM framework is similar



Sensors & Perception
IMU
Left Pan/Tilt Camera (b/w)
Right Pan/Tilt Camera (b/w)
Stereo Matching (SGM on FPGA)
Localization (Chapter 4)
Mapping (Chapter 5)
Loop Closures (Chapter 6)
Planning & Exploration
Control
Motors & Sensors
Local State Estimation (Chapter 4.1)
Wheel Odometry
Visual Odometry
Local Reference Filter
Global Pose Optimization (Chapter 4.2)
SLAM Graph Creation
Incremental Graph Optimization
Map Matching (Chapter 6.1)
Submap Pairing
Submap Matching
Local Mapping (Chapter 5.1)
Obstacle Classification
Submap Creation
Global Mapping (Chapter 5.2)
Submap Exchange
Global Map Composition
Robot Detection (Chapter 6.2)
Marker Detection & Pose Estimation
Uncertainty Estimation
Pose Estimates
Submap Matches
Robot Detections
3D Submaps
Other Robots
Exploration Goals
Local Path Planning & Obstacle Avoidance
Global Planning for Search & Exploration
Platform Control
Wheel Velocity
Wheel Torque
Pan/Tilt & Body Position
Wheels, Body & Pan/Tilt Motors
Motor Control
Motor Sensors (Position, Temperature)
Task Execution
Persistent World State
Mission Control
Process Monitor & Commander

Figure 7.3: *Robot system architecture overview showing the integration of our localization and mapping modules with the other components of LRU*

on all of our robots.[2] In order to obtain depth data, they all feature a hardware-synchronized stereo camera system. We employ a FPGA-based implementation of SGM (Hirschmüller, 2008) to perform dense stereo matching under varying light conditions in both indoor and outdoor environments. We refer to Chapter 3 for details on the architecture of our central navigation components and their interfaces to local and global planning. Exploration goals can be shared between robots to coordinate their strategies to explore unknown areas. They are expressed in the robots' coordinate systems connected via estimates computed by our global multi-robot localization methods. Velocity commands for the whole platform generated by the local planning module are sent to the platform control. It converts them into commands for the individual motors, checks the motor sensor readings and, on the LRU platform, computes wheel velocities that are used to estimate wheel odometry. Autonomous task execution and monitoring on board the robots is implemented with *RMC advanced flow control (RAFCON)*, a powerful tool for visual programming via hierarchical state machines (Brunner et al., 2016). A combination of its task execution engine with a persistent world state allows the robots to execute complex autonomous behavior, including decisions that are based on their past and current sensor readings and estimates. We refer to Brunner et al. (2016, 2018) and Schuster et al. (2017) for more information on RAFCON and its integration on the LRU.

In order to establish the data flow between the various components, we employ three different middlewares, the first two being developed at the Institute of Robotics and Mechatronics (RM): *Links and Nodes* to satisfy the real-time communication requirements for control, *SensorNet* to distribute high-bandwidth vision data over shared memory, as well as the popular *Robot Operating System (ROS)* to connect high-level components, including our mapping pipeline. During an autonomous mission, the LRU typically runs more than 100 software processes in parallel, involving the execution of about 100 different libraries and components developed by more than 20 internal developers. In order to manage this complexity, we employ the process manager of *Links and Nodes* to monitor process output, manage runtime dependencies, and allow the compilation of mission settings by combining predefined modules and configurations. In addition, we contributed to the in-house development of the release and dependency management toolchain RM Package Management (RMPM) that we use to track and deploy consistent software versions to all of our robots as well as to mockup systems and simulations. Furthermore, in order to obtain reproducible builds and raise the quality of our software components, we use a continuous integration workflow, supported by automated builds and unit tests on

[2]In Figure 7.3 and its discussion, we omitted the components related to manipulation or manipulation-specific perception capabilities of LRU2 for brevity and refer to our article by Schuster et al. (2017) for further details.

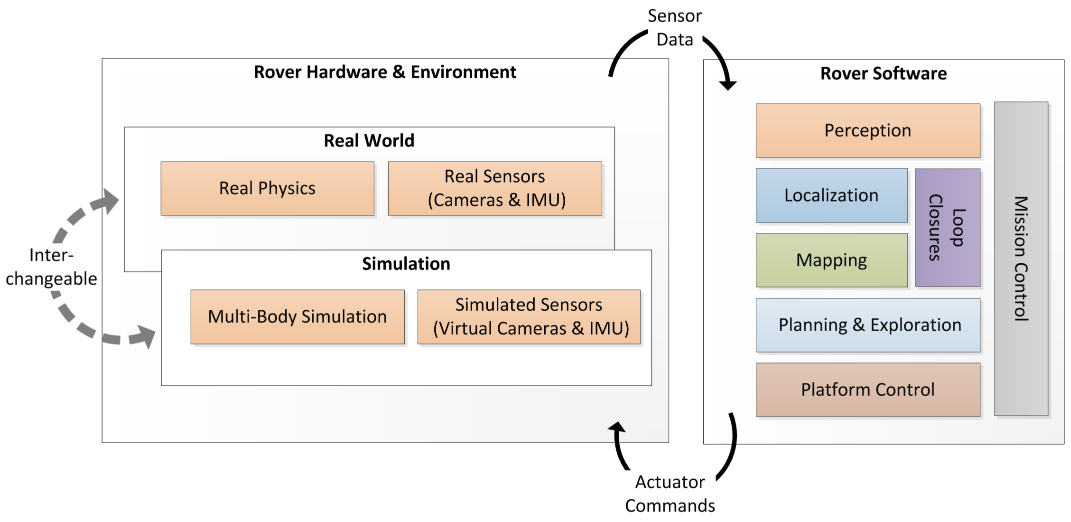
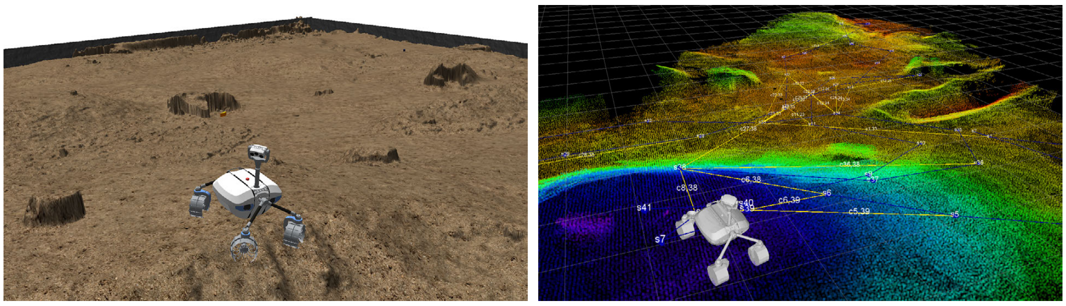


Figure 7.4: *Software-in-the-Loop (SiL) simulation of LRU: The rover software interacts either with the simulated or real sensors and actuators. A similar interface to the simulation and the real rover allow a switch between the two systems without any changes to the software components above this abstraction layer.*

all the code changes committed to our version control system. With this, we try to bridge the gap between the requirement for quick prototyping during research and the need for well-defined software development processes and standards, both to handle such a complex robot system in its current state as well as with regard to the direction of future space qualification.

7.3 Simulation Environment

In addition to real-world experiments, we used a Software-in-the-Loop (SiL) simulation of the LRU in order to test its stereo vision-based perception pipeline, our localization and mapping components, as well as modules for autonomous exploration and high-level task execution. For this, we employed the *RoverSimulation-Toolkit* (Hellerer et al., 2016). It is a multi-body physics simulation and high-fidelity visualization that features virtual sensors such as color and depth cameras as well as IMUs. In Figure 7.4, we visualize the SiL simulation concept: Similar software interfaces between our perception, navigation, and high-level modules and the real rover as well as the simulation allow a switch between the two systems without any further adaption. We describe this integration in more detail in our conference paper by Hellerer et al. (2016) and journal article by Schuster et al. (2017).



(a) *LRU in the simulated environment* ***(b)*** *3D point cloud map from our SLAM system*

Figure 7.5: *Simulation of the LRU in the SpaceBotCup 2013 environment*

As we first introduced the simulation in preparation for the SpaceBotCamp 2015 (see Section 9.2), we based our simulated environment on a publicly available rough-terrain 3D model of the SpaceBotCup 2013 challenge arena. It had been recorded via high-precision LIDARs and published by the team of the University of Bonn, Germany (Schadler et al., 2014; Holz and Behnke, 2014). In Figure 7.5, we show the LRU in this virtual rough-terrain environment, as well as the corresponding 3D reconstruction created by the LRU's image processing and mapping components. While a simulation cannot replace all tests with a real robot due to trade-offs between runtime and precision, simplified models and approximations, as well as different types of sensor noise, it nonetheless allows to test many components and their interplay early on and with less effort compared to real-world tests. This is, in particular, invaluable to support a continuous system integration, including initial interface and communication tests between more than 100 software components as well as regular integration steps to check if any changes on individual modules influence or break the overall system behavior.

7.4 Trajectory Ground Truth

While in the aforementioned simulation environment, ground truth trajectory data is trivial to obtain, it can be a more challenging task in real-world environments. We used a variety of tracking systems to acquire ground truth pose and position data in order to perform the quantitative evaluations of our SLAM system in indoor and outdoor environments presented in Chapter 8.

Indoors, we conducted experiments in two lab environments that feature an *ART-TRACK2* (Advanced Realtime Tracking GmbH, 2014) and a *VICON* (Vicon Motion Systems Ltd, 2019) tracking system respectively. Both are based on a set of ceiling-mounted cameras that track artificial markers in the infrared spectrum. These consist of unique three-dimensional configurations of small retro-reflective balls, allowing to identify our robots as well as to estimate their position and full orientation at frequencies of up to 250 Hz. The tracking cameras cover areas of approx. $3\,\mathrm{m} \times 4\,\mathrm{m}$ in the smaller lab and approx. $11\,\mathrm{m} \times 6.5\,\mathrm{m}$ in the larger one. For several of our larger-scale experiments, we thus could solely evaluate the partial robot trajectories for which ground truth was available.

For outdoor experiments, we used two different systems to acquire ground truth. First, we acquired ground truth position data through a Leica total station (tachymeter) that tracks a prism attached to the robot (see Figure 7.1(a)). This method yields a high tracking accuracy but at all times requires a free, undisturbed line of sight between tachymeter and prism. It thus limits the experiment setup and does not scale well to multi-robot systems. Second, we integrated an optional Differential GPS (DGPS) receiver on each of our LRU robots (see Figure 7.1(d)). While we recorded data during several experiments of the ROBEX Moon-analogue test campaign, we did not use it for quantitative evaluations in this thesis. DGPS scales well to multi-robot systems. Its accuracy, however, heavily depends on several factors such as the number of satellites in direct line of sight, the antenna and its placement w. r. t. electromagnetic interference, etc., typically being significantly worse than tachymeter measurements. In some of our experiments it was approximately ten centimeters, thus rendering it insufficient to evaluate small- to medium-scale trajectories that have little drift. Both tachymeter and GNSS systems such as DGPS acquire only position ground truth as, with a single prism or receiver per robot, they cannot determine its orientation. As our pose estimation and the outdoor tracking systems use different frames of reference, we initially computed a spatial (rigid transformation) and temporal alignment. We employed a least-squares error minimization based on the first 3 – 4 m of the corresponding trajectories, assuming that the drift of the filter within that distance is negligible in comparison to the full trajectories.

Chapter eight

Experimental Evaluation

In this chapter, we present an evaluation of our contributions to methods for multi-robot localization and mapping. For this, we conducted several series of experiments on the robotic systems, which we introduced in Chapter 7. Our goal for the set of experimental evaluations, taken as a whole, is to cover the challenges on SLAM for planetary exploration that we identified in Section 2.1. We structure the set of individual evaluations regarding four central aspects: First, we present our contributions to methods for online loop closure generation with map matches and visual marker detections (Section 8.1). Second, we evaluate our 6D SLAM framework in single-robot experiments, in particular regarding the combination of graph optimization and map matching as well as our novel integration of local reference filter estimates (Section 8.2). Third, we extend our experiments to multi-robot setups with two rovers, featuring different robots and camera setups (Section 8.3). Fourth, we analyze and discuss the computational resources required for our multi-robot experiments, including the requirements on communication bandwidth, processing power, and memory as well as the runtimes of the key components in our SLAM system (Section 8.4). To conclude this chapter, we summarize and discuss our findings, relate them to the aforementioned challenges of planetary exploration, and point to further evaluations in related publications (Section 8.5).

It is important to note that in any trajectory plots and error statistics presented in the following sections, we always consider only the sequentially logged estimates of our filter and SLAM components that could be computed with the sensor input and computational resources available up to that particular point in time. This means, we do not present any SLAM trajectories that have been fully optimized after the experiment with more data available than the robot could have had at each point of its trajectory. We deem this evaluation criterion suitable regarding the main application area for our methods, i. e., the support of robot autonomy during planetary exploration missions. At each individual point in time, any autonomous robot system has only the past and current measurements and estimates available for its online decision making.

8.1 Loop Closure Generation

Loop closures are essential for any data fusion of relative measurements that goes beyond a purely incremental approach. In the context of our SLAM framework, we are concerned with the geometry-based recognition of previously visited locations in semi- and unstructured environments as well as with the marker-based visual pose estimation of artificial landmarks and of other robots in multi-robot teams. We first evaluate our stereo vision-based obstacle classification with regard to its suitability as input for loop closure detection (Section 8.1.1). Second, we analyze the accuracy of our map matching method, which is based on the 3D geometry identified during obstacle classifications, and evaluate the benefits of reducing its dimensionality from 6D to 4D (Section 8.1.2). Third, we evaluate the impact of our uncertainty estimation for marker-based detections on the overall localization accuracy (Section 8.1.3).

8.1.1 Stereo Vision-Based Obstacle Mapping

In Section 5.1.1, we introduced our stereo error-adaptive method for local obstacle classification. In the following evaluation, we demonstrate its suitability to identify distinctive 3D geometry that can be used to support loop closure generation in SLAM systems in unstructured outdoor as well as semi-structured indoor and mixed environments. For this, we employed a Rao-Blackwellized Particle Filter (RBPF)-based single-robot SLAM framework featuring 3D pose (x, y, yaw) estimation and 2D occupancy grid mapping. We developed the 3D SLAM framework as a predecessor to the graph-based 6D SLAM methods presented at the center of this thesis. In addition to evaluating our obstacle classification, we use it as a baseline in comparison to the localization accuracy of our novel methods in Section 8.2.2.

The content of this section is based on our methods and results that we first published in our conference paper by Brand et al. (2014). In the following paragraphs, we provide a short summary of additional processing steps of our RBPF-based SLAM that go beyond or differ from the negative edge detection and stereo error-adaptive obstacle classification that we discussed in detail in Section 5.1.1. Our obstacle classification algorithm, as presented by Brand et al. (2014), includes an additional slope estimation in local neighborhoods using plane fitting, a computationally expensive step that we later dropped due to its marginal additional benefit over using our step detection alone. As post-processing on the resulting obstacle maps, we identify and filter potentially erroneous hypotheses for small individual obstacles based on the traversability values of their neighborhood. This was necessary to mitigate the effects of stereo reconstruction noise in combination with a map resolution of 3 cm, instead of 5 cm in later experiments, and is based on the assumption that relevant obstacles encompass more than a single map cell.

For the integration of our obstacle classification data into a 3D SLAM framework, we performed additional post-processing steps. The mapping pipeline discussed in this section does not yet include an aggregation of data into distinct submaps. Instead, as a predecessor to our submapping approach, we integrated multiple subsequent obstacle classification results using a probabilistic three-dimensional voxel grid within a sliding temporal window of 5 s, using the freely available open-source OctoMap library (Hornung et al., 2013). Similar to our submapping approach, this aggregation of data is based on the filter results and on the corresponding assumption that these are sufficiently accurate over the aforementioned short period of time. This approach has three major advantages in comparison to simply passing the classification results for individual stereo image pairs to the SLAM framework. First, stereo cameras typically have a small horizontal field of view, in particular compared to LIDAR systems. Merging the data from multiple measurements during movements of the robot yields partial maps that cover larger view angles or areas than individual measurements respectively. These are less likely to exhibit geometric ambiguities and thus improve loop closure detection. Second, the aggregation of measurements allows us to run the subsequent SLAM components at a lower frequency than the preceding stereo reconstruction and obstacle classification, thus reducing their computational effort. Third, the probabilistic integration over time acts as an additional noise filter as, by eventually applying a threshold, only obstacles observed in several individual measurements are represented in the final output.

For an introduction to the concept of Rao-Blackwellized Particle Filters (RBPFs), we refer to Section 2.2.3. We decided to use the freely-available implementation of the widely-used and well-optimized GMapping algorithm (Grisetti et al., 2007).

It uses adaptive resampling and a scan matcher to improve its proposal particle distribution, which allows an efficient online estimation of accurate maps with low numbers of particles (e. g., 30 for small maps). However, the GMapping algorithm had originally been developed for 2D LIDAR systems and thus is based on their respective measurement model. In order to provide compatible input data, we sample virtual 2D LIDAR scans from our gravity-aligned local obstacle maps. For this, we place the origin of our virtual sensor at the rotational center point of the robot and cast rays with an angular resolution of 0.6° to the nearest obstacles at any height level. While we are aware of the loss of information involved in this operation, it allowed us to use a well-tested SLAM solution as part of our initial version of a mapping framework. For more details and further evaluations regarding these additional processing steps, we refer to our publication by Brand et al. (2014).

Experimental Setup

For our experiments, we manually controlled the Pioneer 3-AT (P3AT) robot in three different scenarios:

1. *Outdoor:* testbed containing different types of gravel, larger stones as well as an artificial crater. Its steep slopes only allow P3AT to enter from one side. In Figure 8.1, we show a photo of our robot looking into the crater and present an aerial overview of the test site in Figure 8.2. We performed experiments with two different camera configurations with narrow-angle and wide-angle lens setups, 3 cm grid map resolution, and a RBPF with 30 particles.
2. *Indoor:* lab environment with rooms and hallways, as visible in resulting map visualized in Figure 8.5. We used the narrow-angle lens setup only, 3 cm grid map resolution, and 30 particles.
3. *Mixed Indoor & Outdoor:* scenario consists of parts of the indoor scenario as well as a large loop around the lab building, as visualized in the resulting map in Figure 8.6. We used the narrow-angle lens setup only, 5 cm grid map resolution, and 250 particles necessary due to the significantly larger map and loop sizes compared to the other two scenarios.

In the outdoor testbed, we acquired position ground truth for the whole robot trajectory with a tachymeter. In the indoor and mixed indoor & outdoor experiments, we started and stopped the experiments with the robot at the same position and orientation, which we marked on the floor. We then compared the differences between start and final position in our filter and SLAM estimates.

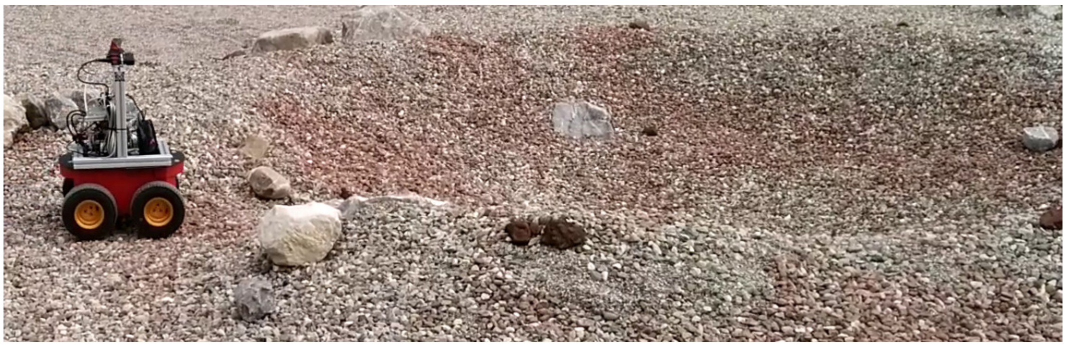


Figure 8.1: *Pioneer 3-AT (P3AT) robot looking into the crater of our outdoor testbed*

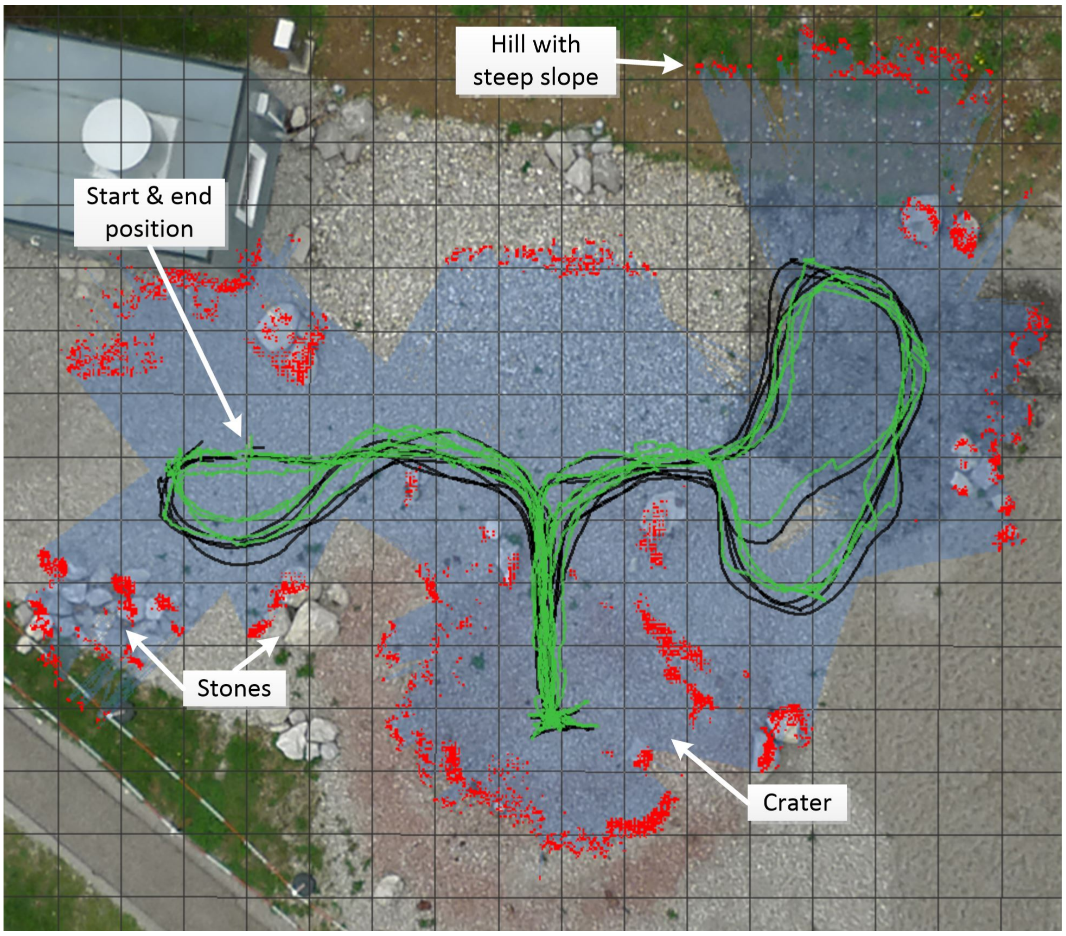
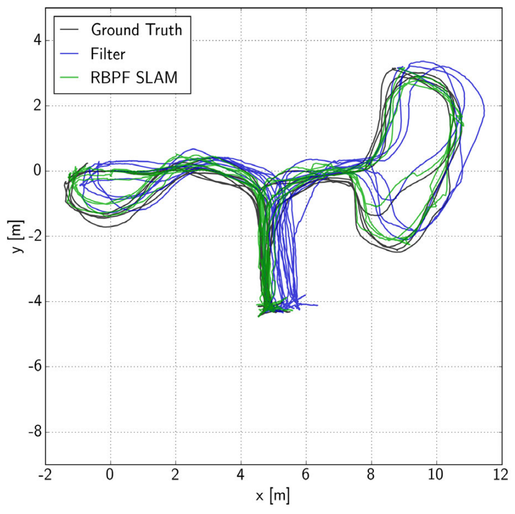
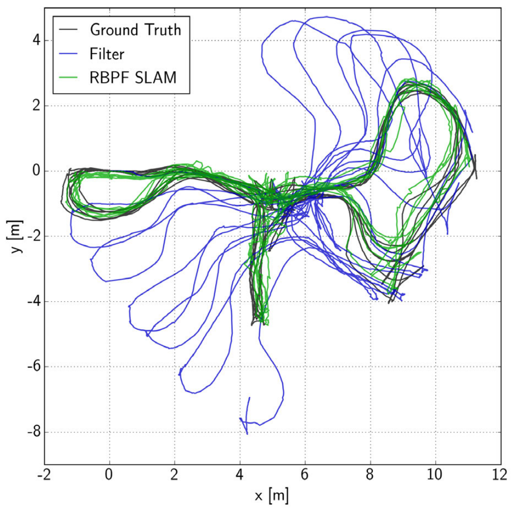


Figure 8.2: *Top-down view on our outdoor experiment: test site with manually overlaid map showing the estimated obstacles (red) and free space (blue). Most obstacle classifications either indicate large stones or steep slopes, which are hard to recognize in the aerial image. The black trajectory shows tachymeter ground truth, the green trajectory our RBPF SLAM estimate. The grid has a cell size of* $1\ \mathrm{m}^2$.

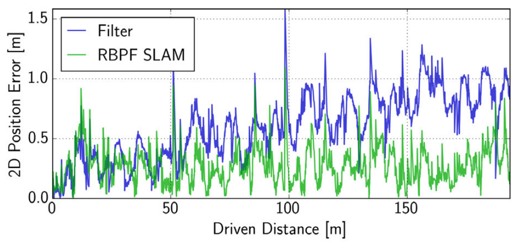


(a) *Narrow-angle camera setup*

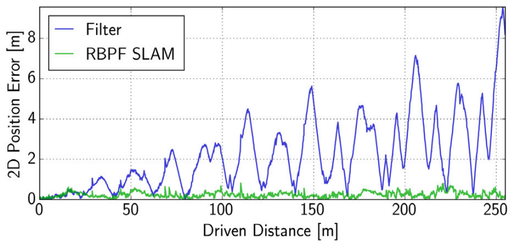


(b) *Wide-angle camera setup*

Figure 8.3: *Comparison of the filter, RBPF SLAM, and ground truth trajectories driven by P3AT in our outdoor experiments*



(a) *Narrow-angle camera setup*



(b) *Wide-angle camera setup*

Figure 8.4: *Comparison of 2D position errors w. r. t. ground truth for the filter and RBPF SLAM trajectories driven by P3AT in our outdoor experiments (see Figure 8.3 for the trajectories)*

Results and Discussion

For evaluation, we compare the ground truth to the trajectory estimates by our 3D SLAM pipeline as well as to those of the Extended Kalman Filter (EKF) fusing visual odometry and IMU data only. In all three scenarios, the RBPF-based SLAM algorithm was able to exploit the results of our obstacle classification method to produce loop closures that significantly reduced the position errors compared to the filter results, thereby eliminating the drift in the estimates.

In Figure 8.2, we present the final obstacle grid map for one of our experiments in the outdoor testbed, shown as an overlay on top of an aerial image together with the robot's estimated and ground truth trajectories. During each loop, the robot entered the small crater depicted in Figure 8.1 that is located in the bottom area of the map.

We conducted experiments with both a narrow-angle and a wide-angle lens stereo camera setup, driving four and six loops in the outdoor testbed respectively. In Figure 8.3, we present the robot trajectories for both experiments and plot their respective positional errors compared to ground truth in Figure 8.4. We list the absolute position errors after each loop as well as the mean errors regarding the full trajectory and their standard deviations in Table 8.1 and Table 8.2 for the narrow-angle and wide-angle experiment respectively. In order to evaluate the influence of our obstacle mapping method presented in Section 5.1.1, we ran different variants of our SLAM algorithm on the same outdoor datasets: In addition to comparing the filter results to the full RBPF SLAM, we performed two evaluations, separately omitting the negative edge detection and the stereo error-adaptive obstacle classification.

The results show the expected positional drift of the estimates computed by the filter, which is only suitable as a local estimation method. The absolute position errors plotted in Figure 8.4 show a periodic behavior. It is related to the profile of the robot's trajectories, which consist of loops that frequently cross previously visited locations. This effect is caused by accumulated positional errors canceling out one another, in particular when the estimated trajectory crosses the ground-truth one near the center of the map, as shown in Figure 8.3. The filter estimates are significantly less accurate for our wide-angle stereo camera setup, in particular w. r. t. the *yaw* angle, as can be observed in Figure 8.3(b). We attribute this effect to the lower angular resolution and the more complex lens distortion of the wide-angle setup. Both can significantly degrade the accuracy of the camera calibration as well as of the visual odometry estimation. Our full SLAM system is able to incorporate a sufficient number of loop closures to correct for these errors, achieving similarly high accuracies with mean trajectory errors of 0.22 m for both camera setups. The final position deviations are below 0.08 % and 0.06 % w. r. t. the full trajectory length for the narrow-angle and wide-angle setup respectively. As can be observed in Table 8.1 and Table 8.2, our two specific obstacle classification steps, the negative edge detection and stereo error-adaptive obstacle classification, have a positive effect on the mean error values in both experiments.

In Figure 8.5 and Figure 8.6, we present the trajectories and maps estimated for our indoor as well as our mixed indoor and outdoor experiments. For the indoor experiment with a total trajectory length of 107.7 m, the final position deviation for the SLAM estimate is 0.06 m compared to 0.38 m for the filter. For the mixed indoor and outdoor experiment with a total trajectory length of 220.1 m, the difference is even larger with a final position deviation for the SLAM estimate of 0.05 m compared to 1.66 m for the filter. With these experiments, we demonstrated the applicability of our obstacle-based SLAM to indoor and mixed environments. Indoors, untextured

Round	1	2	3	4		
Driven distance [m]	49.1	96.6	146.5	194.2		
	Position error [m]				μ [m]	σ [m]
Filter only	0.36	0.50	0.71	0.82	0.58	0.28
RBPF SLAM	0.11	0.16	0.09	0.15	**0.22**	**0.13**
w/o Negative edge detection	0.40	0.23	0.25	0.10	0.32	0.22
w/o Adaptive obstacle classification	0.19	0.37	0.34	0.36	0.49	0.41

Table 8.1: *Comparison of localization errors in our outdoor scenario with the narrow-angle camera setup (50.6 °). The first four columns indicate the absolute 2D position errors at the end of each round, the final two columns show their mean (μ) and standard deviation (σ) over the full trajectory.*

Round	1	2	3	4	5	6		
Driven distance [m]	35.0	74.1	115.5	150.2	207.6	255.0		
	Position error [m]						μ [m]	σ [m]
Filter only	1.02	2.07	3.81	4.69	6.19	8.19	2.18	1.99
RBPF SLAM	0.18	0.15	0.15	0.21	0.24	0.15	**0.22**	**0.13**
w/o Negative edge detection	0.19	0.24	0.25	0.22	0.14	0.26	0.37	0.24
w/o Adaptive obstacle class.	0.24	0.20	0.15	0.24	0.23	0.22	0.36	0.18

Table 8.2: *Comparison of localization errors in our outdoor scenario with the wide-angle camera setup (112.8 °). The first six columns indicate the absolute 2D position errors at the end of each round, the final two columns show their mean (μ) and standard deviation (σ) over the full trajectory.*

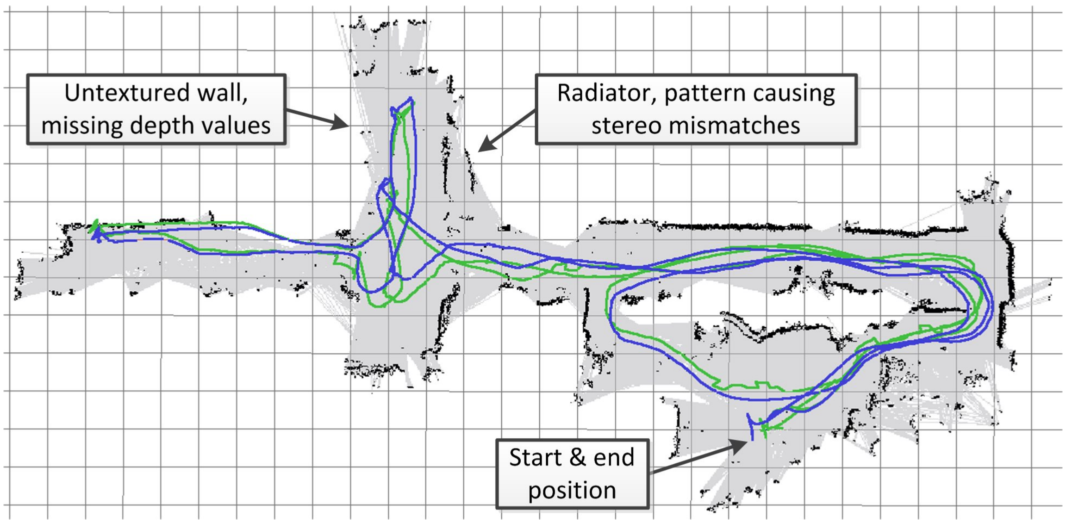

Untextured wall, missing depth values
Radiator, pattern causing stereo mismatches
Start & end position

Figure 8.5: *Indoor experiment: top-down view on occupancy grid map with obstacles (black) and free space (grey) at a resolution of* 3 cm. *The blue trajectory shows the filter only estimate, the green trajectory the RBPF SLAM estimate. The grid has a cell size of* 1 m^2.

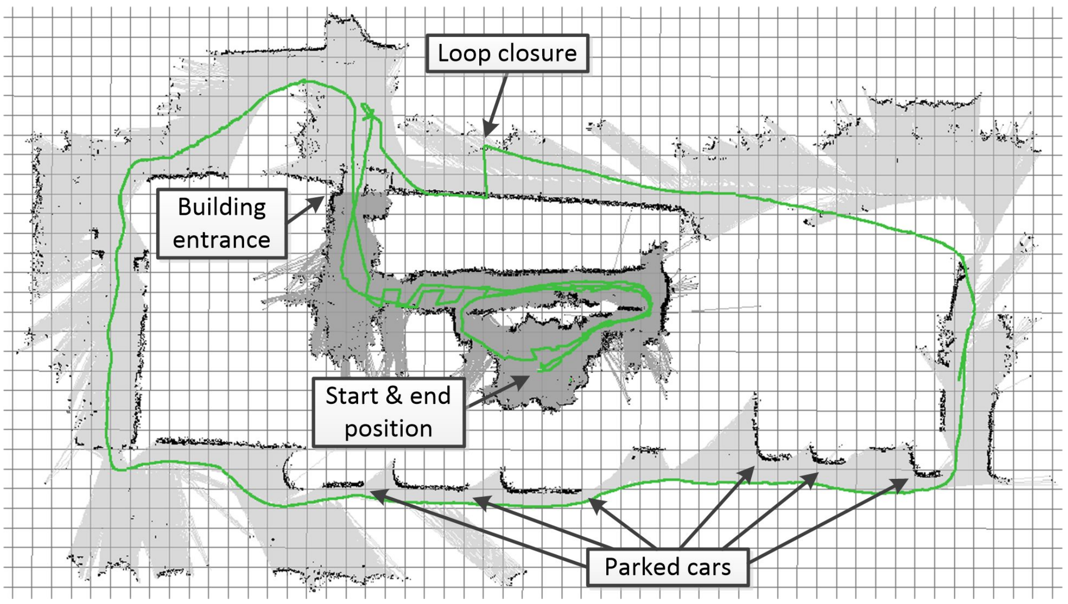


Figure 8.6: *Mixed indoor and outdoor experiment: top-down view on occupancy grid map with obstacles (black), free space outdoors (light grey) and free space indoors (dark grey) at a resolution of* 5 cm. *The green trajectory shows our RBPF SLAM estimate. The grid has a cell size of* 1 m^2.

objects like white walls, reflective surfaces like windows, as well as regular patterns like radiators challenge stereo reconstruction algorithms and thus lead to either missing depth values or mismatches that can be observed as absent or dislocated obstacles in our maps. While these challenges are less likely to occur in our primary application scenario of planetary exploration, our algorithms are nonetheless able to identify sufficiently many obstacles correctly to achieve an overall SLAM localization accuracy similar to the outdoor experiments. In our mixed indoor and outdoor experiment, the robot started indoors, exited the building to drive a large loop outdoors around it and re-entered it to return to its start position. Our SLAM system succeeded in identifying the large loop closure, as we indicated in the top part of Figure 8.6. It, however, required a larger number of particles to represent the space of possible trajectory and map hypotheses that is significantly larger than in the other experiments. With the successful mixed indoor and outdoor experiment, we demonstrated the robustness of our stereo camera-based system. It was able to cope with varying light conditions that ranged from artificial lights in the building's hallway to looking directly into the bright sun outside. Despite these challenging conditions, the robot could acquire sufficiently dense depth data, classify obstacles and detect loop closures therein using the same set of parameters in both indoor and outdoor environments.

In all four experiments with their varying camera setups and scenarios, our SLAM system was able to cope with noisy stereo data and could compensate for drifting filter estimates. We demonstrated the individual benefits of our negative edge detection and stereo error-adaptive obstacle classification that significantly improve the accuracy of our estimates. While these experiments have been conducted with a 3D SLAM system preceding our graph-based 6D SLAM framework, they demonstrate the suitability of our obstacle classification results for loop closure detection, which we exploit in our map matching method.

8.1.2 4D Map Matching

In this section, we evaluate our map matching algorithm for metric place recognition that we introduced in Section 6.1. It computes the relative transformation between two submaps based on their 3D geometry, augmented with the obstacle classification results discussed in the previous section. In the evaluation of our algorithm, we consider two aspects: First, we analyze the accuracy of its resulting match transformations compared to ground truth. Second, we compare our novel 4D matching method to a full 6D matching, which we used in our previous work (Schuster et al., 2015; Brand et al., 2015).[1] We first published the content of this section in our journal article by Schuster et al. (2018).

Experimental Setup

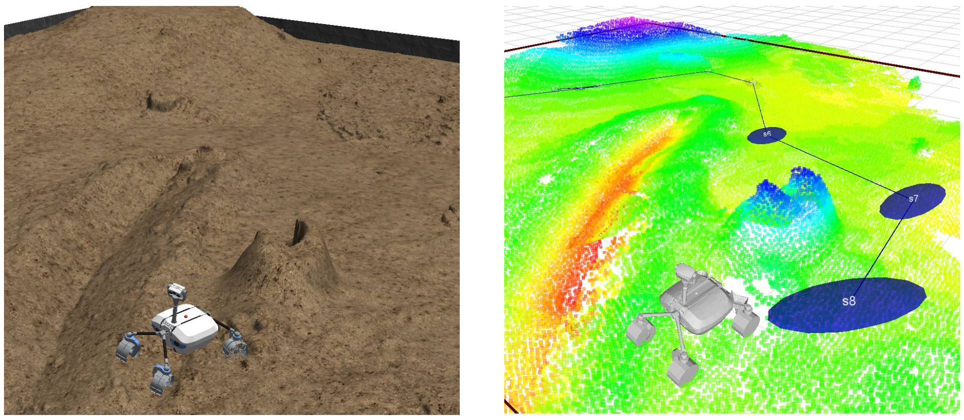
We evaluate our algorithm in two series of in total 53 single-robot experiments in simulated as well as real-world environments, featuring two different scenarios. In both, the LRU used the 3D voxel-grid maps generated by our mapping pipeline to autonomously explore a previously unknown area based on a maximization of information gain and map quality (Lehner et al., 2017).

- *High-Fidelity Simulation of the LRU in Rough Terrain*
  We simulated a model of the LRU using the *RoverSimulationToolkit* (Hellerer et al., 2016), a multi-body physics simulation and high-fidelity visualization. The virtual environment for our experiments is based on the SpaceBotCamp 2013 arena and features a deep ridge, a steep ramp as well as several large rocks. We provide more details on the simulation of LRU in Section 7.3.

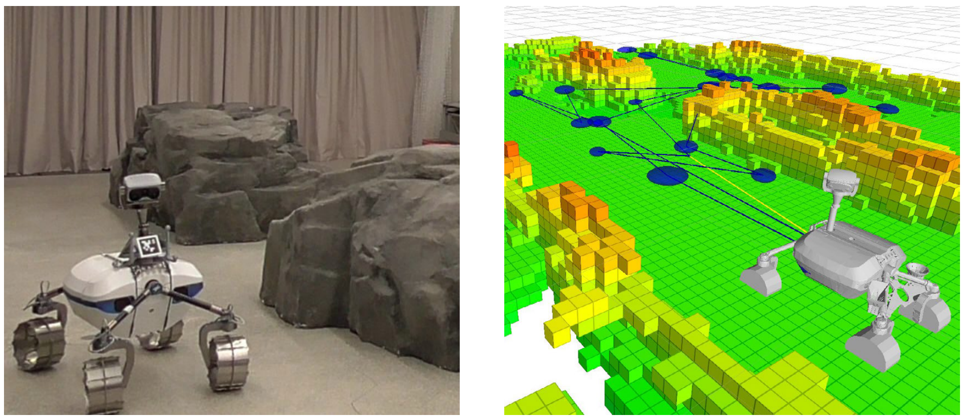
- *Real LRU Indoor Experiments*
  The experiments with one of our real LRUs had been conducted in an exploration scenario in our lab, featuring large artificial rocks as obstacles for rover navigation. Ground truth pose data for the LRU was recorded through our ceiling-mounted tracking system, covering the complete experimental area of approx. $11\,\mathrm{m} \times 6.5\,\mathrm{m}$.

Our datasets consist of 40 experiments in the simulated environment and 13 experiments with our real LRU, exploring an area of on average approx. $197\,\mathrm{m}^2$ and $72\,\mathrm{m}^2$ in each of them respectively. In Figure 8.7, we give an impression of both experimental setups. While the lab experiments are important to test our algorithm under real conditions, in particular w. r. t. the sensors' error characteristics, a high-fidelity simulation allows to conduct a larger number of experiments with perfect ground truth data being available in larger environments. The localization and mapping components are equal to the real setup and we employed the same parameters for map creation and matching in both scenarios.

[1]Note that since the publication of our previous work (Schuster et al., 2015; Brand et al., 2015), several other, mostly minor aspects of the matching algorithm have changed. For the evaluation presented in this section, we consistently employed our novel version, as described in Section 6.1 of this thesis, and solely vary the parts related to 4D and 6D matching in order to produce comparable results.



(a) *Screenshot from LRU simulator and point cloud map*

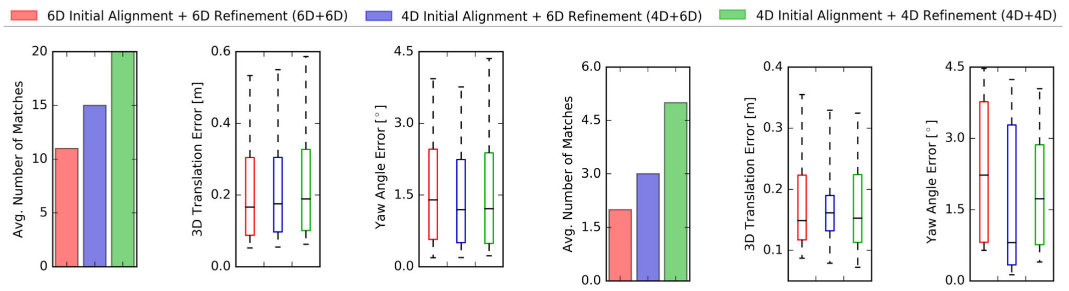


(b) *Photo from real-world experiment and voxel-grid map*

Figure 8.7: *Impressions from exploration experiments used for the evaluation of our 4D submap matching*

Results and Discussion

We compare three different variants of our map matching algorithm, demonstrating the impact of both the 4D initial alignment (see Section 6.1.2) and 4D refinement step (see Section 6.1.2) w. r. t. their match accuracy and number of loop closures. As baseline, we use a *6D initial alignment and 6D refinement (6D+6D)*, similar to our previous work (Brand et al., 2014), with an outlier threshold on *roll* and *pitch* of 10° for the initial alignment step and 1° after the refinement. First, we replace only the initial alignment with our novel 4D method, leading to the combination *4D initial alignment and 6D refinement (4D+6D)*. This allows a separate evaluation of the impact of our changes on the two processing steps. The third variant is our proposed method, using both a *4D initial alignment and 4D refinement (4D+4D)*. We excluded the combination of a 6D initial alignment and 4D refinement (6D+4D) as its initial alignment errors in *roll* and *pitch* could never be corrected by a 4D refinement.



***(a)** Results for our 40 rough-terrain simulator experiments*

***(b)** Results for our 13 real-world lab experiments*

***Figure 8.8:** Average number of submap matches per dataset and distribution of 3D translation and* yaw *angle errors w. r. t. ground truth per match (line: median, box: from lower to upper quartile, whiskers:* 10^{th} *to* 90^{th} *percentile of values, as individual large outliers will be filtered by robust estimators during graph optimization, see Section 4.2.3)*

In Figure 8.8, we present the number of matches as well as the distribution of errors of the estimated relative submap transformations compared to ground truth on both the simulated and the real-world experiments. As we enforce $roll$ and $pitch$ to be zero or smaller than $1°$ for the 4D and 6D cases respectively, we only consider 3D translation and the error in yaw in our comparison.

First, we analyze the overall match accuracy of our algorithm. Figure 8.8 shows that the majority of match error values are in the range of $[0.1\,\text{m}; 0.3\,\text{m}]$. This fits the expected matcher accuracy, which is limited by noisy and imprecise stereo vision-based input data: For the LRU's camera system, the average per-pixel distance errors regarding points in individual submaps reach up to $0.12\,\text{m}$, as derived in Section 6.1.1. It is important to note that the distributions of match errors are similar for all three variants of the algorithm, which is to be expected as we apply similar outlier filters during the matching process.

Second, we analyze the number of matches for the three different variants of the matching algorithm. They greatly depend on the scenario, as the opportunities to generate map matches depend on the environment as well as on the robot's trajectory. Thus, we only average over the number of matches for similar datasets. It is lower for the real-world experiments due to their smaller size and shorter robot trajectories compared to the simulated ones. For both scenarios, Figure 8.8 shows a rise in the number of matches for each of the 4D matching steps. The benefit of our novel method stems from this increased number of loop closure constraints for the graph SLAM. It can improve the robustness of global graph optimization, in particular when the number of loop closures in a scenario is small, as it is for all of our experiments. The influence of individual erroneous data associations can be mitigated by a large number of correct matches, or even eliminated by robust estimation methods in case they contradict the majority of measurements and other estimates (Agarwal et al., 2013).

The increased number of matches of our novel 4D map matching methods comes at no cost in accuracy compared to the 6D approach. In addition, both methods have a comparable computational complexity. Thus, the 4D matching is to be preferred. Its superiority can be explained as follows: In our novel 4D matching, we solely generate 4D hypotheses that, by definition, satisfy our constraints on *roll* and *pitch*. In contrast, during 6D matching, full 6D hypotheses are computed and then filtered w. r. t. these constraints. Both, the Random Sample Consensus (RANSAC) computed for each Hough3D bin during initial alignment, as well as the Iterative Closest Point (ICP) during refinement, generate a single best hypothesis each. Using 4D constraints at this stage thus forces the optimization to find a solution satisfying them, instead of running the risk of generating a single, unconstrained 6D solution that fits better to the respective cost function but will be removed when filtering implausible hypotheses during post-processing.

8.1.3 Uncertainty Estimation for Marker-Based Detections

In Section 6.2, we presented a novel method to estimate the uncertainty of 6D pose estimates for planar marker detections. We use these for the observation of artificial static landmarks as well as to generate inter-robot loop closure constraints via robot detections. Dealing with propagated sensor noise and various types of error sources along the camera-based detection and estimation pipeline is crucial to get accurate global estimates. In this section, we evaluate the impact of our novel uncertainty estimation on the accuracy of global pose optimization via graph SLAM by comparing the resulting trajectory estimates to those computed with simpler uncertainty models.[2]

Experimental Setup

We conducted an experiment with our LRU in the outdoor testbed described in the previous section, featuring rough terrain with large stones and a small crater as pictured in Figure 8.9. We distributed five AprilTag markers (size: 12 cm) in the environment, either by lying them on the ground or leaning them against stones.

[2]Similar to Section 6.2, this experimental evaluation is based on results acquired and first analyzed as part of the master's thesis work by Vetter (2015), which was supervised by Christoph Brand and me, and features a rewritten, updated description, and extended discussion of our findings.

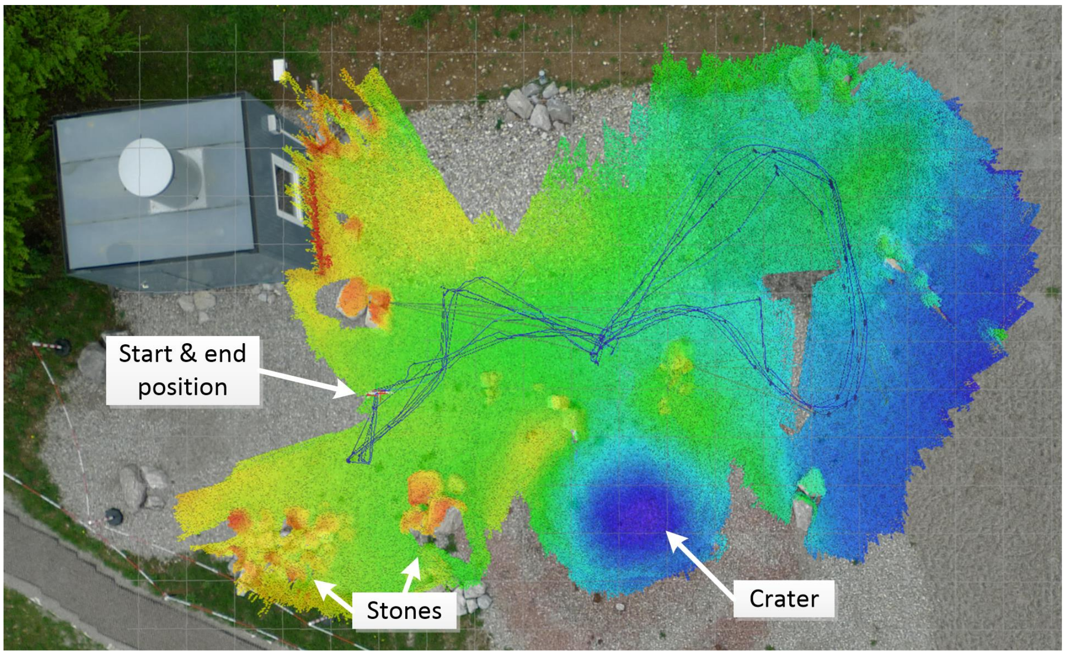


Figure 8.9: *Top-down view on experiment site with manually overlaid height-colored map*

They served as globally identifiable static landmarks, the detections of them being added to the SLAM graph. We did not use any other types of loop closure constraints in this experiment, as our goal is to evaluate the impact of our novel uncertainty estimation method for marker detections on the results of global optimization. We manually controlled our robot to drive three loops in the outdoor terrain, passing the locations of the markers so that they were visible in the camera images, and acquired ground truth position information with a tachymeter (total trajectory length: 106 m).

Results and Discussion

For evaluation, we compare our approach against two simpler models, the first one assuming a constant uncertainty independent of view angle and distance, and the second one assuming a quadratic scaling of the uncertainty with the distance d between camera and marker. For the constant model, we assume a Gaussian uncertainty represented by a diagonal covariance matrix Σ for the variances of $(x, y, z, roll, pitch, yaw)$

defined as

$$\Sigma_{constant} = \begin{pmatrix} 0.0001 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0001 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0005 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0005 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0005 \end{pmatrix} \tag{8.1}$$

corresponding to standard deviations of 1 cm and 1.3° for the translational and rotational degrees of freedom respectively. For the quadratic model, we add a term for the quadratic dependence on the distance d:

$$\Sigma_{quadratic} = \Sigma_{constant} \, (1 + d)^2 \tag{8.2}$$

Our own approach consists of two parts, a lookup of precomputed values and an online simulation of corner detection errors as described in Section 6.2.3. To analyze their individual contributions, we further compare our proposed combined solution to using each of its parts individually.

Those of the methods that did not include our online simulation of corner detection errors significantly underestimated the uncertainty of certain marker poses with ambiguous orientations. The corresponding large rotational pose estimation errors lead to a corruption of the graph optimization results to a degree that rendered them unusable. In order to get comparable results for our evaluation, we thus ran our online estimation in parallel and, for all methods, excluded marker poses with estimated uncertainties above 75°. An alternative for real applications with the other methods would be a general limitation of the maximum distance and view angle for marker detections in order to exclude potentially ambiguous cases. This, however, also means rejecting a large number of valid detections.

We present the SLAM trajectories generated with the different uncertainty estimation methods in Figure 8.10 and their corresponding 3D position errors in Figure 8.11. In Table 8.3, we provide a summary of the mean, maximum and average errors. As the trajectories and error values for our combined method and its "precomputed only" variant are almost identical, we omitted the latter from the plots. This, however, does not mean that our online corner error simulation is unnecessary – to the contrary – it is essential to asses the possibility of large error values for each individual detection. Our aforementioned filtering of all detections with estimated uncertainties above

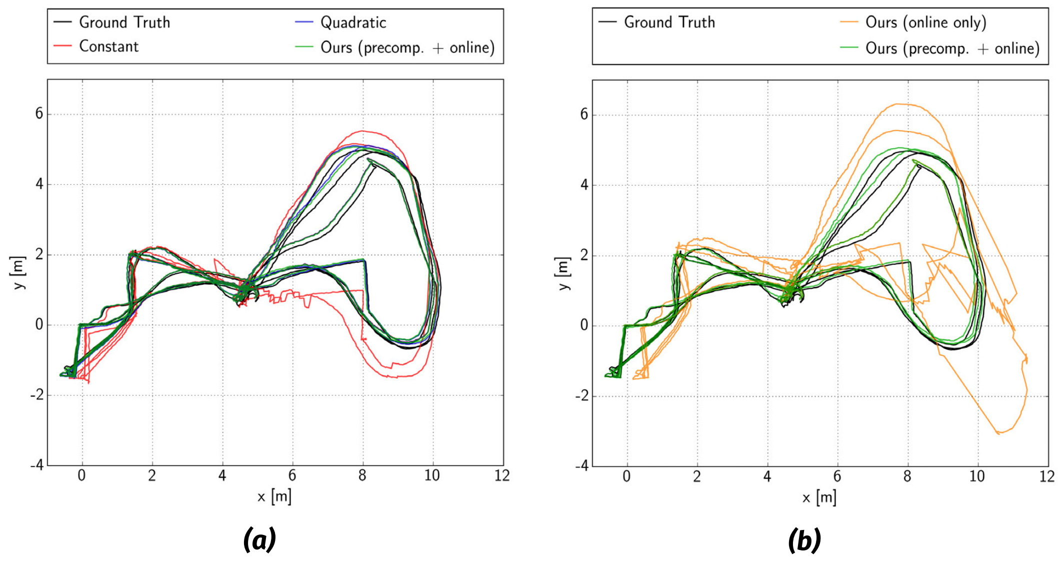
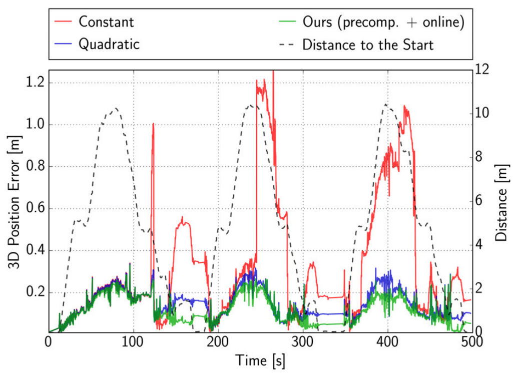


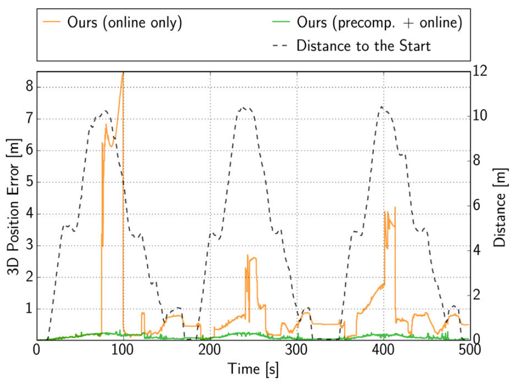
Figure 8.10: *Comparison of SLAM trajectories generated with our combined marker detection uncertainty model against constant and quadratic models (a) as well as against a partial variant without the precomputed data (b)*

75° is based on these estimates and crucial for any of the methods to provide usable results. In our experiment, the remaining differences between the precomputed and online estimated uncertainties have a negligible influence on the results of global optimization, although we would expect this to be different in scenarios featuring smaller markers or a larger number of far-away detections. The error plots in Figure 8.11 show the absolute 3D position errors measured in the robot's start frame. They exhibit some periodicity, each period corresponding to one of the three loops driven by the robot, with peak errors approximately at the positions farthest away from the start. This behavior is caused by the accumulation of incremental localization errors that were compensated whenever the robot detected markers that it had first observed early on, close to its start position.

The comparison shows that our novel method leads to the highest overall SLAM accuracy. It, however, is closely followed by the quadratic uncertainty model. As discussed above, our online corner error simulation is essential for the results of all methods in this evaluation. The precomputed part of our model could be replaced by, for example, a quadratic error model in future work. This would reduce the complexity and number of parameters of the model as well as eliminate the effort of precomputing a large lookup table in exchange for a marginal loss in accuracy. Nonetheless, an identification of the parameters of such a simplified model needs to be performed and validated against real-world experiments. The development of a methodology to do this with minimal effort, i. e., a minimum number of required



(a)



(b)

Figure 8.11: *Comparison of 3D position errors in the start coordinate frame for the SLAM trajectories generated with our combined marker detection uncertainty model against constant and quadratic models (a) as well as against a partial variant without the pre-computed data (b). The dashed lines indicate the distance to the robot's start position, its periodicity indicate the trajectory's three loops shown in the corresponding Figure 8.10.*

Estimation method	**Max. error** [m]	**Mean error** [m]	**RMS error** [m]
Constant	1.26	0.35	0.48
Quadratic	0.34	0.15	0.17
Ours (online only)	8.49	1.06	1.96
Ours (precomputed only)	0.33	0.13	0.14
Ours (precomputed + online)	**0.33**	**0.13**	**0.14**

Table 8.3: *Comparison of 3D position errors of SLAM trajectories computed with different uncertainty estimation methods for the detection and pose estimation of marker-based static landmarks in the outdoor scenario depicted in Figure 8.9. See Figure 8.10 and Figure 8.11 for the corresponding trajectories and error plots respectively.*

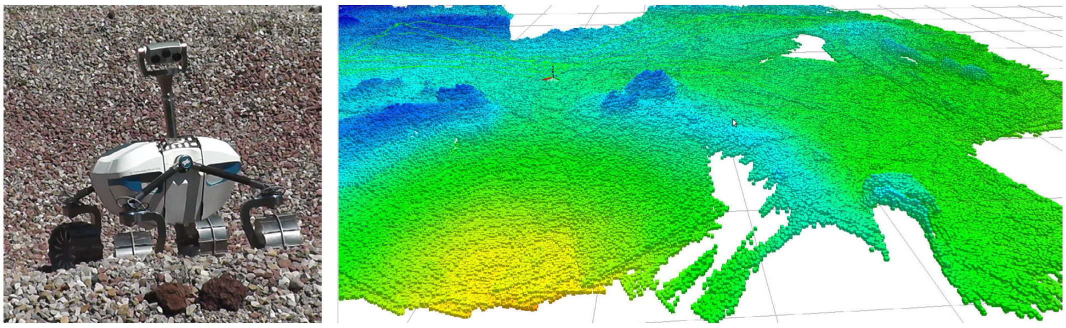


Figure 8.12: *Outdoor experiment: photo of LRU and height-colored 3D point cloud map of outdoor scenario*

experiments, remains a topic for future research.

8.2 Online 6D SLAM

First, we evaluate our 6D SLAM framework for probabilistic online localization, mapping, and place recognition in diverse conditions. For this, we analyze the combination of loop closures from submap matching with graph optimization in single-robot scenarios featuring indoor, outdoor and mixed environments (Section 8.2.1). Second, we evaluate the impact of our novel combination of local and global estimation methods, presented in Chapter 4, on the overall localization accuracy. For this, we compare our novel integration of local reference filter estimates into the SLAM graph to our previous approaches (Section 8.2.2).

8.2.1 Graph SLAM with Map Matching

We conducted our first single-robot experiments on an integration of map matching and graph optimization. This section is based on experimental results that we first presented in our conference paper by Brand et al. (2015). In these early experiments, we used the old graph topology for sequential odometry measurements and our 6D map matcher as described therein.

Experimental Setup

For evaluation, we tested our methods in three different environments, similar to those we used to evaluate our obstacle mapping in Section 8.1.1. We, however, varied the robot trajectories and used the P3AT as well as the LRU robot, both with narrow-angle lens cameras:

1. *Outdoor:* four rounds in an unstructured environment with several types of gravel, larger stones, as well as an artificial crater that the robot traversed during one of its rounds. In Figure 8.12 and Figure 8.13, we present a 3D view and a top-down visualization of our map of the test site respectively. Robot: LRU, stereo camera framerate: 4.8 Hz.

2. *Indoor:* three rounds in our lab environment with rooms and hallways, as visible in the resulting map visualized in Figure 8.14. Robot: P3AT, stereo camera framerate: 14.6 Hz.

3. *Mixed Indoor & Outdoor:* two rounds in parts of the indoor scenario, extended by a large loop around the lab building as visualized in the resulting map in Figure 8.16. Robot: P3AT, stereo framerate: 14.6 Hz.

In this evaluation, we did not use wheel odometry on any of the two robots. For all three scenarios, we created 3D point cloud maps at a resolution of 3 cm. In our outdoor scenario, we deliberately reduced the framerate of our stereo cameras in order to highlight the robustness of our localization and mapping system.

Results and Discussion

In Table 8.4, we present details and error statistics on our experiments conducted in the three aforementioned scenarios. To avoid biases in the comparison of the estimated and ground truth trajectories, we excluded consecutive measurements for periods during which the robot did not move.

In the outdoor scenario, our SLAM algorithm achieved a final 3D position deviation of 0.22% w. r. t. the length of the full trajectory. Its average 3D position error was 0.26 m, compared to 1.10 m for the filter estimate. In Figure 8.13, we visualized a top-down view on the resulting global 3D map, the SLAM graph with filter estimates and submap matches, as well as the estimated robot trajectory.

Indoors, man-made structures featuring untextured and reflective surfaces, like white walls, as well as regular patterns, like radiators, constitute challenges for stereo reconstruction algorithms. The resulting stereo mismatches lead to noisy

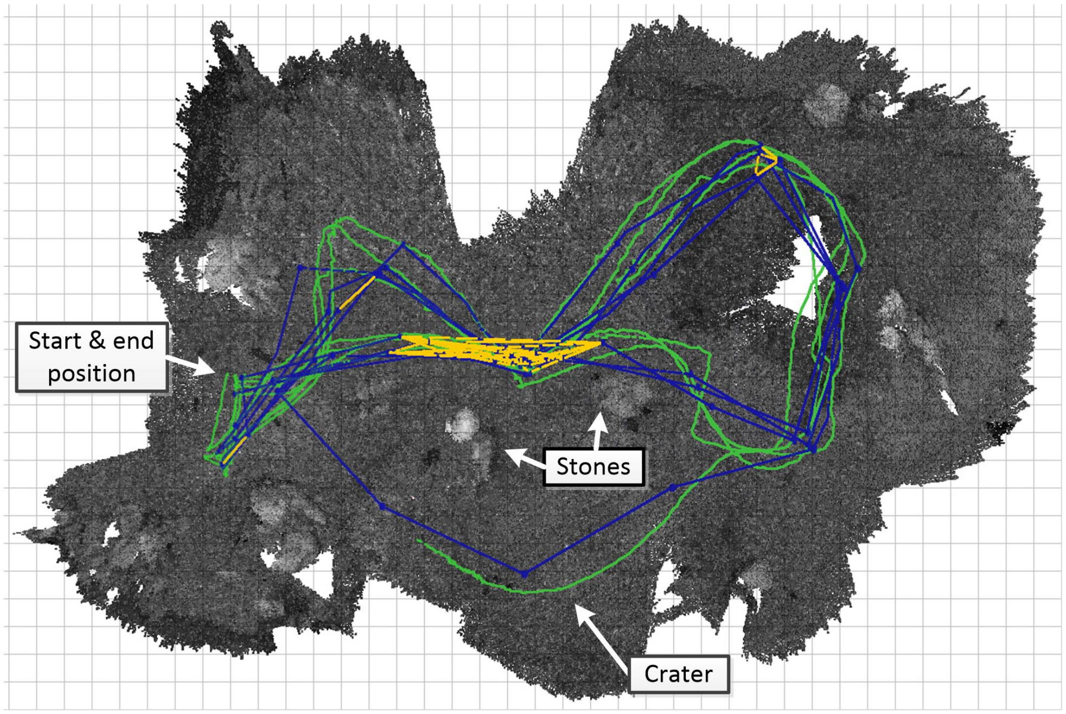


Figure 8.13: *Single-robot outdoor 6D SLAM experiment: top-down view on greyscale 3D point cloud map created by our algorithm. The green trajectory shows the SLAM estimate available to the robot at its respective positions; colored lines represent edges in the SLAM graph (blue: filter estimates between subsequent submaps; yellow: submap matches).*

and corrupted depth images and 3D maps. In our indoor scenario, in particular the corridor, containing planar white walls and a partially reflective ground plane, poses challenging conditions with little texture information as well as almost no unique geometric features. We present the indoor map generated by our SLAM algorithm in Figure 8.14. Despite the aforementioned challenges, our submap matching still provides sufficiently good results to construct a, in most parts, coherent 3D map. In the tracking area in the lower half of the map, we achieve an average 3D trajectory error of 0.13 m for our SLAM system compared to 0.32 m for the filter.

The results of our experiment in the mixed scenario highlight the robustness of our approach as it can cope with indoor and outdoor environments using the same set of parameters. In Table 8.4, we only show the final 3D position error as our tracking area is too small for a meaningful statistical evaluation. We present the final map after a total driven distance of 326.3 m in Figure 8.15, with all submaps being aligned according to their poses estimated by the graph optimization. After the first outdoor loop but before any loop closures, the position errors of the filter and SLAM estimates are both 3.17 m. The blue and green paths represent the sequentially logged filter and SLAM estimates at each particular point in time respectively, not

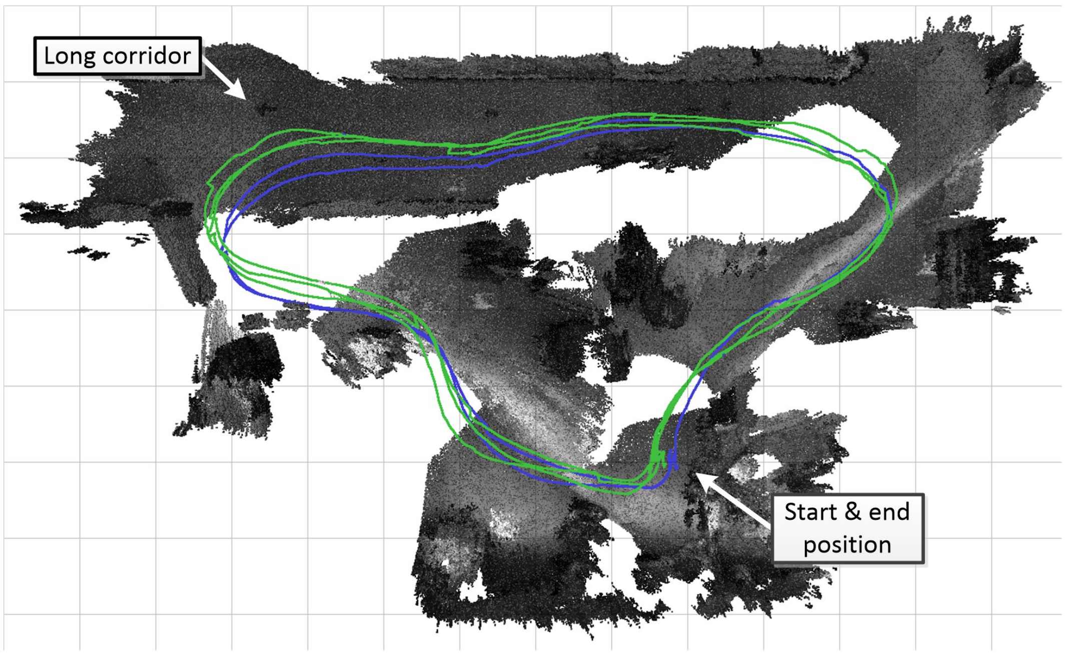


***Figure 8.14:** Single-robot indoor 6D SLAM experiment: top-down view on greyscale 3D point cloud map created by our algorithm. The blue and green trajectories show the filter and SLAM estimates available to the robot at its respective positions.*

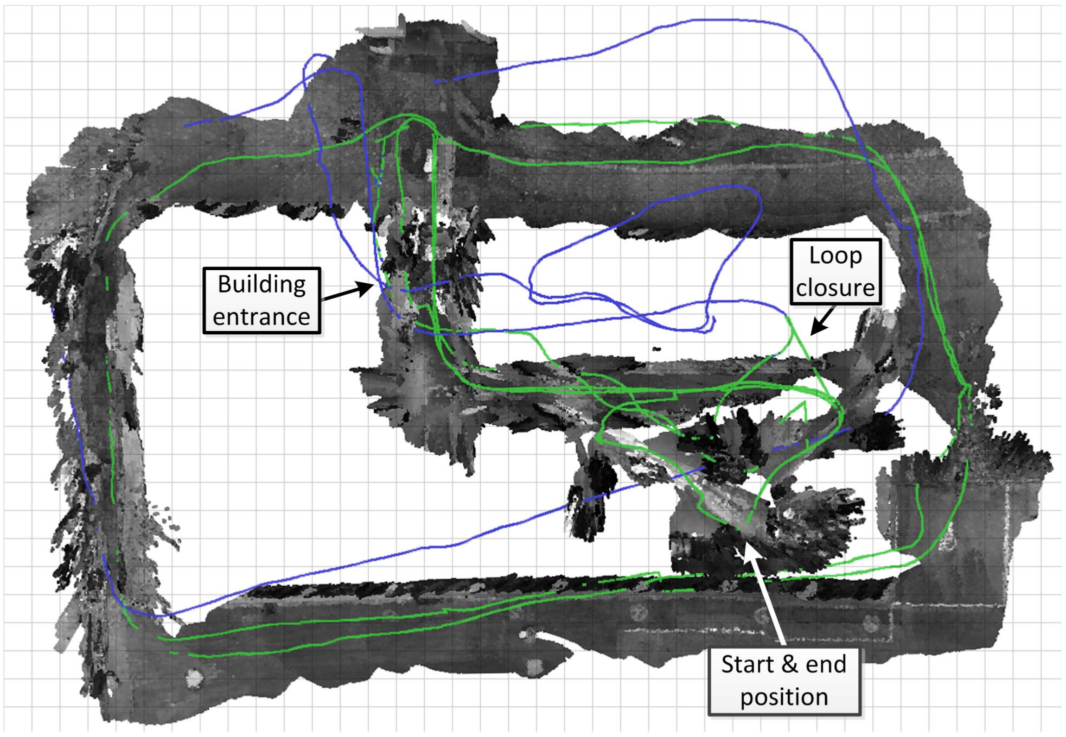


***Figure 8.15:** Single-robot mixed indoor and outdoor 6D SLAM experiment: top-down view on greyscale 3D point cloud map created by our algorithm. The blue and green trajectories show the filter and SLAM estimates available to the robot at its respective positions.*

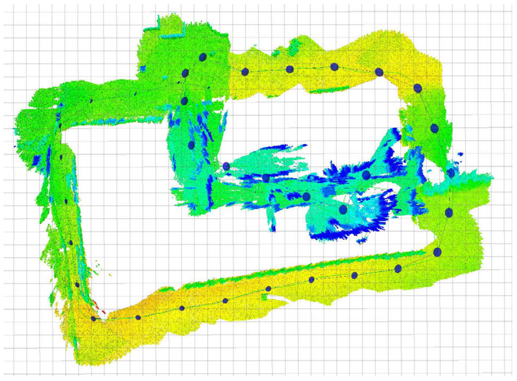
Scenario	**Outdoor**		**Indoor**		**Mixed**	
Robot	**LRU**		**P3AT**		**P3AT**	
2D bounding box [m]×[m]	11×17		8×13		25×37	
Total driven distance [m]	148.7		70.7		326.3	
Mean linear velocity [m/s]	0.355		0.304		0.329	
Experiment duration [s]	652.8		295.2		1187.7	
# of submap matches	12		10		16	
3D position error	**Filter**	**SLAM**	**Filter**	**SLAM**	**Filter**	**SLAM**
Mean [m]	1.10	0.26	0.32	0.13	-	-
Std [m]	0.62	0.14	0.17	0.05	-	-
RMS [m]	1.26	0.30	0.36	0.14	-	-
Max. [m]	2.43	0.73	0.52	0.35	-	-
Final [m]	2.40	0.33	0.49	0.04	6.94	0.25

Table 8.4: *Comparison of 3D localization errors for filter and SLAM in all three scenarios (no ground truth for statistical evaluation available for mixed scenario)*

any fully optimized trajectories. Thus, they are equal and overlap in Figure 8.15 until the first loop closure, which happened after the robot re-entered the lab building after its first round. In Figure 8.16, we present top-down views on the height-colored map created right before and after the first loop closure, showing its impact on the global map. After the second round and multiple loop closures based on submap matches, our SLAM framework achieves a final 3D position error of 0.25 m compared to 6.94 m for the filter. Taking the results of all three experiments into account, we have shown that our localization and mapping approach is able to generate consistent, globally optimized 3D maps with valid loop closures in indoor, outdoor, and mixed environments.

Comparison to 3D RBPF-based SLAM

As an additional evaluation, we compare our 6D SLAM approach to a 3D SLAM based on Rao-Blackwellized Particle Filters (RBPFs), which we introduced in our earlier work by Brand et al. (2014) and discussed in Section 8.1.1 to evaluate our obstacle mapping. In Table 8.5 we present a comparison of statistics on the 2D localization accuracies of both methods in our outdoor and indoor scenarios. Our new 6D SLAM approach achieves an improvement on the mean 2D position error of 27 % and 50 % in the outdoor and indoor scenarios respectively. As in these early experiments, we still used an approximation of a SLAM graph topology for sequential


(a) Before loop closure

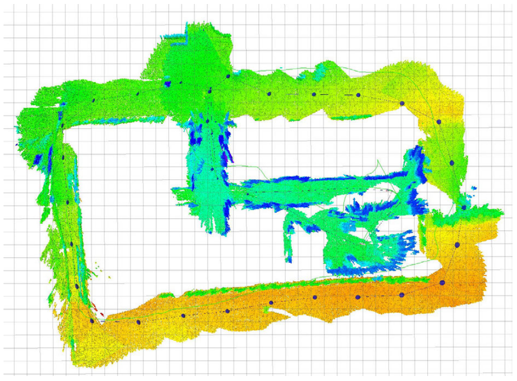

(b) After loop closure

Figure 8.16: *Top-down view on the point cloud map (hight-colored) created right before and after the first loop closure in our mixed indoor and outdoor 6D SLAM experiment*

Scenario	Outdoor					Indoor				
2D pos. error [m]	**Mean**	**Std**	**RMS**	**Max.**	**Final**	**Mean**	**Std**	**RMS**	**Max.**	**Final**
3D RBPF SLAM	0.22	0.13	0.26	0.63	0.22	0.14	0.08	0.17	0.36	0.19
6D graph SLAM	0.16	0.10	0.19	0.41	0.08	0.07	0.03	0.08	0.19	0.03

Table 8.5: *Comparison of the localization accuracy of our 6D graph-based SLAM framework with the 3D RBPF SLAM that we used in our earlier work (Brand et al., 2014) to evaluate our obstacle mapping (see Section 8.1.1)*

odometry measurements, further improvements are to be expected through a better integration of our local reference filter estimation, which we will evaluate in the following section.

8.2.2 Graph Topology for Local Reference Filter Estimates

In the remainder of this section, we present the evaluation of our novel SLAM graph topology for the integration of local reference filter estimates (see Section 4.2.2), which we first published in a conference paper by Schuster et al. (2015).

Experimental Setup

We conducted three single-robot experiments with two different robots in outdoor, indoor, and mixed scenarios similar to those discussed in the previous section. As we performed this evaluation prior to a full integration of our submap matching components, we used static artificial landmarks in the form of AprilTag markers to

Scenario	Outdoor		Indoor		Mixed	
Robot	LRU		P3AT		P3AT	
Driven distance	106 m		69 m		320 m	
Ground truth available	106 m		25 m		10 m	
3D trajectory error	μ [m]	σ [m]	μ [m]	σ [m]	μ [m]	σ [m]
Filter only	0.402	0.192	0.327	0.109	1.324	0.886
Seq. odom. SLAM	0.153	0.083	0.177	0.060	0.121	0.082
Local ref. SLAM	**0.142**	**0.079**	**0.146**	**0.054**	**0.096**	**0.060**

Table 8.6: *Comparison of SLAM graph topologies for sequential odometry measurements and our novel integration of local reference filter estimates (see Section 4.2.2)*

generate loop closure constraints. In order to evaluate the impact of our novel graph topology, we compare the SLAM results to those computed with an approximation of sequential odometry measurements, as used in our earlier work by Brand et al. (2015) and discussed in Section 4.2.2. For this, we executed the implementations of both graph SLAM variants in parallel on the same input datasets and filter estimates.

Results and Discussion

A direct evaluation of the dependencies and covariances estimated with the two different graph topologies is not possible due to a lack of corresponding ground truth data. However, improvements in the approximation of the underlying probabilistic structure are expected to lead to more accurate localization estimates after graph optimization. Therefore, we use the 3D estimation errors in the SLAM trajectories as an indirect measure for the accuracy of the dependencies and uncertainty estimates asserted by the graph topology.

In Table 8.6, we provide a summary of our results, comparing both graph topologies in our three different experiments. They all show an improvement of the 3D localization accuracy of 15 % on average. This indicates the benefits from our adaption of the graph topology to the underlying probabilistic structure of local reference filter estimates, which requires less approximations when integrating their respective covariances. In addition to improving the accuracy, our novel graph topology allows a straightforward integration of delayed measurements without any changes to the existing graph structure, i. e., without the need for any deletion, discounting or replacement of existing factors during incremental graph construction and optimization. This is particularly important for multi-robot systems, where estimates from other robots might be significantly delayed due to temporary communication outages. In contrast, with the standard graph structure for sequential odometry measurements, it is necessary to either modify the graph or wait for the slowest estimation component. In our experiments, this was the marker detection with delays of up to 1 s after image creation.[3] In order to ensure that all poses are added in chronological order to the sequential odometry-based SLAM graph, we added an artificial delay of 2 s to wait for the low-rate tag detection component. In our experiments discussed above, we added this artificial delay for both graph topologies in order to achieve a better comparability by removing its influence from the comparison.

[3]We replaced this slow marker detector by a significantly more efficient and faster implementation for our later multi-robot experiments.

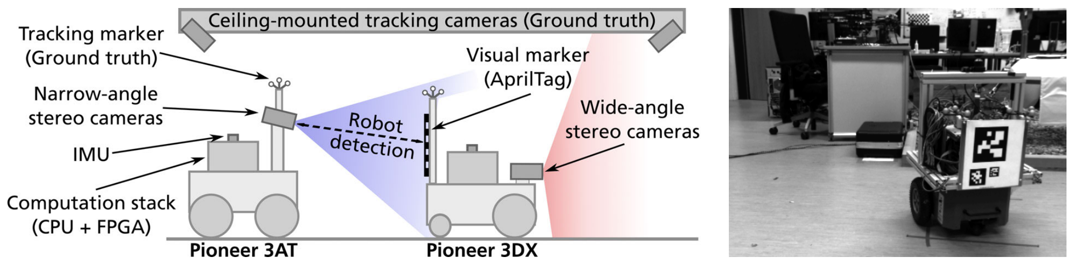

Ceiling-mounted tracking cameras (Ground truth)
Tracking marker (Ground truth)
Narrow-angle stereo cameras
IMU
Computation stack (CPU + FPGA)
Robot detection
Visual marker (AprilTag)
Wide-angle stereo cameras
Pioneer 3AT
Pioneer 3DX

Figure 8.17: *Multi-robot experimental setup and P3DX as seen by P3AT*

8.3 Collaborative 6D Multi-Robot SLAM

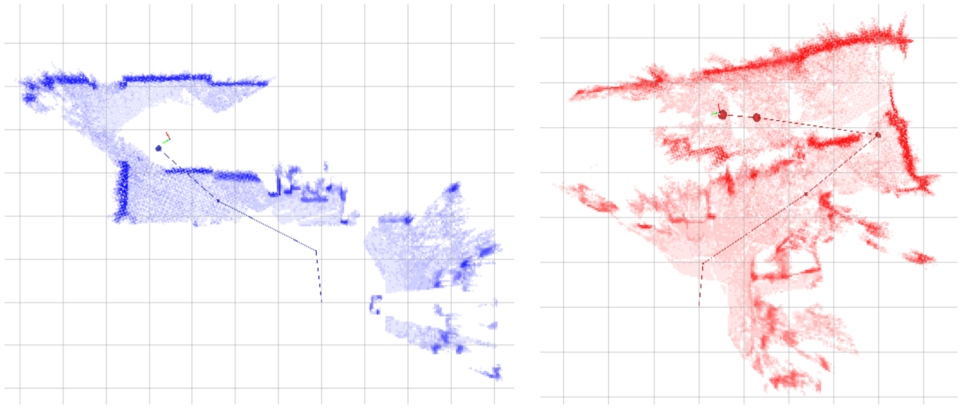
We evaluated and demonstrated our full collaborative 6D multi-robot SLAM framework in a total of eight experiments on two different rover systems. First, we conducted two experiments in a lab environment with a team of two Pioneer rovers that are equipped with heterogeneous stereo camera setups, featuring narrow-angle lenses on one and wide-angle lenses on the other robot (Section 8.3.1). Second, we employed two LRU rovers for five extended multi-robot experiments, featuring larger areas of up to $57\,\mathrm{m} \times 53\,\mathrm{m}$ (Section 8.3.2). In addition, we demonstrated its application for autonomous multi-robot exploration at a Moon-analogue site (Section 9.3.2) and with a heterogeneous team of flying and driving robots (Section 9.4).

8.3.1 Rovers with Heterogeneous Camera Setups

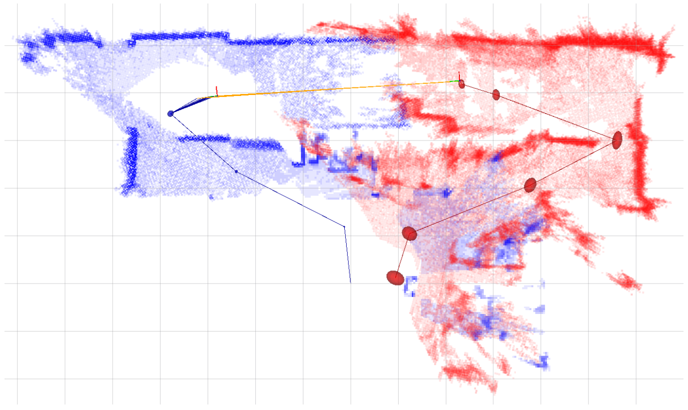
For our first experimental multi-robot evaluation, we employed an early version of our SLAM framework. It lacked the robust error function for graph optimization and used a 6D map matching, as described in our publications by Schuster et al. (2015) and Brand et al. (2015). This section is based on experimental results that we first presented in our conference paper by Schuster et al. (2015).

Experimental Setup

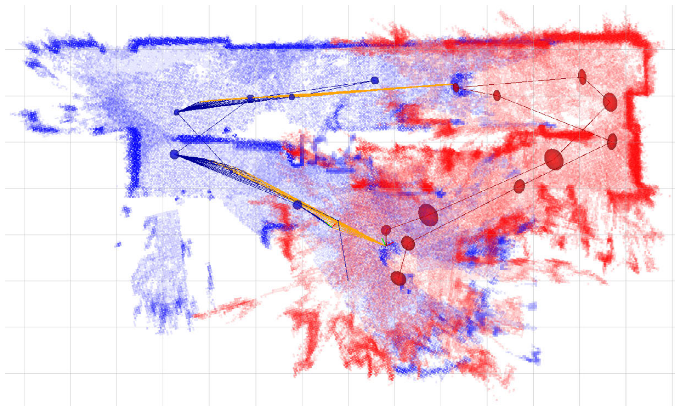
We performed multi-robot experiments with our Pioneer 3-AT (P3AT) and Pioneer 3-DX (P3DX) rovers with narrow- and wide-angle stereo camera setups. We did not use any static artificial landmarks in our multi-robot experiments. P3DX had an AprilTag marker attached to its back in order to allow for visual robot detections by P3AT, as sketched out in Figure 8.17. The scenario features an indoor lab environment of approx. $100\,\mathrm{m}^2$, as depicted in Figure 8.18(c). It consists of a



(a) *Separate maps of P3AT and P3DX before the first inter-robot measurement. Both are in their own coordinate frames.*



(b) *Joint map as computed by P3AT after first robot detections; angular error due to imprecise first long-range detection.*



(c) *Corrected joint map after two series of detections where P3AT observed P3DX*

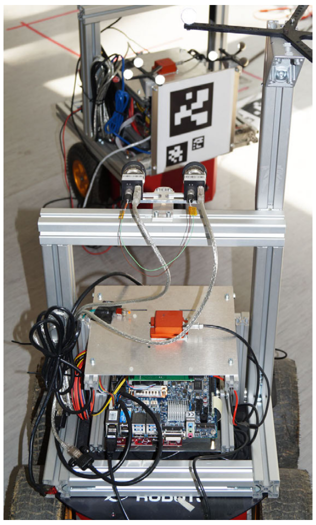
Figure 8.18: *Multi-robot experiment with our two rovers, P3AT (blue) and P3DX (red), and unknown robot start poses; separate maps computed by both robots before (a) and after (b) their first connection, as well as after two series of inter-robot detections (b). The ellipsoids show the submap origins and are scaled to their respective position standard deviation estimates. Blue and red edges connect submaps and robot poses, orange edges represent robot detections.*

long hallway (top) and a large room (bottom), in which our *ARTTRACK2* tracking system (Advanced Realtime Tracking GmbH, 2014) is set up. We conducted two experiments in this setup: In the first experiment, we only use visual robot detections as loop closures, whereas in the second, we added intra- and inter-robot constraints from submap matches.

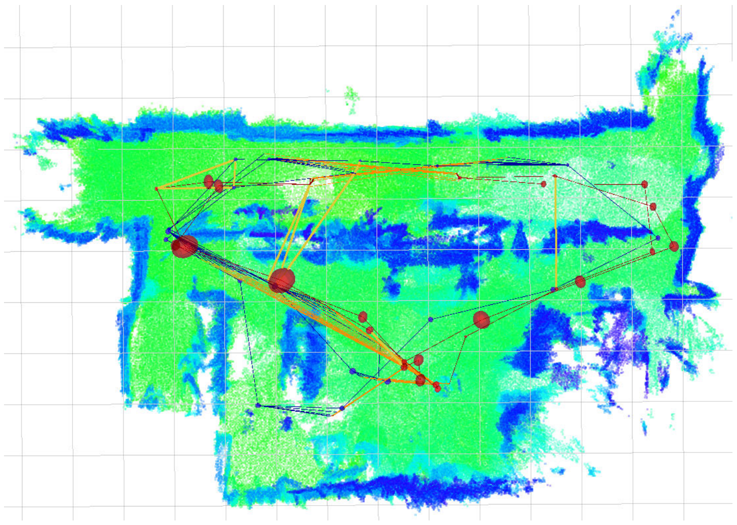
Results and Discussion

In the first experiment, both rovers did not have any initial knowledge about their relative positions and thus started mapping the area separately. We present a time series showing their maps in Figure 8.18. After entering the corridor from different sides, the first inter-robot measurements, i. e., robot detections, could be made. They were imprecise due to an observation distance between the robots greater than 5 m and thus lead to an error in the yaw angle that is visible in the top part of Figure 8.18(b). Later on, the integration of additional detections at a different location allowed a more accurate alignment of the submaps created by both robots, as shown in Figure 8.18(c). In total, the robots P3AT and P3DX created 9 and 11 submaps respectively and connected their localization estimates via 108 robot detections in their SLAM graphs.

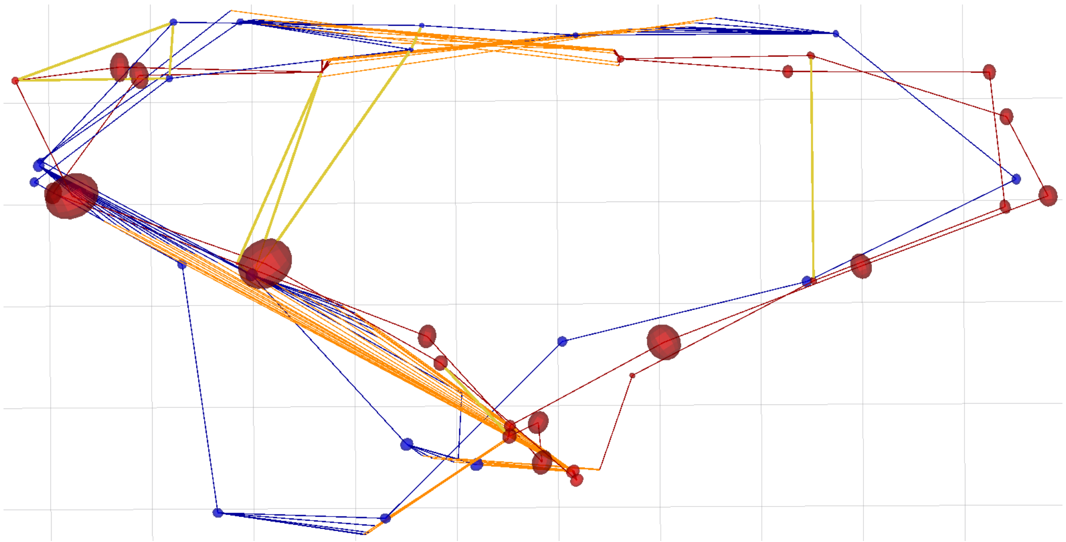
In our second experiment, the initial robot positions allowed robot detections at the beginning that connected both SLAM graphs. During the experiment, both rovers observed most parts of the environment multiple times. In Figure 8.19, we present the final map and SLAM graph. In order to show the benefit of our multi-robot SLAM approach, we compare its localization errors to single-robot SLAM computed for both robots individually on the same dataset. We present the resulting graph and error statistics in Table 8.7 and plot the 3D position and yaw angle errors of filter and SLAM estimates for each robot with respect to partially available ground truth in Figure 8.20. Our multi-robot approach achieved a 32 % higher average accuracy compared to the two separated single-robot estimates. This indicates the benefits of our joint graph optimization compared to the estimation of a single relative transformation between the coordinate frames of multiple robots in order to solely connect their maps. In the multi-robot case, inter-robot submap matches and robot detections add additional loop closure constraints to the optimization. The robots thereby serve as “moving landmarks” for each other. This can improve their localization, in particular when the quality of their intermediate local pose estimates differs between consecutive robot detections. In our experiments, P3DX exhibits larger visual odometry errors than P3AT due to its wide-angle stereo camera setup, as we discussed in Section 8.1.1. These errors propagate to higher position uncertainty estimates, represented by the



(a) *Photo of P3AT and P3DX*



(b) *Top-down view of final 3D map (height-based coloring)*

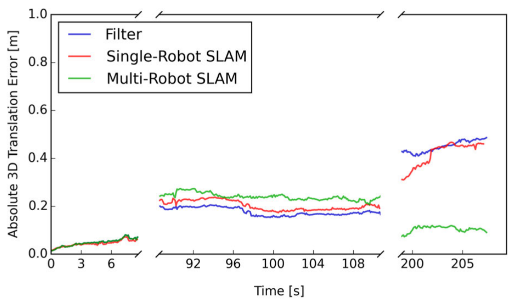


(c) *Multi-robot SLAM graph: ellipsoids show the submap origins (P3AT: blue, P3DX: red) and are scaled to their respective positional standard deviation estimates. Orange edges in the graph represent robot detections (P3AT detects P3DX), yellow edges submap matches.*

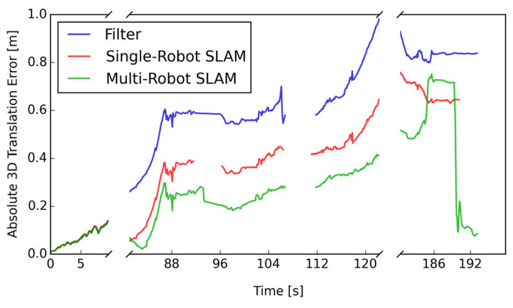
Figure 8.19: *Multi-robot experiment with two Pioneer rovers*

		Multi-Robot		Single-Robot	
		P3AT	P3DX	P3AT	P3DX
Number of robot poses x_i^r		72	72	0	0
Number of submaps s_i^r		21	27	21	27
Number of submap matches c_i	per robot	4	3	2	7
	inter-robot	4		0	
Number of robot detections d_i		72	0	0	0
Total number of nodes θ_i		192		21	27
Total number of factors f_i		274		30	36
Total driven distance [m]		46.49	44.56	46.49	44.56
Ground truth available [m]		9.55	12.48	9.55	12.48
Mean 3D trajectory error [m]		**0.17**	**0.21**	0.22	0.36

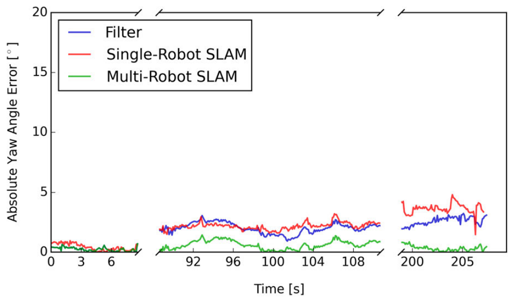
Table 8.7: *Comparison of graph and error statistics for multi-robot and single-robot SLAM for the experiment presented in Figure 8.19*



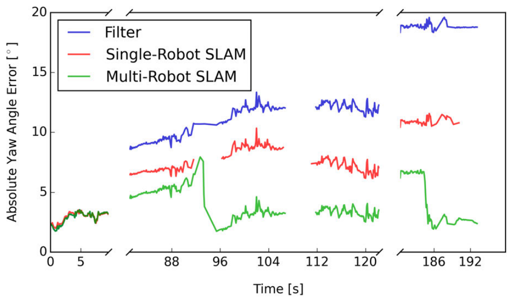
(a) *P3AT 3D position error*



(b) *P3DX 3D position error*



(c) *P3AT yaw error*



(d) *P3DX yaw error*

Figure 8.20: *Pose errors for the multi-robot SLAM experiment presented in Figure 8.19 and Table 8.7. Plots show the partial trajectories for which ground truth is available. All estimates refer to the sequentially logged data available at each particular point in time, not an afterwards fully optimized trajectory.*

larger covariance ellipsoids in the SLAM graph visualized in Figure 8.19(c). The joint graph optimization, balancing errors according to their respective uncertainty estimates, thus primarily improves the localization of P3DX, as can be observed in Figure 8.20. This, however, leads to a temporary slight degradation of the position estimate for P3AT, which can be observed in the middle part of the error plots in Figure 8.20(a). Until this point in time, only few submaps could be matched due to the small overlap of the areas visited and observed by the two robots. Later on, additional submap matches significantly improved the estimates of both robots.

The final maps for both experiments, shown in Figure 8.18(c) and Figure 8.19(b), depict consistent representations of the walls and doorways of our indoor scenario. The indoor environment poses several challenges to stereo vision-based algorithms with its texture-less walls, reflecting surfaces, and regular patterns (e. g., radiators). These lead to visual odometry and depth estimation errors, which can be observed as noise in our maps. Note that we designed our methods primarily in view of outdoor planetary exploration scenarios. Our point cloud-based map representations thus do not make any assumptions about a structured environment, i. e., they are not biased towards straight walls or even floors.

8.3.2 Extended Experiments with two Lightweight Rover Units

We conducted a series of five extended multi-robot experiments with two Lightweight Rover Units (LRUs), which we first presented in our journal article by Schuster et al. (2018). They feature our up-to-date SLAM framework as presented in this thesis.[4]

Experimental Setup

In Figure 8.21, we give an impression of our experimental setup. The scenarios for our five experiments feature our mobile robotics lab as well as adjacent labs, hallways, and a transition between indoor and outdoor areas:

- *Experiment #1*: mobile robotics lab with three artificial large stones, featuring the aforementioned tracking system for ground truth.
- *Experiment #2*: mobile robotics lab (lower right part in the map) as start and finishing point for both rovers. They drove one large loop each, passing through the adjacent lab, the entrance area as well as long hallways (see Figure 8.24).

[4]Except for the filtering of keypoints located close to unknown regions during submap matching (see Section 6.1.1), as we implemented this feature after evaluating our experiments.

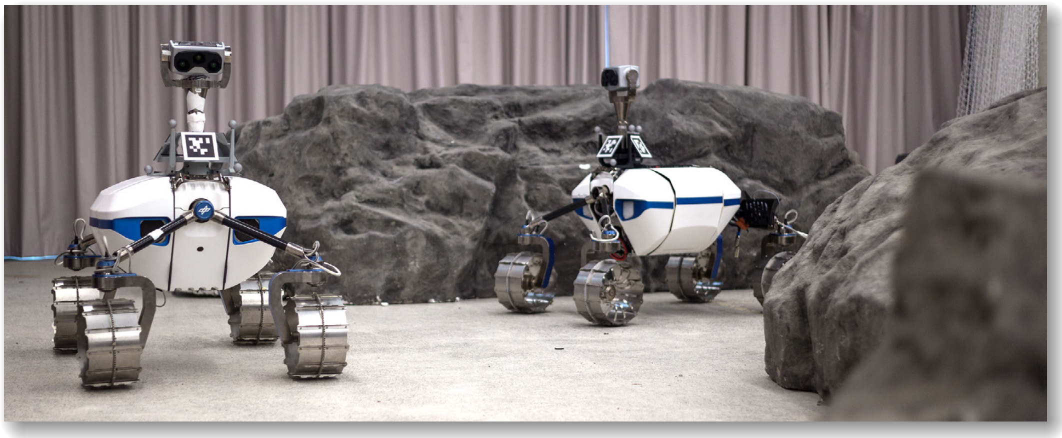


Figure 8.21: *Impression of multi-robot experiment with two rovers, LRU1 and LRU2*

- *Experiment #3*: setup similar to #2, with slightly different robot trajectories, in particular within the mobile robotics lab.
- *Experiment #4*: similar to #2 and #3, but with LRU2 driving two loops through the central lab.
- *Experiment #5*: mobile robotics lab with adjacent labs and two loops of LRU2 leaving and entering the building, thus including indoor and outdoor areas.

During the five experiments, the two rovers traveled trajectories of up to 200 m and 171 m in areas of up to $57\,\text{m} \times 53\,\text{m}$. In Figure 8.22, we provide a sketch of our experimental setup and we present the 3D maps generated by our SLAM system for all five experiments in Figure 8.23. In addition, we exemplarily selected Experiment #2 to show a more detailed map in a visual comparison to an architect's plan in Figure 8.24.

We used our aforementioned *VICON* tracking system (Vicon Motion Systems Ltd, 2019) to acquire (partial) ground truth for the robot poses within the area of the mobile robotics lab. In all five experiments, we did not use any artificial static landmarks in the environment. The intra- and inter-robot loop closure constraints thus solely stem from robot detections and submap matches. In contrast to our autonomous exploration experiments presented in Section 8.1.2, in this setting we rarely moved the robots' pan-tilt units and did not use them to perform full scans of the area. Therefore, we increased the threshold to generate new submaps (see Section 5.1.3) to 7 m of maximum driven distance and 0.2 m uncertainty to allow them to be large enough to contain sufficiently discriminative features for map matching even when the rovers only look straight ahead. We also adapted the map matcher parameters accordingly. As the two robots drive through each other's field

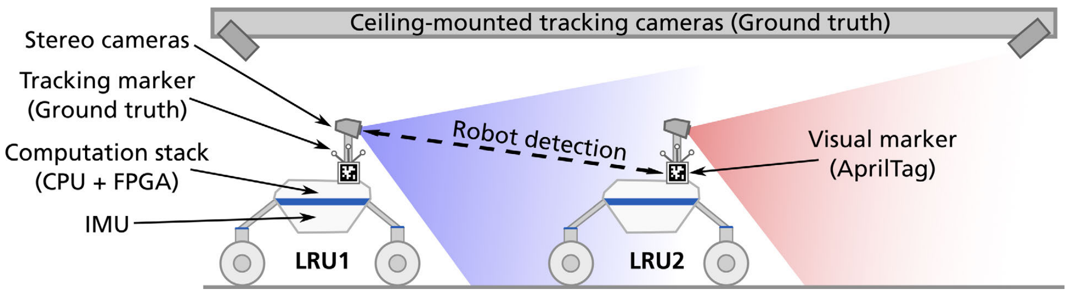

Ceiling-mounted tracking cameras (Ground truth)
Stereo cameras
Tracking marker (Ground truth)
Computation stack (CPU + FPGA)
IMU
Robot detection
Visual marker (AprilTag)
LRU1
LRU2

Figure 8.22: *Setup for multi-robot experiments with two rovers, LRU1 (blue) and LRU2 (red)*

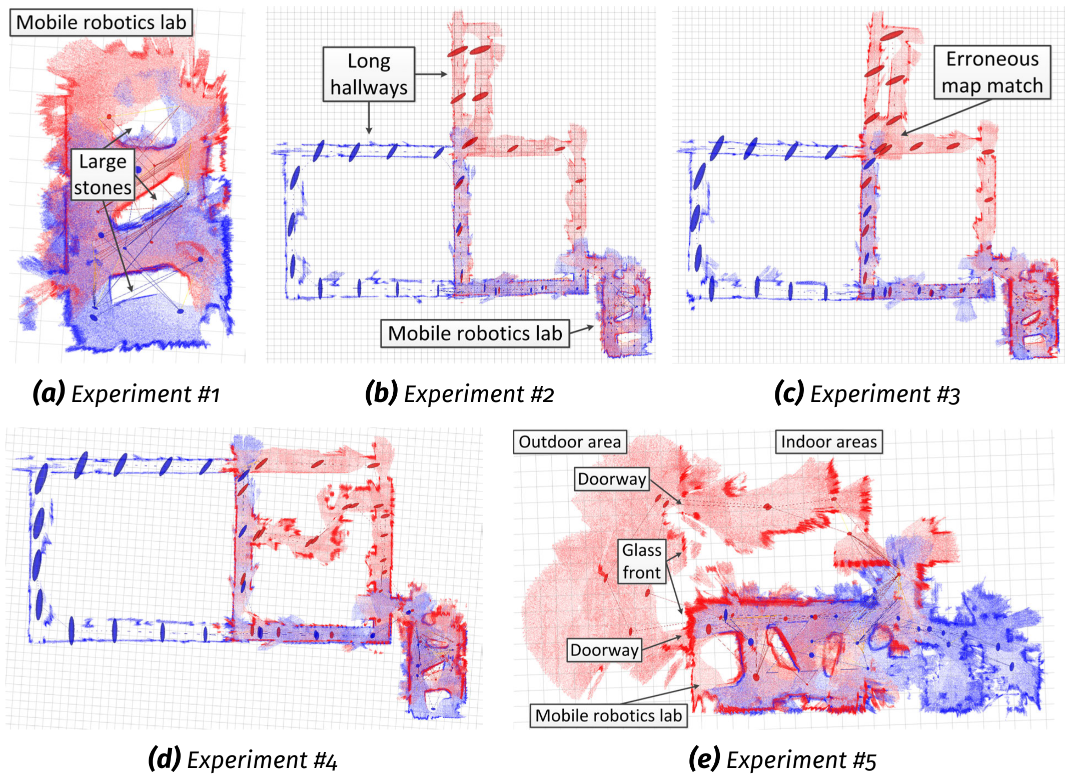


Figure 8.23: *Top-down views on the 3D point cloud maps (resolution* 0.05 m*) created for our five multi-robot experiments with LRU1 (blue) and LRU2 (red), overlaid on grids with* 1 m^2 *cell size to indicate their scale. See Figure 8.24 for details on the experimental setup and an overlay of a floor plan for Experiment #2.*

of view, we exclude their oriented 3D bounding boxes from visual odometry and 3D mapping according to the 6D pose estimates computed by our SLAM framework. While for this evaluation, we manually controlled both robots, in Section 9.3.2 we present an additional experiment with a preliminary extension of our exploration algorithm for multi-robot teams.

Results and Discussion

In the following paragraphs, we present and discuss the results of our five multi-robot experiments, with an exemplary more detailed analysis of Experiment #2. In Figure 8.23, we show the resulting 3D maps for all experiments and in Figure 8.24 a top-down view on the map of Experiment #2. It visually aligns well with the floor plan of the building, exhibiting only small deviations that are to be expected for a stereo-vision based setup. In Figure 8.25, we give an impression of the respective multi-robot 3D voxel grid map created by our mapping system. In all experiments, the robots had no prior knowledge of their relative positions, but could detect each other in the lab and at hallway crossings. In addition, they were able to compute intra- and inter-robot map matches in the lab as well as on the hallways that have been traveled by both of them. In Figure 8.26, we visualized examples for both types of loop closures from Experiment #2, showing their impact on the estimated map and positional uncertainties regarding the submap origins.

As for our early Pioneer experiments discussed in Section 8.3.1, the indoor parts of our scenario again constitute a challenging environment for stereo vision: Low-texture areas, reflective glass surfaces, and regular patterns lead to visual odometry and depth estimation errors, which can be observed as noise in our maps. For example, in the loop driven by LRU1 (blue) in the left part of the maps of experiments #2, #3 and #4, the point clouds are more sparse than in the rest of the scenarios. This was caused by difficulties in the stereo matching due to an untextured floor in this area, which also heavily impacted the performance of visual odometry. The local reference filter, however, was able to compensate for this with wheel odometry and IMU measurements, and our graph SLAM then corrected a large amount of the remaining errors.

In Figure 8.27, we present plots of the 3D trajectory errors over time for both robots in all five multi-robot experiments. All values refer to estimates available to the robots at the respective points in time. The plots are limited to the periods of time for which ground truth measurements from our tracking system are available, i. e., while the rovers were driving inside the mobile robotics lab. We compare the values of our multi-robot SLAM system (green) with the local filter estimates (blue). Jumps

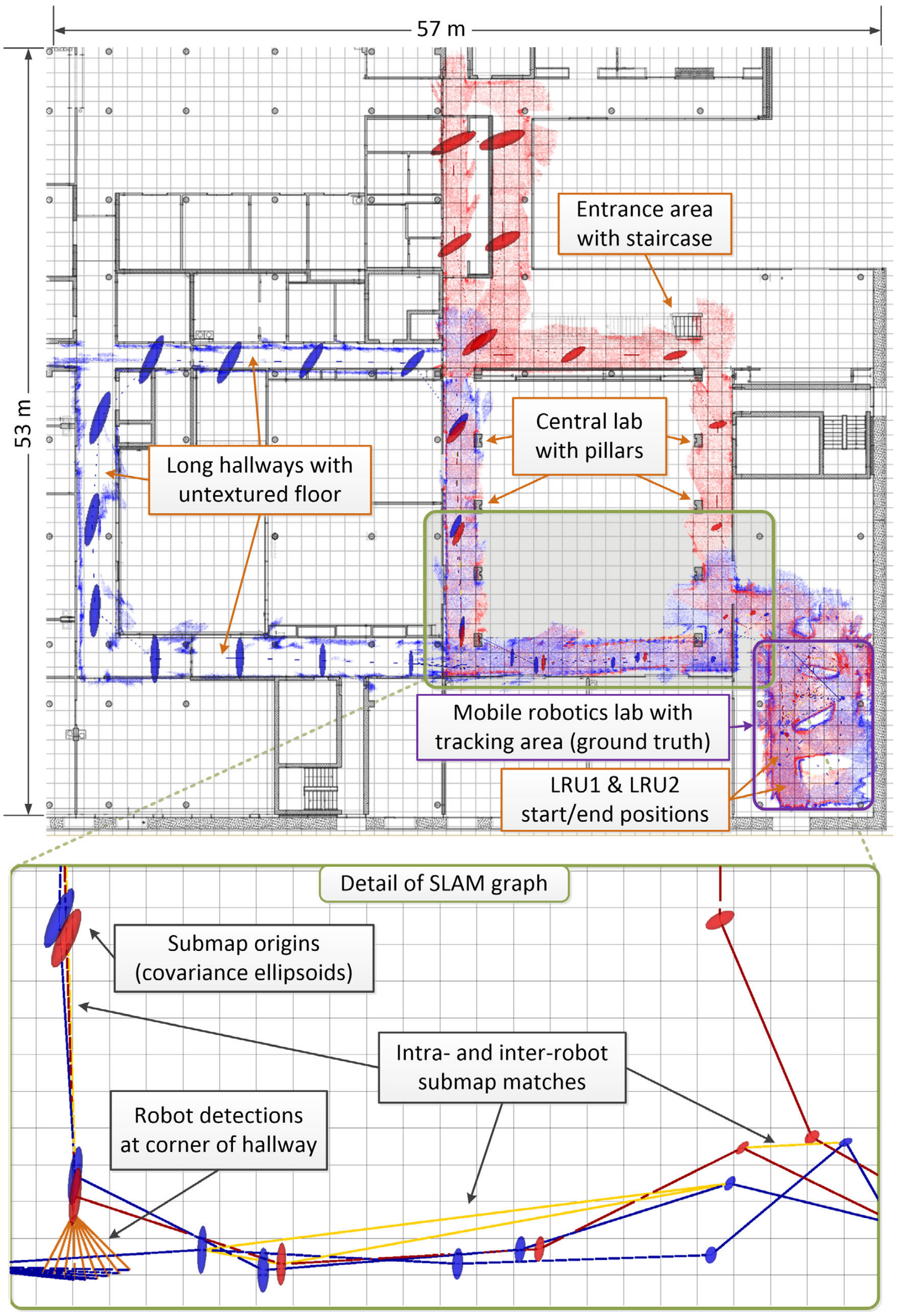

57 m
53 m
Entrance area with staircase
Central lab with pillars
Long hallways with untextured floor
Mobile robotics lab with tracking area (ground truth)
LRU1 & LRU2 start/end positions
Detail of SLAM graph
Submap origins (covariance ellipsoids)
Intra- and inter-robot submap matches
Robot detections at corner of hallway

Figure 8.24: *Details on multi-robot Experiment #2 with two rovers, LRU1 (blue) and LRU2 (red). Top-down view of final SLAM graph and 3D point cloud map with manually aligned floor plan (grid size:* 1 *m). Ellipsoids show the submap origins and are scaled to two times their respective positional standard deviation estimates. Red and blue edges represent filter estimates of the respective robots, yellow edges submap matches, and orange edges robot detections. See Table 8.9 for trajectory and graph statistics.*

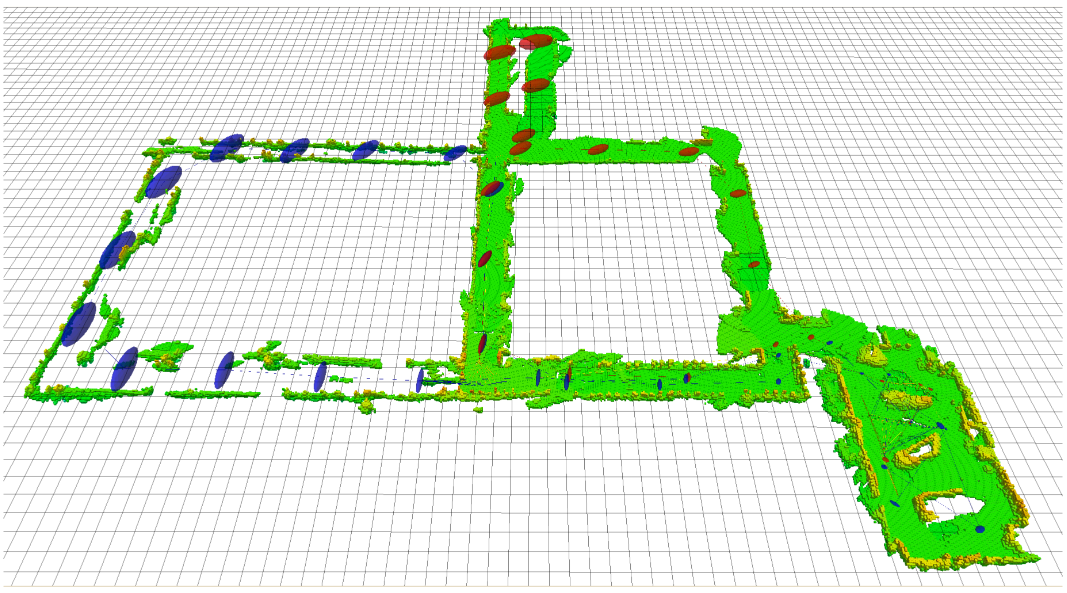
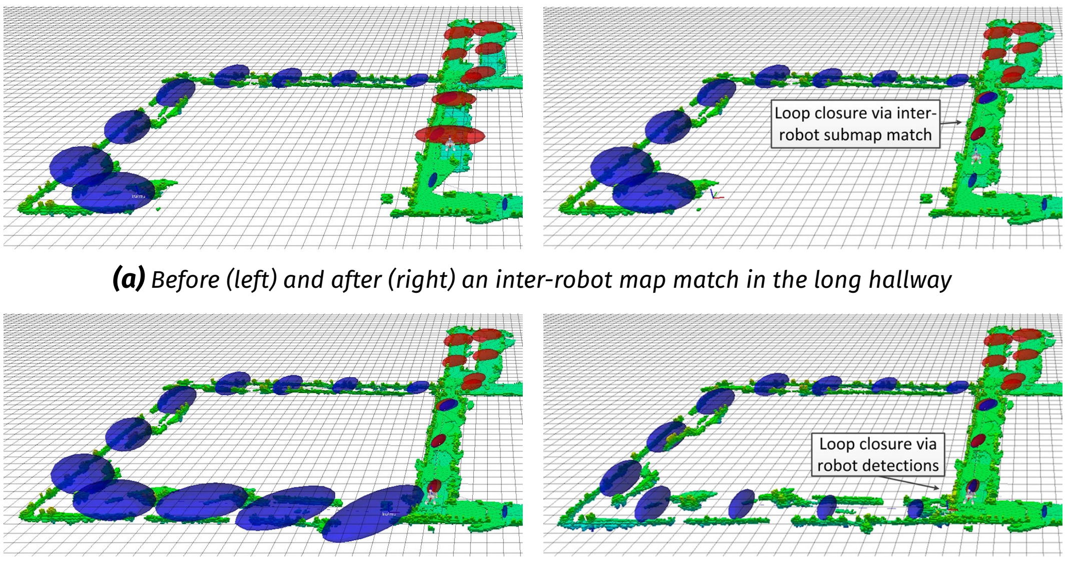


Figure 8.25: *Visualization showing the occupied voxels (height-colored, 10 cm resolution) of the 3D probabilistic voxel-grid representation of the multi-robot map created during our Experiment #2 with LRU1 (blue) and LRU2 (red). See Figure 8.24 and Table 8.9 for further details.*


Loop closure via inter-robot submap match
Loop closure via robot detections

(a) *Before (left) and after (right) an inter-robot map match in the long hallway*

(b) *Before (left) and after (right) robot detections at a hallway corner*

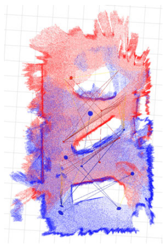
Figure 8.26: *Examples of multi-robot loop closures in Experiment #2*

		Exp. #1		Exp. #2		Exp. #3		Exp. #4		Exp. #5	
		LRU1	LRU2	LRU1	LRU2	LRU1	LRU2	LRU1	LRU2	LRU1	LRU2
# of robot poses x_i^r		15	15	26	26	35	35	58	58	31	31
# of submaps s_i^r		8	8	29	24	32	30	32	31	17	19
# of map matches c_i	per robot	2	3	2	1	4	4	0	3	4	5
	inter-robot	0		9		6		5		1	
# of robot detections d_i		17	0	8	19	15	22	9	55	12	21
Total # of nodes θ_i		46		105		132		179		98	
Total # of factors f_i		67		143		182		250		140	
Total driven dist. [m]		56.33	51.59	199.82	170.85	211.21	197.45	206.99	209.18	118.52	129.81
Ground truth avail. [m]		55.82	50.36	61.89	51.75	64.61	65.19	63.35	54.17	53.93	51.66
Mean 3D traj. error [m]		0.19	0.13	0.29	0.40	0.14	0.27	0.42	0.29	0.25	0.28
Max. 3D traj. error [m]		0.47	0.20	0.70	1.81	0.33	0.55	0.88	0.59	0.62	0.75
Mean yaw error [°]		2.55	1.79	3.08	2.87	1.03	2.27	3.07	3.78	2.10	1.41
Max. yaw error [°]		5.04	3.80	6.63	11.39	3.33	5.69	6.38	6.81	4.03	3.82

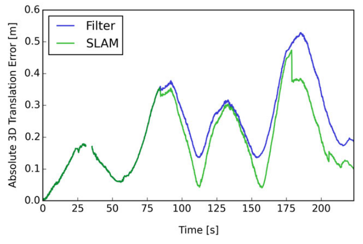
Table 8.8: *Comparison of trajectory and graph statistics (number of nodes and factors) for the five multi-robot experiments presented in Figure 8.27. The mean and maximum error values refer to those parts of the trajectories for which ground truth was available.*

in the green curves indicate the impact of loop closures. For most fractions of the trajectories, the positional error after global multi-robot optimization is significantly lower than when using only the local filter. In Table 8.8, we present additional statistics on the SLAM graphs and trajectories for the five experiments. With our combination of filter and graph optimization to separate high- and low-frequency measurements and estimates, we are able to keep the graph small and sparse, with a maximum total of 179 nodes and 250 factors for our multi-robot joint graph for combined robot trajectory lengths of up to 416 m. This allows for fast online optimization on each robot, as we discuss in Section 8.4 when analyzing the runtimes and computational resources required by our components.

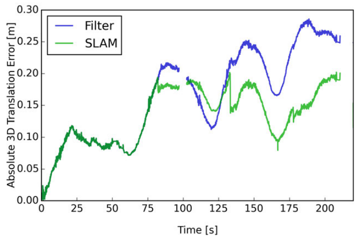
In our experiments, we observed a lower average number of single-robot loop closures compared to the single-robot autonomous exploration experiments that have been conducted in a similar environment inside our mobile robotics lab (see Section 8.1.2 and Figure 8.8). We attribute this mainly to two different causes. First, in all five multi-robot experiments, many loop closure opportunities arose at the end of the experiment, when both rovers returned to a previously visited area inside the mobile robotics lab. As described in Section 6.1, our map matcher runs in the background, processing a working queue of potential match candidates whenever free computational resources are available. However, we ended all experiments shortly after the rovers arrived at their final position and did not wait for the matcher to



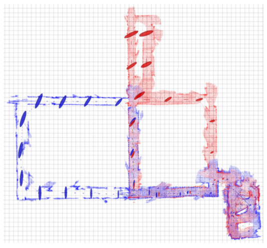
(a) *Map (Exp. #1)*



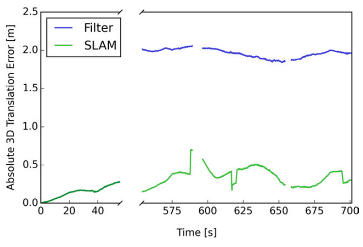
(b) *3D pos. error (LRU1, Exp. #1)*



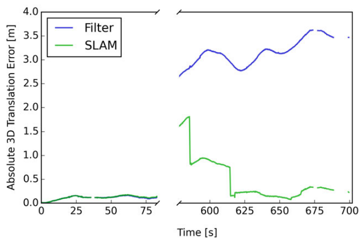
(c) *3D pos. error (LRU2, Exp. #1)*



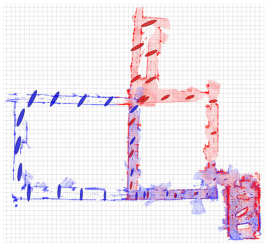
(d) *Map (Exp. #2)*



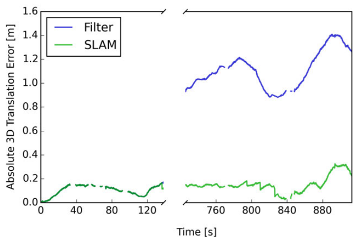
(e) *3D pos. error (LRU1, Exp. #2)*



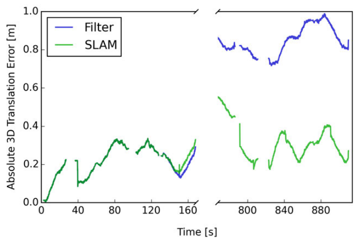
(f) *3D pos. error (LRU2, Exp. #2)*



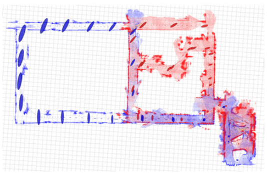
(g) *Map (Exp. #3)*



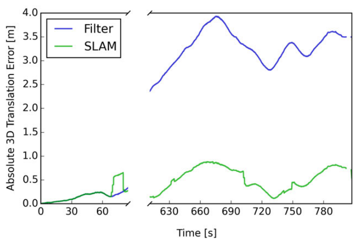
(h) *3D pos. error (LRU1, Exp. #3)*



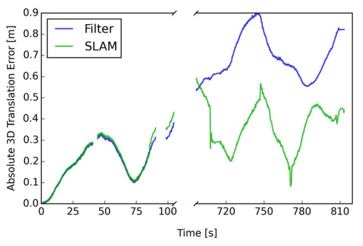
(i) *3D pos. error (LRU2, Exp. #3)*



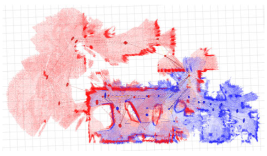
(j) *Map (Exp. #4)*



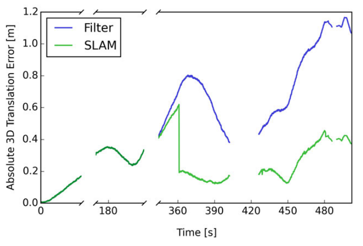
(k) *3D pos. error (LRU1, Exp. #4)*



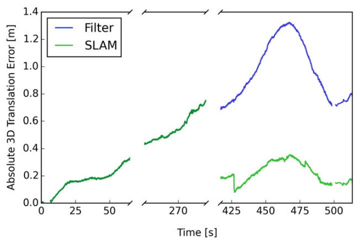
(l) *3D pos. error (LRU2, Exp. #4)*



(m) *Map (Exp. #5)*



(n) *3D pos. error (LRU1, Exp. #5)*



(o) *3D pos. error (LRU2, Exp. #5)*

Figure 8.27: *Maps and 3D trajectory errors for LRU1 (blue) and LRU2 (red) in our five multi-robot experiments. Gaps in the graph are due to a lack of ground truth for the respective areas, jumps in the error values for the SLAM estimate indicate loop closures.*

		Multi-Robot		Single-Robot	
		LRU1	LRU2	LRU1	LRU2
Number of robot poses x_i^r		26	26	0	0
Number of submaps s_i^r		29	24	29	24
Number of submap matches c_i	**per robot**	2	1	0	3
	inter-robot	9		0	
Number of robot detections d_i		8	19	0	0
Total number of nodes θ_i		105		29	24
Total number of factors f_i		143		29	27
Total driven distance [m]		199.82	170.85	199.82	170.85
Ground truth available [m]		61.89	51.75	61.89	51.75
Mean 3D trajectory error [m]		**0.29**	**0.40**	1.64	1.62
Mean angular error [°]		**3.08**	**2.87**	0.58	5.78

Table 8.9: *Comparison of trajectory and graph statistics (number of nodes and factors) for multi-robot and single-robot SLAM for Experiment #2 presented in Figure 8.24*

finish its, then non-empty, queue. This particular limitation in our experimental setup leads to lower numbers of loop closures than would have been possible to compute by our matcher in the respective environments. Second, in all five experiments, we did not move the pan-tilt unit of LRU1 at all and only rarely used that of LRU2. The rovers' thus limited fields of view lead to submaps that contain less information and thus are harder to match than those created during our single-robot exploration experiments, during which LRU regularly performed 360° camera scans.

In Experiment #3, an erroneous map match at a hallway crossing led to distortions of the map and pose estimates. It was caused by an incorrect data association between two submaps of LRU2. To a large degree, the error could be compensated by a number of later loop closures. The final map, visualized in Figure 8.23(c), only exhibits a small offset at the crossing with a slight tilt of the upper part of the map, and the average pose errors recorded within our tracking area are even below those for Experiment #2. This highlights the importance of maximizing the number of loop closures in order to gain robustness w. r. t. such failure cases, which are impossible to rule out when working on limited amounts of noisy input data.

In Table 8.9, we compare the results of Experiment #2 to running single-robot SLAM on the same dataset. The mean SLAM 3D trajectory error for the robots in the tracking area is 0.29 m and 0.40 m, thus altogether less than 0.19 % of their total trajectory length. The differences in numbers of intra-robot map matches between the two setups shown in Table 8.9 likely stem from differing submap pose and uncertainty estimates as input to the map matcher, whereas the low angular error for LRU1 in the single-robot setup likely results from accumulated errors coincidentally canceling each other out. In the single-robot case, LRU1 did not manage to compute map matches before the end of the experiment, at which the matcher process was interrupted. Thus, in this case, its SLAM trajectory matches that of the filter solution. As for our early experiments with Pioneer rovers (Section 8.3.1), the higher positional accuracy in the multi-robot setup indicates the benefit of joint graph optimization compared to estimating a single relative transformation between the robots' coordinate frames in order to just connect their maps. Both robots act as "moving landmarks" for each other, leading to a weighted distribution of the errors, as can be observed in particular for the angular errors. As we perform a joint optimization over the data of both rovers, inter-robot loop closures improving the accuracy of one rover might lead to a, usually smaller, degradation of the other's. This can be observed in Figure 8.27(e) and Figure 8.27(f), showing the positional errors of LRU1 and LRU2 in Experiment #2. Between 580 s and 600 s, the error of LRU2 decreases due to an optimization on inter-robot loop closures, while the error for LRU1 rises slightly. However, in total, a joint optimization brings benefits for all participating robots as the loop closures of one robot help to improve the estimates of the other as well.[5]

8.4 Computational Resources

In this section, based on our journal article by Schuster et al. 2018, we discuss the demand on computational resources posed by our localization and mapping components as well as the bandwidth required for inter-robot data exchange. Our goal is to show the suitability of our decentralized and distributed approach for online and on-board computation of pose and map estimates that can guide multi-robot teams working under bandwidth-constrained conditions.

In our distributed system, each robot processes all high-frequency as well as high-bandwidth data locally (raw stereo streams: 38.75 MB/s at 14.3 Hz and 1.25 MB image size). Thus, they only need to share aggregated 3D data in terms of submaps as well as a small set of filter and robot detection estimates. The average size of a submap was, for example, 700 KB in the multi-robot LRU Experiment #2, which we

[5]We show the multi-robot mapping process of our Experiment #2 in the first half of the video available at `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21812`

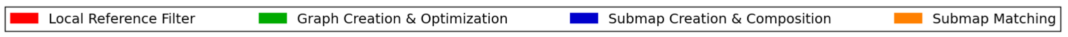
presented in Section 8.3.2. Submaps were exchanged between the rovers whenever a submap was finished, i. e., after a frame switch, which resulted in an average rate of 0.04 Hz. The bandwidth required for the exchange of submap data between the rovers thus was 58 KB/s. This allows a transmission over low-bandwidth connections like, for example, Global System for Mobile Communications (GSM) networks and facilitates an upscaling to setups with a larger number of robots.

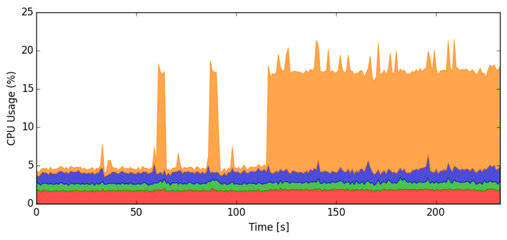
We base our evaluation and discussion on the multi-robot LRU experiments that we introduced in Section 8.3.2. For this, we acquired statistics on runtimes, computational load and memory on desktop computers, one per robot, using a synchronized playback of recorded sensor data at framerate. Their Intel Xeon E5-1620 CPUs (4 real / 8 virtual cores) have similar performance ratings as the Intel i7-3740QM CPUs on our LRU robots. All runtime measurements of particular processing steps refer to wall clock durations, not pure processing times. In particular the maximum computation times can thus be significantly affected by interrupts and waits caused by other processes on the non-real-time systems.

In Figure 8.28 and Figure 8.29, we present the CPU and memory usage of the major components of our SLAM system for our two robots LRU1 and LRU2 respectively. In the following paragraphs, we discuss these results together with runtime measurements exemplarily taken for LRU2 in Experiment #2:

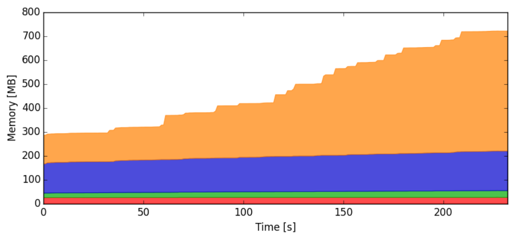
- *Local Reference Filter (Section 4.1)*
  Our local reference filter runs two threads, one for the Strap-Down Algorithm (SDA) integrating IMU data and one for the Kalman Filter (KF) updates themselves. As expected, the filter exhibits approximately constant CPU usage and memory requirements, making it real-time capable. We measured a mean / max. runtime per iteration of < 0.01 ms / 1.2 ms for the SDA and 0.2 ms / 2.7 ms for the filter updates respectively.

- *Graph Creation & Optimization (Section 4.2)*
  The computation in our graph SLAM component is based on a single main thread, apart from a helper thread to publish transformations at regular intervals. The plot of its CPU usage shows a constant part and small spikes of short duration but increasing size. The spikes relate to graph optimization steps performed on large loop closures on the constantly growing graph. We measured a mean / max. runtime of 1.7 ms / 162 ms per iteration for the graph optimization step that computes the maximum likelihood pose estimates. The more expensive part is the computation of the covariance estimates for all submap origins with a mean / max. runtime of 106 ms / 341 ms respectively. In our current implementation, we compute this on every graph update in order

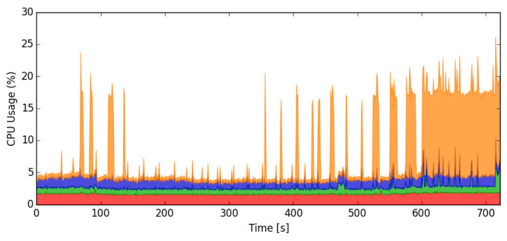

🟥 Local Reference Filter
🟩 Graph Creation & Optimization
🟦 Submap Creation & Composition
🟧 Submap Matching



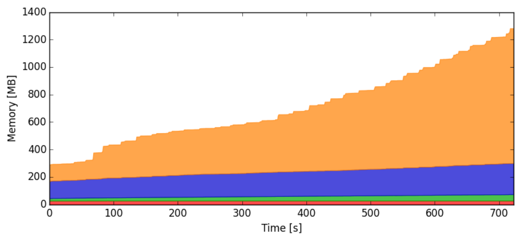
***(a)** CPU usage over time (Exp. #1)*



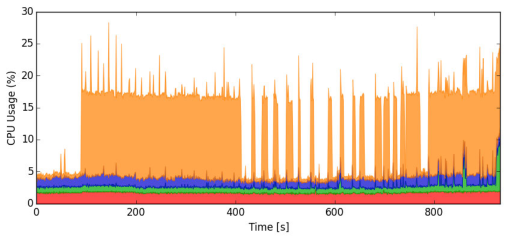
***(b)** Memory usage over time (Exp. #1)*



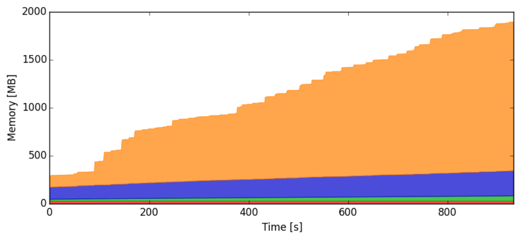
***(c)** CPU usage over time (Exp. #2)*



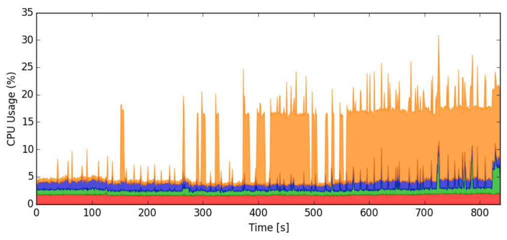
***(d)** Memory usage over time (Exp. #2)*



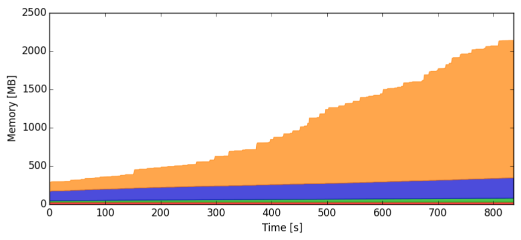
***(e)** CPU usage over time (Exp. #3)*



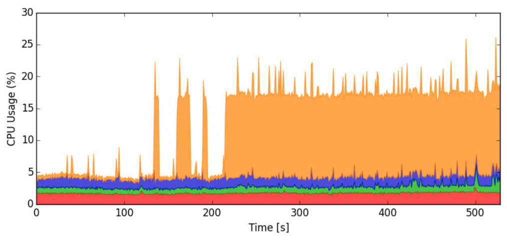
***(f)** Memory usage over time (Exp. #3)*



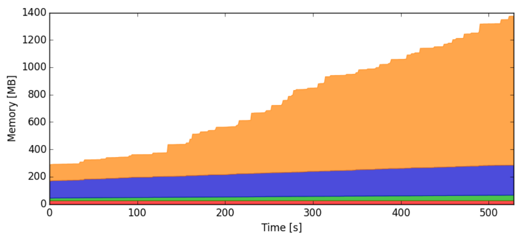
***(g)** CPU usage over time (Exp. #4)*



***(h)** Memory usage over time (Exp. #4)*



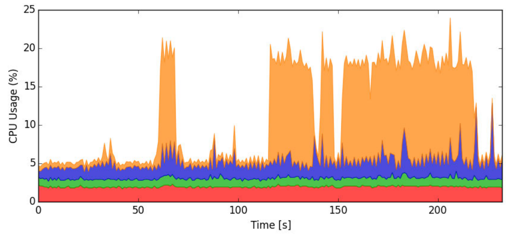
***(i)** CPU usage over time (Exp. #5)*



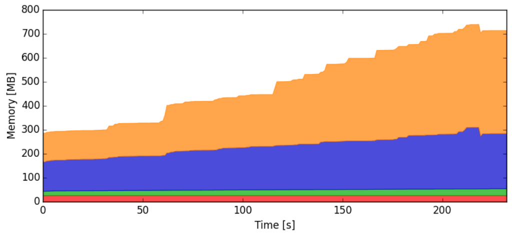
***(j)** Memory usage over time (Exp. #5)*

***Figure 8.28:** Stacked area plots of the computational resources used by our localization and mapping components to process the data for LRU1 in our five multi-robot experiments presented in Section 8.3.2 (100% CPU usage $\widehat{=}$ all cores)*

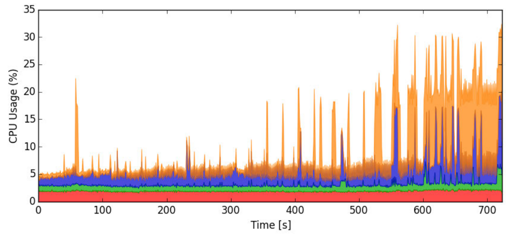
🟥 Local Reference Filter 🟩 Graph Creation & Optimization 🟦 Submap Creation & Composition 🟧 Submap Matching



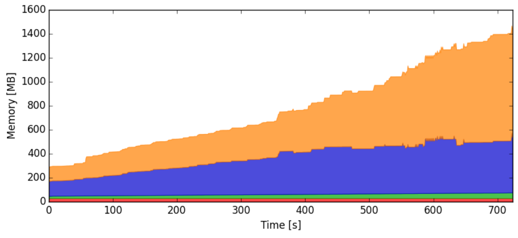
(a) *CPU usage over time (Exp. #1)*



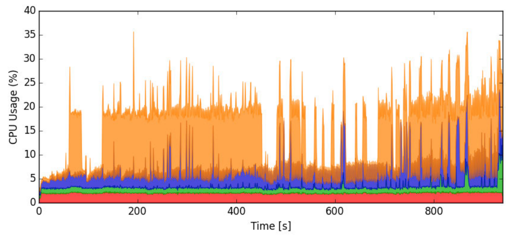
(b) *Memory usage over time (Exp. #1)*



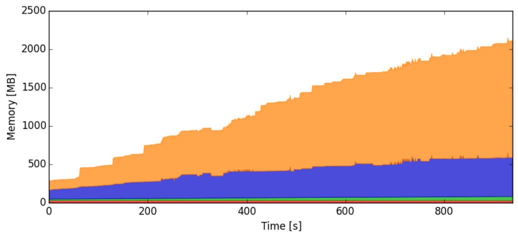
(c) *CPU usage over time (Exp. #2)*



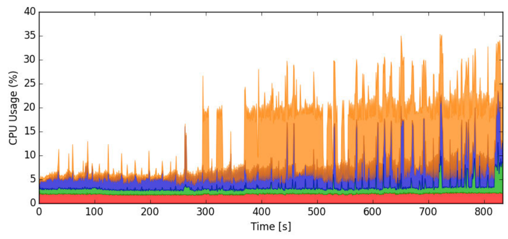
(d) *Memory usage over time (Exp. #2)*



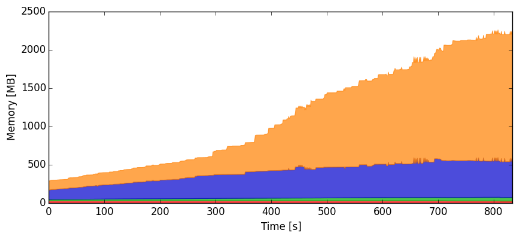
(e) *CPU usage over time (Exp. #3)*



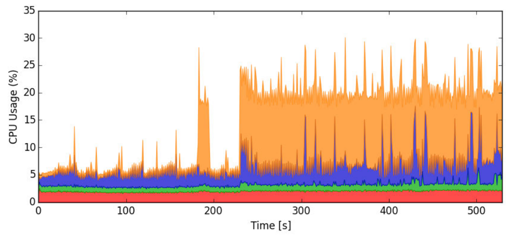
(f) *Memory usage over time (Exp. #3)*



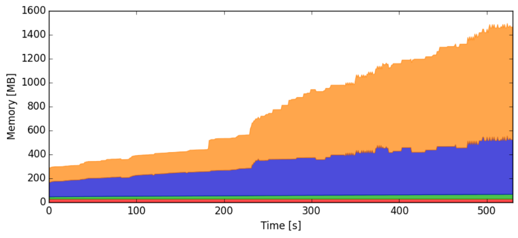
(g) *CPU usage over time (Exp. #4)*



(h) *Memory usage over time (Exp. #4)*



(i) *CPU usage over time (Exp. #5)*



(j) *Memory usage over time (Exp. #5)*

Figure 8.29: *Stacked area plots of the computational resources used by our localization and mapping components to process the data for LRU2 in our five multi-robot experiments presented in Section 8.3.2 (100% CPU usage $\widehat{=}$ all cores)*

to use the most recent values in the map matcher's internal heuristics. It would, however, be sufficient to update these values only after significant changes due to impactful loop closures. An appropriate heuristic is a topic for future work.

- *Submap Creation & Composition (Section 5.1.2 and 5.2.2)*
  We employ two separate threads, one to integrate the stream of 3D data into submaps and one to compose the full multi-robot map at regular intervals (0.5 Hz). The composition of all submaps into a global joint map is an expensive operation, the worst-case computational effort of which increases linearly with the number of submaps. In order to visualize its impact in this evaluation, we executed it only on LRU2, which is reflected in the differences in the CPU usage graphs for LRU1 and LRU2. In many applications, it would be sufficient to compose a full global map only on demand, e. g., when requested by global planning components or operators for visualization, or only compute it for limited areas of interest. The CPU usages of the two threads mainly correspond to the constant part and the spikes of increasing size respectively, the latter only being present in Figure 8.29 for LRU2. The memory consumption and effort for composing a full map grow over time since we do not remove old submaps yet. We measured a mean / max. runtime of 0.01 s / 0.04 s for the integration of new data. The composition of the full map took a mean / max. runtime of 0.26 s / 3.27 s, with the merging of probabilistic voxel-grid submaps being particularly expensive. We cache intermediate results, so that a full computation is only required after significant changes of the submap poses due to impactful loop closures.

- *Submap Matching (Section 6.1)*
  The map matcher runs as a background process with lower priority. It consists of two threads, one for preparing potentially matching pairs and one for the matching itself. Their CPU usages mainly correspond to the constant part and the spikes respectively. Toward the end of the experiment, as large numbers of overlapping submaps and thus match opportunities are available, the matching thread utilizes one CPU core continuously to process its prioritized working queue. It would be straightforward to further parallelize the matcher to process multiple elements from the queue simultaneously. The memory consumption for the map matcher grows faster than for the submapping component as it stores several internal representations, such as the feature descriptors, in its cache. When memory limitations become relevant, these caches could easily either be swapped to disk or deleted and recreated on demand. We measured a mean / max. runtime of 0.39 s / 0.67 s for keypoint selection and feature generation (per map), 3.07 s / 8.71 s for feature matching (per selected pair), 0.02 s /

0.05 s for Hough3D voting, including its RANSAC steps, and 2.38 s / 4.81 s for the ICP refinement. The most expensive steps are the descriptor matching, corresponding to a nearest neighbor search in a high-dimensional space, and the ICP. The former was executed 38 times for LRU2, whereas the final ICP refinement step is only computed for almost certain matches, 5 times in this experiment. While thus optimizations in the other steps might not yet be worth much effort, for future work, we plan to look into lower-dimensional feature descriptors to speed up the matching.

We designed our global optimization to build upon the filter results and thus create graphs with only a small number of nodes. As presented here, we thereby can achieve fast online optimization steps. Compared to the other components, in particular those processing 3D vision data, the overall computational load and memory consumption of the graph optimization is almost negligible. This makes it suitable even for resource-constrained systems. The overhead from running the optimization of the full graph on each robot separately is acceptably low and guarantees an online global estimate on all systems with all available information even during communication losses, thus increasing the robustness of a multi-robot team.

Note that we did not yet optimize most of the implementations of our algorithms w. r. t. their runtime, processing requirements, and memory consumption. Thus, we expect potential for significant future improvements on the values presented above.

8.5 Summary and Discussion

In this section, we summarize the results of our experimental evaluations and map them to the functional and non-functional challenges regarding autonomous planetary exploration that we identified and discussed in Section 2.1.

- *Stereo Vision-Based Obstacle Mapping (Section 8.1.1)*
  We evaluated the suitability of our stereo-error adaptive obstacle classification to support the re-localization in previously visited areas (place recognition) in outdoor, indoor, and mixed environments. With a 3D RBPF-based SLAM approach, we were able to achieve final position deviations below 0.08 % of the full trajectory lengths for both narrow- and wide-angle camera setups. Using stereo cameras, we satisfy the requirement for a space-qualifiable sensor setup.

- *4D Map Matching (Section 8.1.2)*
  We successfully evaluated our map matching method for place recognition in our 6D SLAM framework in a total of 53 experiments in unstructured outdoor and semi-structured indoor environments. It is based on the results of our obstacle classification, which we aggregate together with stereo vision-based 3D data into local submaps. With a reduction of the dimensionality of the matching problem from 6D to 4D, we achieved an increase in the number of map matches of 40 % on average. A higher number of loop closure constraints increases the robustness of the global pose and map estimation, adding tolerance to individual faulty measurements.

- *Uncertainty Estimation for Marker-Based Detections (Section 8.1.3)*
  We demonstrated an improved localization accuracy stemming from our uncertainty estimation method for the 6D pose estimation of visual marker detections. These allow us to meet the requirement for mutual robot localization, while dealing with propagated sensor noise and various types of error sources along the camera-based detection and estimation pipeline.

- *Graph SLAM with Map Matching (Section 8.2.1)*
  We evaluated a first integrated version of our SLAM framework, combining graph optimization for global self-localization with the creation of dense 3D point cloud maps from noisy stereo data and 6D map matching for place recognition. Compared to our RBPF-based 3D SLAM framework mentioned above, we achieved an improvement on the mean 2D position error of 27 % and 50 % in our outdoor and indoor experiments respectively.

- *Graph Topology for Local Reference Filter Estimates (Section 8.2.2)*
  We evaluated our novel graph topology for the integration of local reference filter estimates into our SLAM graph. In our experiments, this combination of local and global methods according to their dependencies and uncertainty estimates led to an improvement of the 3D self-localization accuracy of 15 % on average.

- *Collaborative 6D Multi-Robot SLAM (Section 8.3)*
  We evaluated our full multi-robot SLAM system in a total of seven experiments, featuring different camera setups and robots. It was able to generate accurate joint global maps from the robots' stereo data and estimated their trajectories with average 3D translational errors below 0.5 m w. r. t. partially available ground truth, i. e., errors below 0.5 % of the robots' respective total trajectories. Our separation of local and global methods allowed us to distribute local computations on high-frequency and high-bandwidth data among the individual

robots. Thereby, we achieved fast global optimization steps on a small SLAM graph with less than 180 nodes and 250 factors in all of our experiments.

- *Computational Resources (Section 8.4)*
  For one of our multi-robot experiments, we evaluated the runtimes of our key localization and mapping components as well as their requirements on computational resources, memory, and bandwidth for multi-robot communication. We could run all algorithms online on on-board (or equivalent) processing hardware. Sharing aggregated map data instead of raw image streams between robots allowed us to significantly reduce the required bandwidth, from 38.75 MB/s to 58 KB/s.

All of our SLAM components relate to the non-functional challenge of the *handling of uncertainties* as our components employ probabilistic frameworks where appropriate to deal with the high levels of measurement uncertainties and noise that are unavoidable for camera-based systems. In Table 8.10, we summarize the mapping of our evaluations to the challenges identified in Section 2.1. While it is impossible to thoroughly evaluate each and every aspect of a complex multi-robot SLAM system within the scope of a thesis, we show that our chosen experiments and analyses cover the full spectrum of functional and non-functional challenges regarding autonomous robots for planetary exploration.

As further evaluations that go beyond this thesis, we propose for future work a comparison of our full SLAM system, as a distributed and sub-optimal data fusion method, to full batch optimization. In addition, it would be interesting to compare it to other SLAM frameworks on the same datasets or benchmarks. A framework for multi-robot SLAM, however, involves many application-dependent design decisions and trade-offs that are hard to quantify in a single metric. Throughout this thesis, we thus restricted most comparisons to other SLAM frameworks to discussions regarding these trade-offs, besides a direct comparison for single-robot SLAM to the widely-used RBPF-based GMapping algorithm in Section 8.2.1. A further interesting aspect would be a direct evaluation of the map quality. Reasonable metrics for that are, however, application-dependent and a topic of ongoing research in the SLAM community.

In addition to the evaluations presented in this thesis, several further aspects and applications of our SLAM framework have been evaluated by my colleagues and me in the following publications:

- *Filter Consistency and Stability (Schmid et al., 2014b)*
  Schmid et al. (2014b) demonstrated the consistency and long-term stability of the local reference filter. For this evaluation, they used a quadcopter as a

robot that poses additional challenges due to its inherent instability compared to ground-based robots. They evaluated a simulated 24 h quadcopter flight and showed in real quadcopter experiments the applicability of the local reference filter for the control of highly dynamic systems with limited computational resources.

- *Long-Range Navigation Datasets (Vayugundla et al., 2018)*
  We discussed and published two long-range navigation datasets with the LRU rover in a Moon-analogue site on Mt. Etna, Sicily, Italy. Besides raw sensor

		Robot Localization	Local Mapping and Obstacle Class.	Robot Detection	Place Recognition	(Collaborative) Global Mapping	Space-Qualifiable Sensor Setup	Handling of Uncertainties	Online and Real-time Algorithms	Fault Tolerance	On-board Computation	Low-Bandwidth Communication	Distributed Computation
Evaluation	**Sec.**	**Functional**					**Non-functional**						
Stereo Vision-Based Obstacle Mapping	8.1.1		✔		✔		✓	✔	✓	✓	✓		
4D Map Matching	8.1.2				✔		✓	✓	✓	✔	✓		
Uncertainty Estimation for Marker-Based Detections	8.1.3			✔			✓	✔	✓	✓	✓		
Graph SLAM with Map Matching	8.2.1	✔	✓		✔	✓	✓	✔	✓	✓	✓		
Graph Topology for Local Reference Filter Estimates	8.2.2	✔						✔	✓		✓		
Collaborative 6D Multi-Robot SLAM	8.3	✔	✓	✔	✓	✔	✓	✔	✓	✓	✓	✓	✔
Computational Resources	8.4								✔		✔	✔	✔

Table 8.10: *Mapping between the functional and non-function challenges for SLAM systems for autonomous planetary exploration (see Section 2.1) and our evaluations addressing them directly (✔) or indirectly (✓)*

data, they include the local reference filter estimates based on a fusion of IMU, visual-, and wheel-odometry data.

- *Integration of Multiple Visual Odometries (Müller et al., 2018)*
  We integrated multiple visual odometry estimates on a multicopter equipped with ultra wide-angle (fisheye) cameras for stereo vision and demonstrated the robustness of our state estimation against the failure of individual visual odometries. We ran our global optimization and mapping modules on-board the MAV to create a dense 3D model of the environment. As an advantage of our decoupling of high-frequency local estimation and global optimization, we only needed to integrate the additional visual odometry estimates into the local reference filter. Thus, the SLAM graph was unaffected by the change, neither increasing in size nor in complexity.

- *Force Feedback Computed From Obstacle Maps (Panzirsch et al., 2018)*
  We successfully employed our local obstacle maps for force-feedback teleoperation of the LRU rover over delayed communication channels with round-trip times of up to 800 ms. For this, we computed a virtual force that directed the operator away from obstacles in the robot's assumed path.

- *Single-Robot Autonomous Exploration (Lehner et al., 2017)*
  We applied the global probabilistic voxel-grid maps computed by our SLAM framework for information gain-based autonomous single-robot exploration with active loop closing.

In Chapter 9, we present further applications of our localization and mapping framework on our robot systems in Moon- and Mars-like as well as Moon-analogue environments.

Chapter **nine**

Mapping in Action: Applications and Demonstrations

In this chapter, we present demonstrations of applications of our localization and mapping methods that we conducted in order to validate our approach against the objectives defined in Section 1.2 at the beginning of this thesis. For this, we left our laboratory settings and publicly demonstrated our SLAM system on robots operating in environments that simulate the terrain and visual appearance of lunar and planetary surfaces.

We first demonstrated on-board and online waypoint navigation and mapping for autonomous obstacle avoidance at the ILA 2014 trade fair, running our algorithms on a planetary exploration rover prototype with a stereo vision-based sensor setup (Section 9.1). In 2015, we succeeded at the SpaceBotCamp challenge, a complex robotic mission in a Moon-like scenario. The fully autonomous robot system therein relied on our localization as well as on our local and global 3D mapping components (Section 9.2). As the next big step, we deployed our rovers in a Moon-analogue scenario on Mt. Etna, Sicily, Italy in 2017 (Section 9.3). In this challenging rough-terrain outdoor environment, we demonstrated the application of our global multi-robot mapping system to autonomous exploration planning for a team of two rovers. In 2018, we set up a Mars-like environment at the IAC and therein showed joint localization and mapping with a heterogeneous team consisting of flying and driving robots, which feature very different camera setups and points of view (Section 9.4).

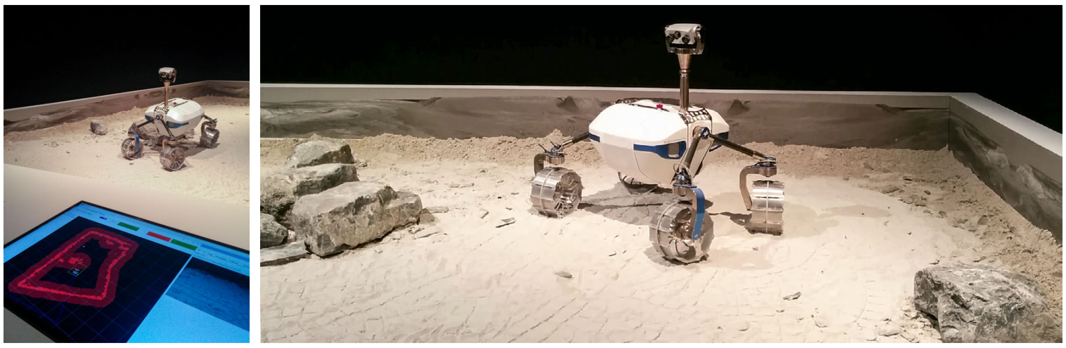


Figure 9.1: *Impressions of our demonstration in the Space Pavilion of the ILA 2014 trade fair: We showed navigation and mapping with our rover prototype LRU in a Moon-like scenario.*

9.1 ILA 2014: 3D SLAM for Autonomous Waypoint Navigation

We presented an initial version of our navigation and mapping algorithms on our planetary exploration rover prototype LRU at the 2014 *Innovation and Leadership in Aerospace (ILA)* trade show (also known as *ILA Berlin Air Show*). It is one of the world's largest aerospace trade shows, which in 2014 had a total of 227,000 visitors during its six days (Messe Berlin, 2014). We set up a Moon-like scenario in a sandbox of approx. 25 m^2 inside the *Space Pavilion*, featuring soft sand as well as several large stones as obstacles for our rover, as shown in Figure 9.1.

We alternated between different demonstrations to show our system's capabilities and to highlight the benefits of autonomous robot operation. Visitors could experience the difficulties arising from time delays between Earth and the Moon by controlling the rover via a SpaceMouse interface. For this, we added a delay of several seconds to both the control input and to the visual feedback, i. e., the camera image streams, in order to simulate the communication round-trip time. As an easier-to-use alternative, we provided a map visualization as an interface to a waypoint navigation: The rover then autonomously approached target points set by the user while avoiding all obstacles on its way. For longer-running demonstrations, we had the rover navigate back and forth between two waypoints located at opposite ends of our sandbox.

All pose and map estimates were computed online and on-board the robot. We employed a filter-based local state estimation with obstacle mapping for local navigation and computed global estimates via a RBPF-based SLAM framework, featuring 3D pose (x, y, yaw) estimation and 2D occupancy grid mapping. We developed this system, first presented by Brand et al. (2014) and summarized at the beginning of Section 8.1.1, as a predecessor to our graph-based 6D SLAM methods used in our

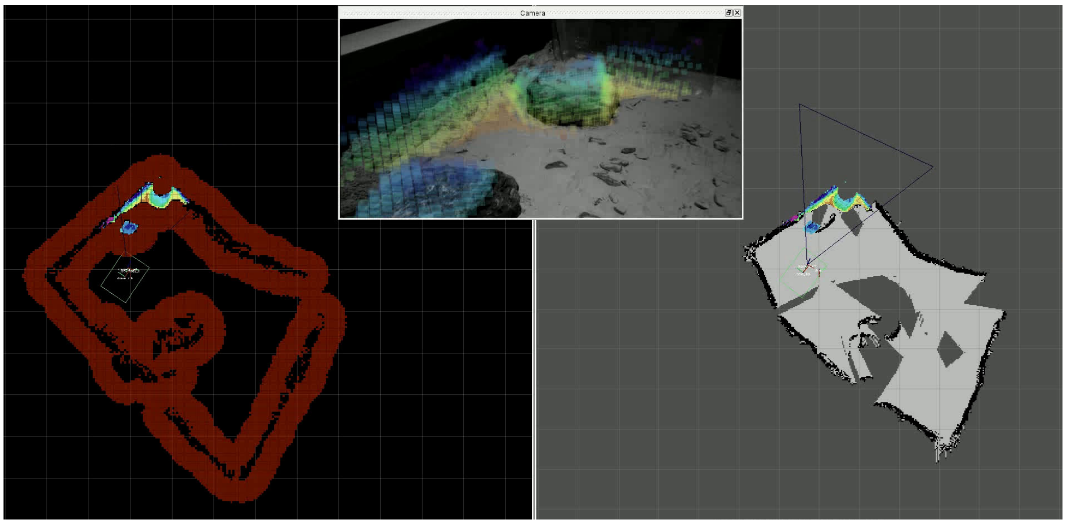


Figure 9.2: *Mapping at ILA 2014: height-colored obstacle classification results projected into the rover's camera image (top, center), inflated obstacle map for path planning (left) and 2D occupancy map created by our RBPF SLAM (right)*

later experiments and demonstrations. In Figure 9.2, we give an impression of our obstacle classification results projected into the robot's camera image as well as of the obstacle and occupancy grid maps that were generated during an autonomous waypoint navigation demonstration.

9.2 SpaceBotCamp 2015: 6D SLAM for Waypoint Navigation in a Moon-Like Scenario

We demonstrated our 6D SLAM framework with great success in a Moon-like environment at the SpaceBotCamp in 2015, as we presented in our conference and journal publications by Schuster et al. (2016, 2017). The SpaceBotCamp is a national German robotics challenge that was organized by the DLR Space Administration. Similar to the DARPA Robotics Challenges (DRC) (Orlowski, 2016) in the USA, its goal is to stimulate innovations, benchmark state-of-the-art technologies in the field of autonomous mobile robotics, and kick-start their integration into working systems. It took place in November 2015 in Hürth, Germany, with ten participating teams from universities and research institutes from all over Germany. All teams had to face a challenging task focused on autonomous exploration and manipulation in an unstructured, previously unknown GNSS-denied Moon-like environment.

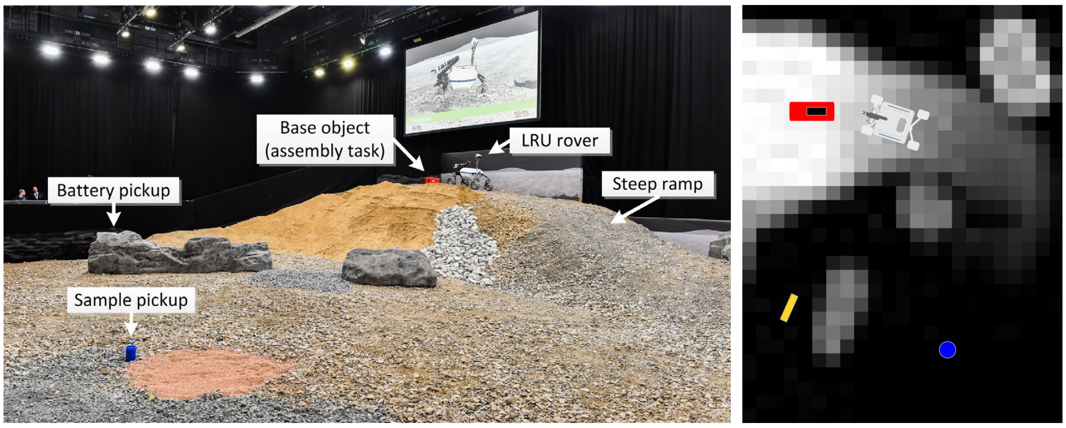
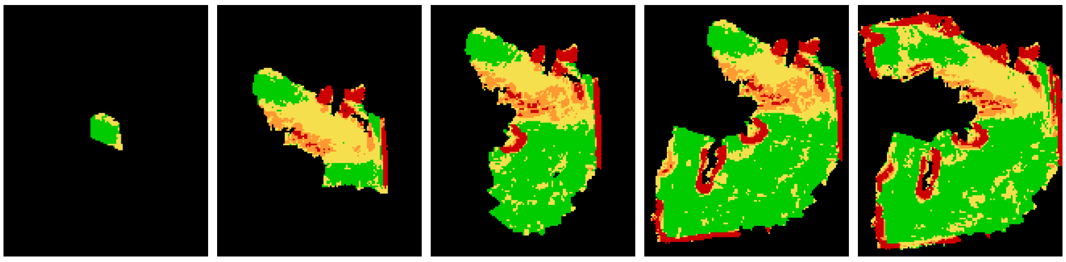


Figure 9.3: *Overview of the SpaceBotCamp scenario. The competition field had an area of* $13\,m \times 18\,m$*. The top-down view is underlaid with the given rough height map of the scenario with a resolution of* $1\,px \widehat{=} 0.5\,m$*. We scaled the sketched LRU and objects up by factors two and four respectively for better visibility. The LRU started next to the red base station on top of a hill of approx.* $2\,m$ *height. In the foreground, a blue container filled with a rock sample can be seen, whereas a yellow battery object is hidden by a large rock in the left image.*

9.2.1 The SpaceBotCamp Scenario

In the SpaceBotCamp scenario, an autonomous robot system, consisting of one or multiple robots with a total mass of less than 100 kg, had to autonomously explore and map an area modeled after a Moon-like, rough-terrain planetary surface. Therein, it had to locate and collect both a blue container with a rock sample (approx. 500 g) as well as a yellow object representing a battery (approx. 800 g). Both objects had to be transported to a red base station and finally assembled. In Figure 9.3, we give an overview showing the scenario and the mission-relevant objects therein. A high level of local robot autonomy was necessary to fulfill the given tasks since an artificial delay of four seconds round trip time was additionally added to the communication link in order to simulate the real delay between Earth and the Moon. The bandwidth was limited to 100 Mbit/s, a data rate that has been reached via Ka-band radio for lunar-Earth distances by the *Lunar Reconnaissance Orbiter* launched in 2009 (Jennings et al., 2011). Furthermore, the uplink to the robotic system was completely blocked, except for up to three five-minute checkpoints. During these limited time frames, a ground station crew was allowed to send commands to the robotic system over a channel with the same delay and bandwidth limitation as the downlink. Apart from that, the teams could only passively monitor the robot. In addition, the ground station crew was located in a separate room and thus without any visual contact to the competition field. After two days of preparation, each team



(a) $t = 0$ min **(b)** $t = 1$ min **(c)** $t = 6$ min **(d)** $t = 10$ min **(e)** $t = 17$ min

Figure 9.4: *Sequence of 2.5D terrain classification maps created by LRU during its SpaceBotCamp run*

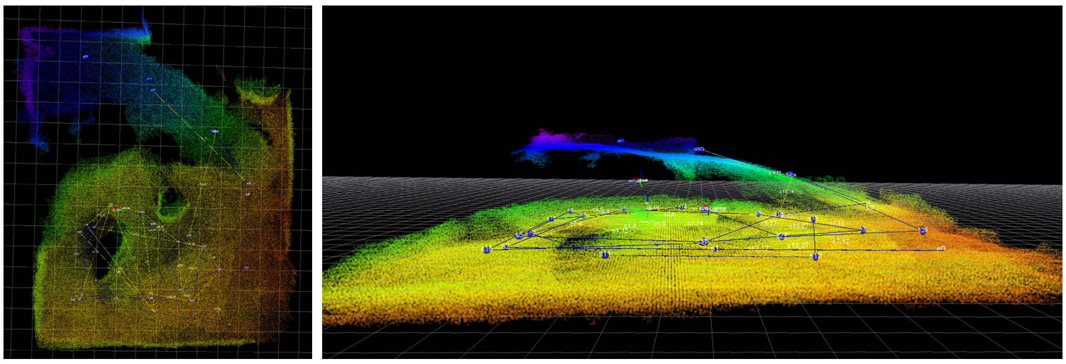


Figure 9.5: *Top-down and side view of height-colored 3D point cloud map of the SpacebotCamp scenario created by our SLAM components (blue ellipsoids: covariance of submap origin position estimates, blue and yellow lines: graph constraints from filter estimates and submap matches). The hill in top left corner, see also Figure 9.3, is clearly visible, its flank ,however, is missing in the point cloud as it was too steep to be observed from above and the LRU did not look up to see it from below.*

had a single opportunity to solve the challenge within a sixty-minute time slot in front of a public audience.

9.2.2 Navigation Methods

In the SpaceBotCamp challenge, a low-resolution elevation map ($1\,\text{px} \mathrel{\widehat{=}} 0.5\,\text{m}$) was available as a-priori knowledge, similar to satellite images that are obtainable prior to real planetary exploration missions. In order to find all target objects, we planned navigation waypoints for the LRU such that they covered the complete field. We therefore manually divided the area into grid cells sized according to the LRU's range of reliable perception. During the search and exploration parts of the mission, it scanned the area on each waypoint with its pan-tilt sensor head, covering a full 360° angle of view in order to detect the target objects.

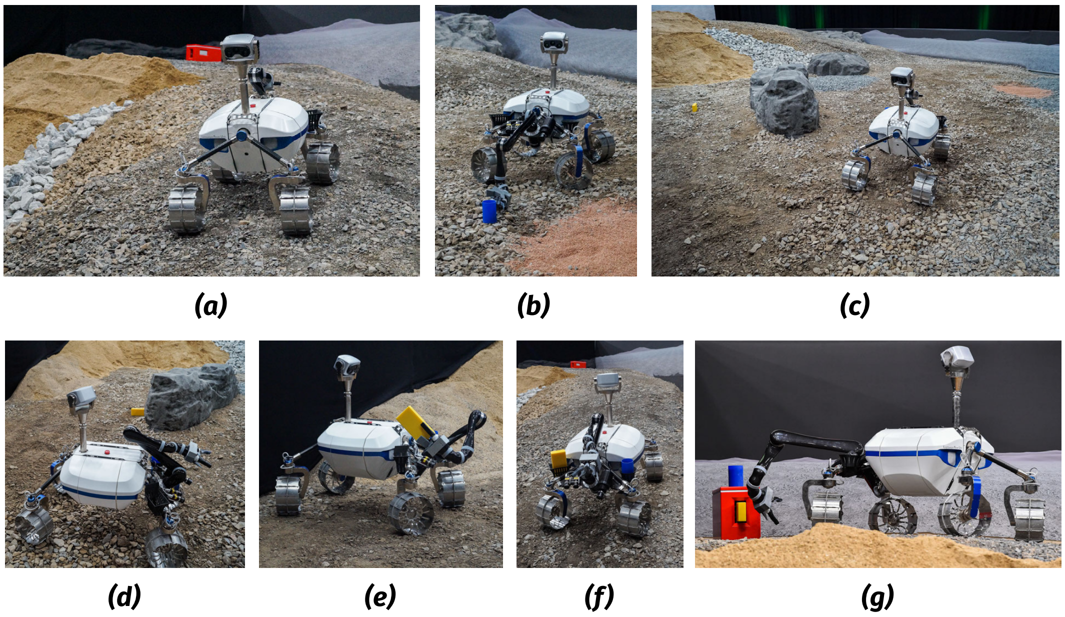


Figure 9.6: *Sequence of our SpaceBotCup run: LRU autonomously navigated to a waypoint at the bottom of the hill (a), located and picked up the blue container (b), explored several waypoints to search for the hidden yellow battery object (c), located it (d), picked it up (e), returned back to the red base station (f) and assembled the objects (g).*

The LRU used a fast, graph-based 2D path planner to navigate autonomously between waypoints, taking into account the surrounding obstacles via its local terrain classification maps. In Figure 9.4, we visualize a sequence of terrain classification maps for different points in time during the SpaceBotCamp run. In addition to local mapping for obstacle avoidance, we employed our 6D SLAM framework with submap matching to create a globally consistent 3D point cloud model of the environment, which we present in Figure 9.5. The height-colored point cloud allows to visually identify the steep ramp, large rocks, as well as the boundaries of the contest area.

9.2.3 Results and Discussion

In Figure 9.6, we present a series of photos showing LRU during its run at the SpaceBotCamp 2015. According to the original rules of the challenge, our team was the only one amongst the ten competitors to fulfill all mandatory tasks. In addition, our robot was able to localize itself in the environment solely relying on its on-board sensors and created a 3D map of the environment. Furthermore, we solved the tasks while facing all of the specified communication constraints from the very beginning of the mission. We accomplished this in just thirty minutes, half of the given time

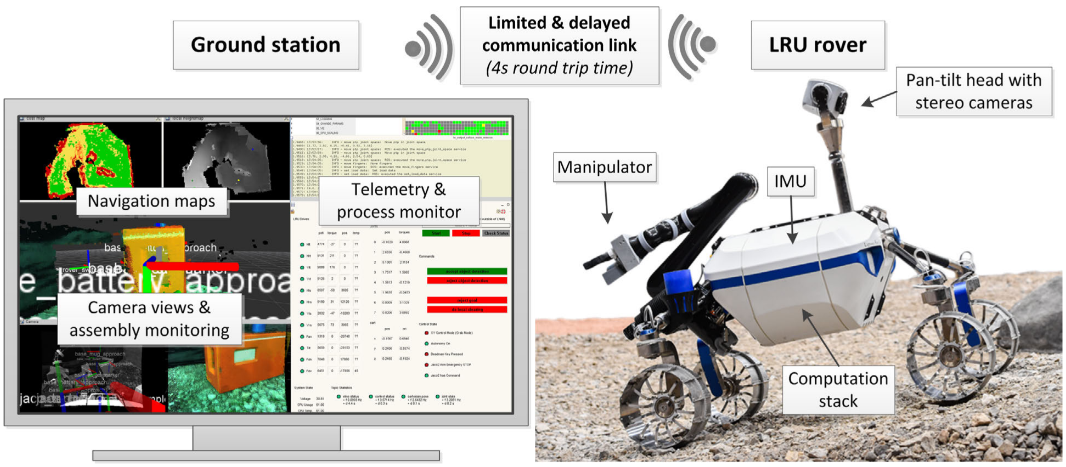

Ground station
Limited & delayed communication link (4s round trip time)
LRU rover
Pan-tilt head with stereo cameras
Manipulator
IMU
Navigation maps
Telemetry & process monitor
Camera views & assembly monitoring
Computation stack

Figure 9.7: *Ground station and communication setup for the SpaceBotCamp*

frame, and with full on-board autonomy. In contrast, many other teams softened the challenging communication restrictions in order to allow for mixed autonomy and teleoperation approaches. We only took a single one of the three allowed checkpoints to double-check the object localization for the base station, as its precision is crucial for flawless assembly. For this, we solely sent four high-level commands to the rover during the whole mission while experiencing the one-way delay of two seconds for all sent commands and received data. Thus, we could demonstrate that we still had full high-level control of the system despite the delayed and constrained communication link between the ground station and our rover. As we perform all processing on board the LRU, the bandwidth-limited communication channel was sufficient to monitor the system, as sketched out in Figure 9.7. In particular for the camera images and navigation maps, we downsampled the data both in frequency and resolution before sending them to the ground station. Furthermore, robust communication relays, as described in our journal article by Schuster et al. (2017), allowed us to deal with packet loss that occurred when approaching the limits of the unidirectional communication channel without acknowledgment mechanisms. Finally, our system was the only robot that managed to climb and descend the steep crushed-stone ramp, shown in Figure 9.3, fully autonomously based on its on-board localization estimates and obstacle classification maps.[1]

[1]We present a video of our run at the SpaceBotCamp 2015 at `https://youtu.be/wCTkSxcna8o`

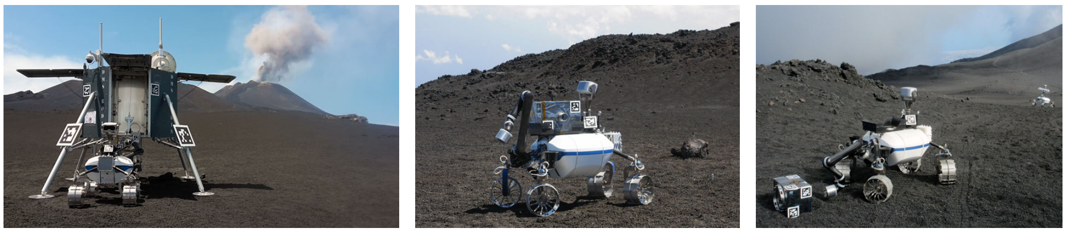


Figure 9.8: *Impressions of the ROBEX space-analogue mission to deploy seismic measurement instruments: picking up an instrument box from the lander (left), transporting it to a target location (center) and placing the box to take a measurement (right)*

9.3 ROBEX 2017: 6D SLAM for Autonomous Multi-Robot Exploration in a Moon-Analogue Environment

Our next step was to move from the previous demonstrations in Moon-like setups to a test campaign in a Moon-analogue environment: In the summer of 2017, we demonstrated our methods for autonomous planetary exploration at a site on the volcano Mt. Etna, Sicily, Italy, featuring a challenging rough-terrain outdoor environment. There we conducted experiments as part of the Helmholtz Alliance Robotic Exploration of Extreme Environments (ROBEX), an association of sixteen universities and institutes that perform space and underwater research in extreme environments (Kanzog, 2017; Wedler et al., 2017, 2018a).

9.3.1 The ROBEX Space-Analogue Mission

The objective of the ROBEX space-analogue mission is to study the lunar crust model with the help of seismic measurements. As we sketched out in our exploration scenario in Section 1.1, seismic measurements can not only directly lead to scientific insights but, in future missions, could also support the discovery of subterranean structures, such as water ice depots or lava tubes, which are worthwhile targets for closer inspection. Mt. Etna was selected as a well-suited Moon-analogue test site as it exhibits natural seismic activity at depths similar to lunar deep quakes. As an important aspect with regard to robot navigation, the volcanic rough-terrain environment is also visually similar to the surface of the Moon.

In the ROBEX main experiment, one of our LRU rovers autonomously picked up seismic measurement instruments from a lander mockup and deployed them at predefined target locations, as shown in the photos of Figure 9.8. We successfully

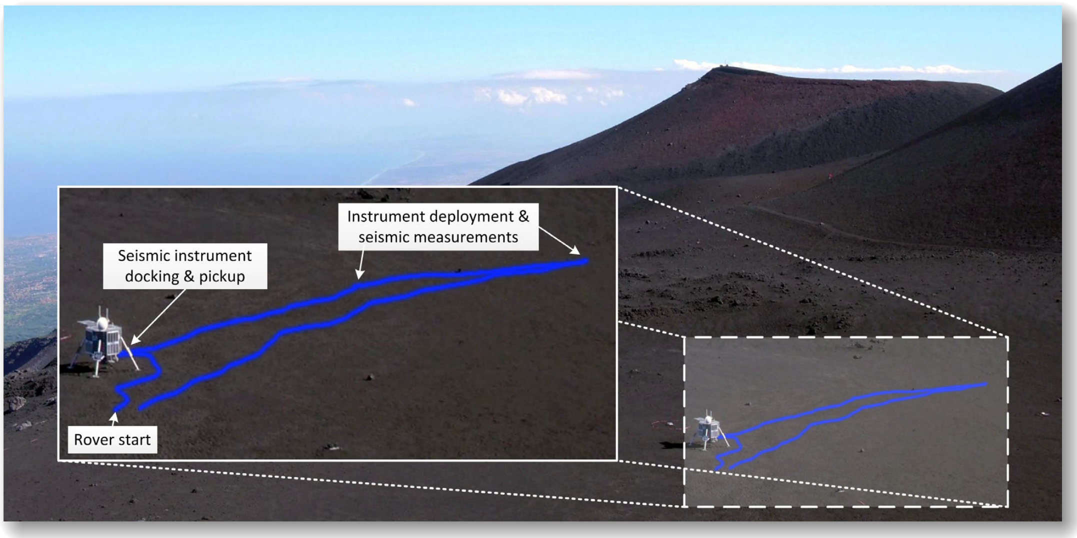

Instrument deployment & seismic measurements
Seismic instrument docking & pickup
Rover start

Figure 9.9: *Semi-automated Google Earth export of the trajectory estimated by LRU during one of the ROBEX experiment to deploy seismic measurement instruments*

employed our SLAM framework for single-robot local and global localization and mapping during this mission, which we describe in more detail in our conference paper by Wedler et al. (2017). In Figure 9.9, we visualized the robot trajectory as estimated on board LRU during one of our experiments. In addition to the seismic measurement mission, we used the test site for further localization, mapping, and exploration experiments. Among others, we recorded long-range navigation datasets with trajectories of up to 1 km in order to evaluate incremental localization and published these datasets from the Moon-analogue environment publicly for other researchers to use (Vayugundla et al., 2018). As the focus of this thesis is on multi-robot SLAM, next we will present the methods and results for one of our additional multi-robot 6D mapping experiments, during which our two rovers used their on-board global joint localization and map estimates to coordinate their autonomous exploration strategy online.

9.3.2 Collaborative Multi-Robot Exploration

We conducted a preliminary collaborative exploration experiment with our two LRU rovers at the Moon-analogue test site on Mt. Etna, depicted in Figure 9.10, in order to demonstrate the applicability of our multi-robot mapping methods for planetary exploration. We first presented and discussed this experiment in our journal article by Schuster et al. (2018). For the experiment, we ran a frontier-based

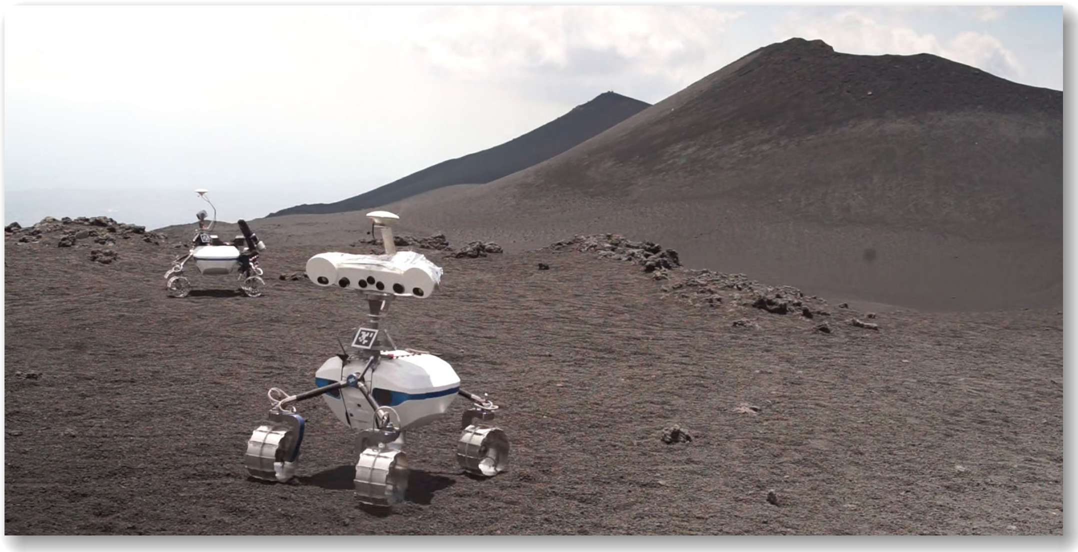


Figure 9.10: *Our two LRU rovers during an autonomous multi-robot exploration experiment at a Moon-analogue test site on the volcano Mt. Etna, Sicily, Italy*

exploration algorithm (Yamauchi, 1998) that employs our global probabilistic 3D voxel-grid maps to compute the expected information gain at each new exploration goal location, as described by Lehner et al. (2017). This information is used to rank the goals and select the respective next location to be explored. To employ these methods for collaborative multi-robot exploration, we extended them in order to spatially distribute goal locations between robots. Each robot communicates each new exploration goal to all other robots and enforces a minimum distance between exploration goals of different robots of at least 7 m, approximately two times the sensor range of our rovers when pointing their pan-tilt units towards leveled ground. Both rovers planned the paths to avoid obstacles on their way to their respective exploration goals based on their local terrain classification maps. In Figure 9.11, we superimposed images from one rover's navigation camera with these maps in order to give an impression of the navigational challenges at our test site. For the experiment, we defined a target region of $25\,\mathrm{m} \times 20\,\mathrm{m}$ to be explored, as indicated by the red polygon in Figure 9.12.

The exploration experiment ran for 35 min, due to a limited availability of fully charged batteries, we, however, were not able to fully explore the target area and in due time manually set waypoints to drive the rovers back close to their start positions. The two rovers traveled a combined distance of 394 m and mapped an area of approx. $650\,\mathrm{m}^2$, including parts outside of the exploration target area that were traversed to avoid obstacles like large stones. In Figure 9.13, we present the final map in its point cloud and probabilistic voxel-grid representations as well as the multi-robot SLAM

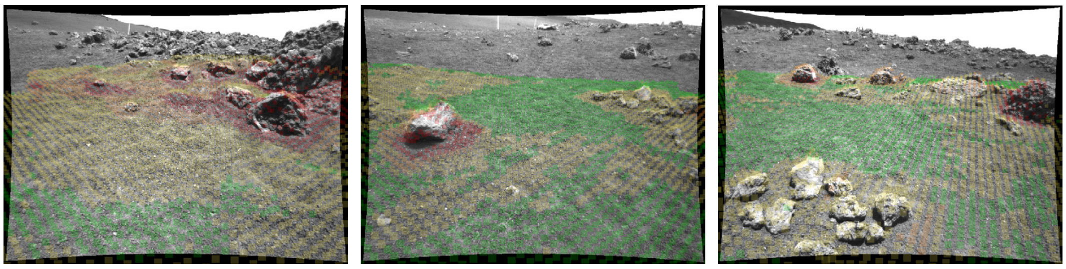


Figure 9.11: *Images taken by the navigation camera of one LRU during the multi-robot exploration experiment. The camera image is superimposed with the rover's terrain classification map (green to red: traversability from easy to hard obstacles).*

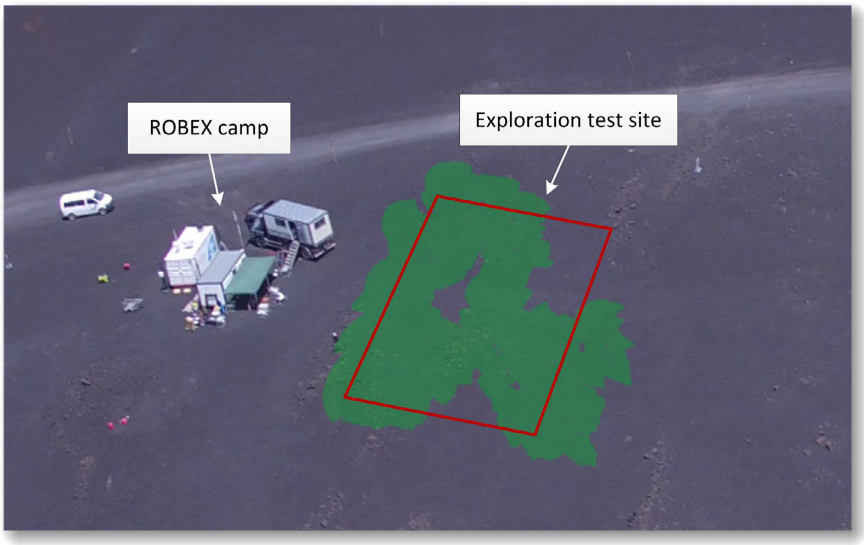


Figure 9.12: *Aerial photo of test site with manually overlaid approximate exploration target area (red) and area mapped during the experiment (green)*

graph, which is, similarly to our multi-robot experiments presented in Section 8.3, small and sparse with a total number of 163 nodes and 224 factors. As for this preliminary experiment we applied a frontier-based exploration algorithm, the rover trajectories exhibited little overlap, resulting in only a single loop closure from our map matching system. To approach this general issue, recent work from our group is concerned with active loop closing that makes a trade-off during the selection of goal locations between exploring new areas and re-visiting visited already mapped places for re-localization, as presented by Lehner et al. (2017).

We faced many additional challenges during the experiments at our volcanic test site at a height of 2645 m above sea level. In contrast to previous indoor and clouded-sky outdoor experiments, the AprilTag markers on our rovers used for mutual robot observations oftentimes could only be detected from one direction. In direct sunlight on Mt. Etna, they turned out to be too reflective, leading to severe overexposure in

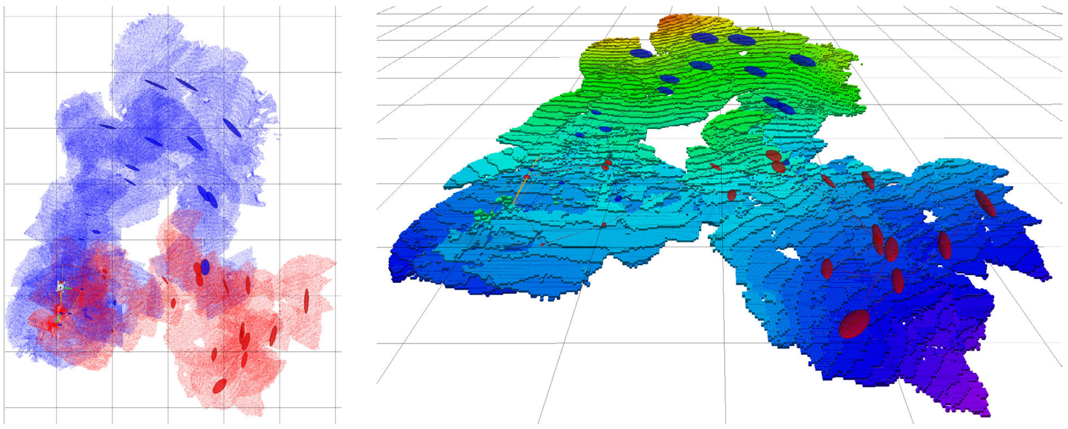


Figure 9.13: *Multi-robot 3D map created during autonomous exploration experiment on Mt. Etna. Left: point cloud-based map (resolution* 0.05 *m) created by our two rovers LRU1 (blue) and LRU2 (red). Right: height-colored voxel grid representation (resolution* 0.1 *m) of the same map, showing the slope of the terrain. The ellipsoids represent the estimated positional standard deviation of the rovers at their submap origins.*

the camera images that made the whole tags appear plain white and thus impossible to decode. For the experiment presented here, we manually ensured that the rovers could see each other's markers at least at the start and end of the experiment. For future work, we, however, plan to extend the robots' behaviors to actively look at each other based on their relative localization estimates to create further loop closure opportunities. While, in general, obstacles often provide unambiguous 3D structures suitable for keypoint-based feature matching, the restriction to these limited the capabilities of our map matcher in the Moon-analogue scenario. In some areas, the rough-terrain ground exhibited recognizable and thus potentially matchable geometry, it, however, was not classified as an obstacle and thus excluded from the feature matching process. Thus, we are currently looking into replacing this by an obstacle-independent selection of the map matcher keypoints. Furthermore, to obtain improved results, it might be beneficial to tune some map matcher parameters, like the aforementioned keypoint selection, to different environments. In this context, we started work on automatic parameter optimization, a topic that Cadena et al. (2016) recently identified as one of the important open challenges for the SLAM community. Despite these open challenges and lessons learned, on Mt. Etna we could successfully demonstrate the applicability of our stereo vision-based multi-robot localization and mapping methods to planetary exploration scenarios.[2]

[2]We give an impression of our multi-robot mapping and exploration experiment on Mt. Etna in the second half of the video available at `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21812`

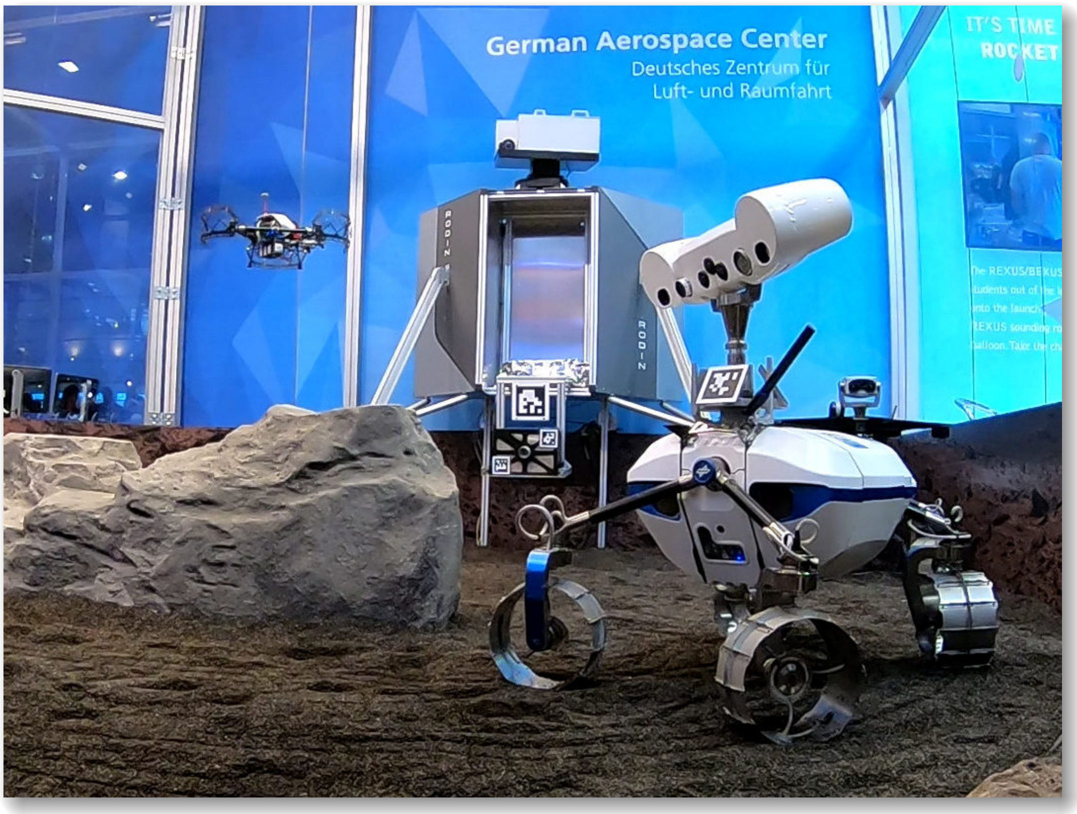


Figure 9.14: *Impression of our heterogeneous robot team with the multicopter Ardea (left) and rover LRU (right) during a collaborative multi-robot localization and mapping demonstration at IAC 2018*

9.4 IAC 2018: 6D SLAM for Heterogeneous Team of Ground and Aerial Robots

Our latest public demonstration took place at the International Astronautical Congress (IAC) 2018, the world's largest annual gathering of space professionals with more than 6500 participants from 82 different countries as well as 13000 visitors on its public day (IAF and ZARM, 2018). It was motivated and funded by the Helmholtz Future Project *Autonomous Robotic Networks to Help Modern Societies (ARCHES)*, the goal of which is the development of heterogeneous, autonomous, and interconnected robotic systems in a consortium of Helmholtz Centers (Wedler, 2018; Wedler et al., 2018b). At IAC, we set up an autonomous planetary exploration scenario in a Mars-like environment in a sandbox of approx. 50 m^2, as shown in Figure 9.14 and Figure 9.15. It featured a lander mockup as well as large artificial rocks as navigational and visual obstacles for our robots. Therein, we ran two different kinds

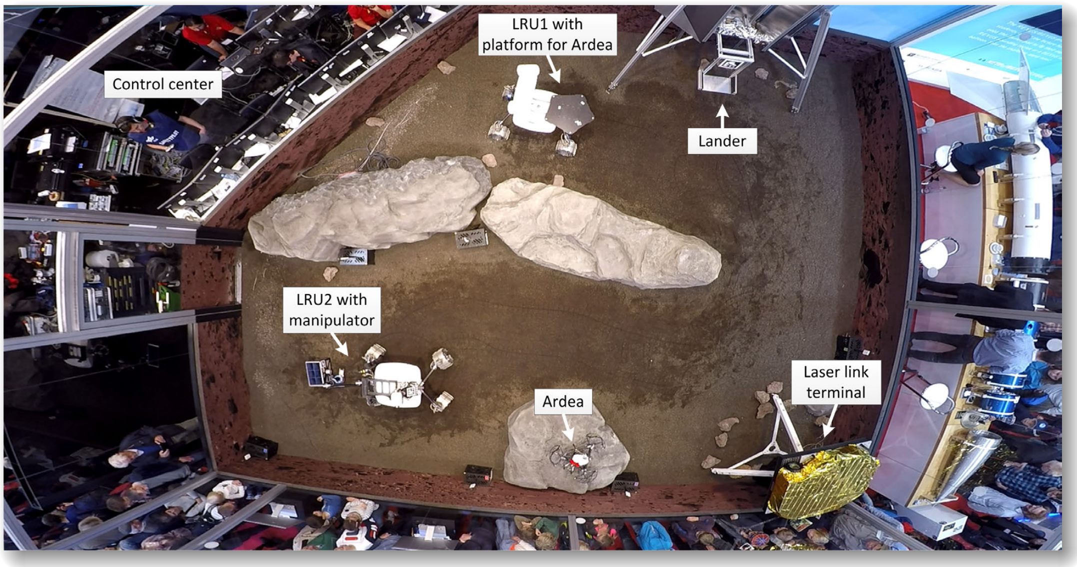


Figure 9.15: *Top-down overview of our Mars-like demonstration area at IAC 2018. It features sand, stones, large rocks, a lander mockup, and an optical communication link between a laser terminal and the lander. It is surrounded by our control center (top left corner) and visitors behind large glass panes on two sides (bottom and right).*

of demonstrations: sample-return with the rover LRU2, equipped with a robotic manipulator, and multi-robot mapping with a heterogeneous team of the multicopter Ardea and the rover LRU1, equipped with a platform for the takeoff of aerial robots. Similar to our SpaceBotCamp demonstration presented in Section 9.2, we used ground station computers only to start the robots, supervise the system and mission status, and visualize the results. All communication between the ground station and the robots was routed via two optical communication terminals, one mounted on the lander, the other one on a tripod at the opposing side of the sandbox area. They were connected via a laser link by our project partners from the DLR Institute of Communications and Navigation (KN) (Calvo et al., 2019) in order to demonstrate technology for future long-distance data connections, e. g., between Earth and the Moon or between an orbital station and a lander on a planetary surface. We ran demonstrations on all five days of the congress with an estimated total of more than 35 runs for each type of demonstration. During all these demonstrations, we showed the live mapping process as computed on board the robots to the visitors. Next, we will describe both demonstrations with focus on the localization and mapping aspects.

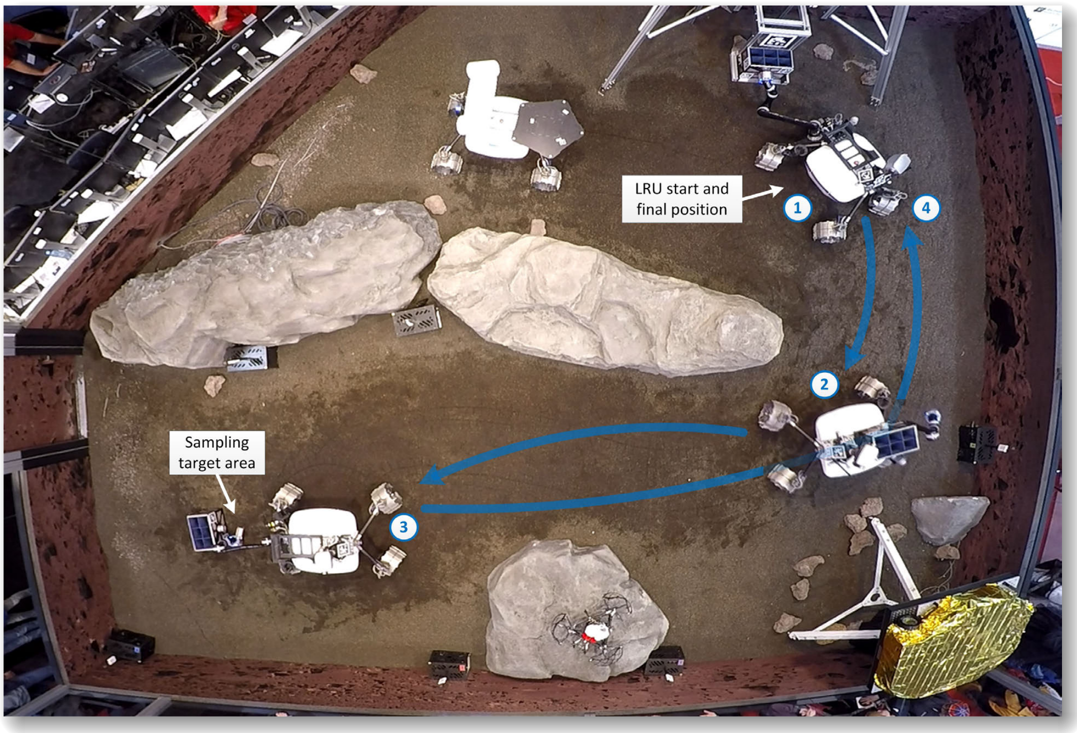


Figure 9.16: *Overview of the sample-return demonstration at IAC 2018: LRU starts in front of the lander and picks up an empty sample box with its manipulator (1), navigates to the target area while avoiding obstacles on its way (2), shovels a soil sample into the box (3), and returns it to lander (4).*

9.4.1 Obstacle Mapping for Sample-Return Navigation

The goal of the sample-return demonstration was to have LRU2 autonomously take a soil sample from a target area and transport it back to the lander for future analysis. We present a visual overview of the scenario and the operational sequence in Figure 9.16 and show photos of the key actions in Figure 9.17. LRU started in front of the lander mockup, where it used its manipulator arm to pick up a payload box from the lander and transported it to the target location at the opposite side of the sandbox. It then placed the box on the ground, attached a shovel to its robotic arm, and used this to take a soil sample into the payload box. Afterwards, it picked the box up, transported it back, and placed it onto its tray attached to the lander. As described in our application scenario in Section 1.1, autonomous sample-return sequences will constitute a core aspect of future planetary exploration missions in order to analyze materials from hard-to-access areas for scientific analyses or the prospecting of resources. Localization and mapping capabilities of the robots allow them to access and safely return from such target areas. During our demonstrations,

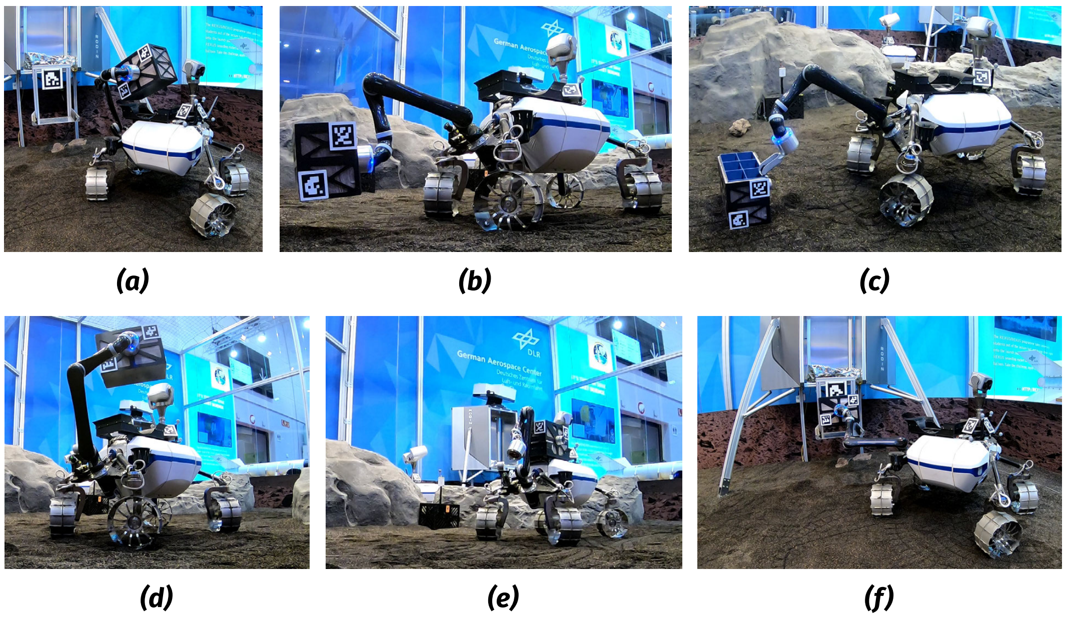


Figure 9.17: *Sequence of our IAC 2018 autonomous sample-return demonstration: LRU picked up a sample box from the lander (a), navigated autonomously to a target location, avoiding obstacles on its way, and placed the box there (b), attached a shovel to its arm and shoveled soil into the sample box (c), picked the box up (d), transported it back to the lander, reusing and updating the terrain classification map created during its forward run (e), and placed the sample box back onto the lander (f).*

LRU used the on-board localization and map estimates computed by our SLAM framework to navigate to the target location and back, avoiding any obstacles on the way. In Figure 9.18, we present a sequence of images showing the terrain classification maps used for navigation. Therein, the large rocks, the lander, and the laser terminal are marked in red as untraversable obstacles. Thus, LRU planned a path around them, driving through the areas with good traversability marked in green. We employed the same methods for navigation and obstacle avoidance for LRU in our multi-robot experiment described in the following section.

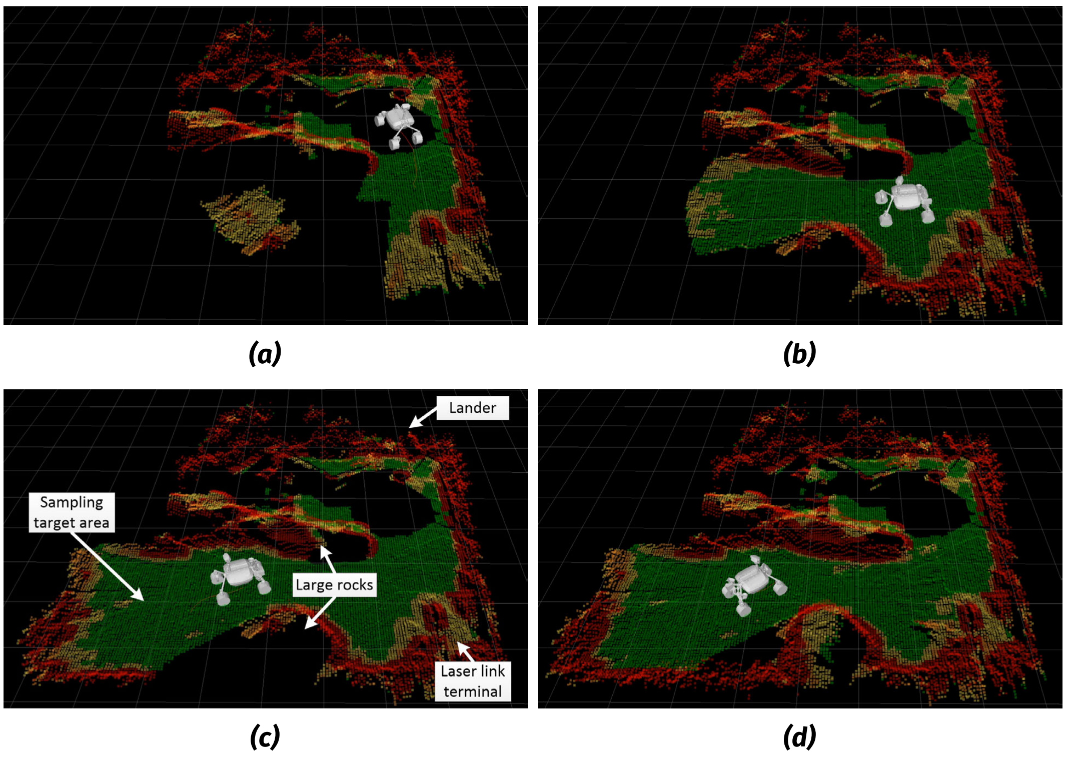


Figure 9.18: *2.5D terrain classification maps computed on board LRU during a sample-return demonstration at IAC 2018. LRU started to navigate towards the sampling area (a), drove around the large rocks that were classified as impassible obstacles (b), approached the target area (c), and returned back to the lander with its successfully taken soil sample (d).*

9.4.2 Multi-Robot 6D SLAM with Heterogeneous Team

The second type of demonstration featured a multi-robot mission to explore and map a target area with a heterogeneous team of our rover LRU and multicopter Ardea. In Figure 9.19, we present an overview of the actions of both robots and give further impressions of the demonstration in action in Figure 9.20. We showed a collaborative 6D localization and 3D mapping with a heterogeneous team of flying and driving robots, as envisioned in our planetary exploration scenario from Section 1.1. Our robots do not only have different locomotion capabilities and points of view on the environment, but also feature different stereo camera setups with narrow-angle lenses on LRU and a four-camera fisheye setup on Ardea. We ran our SLAM framework online and on board both robots to locally create partial 3D maps, exchange, connect, and optimize them as a collaboratively built joint map of the environment. We, however, did not yet match maps between Ardea and LRU. As we described in Section 6.1.1, our map matching algorithm relies for its keypoint selection on the

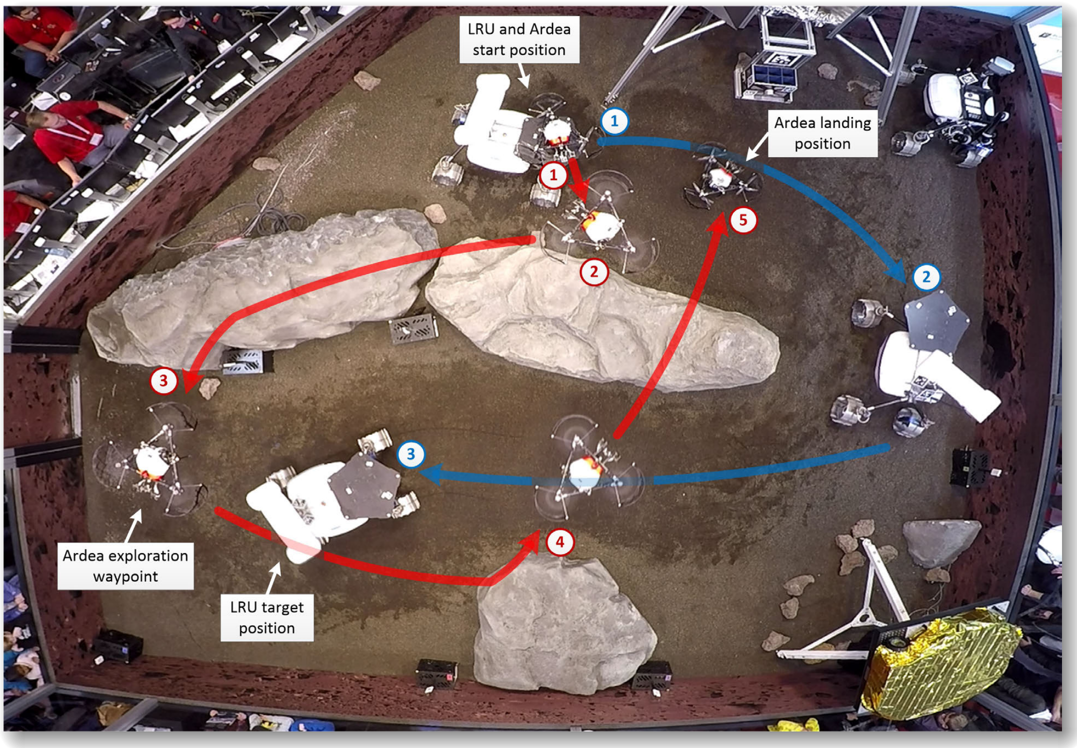


Figure 9.19: *Overview of our multi-robot collaborative mapping demonstration at IAC 2018 with LRU (blue) and Ardea (red): Ardea takes off from LRU's transport platform (1), at its first waypoint turns back and detects LRU to connect their pose and map estimates (2), then performs a* 360° *scan at its waypoint in the target area (3) and flies back (4) to its landing position next to the lander (5). In the meantime, LRU starts driving around the large rocks (2), navigating with autonomous obstacle avoidance to its waypoint in the target area (3).*

local obstacle maps designed for rovers. This type of terrain classification is not required for flying systems and thus not computed on Ardea. A map matching between aerial and ground-based robots thus would either need to estimate the same types of obstacles on all robots or to rely on a different keypoint selection algorithm. This leads to additional trade-offs regarding the reuse of already computed data, the accuracy, and the computational effort required for other keypoint selection methods and remains a topic for future work. Nonetheless, in our demonstration, multiple robot detections between Ardea and LRU provided a sufficient number of loop closure constraints for global optimization, leading to a visually consistent multi-robot 3D model of the environment.

In Figure 9.21, we visualized the final map in its colored point cloud and probabilistic voxel-grid map representations. Note that only Ardea has color cameras and we did not perform any white balancing or color calibration. The map shown in Figure 9.21(a) thus was colored by merging uncalibrated color information from

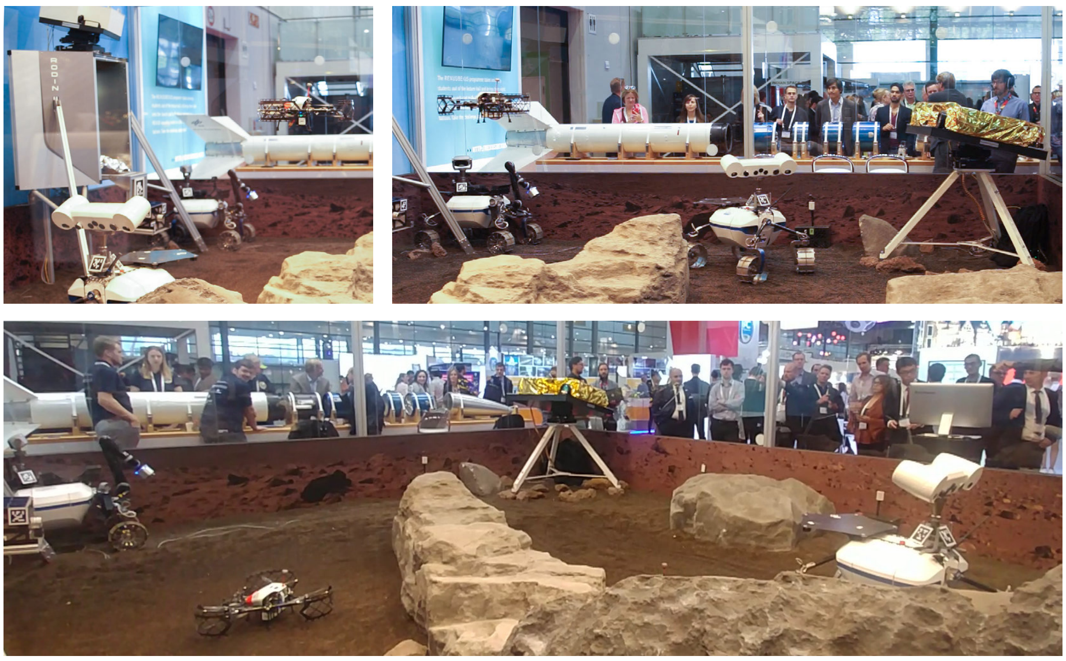
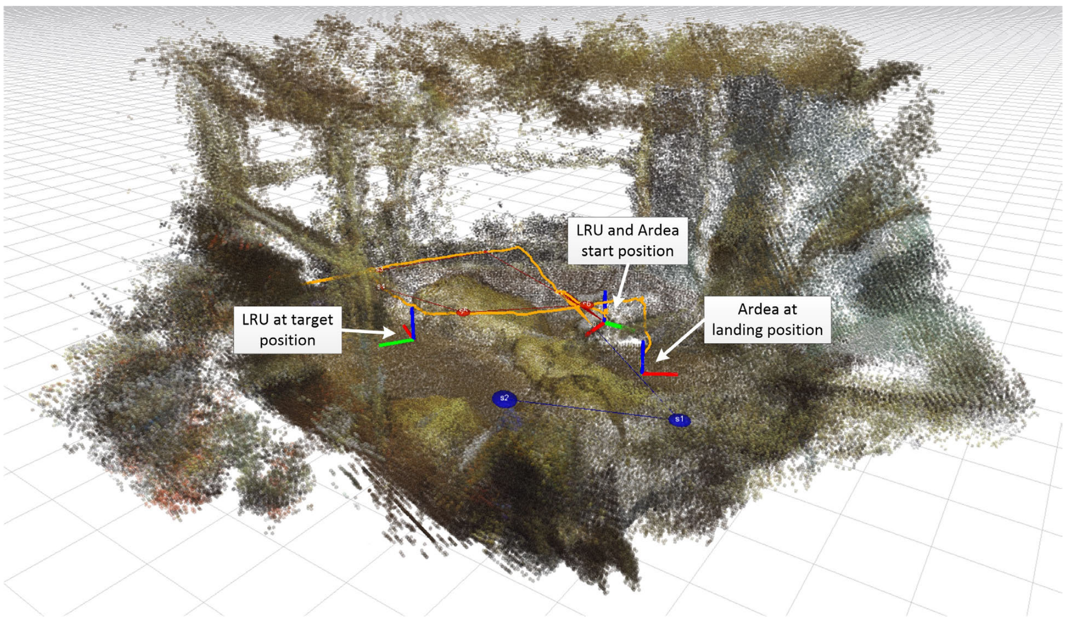


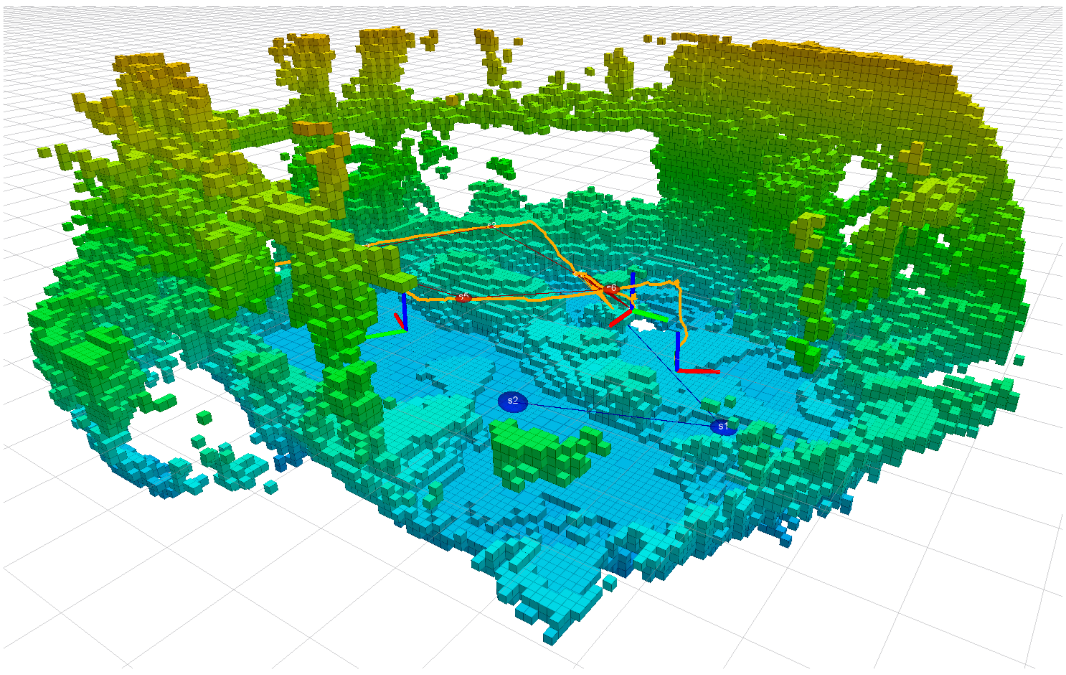
Figure 9.20: *Impressions of our cooperative multi-robot mapping demonstration at IAC 2018 with a heterogeneous team of rover and multi-copter: Ardea observing LRU from its first waypoint after takeoff from the rover's transport platform (top left), Ardea on its return flight while LRU is navigating to the location explored by Ardea (top right) and Ardea landed next to its start position while LRU reached its target destination (bottom).*

Ardea with greyscale data from LRU. In Figure 9.22 and Figure 9.23, we show sequences of images illustrating the mapping process with voxel-grid maps and matching robot-colored point cloud maps that were estimated on board Ardea and LRU respectively.

Both robots started their mission located in front of the lander, Ardea being carried on a platform attached to the back of LRU. First, LRU scanned the area with its pan-tilt camera head and discovered that the direct line-of-sight path to the target location was blocked by large rocks. Thus, the operators decided to dispatch the multicopter Ardea to take off from LRU, see Figure 9.22(a). Its task was to explore and map the area with its fisheye-lens stereo cameras by flying to and rotating 360° at several predefined waypoints. At a first waypoint close to LRU, it was able to observe the AprilTag markers attached to the camera mast of the rover. This allowed both robots to connect their coordinate frames and maps via robot detections as shown in Figure 9.22(b). While the very first detection had high uncertainty – see the large estimated positional uncertainty of Ardea's initial frame in the map of LRU visualized in Figure 9.23(b) – multiple additional robot detections at the same waypoint were subsequently added to the global estimation to visibly lower this uncertainty. Shortly afterwards, Ardea finished its first submap and transferred it to LRU, which integrated it into its global map estimate as depicted in Figure 9.23(c). Next, Ardea flew to its second waypoint, located above the target area, and performed a 360° scan in order to map it, see Figure 9.22(c). After observing the target area from the air, LRU was dispatched to navigate there, avoiding obstacles and mapping its path, see Figure 9.22(d)–(f) and Figure 9.23(d)–(f). In the meantime, Ardea returned to land near its starting position in front of the lander, which acts as the robots' home base in planetary exploration missions. The orange paths in Figure 9.22(d) and Figure 9.23(d) represent the full trajectories traveled and estimated by Ardea and LRU respectively. The final map shown in Figure 9.21 constitutes a dense 3D representation of the whole scenario. We intentionally excluded the ceiling from our voxel map visualizations for better visibility. The large holes in the walls are due to the transparent glass panes surrounding the sandbox that either were invisible to our stereo sensors or induced noise due to partial reflections.

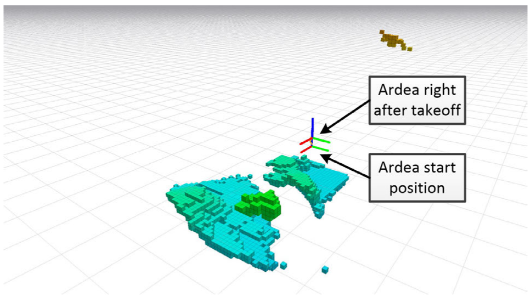


(a) *Colored point cloud representation (resolution* 0.05 *m, no color calibration)*

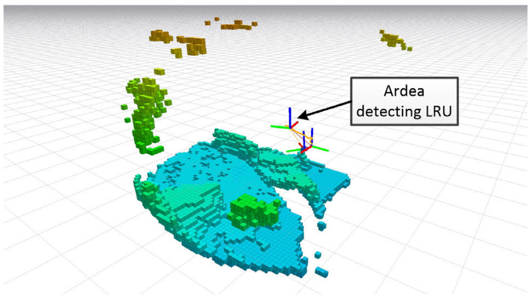


(b) *Height-colored probabilistic voxel-grid representation (resolution* 0.1 *m)*

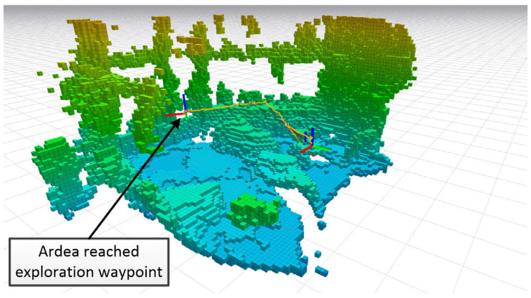
Figure 9.21: *Multi-robot 3D map and SLAM graph computed on board Ardea during one of our heterogeneous multi-robot mapping demonstrations at IAC 2018. The ellipsoids represent the estimated positional standard deviation of the robots' respective submap origins (blue: LRU, red: Ardea). The orange trajectory shows the flight path of Ardea.*



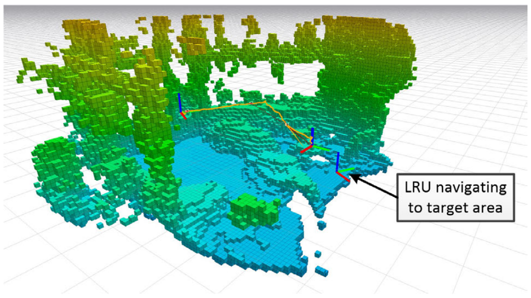
(a) *t* = 1 *min* 27*s*



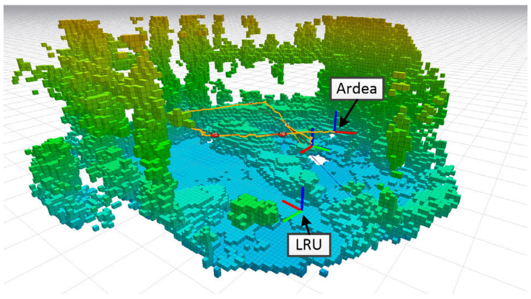
(b) *t* = 1 *min* 41*s*



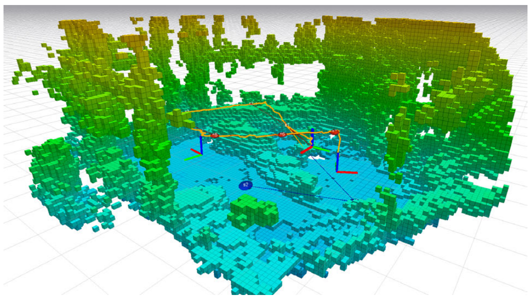
(c) *t* = 2 *min* 09*s*



(d) *t* = 2 *min* 33*s*

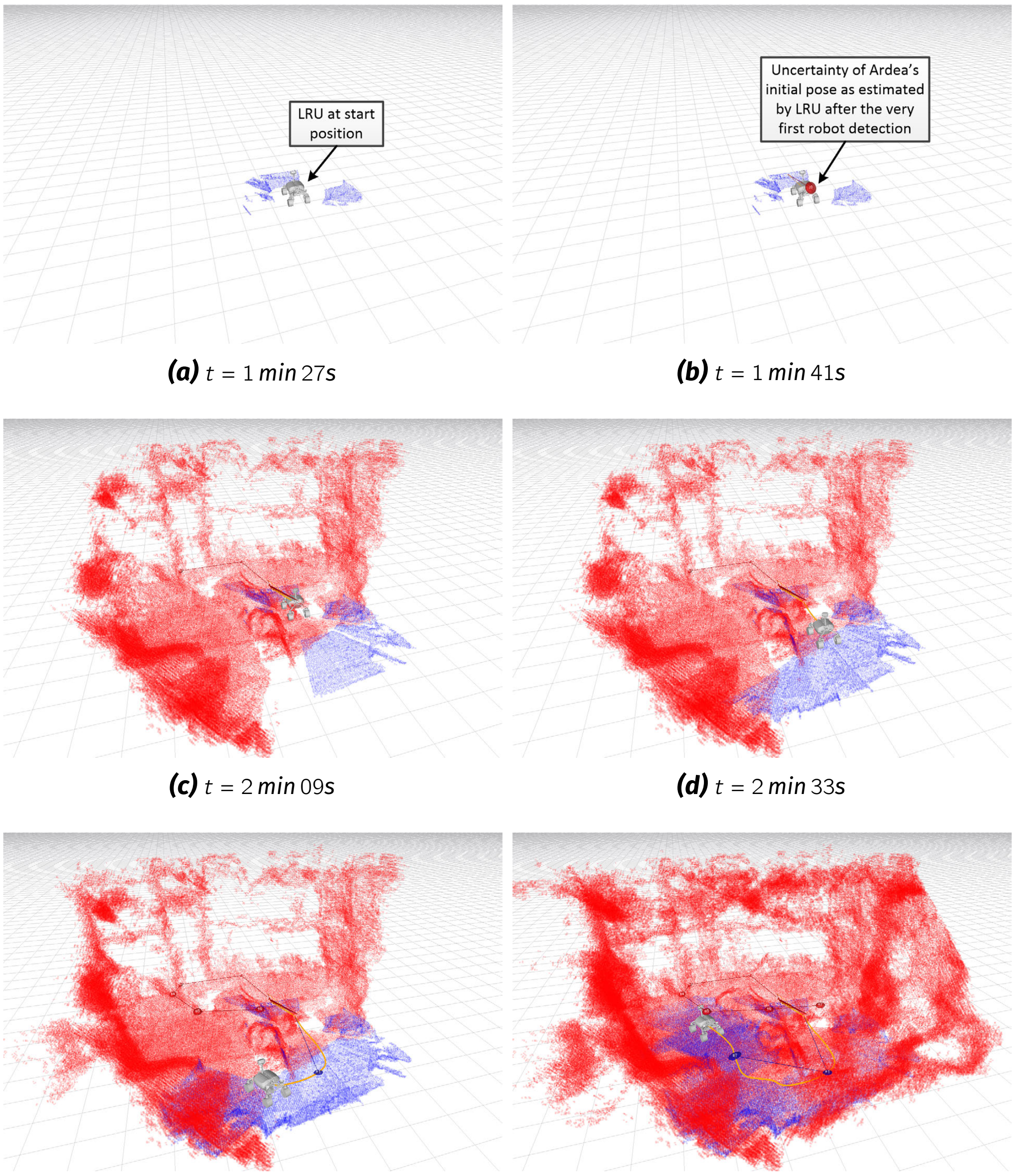


(e) *t* = 3 *min* 00*s*



(f) *t* = 3 *min* 45*s*

Figure 9.22: *Image sequence showing the multi-robot mapping by LRU and Ardea during one of our heterogeneous multi-robot experiments at IAC 2018 as computed on board Ardea (height-colored probabilistic voxel-grid map)*


LRU at start position
Uncertainty of Ardea's initial pose as estimated by LRU after the very first robot detection

(a) $t = 1$ *min* 27*s* ***(b)*** $t = 1$ *min* 41*s*

(c) $t = 2$ *min* 09*s* ***(d)*** $t = 2$ *min* 33*s*

(e) $t = 3$ *min* 00*s* ***(f)*** $t = 3$ *min* 45*s*

Figure 9.23: *Image sequence showing the multi-robot mapping by LRU and Ardea during one of our heterogeneous multi-robot experiments at IAC 2018 as computed on board LRU (robot-colored point cloud map, blue: LRU, red: Ardea)*

9.5 Summary and Discussion

We demonstrated the robustness of our SLAM system by making it an integral part of our approach to complex scenarios with autonomous space exploration robots, which we showed at several occasions in front of public audiences. For successful demonstrations, our localization and mapping modules had to interact with many other software components such as local path planning as well as global multi-robot exploration planning.

We could fulfill all of our objectives defined in Section 1.2 by providing localization and mapping capabilities central to autonomous planetary exploration robots. In all of our four major demonstrations, the robots relied on their on-board local pose and map estimates for local navigation and obstacle avoidance, and combined local submaps to globally optimized models of the environment. During the ROBEX experiments on Mt. Etna as well as the ARCHES demonstrations at IAC, the robots collaboratively built joint 3D maps and localized all team members therein by estimating their 6D poses. These pose and map estimates were used to coordinate a multi-robot autonomous exploration algorithm that decided on target locations for the individual robots at the Moon-analogue test site on top of Mt. Etna. At IAC, our heterogeneous team of flying and driving robots, featuring different stereo sensor setups as well as different points of view, collaboratively created a 3D model of their Mars-like environment.

Furthermore, we showed that our methods can deal with several of the major challenges of planetary exploration scenarios. In all of our experiments and demonstrations, the robots relied solely on a space-qualifiable sensor setup, namely inertial sensors and stereo camera systems, complemented by wheel odometry estimates on some of our rover systems. All four demonstrations featured autonomous robot action. In particular, during the SpaceBotCamp challenge, our robot had to navigate in a fully autonomous mission with limited and heavily delayed communication. Thus, it was crucial for all computation to be performed online and on board the robot. Each of the demonstration scenarios was designed to have visual similarities to the surface of the Moon or Mars – an important aspect regarding our vision-based navigation methods. Our experiments on Mt. Etna took place in a volcanic Moon-analogue rough-terrain environment, thus featuring a test site here on Earth that is as close to real space missions as possible.

At the four events presented here, we applied our stereo vision-based SLAM methods to central tasks defined in our vision of a future planetary exploration scenario sketched out in Section 1.1. For instance, we employed flying robots as scouts to map a previously unknown target area before sending rovers. The latter used their on-board pose and map estimates to navigate there in order to place seismic measurement instruments and conduct sample-return missions. A lander acted as a base station, facilitating the communication between a simulated planetary surface and a control center on Earth, as well as storing infrastructure elements and tools for the rover to pick up. The demonstration missions included the exploration and search for objects based on a very rough prior map, similar to a satellite image, as well as the fully autonomous exploration of previously unknown areas. The latter required a team of multiple robots to coordinate their actions and make decisions on their exploration targets online solely based on their latest collaborative on-board estimates of a map and the localization of all team members therein. Furthermore, similar to the cave exploration envisioned in our planetary scenario, we dispatched a heterogeneous team of flying and driving robots to create a dense 3D model of their environment.

Chapter ten

Conclusion

In this chapter, we conclude the thesis with a summary and a discussion of open challenges that represent topics for future work.

10.1 Summary and Conclusion

In this thesis, we presented a novel approach for collaborative Simultaneous Localization and Mapping (SLAM) with heterogeneous teams of mobile robots deployed to explore the surfaces of foreign moons or planets. The creation of a map as a model of the environment and the localization of all robotic agents therein constitute crucial capabilities for robots on planetary exploration missions, during which they need to operate in previously unknown and GNSS-denied environments. A 3D model of the environment can be used to raise the situational awareness of robot operators, support assisted teleoperation, and enable partial or full autonomy by laying the foundations for obstacle avoidance, path and exploration planning, mission planning, as well as for the coordination of multi-robot teams. Furthermore, maps of previously unknown areas on foreign moons and planets by themselves constitute desired scientific outcomes and can be used for semantic annotations as well as to geo-localize other scientific findings.

The objectives of this thesis are the development, improvement and combination of SLAM methods to provide planetary exploration robots with capabilities for fast local localization and mapping, collaborative online creation of a joint 3D map of the environment, and the localization of all the robots in that map. For this, we designed a SLAM system with the challenges of planetary exploration missions in mind, focusing on the processing of data from space-suitable lightweight proprioceptive and exteroceptive sensors such as stereo cameras, an IMU, as well as wheel odometry sensors. We consider limited communication bandwidth between robots and perform all computations online and on board the individual robots to support local autonomy.

In this thesis, we designed a multi-robot SLAM architecture that distributes the processing of high-frequency and high-bandwidth data in multi-robot teams. It is decentralized so that each robot has its own online global pose and map estimate available at all times, even in case of interrupted communication to any of the other robots. On each robot, we combine local and global estimation methods to get the best of both worlds: A local reference filter provides real-time local state estimates required for robot control and fast reactive behaviors. Online incremental graph optimization then computes global pose and map estimates as input to higher-level autonomy functions such as path planning, exploration, and multi-robot coordination. Our decoupling of local and global methods allows a distributed processing of high-frequency measurements in a multi-robot team and leads to a small SLAM graph for computationally efficient optimization steps. For this, we developed a novel graph topology to incorporate the state estimates from local reference filters according to their dependency assumptions and uncertainties. In our experiments, it leads to an improvement in localization accuracy of 15 % on average compared to using a standard pose graph topology.

In order to model the 3D geometry of the environment, we generate dense point cloud maps (resolution: 0.05 m) and probabilistic 3D voxel-grid maps (resolution: 0.1 m) from noisy stereo data. We distribute the computational load of dense online 3D mapping and reduce the required communication bandwidth between robots by aggregating high-bandwidth vision data into local maps of limited size and uncertainty. We create these so-called submaps by integrating the depth data along the trajectories estimated by the local reference filter that is running on each robot. We then optimize the relative transformations between these submaps during graph-based global estimation and use the results to compose a multi-robot map. Sharing aggregated map data in the form of submaps instead of raw image streams between robots allows us to reduce the communication bandwidth requirements from 38.75 MB/s to 58 KB/s. For obstacle avoidance, we compute a stereo error-adaptive local terrain classification directly on the depth images in order to cope with the quadratic growth of the distance-dependent error of stereo camera systems. In addition, taking the camera viewpoints into account allows the robot to detect and avoid obstacles that are only indirectly observable, such as steep cliffs leading downwards in front of it. We showed in our experiments the suitability of this obstacle classification for the identification of 3D geometry that is suitable to support place recognition in indoor, outdoor, and mixed environments.

We focus on two types of loop closure constraints for our SLAM system: intra- and inter-robot re-localization by matching the stereo vision-based submaps as well as marker-based visual detections of other robots. We developed methods for map matching to recognize previously visited locations in semi- and unstructured environments based on the local geometry represented in the submaps of one or multiple robots. The 3D structure of the environment is its only characteristic that can be estimated mostly invariant to varying light conditions as well as different sensors and viewpoints in heterogeneous multi-robot teams. Thus, we can match submaps created by multiple robots with different camera setups, as we have demonstrated in experiments with two rovers equipped with wide- and narrow-angle stereo setups. The decoupling of observable and unobservable states through partial frame switching in the local reference filter allows us to introduce a novel optimization: By enforcing all submaps to be gravity-aligned, we can reduce the dimensionality of the map matching from 6D to 4D. This leads to a 40% higher average number of loop closure constraints, as we demonstrated in our evaluation based on the data of 40 simulated and 13 real-world experiments.

In multi-robot teams, we use visual fiducial markers on some or all robots in order to detect each other in their camera images and to estimate their respective relative 6D transformations. Image noise, calibration and estimation errors can, however, lead to significant errors and ambiguities in the estimation results. Therefore, we developed a novel method to model the uncertainty of individual measurements with a combination of camera-dependent precomputed error values and online error propagation. These uncertainty estimates are used to integrate the resulting loop closure constraints into the SLAM graph. In our experimental evaluation, we demonstrated an improved localization accuracy stemming from our uncertainty estimation method for the 6D pose estimation of visual marker detections.

We integrated all components for local and global estimation, dense 3D mapping and loop closure generation into our modular mapping architecture. It allows an easy adaption to include resource-limited systems as part of future heterogeneous multi-robot teams: They can, for example, omit the computationally expensive dense 3D mapping and still benefit from joint global pose estimation. We demonstrated the robustness of our methods by integrating them on five different ground-based and aerial mobile robots that were deployed in 31 real-world experiments for quantitative evaluations and many more test and demonstration runs in semi- and unstructured indoor and outdoor settings.

We evaluated our full collaborative 6D SLAM framework in seven real-world multi-robot experiments featuring different camera setups and robots in areas of up to $3000\,\mathrm{m}^2$ (bounding box). It was able to create visually accurate multi-robot global maps from the robots' noisy stereo data and estimated their trajectories with average 3D translational errors below 0.5 m w. r. t. partially available ground truth, i. e., errors below 0.5 % of the robots' respective total trajectories. Our separation of local and global methods allowed us to construct small multi-robot SLAM graphs with less than 180 nodes and 250 factors in all of our experiments, leading to fast global optimization steps.

We validated our SLAM methods through several different demonstrations at four public events in Moon and Mars-like environments that covered many of the challenges posed by planetary exploration scenarios. In all demonstrations, our robots used their online pose and map estimates for on-board autonomous navigation in unstructured rough terrain. Our SLAM framework contributed to our success at the SpaceBotCamp challenge in 2015, where we employed it as the basis for the LRU rover to solve all tasks of a complex exploration and assembly mission autonomously. In this scenario, the communication links to the rover had been limited and delayed in order to simulate the challenges of Moon exploration. During the test campaign of the Robotic Exploration of Extreme Environments (ROBEX) project in 2017, we validated our vision-based system in single- and multi-robot demonstrations in the challenging unstructured environment of a Moon-analogue site on the volcano Mt. Etna. These include the successful deployment of a team of two LRU rovers for collaborative autonomous exploration of a rough-terrain target area guided by their global joint 6D localization and 3D map estimates. At the International Astronautical Congress (IAC) 2018, we deployed a heterogeneous multi-robot team with aerial and ground-based robots for the collaborative mapping of a Mars-like environment and publicly presented this demonstration more than 35 times during the five days of the exhibition.

In this thesis, we developed and demonstrated concepts and methods for multi-robot localization and mapping as an important step towards future space missions featuring heterogeneous teams of autonomous robots deployed to explore the surfaces of moons and foreign planets. Many of these techniques can be adapted and transferred to further, terrestrial application domains such as, for example, search and rescue missions featuring GNSS-denied areas in semi- and unstructured environments.

10.2 Outlook on Future Work

There are several remaining challenges that lie beyond this thesis and constitute topics for future improvements and extensions. Our current submap matching algorithm bases its keypoint selection on the obstacle classification designed for rover systems. While this is computationally efficient and was shown to give good results, it limits the areas that can be matched to those containing sufficiently many and well-distributed obstacles. Furthermore, the classification is specific to a particular type of robot as it is based on its maneuverability constraints. Thus, we propose for future work to adapt the keypoint selection to include traversable but locally discriminative parts of rough terrain in order to obtain matches in areas with no or few obstacles and to allow matches in heterogeneous robotic teams consisting of aerial and ground-based robots.

In our SLAM framework, marker-based landmark or robot detections provide initial estimates between the coordinate frames of multiple robots, which are then improved via additional loop closure constraints from inter-robot map matches. Our map matching method requires this initial guess to filter outliers and limit its computational effort. The purely geometry-based re-localization in an existing map created by other robots or in a multi-session mapping task thus remains a topic for future work that could extend the possible application scenarios of our framework. Another way to approach this challenge would be the integration of additional sensor modalities, such as distance estimates from radio links between mobile robots and static anchors, or between multiple moving robots.

We demonstrated our mapping system in experiments running over half an hour and in areas of up to $3000\,\mathrm{m}^2$ (bounding box). Persistent and multi-session mapping as well as scaling to long-term applications and larger-scale maps, however, remain open challenges. They can, for example, be approached by an intelligent and consistent merging of submaps when no further improvement of their relative transformations is to be expected. On the SLAM graph level, this also requires to marginalize out and replace nodes relating to the merged submaps. Merging or removing old data is necessary to keep the computational and memory requirements in a limited workspace independent of the traveled distance and runtime of the system. Such changes of previously acquired parts of the environment model, however, do not only affect the map itself but also need to be propagated within a multi-robot system while avoiding inconsistencies between the estimates of different robots, in particular when operating under temporarily unavailable communication links.

Almost all sufficiently complex localization and mapping frameworks introduce a set of parameters to be tuned, either regarding the robot systems and sensors, the environment, other components, or all of them together. An automation of this process has recently been identified by Cadena et al. (2016) as an important open challenge for the SLAM community. We agree with the importance of this challenge and thus started to look into an automatic parameter optimization for our map matcher component, which, however, lies beyond the scope of this thesis.

A further important topic for future investigation is the addition of semantic annotations to the maps created by the robots. These can, for example, geolocate scientific measurements, describe known and unknown objects in the environment, or classify different areas or types of terrain. In addition to raising the situational awareness for robot operators and providing knowledge to planetary scientists, semantic information can be used by the robots themselves to plan and distribute tasks in heterogeneous teams according to their individual capabilities. Furthermore, many types of geolocated semantic annotations can be used as additional information to support robot re-localization, either directly at the graph level or to guide map matching algorithms.

Most algorithms presented in this thesis have been developed and implemented to run on terrestrial computation hardware. While this gives more design freedom and allows faster iterations during research and development, real applications to planetary exploration will require their adaption to the limitations of the space-qualified hardware components available in future missions.

Appendix A

Prior Publications

This thesis is in parts based work that we have published in international journals and conferences, which we refer to in the respective sections. For the sake of completeness, see below for a complete list of my prior publications.

Journal Articles

- M. J. Schuster, K. Schmid, C. Brand, and M. Beetz. Distributed stereo vision-based 6d localization and mapping for multi-robot teams. *Journal of Field Robotics (JFR)*, 2018. doi: 10.1002/rob.21812. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21812`
- S. G. Brunner, P. Lehner, M. J. Schuster, S. D. Riedel, R. Belder, D. Leidner, A. Wedler, M. Beetz, and F. Stulp. Design, Execution and Post-Mortem Analysis of Prolonged Autonomous Robot Operations. *IEEE Robotics and Automation Letters*, 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2794580
- M. J. Schuster, S. G. Brunner, K. Bussmann, S. Büttner, A. Dömel, M. Hellerer, H. Lehner, P. Lehner, O. Porges, J. Reill, S. Riedel, M. Vayugundla, B. Vodermayer, T. Bodenmüller, C. Brand, W. Friedl, I. Grixa, H. Hirschmüller, M. Kaßecker, Z.-C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler. Towards Autonomous Planetary Exploration: The Lightweight Rover Unit (LRU), its Success in the

SpaceBotCamp Challenge, and Beyond. *Journal of Intelligent & Robotic Systems (JINT)*, Nov. 2017. ISSN 1573-0409. doi: 10.1007/s10846-017-0680-9. URL `https://doi.org/10.1007/s10846-017-0680-9`

Conference Papers

- M. G. Müller, F. Steidle, M. J. Schuster, P. Lutz, M. Maier, S. Stoneman, T. Tomić, and W. Stürzl. Robust Visual-Inertial State Estimation with Multiple Odometries and Efficient Mapping on an MAV with Ultra-Wide FOV Stereo Vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. doi: 10.1109/IROS.2018.8594117
- A. Wedler, M. Wilde, J. Reill, M. J. Schuster, M. Vayugundla, S. G. Brunner, K. Bussmann, A. Dömel, M. Drauschke, H. Gmeiner, H. Lehner, P. Lehner, M. G. Müller, W. Stürzl, R. Triebel, B. Vodermayer, A. Börner, R. Krenn, A. Dammann, U.-C. Fiebig, E. Staudinger, F. Wenzköfer, S. Flögel, S. Sommer, T. Asfour, M. Flad, S. Hohmann, M. Brandauer, and A. O. Albu-Schäffer. From single autonomous robots toward cooperative robotic interactions for future planetary exploration missions. In *69th International Astronautical Congress (IAC)*, Bremen, Germany, Oct. 2018b
- M. Vayugundla, F. Steidle, M. Smisek, M. J. Schuster, K. Bussmann, and A. Wedler. Datasets of Long Range Navigation Experiments in a Moon Analogue Environment on Mount Etna. In *International Symposium on Robotics (ISR)*, 2018
- M. Panzirsch, H. Singh, M. Stelzer, M. J. Schuster, C. Ott, and M. Ferre. Extended Predictive Model-Mediated Teleoperation of Mobile Robots through Multilateral Control. In *IEEE Intelligent Vehicles Symposium (IV)*, 2018. doi: 10.1109/IVS.2018.8500578
- A. Wedler, M. Vayugundla, H. Lehner, P. Lehner, M. J. Schuster, S. G. Brunner, W. Stürzl, A. Dömel, H. Gmeiner, B. Vodermayer, B. Rebele, I. L. Grixa, K. Bussmann, J. Reill, B. Willberg, A. Maier, P. Meusel, F. Steidle, M. Smisek, M. Hellerer, M. Knapmeyer, F. Sohl, A. Heffels, L. Witte, C. Lange, R. Rosta, N. Toth, S. Voelk, A. Kimpe, P. Kyr, and M. Wilde. First Results of the ROBEX Analogue Mission Campaign: Robotic Deployment of Seismic Networks for Future Lunar Missions. In *International Astronautical Congress (IAC)*, 2017

- H. Lehner, M. J. Schuster, T. Bodenmüller, and S. Kriegel. Exploration with Active Loop Closing: A Trade-off between Exploration Efficiency and Map Quality. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017. doi: 10.1109/IROS.2017.8206521

- M. J. Schuster, C. Brand, S. G. Brunner, P. Lehner, J. Reill, S. Riedel, T. Bodenmüller, K. Bussmann, S. Büttner, A. Dömel, W. Friedl, I. Grixa, M. Hellerer, H. Hirschmüller, M. Kassecker, Z.-C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler. The LRU Rover for Autonomous Planetary Exploration and its Success in the SpaceBotCamp Challenge. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Bragança, Portugal, 2016. doi: 10.1109/ICARSC.2016.62

- M. Hellerer, M. J. Schuster, and R. Lichtenheldt. Software-in-the-loop Simulation of a Planetary Rover. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2016

- M. J. Schuster, C. Brand, H. Hirschmüller, M. Suppa, and M. Beetz. Multi-Robot 6D Graph SLAM Connecting Decoupled Local Reference Filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015. doi: 10.1109/IROS.2015.7354094

- C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa. Submap Matching for Stereo-Vision Based Indoor/Outdoor SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015. doi: 10.1109/IROS.2015.7354182

- A. Wedler, B. Rebele, J. Reill, M. Suppa, H. Hirschmüller, C. Brand, M. Schuster, B. Vodermayer, H. Gmeiner, A. Maier, B. Willberg, K. Bussmann, F. Wappler, M. Hellerer, and R. Lichtenheldt. LRU - Lightweight Rover Unit. In *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Noordwijk, Netherlands, 2015

- C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa. Stereo-Vision Based Obstacle Mapping for Indoor / Outdoor SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, Illinois, USA, 2014. doi: 10.1109/IROS.2014.6942805 *(The first two authors assert equal contribution and joint first authorship.)*

- A. Wedler, A. Maier, J. Reill, C. Brand, H. Hirschmüller, M. J. Schuster, M. Suppa, A. Beyer, N. Y. Lii, M. Maier, H.-J. Sedlmayr, and R. Haarmann. Pan/Tilt-Unit as a Perception Module for Extra-Terrestrial Vehicle and Landing Systems. In

Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA), Noordwijk, Netherlands, 2013

- M. J. Schuster, D. Jain, M. Tenorth, and M. Beetz. Learning Organizational Principles in Human Environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi: 10.1109/ICRA.2012.6224553

- M. J. Schuster, J. Okerman, H. Nguyen, J. M. Rehg, and C. C. Kemp. Perceiving Clutter and Surfaces for Object Placement in Indoor Environments. In *10th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2010b. doi: 10.1109/ICHR.2010.5686328

- M. Schuster, R. Bormann, D. Steidl, S. Reynolds-Haertle, and M. Stilman. Stable Stacking for the Distributor's Pallet Packing Problem. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010a. doi: 10.1109/IROS.2010.5650217

Other Publications

- A. Wedler, K. Bussmann, A. Dömel, M. Drauschke, H. Gmeiner, I. L. Grixa, H. Lehner, M. Vayugundla, M. G. Müller, J. Reill, M. Schuster, W. Stürzl, B. Vodermayer, P. Lehner, S. Brunner, and M. Görner. From the ROBEX Experiment Toward the Robotic Deployment and Maintenance of Scientific Infrastructure for Future Planetary Exploration Missions. In *42nd COSPAR Scientific Assembly*, Pasadena, California, USA, July 2018a *(Extended abstract and presentation)*

- M. J. Schuster. Towards Autonomous Planetary Exploration: Collaborative Multi-Robot Localization and Mapping in GPS-denied Environments. In *International Technical Symposium on Navigation and Timing (ITSNT)*, Toulouse, France, Nov. 2017 *(Presentation)*

- D. Pangercic, K. Mathe, Z.-C. Marton, L. C. Goron, M.-S. Opris, M. J. Schuster, M. Tenorth, D. Jain, T. Ruehr, and M. Beetz. A Robot that Shops for and Stores Groceries. AAAI Video Competition (AIVC 2011), August 7–11 2011. URL `http://youtu.be/x0Ybod_6ADA` *(Video)*

Acronyms

ARCHES Autonomous Robotic Networks to Help Modern Societies 6, 196, 207

CAN Controller Area Network 128

CPU Central Processing Unit 101, 175, 178

CSHOT Color-SHOT 96, 97, 101

DGPS Differential GPS 133

DLR German Aerospace Center 4–6, 125, 128, 197

DRC DARPA Robotics Challenges 186

EKF Extended Kalman Filter 35–39, 45, 52–55, 66, 67, 139

ESA European Space Agency 3

EtherCAT Ethernet for Control Automation Technology 128

FPGA Field Programmable Gate Array 23, 44, 54, 126, 128, 130

GNSS Global Navigation Satellite System 1, 19, 26, 32, 60, 133, 186, 209, 212

GPS Global Positioning System 66

GPU Graphics Processing Unit 101

GSM Global System for Mobile Communications 175

IAC International Astronautical Congress 12, 30, 128, 184, 196–202, 204–207, 212

IAI Institute for Artificial Intelligence 5

ICP Iterative Closest Point 95, 99–102, 147, 179

ILA Innovation and Leadership in Aerospace 12, 184–186

IMU Inertial Measurement Unit 8, 23, 44, 45, 52–55, 63, 123, 126, 131, 139, 168, 175, 183, 210

ISRU In-Situ Resource Utilization 3

JAXA Japan Aerospace Exploration Agency 3

KF Kalman Filter 35, 36, 38, 53, 175

KL Kullback-Leibler 88

KN Institute of Communications and Navigation 128, 197

LIDAR Light Detection and Ranging 23, 48, 102, 132, 136, 137

LRU Lightweight Rover Unit 3, 7, 21, 82, 84, 95, 123–133, 144, 146, 147, 152, 153, 156, 160, 165, 174, 175, 182, 183, 185, 187–192, 194, 196, 198–206, 212

MAP Maximum a Posteriori Probability 28, 35, 38, 67

MAV Micro Aerial Vehicle 36, 44, 47, 49, 81, 125, 183

MER Mars Exploration Rover 103

MSL Mars Science Laboratory 103

NASA National Aeronautics and Space Administration 3

OBC On-Board Computer 126, 128

P*n*P Perspective-*n*-Point 106, 111, 113, 114, 116–118, 121, 122

P3AT Pioneer 3-AT 123, 124, 137–139, 153, 156, 160–163, 165

P3DX Pioneer 3-DX 108, 123, 124, 160–163, 165

PCI-E PCI Express 128

PCL Point Cloud Library 78, 97–99

PF Particle Filter 37

RAFCON RMC advanced flow control 130

RANSAC Random Sample Consensus 93, 98, 99, 147, 179

RBPF Rao-Blackwellized Particle Filter 35, 37, 38, 85, 135–140, 142, 156, 157, 179–181, 185, 186

RGBD (Red, Green, Blue, Depth) 48, 65, 66

RM Institute of Robotics and Mechatronics 4, 5, 125, 130

RMC Robotics and Mechatronics Center 6

RMPM RM Package Management 130

RMSE Root-Mean-Square Error 73, 99

ROBEX Robotic Exploration of Extreme Environments 3, 4, 6, 12, 133, 191, 192, 207, 212

ROS Robot Operating System 130

SDA Strap-Down Algorithm 53, 175

SGM Semi-Global Matching 54, 130

SHOT Signature of Histograms of Orientations 96

SiL Software-in-the-Loop 131

SLAM Simultaneous Localization and Mapping 1, 2, 6, 7, 11, 12, 16, 17, 19, 23–28, 32–40, 45–51, 54–60, 63, 65–67, 70, 85, 89, 90, 94, 95, 99–103, 110, 128, 132, 134–140, 142, 143, 146, 147, 150–160, 162–166, 168, 171, 175, 179–186, 188, 189, 192, 193, 199, 200, 204, 207, 209–214

SURF Speeded Up Robust Features 101

SVD Singular Value Decomposition 98

UAV Unmanned Aerial Vehicle 24

USB Universal Serial Bus 128

UWB Ultra-Wide Band 102

VINS Vision-Aided Inertial Navigation System 52–54, 58

Bibliography

Adept Technology Inc. Pioneer 3-AT. https://www.generationrobots.com/media/Pioneer3AT-P3AT-RevA-datasheet.pdf, 2011a. Accessed 9 May 2019.

Adept Technology Inc. Pioneer 3-DX. https://www.generationrobots.com/media/Pioneer3DX-P3DX-RevA.pdf, 2011b. Accessed 9 May 2019.

Advanced Realtime Tracking GmbH. ARTTRACK2. https://web.archive.org/web/20140606155028/http://www.ar-tracking.com/products/tracking-systems/arttrack-system/arttrack2/, 2014. Accessed 9 May 2019.

P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Robust Map Optimization using Dynamic Covariance Scaling. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013. doi: 10.1109/ICRA.2013.6630557.

A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard. Cooperative Robot Localization and Target Tracking based on Least Squares Minimization. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2013. doi: 10.1109/ICRA.2013.6631396.

L. A. Alexandre. 3D Descriptors for Object and Category Recognition: a Comparative Evaluation. In *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

Allied Vision Technologies GmbH. Guppy F-080. https://www.alliedvision.com/de/produkte/kameras/kameradetails/Guppy/F-080.html, 2019a. Accessed 9 May 2019.

Allied Vision Technologies GmbH. Guppy PRO F-125. https://www.alliedvision.com/en/products/cameras/detail/Guppy%20PRO/F-125.html, 2019b. Accessed 9 May 2019.

Analog Devices Inc. Data Sheet ADIS16407. https://www.analog.com/media/en/

technical-documentation/data-sheets/ADIS16407.pdf, 2011. Accessed 9 May 2019.

C. Avsar, W. Frese, T. Meschede, and K. Brieß. Developing a Planetary Rover with Students: Space Education at TU Berlin. *Journal of Automation Mobile Robotics and Intelligent Systems*, 8, 2014.

T. Bailey and H. Durrant-Whyte. Simultaneous Localization and Mapping (SLAM): Part II. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot. Consistency of the EKF-SLAM Algorithm. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

J. Bell, S. Squyres, K. Herkenhoff, J. Maki, H. Arneson, D. Brown, S. Collins, A. Dingizian, S. Elliot, E. Hagerott, A. Hayes, M. Johnson, J. Johnson, J. Josep, K. Kinch, M. Lemmon, R. Morris, L. Scherr, M. Schwochert, M. Shepard, G. Smith, J. Sohl-Dickstein, R. Sullivan, W. Sullivan, and M. Wadsworth. Mars Exploration Rover Athena Panoramic Camera (Pancam) Investigation. *Journal of Geophysical Research*, 108(E12), 2003. doi: 10.1029/2003JE002070.

C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa. Stereo-Vision Based Obstacle Mapping for Indoor / Outdoor SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, Illinois, USA, 2014. doi: 10.1109/IROS.2014.6942805.

C. Brand, M. J. Schuster, H. Hirschmüller, and M. Suppa. Submap Matching for Stereo-Vision Based Indoor/Outdoor SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015. doi: 10.1109/IROS.2015.7354182.

D. Brown, L. Cantillo, G. Webster, and A. Caputo. Mars Ice Deposit Holds as Much Water as Lake Superior. https://www.jpl.nasa.gov/news/news.php?release=2016-299, 2016. Accessed 28 March 2017.

S. G. Brunner, F. Steinmetz, R. Belder, and A. Dömel. RAFCON: A Graphical Tool for Engineering Complex, Robotic Tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016. doi: 10.1109/IROS.2016.7759506.

S. G. Brunner, P. Lehner, M. J. Schuster, S. D. Riedel, R. Belder, D. Leidner, A. Wedler, M. Beetz, and F. Stulp. Design, Execution and Post-Mortem Analysis of Prolonged

Autonomous Robot Operations. *IEEE Robotics and Automation Letters*, 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2794580.

K. Bussmann, L. Meyer, F. Steidle, and A. Wedler. Slip Modeling and Estimation for a Planetary Exploration Rover: Experimental Results from Mt. Etna. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018. doi: 10.1109/IROS.2018.8594294.

C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics*, 32(6):1309–1332, Dec 2016. ISSN 1552-3098. doi: 10.1109/TRO.2016.2624754.

R. M. Calvo, J. Poliak, J. Surof, A. Reeves, M. Richerzhagen, H. F. Kelemu, R. Barrios, C. Carrizo, R. Wolf, F. Rein, A. Dochhan, S. Karen, and W. Luetke. Optical technologies for very high throughput satellite communications. In *Free-Space Laser Communications XXXI*, 2019. doi: 10.1117/12.2513819.

J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, Nov 1986. ISSN 0162-8828. doi: 10.1109/TPAMI.1986.4767851.

L. Carlone, M. K. Ng, B. Bona, and M. Indri. Rao-Blackwellized Particle Filters Multi Robot SLAM with Unknown Initial Correspondences and Limited Communication. *IEEE International Conference on Robotics and Automation (ICRA)*, May 2010. doi: 10.1109/ROBOT.2010.5509307.

S. Carpin. Merging Maps via Hough transform. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2008. doi: 10.1109/IROS.2008.4650616.

A. Coates, R. Jaumann, A. Griffiths, C. Leff, N. Schmitz, J.-L. Josset, G. Paar, M. Gunn, E. Hauber, C. R. Cousins, R. Cross, P. Grindrod, J. Bridges, M. Balme, S. Gupta, I. Crawford, P. Irwin, R. Stabbins, D. Tirsch, J. Vago, T. Theodorou, M. Caballo-Perucha, and G. Osinski. The PanCam Instrument for the ExoMars Rover. *Astrobiology*, 17(6-7):511–541, 2017. doi: 10.1089/ast.2016.1548.

A. Cunningham, V. Indelman, and F. Dellaert. DDF-SAM 2.0: Consistent Distributed Smoothing and Mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2013. ISBN 978-1-4673-5643-5. doi: 10.1109/ICRA.2013.6631323.

F. Dellaert. GTSAM 3.2.1. `https://collab.cc.gatech.edu/borg/gtsam`, 2015. Accessed 4 January 2017.

F. Dellaert and M. Kaess. Factor Graphs for Robot Perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139, 2017. doi: 10.1561/2300000043.

J. Dong, E. Nelson, V. Indelman, N. Michael, and F. Dellaert. Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015. doi: 10.1109/ICRA.2015.7140012.

H. Durrant-Whyte and T. Bailey. Simultaneous Localization and Mapping : Part I. *IEEE Robotics & Automation Magazine*, 13(2):99–110, 2006. doi: 10.1109/MRA.2006.1638022.

T. Emter and J. Petereit. 3D SLAM With Scan Matching and Factor Graph Optimization. In *International Symposium on Robotics (ISR)*, 2018.

F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An Evaluation of the RGB-D SLAM system. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2012. doi: 10.1109/ICRA.2012.6225199.

C. Forster, M. Pizzoli, and D. Scaramuzza. Air-Ground Localization and Map Augmentation Using Monocular Dense Reconstruction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

G. Grisetti, C. Stachniss, and W. Burgard. Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters. *IEEE Transactions on Robotics*, 23(1):34–46, Feb. 2007. ISSN 1552-3098. doi: 10.1109/TRO.2006.889486.

G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A Tutorial on Graph-Based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

J. P. Grotzinger, J. Crisp, A. R. Vasavada, R. C. Anderson, C. J. Baker, R. Barry, D. F. Blake, P. Conrad, K. S. Edgett, B. Ferdowski, R. Gellert, J. B. Gilbert, M. Golombek, J. Gómez-Elvira, D. M. Hassler, L. Jandura, M. Litvak, P. Mahaffy, J. Maki, M. Meyer, M. C. Malin, I. Mitrofanov, J. J. Simmonds, D. Vaniman, and R. C. Welch, Richard V. Iand Wiens. Mars Science Laboratory Mission and Science Investigation. *Space Science Reviews*, 170(1):5–56, 2012. ISSN 1572-9672. doi: 10.1007/s11214-012-9892-2. URL `http://dx.doi.org/10.1007/s11214-012-9892-2`.

D. Hähnel, W. Burgard, D. Fox, and S. Thrun. An Efficient FastSLAM Algorithm for Generating Maps of Large-Scale Cyclic Environments from Raw Laser Range

Measurements. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

M. Hellerer, M. J. Schuster, and R. Lichtenheldt. Software-in-the-loop Simulation of a Planetary Rover. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 2016.

J. Hidalgo-Carrió, P. Poulakis, and F. Kirchner. Adaptive localization and mapping with application to planetary rovers. *Journal of Field Robotics (JFR)*, 2018. doi: 10.1002/rob.21790. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21790.

H. Hirschmüller. *Stereo Vision Based Mapping and Immediate Virtual Walkthroughs*. PhD thesis, De Montfort University, 2003.

H. Hirschmüller. Stereo Processing by Semiglobal Matching and Mutual Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (IPAMI)*, 30(2): 328–341, Feb. 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1166.

H. Hirschmüller, P. R. Innocent, and J. M. Garibaldi. Fast, Unconstrained Camera Motion Estimation from Stereo without Tracking and Robust Statistics. In *IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pages 1099–1104, 2002. ISBN 981-04-8364-3. doi: 10.1109/ICARCV.2002.1238577.

D. Holz and S. Behnke. Registration of Non-Uniform Density 3D Point Clouds using Approximate Surface Reconstruction. In *International Symposium on Robotics (ISR) and the German Conference on Robotics (ROBOTIK)*, Munich, Germany, 2014.

A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 34(3):189–206, Feb. 2013. ISSN 0929-5593. doi: 10.1007/s10514-012-9321-0. URL http://link.springer.com/10.1007/s10514-012-9321-0.

A. Howard. Multi-robot Simultaneous Localization and Mapping using Particle Filters. *The International Journal of Robotics Research*, 25(12):1243–1256, Dec. 2006. ISSN 0278-3649. doi: 10.1177/0278364906072250. URL http://ijr.sagepub.com/cgi/doi/10.1177/0278364906072250.

M. Huber. NASA Helicopter Could Fly on Mars. http://www.ainonline.com/aviation-news/aerospace/2016-06-23/nasa-helicopter-could-fly-mars, 2016. Accessed 28 March 2017.

IAF and ZARM. 69th International Astronautical Congress Bremen 1-5 Oct 2018, 2018. URL https://www.iac2018.org. Accessed 7 December 2018.

ISECG. Global Exploration Roadmap. Technical report, International Space Exploration Coordination Group, 2018.

E. Jennings, J. Seguí, J. Gao, L. Clare, and D. Abraham. The Impact of Traffic Prioritization on Deep Space Network Mission Traffic. In *IEEE Aerospace Conference*, Mar. 2011. doi: 10.1109/AERO.2011.5747334.

J. Jessup, S. N. Givigi, and A. Beaulieu. Robust and Efficient Multirobot 3-D Mapping Merging With Octree-Based Occupancy Grids. *IEEE Systems Journal*, 11(3):1723–1732, Sept 2015. ISSN 1932-8184. doi: 10.1109/JSYST.2015.2422615.

J. P. Jessup. Merging of Octree Based 3D Occupancy Grid Maps. Master's thesis, Royal Military College of Canada, 2013. URL https://espace.rmc.ca/bitstream/11264/83/1/Thesis_Jamie.pdf.

JPL NASA. Mars Science Laboratory: Science, 2013. URL http://mars.jpl.nasa.gov/msl/mission/science/.

M. Kaess. AprilTags C++ Library. http://people.csail.mit.edu/kaess/apriltags, 2013. Accessed 3 January 2017.

M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. iSAM2 : Incremental Smoothing and Mapping Using the Bayes Tree. *The International Journal of Robotics Research (IJRR)*, 31:217–236, 2012. doi: 10.1177/0278364911430419.

T. Kaku, J. Haruyama, W. Miyake, A. Kumamoto, K. Ishiyama, T. Nishibori, K. Yamamoto, S. T. Crites, T. Michikami, Y. Yokota, R. Sood, H. J. Melosh, L. Chappaz, and K. C. Howell. Detection of Intact Lava Tubes at Marius Hills on the Moon by SELENE (Kaguya) Lunar Radar Sounder. *Geophysical Research Letters*, 44(20): 10,155–10,161, 2017. ISSN 1944-8007. doi: 10.1002/2017GL074998. URL http://dx.doi.org/10.1002/2017GL074998.

C. Kanzog. ROBEX - Robotic Exploration of Extreme Environments. http://www.robex-allianz.de/en/, 2017. Accessed 19 October 2017.

B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller. Multiple Relative Pose Graphs for Robust Cooperative Mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3185–3192, May 2010. doi: 10.1109/ROBOT.2010.5509154.

P. Koch and S. Lacroix. Managing environment models in multi-robot teams. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47(2):498 – 519, 2001.

R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A General Framework for Graph Optimization. *IEEE International Conference on Robotics and Automation (ICRA)*, May 2011a. doi: 10.1109/ICRA.2011.5979949.

R. Kümmerle, B. Steder, C. Dornhege, A. Kleiner, G. Grisetti, and W. Burgard. Large Scale Graph-based SLAM using Aerial Images as Prior Information. *Autonomous Robots*, 30(1):25–39, 2011b. doi: 10.1007/s10514-010-9204-1.

I. S. Kweon and T. Kanade. High-Resolution Terrain Map from Multiple Sensor Data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):278–292, 1992.

M. Labbé and F. Michaud. Online Global Loop Closure Detection for Large-Scale Multi-Session Graph-Based SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014. doi: 10.1109/IROS.2014.6942926.

Y. Latif, C. Cadena, and J. Neira. Robust Graph SLAM Back-ends: A Comparative Analysis. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, IL, USA, 2014. ISBN 9781479969333. doi: 10.1109/IROS.2014.6942929.

M. T. Lázaro, L. M. Paz, P. Piniés, J. A. Castellanos, and G. Grisetti. Multi-Robot SLAM using Condensed Measurements. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013. doi: 10.1109/IROS.2013.6696483.

G. H. Lee, F. Fraundorfer, and M. Pollefeys. Robust Pose-Graph Loop-Closures with Expectation-Maximization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013. ISBN 9781467363587. doi: 10.1109/IROS.2013.6696406.

H. Lehner, M. J. Schuster, T. Bodenmüller, and S. Kriegel. Exploration with Active Loop Closing: A Trade-off between Exploration Efficiency and Map Quality. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017. doi: 10.1109/IROS.2017.8206521.

R. C. Leishman, T. W. McLain, and R. W. Beard. Relative Navigation Approach for Vision-Based Aerial GPS-Denied Navigation. In *IEEE International Conference* *on Unmanned Aircraft Systems (ICUAS)*, May 2013. doi: 10.1109/ICUAS.2013.6564707.

X. Li and I. Guskov. Multi-scale Features for Approximate Alignment of Point-based Surfaces. In *Eurographics Symposium on Geometry Processing*, 2005.

R. D. Lorenz, E. P. Turtle, J. W. Barnes, M. G. Trainer, D. S. Adams, K. E. Hibbard, C. Z. Sheldon, K. Zacny, P. N. Peplowski, D. J. Lawrence, M. A. Ravine, T. G. McGee, K. S. Sotzen, S. M. MacKenzie, J. W. Langelaan, S. Sven, L. S. Wolfarth, and P. D. Bedini. Dragonfly: A Rotorcraft Lander Concept for Scientific Exploration at Titan. Technical report, Johns Hopkins APL Technical Digest, 2017. URL `http://dragonfly.jhuapl.edu/docs/DragonflyTechDigestAPL.pdf`.

S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford. Visual Place Recognition: A Survey. *IEEE Transactions on Robotics*, 32(1): 1–19, Feb 2016. ISSN 1552-3098. doi: 10.1109/TRO.2015.2496823.

P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings of the National Institute of Science of India*, volume 2, pages 49–55, 1936.

M. Maimone, A. Johnson, Y. Cheng, R. Willson, L. e. M. H. Matthies, and O. Khatib. *Experimental Robotics IX*, chapter Autonomous Navigation Results from the Mars Exploration Rover (MER) Mission, pages 3–13. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. ISBN 978-3-540-33014-1. doi: 10.1007/11552246_1. URL `http://dx.doi.org/10.1007/11552246_1`.

J. Mankins. Mars Rover Technology Workshop Proceedings. Technical report, Jet Propulsion Laboratory (JPL) Internal Report D-4788, Pasadena, California, USA, Apr. 1987.

T. K. Marks, A. Howard, M. Bajracharya, G. W. Cottrell, and L. H. Matthies. Gamma-SLAM : Visual SLAM in Unstructured Environments. *Journal of Field Robotics (JFR)*, 26(1):26–51, 2009. doi: 10.1002/rob.20273.

C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. RSLAM: A System for Large-Scale Mapping in Constant-Time Using Stereo. *International Journal of Computer Vision*, 94(2):198–214, June 2010. ISSN 0920-5691. doi: 10.1007/s11263-010-0361-7. URL `http://link.springer.com/10.1007/s11263-010-0361-7`.

E. Mendes, P. Koch, and S. Lacroix. ICP-Based Pose-Graph SLAM. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 195–200, 2016.

Messe Berlin, May 2014. URL https://www.messe-berlin.de/Presse/Pressemitteilungen/News_3650.html. Accessed 30 November 2018.

N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, K. Ohno, E. Takeuchi, and S. Tadokoro. Collaborative Mapping of an Earthquake-Damaged Building via Ground and Aerial Robots. *Journal of Field Robotics (JFR)*, 29(5):832–841, 2012. doi: 10.1002/rob.21436.

G. Mohanarajah, V. Usenko, M. Singh, R. D'Andrea, and M. Waibel. Cloud-Based Collaborative 3D Mapping in Real-Time With Low-Cost Robots. *IEEE Transactions on Automation Science and Engineering*, 12(2):423–431, 2015. doi: 10.1109/TASE.2015.2408456.

M. G. Müller, F. Steidle, M. J. Schuster, P. Lutz, M. Maier, S. Stoneman, T. Tomić, and W. Stürzl. Robust Visual-Inertial State Estimation with Multiple Odometries and Efficient Mapping on an MAV with Ultra-Wide FOV Stereo Vision. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018. doi: 10.1109/IROS.2018.8594117.

K. Nagatani, Y. Okada, N. Tokunaga, S. Kiribayashi, K. Yoshida, K. Ohno, E. Takeuchi, S. Tadokoro, H. Akiyama, I. Noda, T. Yoshida, and E. Koyanagi. Multirobot exploration for search and rescue missions: A report on map building in robocuprescue 2009. *Journal of Field Robotics (JFR)*, 28(3):373–387, 2011. doi: 10.1002/rob.20389.

Y. Nakamura, G. V. Latham, and H. J. Dorman. Apollo Lunar Seismic Experiment – Final Summary. *Journal of Geophysical Research: Solid Earth*, 87(S01), 1982.

K. Neith. Countdown to Mars: Some Curious Facts, 2012. URL http://www.caltech.edu/content/countdown-mars-some-curious-facts.

R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-Time Dense Surface Mapping and Tracking. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Oct. 2011. doi: 10.1109/ISMAR.2011.6092378.

A. Nüchter, K. Lingemann, J. Hertzberg, and H. Surmann. 6D SLAM – 3D Mapping Outdoor Environments. *Journal of Field Robotics (JFR)*, 24(8-9):699–722, 2007.

J. Oberländer, S. Klemm, G. Heppner, A. Roennau, and R. Dillmann. A Multi-Resolution 3-D Environment Model for Autonomous Planetary Exploration. In *IEEE*

International Conference on Automation Science and Engineering (CASE), 2014. doi: 10.1109/CoASE.2014.6899331.

E. Olson. AprilTag: A robust and flexible visual fiducial system. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2011. doi: 10.1109/ICRA.2011.5979561.

E. Olson. AprilTags Visual Fiducial System. `https://april.eecs.umich.edu/software/apriltag.html`, 2016. Accessed 3 January 2017.

E. Olson, J. Strom, R. Goeddel, R. Morton, P. Ranganathan, and A. Richardson. Exploration and Mapping with Autonomous Robot Teams. *Communications of the ACM*, 56:62–70, 03 2013. doi: 10.1145/2428556.2428574.

OpenCV-Team. OpenCV (Open Source Computer Vision Library). `https://opencv.org/`, 2018. Accessed 20 March 2018.

C. Orlowski. DARPA Robotics Challenge (DRC). `http://www.darpa.mil/program/darpa-robotics-challenge`, 2016. Accessed 19 October 2017.

D. Pangercic, K. Mathe, Z.-C. Marton, L. C. Goron, M.-S. Opris, M. J. Schuster, M. Tenorth, D. Jain, T. Ruehr, and M. Beetz. A Robot that Shops for and Stores Groceries. AAAI Video Competition (AIVC 2011), August 7–11 2011. URL `http://youtu.be/x0Ybod_6ADA`.

M. Panzirsch, H. Singh, M. Stelzer, M. J. Schuster, C. Ott, and M. Ferre. Extended Predictive Model-Mediated Teleoperation of Mobile Robots through Multilateral Control. In *IEEE Intelligent Vehicles Symposium (IV)*, 2018. doi: 10.1109/IVS.2018.8500578.

Persistence of Vision Raytracer Pty. Ltd. OpenCV (Open Source Computer Vision Library). `http://www.povray.org/`, 2008. Accessed 23 March 2018.

S. Peters. Mars Science Laboratory: Pointing, Positioning, Phasing, and Coordinate Systems (PPPCS) Document, Volume 9: Surface Remote Sensing and Navigation. Technical report, Jet Propulsion Laboratory, California Institute of Technology, 2016.

P. Piniés and J. D. Tardós. Large-Scale SLAM Building Conditionally Independent Local Maps: Application to Monocular Vision. *IEEE Transactions on Robotics*, 24(5):1094–1106, Oct 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.2004636.

P. B. Quang, C. Musso, and F. Le Gland. An Insight into the Issue of Dimensionality in Particle Filtering. In *13th Conference on Information Fusion (FUSION)*, 2010. doi: 10.1109/ICIF.2010.5712050.

R. Reid and T. Bräunl. Large-scale Multi-robot Mapping in MAGIC 2010. In *IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, 2011. doi: 10.1109/RAMECH.2011.6070489.

R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. doi: 10.1109/ICRA.2011.5980567.

S. Saeedi, M. Trentini, M. Seto, and H. Li. Multiple-robot Simultaneous Localization and Mapping: A Review. *Journal of Field Robotics (JFR)*, 33(1):3–46, 2016. ISSN 1556-4967. doi: 10.1002/rob.21620. URL http://dx.doi.org/10.1002/rob.21620.

M. Schadler, J. Stückler, and S. Behnke. Data set Spacebot Arena. http://www.ais.uni-bonn.de/mav_registration/, 2014. Accessed 30 August 2016.

K. Schmid, F. Ruess, M. Suppa, and D. Burschka. State Estimation for highly dynamic flying Systems using Key Frame Odometry with varying Time Delays. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vilamoura, Portugal, Oct. 2012. doi: 10.1109/IROS.2012.6385969.

K. Schmid, P. Lutz, T. Tomić, E. Mair, and H. Hirschmüller. Autonomous Vision-based Micro Air Vehicle for Indoor and Outdoor Navigation. *Journal of Field Robotics (JFR)*, 31(4):537–570, 2014a. doi: 10.1002/rob.21506. URL https://onlinelibrary.wiley.com/doi/full/10.1002/rob.21506.

K. Schmid, F. Ruess, and D. Burschka. Local Reference Filter for Life-Long Vision Aided Inertial Navigation. In *17th International Conference on Information Fusion*, 2014b.

M. Schuster, R. Bormann, D. Steidl, S. Reynolds-Haertle, and M. Stilman. Stable Stacking for the Distributor's Pallet Packing Problem. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010a. doi: 10.1109/IROS.2010.5650217.

M. J. Schuster. Towards Autonomous Planetary Exploration: Collaborative Multi-Robot Localization and Mapping in GPS-denied Environments. In *International Technical Symposium on Navigation and Timing (ITSNT)*, Toulouse, France, Nov. 2017.

M. J. Schuster, J. Okerman, H. Nguyen, J. M. Rehg, and C. C. Kemp. Perceiving Clutter and Surfaces for Object Placement in Indoor Environments. In *10th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2010b. doi: 10.1109/ICHR.2010.5686328.

M. J. Schuster, D. Jain, M. Tenorth, and M. Beetz. Learning Organizational Principles in Human Environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2012. doi: 10.1109/ICRA.2012.6224553.

M. J. Schuster, C. Brand, H. Hirschmüller, M. Suppa, and M. Beetz. Multi-Robot 6D Graph SLAM Connecting Decoupled Local Reference Filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015. doi: 10.1109/IROS.2015.7354094.

M. J. Schuster, C. Brand, S. G. Brunner, P. Lehner, J. Reill, S. Riedel, T. Bodenmüller, K. Bussmann, S. Büttner, A. Dömel, W. Friedl, I. Grixa, M. Hellerer, H. Hirschmüller, M. Kassecker, Z.-C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler. The LRU Rover for Autonomous Planetary Exploration and its Success in the SpaceBotCamp Challenge. In *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, Bragança, Portugal, 2016. doi: 10.1109/ICARSC.2016.62.

M. J. Schuster, S. G. Brunner, K. Bussmann, S. Büttner, A. Dömel, M. Hellerer, H. Lehner, P. Lehner, O. Porges, J. Reill, S. Riedel, M. Vayugundla, B. Vodermayer, T. Bodenmüller, C. Brand, W. Friedl, I. Grixa, H. Hirschmüller, M. Kaßecker, Z.-C. Márton, C. Nissler, F. Ruess, M. Suppa, and A. Wedler. Towards Autonomous Planetary Exploration: The Lightweight Rover Unit (LRU), its Success in the SpaceBotCamp Challenge, and Beyond. *Journal of Intelligent & Robotic Systems (JINT)*, Nov. 2017. ISSN 1573-0409. doi: 10.1007/s10846-017-0680-9. URL `https://doi.org/10.1007/s10846-017-0680-9`.

M. J. Schuster, K. Schmid, C. Brand, and M. Beetz. Distributed stereo vision-based 6d localization and mapping for multi-robot teams. *Journal of Field Robotics (JFR)*, 2018. doi: 10.1002/rob.21812. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21812`.

M. Schwarz, M. Beul, D. Droeschel, S. Schüller, A. S. Periyasamy, C. Lenz, M. Schreiber, and S. Behnke. Supervised Autonomy for Exploration and Mobile Manipulation in Rough Terrain with a Centaur-Like Robot. *Frontiers in Robotics and AI*, 3:57, 2016. ISSN 2296-9144. doi: 10.3389/frobt.2016.00057. URL `https://www.frontiersin.org/article/10.3389/frobt.2016.00057`.

G. Schweighofer and A. Pinz. Robust Pose Estimation from a Planar Target. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):2024–2030, Dec 2006. ISSN 0162-8828. doi: 10.1109/TPAMI.2006.252.

J. Schwendner, S. Joyeux, and F. Kirchner. Using Embodied Data for Localization and

Mapping. *Journal of Field Robotics (JFR)*, 2014a. doi: https://doi.org/10.1002/rob.21489. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21489.

J. Schwendner, T. M. Roehr, S. Haase, M. Wirkus, M. Manz, S. Arnold, and J. Machowinski. The Artemis Rover as an Example for Model Based Engineering in Space Robotics. In *ICRA Workshop on Modelling, Estimation, Perception and Control of All Terrain Mobile Robots*, 2014b.

S. Siceloff. Extreme Access Flyer to Take Planetary Exploration Airborne, 2015. URL https://www.nasa.gov/feature/extreme-access-flyer-to-take-planetary-exploration-airborne. Accessed 29 November 2018.

A. Stelzer. *Approaches to Efficient Visual Homing of Mobile Robots in Rough Terrain*. PhD thesis, Albert-Ludwigs-Universität Freiburg im Breisgau, 2016.

A. Stelzer, H. Hirschmüller, and G. Martin. Stereo-vision-based navigation of a six-legged walking robot in unknown rough terrain. *The International Journal of Robotics Research*, 31(4):381–402, Feb. 2012. ISSN 0278-3649. doi: 10.1177/0278364911435161. URL http://ijr.sagepub.com/cgi/doi/10.1177/0278364911435161.

K. Strobl and G. Hirzinger. More Accurate Camera and Hand-Eye Calibrations with Unknown Grid Pattern Dimensions. In *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, USA, 2008.

K. H. Strobl and G. Hirzinger. Optimal Hand-Eye Calibration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, Oct. 2006. doi: 10.1109/IROS.2006.282250.

N. Sünderhauf, P. Neubert, M. Truschzinski, D. Wunschel, J. Pöschmann, S. Lange, and P. Protzel. Phobos and Deimos on Mars – Two Autonomous Robots for the DLR Spacebot Cup. In *International Symposium on Artificial Intelligence, Robotics and Automation in (Space-i-SAIRAS)*. The Canadian Space Agency (CSA-ASC), 2014.

J. Thangavelautham, M. S. Robinson, A. Taits, T. McKinney, S. Amidan, and A. Polak. Flying, Hopping Pit-Bots for Cave and Lava Tube Exploration on the Moon and Mars. In *2nd International Workshop on Instrumentation for Planetary Missions*, 2014. URL http://arxiv.org/abs/1701.07799.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, Massachusetts, USA, 2005. doi: 10.1017/S0269888906210993.

F. Tombari and L. Di Stefano. Object Recognition in 3D Scenes with Occlusions and Clutter by Hough Voting. In *Fourth Pacific-Rim Symposium on Image and Video Technology (PSIVT)*. IEEE, 2010. doi: 10.1109/PSIVT.2010.65.

F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part III*, pages 356–369. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-642-15558-1. doi: 10.1007/978-3-642-15558-1_26.

F. Tombari, S. Salti, and L. Di Stefano. A combined texture-shape descriptor for enhanced 3d feature matching. In *IEEE International Conference on Image Processing (ICIP)*, 2011. doi: 10.1109/ICIP.2011.6116679.

R. Triebel, P. Pfaff, and W. Burgard. Multi-Level Surface Maps for Outdoor Terrain Mapping and Loop Closing. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13 (4):376 – 380, 1991. doi: 10.1109/34.88573.

M. Vayugundla, F. Steidle, M. Smisek, M. J. Schuster, K. Bussmann, and A. Wedler. Datasets of Long Range Navigation Experiments in a Moon Analogue Environment on Mount Etna. In *International Symposium on Robotics (ISR)*, 2018.

S. Vetter. Error Analysis and Modelling for Marker-based 6D Pose Estimation. Master's thesis, Universität Passau, Germany, 2015.

Vicon Motion Systems Ltd. VICON Camera Systems. `https://www.vicon.com/products/camera-systems`, 2019. Accessed 9 May 2019.

T. A. Vidal-Calleja, C. Berger, J. Solà, and S. Lacroix. Large scale multiple robot visual mapping with heterogeneous landmarks in semi-structured terrain. *Robotics and Autonomous Systems*, 59(9):654–674, Sept. 2011. ISSN 09218890. doi: 10.1016/j.robot.2011.05.008. URL `http://linkinghub.elsevier.com/retrieve/pii/S0921889011000923`.

VRmagic Imaging GmbH. D3 intelligent components: VRmS-16 Sensor Board. `https://www.vrmagic.com/fileadmin/downloads/imaging/Camera_Datasheets/intelligent_cameras/VRmD3MFC_multisensor.pdf`, 2018. Accessed 9 May 2019.

J. Wang and E. Olson. AprilTag 2: Efficient and robust fiducial detection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

A. Wedler. Helmholtz-Future Topic Project ARCHES, 2018. URL `https://www.arches-projekt.de/en/helmholtz-future-topic-project-arches/`. Accessed 7 December 2018.

A. Wedler, A. Maier, J. Reill, C. Brand, H. Hirschmüller, M. J. Schuster, M. Suppa, A. Beyer, N. Y. Lii, M. Maier, H.-J. Sedlmayr, and R. Haarmann. Pan/Tilt-Unit as a Perception Module for Extra-Terrestrial Vehicle and Landing Systems. In *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Noordwijk, Netherlands, 2013.

A. Wedler, B. Rebele, J. Reill, M. Suppa, H. Hirschmüller, C. Brand, M. Schuster, B. Vodermayer, H. Gmeiner, A. Maier, B. Willberg, K. Bussmann, F. Wappler, M. Hellerer, and R. Lichtenheldt. LRU - Lightweight Rover Unit. In *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*, Noordwijk, Netherlands, 2015.

A. Wedler, M. Vayugundla, H. Lehner, P. Lehner, M. J. Schuster, S. G. Brunner, W. Stürzl, A. Dömel, H. Gmeiner, B. Vodermayer, B. Rebele, I. L. Grixa, K. Bussmann, J. Reill, B. Willberg, A. Maier, P. Meusel, F. Steidle, M. Smisek, M. Hellerer, M. Knapmeyer, F. Sohl, A. Heffels, L. Witte, C. Lange, R. Rosta, N. Toth, S. Voelk, A. Kimpe, P. Kyr, and M. Wilde. First Results of the ROBEX Analogue Mission Campaign: Robotic Deployment of Seismic Networks for Future Lunar Missions. In *International Astronautical Congress (IAC)*, 2017.

A. Wedler, K. Bussmann, A. Dömel, M. Drauschke, H. Gmeiner, I. L. Grixa, H. Lehner, M. Vayugundla, M. G. Müller, J. Reill, M. Schuster, W. Stürzl, B. Vodermayer, P. Lehner, S. Brunner, and M. Görner. From the ROBEX Experiment Toward the Robotic Deployment and Maintenance of Scientific Infrastructure for Future Planetary Exploration Missions. In *42nd COSPAR Scientific Assembly*, Pasadena, California, USA, July 2018a.

A. Wedler, M. Wilde, J. Reill, M. J. Schuster, M. Vayugundla, S. G. Brunner, K. Bussmann, A. Dömel, M. Drauschke, H. Gmeiner, H. Lehner, P. Lehner, M. G. Müller, W. Stürzl, R. Triebel, B. Vodermayer, A. Börner, R. Krenn, A. Dammann, U.-C. Fiebig, E. Staudinger, F. Wenzköfer, S. Flögel, S. Sommer, T. Asfour, M. Flad, S. Hohmann, M. Brandauer, and A. O. Albu-Schäffer. From single autonomous robots toward cooperative robotic interactions for future planetary exploration missions. In *69th International Astronautical Congress (IAC)*, Bremen, Germany, Oct. 2018b.

S. M. Weiss. *Vision Based Navigation for Micro Helicopters*. PhD thesis, Eidgenössische Technische Hochschule ETH Zürich, Zürich, Switzerland, 2012.

J. Welle, D. Schulz, T. Bachran, and A. B. Cremers. Optimization Techniques for Laser-Based 3D Particle Filter SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, Alaska, USA, May 2010. doi: 10.1109/ROBOT.2010.5509992.

D. Wettergreen, M. Wagner, D. Jonak, V. Baskaran, M. Deans, S. Heys, D. Pane, T. Smith, J. Teza, D. R. Thompson, P. Tompkins, and C. Williams. Long-Distance Autonomous Survey and Mapping in the Robotic Investigation of Life in the Atacama Desert. In *International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS)*, 2008.

S. Williams, V. Indelman, M. Kaess, R. Roberts, J. J. Leonard, and F. Dellaert. Concurrent Filtering and Smoothing : A Parallel Architecture for Real-Time Navigation and Full Smoothing. *International Journal of Robotics Research*, 33(12):1–47, 2014. doi: 10.1177/0278364914531056.

N. Wolchover. NASA Gives Up On Stuck Mars Rover Spirit, 2011. URL `http://www.space.com/11773-nasa-mars-rover-spirit-mission-ends.html`.

K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: A Probabilistic, Flexible, and Compact 3D Map Representation for Robotic Systems. In *Proc. of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation*, Anchorage, AK, USA, May 2010.

Xsens. MTi 10-series. `https://www.xsens.com/products/mti-10-series/`, 2019. Accessed 9 May 2019.

B. Yamauchi. Frontier-Based Exploration Using Multiple Robots. In *Autonomous Agents*, pages 47–53. ACM, 1998. doi: 10.1145/280765.280773.

K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad. Real-Time RGB-D Registration and Mapping in Texture-less Environments Using Ranked Order Statistics. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Chicago, IL, USA, 2014. doi: 10.1109/IROS.2014.6942925.

Y. Yue, D. Wang, P. G. C. N. Senarathne, and D. Moratuwage. A Hybrid Probabilistic and Point Set Registration Approach for Fusion of 3D Occupancy Grid Maps. In *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2016. doi: 10.1109/SMC.2016.7844529.