Received: 21 December 2017 | Revised: 11 July 2018 | Accepted: 13 June 2018
DOI: 10.1002/rob.21812

REGULAR ARTICLE

Distributed stereo vision-based 6D localization and mapping for multi-robot teams

Martin J. Schuster[1] | Korbinian Schmid[2] | Christoph Brand[1] | Michael Beetz[3]

[1]Department of Perception and Cognition, Robotics and Mechatronics Center (RMC), German Aerospace Center (DLR), Weßling, Germany

[2]Roboception GmbH, München, Germany

[3]Institute for Artificial Intelligence and Center for Computing Technologies (TZI), Faculty of Computer Science, University Bremen, Bremen, Germany

Correspondence
Martin J. Schuster, Department of Perception and Cognition, Robotics and Mechatronics Center (RMC), German Aerospace Center (DLR), Münchener Str. 20, 82234 Weßling, Germany.
Email: martin.schuster@dlr.de

Funding information
Helmholtz-Gemeinschaft, Grant/Award Numbers: HA-304, ZT-0033; FP7 Information and Communication Technologies

Abstract

Joint simultaneous localization and mapping (SLAM) constitutes the basis for cooperative action in multi-robot teams. We designed a stereo vision-based 6D SLAM system combining local and global methods to benefit from their particular advantages: (1) Decoupled local reference filters on each robot for real-time, long-term stable state estimation required for stabilization, control and fast obstacle avoidance; (2) Online graph optimization with a novel graph topology and intra- as well as inter-robot loop closures through an improved submap matching method to provide global multi-robot pose and map estimates; (3) Distribution of the processing of high-frequency and high-bandwidth measurements enabling the exchange of aggregated and thus compacted map data. As a result, we gain robustness with respect to communication losses between robots. We evaluated our improved map matcher on simulated and real-world datasets and present our full system in five real-world multi-robot experiments in areas of up 3,000 m^2 (bounding box), including visual robot detections and submap matches as loop-closure constraints. Further, we demonstrate its application to autonomous multi-robot exploration in a challenging rough-terrain environment at a Moon-analogue site located on a volcano.

KEYWORDS
graph SLAM, map matching, mobile robots, multi-robot, navigation filter

1 | INTRODUCTION

The exploration of moons and foreign planets is an important current and future application for mobile robots as their surfaces are difficult to reach and hard to access for humans. The application of huge and complex robot systems such as Curiosity, landed on Mars in 2012, creates many single points of failure for a mission. As a consequence, these rovers have to move very slowly and carefully to avoid getting stuck, as the Mars rover Spirit did in 2009 (Wolchover, 2011). The future deployment of teams of multiple robots can avoid these single points of failure by gaining robustness through redundancy and, in addition, can improve efficiency through parallelization. The robots have to travel through previously unknown unstructured rough terrain, operating in areas where external methods for localization like global navigation satellite systems (GNSS) are not available or expensive to set up. Communication links to the robots are limited and heavily delayed, featuring for example 8–40 min round trip time between Earth and Mars. Furthermore, communication between the robots cannot be guaranteed at all times, in particular at scientifically interesting places such as craters, canyons, or caves. As teleoperation therefore becomes inefficient or infeasible, robot autonomy is a key aspect for future planetary exploration missions. Any coordinated (semi-)autonomous operation in such challenging environments requires up-to-date localization estimates for all robots in a team as well as a joint map to operate on.

To tackle these challenges, we designed a framework for 6D local and global simultaneous localization and mapping (SLAM) for heterogeneous multi-robot teams. We therein combine keyframe-based local reference filters (Schmid, Ruess, & Burschka, 2014b) and multi-robot graph SLAM with incremental optimization (Schuster, Brand, Hirschmüller, Suppa, & Beetz, 2015). A decoupled integration of these local and global methods allows us to benefit from their particular advantages: 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 to independently estimate its own pose and map on-board and online at all times. We thereby can distribute significant parts of the computational workload, avoid single points of failure and gain robustness with respect to interrupted communication and failure of individual robots. A novel SLAM graph topology, which we first introduced in Schuster et al. (2015), allows a better integration of the filter results according to their estimated uncertainty and independence assumptions, leading to more accurate estimates.

We equip all of our robots with stereo cameras as space-qualifiable vision sensors and employ semi-global matching (SGM; Hirschmüller, 2008) to compute dense 3D data under varying light conditions. On each robot, we aggregate the 3D data along the trajectories estimated by our local reference filters into local submaps of limited size and uncertainty. We then apply online graph SLAM to create and optimize a dense joint 3D map as well as to compute 6D pose estimates for all participating robots. In Figure 1, we present a joint 3D probabilistic voxel-grid map created by two lightweight rover units (LRU; Schuster et al., 2017) and give an impression of our multi-robot experimental setup. We exchange 3D data between robots only in the aggregated format of submaps to reduce bandwidth requirements and distribute the computational workload. 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 submaps with limited drift allows us to tackle this challenge: In Brand, Schuster, Hirschmüller, and Suppa (2015), we first presented a novel approach to select and match submaps, thereby computing an estimate for their relative transformation as well as for its uncertainty. For global pose and map optimization, we integrate marker-based visual robot detections as well as these submap matches as intra- and inter-robot loop-closure constraints into our SLAM graph.

In this study, we combine and summarize methods first presented in conference papers on

- Local reference filters for long-term stable real-time state estimation (Schmid et al., 2014b)
- Multi-robot graph SLAM for global joint localization and mapping (Schuster et al., 2015)
- Submap matching for loop-closure generation (Brand et al., 2015)

and describe central aspects in more detail. In addition, we present novel contributions going beyond the three conference papers:

- Improved map matching method to maximize the number of loop closures: We leverage the properties of our local reference filter to separate observable and unobservable system states and can thereby reduce the dimensionality of the matching problem from 6D to 4D. We evaluated it in 40 simulated and 13 real experiments and achieved an average increase in the number of map matches of 40%.
- Evaluation of our full localization and mapping system in five novel multi-robot experiments and discussion of the resources required by its individual components. Compared to our previous work (Schuster et al., 2015), the experiments feature different robots and larger areas of up to 57 m × 53 m.
- Demonstration of the application of our localization and mapping system to autonomous multi-robot exploration in a novel experiment, during which our two LRU rovers mapped an area of approx. 650 m^2 in a rough-terrain outdoor environment at a Moon-analogue site on the volcano Mt. Etna.

We start with a survey of related work in Section 2 and give an overview over our modular system architecture in Section 3. We summarize our local reference filter for real-time vision-based

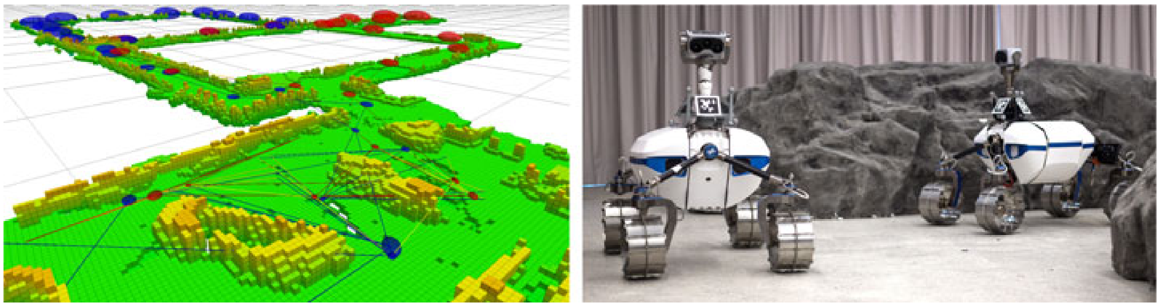

FIGURE 1 Left: Multi-robot 3D probabilistic voxel-grid map (height-colored, resolution: 10 cm, grid size: 5 m) and SLAM graph with pose covariance ellipsoids, created by our two lightweight rover units LRU1 (blue) and LRU2 (red). We provide a more detailed description in Section 7.3 and Figure 10. Right: Experimental setup with our two rovers

inertial navigation in Section 4. In the subsequent Section 5, we introduce our 3D mapping system, describing its individual components from submap creation to our novel 4D map matching in the respective subsections. In Section 6, we present in detail our incremental graph SLAM with its multi-robot graph topology to connect local reference filter estimates. Afterwards, in Section 7, we discuss both the evaluation of our novel 4D map matching method as well as five extended multi-robot experiments. In the subsequent Section 8, we present an application of our system to multi-robot autonomous exploration in a novel experiment at a Moon-analogue site and discuss the lessons learned in this challenging environment. In the final Section 9, we summarize our contributions on distributed 6D multi-robot localization and mapping and provide an outlook on topics for future work.

2 | RELATED WORK

Within the large body of related work on SLAM, three fundamental techniques can be identified: Kalman Filter-based methods such as Extended Kalman Filters (EKF), Rao-Blackwellized particle filters (RBPF), and graph optimization. Recent overviews on the challenges and approaches for SLAM systems in general and for multi-robot SLAM in particular are given by Cadena et al. (2016) and Saeedi, Trentini, Seto and Li (2016), respectively.

We employ an EKF in our keyframe-based local reference inertial navigation filter (LR-VINS) to guarantee bounded computation times for local state estimation, making it suitable for real-time applications such as control. EKF-based vision-aided inertial navigation systems (VINS) exist in different coupling configurations: Weiss (2012) demonstrated a loosely coupled system, fusing poses from a keyframe-based mono-SLAM with IMU measurements. Tightly coupled systems as the MSCKF (Mourikis & Roumeliotis, 2007) directly include feature measurements into an EKF. In the work by Hesch, Kottas, Bowman, and Roumeliotis (2014), consistency of the MSCKF is further improved by modifications of the propagation and measurement matrices according to nonobservability properties. To remove initial conditions for VINS systems and to reduce the effect of nonlinearities in the estimation process, Forster, Carlone, Dellaert and Scaramuzza (2017) and Lupton and Sukkarieh (2012) use IMU data preintegration. We will further discuss these works in the context of our LR-VINS in Section 4.

For the global optimization problem, we regard the technique of EKFs as less suited. As EKF SLAM models landmark-based maps as multivariate Gaussians, they typically imply a computational effort that grows quadratically with the number of landmarks (Durrant-Whyte & Bailey, 2006). While RBPFs (Grisetti, Stachniss, & Burgard, 2007) have been widely used for LIDAR-based planar localization and mapping, they are unsuitable for multi-robot 6D SLAM as the required number of particles grows exponentially with the size of the state space (Quang, Musso, & Le Gland, 2010). Graph SLAM started out with batch optimization methods (Kümmerle, Grisetti, Strasdat, Konolige, & Burgard, 2011) for offline least-squares error minimization. Incremental approaches like iSAM2 (Kaess et al., 2012) enabled its application for online robot localization and mapping. Robot and landmark poses are modeled as nodes in a graph that reflects their, typically sparse, dependencies. They are connected via measurement constraints, represented as edges weighted according to their respective Gaussian uncertainty estimates. The worst-case computational effort on loop closures typically grows with the number of measurements and thus with traveled distance, a challenge that can be approached by constraining the optimization to local regions (Mei, Sibley, Cummins, Newman, & Reid, 2011) or removing nodes through marginalization (Williams et al., 2014). For 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 manageable (Ahmad, Tipaldi, Lima, & Burgard, 2013).

While graph optimization constitutes the back-end of a SLAM system, the front-end is concerned with data association. It can, for example, be approached by using image features as identifiable landmarks (Endres et al., 2012), by matching visual keyframes (Leishman, McLain, & Beard, 2013) or through the registration of depth data (Newcombe et al., 2011). In this study, we employ visual detections of other robots as well as submap matching to generate loop-closure constraints. The creation of submaps is a technique to locally aggregate sensor data into maps of limited size (Reid & Bräunl, 2011; Vidal-Calleja, Berger, Sola, & Lacroix, 2011; Williams, Dissanayake, & Durrant-Whyte, 2002). Their origins can be attached as nodes to the slam graph, the graph optimization thereby affecting their relative transformations. This leads to a sparse graph, while allowing us to keep more information through aggregation into submaps than in keyframe-based mapping approaches (Leishman et al., 2013; Mohanarajah, Usenko, Singh, D'Andrea, & Waibel, 2015), which apply sparse temporal sampling on high-bandwidth sensor data.

Submap matching for multi-robot systems has been proposed by Williams et al. (2002) to match sparse feature-based local maps against a global model. In dense mapping, submaps typically are matched against each other as a global model becomes computationally challenging to handle. Forster, Pizzoli, and Scaramuzza (2013) and Reid and Bräunl (2011) successfully used submap matching using a brute-force correlation search on 2D and 2.5D elevation maps, respectively. This, however, requires the computational resources of a GPU even when limiting the search space to a discretization of $\mathbb{R}^3$ or $\mathbb{R}^2$, respectively. Nagatani et al. (2011) use an iterative closest point (ICP) algorithm to match LIDAR-based 2.5D maps created by multiple robots. Similar to our approach, they match submaps of limited size, restrict the matching to neighboring submaps and use the resulting transformations as constraints for graph optimization. For 3D mapping, Labbé and Michaud (2014) and Mohanarajah et al. (2015) employ graph SLAM with SURF features for loop-closure generation. Such image-based features, however, are not robust to changes in viewpoint and illumination and thereby not well suited for heterogeneous multi-robot teams (Forster et al., 2013). On 3D geometry, ICP algorithms have been shown to work well for frame-to-frame registration (Newcombe et al., 2011; Nüchter, Lingemann, Hertzberg & Surmann, 2007). They can be used for map matching when a close initial estimate is available (Mendes, Koch, & Lacroix, 2016), for example, 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 (Alexandre, 2012; Li & Guskov, 2005; Tombari, Salti, & DiStefano, 2011) have become popular to find correspondences between point clouds, a central challenge being the selection and description of robust geometric features (Yousif, Bab-Hadiashar, & Hoseinnezhad, 2014). The estimation of initial inter-robot pose constraints is approached by Dong, Nelson, Indelman, Michael, and Dellaert (2015) for 2D LIDAR maps by clustering a large number of transformation hypotheses from scan matches. While this could complement our visual robot detections, the computational effort to generate transformation hypotheses would be significantly higher for 3D maps. In addition, our aggregation into submaps leads to a lower number of potential hypotheses, which might not be sufficient for a clustering-based method.

We integrate filter and graph SLAM methods to fulfill both local real-time requirements as well as to compute global multi-robot estimates. While Leishman et al. (2013) and Mohanarajah et al. (2015) also combine keyframe-based approaches with a pose graph for global localization, in contrast to our work, they explicitly represent all RGBD keyframes as nodes in their graph. We decouple the graph from such low-level states and trigger the creation of local reference frames as new graph nodes from our higher-level mapping modules. Thereby, we keep the size of the graph independent from the number of keyframes, which typically is orders of magnitude larger than the number of submaps required in our system. 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 keep it small for fast global optimization. In addition, we do not feed back loop-closure results into our filter, allowing it to generate smooth estimates required for stabilization and control of highly dynamic systems.

We considered several existing 3D multi-robot graph SLAM approaches, however, each of them has its own limitations, either restricting the enforcement of loop closures (Vidal-Calleja et al., 2011), assuming unlimited communication (Kim et al., 2010) or having been evaluated in simulation under simplifying assumptions (Cunningham, Indelman, & Dellaert, 2013). The latter two connect pose graphs created by multiple robots through frame-of-reference constraints represented as nodes in the graph. Cunningham et al. (2013) and Lázaro, Paz, Piniés, Castellanos, and Grisetti (2013) exchange condensed graphs between robots, the latter explicitly removing double-counted information by introducing antifactors. Double-counting cannot occur in our proposed system as each robot adds all estimates and measurements to their own graph only once. We do not expect significant benefits from an exchange of optimized partial graphs as our combination of local reference filters with a submapping approach leads to a small joint graph in the first place.

3 | MULTI-ROBOT LOCALIZATION AND MAPPING ARCHITECTURE

We present an overview of our software architecture in Figure 2. It shows in detail our localization and mapping layer and indicates its connections to the robots' sensors and lower-level perception as well as to the robot control and higher-level planning components. To establish the data flow between our components, we employ three different middlewares, the first two being developed at our institute: 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. As sensors, we use cameras as well as an inertial measurement unit (IMU). On our ground-based robots, we additionally include wheel odometry measurements to improve the accuracy of the filter output, as indicated in Figure 2. We expect these three sensor modalities to be available on real planetary rovers, see for example the self-localization architecture proposed by Souvannavong, Lemaréchal, Rastel, and Maurette (2010) for the ExoMars rover, which is planned to be launched in 2020. Within our perception layer, we employ semi-global matching (SGM; Hirschmüller, 2008) on an FPGA for dense stereo reconstruction and use this to compute keyframe-based visual odometry (Hirschmüller, Innocent, & Garibaldi, 2002). In addition, we perform marker-based visual detections of other robots and estimate their 6D poses (Olson, 2011). For both visual odometry and robot detections, we estimate the uncertainty and pass it to the subsequent filter and SLAM.

Our focus is on the localization and mapping layer, which contains three major modules: First, we fuse IMU and visual odometry estimates in a local reference filter (Schmid et al., 2014b) for real-time robust local state estimation, which we discuss in detail in Section 4. Second, on our ground-based rovers that operate in rough terrain, we compute a stereo-error adaptive local obstacle and terrain classification directly on depth images (Brand, Schuster, Hirschmüller, & Suppa, 2014). Performing this step early on allows a consideration of the association between original camera viewpoints and depth data, which is lost when aggregating it into maps later on. The results can be used for fast obstacle avoidance and local path planning on local 2.5D cost maps. Third, our 6D multi-robot SLAM

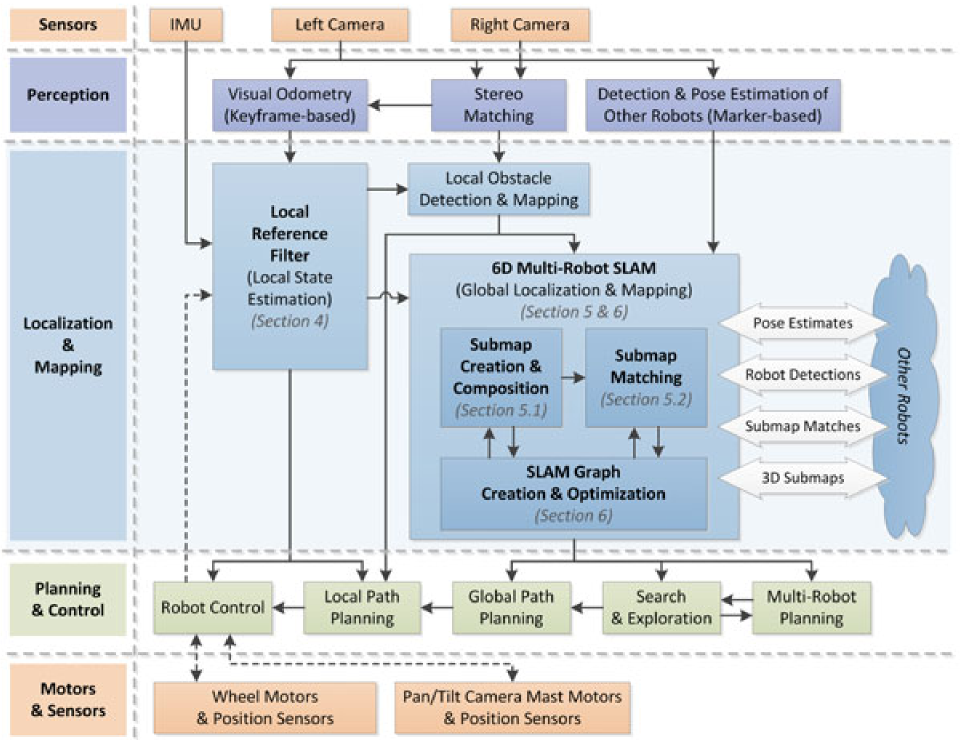
Sensors
IMU
Left Camera
Right Camera
Perception
Visual Odometry (Keyframe-based)
Stereo Matching
Detection & Pose Estimation of Other Robots (Marker-based)
Localization & Mapping
Local Reference Filter (Local State Estimation) (Section 4)
Local Obstacle Detection & Mapping
6D Multi-Robot SLAM (Global Localization & Mapping) (Section 5 & 6)
Submap Creation & Composition (Section 5.1)
Submap Matching (Section 5.2)
SLAM Graph Creation & Optimization (Section 6)
Pose Estimates
Robot Detections
Submap Matches
3D Submaps
Other Robots
Planning & Control
Robot Control
Local Path Planning
Global Path Planning
Search & Exploration
Multi-Robot Planning
Motors & Sensors
Wheel Motors & Position Sensors
Pan/Tilt Camera Mast Motors & Position Sensors

FIGURE 2 Multi-robot navigation and mapping architecture [Color figure can be viewed at wileyonlinelibrary.com]

framework contains components to create submaps by integrating depth data along the trajectories given by our local pose estimates, for incremental online SLAM graph optimization, as well as for the generation of loop-closure constraints. For relocalization in areas previously visited by the same and by other robots, we designed a method to match local, partial maps based on the 3D geometry of the environment. We present the individual components in detail in Sections 5 and 6.

As introduced in Schuster et al. (2015), the localization and mapping modules are executed on board all robots within a multi-robot team in a distributed fashion. This ensures the online availability of up-to-date pose and map estimates on each robot at all times, increasing robustness in particular in light of communication losses within the robot team. As we do not assume any initial knowledge about the starting positions of the robots, we do not introduce any shared global coordinate frame. Each robot computes its own maximum likelihood estimate for the poses of every robot in its team, given all measurement data available to it. Further, each robot adds the submaps created by the other robots to its own model of the environment as soon as connections between the robots can be made in the SLAM graph. The modularity of the localization and mapping system gives us the option to run only some of its components on resource-constrained systems like micro aerial vehicles (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 both contribute to and benefit from the joint localization with other robots in a heterogeneous team.

We illustrate exemplary applications of the local and global estimates from our localization and mapping components in the planning and control layer depicted in Figure 2. The local state estimates from our local reference filter constitute valuable input for robot control. By satisfying real-time properties and allowing high output frequencies, they are also suitable for stabilization of highly dynamic systems like multicopters (Schmid, Lutz, Tomić, Mair, & Hirschmüller, 2014a). We keep these system-critical components decoupled from our higher-level modules and thus do not feed any estimates from global optimization back into the filter. Combining its estimates with depth data from stereo vision allows us to realize local path planning with fast obstacle avoidance. Our multi-robot SLAM system runs at a slower rate than the filter, providing online estimates suitable for global path and exploration planning. In addition, in a multi-robot setup, the availability of a joint map and pose estimates for all robots supplies the foundation for coordinated cooperative action.

4 | LOCAL REFERENCE FILTER

The local reference filter is a loosely coupled vision-aided inertial navigation system (LR-VINS) fusing delta poses from a stereo odometry system with high-frequency data from an IMU. On our ground-based rovers, we additionally integrate high-frequency measurements of the wheel encoders in form of velocity measurements. The LR-VINS estimates 6D position and orientation, linear velocities as well as sensor biases for the accelerometers and gyroscopes employing an Extended Kalman Filter (EKF). The EKF allows to meet hard real-time constraints imposed by control systems. It has been shown that position (x, y, z) and heading angle (*yaw*) are unobservable within a VINS, that is, their errors and the corresponding estimated uncertainties of unobservable modes rise unbounded (Weiss, 2012). This property is challenging for EKF-based VINS: It leads to inconsistencies in global estimation and can, further, cause numerical issues in long-term operation. Inconsistencies rise with the unobservable *yaw* error (Bailey, Nieto, Guivant, Stevens, & Nebot, 2006) and, further, due to spurious observability caused by changing linearization points (Li & Mourikis, 2013).

These findings motivated us to split state estimation by observability: observable modes are estimated within the filter, where convergence to their real values can be expected. In contrast, unobservable modes are only locally estimated within the filter but furthermore globally optimized in the SLAM graph (see Section 6). The local reference filter (Schmid et al., 2014b) was shown to enable consistent, long-term stable, real-time state estimation for unobservable systems. The approach defines a local state reference consisting at least of the unobservable system states. We periodically switch the reference of the filter into such a local reference frame, that is, we change its reference system, transforming all states and their corresponding covariances. In a VINS, by definition, the uncertainties of position and *yaw* of the reference frame drop to zero, all other uncertainties are reduced relative to the new reference. Relative measurements (as for example a delta pose) in a global frame can be turned into absolute measurements in a local frame, rendering the estimation locally observable. Thereby, the globally unbounded uncertainty for unobservable system modes becomes locally bounded. In this way, the consistency of the filter is improved. The global state is then computed within the graph optimization process, which can improve global consistency through relinearization. Even though the effect of inconsistency in the EKF was shown to be drastically reduced by the LR-VINS, spurious observability can still occur if the local reference is not measurable at all times. The findings of Hesch et al. (2014) to remove spurious observability could also be applied to the LR-VINS to further improve its consistency, if necessary.

In our localization and mapping framework, we define the local reference as the origin of a submap. The switch is actively triggered by the mapping system, as described in Section 5.1.2. For a better understanding of our framework, we summarize the general principles of the vision-based keyframe INS (Schmid, Ruess, Suppa, & Burschka, 2012) and the local reference-VINS (LR-VINS) algorithm (Schmid et al., 2014b) in the following.

4.1 | Vision-based keyframe inertial navigation

In our VINS, we (double) integrate high frequency acceleration and gyroscope measurements of the IMU within the strap down algorithm resulting in the direct system state $\boldsymbol{x}$. Within the EKF, we estimate the indirect state $\boldsymbol{\delta}$, representing the errors of the direct state. Both states are defined as follows:

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{v} \\ \boldsymbol{q} \\ \boldsymbol{b}_a \\ \boldsymbol{b}_\omega \end{pmatrix} \in \mathbb{R}^{16}, \qquad \boldsymbol{\delta} = \begin{pmatrix} \delta\boldsymbol{p} \\ \delta\boldsymbol{v} \\ \delta\boldsymbol{\sigma} \\ \delta\boldsymbol{b}_a \\ \delta\boldsymbol{b}_\omega \end{pmatrix} \in \mathbb{R}^{15} \tag{1}$$

The direct state includes, in corresponding order, the IMU position, velocity and orientation quaternion relative to the current local navigation frame as well as the IMU accelerometer and gyroscope biases in the IMU frame. The indirect state corresponds to the direct state errors, with attitude errors being minimally parameterized as a three-dimensional orientation vector. We apply the standard INS error propagation equations within the EKF prediction step to calculate error uncertainties. The use of the IMU data preintegration model (Forster et al., 2017; Lupton & Sukkarieh, 2012) could improve linearity of the propagation process and eliminate initial state conditions. Nevertheless, besides vision, we also include high-frequency wheel odometry as velocity measurements, which would complicate preintegration between camera frames. However, we consider the integration of the model as a topic for future work.

Our visual odometry algorithm (Hirschmüller et al., 2002) provides a relative transformation measurement from a keyframe in the past to the last captured image with the according measurement noise. We realize a locally drift-free estimation by re-referencing a keyframe from the past. In total, we keep a history of $n = 5$ keyframes to improve robustness. To represent the keyframes in our local reference filter, we augment the system (error) state by the IMU (error) pose at the exact capture time of each camera image. Referencing the corresponding augmented states, we process the delta pose measurements and compensate for measurement delays introduced by the vision pipeline. On our systems, the delay is approx. 250 ms. Thus the delay-compensated state estimate can be directly used for control.

4.2 | State augmentation, marginalization, and reference switching

The processing of delta pose measurements requires state cloning and marginalization. Both processing steps, as well as the switching of the filter into a new local reference frame, can be included into the prediction equation of a Kalman Filter. State augmentation is the general form of state cloning and can be written by introducing a state augmentation function $\boldsymbol{g}$ that builds the new state vector $\bar{\boldsymbol{x}}_k$ at time step k consisting of the current state $\boldsymbol{x}_k$ and the newly augmented state $\boldsymbol{x}_{\text{aug}}$ (in our case a 6D pose). For simplicity we derive the state and covariance transformations in direct state representation. The corresponding transformations for the indirect state can be calculated analogously:

$$\bar{\boldsymbol{x}}_k = \begin{pmatrix} \boldsymbol{x}_k \\ \boldsymbol{x}_{\text{aug}} \end{pmatrix} = \boldsymbol{g}(\boldsymbol{x}_k, \boldsymbol{z}_k) \tag{2}$$

where $\boldsymbol{z}_k$ is a sensor measurement disturbed by additive white Gaussian noise with covariance $\boldsymbol{R}_k$. The filter covariance $\bar{\boldsymbol{P}}_k$ for the new state vector is calculated using the Jacobian of $\boldsymbol{g}$, evaluated at the current state estimate $\hat{\boldsymbol{x}}_k$:

$$\begin{aligned}\bar{\boldsymbol{P}}_k &= \left(\frac{\partial \bar{x}_k}{\partial x_k}\,|_{x_k=\hat{x}_k}\right)\boldsymbol{p}_k\left(\frac{\partial \bar{x}_k}{\partial x_k}\,|_{x_k=\hat{x}_k}\right)^T + \left(\frac{\partial \bar{x}_k}{\partial z_k}\,|_{x_k=\hat{x}_k}\right)\boldsymbol{R}_k\left(\frac{\partial \bar{x}_k}{\partial z_k}\,|_{x_k=\hat{x}_k}\right)^T \\ &= \boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T + \boldsymbol{T}_k\boldsymbol{R}_k\boldsymbol{T}_k^T\end{aligned} \quad (3)$$

where we define $\boldsymbol{A}_k$ as the augmentation matrix and $\boldsymbol{T}_k$ as the noise transformation matrix. For stochastic cloning (Roumeliotis & Burdick, 2002), the augmentation matrix has exactly one **1** per row and the noise transformation matrix vanishes.

We can apply a regular Kalman filter prediction step to the augmented state, which results for the covariance prediction in:

$$\bar{\boldsymbol{P}}_{k+1} = \Phi_k\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T\Phi_k^T + (\boldsymbol{G}_k\boldsymbol{Q}_k\boldsymbol{G}_k^T + \boldsymbol{T}_k\boldsymbol{R}_k\boldsymbol{T}_k^T) \quad (4)$$

where the augmented system matrix Φ_k is an identity matrix of corresponding size with the original system matrix in the upper left corner. The augmented noise propagation matrix $\boldsymbol{G}_k$ is a zero matrix of corresponding size with the original noise propagation matrix in the top rows. $\boldsymbol{Q}_k$ corresponds to the covariance of the system noise.

Analog to Equation (2), we can define a transformation function $\boldsymbol{f}$ that transforms the system states at time step $k + 1$ (including all augmentations) into the new reference frame defined by the augmented state and, at the same time, removes the augmented reference state from the filter:

$$\bar{\bar{\boldsymbol{x}}}_{k+1} = \boldsymbol{f}(\bar{\boldsymbol{x}}_{k+1}) \quad (5)$$

With $\boldsymbol{S}_k$ being the Jacobian of $\boldsymbol{f}$ evaluated at the current state estimate $\hat{\boldsymbol{x}}_{k+1}$, we calculate the transformed state covariance $\bar{\bar{\boldsymbol{P}}}_{k+1}$ as follows:

$$\begin{aligned}\boldsymbol{S}_k &= \frac{\partial \bar{\bar{x}}_{k+1}}{\partial \bar{x}_{k+1}}\,|_{\bar{x}_{k+1}=\hat{\bar{x}}_{k+1}} \\ \bar{\bar{\boldsymbol{P}}}_{k+1} &= \boldsymbol{S}_k\bar{\boldsymbol{P}}_{k+1}\boldsymbol{S}_k^T = \boldsymbol{S}_k\Phi_k\boldsymbol{A}_k\boldsymbol{P}_k\boldsymbol{A}_k^T\Phi_k^T\boldsymbol{S}_k^T + \boldsymbol{S}_k(\boldsymbol{G}_k\boldsymbol{Q}_k\boldsymbol{G}_k^T + \boldsymbol{T}_k\boldsymbol{R}_k\boldsymbol{T}_k^T)\boldsymbol{S}_k^T\end{aligned} \quad (6)$$

Equation (6) can be rewritten as:

$$\bar{\bar{\boldsymbol{P}}}_{k+1} = \tilde{\Phi}_k\boldsymbol{P}_k\tilde{\Phi}_k^T + \tilde{\boldsymbol{G}}_k\tilde{\boldsymbol{Q}}_k\tilde{\boldsymbol{G}}_k^T \quad (7)$$

with $\tilde{\Phi}_k$ as the modified system matrix, $\tilde{\boldsymbol{Q}}_k$ as a combination of $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$, and $\tilde{\boldsymbol{G}}_k$ as the corresponding noise propagation matrix. In this form, Equation (7) has exactly the form of a Kalman Filter prediction step, except for the potentially non-quadratic shape of $\tilde{\Phi}_k$.

We exploit the square root UD filter for our implementation: The covariance matrix is kept in decomposed form as $\boldsymbol{P} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$ with $\boldsymbol{U}$ as unit upper triangular matrix and $\boldsymbol{D}$ as diagonal matrix. This decomposition guarantees the symmetry and positive definiteness properties of the covariance matrix and improves numerical stability. While state augmentation and marginalization is trivial in direct covariance representation, it is not directly obvious in $\boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$ representation. Considering for example marginalization, rows of $\boldsymbol{U}$ are removed, destroying the quadratic form of $\boldsymbol{U}$. Nevertheless, the size of $\boldsymbol{D}$ is still quadratic. The triangularization process for covariance propagation in square root UD form restores the quadratic shape of $\boldsymbol{U}$ and the corresponding quadratic shape of $\boldsymbol{D}$. Therefore, it is convenient to formulate covariance manipulations as propagation.

4.3 | Reference switching for VINS

Reference switching can be formulated by defining the function $\boldsymbol{f}$ of Equation (5) as an ordinary frame transformation. In the following, we use subscripts for vectors to indicate their frame and superscripts to indicate the frame they are expressed in. The new reference frame N_{x+1} is defined by an arbitrary augmented pose with a position $\boldsymbol{P}_{N_x+1}^{N_x}$ expressed in the current frame N_x and its corresponding quaternion $\boldsymbol{q}_{N_x+1}^{N_x}$. With these frame definitions, we introduce an example state $\boldsymbol{x}$ consisting of our IMU state and one single augmented pose defining a new frame with index N_{x+1}. All other state variables are expressed in the current local frame N_x or in the IMU body frame B, respectively. Omitting the time index k, the direct and its corresponding indirect state are given by:

$$\boldsymbol{x} = \begin{pmatrix}\boldsymbol{p}_B^{N_x}\\ \boldsymbol{v}_B^{N_x}\\ \boldsymbol{q}_B^{N_x}\\ \boldsymbol{b}_a^B\\ \boldsymbol{b}_\omega^B\\ \boldsymbol{p}_{N_x+1}^{N_x}\\ \boldsymbol{q}_{N_x+1}^{N_x}\end{pmatrix}, \quad \delta = \begin{pmatrix}\delta\boldsymbol{p}_B^{N_x}\\ \delta\boldsymbol{v}_B^{N_x}\\ \delta\boldsymbol{\sigma}_B^{N_x}\\ \delta\boldsymbol{b}_a^B\\ \delta\boldsymbol{b}_\omega^B\\ \delta\boldsymbol{p}_{N_x+1}^{N_x}\\ \delta\boldsymbol{\sigma}_{N_x+1}^{N_x}\end{pmatrix} \quad (8)$$

For simplicity, we split the definition of the transformation function $\boldsymbol{f}$ into transformations for positions, orientations, and velocities of an arbitrary frame m as well as for the IMU biases:

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{cases}\boldsymbol{P}_m^{N_x+1} &= \boldsymbol{C}_{N_x}^{N_x+1}(\boldsymbol{P}_m^{N_x} - \boldsymbol{p}_{N_x+1}^{N_x})\\ \boldsymbol{Q}_m^{N_x+1} &= \boldsymbol{q}_{N_x}^{N_x+1}\boldsymbol{Q}_m^{N_x}\\ \boldsymbol{v}_m^{N_x+1} &= \boldsymbol{C}_{N_x}^{N_x+1}\boldsymbol{v}_m^{N_x}\\ \boldsymbol{b}^B &= \boldsymbol{b}^B \quad \text{for all IMU biases}\end{cases} \quad (9)$$

The matrix $\hat{\boldsymbol{C}}_{N_x}^{N_x+1}$ rotates a vector from frame N_x to N_{x+1} and is calculated from $\boldsymbol{Q}_{N_x}^{N_x+1}$. A frame switch does not change the reference frame of the estimated IMU biases, the corresponding transformation part in $\boldsymbol{f}$ is therefore the identity transform. Our direct state vector can be transformed using Equation (9).

For the transformation of the covariances of the indirect state, we use the relation between true value, estimated value and error as $\boldsymbol{y} = \hat{\boldsymbol{y}} - \delta\boldsymbol{y}$ for vectors and as $\boldsymbol{C}_m^{N_x} = (\boldsymbol{I} - \lfloor\delta\boldsymbol{\sigma}_m^{N_x}\rfloor)\hat{\boldsymbol{C}}_m^{N_x}$ for rotations, where $\boldsymbol{I}$ is the 3×3 identity matrix and $\lfloor\ldots\rfloor$ the skew operator of a 3×1 vector. Our direct state vector holds quaternions corresponding to a transformation $\hat{\boldsymbol{C}}_m^{N_x}$ (vector transformation from arbitrary frame m to frame N_x), whereas we need the inverse transformation for state switching: $\boldsymbol{C}_{N_x}^m = \hat{\boldsymbol{C}}_{N_x}^m(\boldsymbol{I} + \lfloor\delta\boldsymbol{\sigma}_m^{N_x}\rfloor)$. With these error definitions, small-angle assumptions and Equation (9), we find the linearized transformations for the indirect states as:

$$
\begin{aligned}
\delta \boldsymbol{P}_m^{N_{x+1}} &= \hat{\boldsymbol{C}}_{N_x}^{N_{x+1}} \left(\delta \boldsymbol{p}_m^{N_x} - \delta \boldsymbol{P}_{N_{x+1}}^{N_x} + \left\lfloor \hat{\boldsymbol{p}}_m^{N_x} - \hat{\boldsymbol{p}}_{N_{x+1}}^{N_x} \right\rfloor \delta \boldsymbol{\sigma}_{N_{x+1}}^{N_x} \right) \\
\delta \boldsymbol{\sigma}_m^{N_{x+1}} &= \hat{\boldsymbol{C}}_{N_x}^{N_{x+1}} \left(\delta \boldsymbol{\sigma}_m^{N_x} - \delta \boldsymbol{\sigma}_{N_{x+1}}^{N_x} \right) \\
\delta \boldsymbol{v}_m^{N_{x+1}} &= \hat{\boldsymbol{C}}_{N_x}^{N_{x+1}} \left(\delta \boldsymbol{v}_m^{N_x} + \lfloor \hat{\boldsymbol{v}}_m^{N_x} \rfloor \delta \boldsymbol{\sigma}_{N_{x+1}}^{N_x} \right) \\
\delta \boldsymbol{b}^B &= \delta \boldsymbol{b}^B \quad \text{for all IMU biases}
\end{aligned}
\tag{10}
$$

We analyze the effect of state switching for a covariance prediction given in Equation (6) for time step k. Let us assume for simplicity and without loss of generality an identity prediction matrix Φ_k, an identity augmentation matrix $\boldsymbol{A}_k$ and the state defined in Equation (8). We find the pure switching matrix $\boldsymbol{S}_k^{'}$ (without marginalization of the reference state) from Equation (10) by linearization around δ using the current estimate $\hat{\boldsymbol{x}}$ of our state vector:

$$
\begin{array}{c|ccccccc}
 & \delta \boldsymbol{p}^{N_x} & \delta \boldsymbol{v}^{N_x} & \delta \boldsymbol{\sigma}^{N_x} & \delta \boldsymbol{b}_a^B & \delta \boldsymbol{b}_\omega^B & \delta \boldsymbol{p}_{N_{x+1}}^{N_x} & \delta \boldsymbol{\sigma}_{N_{x+1}}^{N_x} \\
\hline
\delta \boldsymbol{p}^{N_{x+1}} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & -\hat{\boldsymbol{C}}_{N_x}^{N_x+1} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} \left\lfloor \hat{\boldsymbol{p}}^{N_x} - \hat{\boldsymbol{p}}_{N_{x+1}}^{N_x} \right\rfloor \\
\delta \boldsymbol{v}^{N_{x+1}} & \boldsymbol{0} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} \left\lfloor \hat{\boldsymbol{v}}^{N_x} \right\rfloor \\
\delta \boldsymbol{\sigma}^{N_{x+1}} & \boldsymbol{0} & \boldsymbol{0} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & -\hat{\boldsymbol{C}}_{N_x}^{N_x+1} \\
\delta \boldsymbol{b}_a^B & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\
\delta \boldsymbol{b}_\omega^B & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\
\delta \boldsymbol{p}_{N_{x+1}}^{N_{x+1}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} - \hat{\boldsymbol{C}}_{N_x}^{N_x+1} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} \left\lfloor \hat{\boldsymbol{p}}_{N_{x+1}}^{N_x} - \hat{\boldsymbol{p}}_{N_{x+1}}^{N_x} \right\rfloor \\
\delta \boldsymbol{\sigma}_{N_{x+1}}^{N_{x+1}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \hat{\boldsymbol{C}}_{N_x}^{N_x+1} - \hat{\boldsymbol{C}}_{N_x}^{N_x+1}
\end{array}
= \boldsymbol{S}_k'
\tag{11}
$$

The elements in the rows corresponding to the state defining frame N_{x+1} cancel out. As during propagation the covariance matrix is multiplied from the left by $\boldsymbol{S}_k$ and from the right by $\boldsymbol{S}_k^T$, zero rows will cancel out all correlations of the corresponding state. Therefore, the covariance matrix will be rank deficient. By defining $\boldsymbol{S}_k$ as $\boldsymbol{S}_k^{'}$ without its zero rows, the switching state is marginalized out during the switching operation.

After the transformation of the covariance matrix, we can apply Equation (9) on the direct state. By definition, $\boldsymbol{p}_{N_{x+1}}^{N_{x+1}}$ is zero and $\boldsymbol{q}_{N_{x+1}}^{N_{x+1}}$ represents the identity rotation. For simplicity, we analyzed a full frame switch (including the entire orientation). As *roll* and *pitch* angles are observable, in our implementation, we do a partial frame switch for position and *yaw* only, keeping the navigation frame horizontally aligned to the gravity vector. This is realized by manipulating the rotation matrix $\boldsymbol{C}_{N_x}^{N_x+1}$ to include only *yaw* components. Please consider that for the implementation of a full state switch, including *roll* and *pitch*, the gravity vector has to be included into the state, similar to the work of Lupton and Sukkarieh (2012).

5 | 3D MAPPING AND MAP MATCHING

In this section, we describe our approach to create local and global 3D maps from stereo data and present our method to match these local maps to generate intra- and inter-robot loop-closure constraints for our multi-robot graph SLAM discussed in detail in Section 6.

5.1 | Map creation and composition

To create global 3D maps, we first create local submaps and then compose them, globally optimized, into a joint map. A submap is defined by a local reference frame, that is, 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 the submap origins to our SLAM graph for global optimization.

5.1.1 | Submap generation

To create 3D submaps, we aggregate depth data and obstacle classification results along the trajectories estimated by our local reference filter. Each submap has two application-dependent 3D representations: First, we create a 3D point cloud at a resolution of 5 cm, with color information and a binary obstacle classification for each point. The latter results from an earlier step in our mapping pipeline and is computed anyway for local obstacle avoidance by a stereo error-adaptive terrain classification algorithm (Brand et al., 2014). We use the point cloud representation for map visualization and map matching, see Section 5.2. Second, we integrate the depth data into a probabilistic voxel space representation. For this, we employ the freely available open-source OctoMap library (Hornung, Wurm, Bennewitz, Stachniss, & Burgard, 2013) and create octrees with a resolution of 10 cm as a trade-off between computational load and required precision. We currently use the resulting joint maps for autonomous exploration based on expected information gain and plan to employ them for whole-body path planning in future work. Both applications require an explicit distinction between occupied, free and unknown space, which is not available in simple point cloud representations.

5.1.2 | Submap partitioning

We aggregate the dense 3D data into submaps assuming that our filter estimates on each robot are locally sufficiently accurate. To satisfy this assumption, we create a new submap once the filter's estimated uncertainty grows above map resolution. Applying a threshold of 0.1 m on the standard deviation of the robot's position, we not only ensure a limited drift within each submap but, by triggering a frame switch, also a limited accumulated error in the local reference filter. In addition, we employ an empirically determined threshold of 3.5 m accumulated driven distance, restricting the size of the individual submaps to limit their memory and processing time requirements during exchange and matching. As the localization error within a submap thus is limited, we perform global corrections only between the submaps of one or multiple robots. This approach allows us to reduce the high-bandwidth 3D data by aggregating it locally into submaps and exchange only this aggregated data between robots in a multi-robot system, thus saving 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.

5.1.3 | Global map composition

To generate joint global maps, 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 for visualization or on demand for path and exploration planning. As the computational effort of this merging step grows with the number of submaps, we cache the combined map, except for the currently active one and thus frequently changing submap, and invalidate this cache only on significant changes of the respective estimated submap origins, that is, after new loop closures. For future work we plan to limit the exchange of submaps as well as their composition into a global map to an application-dependent area of interest, as for example proposed by Koch and Lacroix (2016). In addition, we plan to merge submaps after successful matching, similar to Mohanarajah et al. (2015), and to remove deprecated submaps when the same area has been exhaustively mapped again more recently.

5.2 | Submap matching

We perform a pairwise matching of the 3D structure of submaps to compute relative transformations accompanied by uncertainty estimates between their origins and thus generate intra- as well as inter-robot loop-closure constraints. As input data, our submap matching process receives the submaps from all robots in a multi-robot team as well the latest pose and uncertainty estimates of their origins from the graph SLAM component of the robot on which it is running.

In this study, we improved and extended our map matching concepts, which we first presented in Brand et al. (2015) and applied to multi-robot systems in Schuster et al. (2015). As an extension of our previous work, 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 thus can generate only small numbers 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 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, 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, that is, 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 each robot's observable states, including its *roll* and *pitch* angles, are

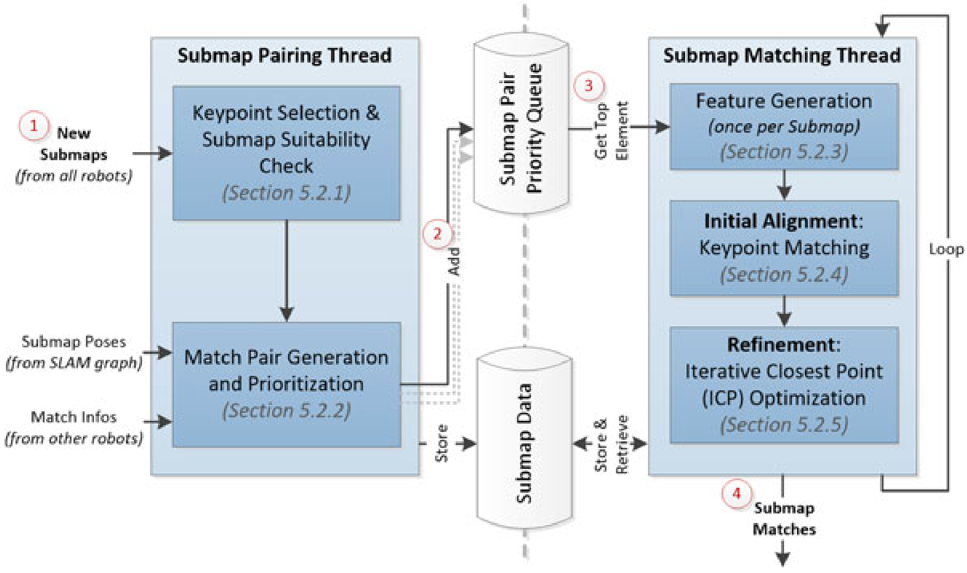
Submap Pairing Thread
Keypoint Selection & Submap Suitability Check (Section 5.2.1)
Match Pair Generation and Prioritization (Section 5.2.2)
1 New Submaps (from all robots)
Submap Poses (from SLAM graph)
Match Infos (from other robots)
2 Add
Store
Submap Pair Priority Queue
Submap Data
3 Get Top Element
Store & Retrieve
Submap Matching Thread
Feature Generation (once per Submap) (Section 5.2.3)
Initial Alignment: Keypoint Matching (Section 5.2.4)
Refinement: Iterative Closest Point (ICP) Optimization (Section 5.2.5)
Loop
4 Submap Matches

FIGURE 3 Submap matcher architecture with two threads for parallel execution of pairing and matching (numbers 1-4 indicate typical order of main data flow) [Color figure can be viewed at wileyonlinelibrary.com]

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 with respect to *roll* and *pitch* leads to an increased number of valid matches, as we discuss in the following (Sections 5.2.4 and 5.2.5) and demonstrate in our experimental evaluation (Section 7.2).

In Figure 3, we present the architecture of our submap matching module. It runs 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 implemented a similar type of parallelization across machines by executing matcher processes on multiple robots in parallel. They share their results to avoid duplicate work as well as double-counting of successful matches.

We base the pairwise matching between submaps on their point clouds and follow a two-step approach: First, we compute potential initial alignments through 3D feature matching on obstacle points and select the best model. Second, we perform an ICP step on the full point cloud for refinement and match uncertainty estimation. In Figure 4, we show a submap match, depicting the keypoints and correspondences used during initial alignment as well as the final transformation after refinement. We will explain the individual steps of pairing and matching in the following subsections.

5.2.1 | Keypoint selection and submap suitability check

To compute 3D features, we select distinctive keypoints based on our precomputed obstacle classification in the point clouds. We employ a 3D voxel grid filter with a bin size of 0.1 m on them to reduce the computational effort of the subsequent steps. As we argue in Brand et al. (2014) and Brand et al. (2015), 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. To distinguish these valuable arrangements of obstacles from simple structures like straight walls or ambiguous ones like single stones, we check the distribution of obstacle points within each submap. Therefore, we fit a 3D line model to the keypoints using random sample consensus (RANSAC) and check that at least 10 keypoints are further away from it than an empirically determined threshold of 0.5 m. We thereby replace the heuristic based on bounding boxes in the *xy*-plane described in Brand et al. (2015) with a more general approach.

In addition, as a first step, we filter out submaps if their point cloud contains less than 1,000 points or less than 100 obstacle points. These suitability checks allow us to dismiss uninformative or ambiguous submaps before even including them into the matching process. As the keypoints for each submap are independent of all other submaps, we compute them just once and store them together with the submap data.

5.2.2 | 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 preselection 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 $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ 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 later 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:

$$overlap'_t = overlap_t + 2 \cdot \Delta\sigma_t \quad \text{for} \quad t \in \{x, y\} \tag{12}$$

$$overlap'_{x,y} = overlap'_x \cdot overlap'_y \tag{13}$$

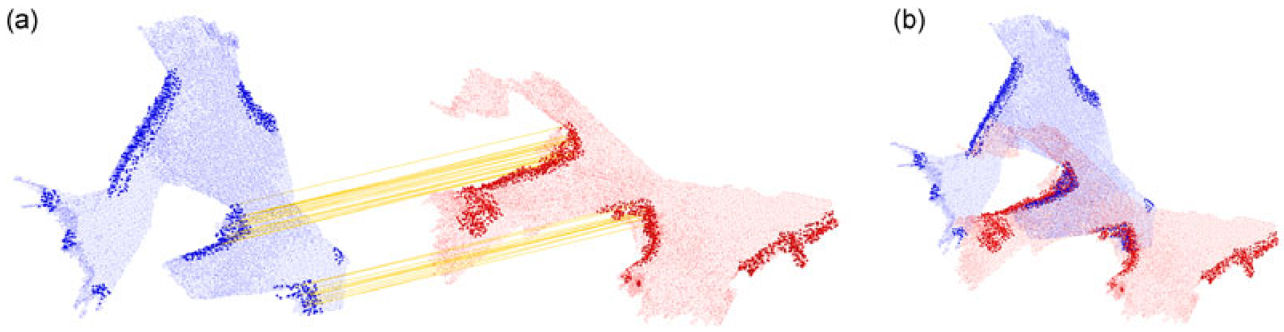

FIGURE 4 Visualization of the initial alignment (Section 5.2.4) and refinement (Section 5.2.5) steps of a match between submaps created by LRU1 (left, blue) and LRU2 (right, red) during our multi-robot experiment #2 described in Section 7.3. The larger points indicate keypoints located on obstacles (in this case large artificial rocks, similar to those in Figure 7b) that have been used to compute correspondences for the initial alignment. a, Filtered correspondences used for initial alignment between the two submaps. b, Transformation after refinement [Color figure can be viewed at wileyonlinelibrary.com]

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 study: $\Delta\sigma_t \approx \sqrt{|\Sigma^{\boldsymbol{s}_i}_{t,t} - \Sigma^{\boldsymbol{s}_j}_{t,t}|}$ with $\Sigma^{\boldsymbol{s}_i}_{t,t}$ and $\Sigma^{\boldsymbol{s}_j}_{t,t}$ referring to the variances in the dimension $t \in \{x, y, z\}$ of the poses of the origins of $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ 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 $overlap'_{x,y} > 2\,\text{m}^2$ in order for a pair of submaps to be added to the match queue, and in addition exclude successive submaps from the same robot r, that is, $\boldsymbol{s}^r_i$ and $\boldsymbol{s}^r_{i+1}$. Furthermore, we filter out pairs for which the estimated pose uncertainty is below the precision of the matcher. We approximate the latter by the stereo error, which grows quadratically with the distance Δd to the cameras and is significantly large for our camera setups (Brand et al., 2014). It can be estimated as $e_z = e_p \frac{\Delta d^2}{f \cdot b}\sqrt{2}$ and is 0.12 m at a maximum view distance of 4 m for our robots' stereo camera setup with focal length $f = 1,080\,\text{px}$, baseline $b = 0.09\,\text{m}$ and a pixel error of $e_p = 0.5\,\text{px}$. We then compare it to the approximated mean translational standard deviation by checking the following constraint:

$$\frac{\Delta\sigma_x + \Delta\sigma_y + \Delta\sigma_z}{3} < e_z \qquad (14)$$

We add all selected pairs of submaps to a priority working queue, prioritizing them to try the most promising pairs first. We thus rank them according to:

$$score = (\alpha \cdot overlap'_{x,y}) + (\Delta\sigma_x + \Delta\sigma_y + \Delta\sigma_z) \qquad (15)$$

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.

5.2.3 | 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 later match attempts. Thus, we can avoid unnecessary computation in case a submap is never included in a match attempt. To characterize geometric features in the environment, we chose CSHOT feature descriptors (Tombari et al., 2011) as they exhibit a good tradeoff between performance and computational complexity (Alexandre, 2012). Based on SHOT descriptors (Tombari, Salti, & DiStefano, 2010), they characterize the 3D information of the local spatial neighborhood through unique signatures of histograms of orientations with respect to local 3D reference frames. They are rotational invariant and robust to noise and clutter. The CSHOT extension additionally includes color texture information, our robots are, however, only equipped with black&white cameras. It is a topic for future work to analyze its benefits, in particular for heterogeneous multi-robot systems involving different camera setups.

5.2.4 | 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]. 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 KDTrees 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 & 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, from $\boldsymbol{s}_i$ to $\boldsymbol{s}_j$ and vice versa to obtain more hypotheses and make them independent of the order of submaps. 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 to determine the most likely transformation, including a 3D orientation, associated with it. Compared to a closed-form 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 submaps' point cloud data.

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. We thus 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 & 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 $\boldsymbol{p}^0_i$ and $\boldsymbol{p}^1_i$ from the cluster of each of the two submaps $i \in \{0, 1\}$. We then use

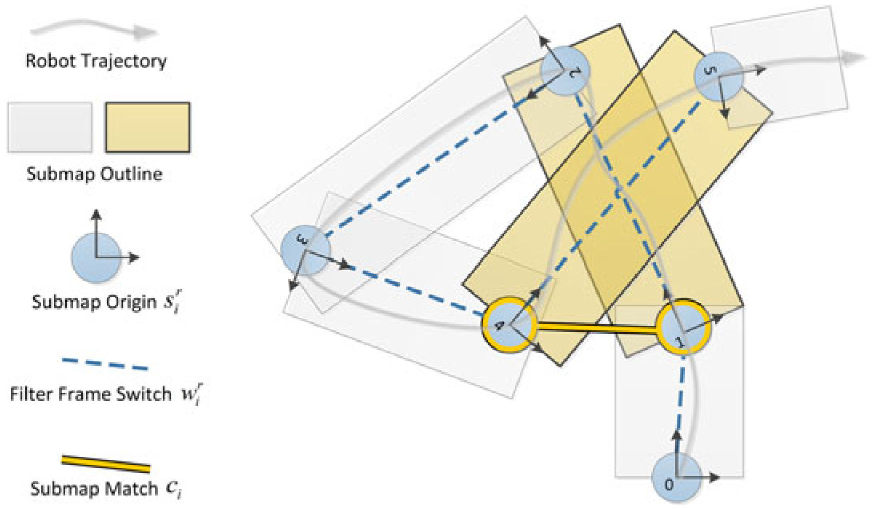

FIGURE 5 Schematic of SLAM graph with submap origins and bounding boxes. The overlapping highlighted rectangles represent submaps that match, resulting in a loop-closure constraint that is added to the graph [Color figure can be viewed at wileyonlinelibrary.com]

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:

$$\left.\begin{aligned} \boldsymbol{z} &= (0, 0, 1) \\ \bar{\boldsymbol{p}}_i &= \left(\boldsymbol{p}_i^0 + \boldsymbol{p}_i^1\right)/2 \\ \Delta\boldsymbol{p}_i &= \left(\boldsymbol{p}_i^1 - \bar{\boldsymbol{p}}_i\right) - \left(\boldsymbol{p}_i^0 - \bar{\boldsymbol{p}}_i\right) \\ \Delta\boldsymbol{p}_i^{'} &= \Delta\boldsymbol{p}_i - \left(\Delta\boldsymbol{p}_i \cdot \boldsymbol{z}\right)\cdot\boldsymbol{z} \\ \boldsymbol{C}_i &= \left[(\Delta\boldsymbol{p}_i^{'})^T, (\Delta\boldsymbol{p}_i^{'} \times \boldsymbol{z})^T, \boldsymbol{z}^T\right]^T \end{aligned}\right\} \text{ for } i \in \{0, 1\} \tag{16}$$

We normalize the points by subtracting their mean $\bar{\boldsymbol{p}}_i$ and compute the vector $\Delta\boldsymbol{p}_i$ between them. We then adapt it to $\Delta\boldsymbol{p}_i^{'}$ such 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 $\boldsymbol{C}_i$. With this, we compute the transformation hypothesis between the submaps with rotation matrix $\boldsymbol{C}$ and translation $\boldsymbol{t}$ as:

$$\boldsymbol{C} = \boldsymbol{C}_0\cdot(\boldsymbol{C}_1)^T \tag{17}$$

$$\boldsymbol{t} = \bar{\boldsymbol{p}}_0 - \boldsymbol{C}\cdot\bar{\boldsymbol{p}}_1 \tag{18}$$

As the next step, we filter out all hypotheses that exceed the $2\Delta\sigma$ error bounds of the approximated uncertainty between both submaps in any dimension. In addition, for each submap, we check the distribution of keypoints for each hypothesis separately by fitting a line model, similar to the submap suitability check described in Section 5.2.1. We thereby dismiss matches based on features that are not well distributed and thus might be ambiguous with respect to 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.

5.2.5 | Refinement: ICP optimization

As the final step, we perform an ICP optimization on the 3D point clouds of the two submaps. As this optimization 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 employ 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 with respect to an alignment of the ground planes. While such areas lack robust 3D features for matching, they work well for an ICP.

We also restrict the refinement step to 4D by removing the *roll* and *pitch* angles from the ICP optimization problem. Similar to our adapted Hough3D voting, the early incorporation of prior knowledge to reduce the dimensionality of the optimization directs the optimizer toward 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 7.2 that our 4D matching thus yields a larger number of map matches, that is, 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 $2\Delta\sigma$ error bounds of the latest SLAM estimates. To weigh 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 5, we present a sketch showing the integration of a match of two overlapping submaps into the SLAM graph.

6 | INCREMENTAL MULTI-ROBOT GRAPH SLAM

In this section, we introduce our multi-robot SLAM graph topology for the decoupled integration of local reference filter estimates. While we first presented this graph topology in our conference paper

Schuster et al. (2015), in this study we provide a more detailed description and extended discussion of its properties.

We employ a factor graph (Kschischang, Frey, & Loeliger, 2001) to formulate the SLAM problem in a graphical model, following the formalization used by Kaess et al. (2012). A factor graph is a bipartite graph $G = (\mathcal{F}, \Theta, \mathcal{E})$ with factor nodes $\boldsymbol{f}_i \in \mathcal{F}$, variable nodes $\theta_i \in \Theta$ 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 $\boldsymbol{f}(\Theta) = \prod_i \boldsymbol{f}_i(\Theta_i)$ with Θ_i denoting the set of variables adjacent to the factor $\boldsymbol{f}_i$. In our SLAM context, the variable nodes θ_i represent the 6D poses of interest (robot poses, submap origins, landmarks). The factors $\boldsymbol{f}_i$ represent constraints given through measurements or "virtual measurements," that is, estimates from other modules like our local reference filter. In general, factors $\boldsymbol{f}_i$ can connect an arbitrary number of variable nodes θ_i. In this study, we only refer to binary measurement factors, except for a single unary prior factor per graph.

The goal of graph optimization is to find an assignment of variables Θ^* that maximizes this function, that is, $\Theta^* = \arg\max_\Theta \boldsymbol{f}(\Theta)$. Under the assumption of zero-mean Gaussian noise, this maximization problem can be formulated as a nonlinear least-squares minimization on the differences between the measurement functions $\boldsymbol{h}_i$ and the actual measurements $\boldsymbol{z}_i$. 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 Σ_i denoting the measurement noise covariance matrices:

$$\arg\min_\Theta(-\log \boldsymbol{f}(\Theta)) = \arg\min_\Theta \frac{1}{2} \sum_i ||\boldsymbol{h}_i(\Theta_i) - \boldsymbol{z}_i||^2_{\Sigma_i} \qquad (19)$$

Graph SLAM systems can be structured into two parts, a front-end and a back-end. While the back-end deals with the optimization problem, that is, the aforementioned minimization of a nonlinear 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, Kümmerle, Stachniss, & Burgard, 2011). In the following, 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.

6.1 | SLAM front-end: Multi-robot graph topology

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

- Robot poses $\{\boldsymbol{x}_i^r\}$: $\boldsymbol{x}_i^r$ represents the ith pose of the robot $r \in \{0, \ldots, R-1\}$. We sample the robot poses sparsely by only adding them to the graph if they are connected to another robot's pose or a landmark via a measurement edge (see below).
- Submap poses $\{\boldsymbol{s}_i^r\}$: $\boldsymbol{s}_i^r$ represents the pose of the origin of the ith submap of the robot $r \in \{0, \ldots, R-1\}$.
- Landmark poses $\{\boldsymbol{l}_i\}$ (optional): $\boldsymbol{l}_i$ represents the pose of the ith globally identifiable (robot-independent) landmark.

Each of them represents a 6D pose with Gaussian uncertainty. On each robot r_i that runs a graph optimization module, its first submap pose $\boldsymbol{s}_0^{r_i}$ 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 tradeoff 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 $\boldsymbol{p}_i^r$ for each robot r with respect to the map origin at the latest filter time step t_i^r by combining the output of our local reference filter and graph SLAM. 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 robot's latest filter estimate $\boldsymbol{v}_i^r$:

$$\boldsymbol{p}_i^r = \boldsymbol{s}_j^r \oplus \boldsymbol{v}_i^r \quad \text{with} \quad t_j^r \le t_i^r < t_{j+1}^r \qquad (20)$$

6.1.1 | Intra- and inter-robot measurements

We represent three different types of measurements as factor nodes in our 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 AprilTag markers (Olson, 2011) to the robots in our multi-robot team and detect these in the other robots' camera images, utilizing an open source detector implementation (Kaess, 2013). The quality of pose estimates for planar markers highly varies, in particular depending on view distance and view angle, leading to pose ambiguities (Schweighofer & Pinz, 2006). We therefore perform a worst-case error approximation depending on distance, view angle and camera parameters, which we base on simulations of detections and detection errors as well as their propagation to 6D transformations. This allows us to avoid overconfidence in measurements that could later on degrade the results of graph optimization. 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 the pose estimates from the filters of both robots at the point in time $t_i = t_j$ of the detection. Each robot therefore holds a buffer, implemented as a hashmap for constant time lookup, with its filter estimates to recover this information in case of communication delays or interruptions. The memory requirements of this buffer are negligible compared to the dense 3D data.
- Landmark observations $\{\boldsymbol{o}_i^r\}$: $\boldsymbol{o}_i^r$ represents the transformation between a robot r and a static, globally identifiable landmark. In some of the experiments on the evaluation of our novel graph structure presented in Schuster et al. (2015), we defined and

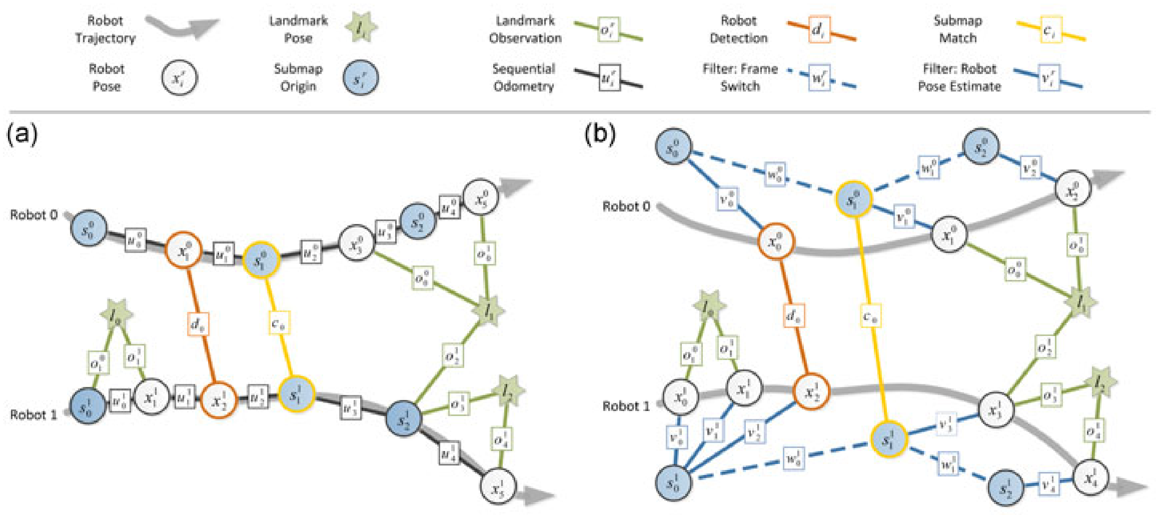

FIGURE 6 Comparison of SLAM graph topologies with robot detections $\boldsymbol{d}_i$, submap matches $\boldsymbol{c}_i$ and landmarks observations $\boldsymbol{o}_i^r$ as inter-robot measurements. In the experiments presented here, we did not use any artificial static global landmarks. (a), Graph topology for sequential odometry measurements with submap origins at robot poses. (b), Novel graph topology with submap origins $\mathbf{s}_i^r$ (local reference frames) separated from robot poses $\mathbf{x}_i^r$

observed such landmarks with the help of AprilTag markers, similar to the robot detections described above. As such landmarks are typically not available in natural environments, we instead aim to exploit the geometric structure of the environment to generate loop closures through map matching. We thus did not utilize any static landmarks in the experiments presented here. However, for planetary exploration, it is reasonable to use static structures, such as a stationary lander (Wedler et al., 2017), as landmarks to improve localization and provide transformations between multiple robots.

- 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 5.2. This can lead to intra- as well as inter-robot loop closures.

In addition, we integrate the estimates of our local reference filters, thereby connecting robot and submap poses. 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., 2011) 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 with respect to other types of measurements. In the following sections, we will discuss and compare the two different graph topologies that we outlined in Figure 6.

6.1.2 | Graph with sequential odometry measurements

The graph topology typically found in SLAM literature sequentially connects robot poses through odometry-like measurements $\boldsymbol{u}_i^r$ between $\boldsymbol{x}_i^r$ and $\boldsymbol{x}_{i+1}^r$, as pictured in Figure 6a. The set of submap origins is thereby a subset of the set of robot poses: $\{\boldsymbol{s}_i^r\} \subseteq \{\boldsymbol{x}_j^r\}$. 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 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 (see Section 4). The filter estimates can depend on each other through filter-internal states like the augmentations made on keyframes. In our previous 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: $\boldsymbol{u}_i^r = \boldsymbol{x}_{i+1}^r \ominus \boldsymbol{x}_i^r$ and $\Sigma_{\boldsymbol{u}_i^r} = \max(\Sigma_{\boldsymbol{x}_{i+1}^r} - \Sigma_{\boldsymbol{x}_i^r},\ \boldsymbol{I}_6 \cdots 10^{-10})$ as an approximation of their Gaussian measurement uncertainty. We thereby 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, we therefore introduce an adaption of the graph topology that allows a more suitable integration of local reference filter estimates.

6.1.3 | Graph with local reference filter estimates

In Figure 6b, we sketched our novel graph topology, as first proposed in Schuster et al. (2015), in which 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 $\boldsymbol{x}_k^r$ that is estimated by the filter with respect to 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 with respect to gravity, as discussed in Section 4. Thus, with this graph topology, they can be dissimilar from the robot poses $\boldsymbol{x}_i^r$ with respect to the *roll* and *pitch* angles, as we indicated in Figure 6b by drawing them separated. Compared to the graph topology for sequential odometry measurements, 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 ($\hat{=}$ 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 antifactors (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. The design decisions are a tradeoff between modularity, computational efficiency and the integration of all available information in a smoothing process. Our explicit decoupling of filter and graph SLAM thereby gives us greater design freedom for both subsystems compared to tightly coupled solutions such as, for example, concurrent filtering and smoothing (Williams et al., 2014).

6.2 | SLAM back-end: Incremental optimization

The computational cost of batch graph optimization grows with the size of the graph and is therefore not suitable for online global optimization. We thus decided to utilize the incremental iSAM2 optimizer (Kaess et al., 2012), which is available as open source software 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 subtrees that are not affected by local loop closures unchanged. In our system, it thus allows for fast average optimization steps on the addition of new measurements and filter estimates, while slower, 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 to reduce the influence of outliers (Agarwal, Tipaldi, Spinello, Stachniss, & Burgard, 2013; Latif, Cadena, & Neira, 2014). We replace the quadratic error term in Equation (19) 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, Fraundorfer, & Pollefeys, 2013). The optimization problem can thus be formulated as an iterative reweighted least-squares minimization with the weights in the kth iteration being computed as

$$\boldsymbol{w}^k = \frac{c^2}{c^2 + \|\boldsymbol{h}_i(\Theta_i^{k-1}) - \boldsymbol{z}_i\|^2_{\Sigma_i}} \tag{21}$$

where the value of the constant c determines the range of the Mahalanobis distances 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.

7 | EXPERIMENTAL EVALUATION

In the following, we present and discuss novel experiments. First, in Section 7.2, we evaluate the impact of our novel 4D map matching optimizations introduced in this study on a total of 53 single-robot datasets. Second, in Section 7.3, we demonstrate the applicability of our full SLAM system in five new experiments, four of them featuring a significantly larger scenario than in Schuster et al. (2015). Third, in Section 8, we present an application of our localization and mapping system in a novel multi-robot autonomous exploration experiment conducted in the rough-terrain environment of a Moon-analogue test site located on a volcano.

In our previous publications, we already evaluated several aspects of our localization and mapping system. For a better overview, we give a short summary of previous experimental evaluations:

- Filter consistency and stability: In Schmid et al. (2014b), we demonstrated the consistency and long-term stability of our local reference filter. For this evaluation, we used a quadcopter as a robot that poses additional challenges due to its inherent instability compared to our rover systems discussed here. We evaluated a simulated 24 h quadrotor 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.
- Accuracy of SLAM pipeline with map matching: We compared our stereo vision-based 6D SLAM system with submap matching to a particle filter-based 3D localization in Brand et al. (2015) and could show an improved 2D localization accuracy of at least 27% in indoor and outdoor scenarios, in addition to estimating three additional degrees of freedom. A comparison of our full SLAM system, as a distributed and suboptimal data fusion method, to full batch optimization, however, remains a topic for future work.
- Impact of SLAM graph topology and heterogeneous multi-robot SLAM: In Schuster et al. (2015), we evaluated the impact of our novel SLAM graph topology on the overall localization accuracy and demonstrated an improvement of 15% on three different datasets compared to a SLAM graph with sequential odometry graph topology as used previously in Brand et al. (2015). In addition, we presented multi-robot experiments with a team of two rovers with heterogeneous camera systems — one was equipped with small-angle lens cameras ($f = 5$ mm) and the other one with wide-angle lenses ($f = 1.28$ mm). Our joint localization and mapping gives an improvement over single-robot SLAM of 32%.
- Applications in Moon-like environment and for single-robot autonomous exploration: In Schuster et al. (2017), we present our success at the SpaceBotCamp 2015 national robotics challenge, at which we demonstrated our mapping system in a Moon-like environment. During the mission, the robot fully autonomously navigated and mapped its surroundings based on predefined waypoints, located therein three known objects and assembled them. In Lehner, Schuster, Bodenmüller, and Kriegel (2017), we applied our global probabilistic voxel-grid maps for information gain-based autonomous single-robot exploration.

7.1 | Robot hardware setup

For our novel experiments, we deployed two of our lightweight rover units (LRUs). Each of these small rough-terrain rover prototypes 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. They are equipped with a pair of Guppy PRO F-125B cameras ($1/3''$ chip size, resolution: $1{,}292 \times 964, f = 5$ mm) on a pan/tilt camera mast and an Xsens MTi-10 IMU in the body. We employ an on-board Spartan 6 LX75 FPGA for dense stereo matching ($1{,}024 \times 508$ px at 14 Hz), all other computation is performed on an Intel quadcore CPU. For a more detailed description of the LRU's hardware and software architecture, see Schuster et al. (2017).

7.2 | Comparison of 4D and 6D map matching

In this section, we present an evaluation of our novel 4D map matching algorithm, comparing it to a 6D matching step used in our previous work (Brand et al., 2015).

7.2.1 | Experimental setup

We evaluate our algorithm in two series of 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 employ the RoverSimulationToolkit (Hellerer, Schuster, & Lichtenheldt, 2016), a multibody physics simulation and high-fidelity visualization featuring virtual sensors like IMUs, color and depth cameras to simulate a model of our lightweight rover unit (LRU). We based our simulated environment on a rough-terrain 3D model of the publicly available 2013 SpaceBotCup challenge arena (Holz & Behnke, 2014; Schadler, Stückler, & Behnke, 2014), featuring a deep ridge, a steep ramp as well as several large rocks.
- Real LRU indoor experiments: The experiments with one of our real LRUs have 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 a ceiling-mounted Vicon tracking system with 14 cameras, covering the complete experimental area of approx. 11 m × 6.5 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 m^2 and 72 m^2 in each of them, respectively. In Figure 7, we give an impression of both experimental setups. While the lab experiments are important to test our algorithm under real conditions, in particular with respect to the sensors' error characteristics, a high-fidelity simulation allows to conduct a larger number of experiments with perfect ground truth

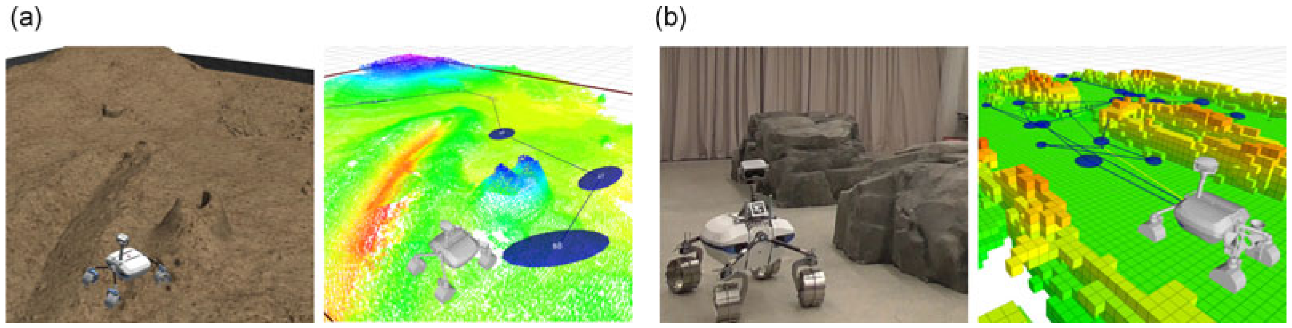

FIGURE 7 Impressions from exploration experiments. a, Screenshot from LRU simulator and point-cloud map. b, Photo from real-world experiment and voxel-grid map

data being available in larger environments. The localization and mapping components, as depicted in Figure 2, are equal to the real setup and we used the same parameters for map creation and matching in both scenarios.

7.2.2 | Results and discussion

In this evaluation, we compare three different variants of the map matching algorithm, demonstrating the impact of both the 4D initial alignment (see Section 5.2.4) and 4D refinement step (see Section 5.2.5) with respect to the resulting number of loop closures. As the baseline, we use a 6D initial alignment and 6D refinement (6D + 6D), similar to our previous work (Brand et al., 2015), 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.

In Figure 8, we present the number of matches as well as the distribution of errors of the estimated transformation 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. The number of matches greatly depends on the scenario as the opportunities to generate map matches depend on the environment as well as on the robot's trajectory. We thus 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. In both scenarios, Figure 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 majority of match error values shown in Figure 8 are in the range of the expected matcher accuracy, which is limited by noisy and imprecise stereo vision-based input data, see Section 5.2.2. 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. This means

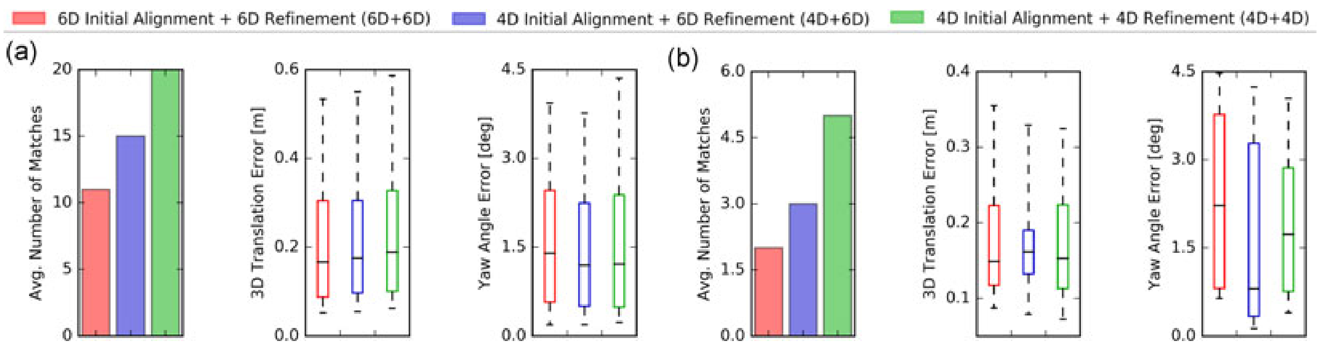

FIGURE 8 Average number of submap matches per data set and distribution of 3D translation and *yaw* angle errors per match with respect to ground truth (line: median, box: from lower to upper quartile, whiskers: 10th to 90th percentile of values, as individual large outliers will be filtered by robust estimators during graph optimization, see Section 6.2). a, Results for our 40 rough-terrain simulator experiments. b, Results for our 13 real-world lab experiments

that the increased number of matches of the novel 4D method comes at no cost in accuracy compared to the 6D method. As they also have a comparable computational complexity, the 4D matching is to be preferred.

The superiority of the 4D approach can be explained as follows: In our novel 4D matching, we solely generate 4D hypothesis that, by definition, satisfy our constraints on *roll* and *pitch*. In contrast, during 6D matching, full 6D hypotheses are computed and then filtered with respect to these constraints. Both, the RANSAC computed for each Hough3D bin during initial alignment, as well as the 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.

7.3 | Collaborative localization and mapping

To demonstrate our full system for collaborative multi-robot localization and mapping, we performed five extended multi-robot experiments with two LRU robots in our lab building. Featuring different robot trajectories of up to 200 m and 171 m in areas of up to 57 m × 53 m, they surpasses our previously published experiments (Schuster et al., 2015) and demonstrate the applicability on different robot systems.

7.3.1 | Experimental setup

The scenarios for our five experiments feature our mobile robotics lab as well as adjacent labs, hallways 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 10 for details).
- 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, thereby including indoor as well as outdoor areas.

In Figure 9, we present the 3D maps generated by our SLAM system for all five experiments. In addition, in Figure 10, we provide a sketch of our experimental setup and exemplarily selected Experiment #2 to show a more detailed map in a visual comparison to an architect's plan.

We used our aforementioned tracking system 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 aforementioned autonomous exploration experiments, in this setting we rarely moved the robots' pan/tilt unit and did not use it to perform full scans of the area. We therefore increased the threshold to generate new submaps (see Section 5.1.2) 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 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 8 we present an additional experiment with a preliminary extension of our exploration algorithm for multi-robot teams.

7.3.2 | Results and discussion

In the following, we present and discuss the results of our five multi-robot experiments, with an exemplary more detailed analysis of Experiment #2. In Figure 9, we show the resulting 3D maps for all experiments and present a sketch of our experimental setup in the right part of Figure 10. Next to it, we give a top-down view on the map of Experiment #2, which 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 1, 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 about their relative positions, but could detect each other in the lab and at hallway corners. 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. The building constitutes a challenging environment for stereo vision with low-texture areas, reflective glass surfaces and regular patterns that 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. The cause have been difficulties with stereo matching due to an untextured floor in this area, which also heavily impacted the performance of visual odometry. Our local reference filter, however, was able to compensate for this with wheel odometry and IMU measurements, and the graph SLAM then corrected a large amount of the remaining errors. It is important to note that our map representations do not contain any assumptions about a structured environment, that is, no biases toward even floors or straight walls.

In Figure 11, we present plots of the 3D trajectory errors over time for both robots in all five multi-robot experiments. All values refer to the estimates available to the robots at the respective points in time. The plots are limited to the periods of time during which ground truth measurements from our tracking system are available, that is, while the rovers were driving inside the mobile robotics lab. We compare the values of our multi-robot SLAM system with the local filter estimates. Jumps in the SLAM curves

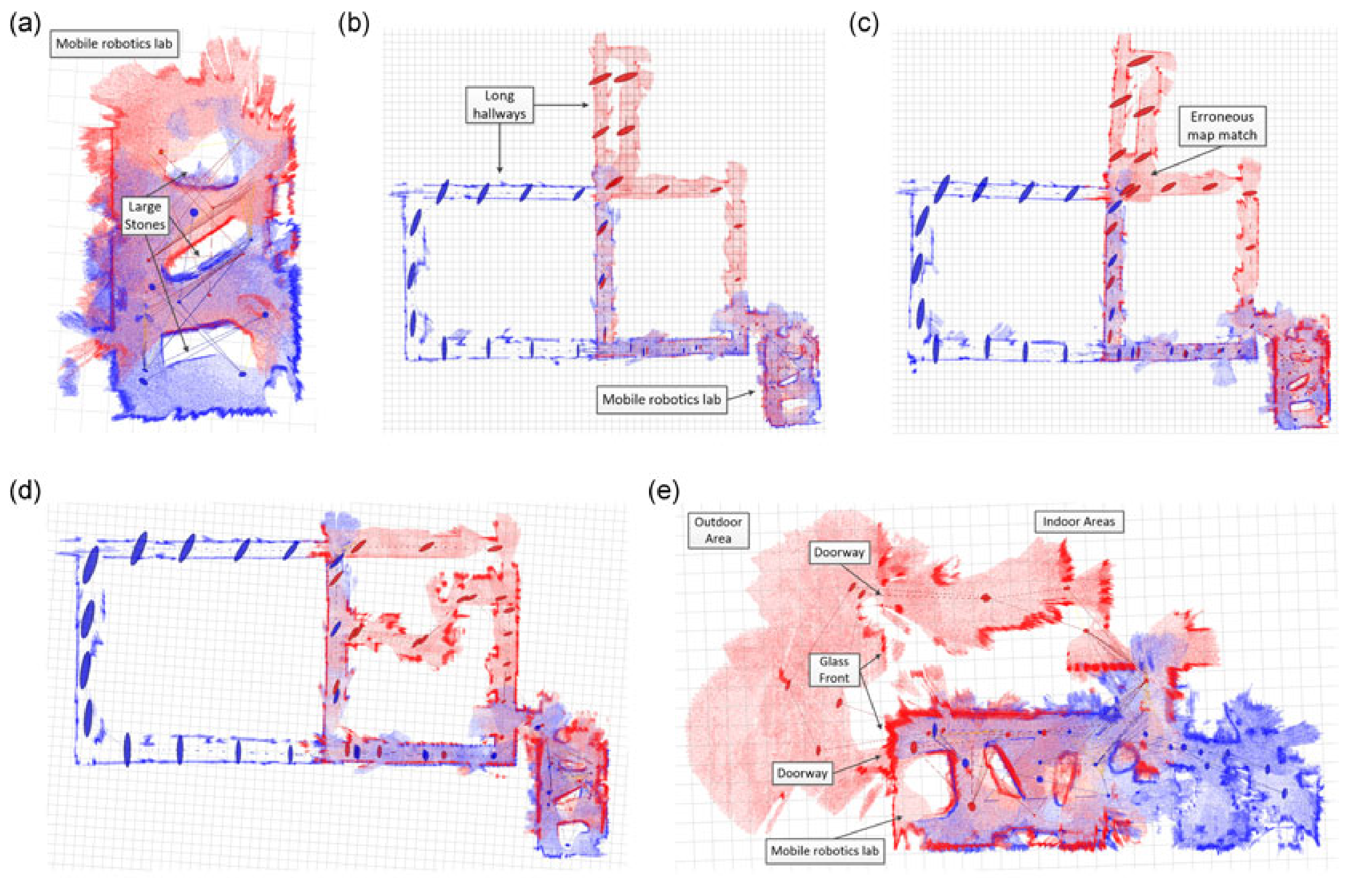

FIGURE 9 Top-down views on the 3D point cloud maps (resolution 0.05 m) created for our five multi-robot experiments with LRU1 (blue) and LRU2 (red), overlayed on grids with 1 m^2 cell size to indicate their scale. See Figure 10 for details on the experimental setup and an overlay of a floor plan for Experiment #2. a, Experiment #1. b, Experiment #2. c, Experiment #3. d, Experiment #4. e, Experiment #5

i-

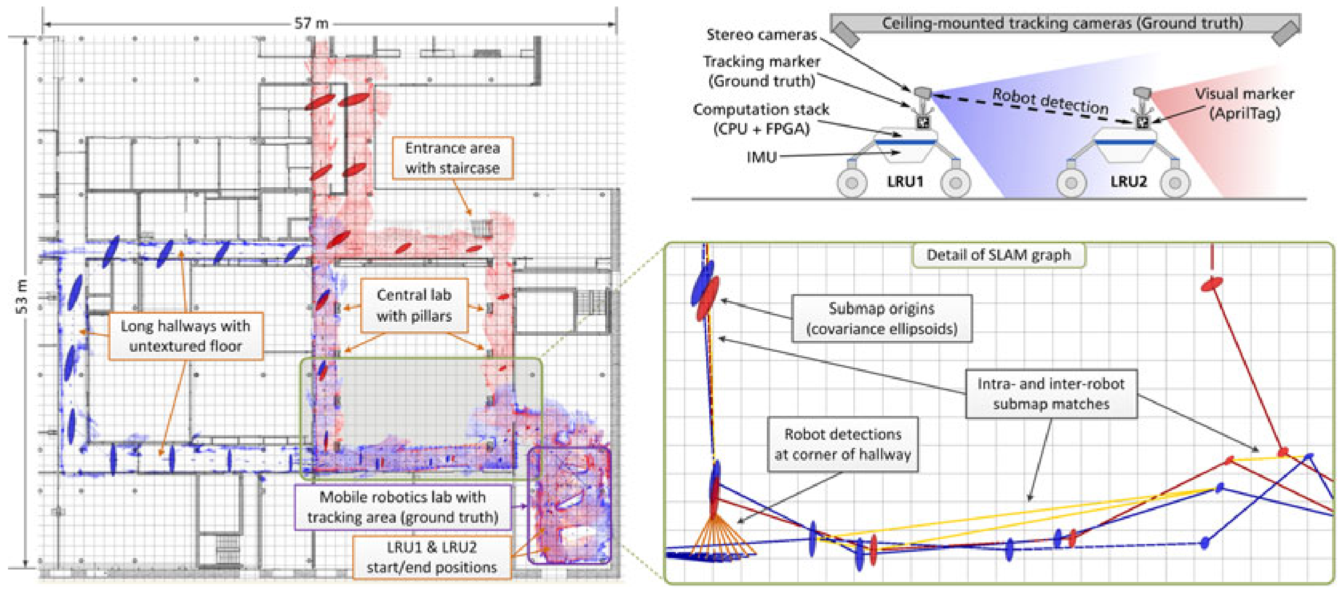
57 m
53 m
Entrance area with staircase
Central lab with pillars
Long hallways with untextured floor
Mobile robotics lab with tracking area (ground truth)
LRU1 & LRU2 start/end positions
Ceiling-mounted tracking cameras (Ground truth)
Stereo cameras
Tracking marker (Ground truth)
Computation stack (CPU + FPGA)
IMU
Robot detection
Visual marker (AprilTag)
LRU1
LRU2
Detail of SLAM graph
Submap origins (covariance ellipsoids)
Intra- and inter-robot submap matches
Robot detections at corner of hallway

FIGURE 10 Details on multi-robot experiment #2 with two rovers, LRU1 (blue) and LRU2 (red): Sketch of experimental setup and top-down view of final SLAM graph and 3D point cloud map with manually aligned floor plan. 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 2 for trajectory and graph statistics

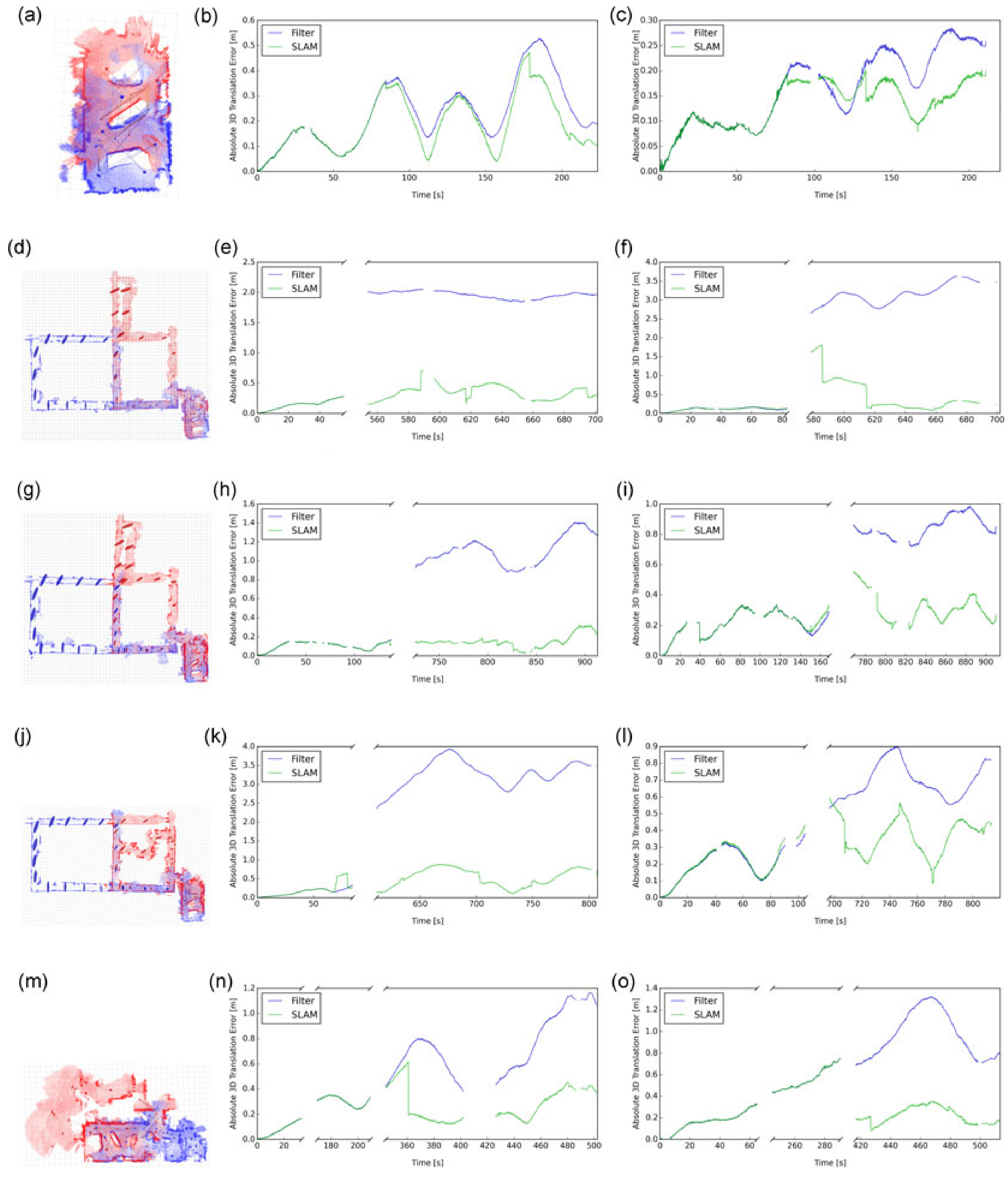

FIGURE 11 Maps (left column) and 3D trajectory errors for LRU1 (middle column) and LRU2 (right column) in our five multi-robot experiments. Gaps in the graph are due to a lack of ground truth for the respective areas, jumps in the error values for the SLAM estimate indicate loop closures. a, Map (Exp. #1). b, 3D position error (LRU1, Exp. #1). c, 3D position error (LRU2, Exp. #1). d, Map (Exp. #2). e, 3D position error (LRU1, Exp. #2). f, 3D position error (LRU2, Exp. #2). g, Map (Exp. #3). h, 3D position error (LRU1, Exp. #3). i, 3D position error (LRU2, Exp. #3). j, Map (Exp. #4). k, 3D position error (LRU1, Exp. #4). l, 3D position error (LRU2, Exp. #4). m, Map (Exp. #5). n, 3D position error (LRU1, Exp. #5). o, 3D position error (LRU2, Exp. #5) [Color figure can be viewed at wileyonlinelibrary.com]

TABLE 1 Comparison of trajectory and graph statistics (number of nodes and factors) for the five multi-robot experiments presented in Figure 11. The mean and maximum error values refer to those parts of the trajectories for which ground truth was available.

		Exp. #1		Exp. #2		Exp. #3		Exp. #4		Exp. #5	
		LRU1	LRU2	LRU1	LRU2	LRU1	LRU2	LRU1	LRU2	LRU1	LRU2
Number of robot poses $\boldsymbol{x}_i^r$		15	15	26	26	35	35	58	58	31	31
Number of submaps $\boldsymbol{s}_i^r$		8	8	29	24	32	30	32	31	17	19
Number of map	per robot	2	3	2	1	4	4	0	3	4	5
Matches $\boldsymbol{c}_i$	inter-robot	0		9		6		5		1	
Number of robot detections $\boldsymbol{d}_i$		17	0	8	19	15	22	9	55	12	21
Total number of nodes θ_i		46		105		132		179		98	
Total number of factors $\boldsymbol{f}_i$		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 trajectory error (m)		0.19	0.13	0.29	0.40	0.14	0.27	0.42	0.29	0.25	0.28
Max. 3D trajectory 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 (deg)		2.55	1.79	3.08	2.87	1.03	2.27	3.07	3.78	2.10	1.41
Max. yaw error (deg)		5.04	3.80	6.63	11.39	3.33	5.69	6.38	6.81	4.03	3.82

ndicate the impact of loop closures. For most fractions of the trajectories, the positional error after global multi-robot optimization is significantly lower than using only the local filter. In Table 1, 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 total robot trajectory lengths of up to 416 m. This allows for fast online optimization on each robot, as we discuss in Section 7.3.3 when analyzing the 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 7.2 and Figure 8). We attribute this mainly to two different causes. First, in all five multi-robot experiments, many loop-closure opportunities arise at the end of the experiment, when both rovers return to a previously visited area inside the mobile robotics lab. As described in Section 5.2, our map matcher runs in the background, processing a working queue of potential match candidates whenever free computational resources are available. We, however, ended all experiments shortly after the rovers arrived at their final position and did not wait for the matcher to finish its then nonempty queue. This particular limitation in our experimental setup leads to a 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

TABLE 2 Comparison of trajectory and graph statistics (number of nodes and factors) for multi-robot and single-robot SLAM for Experiment #2 presented in Figure 10

		Multi-Robot		Single-Robot	
		LRU1	LRU2	LRU1	LRU2
Number of robot poses $\boldsymbol{x}_i^r$		26	26	0	0
Number of submaps $\boldsymbol{s}_i^r$		29	24	29	24
Number of	per robot	2	1	0	3
Submap matches $\boldsymbol{c}_i$	inter-robot	9		0	
Number of robot detections $\boldsymbol{d}_i$		8	19	0	0
Total number of nodes θ_i		105		29	24
Total number of factors $\boldsymbol{f}_i$		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 (deg)		**3.08**	**2.87**	0.58	5.78

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 leads 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 9c, 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 to gain robustness with respect to such failure cases, which are impossible to rule out when working on limited amounts of noisy input data.

In Table 2, we compare the results of Experiment #2 to running single-robot SLAM on the same data set. 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 2 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 gets interrupted. Thus in this case, its SLAM trajectory matches that of the filter solution. 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 to just connect their maps. The robots both 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 11e,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. Accompanying this study, we present a video of our multi-robot online mapping for Experiment #2.

7.3.3 | Computational resources

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. In the multi-robot Experiment #2, the average size of a submap was 700 kB. Submaps were exchanged between the rovers whenever a submap was finished, that is, 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, GSM (Global System for Mobile Communications) networks and facilitates an upscaling to setups with a larger number of robots.

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 robots. All runtime measurements of particular processing steps refer to wall clock durations, not pure processing times. In particular the maximum computation times thus can be significantly affected by interrupts and waits caused by other processes on non-realtime systems.

We base our discussion on the results for LRU2 in our multi-robot Experiment #2, the data of the other four experiments exhibit similar effects. In Figure 12, we present the CPU and memory usage of the major components of our SLAM system, which we discuss in the following:

- Local reference filter (see Section 4): Our local reference filter runs two threads, one for the strap-down algorithm (SDA) integrating IMU data and one for the Kalman filter 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 <0.01/1.2 ms for the SDA and 0.2/2.7 ms for the filter updates, respectively.
- Graph creation and optimization (see Section 6): 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

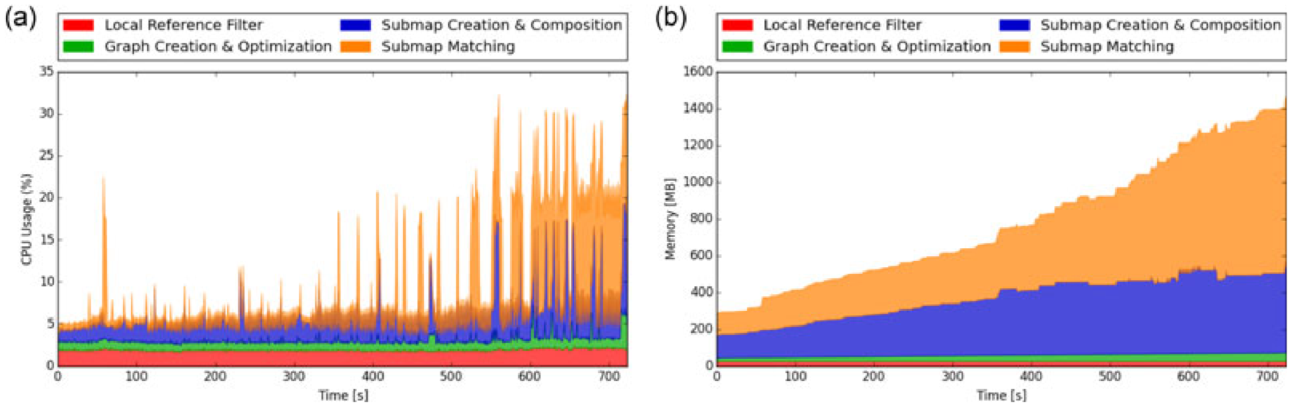

FIGURE 12 Stacked area plots of the computational resources used by our localization and mapping components to process the data of Experiment #2 for LRU2 (from bottom to top: Local reference filter, graph creation & optimization, submap creation & composition, submap matching). a, CPU usage over time (100% b = all cores). b, Memory usage over time [Color figure can be viewed at wileyonlinelibrary.com]

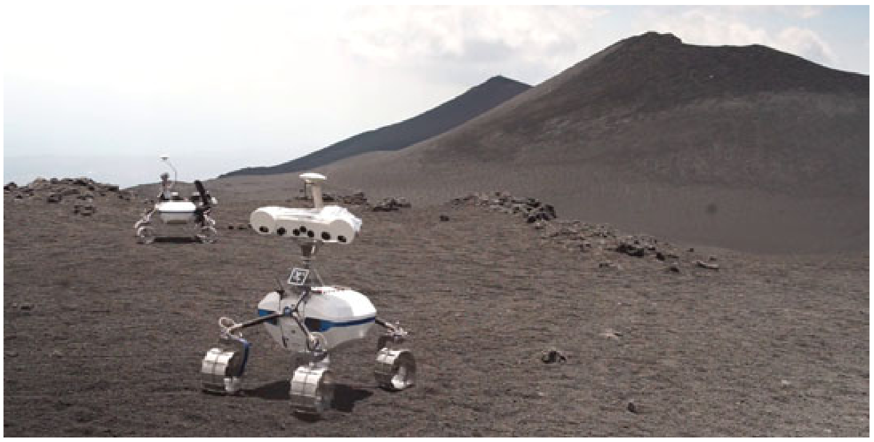
FIGURE 13 Our two LRU rovers during an autonomous multi-robot exploration experiment at a Moon-analogue test site on the volcano Mt. Etna, Sicily, Italy [Color figure can be viewed at wileyonlinelibrary.com]

the constantly growing graph. We measured a mean/max. runtime of 1.7/162 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 106/341 ms, respectively. In our current implementation, we compute this on every graph update 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 represents a topic for future work.

- Submap creation and composition (see Section 5.1): 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). Their CPU usages in Figure 12a mainly correspond to the constant part and the spikes of increasing size, respectively. 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 0.01/0.04 s for the integration of new data. The composition of the full map took a mean/max. runtime of 0.26/3.27 s, with the merging of probabilistic voxel-grid submaps being particularly expensive. We cache intermediate results, such that a full computation is only required after significant changes of the submap poses due to impactful loop closures.
- Submap matching (see Section 5.2): 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 0.39/0.67 s for keypoint selection and feature generation (per map), 3.07/8.71 s for feature matching (per selected pair), 0.02/0.05 s for Hough3D voting, including its RANSAC steps, and 2.38/4.81 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, five 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 thus 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 with respect to their runtime, processing requirements and memory consumption. We thus expect potential for significant future improvements on the values presented above.

8 | DEMONSTRATION IN MOON-ANALOGUE ENVIRONMENT

In addition to the experimental evaluation in simulation and real-world indoor environments presented in the previous section, we

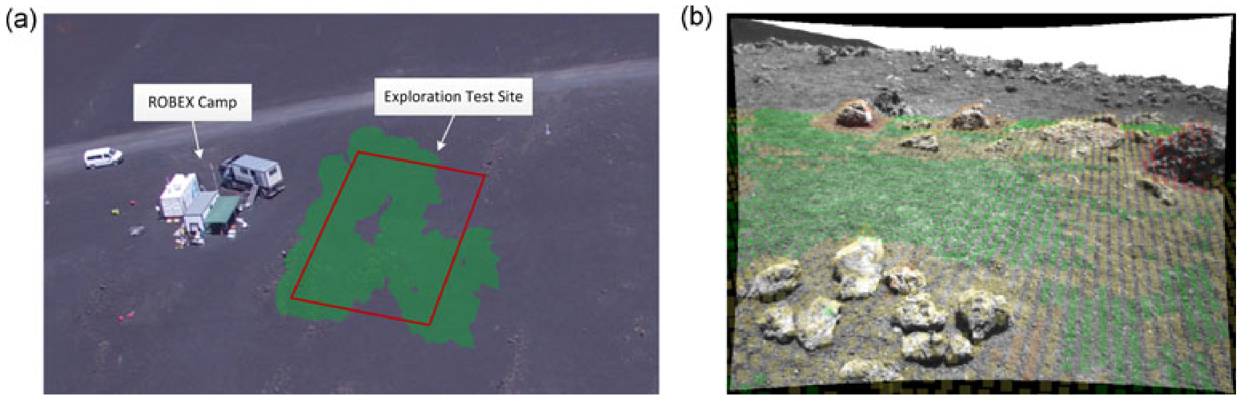

FIGURE 14 Overview of the site on Mt. Etna for our autonomous exploration experiment and view of navigation stereo camera during the experiment. a, Aerial image of test site with manually overlayed approximate exploration target area (red) and area mapped during the experiment (green). b, Image taken by LRU's navigation camera with local terrain classification overlay (green to red: traversibility from easy to hard obstacles)

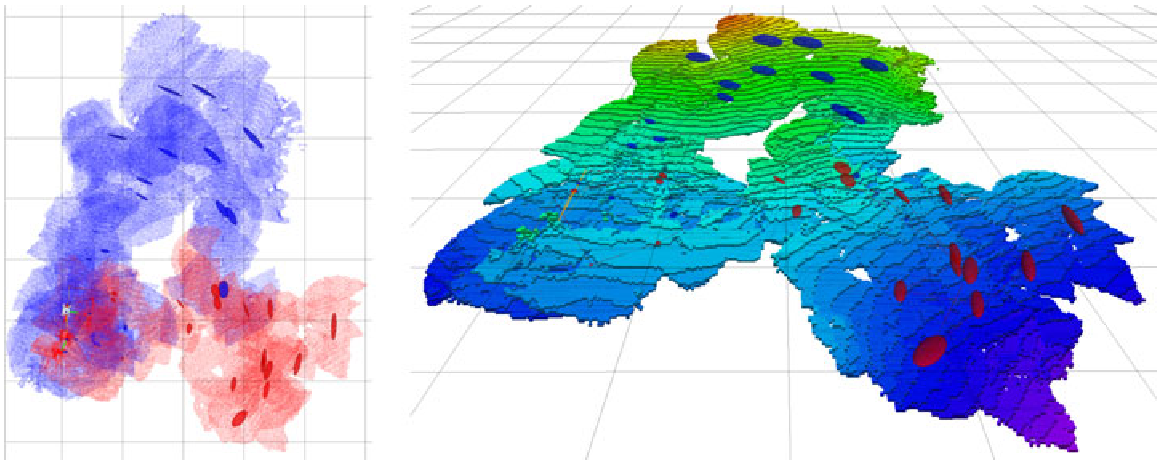

FIGURE 15 Multi-robot 3D map created during our autonomous exploration experiment on Mt. Etna. Left: Point cloud-based map (resolution 0.05 m, grid size: 5 m) created by our two rovers LRU1 (blue) and LRU2 (red). Right: Height-colored voxel grid representation (resolution 0.1 m) of the same map, showing the slope of the terrain. Similar to Figure 10, the ellipsoids represent the estimated positional standard deviation of the rovers at their submap origins with respect to the start position of LRU2

demonstrate the applicability of our methods for autonomous planetary exploration at a Moon-analogue site on the volcano Mt. Etna, Sicily, Italy, featuring a challenging, rough-terrain outdoor environment. There we conducted experiments in summer 2017 as part of the Helmholtz Alliance Robotic Exploration of Extreme Environments (ROBEX), an association of 16 universities and institutes that perform space and underwater research in extreme environments (Kanzog, 2017).

8.1 | 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. Mt. Etna has been 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 roves autonomously picked up seismic measurement instruments from a lander mockup and deployed them at predefined target locations, see Wedler et al. (2017) for details. We successfully used our localization and mapping framework for single-robot local and global localization and mapping during this mission and used the test site for an additional multi-robot exploration experiment.

8.2 | Collaborative multi-robot exploration

We conducted a preliminary collaborative exploration experiment with our two LRU rovers at the Moon-analogue site on Mt. Etna, depicted in Figure 13, to demonstrate the applicability of our multi-robot mapping methods for planetary exploration. For the experiment, we ran a frontier-based 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 apply these methods to collaborative multi-robot exploration, we extended them to spatially distribute goal locations between robots. Therefore, 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 toward leveled ground. For the experiment, we defined a target region of 25 m × 20 m to be explored, as indicated by the red polygon in Figure 14.

The exploration experiment ran for 35 min. The two rovers traveled a combined distance of 394 m and mapped an area of approx. 650 m^2, including parts outside of the exploration target area that have been traversed to avoid obstacles like large stones. Due to a limited availability of fully charged batteries, we 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. In Figure 15, we present the final map in its point cloud and probabilistic voxel-grid representations as well as the multi-robot SLAM graph, which, similarly to the previous experiments, is small and sparse with 163 nodes and 224 factors. As we applied a frontier-based exploration algorithm, the rover trajectories exhibited few 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 revisiting already mapped places for relocalization, as presented in Lehner et al. (2017).

We faced many additional challenges during the experiments at our volcanic test site at a height of 2,645 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 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 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. Although the rough-terrain ground exhibited recognizable and thus potentially matchable geometry in some areas, it was not classified as an obstacle and thus excluded from the feature matching process. We are currently looking into replacing this by an obstacle-independent selection of the map matcher keypoints. Further, 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 methods to planetary exploration scenarios. We present the mapping process of the experiment discussed here in the second half of the accompanying video.

9 | CONCLUSION AND FUTURE WORK

In this study, we have presented a multi-robot 6D localization and mapping system that combines real-time local state estimation with global online optimization. The decoupling of these allows a distributed processing of high-frequency measurements in a multi-robot team and leads to a small graph for computationally efficient optimization steps. For this, we employ a novel graph topology to incorporate the results of local reference filters according to their uncertainty estimates. In our distributed system, 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.

For 3D mapping, we aggregate high-bandwidth sensor data into submaps to online generate dense point cloud maps (resolution 0.05 m) and probabilistic 3D voxel-grid maps (resolution 0.1 m) from noisy stereo data. Sharing aggregated map data instead of raw image streams between robots allows us to reduce the required bandwidth from 38.75 MB/s to 58 kB/s. We generate loop-closure constraints from visual robot detections as well as intra- and inter-robot submap matches and present in detail our technique for matching stereo vision-based submaps on geometric environment features. In this context, the decoupling of observable and unobservable states through partial frame switching in the local reference filter allows us to introduce a novel optimization: A reduction of the dimensionality from 6D to 4D of the map matching itself leads to a, on average, 40% higher number of loop-closure constraints, as we demonstrated in our evaluation based on the data of 40 simulated and 13 real-world experiments. We integrated all these components into our modular mapping architecture that allows an easy adaption to include resource-limited systems as part of future heterogeneous multi-robot teams.

To evaluate our full SLAM system, we conducted five multi-robot experiments with two rovers in areas of up to 57 m × 53 m. Our localization and mapping components generated accurate joint maps from both robots' stereo data and estimated their trajectories with average 3D translational errors below 0.5 m with respect to partially available ground truth, that is, errors below 0.4% of the robots' respective total trajectories. In addition, we successfully demonstrated the application of our localization and mapping framework for autonomous multi-robot exploration and presented insights gained from a novel experiment conducted in the challenging environment of a Moon-analogue test site located on the volcano Mt. Etna, Sicily, Italy.

For future work, we plan to further improve our map matching algorithm by adapting the keypoint selection to include traversable but locally discriminative parts of rough terrain to obtain matches in environments with no or few obstacles. Another open challenge is the intelligent and consistent merging of submaps in a multi-robot setup to allow long-term mapping. On the graph optimization level, we work on a separation of observable and unobservable states, similar to filter and map matcher, expecting benefits in large-scale multi-robot scenarios. As we designed our mapping system for the requirements of heterogeneous robot teams, we aim for an evaluation of our approach with a team of flying and driving robots, employing the resulting maps for autonomous multi-robot exploration.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments and suggestions. We thank the current and former members of the Mobile Robots Group at DLR-RMC, especially Felix Ruess, Sebastian Vetter and Sebastian G. Brunner for their works on the local reference filter, uncertainty estimation for marker detections and pose covariance transforms, respectively. In addition, we thank Hannah Lehner for providing exploration experiment data, Philip Heijkoop, Marcus Müller, Mallikarjuna Vayugundla for their assistance with the experiments, the ROBEX team for their support on Mt. Etna, and Dr. Tim Bodenmüller, Dr. Simon Kriegel and PD Dr. habil. Rudolph Triebel for their support and many valuable discussions. This study was supported by the Helmholtz Association, project alliance ROBEX (contract number HA-304) and project ARCHES (contract number ZT-0033). The last author was funded by the European Commission, FP7 project SHERPA.

ORCID

Martin J. Schuster http://orcid.org/0000-0002-6983-3719

REFERENCES

Agarwal, P., Tipaldi, G. D., Spinello, L., Stachniss, C., & Burgard, W. (2013). Robust map optimization using dynamic covariance scaling. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA.2013.6630557

Ahmad, A., Tipaldi, G. D., Lima, P., & Burgard, W. (2013). Cooperative robot localization and target tracking based on least squares minimization. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA.2013.6631396

Ahmad, L. A. (2012). 3D descriptors for object and category recognition: A comparative evaluation. Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Bailey, T., Nieto, J., Guivant, J., Stevens, M., & Nebot, E. (2006). Consistency of the EKF-SLAM algorithm. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Brand, C., Schuster, M. J., Hirschmüller, H., & Suppa, M. (2014). Stereo-vision based obstacle mapping for indoor/outdoor SLAM. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago. https://doi.org/10.1109/IROS.2014.6942805

Brand, C., Schuster, M. J., Hirschmüller, H., & Suppa, M. (2015). Submap matching for stereo-vision based indoor/outdoor SLAM. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany. https://doi.org/10.1109/IROS.2015.7354182

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., & Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, *32*(6), 1309–1332. https://doi.org/10.1109/TRO.2016.2624754

Cunningham, A., Indelman, V., & Dellaert, F. (2013). DDF-SAM 2.0: Consistent distributed smoothing and mapping. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA.2013.6631323

Dellaert, F. (2015). GTSAM 3.2.1. https://collab.cc.gatech.edu/borg/gtsam Accessed 4 January 2017.

Dong, J., Nelson, E., Indelman, V., Michael, N., & Dellaert, F. (2015). Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach. IEEE International Conference on Robotics and Automation (ICRA), pp 5807–5814. .https://doi.org/10.1109/ICRA.2015.7140012

Durrant-Whyte, H., & Bailey, T. (2006). Simultaneous localization and mapping: Part I. *IEEE Robotics & Automation Magazine*, *13*(2), 99–110. https://doi.org/10.1109/MRA.2006.1638022

Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., & Burgard, W. (2012). An evaluation of the RGB-D SLAM system. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/ 10.1109/ICRA.2012.6225199

Forster, C., Carlone, L., Dellaert, F., & Scaramuzza, D. (2017). On-manifold preintegration for real-time visual-inertial odometry. *IEEE Transactions on Robotics*, *33*(1), 1–21. https://doi.org/10.1109/TRO.2016.2597321

Forster, C., Pizzoli, M., & Scaramuzza, D. (2013). Air-ground localization and map augmentation using monocular dense reconstruction. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2013.6696924

Grisetti, G., Kümmerle, R., Stachniss, C., & Burgard, W. (2011). A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, *2*(4), 31–43. https://doi.org/10.1109/MITS.2010.939925

Grisetti, G., Stachniss, C., & Burgard, W. (2007). Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, *23*(1), 34–46. https://doi.org/10.1109/TRO.2006.889486

Hellerer, M., Schuster, M. J., & Lichtenheldt, R. (2016). Software-in-the-loop simulation of a planetary rover. International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS).

Hesch, J. A., Kottas, D. G., Bowman, S. L., & Roumeliotis, S. I. (2014). Camera-imu-based localization: Observability analysis and consistency improvement. *International Journal of Robotics Research*, *33*(1), 182–201. https://doi.org/10.1177/0278364913509675

Hirschmüller, H. (2008). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence (IPAMI)*, *30*(2), 328–341. https://doi.org/10.1109/TPAMI.2007.1166

Hirschmüller, H., Innocent, P. R., & Garibaldi, J. M. (2002). Fast, unconstrained camera motion estimation from stereo without tracking and robust statistics.IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV). https://doi.org/10.1109/ICARCV.2002.1238577

Holz, D. & Behnke, S. (2014). Registration of non-uniform density 3D point clouds using approximate surface reconstruction. International Symposium on Robotics (ISR) and the German Conference on Robotics (ROBOTIK), Munich, Germany.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, *34*(3), 189–206. https://doi.org/10.1007/s10514-012-9321-0. http://octomap.github.com

Kaess, M. (2013). AprilTags C++ Library. http://people.csail.mit.edu/kaess/apriltags Accessed 3 January 2017.

Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J. J., & Dellaert, F. (2012). iSAM2: Incremental smoothing and mapping using the Bayes tree. *International Journal of Robotics Research*, *31*, 217–236. https://doi.org/10.1177/0278364911430419

Kanzog, C. (2017). ROBEX - Robotic Exploration of Extreme Environments. http://www.robex-allianz.de/en/ Accessed 19 October 2017.

Kim, B., Kaess, M., Fletcher, L., Leonard, J., Bachrach, A., Roy, N., & Teller, S. (2010). Multiple relative pose graphs for robust cooperative mapping. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ROBOT.2010.5509154

Koch, P. & Lacroix, S. (2016). Managing environment models in multi-robot teams. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp 5722–5728. https://doi.org/10.1109/IROS.2016.7759842

Kschischang, F. R., Frey, B. J., & Loeliger, H. -A. (2001). Factor graphs and the sum-product Algorithm. *IEEE Transactions on Information Theory*, *47*(2), 498–519. https://doi.org/10.1109/18.910572

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., & Burgard, W. (2011). g2o: A general framework for graph optimization. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA.2011.5979949

Kümmerle, R., Steder, B., Dornhege, C., Kleiner, A., Grisetti, G., & Burgard, W. (2011). Large scale graph-based SLAM using aerial images as prior information. *Autonomous Robots*, *30*(1), 25–39. https://doi.org/10.1007/s10514-010-9204-1

Labbé, M. & Michaud, F. (2014). Online global loop closure detection for large-scale multi-session graph-based SLAM. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2014.6942926

Latif, Y., Cadena, C., & Neira, J. (2014). Robust graph SLAM back-ends: A comparative analysis. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2014.6942929

Lázaro, M. T., Paz, L. M., Piniés, P., Castellanos, J. A., & Grisetti, G. (2013). Multi-robot slam using condensed measurements. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2013.6696483

Lee, G. H., Fraundorfer, F., & Pollefeys, M. (2013). Robust pose-graph loop-closures with expectation-maximization. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2013.6696406

Lehner, H., Schuster, M. J., Bodenmüller, T., & Kriegel, S. (2017). Exploration with active loop closing: A trade-off between exploration efficiency and map quality. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Leishman, R. C., McLain, T. W., & Beard, R. W. (2013). Relative navigation approach for vision-based aerial GPS-denied navigation. International Conference on Unmanned Aircraft Systems (ICUAS). https://doi.org/10.1109/ICUAS.2013.6564707

Li, M., & Mourikis, A. I. (2013). High-precision, consistent EKF-based visual-inertial odometry. *International Journal of Robotics Research*, *32*(6), 690–711. https://doi.org/10.1177/0278364913481251

Li, X., & Guskov, I. (2005). Multi-scale features for approximate alignment of point-based surfaces. Eurographics Symposium on Geometry Processing.

Lupton, T., & Sukkarieh, S. (2012). Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions. *IEEE Transactions on Robotics*, *28*(1), 61–76. https://doi.org/10.1109/TRO.2011.2170332

Mei, C., Sibley, G., Cummins, M., Newman, P., & Reid, I. (2011). RSLAM: A system for large-scale mapping in constant-time using stereo. *International Journal of Computer Vision*, *94*(2), 198–214. https://doi.org/10.1007/s11263-010-0361-7

Mendes, E., Koch, P., & Lacroix, S. (2016). ICP-based pose-graph SLAM. IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), 195-200. IEEE.

Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., & Tadokoro, S. (2012). Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, *29*(5), 832–841. https://doi.org/10.1002/rob.v29.5

Mohanarajah, G., Usenko, V., Singh, M., D'Andrea, R., & Waibel, M. (2015). Cloud-based collaborative 3D mapping in real-time with low-cost robots. *IEEE Transactions on Automation Science and Engineering*, *12*(2), 423–431. https://doi.org/10.1109/TASE.2015.2408456

Mourikis, A. I., & Roumeliotis, S. I. (2007). A multi-state constraint Kalman filter for vision-aided inertial navigation. IEEE international Conference on Robotics and Automation (ICRA).https://doi.org/10.1109/ROBOT.2007.364024

Nagatani, K., Okada, Y., Tokunaga, N., Kiribayashi, S., Yoshida, K., Ohno, K., & Koyanagi, E. (2011). Multirobot exploration for search and rescue missions: A report on map building in RoboCupRescue 2009. *Journal of Field Robotics*, *28*(3), 373–387. https://doi.org/10.1002/rob.v28.3

Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., … Fitzgibbon, A. (2011). KinectFusion: Real-time dense surface mapping and tracking. IEEE International Symposium on Mixed and Augmented Reality (ISMAR). https://doi.org/10.1109/ISMAR.2011.6092378

Nüchter, A., Lingemann, K., Hertzberg, J., & Surmann, H. (2007). 6D SLAM – 3D mapping outdoor environments. *Journal of Field Robotics*, *24*(8-9), 699–722. https://doi.org/10.1002/(ISSN)1556-4967

Olson, E. (2011). AprilTag: A robust and flexible visual fiducial system. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA.2011.5979561

Piniés, P., & Tardós, J. D. (2008). Large-scale SLAM building conditionally independent local maps: Application to monocular vision. *IEEE Transactions on Robotics*, *24*(5), 1094–1106. https://doi.org/10.1109/TRO.2008.2004636

Quang, P. B., Musso, C., & Le Gland, F. (2010). An insight into the issue of dimensionality in particle filtering. Conference on Information Fusion (FUSION). https://doi.org/10.1109/ICIF.2010.5712050

Reid, R. & Bräunl, T. (2011). Large-scale multi-robot mapping in MAGIC 2010. IEEE Conference on Robotics, Automation and Mechatronics (RAM), pp 239–244. https://doi.org/10.1109/RAMECH.2011.6070489

Roumeliotis, S. I. & Burdick, J. W. (2002). Stochastic cloning: A generalized framework for processing relative state measurements. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ROBOT.2002.1014801

Rusu, R. B. & Cousins, S. (2011). 3D is here: Point cloud library (PCL). IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ICRA.2011.5980567

Saeedi, S., Trentini, M., Seto, M., & Li, H. (2016). Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics*, *33*(1), 3–46. https://doi.org/10.1002/rob.21620 Accessed 30 August 2016.

Schadler, M., Stückler, J., & Behnke, S. (2014). Data set Spacebot Arena. http://www.ais.uni-bonn.de/mav_registration/

Schmid, K., Lutz, P., Tomić, T., Mair, E., & Hirschmüller, H. (2014a). Autonomous vision-based micro air vehicle for indoor and outdoor navigation. *Journal of Field Robotics*, *31*(4), 537–570. https://doi.org/10.1002/rob.21506

Schmid, K., Ruess, F., & Burschka, D. (2014b). Local reference filter for life-long vision aided inertial navigation. Conference on Information Fusion (FUSION).

Schmid, K., Ruess, F., Suppa, M., & Burschka, D. (2012). State estimation for highly dynamic flying Systems using key frame odometry with varying time delays. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2012.6385969

Schuster, M. J., Brand, C., Hirschmüller, H., Suppa, M., & Beetz, M. (2015). Multi-robot 6D Graph SLAM connecting decoupled local reference filters. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany. https://doi.org/10.1109/IROS.2015.7354094

Schuster, M. J., Brunner, S. G., Bussmann, K., Büttner, S., Dömel, A., Hellerer, M., … Wedler, A. (2017). Towards autonomous planetary exploration: The lightweight rover unit (LRU), its success in the SpaceBotCamp challenge, and beyond. Journal of Intelligent & Robotic Systems (JINT). https://doi.org/10.1007/s10846-017-0680-9

Schweighofer, G., & Pinz, A. (2006). Robust pose estimation from a planar target. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(12), 2024–2030. https://doi.org/10.1109/TPAMI.2006.252

Souvannavong, F., Lemaréchal, C., Rastel, L., & Maurette, M. (2010). Vision-based motion estimation for the ExoMars rover. International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS).

Tombari, F. & Di Stefano, L. (2010). Object recognition in 3D scenes with occlusions and clutter by Hough voting. Fourth Pacific-Rim Symposium on Image and Video Technology (PSIVT). https://doi.org/10.1109/PSIVT.2010.65

Tombari, F., Salti, S., & DiStefano, L. (2010). Unique signatures of histograms for local surface description. In Daniilidis, K, Maragos, P., & Paragios, N. (Eds.), *Computer Vision – ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5–11, 2010, Proceedings, Part III* (pp. 356–369). Berlin, Heidelberg: Springer.

Tombari, F., Salti, S., & DiStefano, L. (2011). A combined texture-shape descriptor for enhanced 3D feature matching. IEEE International Conference on Image Processing (ICIP). https://doi.org/10.1109/ICIP.2011.6116679

Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(4), 376–380. https://doi.org/10.1109/34.88573

Vidal-Calleja, T. A., Berger, C., Sola`, J., & Lacroix, S. (2011). Large scale multiple robot visual mapping with heterogeneous landmarks in semi-structured terrain. *Robotics and Autonomous Systems*, *59*(9), 654–674. https://doi.org/10.1016/j.robot.2011.05.008

Wedler, A., Vayugundla, M., Lehner, H., Lehner, P., Schuster, M., Brunner, S., … Wilde, M. (2017). First results of the ROBEX analogue mission campaign: Robotic deployment of seismic networks for future Lunar missions. International Astronautical Congress (IAC).

Weiss, S. M. (2012). Vision based navigation for micro helicopters (PhD thesis). Zürich, Switzerland, Eidgenössische Technische HochschuleETH Zürich.

Williams, S., Indelman, V., Kaess, M., Roberts, R., Leonard, J. J., & Dellaert, F. (2014). Concurrent filtering and smoothing: A parallel architecture for real-time navigation and full smoothing. *International Journal of Robotics Research*, *33*(12), 1544–1568. https://doi.org/10.1177/0278364914531056

Williams, S. B., Dissanayake, G., & Durrant-Whyte, H. (2002). Towards multi-vehicle simultaneous localisation and mapping. IEEE International Conference on Robotics and Automation (ICRA). https://doi.org/10.1109/ROBOT.2002.1013647

Wolchover, N. (2011). NASA Gives Up On Stuck Mars Rover Spirit. http://www.space.com/11773-nasa-mars-rover-spirit-mission-ends.html Accessed 20 December 2017.

Yamauchi, B. (1998). Frontier-based exploration using multiple robots. Autonomous Agents. https://doi.org/10.1145/280765.280773

Yousif, K., Bab-Hadiashar, A., & Hoseinnezhad, R. (2014). Real-time RGB-D registration and mapping in texture-less environments using ranked order statistics. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). https://doi.org/10.1109/IROS.2014.6942925

SUPPORTING INFORMATION

Additional supporting information may be found online in the Supporting Information section at the end of the article.

How to cite this article: Schuster MJ, Schmid K, Brand C, Beetz M. Distributed stereo vision-based 6D localization and mapping for multi-robot teams. *J Field Robotics*. 2018;1–28. https://doi.org/10.1002/rob.21812

APPENDIX: INDEX TO MULTIMEDIA EXTENSIONS

Extension	Media type	Description
1	Video	Localization and mapping with two LRU rovers in our multi-robot experiment #2 discussed in Section 7.3 and in our outdoor multi-robot exploration experiment at the Moon-analogue site on the volcano Mt. Etna presented in Section 8