2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Kimera-Multi: a System for Distributed Multi-Robot Metric-Semantic Simultaneous Localization and Mapping

Yun Chang, Yulun Tian, Jonathan P. How, Luca Carlone

***Abstract*—We present the first fully distributed multi-robot system for dense metric-semantic Simultaneous Localization and Mapping (SLAM). Our system, dubbed** Kimera-Multi**, is implemented by a team of robots equipped with visual-inertial sensors, and builds a 3D mesh model of the environment in real-time, where each face of the mesh is annotated with a semantic label (*e.g.,* building, road, objects). In** Kimera-Multi**, each robot builds a local trajectory estimate and a local mesh using** Kimera **[1]. Then, when two robots are within communication range, they initiate a distributed place recognition and robust pose graph optimization protocol with a novel incremental maximum clique outlier rejection; the protocol allows the robots to improve their local trajectory estimates by leveraging inter-robot loop closures. Finally, each robot uses its improved trajectory estimate to correct the local mesh using mesh deformation techniques. We demonstrate** Kimera-Multi **in photo-realistic simulations and real data.** Kimera-Multi **(i) is able to build accurate 3D metric-semantic meshes, (ii) is robust to incorrect loop closures while requiring less computation than state-of-the-art distributed SLAM back-ends, and (iii) is efficient, both in terms of computation at each robot as well as communication bandwidth.**

I. Introduction

Multi-robot systems have been the subject of continuous research by the robotics community due to their capability to sense and act over large-scale environments. This capability is key to improving efficiency and robustness in several applications, including factory automation, search & rescue, intelligent transportation, planetary exploration, and surveillance and monitoring in military and civilian endeavors.

In this paper, we are concerned with the problem of using a team of robots to gain situational awareness over a large environment, under realistic constraints on communication bandwidth, local sensing, and computation at each robot. In particular, our goal is to estimate a *metric-semantic* 3D model of the environment that describes the geometry of the scene the robots operate in (*e.g.,* presence and shape of obstacles), as well as its semantics, where the robots are tasked with annotating the obstacles with human-understandable labels in a given dictionary (*e.g.,* "building", "road"). The last decade has seen a renaissance in single-robot metric-semantic SLAM, pioneered by works such as SLAM++ [2] and SemanticFusion [3]. Recent work includes systems that can build metric-semantic 3D models in real-time using a multi-core CPU, including Kimera [1] and Voxblox++ [4]. These research efforts are marking a steady transition from traditional geometric SLAM to *spatial perception* (or *Spatial AI* [5]) approaches that aim at constructing high-level representations of the environment [6]. This ongoing success achieved by single-robot perception systems has not been fully harnessed in multi-robot systems. This is partially due to the complexity of designing and deploying multi-robot systems, as well as the fact that building "richer" metric-semantic representations might negatively impact the amount of communication and computation needed for the robots to build such a model. For these reasons, current multi-robot systems have mostly focused on geometric reasoning, with attention to communication aspects [7, 8] or robustness [9, 10], while disregarding dense semantics. This work advances the state of the art by developing the first system for distributed and dense metric-semantic SLAM.

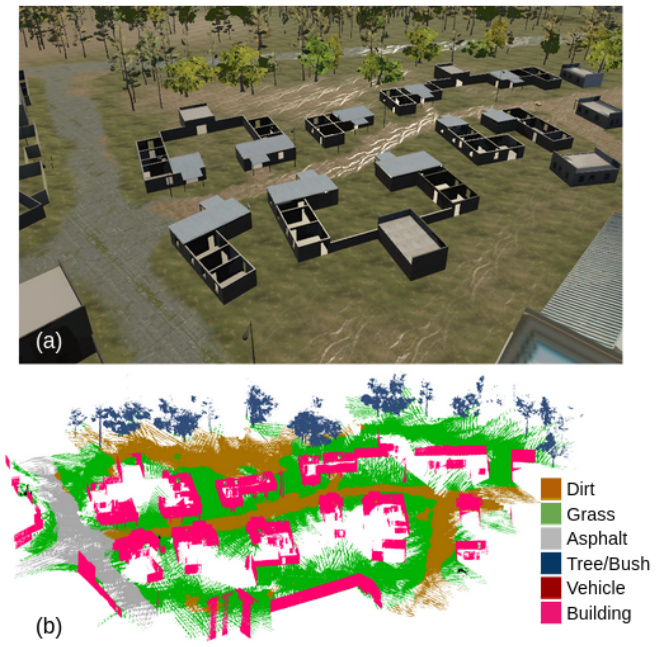


Fig. 1: (a) Camp scene generated by the Unity-based DCIST multi-robot simulator [11]. (b) Dense metric-semantic 3D mesh model generated by Kimera-Multi with three robots.

Related Work. Related work on Collaborative SLAM (CSLAM) has focused on developing multi-robot SLAM front-ends (*e.g.,* to find inter-robot loop closures) and back-ends (*e.g.,* distributed pose graph optimization). Inter-robot loop closures are critical to align the trajectories of the robots in a common reference frame and to improve their trajectory estimates. In a centralized setup, a common way to obtain loop closures is to use visual place recognition methods, based on image or keypoint descriptors [12–17]. Recent works in-

Y. Chang, Y. Tian, J.P. How, and L. Carlone are with the Laboratory for Information & Decision Systems, Massachusetts Institute of Technology, Cambridge, MA, USA, {yunchang,yulun,jhow,lcarlone}@mit.edu

This work was partially funded by ARL Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance (DCIST CRA) under agreement W911NF-17-2-0181.

vestigate *distributed* inter-robot loop closure detection, where the images are not collected at a single location and loop closures are found through local communication among the robots [7, 18–23] Centralized back-end approaches for multi-robot pose graph optimization (PGO) collect measurements at a central station, which computes the trajectory estimates for all robots [24–29]. Since centralized approaches become impractical with large teams or in the presence of tight communication constraints, the research community has recently investigated *distributed* PGO [8, 30–34] and distributed factor graph solvers [35, 36]. Front-end and back-end algorithms have been also demonstrated in complete CSLAM systems such as [7, 8, 10, 37].

While the multi-robot literature [38] has mostly focused on dense geometric representations (*e.g.,* occupancy maps [8]) or sparse landmark maps [39], single-robot SLAM research is steadily moving towards systems that can build *metric-semantic* maps [40–43, 3, 44–50, 1, 4, 6]. Related research efforts include systems building voxel-based models [3, 44–50], ESDF and meshes [1, 51, 4], or 3D scene graphs [6].

Contribution. We present Kimera-Multi, the first fully distributed system for multi-robot metric-semantic dense SLAM. The system enables a team of robots to build a semantically-annotated 3D mesh model of the environment in real-time by leveraging local sensing and computation, and intermittent communication (Sec. II). In Kimera-Multi, each robot builds a local trajectory estimate and a local mesh by processing visual-inertial sensor data using Kimera [1]. Then, when a pair of robots is within communication range, they initiate a distributed place recognition and robust PGO protocol with a maximum clique outlier rejection (Sec. III). The distributed front-end and outlier rejection modules in Kimera-Multi are similar to those of DOOR-SLAM [10], except for our more efficient *incremental* maximum clique search heuristic [52]. The distributed PGO back-end in Kimera-Multi is based on the Riemannian Block-Coordinate Descent (RBCD) method of Tian *et al.* [31], which has been shown to outperform the Gauss-Seidel back-end in [8] (also used in [10] and [7]). As a result of RBCD, the robots obtain an improved local trajectory estimate. After this distributed protocol is executed, each robot uses its improved trajectory estimate to correct the local mesh consistently with the loop closure (Sec. IV). We develop a real-time implementation of the mesh deformation approach of Sumner *et al.* [53] to correct the mesh.

We demonstrate Kimera-Multi in photo-realistic large-scale simulations and on real data (Sec. V). The results show that Kimera-Multi (i) is able to build accurate 3D metric-semantic meshes, (ii) is robust to incorrect loop closures while requiring significantly less computation compared to [10, 9] when rejecting outliers, and (iii) is efficient in terms of computation at each robot while achieving as much as 80% reduction in communication compared to a naïve centralized system.

II. Kimera-Multi: System Overview

The Kimera-Multi architecture –used by each robot in the team– is displayed in Fig. 2. Kimera-Multi includes five main modules: (i) single-robot Kimera, (ii) distributed loop closure detection, (iii) distributed outlier rejection, (iv) distributed PGO, and (v) local mesh optimization.

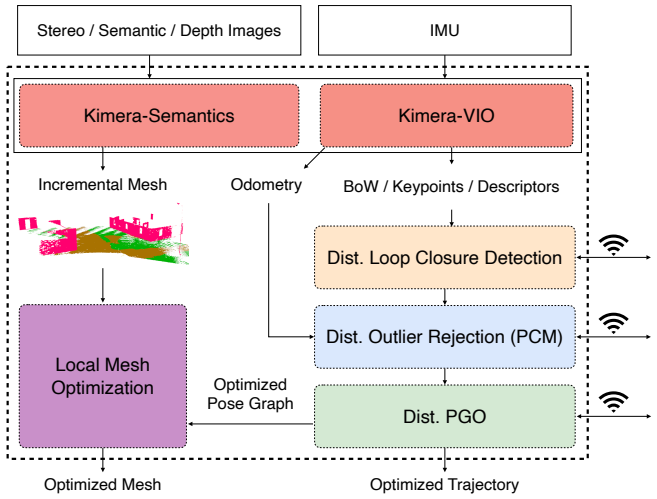


Fig. 2: Kimera-Multi: system architecture. Each robot runs Kimera (including Kimera-VIO and Kimera-Semantics) to estimate local trajectory and mesh. Robots then communicate to perform distributed loop closure detection, outlier rejection, and PGO. Given the optimized trajectory, each robot performs local mesh optimization.

Kimera [1] is implemented by each robot $\mathcal{R}_i$, and estimates the local trajectory of the robot and a local mesh corresponding to the portion of the map seen by the robot. In particular, we use Kimera-VIO [1] as a visual-inertial odometry module, which processes raw stereo images and IMU data to obtain an estimate of the odometric trajectory of the robot. Moreover, we use Kimera-Semantics [1] to process depth images (possibly obtained by stereo matching) and a 2D semantic segmentation [54] and obtain a dense metric-semantic 3D mesh. Kimera-VIO also computes a Bag-of-Words (BoW) representation of each keyframe using ORB features and DBoW2 [55], which is used for distributed loop closure detection.

Distributed Loop Closure Detection (Sec. III-A) is executed whenever two robots $\mathcal{R}_i$ and $\mathcal{R}_j$ are within communication range. The robots exchange BoW descriptors of the keyframes they collected. When the robots find a pair of matching descriptors –which typically correspond to poses observing the same place– they estimate a relative pose using standard geometric verification. The relative pose corresponds to a putative inter-robot loop closure between $\mathcal{R}_i$ and $\mathcal{R}_j$.

Distributed Outlier Rejection (Sec. III-B) is then used to vet the quality of the putative inter-robot loop closure using Pairwise Consistency Maximization (PCM) [9]. We implement a fast incremental maximum clique heuristic based on [52] to find consistent inter-robot loop closures.

Distributed PGO (Sec. III-C) takes as input the inter-robot loop closures that pass the PCM check, along with the odometry measurements produced by Kimera-VIO, and finds the optimal trajectory estimate given the measurements by both robots $\mathcal{R}_i$ and $\mathcal{R}_j$ using the RBCD pose graph solver from [31].

Local Mesh Optimization (Sec. IV) is executed after distributed PGO is complete, and is a local processing that deforms the mesh at each robot to enforce consistency with

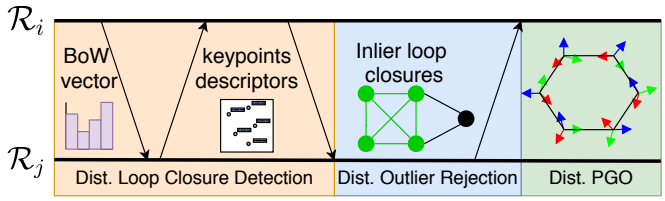


Fig. 3: Communication protocol and data flow between pair of robots.

the trajectory estimate resulting from distributed PGO.

Kimera-Multi is implemented in C++ and uses the Robot Operating System (ROS) [56] as a communication layer between robots and between the modules executed on each robot. The systems runs in real-time on a CPU and is modular, thus allowing modules to be replaced or removed. For instance, the system can also produce a dense *metric* mesh if semantic labels are not available, or only produce the optimized trajectory if the dense reconstruction is not required by the user.

III. Distributed Trajectory Estimation

This section describes the Distributed Loop Closure Detection, Distributed Outlier Rejection, and Distributed PGO modules in Fig. 2. These are the only modules in Kimera-Multi that involve communication between robots. Fig. 3 shows the data flow implemented by these modules.

A. Distributed Loop Closure Detection

Each robot $\mathcal{R}_i$ sends the BoW descriptors [55] of its keyframes to the other robot $\mathcal{R}_j$. When robot $\mathcal{R}_j$ receives a new query from robot $\mathcal{R}_i$, it searches among its own keyframes for candidate matches whose normalized visual similarity scores exceed a threshold (≥ 0.5 in our code). When a potential loop closure is identified, the robots perform standard geometric verification (details in [1]) to estimate the relative transformation between the two matched keyframes. In our implementation, robot $\mathcal{R}_j$ first requests the 3D keypoints and associated descriptors of the matched keyframe from robot $\mathcal{R}_i$ (Fig. 3). Subsequently, robot $\mathcal{R}_j$ computes putative correspondences by descriptor matching, and aligns the two sets of keypoints using Arun's method [57] with a 3-point RANSAC [58]. If the final result contains sufficient inlier correspondences (≥ 15 inliers in our code), the loop closure is accepted and sent to the distributed outlier rejection module.

B. Distributed Outlier Rejection

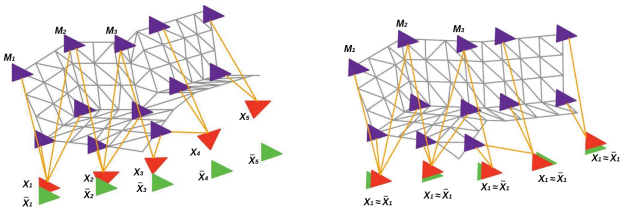
Similar to DOOR-SLAM [10], robot $\mathcal{R}_i$ then uses PCM [9] to reject spurious inter-robot loop closures. This is done by forming an undirected graph where nodes represent *all* (new/old and inlier/outlier) inter-robot loop closures between robots $\mathcal{R}_i$ and $\mathcal{R}_j$, and edges represent pairs of inter-robot loop closures that are (pairwise) "consistent". The main idea behind PCM is that for any two inlier loop closures, composing the measurements along the cycle formed by the two loop closures and the (outlier-free) odometry in the pose graph must result in the identity transformation. Based on this insight, PCM then classifies two inter-robot loop closures as pairwise consistent if the composed noisy transformation along the cycle is sufficiently close to the identity. We implement the consistency check and set the noise bounds as in [59, Sec. II-E]. A maximum clique in the PCM graph then corresponds to the largest set of *mutually consistent* inter-robot loop closures between the two robots, which is selected as the set of inliers for distributed PGO (Sec. III-C).

Incremental Approximate Maximum Clique. The main challenge with PCM is that finding a maximum clique is NP-hard. Therefore, here we propose an *incremental* approximate solution, which *updates* the maximum clique after each loop closure is detected, rather than *initiating a full search from scratch* as in [1, 10, 52]. Our main observation is that after adding a new set of loop closures to the previous PCM graph, exactly one of the following cases will occur: (i) the maximum clique identified in the previous PCM graph remains a maximum clique for the new PCM graph, which in turn implies that the inlier set remains unchanged; or (ii) there exists a larger clique in the new PCM graph, in which case the inlier set identified in the previous PCM graph must be replaced by a larger set. Therefore, our incremental search is based on the insight that in the latter case, the new (larger) maximum clique (inlier set) *must* contain *at least one of the new loop closures* (i.e., new vertices added to the PCM graph). This simple observation suggests the following algorithm: rather than searching for the maximum clique from scratch in the new PCM graph, it suffices to search for a largest clique that contains at least one of the new loop closures and update the inlier set *only if* the size of this clique is larger than the size of the maximum clique in the previous PCM graph.

In practice, we perform this *restricted search* for each set of new loop closures and use the search heuristic of [52] to find the maximum clique in the subgraph induced by the new loop closures. We also leverage our knowledge of the size of the (approximate) maximum clique in the previous graph to eliminate parts of the search space in our restricted search that *cannot* lead to larger cliques; see the pruning strategies proposed in [52]. This allows us to effectively prune the search space that continuously grows as new loop closures are detected. Overall, our incremental PCM leverages the fact that the loop closures are added over time to reduce the computational effort of the batch solutions in Kimera [1] and DOOR-SLAM [10].

C. Distributed Pose Graph Optimization

The odometry measurements and inlier loop closures returned by PCM are passed to distributed pose graph optimization to estimate the trajectories of all robots. We use the state-of-the-art Riemannian Block-Coordinate Descent (RBCD) solver [31] as our PGO backend. In short, RBCD solves the rank-restricted relaxation of PGO [60] (the default relaxation rank is set to 5) in a distributed fashion, and the solutions are subsequently projected to the special Euclidean group. Similar to the distributed Gauss-Seidel (DGS) method [8], RBCD only requires robots to exchange "public poses" (those that have inter-robot loop closures), and thereby preserves privacy (and saves communication effort) over the remaining poses. The main advantages of RBCD over DGS lies in the fact that it has provable convergence guarantees, while DGS uses an



(a) Undeformed mesh. (b) Deformed mesh.

Fig. 4: LMO deformation graph including mesh vertices (violet) and keyframe vertices (red). Edges connect two mesh vertices that are adjacent in the mesh (gray links), as well as mesh vertices with the keyframe vertices they are observed in (orange links).

approximate decoupling. Moreover, RBCD can be used as an anytime algorithm, since each iteration is guaranteed to improve over the previous iterates by reducing the PGO cost function, while DGS requires completing rotation estimation before initiating pose estimation.

IV. Local Mesh Optimization

Kimera-Semantics builds the local 3D mesh at each robot $\mathcal{R}_i$ using the pose estimates from Kimera-VIO. During this process, we keep track of the subset of 3D mesh vertices seen in each keyframe from Kimera-VIO. This allow us to implement a *local mesh optimization* (LMO) approach to correct the mesh in response to changes in the keyframe poses –due to distributed PGO– using *deformation graphs* [53]. *Deformation graphs* [53] are a model from computer graphics that deforms a given mesh in order to anchor points in this mesh to user-defined locations while ensuring that the mesh remains locally rigid; deformation graphs are typically used for 3D animations, where one wants to animate a 3D object while ensuring it moves smoothly and without artifacts [53].

In our LMO approach, we first subsample the mesh from Kimera-Semantics to obtain a simplified mesh. Then, the vertices of this simplified mesh and the corresponding keyframe poses are added as *vertices* in the deformation graph; we are going to refer to the corresponding vertices in the deformation graph as *mesh vertices* and *keyframe vertices*. Moreover, we add two types of *edges* to the deformation graph: *mesh edges* (corresponding to pairs of mesh vertices sharing a face in the mesh), and *keyframe edges* (connecting a keyframe with the set of mesh vertices it observes). For each mesh vertex k in the deformation graph, we assign a transformation $\boldsymbol{M}_k = (\boldsymbol{R}_k^M, \boldsymbol{t}_k^M)$, where $\boldsymbol{R}_k^M \in \mathrm{SO}(3)$ and $\boldsymbol{t}_k^M \in \mathbb{R}^3$; the pair $(\boldsymbol{R}_k^M, \boldsymbol{t}_k^M)$ defines a local coordinate frame, where $\boldsymbol{R}_k$ is initialized to the identity and $\boldsymbol{t}_k$ is initialized to the position $\boldsymbol{g}_k$ of the mesh vertex from Kimera-Semantics (*i.e.,* without accounting for loop closures). We also assign a pose $\boldsymbol{X}_i = (\boldsymbol{R}_k^x, \boldsymbol{t}_k^x)$ to each keyframe vertex i in the deformation graph. The pose is initialized to the pose estimates from Kimera-VIO. Therefore, our goal is to adjust these poses (and the mesh vertex positions) to "anchor" the keyframe poses to the latest estimates from distributed PGO as shown in Fig. 4.

Denoting the optimized poses from distributed PGO as $\bar{\boldsymbol{X}}_i$, and calling n the number of keyframes in the trajectory and m the total number of mesh vertices in the deformation graph, we compute updated poses $\boldsymbol{X}_i$, $\boldsymbol{M}_k$ of the vertices in the deformation graph by solving the following optimization:

$$\begin{aligned}\underset{\substack{\boldsymbol{X}_1,\dots,\boldsymbol{X}_n\in\mathrm{SE}(3)\\ \boldsymbol{M}_1,\dots,\boldsymbol{M}_m\in\mathrm{SE}(3)}}{\arg\min} & \sum_{i=0}^{n} ||\boldsymbol{X}_i \boxminus \bar{\boldsymbol{X}}_i||^2_{\Sigma_x} + \\ & \sum_{k=0}^{m} \sum_{l\in\mathcal{N}^M(k)} ||\boldsymbol{R}_k^M(\boldsymbol{g}_l - \boldsymbol{g}_k) + \boldsymbol{t}_k^M - \boldsymbol{t}_l^M||^2_{\Sigma} + \\ & \sum_{i=0}^{n} \sum_{l\in\mathcal{N}^M(i)} ||\boldsymbol{R}_i^x \widetilde{\boldsymbol{g}}_{il} + \boldsymbol{t}_i^x - \boldsymbol{t}_l^M||^2_{\Sigma} \end{aligned} \tag{1}$$

where, as before, $\boldsymbol{g}_k$ denotes the non-deformed position of vertex k in the deformation graph, $\widetilde{\boldsymbol{g}}_{il}$ denotes the non-deformed position of vertex l in the coordinate frame of keyframe i, $\mathcal{N}^M(k)$ denotes all the mesh vertices in the deformation graph connected to vertex k, and $\boxminus$ denotes a tangent-space representation of the relative pose between $\boldsymbol{X}_i$ and $\bar{\boldsymbol{X}}_i$ [61, 7.1]. Intuitively, the first term in the minimization (1) enforces ("anchors") the poses of each keyframe $\boldsymbol{X}_i$ to match the optimized poses $\bar{\boldsymbol{X}}_i$ from distributed PGO. The second term enforces local rigidity of the mesh by minimizing the mismatch with respect to the non-deformed configuration $\boldsymbol{g}_k$. The third term enforces local rigidity of the relative positions between keyframes and mesh vertices by minimizing the mismatch with respect to the non-deformed configuration in the local frame of pose $\boldsymbol{X}_i$. We optimize (1) using a Gauss-Newton method in GTSAM [62].

Since the deformation graph contains a subsampled version of the original mesh, after the optimization, we retrieve the location of the remaining vertices as in [53]. In particular, the positions of the vertices of the complete mesh are updated as affine transformations of nodes in the deformation graph:

$$\widetilde{\boldsymbol{v}}_i = \sum_{j=1}^{m} w_j(\boldsymbol{v}_i)[\boldsymbol{R}_j^M(\boldsymbol{v}_i - \boldsymbol{g}_j) + \boldsymbol{t}_j^M] \tag{2}$$

where $\boldsymbol{v}_i$ indicates the original vertex positions and $\widetilde{\boldsymbol{v}}_i$ are the new deformed positions. The weights w_j are defined as

$$w_j(\boldsymbol{v}_i) = (1 - ||\boldsymbol{v}_i - \boldsymbol{g}_j||/d_{\max})^2 \tag{3}$$

and then normalized to sum to one. Here $d_{\max}$ is the distance to the $k+1$ nearest node as described in [53] (we set $k = 4$).

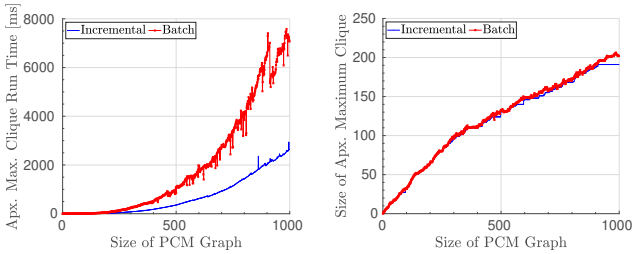
Note that the Kimera-Semantics mesh also includes semantic labels, which remain untouched in the mesh deformation.

V. Experiments

We evaluate the trajectory and the metric-semantic mesh produced by Kimera-Multi in photo-realistic simulations and real data. Our results show that Kimera-Multi is an efficient, accurate, and robust solution for distributed metric-semantic SLAM.

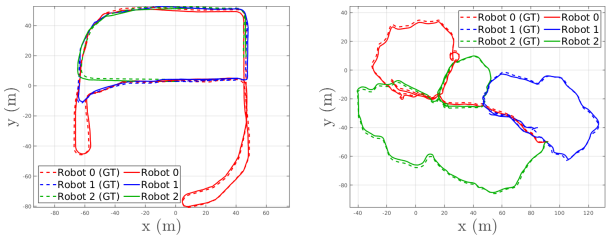
A. Experimental setup

Simulations. We evaluate Kimera-Multi in two large-scale simulation scenarios, Camp (Fig. 1) and City (Fig. 8), using the Unity-based simulator developed by the Army Research Laboratory *Distributed and Collaborative Intelligent Systems and Technology* (DCIST) Collaborative Research Alliance [11]. Simulated robots are equipped with two RGB-D cameras and an IMU. The simulator provides ground-truth trajectories and map models (that we only use for benchmarking) as well as



(a) Run Time (b) Clique Size

Fig. 5: (a) Approximate maximum clique search run times for our incremental PCM and the original (batch) PCM; (b) Size of the clique found by each method after adding each new loop closure.



(a) City (b) Camp

Fig. 6: Estimated trajectories in two simulation environments (top-down view). Robots' trajectories are shown in different colors. Corresponding ground truth are shown as dashed trajectories.

ground-truth 2D semantic segmentation that we use in Kimera-Semantics. In each scenario, we simulate 3 driving sequences (*i.e.,* 3 different robots) that we process with Kimera-Multi.

In addition, we use the Manhattan [63] synthetic PGO dataset to evaluate the performance of our *incremental* approximate maximum clique search for outlier rejection.

Real data. We test Kimera-Multi on the EuRoc dataset [64]. EuRoc includes data collected by a micro-aerial vehicle equipped with a grayscale stereo camera and IMU. We consider multiple EuRoc sequences collected in the same environment (Vicon Room 1 and 2) and process them as they were sensor feeds from different robots, hence treating Vicon Room 1 and 2 (each with 3 sequences) as two multi-robot datasets.

B. Incremental Approximate Maximum Clique

We begin by comparing the run time of our new *incremental* PCM (Sec. III-B) with the original (batch) PCM [9] used in DOOR-SLAM [10] and Kimera [1] for outlier rejection. To investigate how the run times scale with the size of PCM graph in larger problems, we use the Manhattan dataset [63]. The original dataset has more than 2000 loop closures. We added 1000 randomly generated outlier loop closures to this dataset using the tool provided by Vertigo [65]. Loop closures that are *not* consistent with the odometry are immediately discarded [59, Sec. II.E]. The remaining loop closures are added to the PCM graph one by one. Upon adding each PCM vertex (loop closure), the edges of PCM graph are updated based on pairwise consistency checks, and then PCM (batch or incremental) is used to search for an approximate maximum clique using the heuristic [52]. Fig. 5 shows (a) the run times and (b) the size of the approximate maximum clique found by each approach for up to 1000 PCM vertices. As expected, our incremental approach significantly reduces the outlier rejection run time, while producing cliques (inlier sets) of comparable size. Small differences between the clique sizes are caused by the underlying approximate search heuristic [52]. The improvement in run time becomes more significant as the size of the PCM graph (number of loop closures) increases.

C. Distributed Trajectory Estimation

In our experiments, we assume that robots are constantly in communication range. Fig. 6 visualizes the trajectory estimates produced by Kimera-Multi in the two simulated scenes. Table I reports the absolute trajectory errors (ATE) and final PGO costs. To demonstrate the flexibility of RBCD as an anytime algorithm, we include the performance of an "early stopped" (label: "RBCD (ES)") variant which terminates after 50 iterations. We compare the performance of RBCD against a baseline method that transforms each robot's locally optimized trajectory to the global frame using a single inter-robot loop closure (label: "local PGO"). In addition, we also compare against the two-stage distributed Gauss-Seidel method [8] (label: "DGS"). In our experiment, we run the first stage of DGS (rotation recovery) until convergence, and then run the second stage (pose recovery) for the same number of iterations as RBCD. Lastly, we also include the centralized SE-Sync algorithm [60] for reference. As shown in the table, RBCD outperforms local PGO, which confirms the benefits of performing inter-robot loop closure detection and PGO. In simulation, RBCD performs better compared to DGS in terms of both ATE and PGO cost, while the performances are similar on EuRoc. On all datasets, the early stopped variant of RBCD yields reasonably good trajectory estimates, which makes it a favorable choice in scenarios with run time constraints. Lastly, we observe that lower PGO costs generally translate to lower ATE, except on the Vicon Room 2 dataset in which the detected loop closures contain higher level of noise.

Table II shows the communication and computation costs of the distributed modules in our system. We compare the amount of data transmission against a naïve centralized SLAM system that transmits either raw images or extracted keypoints and descriptors to a base station. Overall, our system only uses 21%-38% of the total communication used by the centralized system that transmits image keypoints, which clearly demonstrates its communication efficiency.

We note that under unicast (point-to-point) communication, the data transmitted by our place recognition module scales quadratically with the number of robots. This quadratic communication complexity can be effectively reduced using methods in [21, 7], which is particularly helpful for scaling to a large team of robots. For computation, we report the total time and iterations spent by RBCD and its early stopped variant. For reference, we also include the run time of SE-Sync. All runtime results are generated on a Linux computer with Intel i7-7700K CPU and 16GB memory. As a first-order distributed method, RBCD is slower than the centralized SE-Sync algorithm. Nevertheless, the early stopped variant of RBCD is quite fast and still produces satisfactory solutions (see Table I).

TABLE I: Trajectory evaluation. We report the absolute trajectory error (ATE) in meters with respect to ground truth as well as the final PGO costs. For each dataset, the best performing distributed method is highlighted in **bold** and SE-Sync performance is highlighted in blue.

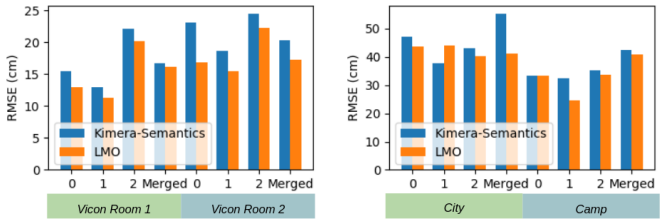
Dataset	ATE [m]					PGO cost				
	Local PGO	RBCD	RBCD (ES)	DGS	SE-Sync	Local PGO	RBCD	RBCD (ES)	DGS	SE-Sync
City	6.09	**2.38**	2.88	2.62	1.46	168532	**1113.1**	3192.9	1207.2	1047.3
Camp	2.81	**2.28**	2.52	2.58	2.28	96695	**140.45**	140.97	171.03	140.32
Vicon Room 1	0.455	**0.317**	0.348	0.346	0.308	6451.9	96.97	138.18	**91.97**	90.61
Vicon Room 2	0.473	0.453	**0.413**	0.503	0.453	432.42	**5.44**	7.19	5.60	5.15

TABLE II: Communication and computation usage of the proposed system. For communication, we show the total data transmitted in megabytes (MB) during place recognition (PR), geometric verification (GV), and distributed PGO (DPGO). We compare against centralized system that either transmits raw images or transmits detected keypoints and descriptors. For computation, we show the total number of iterations and run time for RBCD and its early stopped (ES) variant. For reference, we also report the run time of the centralized SE-Sync method [60].

Dataset	# Poses	# Edges	Communication [MB]						Computation		
			PR	GV	DPGO	Total	Centralized (Image)	Centralized (Keypoints)	RBCD Iters / Time [sec]	RBCD (ES) Iters / Time [sec]	SE-Sync Time [sec]
City	3238	3428	16.43	36.80	4.27	57.50	1989.4	149.48	500 / 34.90	50 / 2.76	2.23
Camp	5189	5411	40.01	9.59	1.17	50.77	3188.1	242.66	128 / 33.16	50 / 6.49	4.34
Vicon Room 1	1690	1730	9.52	7.32	0.16	17.00	1223.9	73.76	95 / 3.02	50 / 1.53	0.33
Vicon Room 2	1526	1544	8.95	8.63	0.20	17.78	1105.1	69.35	173 / 5.84	50 / 1.72	0.47

TABLE III: Semantic reconstruction evaluation. Semantic labels accuracy before and after correction by LMO in the DCIST simulator.

Dataset	Robot ID	Kimera-Semantics (%)	LMO (%)
City	0	**85.52**	**85.52**
	1	**88.52**	85.65
	2	85.74	**90.31**
	Merged	67.91	**83.93**
Camp	0	**95.83**	94.67
	1	96.65	**97.92**
	2	95.28	**95.42**
	Merged	94.22	**95.13**



(a) Euroc (b) Simulator

Fig. 7: Metric reconstruction evaluation. Mesh error (in centimeters) for the 3D meshes by Kimera-Semantics and Kimera-Multi's LMO.

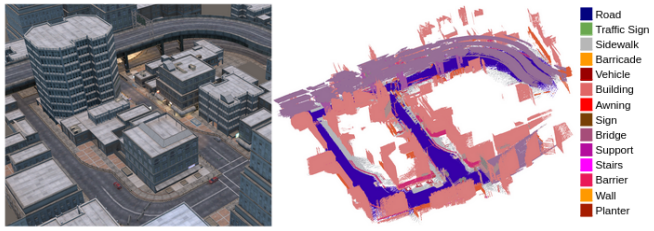


Fig. 8: Dense metric-semantic 3D mesh model generated by Kimera-Multi with three robots in the simulated City scene.

D. Local Mesh Optimization

We use the ground-truth point clouds available in the EuRoc Vicon Room 1 and 2 datasets, and the ground-truth mesh (and its semantic labels) available in the DCIST simulator to evaluate the accuracy of the 3D metric-semantic mesh by Kimera-Semantics and the impact of the local mesh optimization (LMO). For evaluation, the estimated and ground-truth meshes are sampled with a uniform density of 10^3 points/m^2 as in [1]. The resulting semantically-labeled point clouds are then registered using the ICP [66] implementation in *Open3D* [67]. Then, we calculate the mean distance between each point in the ground-truth point cloud to its nearest neighbor in the estimated point cloud to obtain the metric accuracy of the 3D mesh. In addition, we evaluate the semantic reconstruction accuracy by calculating the percentage of correctly labeled points [1] relative to the ground truth using the correspondences given by ICP. Fig. 7 reports the metric accuracy of the individual meshes constructed by each robot as well as the merged global mesh, and Table III shows the semantic reconstruction accuracy in the simulator (EuRoc does not provide ground-truth semantics). In general, the metric-semantic mesh accuracy improves after LMO for both individual and merged 3D meshes, demonstrating the effectiveness of LMO in conjunction with our distributed trajectory optimization. The dense metric-semantic meshes are shown in Fig. 1 and Fig. 8.

VI. Conclusion

We present Kimera-Multi, the first fully distributed system that leverages a team of robots to build a dense metric-semantic 3D mesh model of a large environment. Kimera-Multi combines recent advances in CPU-based metric-semantic mapping (*i.e.,*Kimera [1]), with state-of-the-art techniques for distributed pose graph optimization [31] (to which we add an incremental maximum clique outlier rejection scheme), and mesh deformation [53]. We demonstrate Kimera-Multi in two photo-realistic large-scale simulations (Camp and City environments) and on real data (EuRoc). Kimera-Multi is robust, efficient, and builds accurate 3D metric-semantic meshes. Future work includes (i) further enhancing Kimera-Multi with the Nesterov acceleration and solution verification techniques of [31], (ii) investigating distributed implementations of robust back-ends based on graduated non-convexity [68], and (iii) unifying pose graph optimization and mesh deformation into a single optimization.

References

[1] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020, arXiv preprint arXiv: 1910.02490, (video), (code), (pdf).

[2] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison, "SLAM++: Simultaneous localisation and mapping at the level of objects," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[3] J. McCormac, A. Handa, A. J. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D Semantic Mapping with Convolutional Neural Networks," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2017.

[4] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, "Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 3037–3044, 2019.

[5] A. J. Davison, "Futuremapping: The computational structure of spatial AI systems," 2018.

[6] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans," in *Robotics: Science and Systems (RSS)*, 2020, (pdf), (media), (video).

[7] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-efficient decentralized visual SLAM," *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018.

[8] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. Christensen, and F. Dellaert, "Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models, accepted," *Intl. J. of Robotics Research*, 2017, arxiv preprint: 1702.03435, (pdf) (web) (code) (code) (code) (video) (video) (video) (video).

[9] J. G. Mangelson, D. Dominic, R. M. Eustice, and R. Vasudevan, "Pairwise consistent measurement set maximization for robust multi-robot map merging," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 2916–2923.

[10] P. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: distributed, online, and outlier resilient slam for robotic teams," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 1656–1663, 2020, (video), (code), (pdf).

[11] Army Research Laboratory, "Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance (DCIST CRA)," https://www.dcist.org/, 2020.

[12] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *Intl. J. of Computer Vision*, vol. 42, pp. 145–175, 2001.

[13] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, vol. 2, April 2000, pp. 1023 – 1029.

[14] D. Lowe, "Object recognition from local scale-invariant features," in *Intl. Conf. on Computer Vision (ICCV)*, 1999, pp. 1150–1157.

[15] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: speeded up robust features," in *European Conf. on Computer Vision (ECCV)*, 2006.

[16] J. Sivic and A. Zisserman, "Video google: a text retrieval approach to object matching in videos," in *Intl. Conf. on Computer Vision (ICCV)*, 2003.

[17] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5297–5307.

[18] Y. Tian, K. Khosoussi, and J. P. How, "A resource-aware approach to collaborative loop closure detection with provable performance guarantees," *arXiv preprint arXiv:1907.04904*, 2019.

[19] M. Giamou, K. Khosoussi, and J. P. How, "Talk resource-efficiently to me: Optimal communication planning for distributed loop closure detection," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 1–9.

[20] Y. Tian, K. Khosoussi, M. Giamou, J. P. How, and J. Kelly, "Near-optimal budgeted data exchange for distributed loop closure detection," in *Robotics: Science and Systems (RSS)*, 2018.

[21] T. Cieslewski and D. Scaramuzza, "Efficient decentralized visual place recognition using a distributed inverted index," *IEEE Robotics and Automation Letters (RA-L)*, vol. 2, no. 2, pp. 640–647, 2017.

[22] D. Van Opdenbosch and E. Steinbach, "Collaborative visual slam using compressed feature exchange," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 1, pp. 57–64, 2018.

[23] D. Tardioli, E. Montijano, and A. R. Mosteo, "Visual data association in narrow-bandwidth networks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2015, pp. 2572–2577.

[24] L. Andersson and J. Nygards, "C-SAM : Multi-robot SLAM using square root information smoothing," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2008.

[25] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller, "Multiple relative pose graphs for robust cooperative mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Anchorage, Alaska, May 2010, pp. 3185–3192.

[26] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte, "Decentralised cooperative localisation for heterogeneous teams of mobile robots," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011, pp. 2859–2865.

[27] M. Lazaro, L. Paz, P. Pinies, J. Castellanos, and G. Grisetti, "Multi-robot SLAM using condensed measurements," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 1069–1076.

[28] J. Dong, E. Nelson, V. Indelman, N. Michael, and F. Del-

laert, "Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Seattle, WA, May 2015, pp. 5807–5814.

[29] P. Schmuck and M. Chli, "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," *JFR*, vol. 36, no. 4, pp. 763 – 781, 2018.

[30] R. Aragues, L. Carlone, G. Calafiore, and C. Sagues, "Multi-agent localization from noisy relative pose measurements," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011, pp. 364–369.

[31] Y. Tian, K. Khosoussi, D. Rosen, and J. How, "Distributed certifiably correct pose-graph optimization," *arXiv preprint arXiv:1911.03721*, 2019.

[32] Y. Tian, A. Koppel, A. S. Bedi, and J. P. How, "Asynchronous and parallel distributed pose graph optimization," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5819–5826, 2020.

[33] T. Fan and T. Murphey, "Majorization minimization methods to distributed pose graph optimization with convergence guarantees," *arXiv preprint arXiv:2003.05353*, 2020.

[34] E. Cristofalo, E. Montijano, and M. Schwager, "Geod: Consensus-based geodesic distributed pose graph optimization," *arXiv preprint arXiv:2010.00156*, 2020.

[35] A. Cunningham, M. Paluri, and F. Dellaert, "DDF-SAM: Fully distributed slam using constrained factor graphs," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[36] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Karlsruhe, Germany, May 2013.

[37] W. Wang, N. Jadhav, P. Vohs, N. Hughes, M. Mazumder, and S. Gil, "Active rendezvous for multi-robot pose graph optimization using sensing over Wi-Fi," *CoRR*, vol. abs/1907.05538, 2019. [Online]. Available: http://arxiv.org/abs/1907.05538

[38] G. S. Saeedi, M. Trentini, M. L. Seto, and H. Li, "Multiple-robot simultaneous localization and mapping: A review," *J. of Field Robotics*, vol. 33, pp. 3–46, 2016.

[39] V. Tchuiev and V. Indelman, "Distributed consistent multi-robot semantic localization and mapping," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4649–4656, 2020.

[40] K. Tateno, F. Tombari, I. Laina, and N. Navab, "CNN-SLAM: Real-time dense monocular slam with learned depth prediction," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[41] K.-N. Lianos, J. L. Schönberger, M. Pollefeys, and T. Sattler, "Vso: Visual semantic odometry," in *European Conf. on Computer Vision (ECCV)*, 2018, pp. 246–263.

[42] J. Dong, X. Fei, and S. Soatto, "Visual-inertial-semantic scene representation for 3D object detection," 2017.

[43] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in *Intl. Conf. on Computer Vision (ICCV)*, 2019.

[44] L. Zheng, C. Zhu, J. Zhang, H. Zhao, H. Huang, M. Niessner, and K. Xu, "Active scene understanding via online semantic reconstruction," *arXiv preprint:1906.07409*, 2019.

[45] K. Tateno, F. Tombari, and N. Navab, "Real-time and scalable incremental segmentation on dense slam," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015, pp. 4465–4472.

[46] C. Li, H. Xiao, K. Tateno, F. Tombari, N. Navab, and G. D. Hager, "Incremental scene understanding on dense SLAM," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016, pp. 574–581.

[47] J. McCormac, R. Clark, M. Bloesch, A. J. Davison, and S. Leutenegger, "Fusion++: Volumetric object-level SLAM," in *Intl. Conf. on 3D Vision (3DV)*, 2018, pp. 32–41.

[48] M. Runz, M. Buffier, and L. Agapito, "Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2018, pp. 10–20.

[49] M. Rünz and L. Agapito, "Co-fusion: Real-time segmentation, tracking and fusion of multiple objects," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4471–4478.

[50] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger, "MID-Fusion: Octree-based object-level multi-instance dynamic slam," 2019, pp. 5231–5237.

[51] A. Rosinol, T. Sattler, M. Pollefeys, and L. Carlone, "Incremental visual-inertial 3d mesh generation with structural regularities," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2019.

[52] B. Pattabiraman, M. M. A. Patwary, A. H. Gebremedhin, W. K. Liao, and A. Choudhary, "Fast algorithms for the maximum clique problem on massive graphs with applications to overlapping community detection," *Internet Mathematics*, vol. 11, no. 4-5, pp. 421–448, 2015.

[53] R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," *ACM SIGGRAPH 2007 papers on - SIGGRAPH '07*, 2007.

[54] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. García-Rodríguez, "A review on deep learning techniques applied to semantic segmentation," *ArXiv Preprint: 1704.06857*, 2017.

[55] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, October 2012.

[56] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[57] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, no. 5, pp. 698–700, sept. 1987.

[58] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, 1981.

[59] K. Ebadi, Y. Chang, M. Palieri, A. Stephens, A. Hatteland, E. Heiden, A. Thakur, B. Morrell, L. Carlone, and A. Aghamohammadi, "LAMP: large-scale autonomous mapping and positioning for exploration of perceptually-degraded subterranean environments," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[60] D. Rosen, L. Carlone, A. Bandeira, and J. Leonard, "SE-Sync: a certifiably correct algorithm for synchronization over the Special Euclidean group," *Intl. J. of Robotics Research*, 2018, accepted, arxiv preprint: 1611.00128, (pdf).

[61] T. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[62] F. Dellaert et al., "Georgia Tech Smoothing And Mapping (GTSAM)," https://gtsam.org/, 2019.

[63] E. Olson, "Robust and efficient robotic mapping," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, June 2008.

[64] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *Intl. J. of Robotics Research*, 2016.

[65] N. Sünderhauf, "Vertigo: Versatile extensions for robust inference using graph optimization." [Online]. Available: http://openslam.org/vertigo.html

[66] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, no. 2, 1992.

[67] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[68] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 1127–1134, 2020, arXiv preprint arXiv:1909.08605 (with supplemental material), (pdf).