

Toward Long-term, Automated Ship Hull Inspection with Visual SLAM, Explicit Surface Optimization, and Generic Graph-Sparsification

Paul Ozog and Ryan M. Eustice

Abstract—This paper reports on a method for an autonomous underwater vehicle to perform real-time visual simultaneous localization and mapping (SLAM) on large ship hulls over multiple sessions. Along with a monocular camera, our method uses a piecewise-planar model to explicitly optimize the ship hull surface in our factor-graph framework, and anchor nodes to co-register multiple surveys. To enable real-time performance for long-term SLAM, we use the recent Generic Linear Constraints (GLC) framework to sparsify our factor-graph. This paper analyzes how our single-session SLAM techniques can be used in the GLC framework, and describes a particle filter reacquisition algorithm so that an underwater session can be automatically re-localized to a previously built SLAM graph. We provide real-world experimental results involving automated ship hull inspection, and show that our localization filter out-performs Fast Appearance-Based Mapping (FAB-MAP), a popular place-recognition system. Using our approach, we can automatically align surveys that were taken days, months, and even years apart.

I. INTRODUCTION

Ship hull inspection for mine detection, structural assessment, and routine maintenance using an autonomous underwater robot remains a challenging research problem. This task involves deploying an unmanned underwater robot to map and inspect large underwater structures *in-situ* [1]–[3]. Automating this behavior is preferable to deploying expert human divers or trained mammals. Furthermore, drydocking the vessel to assess structural damage is much more time-consuming and expensive than using a robot.

A number of underwater robots have been used to perform ship hull inspection, ranging from free-floating robots [4] to robots that maintain constant physical contact with the hull [5]. This work is concerned with the Hovering Autonomous Underwater Vehicle (HAUV) developed by Bluefin Robotics, which uses a Doppler velocity log (DVL) for hull-relative navigation (odometry) and supports both camera and sonar sensors [6]. These sensors act to correct the unbounded navigation error from the DVL in a technique known as simultaneous localization and mapping (SLAM). The HAUV, along with examples of the vessels used in our field trials, are depicted in Fig. 2.

The SLAM system used on the Bluefin HAUV has evolved from an offline filtering-based approach with manual data

*This work was supported by the Office of Naval Research under award N00014-12-1-0092.

P. Ozog is with the Department of Electrical Engineering & Computer Science, University of Michigan, Ann Arbor, MI 48109, USA paulozog@umich.edu.

R. Eustice is with the Department of Naval Architecture & Marine Engineering, University of Michigan, Ann Arbor, MI 48109, USA eustice@umich.edu.

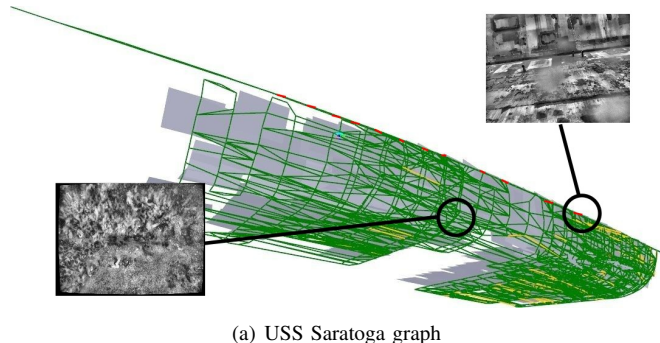


Fig. 1. GLC-sparsified graph of the underwater portion of the *USS Saratoga* with eight SLAM sessions that span approximately four months. Green lines represent GLCs, and yellow lines denote piecewise-planar factors from [13]. Gray panels are the planar features that model the ship hull as piecewise-planar.

association [7], to a real-time factor-graph system that supports both sonar and monocular camera measurements [8], [9] using the incremental smoothing and mapping (iSAM) algorithm [10]–[12] as the back-end. Despite real-time performance, the work from [8], [9] only supports single-session surveys; when the robot starts a new survey minutes or days later, the navigation is reset and all the information from the previous dive is no longer used for navigation correction. The goal of this work is to extend the HAUV visual SLAM system to support automated registration with previous maps and to marginalize redundant or unnecessary nodes from the factor-graph once the sessions are aligned into a common reference frame, as overviewed in Fig. 1.

A. Related Work

Graph-based SLAM solvers are limited by computational complexity in many long-term SLAM applications. As a robot explores an area and localizes to its map, pose nodes are continually added to the graph. Therefore, optimizing a graph of modest spatial extent becomes intractable if the duration of the robot’s exploration becomes sufficiently large. A number of proposed methods attempt to address this problem, with a recent emphasis on measurement composition for full-rank factors, such as laser scan matching or odometry [14]–[16]. However, these methods fail for factor-graphs containing low-rank measurements, such as those produced by a monocular camera.

Recently, [17], [18] developed a generic method for variable node-marginalization in factor-graph-based SLAM called Generic Linear Constraints (GLC). Unlike previous methods, the GLC-based approach is able to marginalize

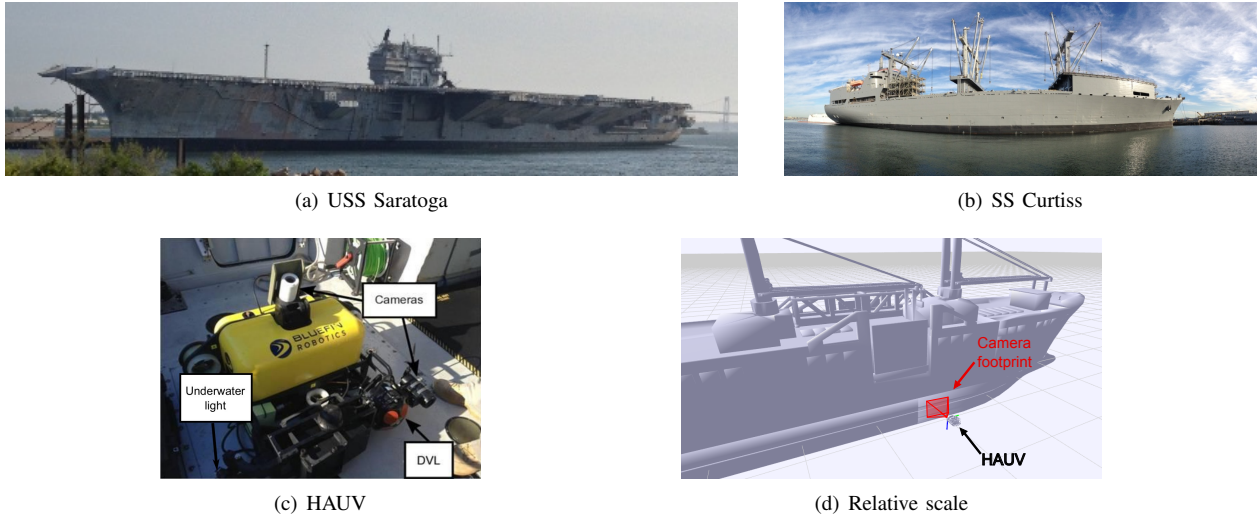


Fig. 2. The two vessels reported in this work are the *USS Saratoga*, shown in (a) and the *SS Curtiss*, shown in (b). The HAUV and important sensors are shown in (c), along with an underwater light that provides a source of illumination. In (d), we show the relative size of the robot compared to a typical ship. In this figure, the camera footprint is shown in red.

nodes with low-rank factors in the elimination clique. This is a critical requirement for the HAUV because low-rank factors from either a monocular camera or an imaging sonar are the primary means of correcting navigation error.

Additionally, the HAUV SLAM system makes use of planar patches as nodes in the graph using the piecewise-planar model from [13]. Effectively, this method provides an explicit representation of the ship hull surface in the factor graph, and produces accurate and efficient maps using only a sparse 3D point cloud (such as one produced by the underwater DVL navigation sensor). This distinguishes the work from others that use planar primitives in SLAM, such as [19], [20], which make use of data-rich 3D laser scanners in structured environments, like office buildings. Furthermore, our method is not overly reliant on prior knowledge of the ship hull curvature.

Finally, to support multi-session SLAM we require a place-recognition system to align the current survey to a past graph. Appearance-only methods such as [21], [22] utilize a bag-of-words representation for visual feature descriptors, where each descriptor is quantized to a word in the vocabulary. It is common practice to use these place recognition methods in a real-time SLAM front-end.

B. Outline

In §II, we describe how our SLAM system for the HAUV can be adapted for the GLC framework, thus enabling long-term mapping capabilities. In §II-A, we offer an overview of the GLC method, which necessitates some modifications to our previous work. In particular, §II-B describes updates to our representation of planar nodes and factors. In §II-C, we describe how relative pose-graphs containing anchor nodes can also be used with GLC. In §II-D, we provide a practical overview of the system’s behavior, and in §II-E we describe a particle filter that we use to visually re-localize with respect to past sessions. Finally, in §III, we experimentally validate

our approach using real-world data from the Bluefin HAUV, and conclude with a discussion in §IV.

II. APPROACH

Our visual SLAM system has been successfully used in single-session applications. Recent developments include explicitly representing the ship hull in the visual SLAM pose graph, and the use of anchor nodes for multi-session SLAM. It is not immediately clear how to use these techniques in the GLC framework, which provides capabilities for long-term SLAM. This section provides an overview of this sparsification method, and details how it can be used in our recent work from [9], [13].

For the remainder of this section, let $\mathbf{x}_{ij} = [x_{ij}, y_{ij}, z_{ij}, \phi_{ij}, \theta_{ij}, \psi_{ij}]^\top$ be the 6-degree of freedom (DOF) relative-pose of frame j as expressed in frame i , where x, y, z are the Cartesian translation components, and ϕ_{ij} , θ_{ij} , and ψ_{ij} denote the roll (x -axis), pitch (y -axis), and yaw (z -axis) Euler angles, respectively. R_j^i is the rotation matrix that rotates vectors in j ’s frame to vectors in i ’s frame, and \mathbf{t}_{ij}^i is the translation from i to j expressed in i ’s frame. Finally, g will refer to the global, or world-frame.

A. GLC-based Graph Sparsification

A detailed derivation and performance analysis of GLC node marginalization and removal can be found in [17], [18]. GLC-based graph sparsification makes use of an n -ary factor that captures the information induced by marginalization over an elimination clique. In short, this n -ary factor is computed by considering all the measurements contained in the Markov blanket of the node that is to be marginalized. To handle low-rank measurements, this method computes eigenvalue-decomposition of the rank- q target information, Λ_t , which forms as a generic observation model for the n -ary factor:

$$\mathbf{z}_{glc} = \mathbf{G}\mathbf{x}_c + \mathbf{w}', \quad (1)$$

where $\mathbf{w}' \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{q \times q})$, $\mathbf{G} = \mathbf{D}^{1/2} \mathbf{U}^\top$, and $\Lambda_t = \mathbf{U} \mathbf{D} \mathbf{U}^\top$. \mathbf{x}_c is the current linearization of nodes contained in the elimination clique. Next, this newly-computed factor is sparsely approximated using a Chow-Liu Tree (CLT) structure, which then simply replaces the node's original surrounding factors.

GLC supports node reparameterization into a local reference frame around \mathbf{x}_c when evaluating the observation model from (1), which optionally avoids committing to a world-frame linearization if the nodes are not well-optimized. Rather, GLC supports a root-shift operation where all nodes in the elimination are linearized about local relative-frame transformations. This relative transformation arbitrarily picks a single pose node in the elimination clique as the common frame, or root, of these transformations. In this section, we describe how to make our SLAM system compatible with GLC's ability for node reparameterization, which reduces the effect of world-frame linearization error.

B. Global Parameterization of Planar Nodes

Let π_{ik} be the plane indexed by k , expressed in frame i . As in [13], this plane corresponds to a node in our factor-graph. However, we no longer assume that the *node* indexed by k is expressed in the pose from which the plane was observed. Here, we express this plane node in the global-frame, g . Furthermore, we use a translational parameterization of planar measurements rather than a direction-magnitude representation. Therefore, $\pi_{gk} = [n_{gk}^x, n_{gk}^y, n_{gk}^z]^\top$, where n_{gk}^x , n_{gk}^y , and n_{gk}^z are the x , y , and z components of the normal vector to the plane k , expressed in the global-frame, g .

This global parameterization of planes allows us to define a root-shift operation as required for the GLC framework summarized in §II-A. To simplify the notation, we define two operators, \boxminus and \boxplus , that operate on a 6-DOF pose and 3-DOF plane.

Let $\mathbf{x}_{ij} \boxminus \pi_{ik} = \pi_{jk} = d_{jk} \mathbf{n}_{jk}$, where

$$\mathbf{n}_{jk} = \mathbf{R}_i^j \left(\frac{\pi_{ik}}{\|\pi_{ik}\|} \right)$$

$$d_{jk} = \mathbf{t}_{ij}^{i\top} \mathbf{n}_{jk}.$$

We also define the \boxplus operator, where $\mathbf{x}_{ij} \boxplus \pi_{jk} = \pi_{ik}$. This can be expressed in terms of the \boxminus operator by

$$\mathbf{x}_{ij} \boxplus \pi_{jk} = \ominus \mathbf{x}_{ij} \boxminus \pi_{jk},$$

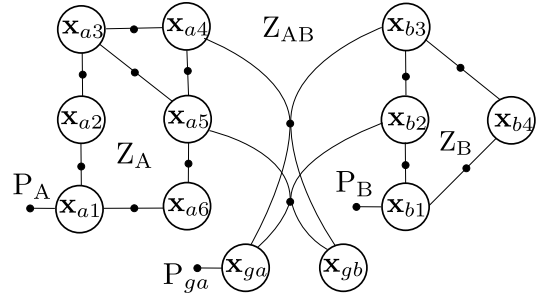
where \ominus is the pose-inversion operation defined in [23].

1) *Planar root-shift operation:* The root-shift operation needed to prevent world-frame linearization for planar nodes is simply the \boxminus operator. Given a pose \mathbf{x}_{gi} in the global-frame, and a plane π_{gj} in the global-frame, the plane node as expressed in frame i is $\mathbf{x}_{gi} \boxminus \pi_{gj}$.

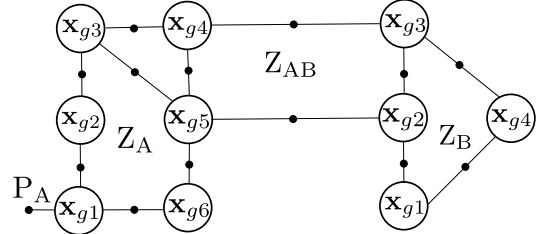
2) *Factor potentials used for planar measurements:* A binary factor potential of the form

$$\Psi(\mathbf{x}_{gi}, \pi_{gl}; \mathbf{z}_{\pi_{il}}, \Sigma_{\mathbf{z}_{\pi_{il}}}) = \|\mathbf{z}_{\pi_{il}} - (\mathbf{x}_{gi} \boxminus \pi_{gl})\|_{\Sigma_{\mathbf{z}_{\pi_{il}}}}^2 \quad (2)$$

is used when the DVL sensor determines a good planar fit of a sliding time-window of 3D points. The covariance matrix, $\Sigma_{\mathbf{z}_{\pi_{il}}}$, is determined by first-order approximation of



(a) Factor-graph with anchor nodes



(b) Factor-graph with global reference frame, g

Fig. 3. Each figure shows a factor-graph topology for a simple SLAM example involving two sessions, A and B. In (a), the graph encodes the distribution from (4), where each session has an associated anchor node. When converted to a global representation, as in (5), all of the measurements contained in and between the two sessions are preserved. The only discarded factors are the full-state prior factors in session B and on the anchor node \mathbf{x}_{ga} . These prior factors have little probabilistic importance and function primarily to keep the SLAM information matrix non-singular.

a function that fits a plane from the 3D points, which are assumed to be corrupted by Gaussian noise.

The piecewise-factor potential from [13] is now a ternary factor over a pose node and two planar nodes. It is given by the expression

$$\Omega(\mathbf{x}_{gj}, \pi_{gk}, \pi_{gl}; \mathbf{x}_{gi}, \mathbf{W}_{ij}^{\pi_{jl}}) = \|\mathbf{x}_{gj} \boxminus \pi_{gk} - (\mathbf{x}_{gj} \boxminus \pi_{gl})\|_{\mathbf{W}_{ij}^{\pi_{jl}}} \quad (3)$$

This factor potential differs from [13] in that the pose node value, \mathbf{x}_{gi} , is no longer needed to compute the non-weighted residual error because planes k and l are expressed in a common reference frame. However, it is still needed to compute a weight matrix, $\mathbf{W}_{ij}^{\pi_{jl}}$, based on the characteristic curvature of the ship hull. This weight matrix, along with an explanation of the characteristic curvature model, is described in detail in [13].

C. Global Multi-session SLAM from Anchor Nodes

For a robot or multiple robots performing multi-session SLAM, a pose-graph containing anchor nodes is a popular method and is described in [24]. The important advantages of anchor nodes over global multi-session SLAM are *i*) faster convergence of the nonlinear least-squares optimizer, and *ii*) multiple sessions can optimize their pose graphs before any measurements between the relative pose-graphs are observed.

For a two-session case consisting of sessions A and B, the

pose-graph encodes a distribution of the form

$$\begin{aligned}
 p(X \mid Z_A, Z_B, Z_{AB}, P_A, P_B, P_{ga}) = \\
 p(\mathbf{x}_{ga}, \mathbf{x}_{a1A}, \mathbf{x}_{a2A}, \dots, \mathbf{x}_{aN_A}, \\
 \mathbf{x}_{gb}, \mathbf{x}_{b1B}, \mathbf{x}_{b2B}, \dots, \mathbf{x}_{bM_B} \mid Z_A, Z_B, Z_{AB}, P_A, P_B, P_{ga}),
 \end{aligned} \quad (4)$$

where \mathbf{x}_{ga} is an anchor node representing the 6-DOF transformation from the global-frame, g , to the reference frame a of session A. \mathbf{x}_{aiA} is the 6-DOF relative pose from frame a to frame i of session A. X denotes the set of variable nodes in the factor graph (i.e., the unknowns). Z_A and Z_B denote the sensor and odometry measurements contained in sessions A and B, respectively, and Z_{AB} denotes the sensor measurements between nodes in sessions A and B. P_A and P_B denote full-state priors in sessions A and B, and P_{ga} denotes the full-state prior on the anchor node, \mathbf{x}_{ga} . Finally, N and M are the number of nodes in sessions A and B, respectively, not including the anchor node (Fig. 3).

This distribution from (4) is somewhat inconvenient because in order to place the nodes into a common frame (when visualizing the distribution, for instance, or when computing the expected information gain of a measurement between two nodes), a sparse nonlinear function, f , must be applied to the distribution:

$$\mu_{f(X)} = \begin{bmatrix} \mathbf{x}_{ga} \oplus \mathbf{x}_{a1A} \\ \vdots \\ \mathbf{x}_{ga} \oplus \mathbf{x}_{aN_A} \\ \mathbf{x}_{gb} \oplus \mathbf{x}_{b1B} \\ \vdots \\ \mathbf{x}_{gb} \oplus \mathbf{x}_{bM_B} \end{bmatrix},$$

where \oplus is defined in [23] as the ‘‘head-to-tail’’ operation. The covariance of the resulting distribution is computed to first order as

$$\Sigma_{f(X)} = J_f \Sigma_X J_f^T.$$

Extending this analysis to more than two sessions is trivial. It is also straightforward when X contains plane nodes. In this case, the sparse nonlinear function f contains the \boxplus operator discussed in §II-B.

The GLC reparameterization framework assumes that all nodes are in a common frame. To convert the distribution from (4), we first discard the two anchor nodes. Next, we discard the full-state priors for nodes in session B and the anchor node in session A, leaving us with the distribution

$$\begin{aligned}
 p(X' \mid Z_A, Z_B, Z_{AB}, P_A) = \\
 p(\mathbf{x}_{g1A}, \mathbf{x}_{g2A}, \dots, \mathbf{x}_{gN_A}, \\
 \mathbf{x}_{g1B}, \mathbf{x}_{g2B}, \dots, \mathbf{x}_{gM_B} \mid Z_A, Z_B, Z_{AB}, P_A).
 \end{aligned} \quad (5)$$

In this way, none of the sensor measurements are discarded when converting the relative pose-graphs to the global-frame. In practice, we perform this operation immediately after the robot completes a survey, and before performing offline GLC-sparsification.

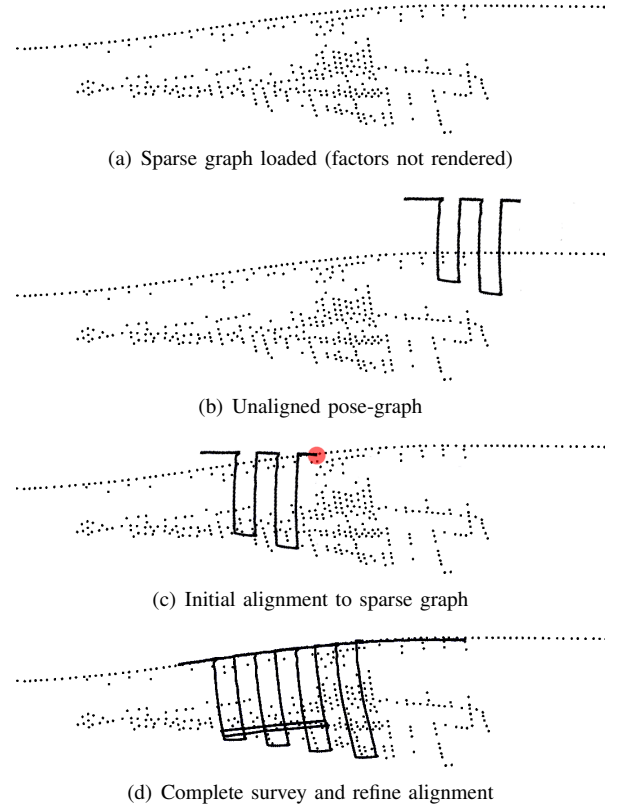


Fig. 4. Depiction of multi-session SLAM using the techniques provided in §II. The red region in (c) denotes the point of map reacquisition, where the robot determines a rough initial alignment to the graph from (a). We refine this alignment in real-time using the same techniques as our single-session system from previous work.

D. Operational Overview of Multi-session SLAM

The methods discussed in §II-A, §II-B, and §II-C are key components of our SLAM system. To illustrate how these techniques are used in an operational setting, we provide an outline of a multi-session survey as follows:

- Load past sessions as a sparsified factor graph,
- Deploy the HAUV and start building a separate, unaligned pose-graph,
- Localize to the past session with an initial guess of the alignment,
- Refine alignment using monocular camera measurements and piecewise-planar constraints anchor nodes.

These steps are illustrated in Fig. 4.

E. Particle Filtering Reacquisition to Sparsified Graph

This section describes how we accomplish the initial alignment step mentioned in §II-D. Once a graph has been sparsified using GLC, we use a particle filter to estimate a distribution of likely poses in the reference frame of the past session. Our particle filter is based on a classic Monte-Carlo localization framework. We use odometry, depth, pitch, and roll measurements published to our SLAM back-end to propagate particles to their next state.

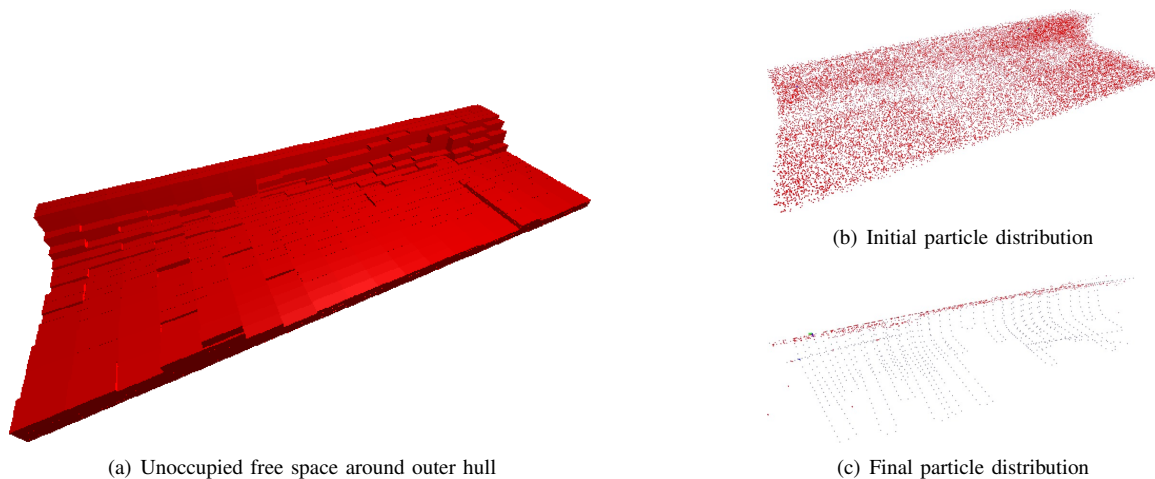


Fig. 5. Particle filter reacquisition into a previous session SLAM graph. We improve the performance of our particle filter by computing a simple occupancy grid, shown in (a), based upon the planar features of the sparsified graph. We uniformly distribute particles, shown as red dots in (b), over the free space of the outer hull. After just two planar measurements, the distribution of particles provides a relatively small set of possible candidate keyframes to search for loop closures (c).

1) *Weighting particles from planar measurements:* To weight the particles as new plane measurements are observed, we use a method similar to the computation of potentials used in our factor-graph, described in §II-B. When a plane-fitting measurement, $\mathbf{z}_{\pi_{ik}}$ from (2), is received from the particle filter, it finds the nearest neighboring pose node i according to

$$i = \operatorname{argmin}_{i \in I_{\Pi}} \|\mathbf{t}_{gp_i}^g - \mathbf{t}_{g_i}^g\|,$$

where I_{Π} is the set of pose indices in the GLC-sparsified graph that have a corresponding plane. Finding the minimum over all I_{Π} is quite slow for large graphs, so this operation is approximated using a k -dimensional (KD)-tree from a heavily-optimized library [25].

Next, we take i' to be the index of the plane that is observed from pose i . Finally, we set the weight, w_{p_i} , by computing the difference between the observed and expected planar measurement:

$$w_{p_i} = \|\mathbf{z}_{\pi_{p_i k}} - (\mathbf{x}_{g_i} \boxminus \mathbf{x}_{g_{i'}})\|_{\Sigma_{i'}}^2.$$

To get the initial distribution of particles, we compute a simple binary 3D occupancy grid from the GLC-sparsified graph, and remove any particles that are assigned to occupied cells. For each cell, if it lies outside the nearest pose-plane pair, that cell is marked as “not occupied.” Otherwise, the cell is marked as “occupied.” An example free space grid for the *USS Saratoga* is shown in Fig. 5(a).

2) *Keyframe Matching with SIFT and RANSAC:* If after resampling, the distribution of particles physically overlaps with a sufficiently small set of candidate images associated with the GLC-sparsified graph, we do a brute-force search over this set to find the best match. This procedure is described in Algorithm 1. The final step, `FINDBESTMATCH`, uses a graphics processing unit (GPU) for scale-invariant feature transform (SIFT) descriptor extraction and matching, and finally rejects outliers by using a eight-point random

sample consensus (RANSAC) algorithm to fit a fundamental matrix between the current keyframe and a candidate image extracted from the set. The keyframe with the most inliers above a threshold is selected to be the best match. For our application, we conservatively choose a threshold of 16 inliers to avoid false positives. If no matches are found, the search is repeated when the robot moves approximately three meters from the point at which the previous search was attempted. For our application, the `FINDBESTMATCH` step may fail when the robot is viewing non-salient imagery such as above-water metallic surfaces with intense sunlight reflection, or underwater portions of the hull with few visual features.

Algorithm 1 Match current keyframe to candidate keyframes based on particle distribution

- 1: **Input:** N samples from particle distribution, current keyframe k , set of all pose indices in GLC-sparsified graph with corresponding keyframe I_K
 - 2: $S \leftarrow \emptyset$
 - 3: **for** $p_i \in \{p_1 \dots p_N\}$ **do**
 - 4: $S \leftarrow S \cup \text{NEIGHSTNEIGHBOR}(p_i, I_K)$ ▷ Uses kd-tree
 - 5: **end for**
 - 6: **Output:** `FINDBESTMATCH`(k, S) ▷ Uses GPU
-

Algorithm 2 Match current keyframe to feasible keyframes from place-recognition system

- 1: **Input:** Set of all keyframes, K , in GLC-sparsified graph, current keyframe k , belief threshold τ , ship-specific vocabulary, V
 - 2: $S \leftarrow \text{FAB-MAPV2}(k, K, V, \tau)$
 - 3: **Output:** `FINDBESTMATCH`(k, S) ▷ Uses GPU
-

III. EXPERIMENTAL TRIALS

This section describes experimental trials with the HAUV performing automated inspections on the *USS Saratoga* and

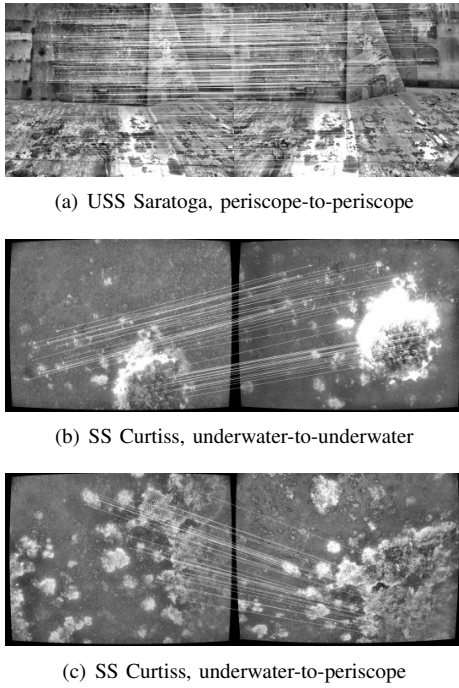


Fig. 6. Examples of matching keyframes from different SLAM sessions that our algorithm automatically detects. White horizontal lines denote matching SIFT keypoints. In (a), a match between two images of the superstructure, captured by the periscope camera is shown. In (b), we detected two corresponding biogrowth patterns using the underwater camera. In (a) and (b), the matches are easy for a human to identify, but in (c), the pair is a difficult case for a human.

SS Curtiss vessels depicted in Fig. 2. The vehicle has two cameras: an underwater camera, which is actuated to always point nadir to the ship hull, and the periscope camera, which is rigidly attached to the top of the vehicle so as to capture images of the superstructure when the vehicle is at the surface.

The HAUV executes one of the following types of missions:

- Surface survey using only periscope camera,
- Underwater survey using underwater imaging,
- Underwater survey using periscope imaging,
- Underwater survey with periodic surfacings for periscope.

While the current mission executes, it is localized to a GLC-sparsified graph using Algorithm 1, which is seeded by the output of our particle filter. Examples of matches using this approach are shown in Fig. 6.

To baseline the performance of our particle filter, we use an open-source implementation of Fast Appearance-Based Mapping (FAB-MAP) version 2.0 [26] as an appearance-only method for place recognition, which represents each keyframe as a visual bag-of-words (BoW) using a vocabulary that is learned offline. FAB-MAP provides the Bayesian probability that a candidate image is taken from the same place as another image in the test set, or represents a new place. If a match is detected with significant probability, a threshold is used to determine if the SLAM front-end should

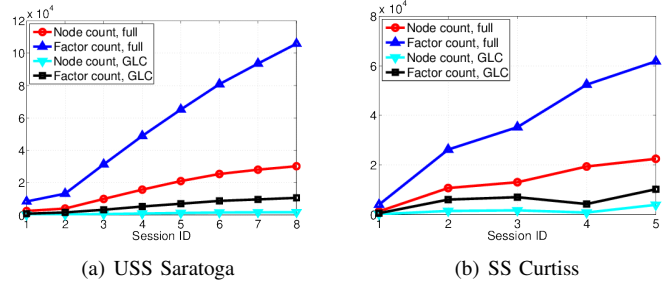


Fig. 7. Graph complexity over multiple sessions for the *USS Saratoga*, in (a), and the *SS Curtiss*, in (b). The number of nodes and factors grows approximately linearly for the full case, but the growth rate of the sparsified graph is bounded. For the *USS Saratoga*, sessions 1 through 7 occur within a week of each other, but session 8 occurs four months later. For the *SS Curtiss*, session 5 occurs approximately two years after sessions 1 through 4.

accept the match. This threshold is typically set very high to avoid false-positives, but we use a very low threshold of 0.0001 for our application to ensure that FAB-MAP returns as many candidates as possible. These candidates are robustly verified using RANSAC in the final step of Algorithm 2. We assume a uniform motion model and enforce no smoothing on the observation likelihoods in order to keep FAB-MAP’s recall as high as possible, with little regard for false positive rate. Furthermore, we learn a separate SIFT vocabulary and CLT over the distribution of codewords for each vessel.

A. Graph Complexity Over Time

For selecting which nodes should be marginalized, we primarily use a simple criteria that selects spatially redundant nodes. Our results suggest that the proposed visual saliency score from [9] also acts as a good criteria for sparsifying a visual SLAM graph. This local saliency score measures the entropy of the distribution of words in the BoW representation in each keyframe. It is a measure of how feature-rich the image is, which acts as a strong indication of a front-end’s chances of successfully estimating a monocular camera measurement using that image. The results of this method are shown in Fig. 7, where the graph complexity is plotted over each successive session. Using the CLT approximation during graph sparsification is critical for maintaining real-time performance, and is constructed in such a way that the Kullback-Leibler Divergence (KLD) from the unsparsified graph’s distribution is minimized.

We find that the visual saliency criteria performs well on the *SS Curtiss* data, where the ship has local clusters of salient regions, but has little affect on the *USS Saratoga*, which is more uniformly salient. This is visually apparent in Fig. 9, to be shown.

B. Comparison to Bag-of-Words Place Recognition

Based upon our experiments, FAB-MAP’s performance is acceptable when matching images of the ship’s above-water superstructure, like the ones shown in Fig. 6(a) or Fig. 8, top row. For these images, our particle filter returns a comparable amount of keyframes. However, for the underwater images

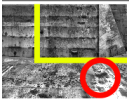
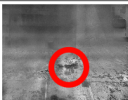
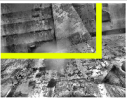
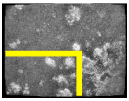
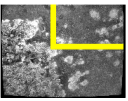
Keyframe to Match	No. of candidates (FAPMAP v2)	No. of candidates (Particle Filter)	Best Match (FABMAP)	Best Match (Particle Filter)
	160	136		
	1	203	New Place	

Fig. 8. Two representative attempts at aligning the current survey into past sessions using FAB-MAP and our particle filter. Corresponding image regions, where identified, are highlighted. We allow FAB-MAP to return as many candidates as possible by setting the loop-closure probability threshold low. The number of candidate matches identified by our particle filter is also large, but matching them is fast because each keyframe only takes roughly 70 ms when using a consumer-grade GPU. FAB-MAP performs comparably to our method for periscope images (top row) but fails using underwater images (bottom row), where it consistently assigns the “new place” label.

TABLE I
TIME TO LOCALIZE TO PAST GLC GRAPH

Session	Time until alignment (sec)	
	FAB-MAP	Particle Filter Search
<i>USS Saratoga</i> 2013 Session 7, starting from surface	7.1	5.2
<i>USS Saratoga</i> 2013 Session 7, starting underwater	143.2	20.6
<i>SS Curtiss</i> 2011 Session 4, underwater-only	N/A	15.3

in our application, FAB-MAP performs poorly, and we were not able to successfully identify a underwater loop-closure in any of our experiments, even with a very low loop-closure probability threshold. In these cases, FAB-MAP typically returns only a few candidate loop-closures. These two typical cases are shown visually in Fig. 8. We do not provide precision-recall curves because we use FAB-MAP and our particle-filter for a one-time-only localization step, not as a persistent component of a SLAM front-end. Furthermore, we do not have ground-truth and therefore no way of accurately counting false negatives.

Where our method excels over FAB-MAP is the ability re-localize into a previous session while underwater. We consider three scenarios: *i*) starting the survey with the robot at the surface, *ii*) starting submerged, and *iii*) a survey conducted entirely underwater. The results summarized in Table I are follows: both methods are comparable when at the surface (first row), but FAB-MAP is unable to re-localize while starting underwater, and only when the robot surfaces can it find a good match (second row). For an entirely underwater survey (third row), FAB-MAP is unable to localize. As a whole, our system, which is specifically engineered for the sensor payload of the HAUV, can more quickly and more robustly localize to our long-term SLAM graphs.

IV. CONCLUSION

We provided an overview of how to adapt our visual SLAM algorithm for long-term use on large ship hulls. We use the GLC framework to remove redundant or unneeded nodes from a factor graph. Doing so involves supporting a node reparameterization (root-shift) operation to avoid unnecessary error induced by world-frame linearization. Furthermore, we described a particle filtering algorithm that can use planar surface measurements to narrow a search space over past images that match the current image.

We showed results from our localization algorithm automatically aligning SLAM sessions separated in time by days, months, and years. Once sessions were aligned to a past graph, the result was sparsified and the process was repeated. Using simple sparsification criteria, we show that the complexity of our factor graphs remain bounded over the long-term.

REFERENCES

- [1] A. Carvalho, L. Sagrilo, I. Silva, J. Rebello, and R. Carneval, “On the reliability of an automated ultrasonic system for hull inspection in ship-based oil production units,” *Applied Ocean Research*, vol. 25, no. 5, pp. 235 – 241, 2003.
- [2] S. Negahdaripour and P. Firoozfam, “An ROV stereovision system for ship-hull inspection,” *IEEE J. Ocean. Eng.*, vol. 31, no. 3, pp. 551–564, 2006.
- [3] P. Rida, M. Carreras, D. Ribas, and R. Garcia, “Visual inspection of hydroelectric dams using an autonomous underwater vehicle,” *J. Field Robot.*, vol. 27, no. 6, pp. 759–778, Nov. 2010.
- [4] G. Trimble and E. Belcher, “Ship berthing and hull inspection using the CetusII AUV and MIRIS high-resolution sonar,” in *Proc. IEEE/MTS OCEANS Conf. Exhib.*, vol. 2, 2002, pp. 1172–1175.
- [5] K. Ishizu, N. Sakagami, K. Ishimaru, M. Shibata, H. Onishi, S. Murakami, and S. Kawamura, “Ship hull inspection using a small underwater robot with a mechanical contact mechanism,” in *Proc. IEEE/MTS OCEANS Conf. Exhib.*, 2012, pp. 1–6.
- [6] M. Kaess, H. Johannsson, B. Englot, F. Hover, and J. Leonard, “Towards autonomous ship hull inspection using the Bluefin HAUV,” in *Proc. Int. Symp. on Tech. and the Mine Prob.*, Naval Postgraduate School, Monterey, USA, May 2010.
- [7] M. Walter, F. Hover, and J. Leonard, “SLAM for ship hull inspection using exactly sparse extended information filters,” in *Proc. IEEE Int. Conf. Robot. and Automation*, Pasadena, USA, 2008, pp. 1463–1470.
- [8] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, “Advanced perception, navigation and planning for autonomous in-water ship hull inspection,” *Int. J. Robot. Res.*, vol. 31, no. 12, pp. 1445–1464, Oct. 2012.
- [9] A. Kim and R. M. Eustice, “Real-time visual SLAM for autonomous underwater hull inspection using visual saliency,” *IEEE Trans. Robot.*, vol. 29, no. 3, pp. 719–733, Jun. 2013.
- [10] M. Kaess, A. Ranganathan, and F. Dellaert, “iSAM: Incremental smoothing and mapping,” *IEEE Trans. Robot.*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.
- [11] M. Kaess and F. Dellaert, “Covariance recovery from a square root information matrix for data association,” *Robot. and Auton. Syst.*, vol. 57, pp. 1198–1210, Dec. 2009.
- [12] M. Kaess, H. Johannsson, D. Rosen, N. Carlevaris-Bianco, and J. Leonard, “Open source implementation of iSAM,” <http://people.csail.mit.edu/kaess/isam>, 2010.
- [13] P. Ozog and R. M. Eustice, “Real-time SLAM with piecewise-planar surface models and sparse 3d point clouds,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Tokyo, Japan, Nov. 2013, Accepted, To Appear.
- [14] K. Konolige and J. Bowman, “Towards lifelong visual maps,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, St. Louis, USA, 2009, pp. 1156–1163.
- [15] E. Eade, P. Fong, and M. Munich, “Monocular graph SLAM with complexity reduction,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Taipei, Taiwan, 2010, pp. 3017–3024.

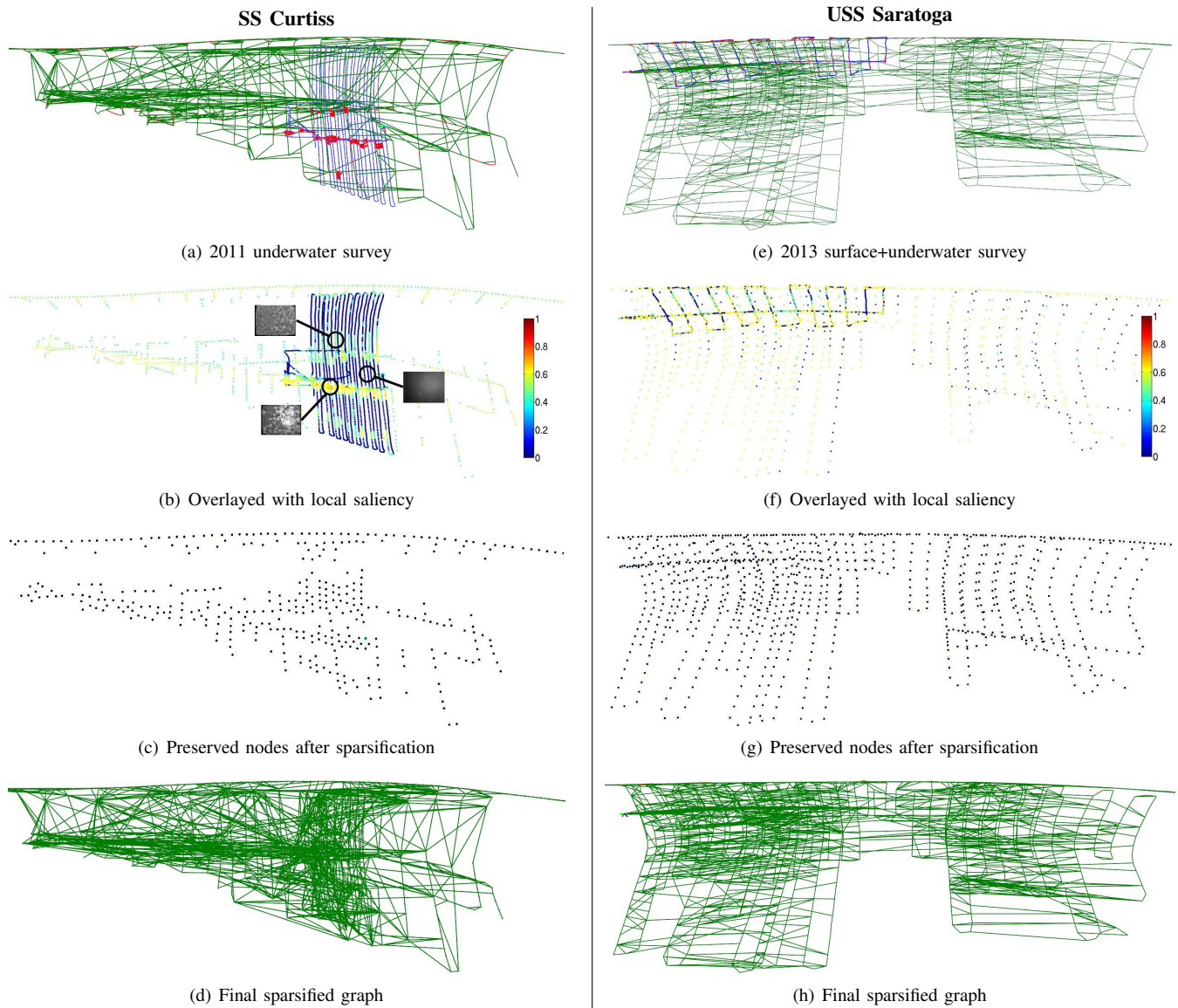


Fig. 9. Example of underwater surveys that are automatically aligned with a GLC-sparsified graph. The left and right columns show examples from the *SS Curtiss* and the *USS Saratoga*, respectively. In these figures, we disable rendering the planar patches and factors for the sake of visual clarity. In (b) and (f), we overlay the nodes with their local saliency score to show that low scores, in blue, will be sparsified while nodes with high scores, shown in yellow, will be kept so as to provide a more visually informative set of nodes for future localization. Despite the increased spatial extent of the green GLCs, the graphs in (d) and (h) have significantly fewer edges than the graphs in (a) and (e), as also shown in Fig. 7.

- [16] H. Kretschmar and C. Stachniss, "Information-theoretic compression of pose graphs for laser-based SLAM," *Int. J. Robot. Res.*, vol. 31, pp. 1219–1230, 2012.
- [17] N. Carlevaris-Bianco and R. M. Eustice, "Generic factor-based node marginalization and edge sparsification for pose-graph SLAM," in *Proc. IEEE Int. Conf. Robot. and Automation*, Karlsruhe, Germany, May 2013, pp. 5728–5735.
- [18] —, "Long-term simultaneous localization and mapping with generic linear constraint node removal," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Tokyo, Japan, Nov. 2013, Accepted, To Appear.
- [19] J. Weingarten and R. Siegwart, "3D SLAM using planar segments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 2006, pp. 3062–3067.
- [20] A. J. B. Trevor, J. G. Rogers, and H. I. Christensen, "Planar surface SLAM with 3D and 2D sensors," in *Proc. IEEE Int. Conf. Robot. and Automation*, 2012, pp. 3041–3048.
- [21] D. Nistér and H. Stewénius, "Scalable recognition with a vocabulary tree," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, vol. 2, 2006, pp. 2161–2168.
- [22] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Int. J. Robot. Res.*, vol. 27, no. 6, pp. 647–665, Jun. 2008.
- [23] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in *Autonomous Robot Vehicles*, I. Cox and G. Wilfong, Eds. Springer-Verlag, 1990, pp. 167–193.
- [24] B. Kim, M. Kaess, L. Fletcher, J. J. Leonard, A. Bachrach, N. Roy, and S. Teller, "Multiple relative pose graphs for robust cooperative mapping," in *Proc. IEEE Int. Conf. Robot. and Automation*, Anchorage, Alaska, May 2010, pp. 3185–3192.
- [25] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Int. Conf. Computer Vision Theory and Application*, 2009, pp. 331–340.
- [26] A. Glover, W. Maddern, M. Reid, S. Reid, M. Milford, and G. Wyeth, "OpenFABMAP: An open source toolbox for appearance-based loop closure detection," in *Proc. IEEE Int. Conf. Robot. and Automation*, St. Paul, USA, 2012, pp. 4730–4735.