Generic Node Removal for Factor-Graph SLAM

Nicholas Carlevaris-Bianco, *Student Member, IEEE,* Michael Kaess, *Member, IEEE,* and Ryan M. Eustice, *Senior Member, IEEE*

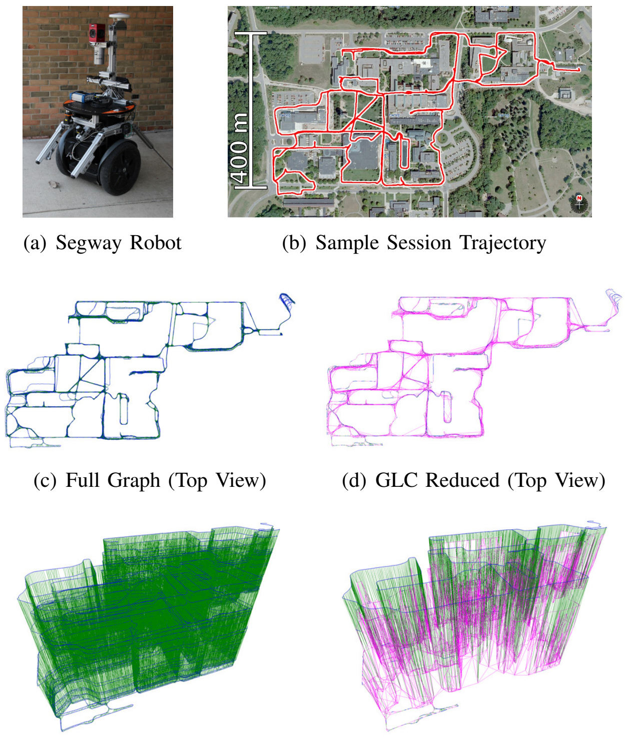
***Abstract*—This paper reports on a generic factor-based method for node removal in factor-graph simultaneous localization and mapping (SLAM), which we call generic linear constraints (GLCs). The need for a generic node removal tool is motivated by long-term SLAM applications whereby nodes are removed in order to control the computational cost of graph optimization. GLC is able to produce a new set of linearized factors over the elimination clique that can represent either the true marginalization (i.e., dense GLC), or a sparse approximation of the true marginalization using a Chow-Liu tree (i.e., sparse GLC). The proposed algorithm improves upon commonly used methods in two key ways: First, it is not limited to graphs with strictly full-state relative-pose factors and works equally well with other low-rank factors such as those produced by monocular vision. Second, the new factors are produced in a way that accounts for measurement correlation, a problem encountered in other methods that rely strictly upon pairwise measurement composition. We evaluate the proposed method over multiple real-world SLAM graphs and show that it outperforms other recently-proposed methods in terms of Kullback-Leibler divergence. Additionally, we experimentally demonstrate that the proposed GLC method provides a principled and flexible tool to control the computational complexity of long-term graph SLAM, with results shown for 34.9 h of real-world indoor-outdoor data covering 147.4 km collected over 27 mapping sessions spanning a period of 15 months.**

***Index Terms*—Simultaneous localization and mapping (SLAM), long-term autonomy, mobile robotics, factor graphs, marginalization.**

I. Introduction

Graph based simultaneous localization and mapping (SLAM) [3–9] has been demonstrated successfully over a wide variety of applications. Unfortunately, the standard graph SLAM formulation, which does not marginalize out past robot states, is not ideal for long-term applications. The robot must continually add nodes and measurements to stay localized even if it is working in a finite region, causing the size of the graph to grow with spatial extent and exploration time (Fig. 1(c) and (e)).

This paper proposes a method to address the long-term SLAM complexity challenge by allowing one to arbitrarily remove nodes from the graph, thereby reducing inference complexity and allowing for graph maintainability. We refer to this method as generic linear constraints (GLCs). We demonstrate that by removing spatially redundant nodes using GLC, the computational complexity of the graph optimization can be bounded with respect to exploration time (Fig. 1(d) and (f)).



(a) Segway Robot (b) Sample Session Trajectory

(c) Full Graph (Top View) (d) GLC Reduced (Top View)

(e) Full Graph (Time Scaled) (f) GLC Reduced (Time Scaled)

Fig. 1: SLAM graphs constructed by a Segway robot (a) after 27 mapping sessions (b) spanning a period of 15 months. The full graph without node removal ((c) and (e)) is compared with a graph from which nodes have been removed using GLC ((d) and (f)). Links include odometry (blue), 3D LIDAR scan matching (green) and generic linear constraints (magenta). The bottom row ((e) and (f)) shows an oblique view scaled by time in the z-axis, highlighting the difference between the full and GLC-reduced graphs; each layer along the z-axis represents a mapping session.

A. Previous Work

Several prior methods have been proposed to slow the rate of growth of the graph. In Ila et al. [10], an information-theoretic approach is used to add only non-redundant nodes and highly-informative measurements to the graph. This slows the rate of growth but does not bound it. In Johannsson et al. [11], new constraints are induced between existing nodes when possible, instead of adding new nodes to the graph. In this formulation the number of nodes grows only with spatial

This work was supported in part by the National Science Foundation under award IIS-0746455, by the Office of Naval Research under awards N00014-12-1-0092 and N00014-12-1-0093, and by the Naval Sea Systems Command through the Naval Engineering Education Center under award N65540-10-C-0003. Portions of this work were presented in part at the 2013 IEEE International Conference on Robotics and Automation [1], and the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems [2].

N. Carlevaris-Bianco is with the Department of Electrical Engineering & Computer Science, University of Michigan, carlevar@umich.edu.

M. Kaess is with the Robotics Institute, Carnegie Mellon University, kaess@cmu.edu.

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

extent, not with mapping duration—though the number of factors and connectivity density within the graph still grow with time.

Methods that work directly on the linearized information matrix (best suited for filtering-based SLAM solutions) include [12–14]. In Thrun et al. [12], weak links between nodes are removed to enforce sparsity. Unfortunately, this removal method causes the resulting estimate to be overconfident [15]. In Walter et al. [13], odometry links are removed in order to enforce sparsity in feature-based SLAM. Recently, Vial et al. [14] proposed an optimization-based method that minimizes the Kullback-Leibler divergence (KLD) of the information matrix while enforcing a sparsity pattern and the requirement that the estimated information is conservative. This method performs favorably in comparison with [12] and [13], but requires a large matrix inversion in order to reduce the scope of the optimization problem, limiting its online utility.

Recently, many works have proposed removing nodes from the SLAM graph as a means to control the computational complexity of the associated optimization problem [16–20]. In Konolige and Bowman [16], the environment is spatially divided into neighborhoods and then a least-recently-used criteria is used to remove nodes with the goal of keeping a small set of example views that capture the changing appearance of the environment. In Kretzschmar and Stachniss [18], nodes that provide the least information to an occupancy grid are removed. In Eade et al. [17], nodes without associated imagery are removed. In Walcott-Bryant et al. [19], "inactive" nodes that no longer contribute to the laser-based map (because the environment has changed) are removed. Finally, in Wang et al. [20], nodes are removed based on an approximation of their information contribution to the graph.

Each of the methods described in [11, 16–20] provides insight into the question of which nodes should be removed from the graph; however, they all rely upon pairwise measurement composition over full-state constraints, as described in [21], to produce a new set of factors over the elimination clique after a node is removed from the graph. Unfortunately, pairwise measurement composition has two key drawbacks when used for node removal. First, it is not uncommon for a graph to be composed of many different types of "low-rank" constraints, such as bearing-only, range-only and other partial-state constraints. In these heterogeneous cases, measurement composition, if even possible, quickly becomes complicated as the constraint composition rules for all possible pairs of measurement types must be well defined. Second, the new constraints created by measurement composition are often correlated. Ignoring this correlation leads to inconsistent estimates because measurements are double counted. In some cases it is possible to avoid double counting measurements by discarding some of the composed measurements; however, this comes at the cost of information loss. Additionally, for general graph topologies, double counting measurements may be unavoidable when using a pairwise composition scheme as illustrated by the simple graph in Fig. 2.

The exact procedure for measurement-composition-based node removal varies amongst existing methods. In [16–18] the correlation between composed measurements is ignored. In [16], all composed constraints are kept, causing fill-in within the graph. In order to preserve sparsity, a subset of the composed edges are pruned in [17] using a heuristic based on node degree. In [18], composed-edge removal is guided by a Chow-Liu tree calculated over the conditional information of the elimination clique. To avoid measurement double counting, [11] discards an odometry link and performs re-localization (along the lines of [13]). Similarly, [19] uses a maximum of two newly composed constraints at the beginning and end of a "removal chain" (a sequence of nodes to remove) to ensure connectivity without double counting measurements. In [20], only the two sequential odometry constraints are compounded, which also avoids double counting measurements.

Methods that remove nodes without measurement composition have been proposed in [1, 2, 22–24]. These methods are based upon replacing the factors in the marginalization clique with a linearized potential or a set of linearized potentials, instead of potentials produced by measurement composition. In Folkesson and Christensen [22], these linearized potentials are referred to as "star nodes." The dense formulation of our proposed GLC [1] is essentially equivalent to "star nodes" while the sparse approximate GLC replaces the dense n-nary connectivity with a sparse tree structure. In Frese [23], linearized potentials are used to remove nodes in cliques within the author's Treemap [25] algorithm.

The method recently proposed in Huang et al. [24] uses dense linear potentials similar to star-nodes and dense-GLCs to remove nodes from the graph. To perform edge sparsification, the authors formulate an optimization problem that seeks to minimize the KLD of the approximation while requiring a conservative estimate and encouraging sparsity through $\mathcal{L}_1$ regularization. This optimization problem is then applied to the linearized information matrix associated with the entire graph, which limits its applicability to relatively small problems, and prevents relinearization after sparsification. Using $\mathcal{L}_1$ regularization to promote sparsity is appealing because it does not require the sparsity pattern to be specified—instead, it automatically removes the least important edges. However, because the sparsity pattern produced is arbitrary, it is unclear how the resulting information matrix might be decomposed into a sparse set of factors, which is important if one wishes to exploit existing graph SLAM solvers such as iSAM [5, 9] or g^2o [26].

The recent work by Mazuran et al. [27] replaces the clique factors with a set of non-linear virtual measurements and then uses a numerical optimization method to find the appropriate measurement noise for each virtual measurement. This method produces similar results to GLC when removing nodes with a good linearization point and improves the KLD when removing nodes with a poor linearization as the measurement can subsequently be re-linearized. The authors also propose adding additional virtual factors in the elimination clique beyond a pairwise tree, which can further reduces the KLD. One limitation of the method is that it requires the specification of the virtual measurements and their Jacobians such that their rank is appropriate for the information in the marginalization potential. This is straightforward in homogeneous graphs with full-rank constraints; however, it is not clear how one would specify the virtual measurements in heterogeneous graphs with low-rank constraints.

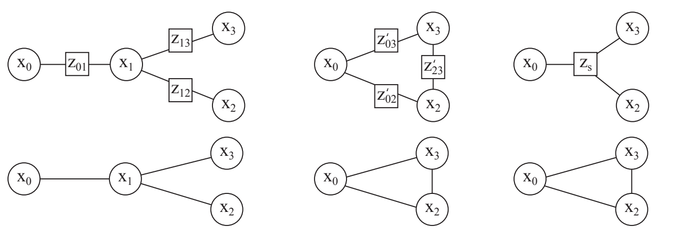
Linearized potentials representing the result of marginalization are also used in Cunningham et al. [28] to reduce bandwidth while transmitting graphs between robots in a multi-robot distributed estimation framework. Nodes that are not part of the interaction between the robots' graphs are removed from linearized potentials, and a graph of these linearized potentials, referred to as a "summarized map", is transmitted between robots.

B. Proposed Method

The aforementioned composition-based methods have many desirable properties. They produce a new set of factors using the existing factors as input, the computational complexity is only dependent upon the number of nodes and factors in the elimination clique, and the new factors can be re-linearized during subsequent optimization. Our proposed algorithm, referred to as generic linear constraint node removal, also seeks to remove nodes from the graph—retaining the desirable properties of composition-based methods, yet avoiding their pitfalls. Specifically, the algorithm was designed so that it meets the following criteria:

- The algorithm works equally well with non-full-state constraints. Constraints with lower degree of freedom (DOF) than full-state (e.g., bearing-only, range-only and partial state constraints) are handled under the same framework as full-state constraints, without special consideration.
- The new factors are produced in a way that does not double count measurement information. As we will show in §II, methods based on the pairwise composition of measurements produce pairwise constraints that are not independent, which leads to inconsistency in the graph.
- The algorithm produces a new set of independent factors using the current graph factors as input. The method does not require the full linearized information matrix as input.
- The algorithm is able to produce constraints that can represent exact node marginalization, as well as constraints that can represent a sparse Chow-Liu tree approximation of the dense marginal.
- The computational complexity of the algorithm is dependent only upon the number of nodes and factors in the elimination clique, not on the size of the graph beyond the clique.
- The algorithm does not require committing to a world-frame linearization point, rather, the new factors are parametrized in such a way as to use a local linearization that is valid independent of the global reference frame. This allows for the exploitation of methods that re-linearize during optimization (e.g., [5, 6, 9]).

This manuscript incorporates the initial technical description of GLC node removal, [1], and the evaluation of its applicability to long-term SLAM, [2]. Additionally, we provide new results over several standard datasets. The remainder of this paper is outlined as follows: In Section II we discuss the pitfalls associated with the use of measurement composition for node removal. Our proposed method is then described in Section III and experimentally evaluated in Section IV. Finally, Sections V and VI offer a discussion and concluding remarks.



(a) Original graph (b) Composition (c) Marginalization

Fig. 2: Measurement composition versus marginalization. Here node $\mathbf{x}_1$ is removed from the original graph (a). The top row shows the factor graph; the bottom row shows its Markov random field.

II. Pairwise Composition $\neq$ Marginalization

Consider the simple graph depicted in Fig. 2(a) where we show both its factor graph and Markov random field (MRF) representations. Suppose that we wish to marginalize node $\mathbf{x}_1$. Using the composition notation of [21], we can compose the pairwise measurements to produce the graph depicted in Fig. 2(b) as follows,

$$\begin{aligned}\mathbf{z}'_{02} &= h_1(\mathbf{z}_{01}, \mathbf{z}_{12}) = \mathbf{z}_{01} \oplus \mathbf{z}_{12},\\ \mathbf{z}'_{03} &= h_2(\mathbf{z}_{01}, \mathbf{z}_{13}) = \mathbf{z}_{01} \oplus \mathbf{z}_{13},\\ \mathbf{z}'_{23} &= h_3(\mathbf{z}_{12}, \mathbf{z}_{13}) = \ominus\mathbf{z}_{12} \oplus \mathbf{z}_{13}.\end{aligned} \tag{1}$$

These composed measurements are meant to capture the fully connected graph topology that develops in the elimination clique once $\mathbf{x}_1$ has been marginalized. In [17, 18], this composition graph forms the conceptual basis from which their link sparsification method then acts to prune edges and produce a sparsely connected graph. The problem with this composition is that the pairwise edges/factors in Fig. 2(b) are assumed to be independent, which they are not.

It should be clear that the composed measurements in (1) are correlated, as $\mathbf{z}'_{02}$, $\mathbf{z}'_{03}$ and $\mathbf{z}'_{23}$ share common information (e.g., $\mathbf{z}'_{02}$ and $\mathbf{z}'_{03}$ both share $\mathbf{z}_{01}$ as input), yet, if we treat these factors as strictly pairwise, we are unable to capture this correlation. Now consider instead a stacked measurement model defined as

$$\mathbf{z}_s = \begin{bmatrix}\mathbf{z}'_{02}\\ \mathbf{z}'_{03}\\ \mathbf{z}'_{23}\end{bmatrix} = h\left(\begin{bmatrix}\mathbf{z}_{01}\\ \mathbf{z}_{12}\\ \mathbf{z}_{13}\end{bmatrix}\right) = \begin{bmatrix}\mathbf{z}_{01} \oplus \mathbf{z}_{12}\\ \mathbf{z}_{01} \oplus \mathbf{z}_{13}\\ \ominus\mathbf{z}_{12} \oplus \mathbf{z}_{13}\end{bmatrix}. \tag{2}$$

Its first-order uncertainty is given as

$$\Sigma_s = \mathrm{H}\begin{bmatrix}\Sigma_{01} & 0 & 0\\ 0 & \Sigma_{12} & 0\\ 0 & 0 & \Sigma_{13}\end{bmatrix}\mathrm{H}^\top,$$

where

$$\mathrm{H} = \begin{bmatrix}\frac{\partial \mathbf{z}'_{02}}{\partial \mathbf{z}_{01}} & \frac{\partial \mathbf{z}'_{02}}{\partial \mathbf{z}_{12}} & 0\\ \frac{\partial \mathbf{z}'_{03}}{\partial \mathbf{z}_{01}} & 0 & \frac{\partial \mathbf{z}'_{03}}{\partial \mathbf{z}_{13}}\\ 0 & \frac{\partial \mathbf{z}'_{23}}{\partial \mathbf{z}_{12}} & \frac{\partial \mathbf{z}'_{23}}{\partial \mathbf{z}_{13}}\end{bmatrix}.$$

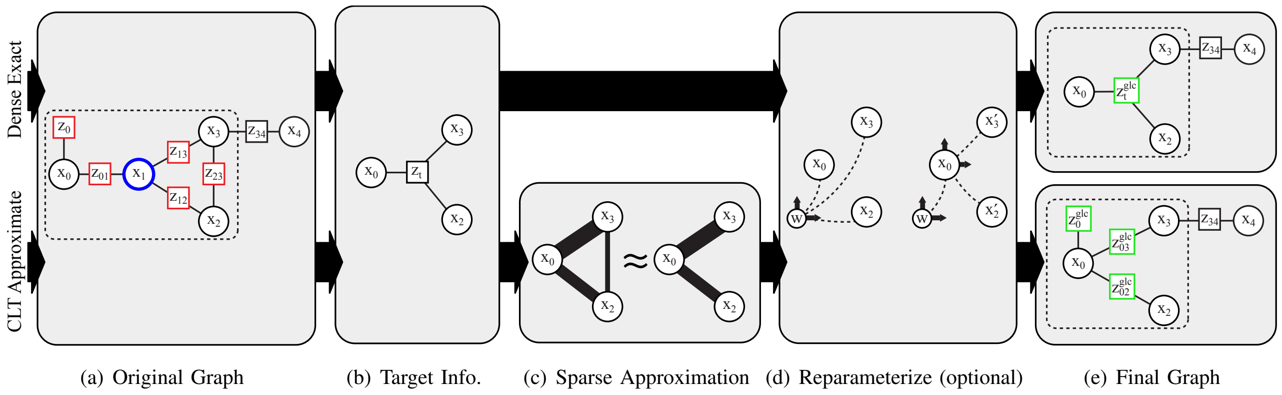


Fig. 3: GLC node removal algorithm. The dense exact version follows the top row, while the sparse CLT approximate version follows the bottom row. A sample factor graph where node $\mathbf{x}_1$ is to be removed is shown in (a). Here $\mathbf{X}_m = [\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3]$. The factors $\mathbf{Z}_m = [\mathbf{z}_0, \mathbf{z}_{01}, \mathbf{z}_{12}, \mathbf{z}_{13}, \mathbf{z}_{23}]$ (highlighted in red in (a)) are those included in calculating the target information, Λ_t, which defines a linear potential, $\mathbf{z}_t$, over the marginalization clique $\mathbf{X}_t = [\mathbf{x}_0, \mathbf{x}_2, \mathbf{x}_3]$ (b). In the case of CLT approximate node removal the original distribution associated with the target information, $p(\mathbf{X}_t|\mathbf{Z}_m)$, is approximated using the Chow-Liu maximum-mutual-information spanning tree as $p(\mathbf{x}_0|\mathbf{Z}_m)p(\mathbf{x}_2|\mathbf{x}_0, \mathbf{Z}_m)p(\mathbf{x}_3|\mathbf{x}_0, \mathbf{Z}_m)$ (c). Optionally, the potentials are reparameterized with respect to $\mathbf{x}_0$ to avoid linearization in the world-frame (d). New GLC factors are computed and inserted into the graph replacing $\mathbf{Z}_m$ (highlighted in green in (e)). Note that node removal only affects the nodes and factors within the Markov blanket of $\mathbf{x}_1$ (dashed line).

Here we see that Σ_s will be dense in order to capture the correlation between the compounded measurements. Expressing this correlation requires a trinary factor with support including all three variables. Therefore, the joint composition in (2) produces the factor graph shown in Fig. 2(c).

It is this inability of pairwise factors to capture correlation between composed measurements that causes simple compounding to be wrong. Note that the graphs in Fig. 2(b) and Fig. 2(c) have the same Markov random field representation and information matrix sparsity pattern. The difference between the binary and trinary factorization is only made explicit in the factor graph representation. It is also interesting to note that even if we were to approximate the dense connectivity with a spanning tree constructed from binary factors, as in [18], the resulting estimate would still be inconsistent as any pair of factors are correlated.

These two observations: *(i)* that composed measurements are often correlated, and *(ii)* that representing the potential of an elimination clique with n nodes requires n-nary factors, will prove important in the remainder of the paper.

III. Proposed Method

The proposed method, illustrated in Fig. 3, is summarized as follows: First, the factors that are supported by the node to be removed and the nodes in its elimination clique (Fig. 3(a)) are used to compute the linear potential induced by marginalization over the elimination clique. This potential is characterized by its distribution's information matrix, which we refer to as the target information, Λ_t (Fig. 3(b)). Next we use either *(i)* Λ_t directly to compute an exact n-nary potential that produces a marginalization-equivalent potential over the elimination clique (in the case of dense node removal), or *(ii)* approximate Λ_t as a sparse set of binary potentials that best approximate the true distribution over the elimination clique using a Chow-Liu tree (in the case of sparsified node removal, Fig. 3(c)). Before creating new GLC factors one can optionally reparameterize the variables in each potential so that the constraint will be linearized in a relative frame as opposed to a global frame (Fig. 3(d)). Finally, a new GLC factor is created for each potential and we can simply remove the marginalization node from the graph and replace its surrounding factors with the newly computed set (Fig. 3(e)).

In the following sections we derive the proposed method and describe each step in detail. This description makes use of many standard concepts from prior work in SLAM including: graphical interpretations of SLAM, the underlying least-squares problem, node removal / marginalization, graph sparsification, the manipulation of information-form multivariate Gaussian distributions, and the representation of robot poses. We recommend [5, 7, 12, 21] as background material to readers who may be less familiar with these concepts.

A. Building the Target Information

The first step in the algorithm is to correctly identify the target information, Λ_t (Fig. 3(b)). Letting $\mathbf{X}_m \subset \mathbf{X}$ be the subset of nodes including the node to be removed and the nodes in its Markov blanket, and letting $\mathbf{Z}_m \subset \mathbf{Z}$ be the subset of measurement factors that only depend on the nodes in $\mathbf{X}_m$, we consider the distribution $p(\mathbf{X}_m|\mathbf{Z}_m) \sim \mathcal{N}^{-1}(\boldsymbol{\eta}_m, \Lambda_m)$. From Λ_m we can then compute the desired target information, Λ_t, by marginalizing out the elimination node using the standard Schur-complement form. For example, in the graph shown in Fig. 3(a), to eliminate node $\mathbf{x}_1$ we would first calculate Λ_m using the standard information-form measurement update equations [12, 15] as

$$\begin{aligned}\Lambda_m = \quad & \mathrm{H}_0^\top \Lambda_0 \mathrm{H}_0 + \mathrm{H}_{01}^\top \Lambda_{01} \mathrm{H}_{01} + \mathrm{H}_{12}^\top \Lambda_{12} \mathrm{H}_{12} \\ & + \mathrm{H}_{23}^\top \Lambda_{23} \mathrm{H}_{23} + \mathrm{H}_{13}^\top \Lambda_{13} \mathrm{H}_{13},\end{aligned}$$

where H_{ij} are the Jacobians of the observation models for measurements $\mathbf{z}_{ij}$ with information matrices Λ_{ij}, and then compute the target information as

$$\Lambda_t = \Lambda_{\alpha\alpha} - \Lambda_{\alpha\beta}\Lambda_{\beta\beta}^{-1}\Lambda_{\alpha\beta}^\top,$$

where $\Lambda_{\alpha\alpha}$, $\Lambda_{\alpha\beta}$ and $\Lambda_{\beta\beta}$ are the required sub-blocks of Λ_m with $\alpha = [\mathbf{x}_0, \mathbf{x}_2, \mathbf{x}_3]$ and $\beta = [\mathbf{x}_1]$. Note that, though this example only contains unary and binary factors, general n-nary factors are equally acceptable.

While computing Λ_m one could exclude intra-clique factors that are not connected to the marginalization node, for example $\mathbf{z}_0$ and $\mathbf{z}_{23}$ in Fig. 3(a), and simply leave them in the graph. In fact the only strict requirement is that all factors which include the marginalization node be included in Λ_m. However, in §III-D we wish to sparsely approximate the marginalization clique factors, and including all intra-clique factors assures that the resulting connectivity will be sparse. For consistency we include all intra-clique factors in Λ_m throughout the algorithm and in all experimental results.

The key observation when identifying the target information is that, for a given linearization point, a single n-nary factor can recreate the potential induced by the original pairwise factors by adding the same information (i.e., Λ_m) back to the graph. Moreover, because marginalization only affects the information matrix blocks corresponding to nodes *within* the elimination clique, an n-nary factor that adds the information contained in Λ_t to the graph will induce the same potential in the graph as true node marginalization at the given linearization point.

Note that the target information, Λ_t, *is not* the conditional distribution of the elimination clique given the rest of the nodes, i.e., $p(\mathbf{x}_0, \mathbf{x}_2, \mathbf{x}_3 | \mathbf{x}_4, \mathbf{Z})$, nor is it the marginal distribution of the elimination clique, i.e., $p(\mathbf{x}_0, \mathbf{x}_2, \mathbf{x}_3 | \mathbf{Z})$. Using either of these distributions as the target information results in a wrong estimate as information will be double counted when the n-nary factor is reinserted into the graph.

It is also important to note that the constraints in $\mathbf{Z}_m$ may be purely relative and/or low-rank (e.g., bearing or range-only) and, therefore, may not fully constrain $p(\mathbf{X}_m | \mathbf{Z}_m)$. This can cause Λ_t to be singular. Additionally, some of Λ_t's block-diagonal elements may also be singular. This will require special consideration in subsequent sections.

B. Generic Linear Constraints

Having defined a method for calculating the target information, Λ_t, we now seek to produce an n-nary factor that captures the same potential. We refer to this new n-nary factor as a generic linear constraint (GLC). Letting $\mathbf{x}_t$ denote a stacked vector of the variables within the elimination clique after node removal, we begin by considering an observation model that directly observes $\mathbf{x}_t$ with a measurement uncertainty that is defined by the target information:

$$\mathbf{z}_t = \mathbf{x}_t + \mathbf{w} \;\text{ where }\; \mathbf{w} \sim \mathcal{N}^{-1}\big(\mathbf{0}, \Lambda_t\big). \tag{3}$$

Setting the measurement value, $\mathbf{z}_t$, equal to the current linearization point, $\hat{\mathbf{x}}_t$, induces the desired potential in the graph. Unfortunately, the target information, Λ_t, may not be full rank, which is problematic for optimization methods that rely upon a square root factorization of the measurement information matrix [5, 9]. We can, however, use principle component analysis to transform the measurement to a lower-dimensional representation that is full rank.

We know that Λ_t will be a real, symmetric, positive semi-definite matrix due to the nature of its construction. In general then, it has an eigen-decomposition given by

$$\Lambda_t = \begin{bmatrix}\mathbf{u}_1 & \cdots & \mathbf{u}_q\end{bmatrix} \begin{bmatrix}\lambda_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \lambda_q\end{bmatrix} \begin{bmatrix}\mathbf{u}_1^\top \\ \vdots \\ \mathbf{u}_q^\top\end{bmatrix} = \mathrm{U}\mathrm{D}\mathrm{U}^\top, \tag{4}$$

where U is a $p \times q$ matrix, D is a $q \times q$ matrix, p is the dimension of Λ_t, and $q = \mathrm{rank}(\Lambda_t)$. Letting $\mathrm{G} = \mathrm{D}^{\frac{1}{2}}\mathrm{U}^\top$ allows us to write a transformed observation model,

$$\mathbf{z}_{glc} = \mathrm{G}\mathbf{z}_t = \mathrm{G}\hat{\mathbf{x}}_t + \mathbf{w}' \;\text{ where }\; \mathbf{w}' \sim \mathcal{N}^{-1}\big(\mathbf{0}, \Lambda'\big). \tag{5}$$

Using the pseudo-inverse [29], $\Lambda_t^+ = \mathrm{U}\mathrm{D}^{-1}\mathrm{U}^\top$, and noting that $\mathrm{U}^\top\mathrm{U} = \mathrm{I}_{q\times q}$, we find that

$$\Lambda' = (\mathrm{G}\Lambda_t^+\mathrm{G}^\top)^{-1} = (\mathrm{D}^{\frac{1}{2}}\mathrm{U}^\top(\mathrm{U}\mathrm{D}^{-1}\mathrm{U}^\top)\mathrm{U}\mathrm{D}^{\frac{1}{2}})^{-1} = \mathrm{I}_{q\times q}.$$

This GLC factor will contribute the desired target information back to the graph, i.e.,

$$\mathrm{G}^\top\Lambda'\mathrm{G} = \mathrm{G}^\top\mathrm{I}_{q\times q}\mathrm{G} = \Lambda_t,$$

but is itself non-singular. This is a key advantage of the proposed GLC method; it automatically determines the appropriate measurement rank such that Λ' is $q \times q$ and invertible, and G is a $q \times p$ new observation model that maps the p-dimensional state to the q-dimensional measurement.

C. Avoiding World-Frame Linearization in GLC

In the case where the nodes involved are robot poses or landmark locations, GLC, as proposed so far, would linearize the potential with respect to the state variables in the *world-frame*. This may be acceptable in applications where a good world-frame linearization point is known prior to marginalization; however, in general, a more tenable assumption is that a good linearization point exists for the local *relative-frame* transforms between nodes within the elimination clique.

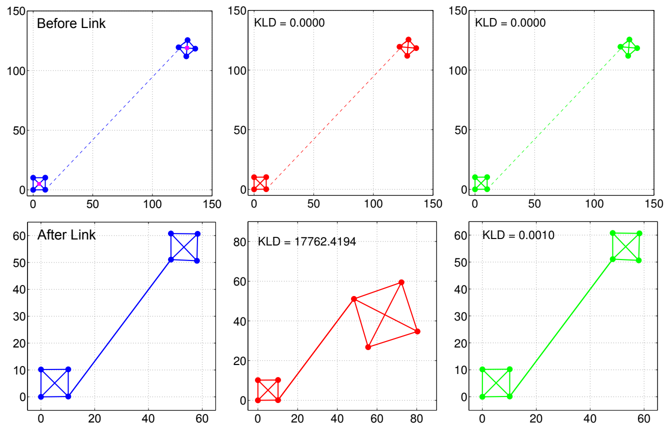
To adapt GLC so that it only locally linearizes the relative transformations between variables in the elimination clique, we first define a "root-shift" function that maps its world-frame coordinates, $\mathbf{x}_t$, to relative-frame coordinates, $\mathbf{x}_r$. Letting $\mathbf{x}_j^i$ denote the j^{th} pose in the i^{th} frame, the root-shift function for $\mathbf{x}_t$ becomes

$$\mathbf{x}_r = \begin{bmatrix}\mathbf{x}_w^1 \\ \mathbf{x}_2^1 \\ \vdots \\ \mathbf{x}_n^1\end{bmatrix} = r\left(\mathbf{x}_t\right) = r\left(\begin{bmatrix}\mathbf{x}_1^w \\ \mathbf{x}_2^w \\ \vdots \\ \mathbf{x}_n^w\end{bmatrix}\right) = \begin{bmatrix}\ominus\mathbf{x}_1^w \\ \ominus\mathbf{x}_1^w \oplus \mathbf{x}_2^w \\ \vdots \\ \ominus\mathbf{x}_1^w \oplus \mathbf{x}_n^w\end{bmatrix}. \tag{6}$$

In this function the first node is arbitrarily chosen as the root of all relative transforms. The inclusion of the inverse of the root pose, $\mathbf{x}_w^1$, is important as it ensures that the Jacobian of the root-shift operation, R, is invertible, and allows for the representation of target information that is not purely relative.

To derive, instead of starting with a direct observation of the state variables, as in (3), we instead start with their root-shifted relative transforms,

$$\mathbf{z}_r = \mathbf{x}_r + \mathbf{w}_r \;\text{ where }\; \mathbf{w}_r \sim \mathcal{N}^{-1}\big(\mathbf{0}, \Lambda_r\big). \tag{7}$$



(a) Truth (b) World-frame GLCs (c) Root-shifted GLCs

Fig. 4: Demonstration of root-shifted versus world-frame GLC factors. Depicted is a simple graph (a) that is initially constructed with two well-connected clusters connected by a highly-uncertain and inaccurate link. The center (magenta) node in each cluster is removed inducing a GLC factor over each cluster. Subsequently, a second measurement is added between the two clusters, correcting the location of the upper-right cluster, and drastically changing its world-frame linearization point. After adding the strong inter-cluster constraint, the graph with the world-frame linearized GLCs fails to converge to the correct optima (b), while the graph with root-shifted GLCs does (c). The Kullback-Leibler divergence from the true marginalization is displayed for each test.

Here, the root-shifted target information, Λ_r, is calculated using the fact that the root-shift Jacobian, R, is invertible,

$$\Lambda_r = \mathrm{R}^{-\top}\Lambda_t\mathrm{R}^{-1}. \tag{8}$$

Like the original target information, the root-shifted target information, Λ_r, may also be low-rank. Following the same principal component analysis procedure as before, we perform the low-rank eigen-decomposition $\Lambda_r = \mathrm{U}_r\mathrm{D}_r\mathrm{U}_r^\top$, which yields a new observation model,

$$\mathbf{z}_{glc_r} = \mathrm{G}_r r(\hat{\mathbf{x}}_t) + \mathbf{w}'_r \quad \text{where} \quad \mathbf{w}'_r \sim \mathcal{N}^{-1}\big(\mathbf{0}, \Lambda'_r\big), \tag{9}$$

where $\mathrm{G}_r = \mathrm{D}_r^{\frac{1}{2}}\mathrm{U}_r^\top$, and measurement information $\Lambda'_r = \mathrm{I}_{q\times q}$. Using the root-shifted linearization point to compute the measurement value, $\mathbf{z}_{glc_r} = \mathrm{G}_r r(\hat{\mathbf{x}}_t)$, will again induce the desired potential in the graph. Now, however, the advantage is that the GLC factor embeds the linearized constraint within a relative coordinate frame defined by the clique, as opposed to an absolute coordinate world-frame. Fig. 4 demonstrates this benefit.

It is important to note that this reparameterization step is optional and that it is the only step in GLC that is dependent on the parameterization of the state vector. It is also important to note that reparameterization may not even be necessary if the parameters are already defined in a relative frame as opposed to in the global frame. The root-shift reparameterization defined above is designed for graphs with nodes representing robot poses or landmark locations in the world frame, and is only one example of a possible transformation.

In cases where graph nodes represent other parameters beyond world-frame robot poses or point landmarks, it may be beneficial to reparameterize the variables that support the GLC factor using a different transformation. Any invertible reparameterization of the support variables is acceptable, allowing for large flexibility in designing reparameterizations appropriate for the user's application. In our public implementation [30] we provide a simple callback for user defined reparameterizations. This is exploited in [31], where a reparameterization is defined for use in a more complicated graph with nodes describing a piecewise-planer model of the environment in addition to robot pose nodes.

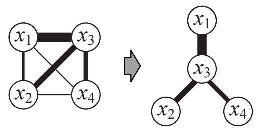


Fig. 5: Illustration of the Chow-Liu tree approximation. The magnitude of mutual information between variables is indicated by line thickness. The original distribution $p(x_1, x_2, x_3, x_4)$ (left), is approximated as $p(x_1)p(x_3|x_1)p(x_2|x_3)p(x_4|x_3)$ (right).

D. Sparse Approximate Node Removal

Exact node marginalization causes dense fill-in. As the number of marginalized nodes increases, this dense fill-in can quickly reduce the graph's sparsity and greatly increase the computational complexity of optimizing the graph [5, 9]. In [18], Kretzschmar and Stachniss insightfully propose the use of a Chow-Liu tree (CLT) [32] to approximate the individual elimination cliques as sparse tree structures.

The CLT approximates a joint distribution as the product of pairwise conditional distributions,

$$p(\mathbf{x}_1, \cdots, \mathbf{x}_n) \approx p(\mathbf{x}_1)\prod_{i=2}^{n} p(\mathbf{x}_i|\mathbf{x}_{\mathrm{p}(i)}), \tag{10}$$

where $\mathbf{x}_1$ is the root variable of the CLT and $\mathbf{x}_{\mathrm{p}(i)}$ is the parent of $\mathbf{x}_i$. The pairwise conditional distributions are selected such that the KLD between the original distribution and the CLT approximation is minimized. To construct it, the maximum spanning tree over all possible pairwise mutual information pairings is found (Fig. 5), where the mutual information between two Gaussian random vectors,

$$p(\mathbf{x}_i, \mathbf{x}_j) \sim \mathcal{N}\Big(\Big[\begin{smallmatrix}\boldsymbol{\mu}_i\\ \boldsymbol{\mu}_j\end{smallmatrix}\Big], \Big[\begin{smallmatrix}\Sigma_{ii} & \Sigma_{ij}\\ \Sigma_{ji} & \Sigma_{jj}\end{smallmatrix}\Big]\Big) \equiv \mathcal{N}^{-1}\Big(\Big[\begin{smallmatrix}\boldsymbol{\eta}_i\\ \boldsymbol{\eta}_j\end{smallmatrix}\Big], \Big[\begin{smallmatrix}\Lambda_{ii} & \Lambda_{ij}\\ \Lambda_{ji} & \Lambda_{jj}\end{smallmatrix}\Big]\Big), \tag{11}$$

is given by [33]

$$I(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2}\log\left(\frac{|\Lambda_{ii}|}{|\Lambda_{ii} - \Lambda_{ij}\Lambda_{jj}^{-1}\Lambda_{ji}|}\right). \tag{12}$$

Like [18], we leverage the CLT approximation to sparsify our n-nary GLC factors; however, our implementation of CLT-based sparsification actually implements the true CLT of the marginalization potential over the elimination clique. In [18], the maximum mutual information spanning tree is computed over the conditional distribution of the elimination clique given the remainder of the graph. This tree is then used as a heuristic to guide which edges should be composed and which edges should be excluded. There are two issues with this—first, these composed edges do not actually implement the true CLT. Second, the conditional distribution of the elimination clique is not the distribution that we wish to reproduce by our new factors (see §III-A).

We address these issues by computing the CLT distribution (10) from the target information, Λ_t, which parameterizes the distribution that we wish to approximate, and then represent the CLT's unary and binary potentials as GLC factors.

1) Chow-Liu Tree Factors: The CLT has two types of potentials, a unary potential on the root node and binary potentials between the rest of the nodes in the tree. We first consider the CLT's binary potentials, $p(\mathbf{x}_i|\mathbf{x}_{\mathrm{p}(i)})$, and in the following use $\mathbf{x}_j = \mathbf{x}_{\mathrm{p}(i)}$ as shorthand for the parent node of $\mathbf{x}_i$. We note that the target-information-derived joint marginal, $p_t(\mathbf{x}_i, \mathbf{x}_j)$, can be computed from Λ_t and written as in (11).[1] From this joint marginal, we can then easily write the desired conditional, $p_t(\mathbf{x}_i|\mathbf{x}_j) = \mathcal{N}\big(\boldsymbol{\mu}_{i|j}, \Sigma_{i|j}\big) \equiv \mathcal{N}^{-1}\big(\boldsymbol{\eta}_{i|j}, \Lambda_{i|j}\big)$, and express it as a constraint as

$$\mathbf{e} = \mathbf{x}_i - \boldsymbol{\mu}_{i|j} = \mathbf{x}_i - \Lambda_{ii}^{-1}(\boldsymbol{\eta}_i - \Lambda_{ij}\mathbf{x}_j), \tag{13}$$

where $\mathbf{e} \sim \mathcal{N}^{-1}\big(\mathbf{0}, \Lambda_{i|j}\big)$, and with Jacobian,

$$\mathrm{E} = \begin{bmatrix} \frac{\partial \mathbf{e}}{\partial \mathbf{x}_i} & \frac{\partial \mathbf{e}}{\partial \mathbf{x}_j} \end{bmatrix} = \begin{bmatrix} \mathrm{I} & \Lambda_{ii}^{-1}\Lambda_{ij} \end{bmatrix}. \tag{14}$$

Therefore, using the standard information-form measurement update, we see that this constraint adds information

$$\mathrm{E}^\top \Lambda_{i|j} \mathrm{E}, \tag{15}$$

where $\Lambda_{i|j}$ is simply the sub-block Λ_{ii}.

Treating (15) as the input target information, we can produce an equivalent GLC factor for this binary potential using the techniques described in §III-B and §III-C. Similarly, the CLT's root unary potential, $p_t(\mathbf{x}_1)$, can also be implemented as a GLC factor by using the target-information-derived marginal information, Λ_{11}, and the same techniques.

E. Implementation considerations

1) Pseudo-Inverse: As discussed in §III-A, the target information, Λ_t, is generally low rank. This is problematic for the joint marginal (11) and conditioning (13)–(14) calculations used to compute the CLT, as matrix inversions are required. To address this issue, in place of the inverse we use the generalized- or pseudo-inverse [29, §10.5], which can be calculated via an eigen-decomposition for real, symmetric, positive semi-definite matrices. For full-rank matrices the pseudo-inverse produces the same result as the true inverse, while for low rank matrices it remains well defined. Calculating the pseudo-inverse numerically requires defining a tolerance below which eigenvalues are considered to be zero. We found that our results are fairly insensitive to this tolerance and that automatically calculating the numerical tolerance using the machine epsilon produced good results. In our experiments we use $\epsilon \times n \times \lambda_{max}$ (the product of the machine epsilon, the size of the matrix, and the maximum eigenvalue) as the numerical tolerance.

[1] In this section, when we refer to marginal and conditional distributions, they are with respect to the target information, Λ_t, *not* with respect to the distribution represented by the full graph.

2) Pinning: When calculating the pairwise mutual information, the determinants of both the conditional and marginal information matrices in (12) must be non-zero, which is again problematic because these matrices are generally low-rank as calculated from the target information, Λ_t. It has been proposed to consider the product of the non-zero eigenvalues as a pseudo-determinant [29, 34] when working with singular, multivariate, Gaussian distributions. Like the pseudo-inverse, this requires determining zero eigenvalues numerically. However, we found that this can cause the mutual information computation to be numerically unstable if the matrices involved have eigenvalues near the threshold. This numerical instability causes the edges to be sorted incorrectly in some cases. This results in a non-optimal structure when the maximum mutual information spanning tree is built and, therefore, a slightly higher KLD from the true marginalization in some graphs.

Instead, we recognize that the CLT's construction requires only the ability to *sort* pairwise links by their relative mutual information (12), and not the actual value of their mutual information. A method that slightly modifies the input matrix so that its determinant is non-zero, without significantly affecting the relative ordering of the edges, would also be acceptable. Along these lines we approximate the determinant of a singular matrix using

$$|\Lambda| \approx |\Lambda + \alpha \mathrm{I}|. \tag{16}$$

This can be thought of as applying a low-certainty prior on the distribution, and we therefore refer to it as "pinning".[2] Pinning always results in a numerically stable mutual information computation, the only concern is that the relative ordering of the mutual information values remains the same. Experimentally we found the quality of the results to be less sensitive to the pinning α value than the numerical epsilon in the pseudo-determinant. We, therefore, elected to use pinning with $\alpha = 1$ in our experiments when evaluating the determinants in the pairwise mutual information (12).

F. Computational Complexity

The core operations that GLC relies on, in and of themselves, are computationally expensive. The CLT approximation has a complexity of $\mathcal{O}(m^2 \log m)$, where m is the number of nodes. Matrix operations on the information matrix with n variables, including the eigen-decomposition, matrix multiplication, and inversion operations, have a complexity of $\mathcal{O}(n^3)$. Fortunately, the input size for these operations is limited to the number of nodes within the elimination clique, which in a SLAM graph is controlled by the perceptual radius. In general, the number of nodes and variables in an elimination clique is much less than the total number of nodes in the full graph, which makes GLC's calculations easily feasible. We will see in §IV-C that the algorithm run-time is sufficiently small for real-time removal of nodes from the graph and for large batch removal of nodes.

TABLE I: Experimental Datasets

Dataset	Node Types	Factor Types	# Nodes	# Factors
Intel Lab	3-DOF pose	3-DOF odometry, 3-DOF laser scan-matching	910	4,454
Killian Court	3-DOF pose	3-DOF odometry, 3-DOF laser scan-matching	1,941	2,191
Duderstadt Center	6-DOF pose	6-DOF odometry, 6-DOF laser scan-matching	552	1,774
EECS Building	6-DOF pose	6-DOF odometry, 6-DOF laser scan-matching	611	2,134
Victoria Park	3-DOF pose, 2-DOF Landmark	3-DOF odometry, 2-DOF landmark observation	7,120	10,609
USS Saratoga	6-DOF pose	6-DOF odometry, 5-DOF monocular-vision, 1-DOF depth	1,513	5,433

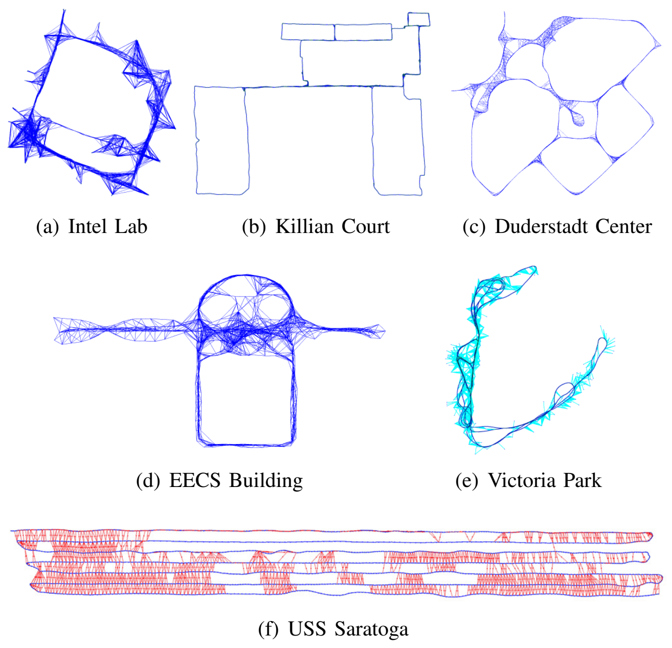


Fig. 6: Graphs used in GLC's evaluation. Blue links represent full-state (3-DOF or 6-DOF) relative-pose constraints from odometry and laser scan-matching. Red links represent 5-DOF relative-pose constraints modulo-scale from monocular vision. Cyan links represent landmark observation factors.

IV. Experimental Evaluation of GLC Node Removal

First we directly evaluate GLC node removal over a variety of SLAM graphs (summarized in Fig. 6 and Table I), including:

- Two standard 3-DOF pose-graphs, *Intel Lab* and *Killian Court*.
- Two 6-DOF pose-graphs built using data from a Segway ground robot equipped with a Velodyne HDL-32E laser scanner as the primary sensing modality, *Duderstadt Center* and *EECS Building*.
- The *Victoria Park* 3-DOF graph with poses and landmarks.
- A 6-DOF graph produced by a Hovering Autonomous Underwater Vehicle (HAUV) performing monocular SLAM for autonomous ship hull inspection [35], *USS Saratoga*.

The proposed algorithm was implemented using iSAM [9, 36] as the underlying optimization engine. The code is open-source and available for download within the iSAM repository [30]. For comparison, a dense measurement composition (MC) method as described in §II, and a sparse MC method based upon CLT-guided node removal, as proposed in [18], were also implemented.

For each graph, the original full graph is first optimized using iSAM. Then the different node removal algorithms are each performed to remove a varying percentage of nodes evenly spaced throughout the trajectory. Finally, the graphs are again optimized in iSAM.

For each experiment the true marginal distribution is recovered by obtaining the linearized information matrix from the full graph about the optimization point and performing Schur-complement marginalization. This provides a ground-truth distribution that we can directly compare the distribution after node removal against.

A summary of our results are provided in Table II, which shows the KL-divergence (normalized by the DOF of the distribution after node removal) from the true marginalization as an increasing percentage of nodes are removed from the graph. Results for dense-exact and sparse-approximate GLC are provided for all six graphs, while results for dense and sparse-approximate MC are provided only for the pose-graphs with full-state constraints. The Saratoga graph is excluded as it contains 5-DOF monocular relative-pose constraints for which MC is ill-defined.

A. Dense GLC Node Removal

We first consider the results for our method when performing exact node removal with dense fill-in. Visual examples of the resulting dense GLC graphs are shown in Fig. 7.

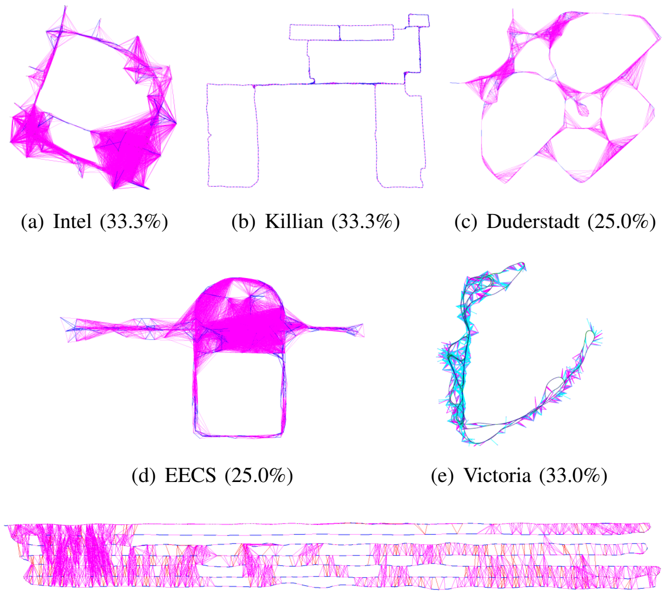
To put GLC's KLD values from Table II into perspective, we look at the case with the highest KLD, which is the Saratoga graph with 25% of nodes removed (i.e., KLD $=$ 0.016). Under these conditions the reconstructed graph has a mean error in translation and rotation of $18.9\,\mathrm{mm}$ and $3.8\,\mathrm{mrad}$, respectively, when compared to the original baseline pose-graph SLAM result. To more systematically investigate the accuracy of GLC's marginal pose uncertainties in Fig. 8 we consider the eigenvalues of the difference between the marginal covariances of the GLC-derived and the true distribution, $\mathrm{eig}(\Sigma_{ii}^{\mathrm{GLC}} - \Sigma_{ii}^{\mathrm{TRUE}})$. In the ideal case the eigenvalues of this difference will be zero, indicating perfect agreement between GLC and the true marginalization. Values larger than zero indicate conservative estimates while those less than zero indicate over-confidence. Conservative estimates are generally preferred to overconfident estimates in robotics, as overconfidence can lead to data association failure [37] and unsafe path planning and obstacle avoidance. For dense GLC we see that the eigenvalues are almost zero (note the 10^{-6} scale), indicating excellent agreement between GLC and the true marginalization. Additionally, visual examples of the

[2]This is related to the derivation of the pseudo-determinant in [34], which uses a similar form in the limit as $\alpha \to 0$.

TABLE II: Experimental Normalized KLD From True Marginalization

	Dense GLC		CLT Sparse GLC						
% Nodes Removed	25.0 %	33.3 %	25.0 %	33.3 %	50.0 %	66.6 %	75.0 %	83.3 %	87.5 %
Intel Lab	0.002	0.002	0.096	0.110	0.128	0.126	0.131	0.170	0.139
Killian Court	0.001	0.002	0.006	0.008	0.013	0.020	0.023	0.028	0.033
Duderstadt Center	0.001	0.000	0.003	0.003	0.003	0.005	0.008	0.018	0.024
EECS Building	0.003	0.002	0.005	0.005	0.004	0.010	0.017	0.035	0.049
Victoria Park	0.001	0.002	0.005	0.007	0.011	0.017	0.024	0.042	0.057
USS Saratoga	0.016	0.013	0.017	0.015	0.001	0.002	0.001	0.001	0.003

	Dense Pairwise MC		Sparse Pairwise MC						
% Nodes Removed	25.0 %	33.3 %	25.0 %	33.3 %	50.0 %	66.6 %	75.0 %	83.3 %	87.5 %
Intel Lab	1.57E3	7.19E5	0.023	0.038	0.108	0.280	0.428	0.800	1.295
Killian Court	0.01	0.02	0.005	0.007	0.013	0.023	0.031	0.042	0.048
Duderstadt Center	0.18	42.69	0.002	0.008	0.008	0.025	0.044	0.070	0.100
EECS Building	160.76	9.32E4	0.003	0.005	0.010	0.027	0.043	0.113	0.170



(a) Intel (33.3%) (b) Killian (33.3%) (c) Duderstadt (25.0%)

(d) EECS (25.0%) (e) Victoria (33.0%)

(f) USS Saratoga (25.0%)

Fig. 7: Example graphs after dense GLC node removal. New GLC factors are shown in magenta. Note that this is the MRF representation of the graph connectivity. The percentage indicates the percentage of nodes that have been removed.

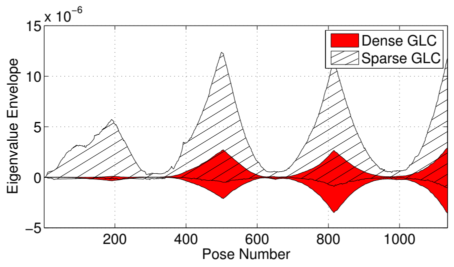
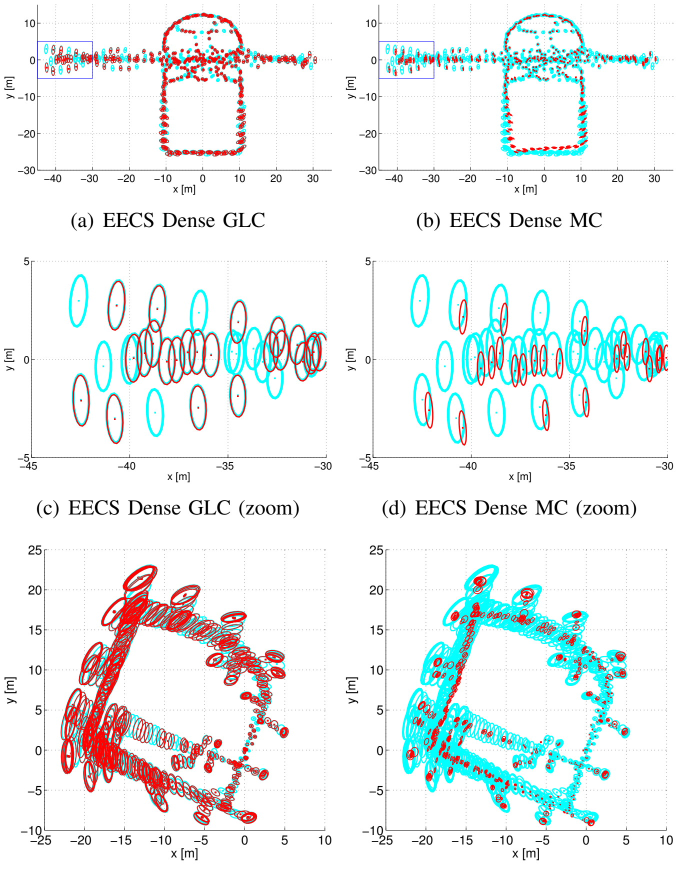


Fig. 8: Accuracy of GLC-derived marginals for the USS Saratoga dataset with 25% of nodes removed. The range of the eigenvalues of the difference between the covariances of the GLC-derived marginals and true marginals, $\text{eig}(\Sigma_{ii}^{\text{GLC}} - \Sigma_{ii}^{\text{TRUE}})$, is shown for both dense and sparse GLC. Note 10^{-6} scale.



(a) EECS Dense GLC (b) EECS Dense MC

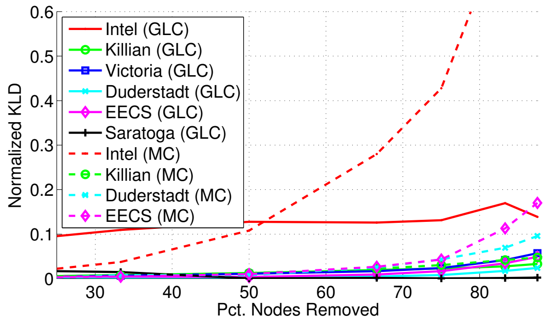
(c) EECS Dense GLC (zoom) (d) EECS Dense MC (zoom)

(e) Intel Dense GLC (f) Intel Dense MC

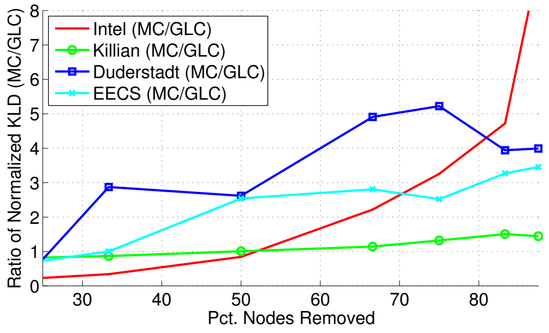
Fig. 9: Sample 3-σ uncertainty ellipses for the EECS graph and the Intel graph with 33.3% node removal using dense GLC and dense MC. The true marginalization uncertainties are shown in cyan. Note that fewer red ellipses are plotted than cyan because fewer nodes remain in the graph after node removal.

marginal covariances for the EECS and Intel graphs are shown in Fig. 9(a) and Fig. 9(e), respectively.

As more nodes are removed from the graph, dense GLC node removal quickly becomes computationally expensive due to an increase in the size of the elimination cliques. Removing nodes may take on the order of tens of seconds [1] per node. This, combined with increased optimization cost due to the dense connectivity, limits the applicability of dense node removal to applications where only a small percentage (in



(a) Normalized KLD for Sparse-Approximate Methods



(b) Ratio of Normalized KLD (MC / GLC)

Fig. 10: KLD (normalized by the DOF of the distribution after node removal) for the GLC and MC-based sparse approximate node removal methods are shown in (a). In (b) the ratio of KLD between MC and GLC is plotted, highlighting that, in most cases, as more nodes are removed MC induces several times higher KLD than GLC.

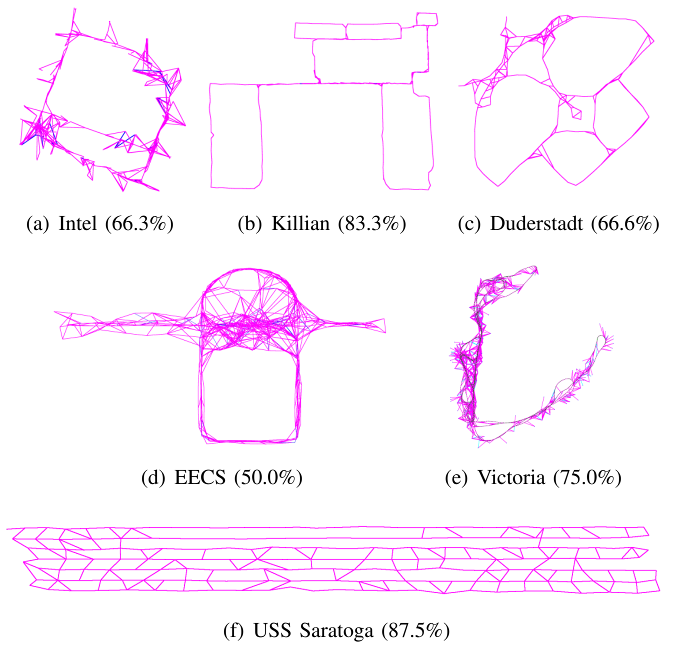
our experiments around one-third to one-half) of nodes are removed.

Considering the results for dense MC, Table II shows that it performs quite poorly—as more nodes are removed, the KLD quickly increases. This is because dense pairwise MC fails to properly track the correlation that develops between composed measurements (as demonstrated in §II); thus, the higher the connectivity in the graph, the more measurement information gets double counted when compounding. This results in overconfidence as well as a shift in the optimal mean (Fig. 9(b) and Fig. 9(f)).

B. CLT Sparse-Approximate GLC Node Removal

Next we consider the results for sparse-approximate GLC node removal. Table II shows that in many graphs, including Killian, Duderstadt, EECS, and USS Saratoga, the KLD for sparse-approximate GLC is only slightly worse than that of dense-exact GLC—indicating that very little graph information is lost due to the CLT approximation. Fig. 10 illustrates the KLD for the sparse-approximate versions of GLC and MC, again normalizing the KLD by the number of degree of freedom in the graph after node removal. Visual examples for sparsification on the graphs are shown in Fig. 11. Examples of the marginal covariances for the EECS and Intel graphs are show in Fig. 12.

Considering the results for sparse MC, Table II shows that, unlike dense MC, sparse MC performs reasonably well when removing a smaller percentage of nodes. This is because information double counting during measurement composition



(a) Intel (66.3%) (b) Killian (83.3%) (c) Duderstadt (66.6%)

(d) EECS (50.0%) (e) Victoria (75.0%)

(f) USS Saratoga (87.5%)

Fig. 11: Example graphs after CLT sparse GLC node removal. New GLC factors are shown in magenta. The percentage indicates the percentage of nodes that have been removed.

accumulates to a lesser extent than in the dense case because of sparsification. However, as the percentage of removed nodes increases, we see that sparse MC produces less accurate and more inconsistent results than sparse GLC. This is illustrated in Fig. 10(b), which highlights the ratio in the normalized KLD between MC and GLC, and in Fig. 12 and Fig. 13, which compare the marginal covariances of the distributions.

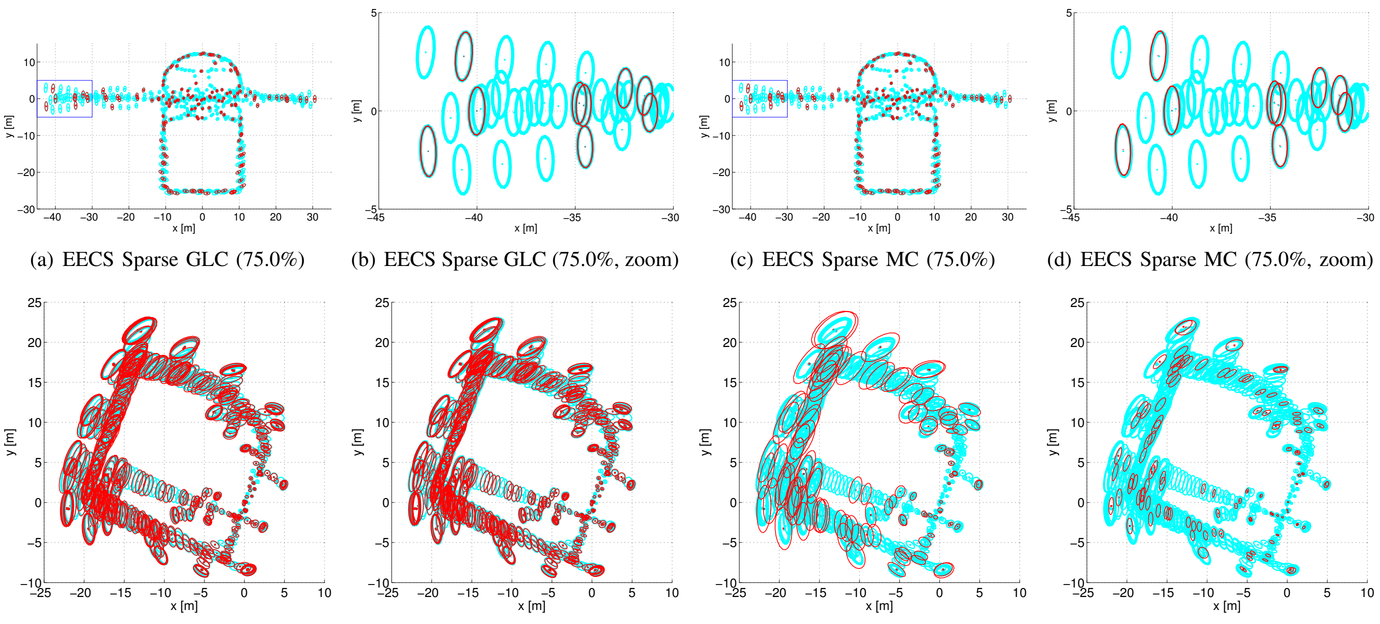
It is important to note that the proposed method is not guaranteed to be conservative. This is due to the fact that the CLT approximation simply seeks to produce the minimum KLD and does not guarantee a conservative approximation. To address this we have recently proposed an extension to GLC [38] that provides a guaranteed-conservative approximation, while still producing a low KLD.

In the case of the Intel graph, MC achieves a significantly better KLD than GLC when removing a small percentage of nodes. This is due to the fact that when removing a small number of nodes, GLC is slightly conservative (Fig. 12(e)), while MC's inconsistency coincidentally yields a less conservative estimate with a better KLD (Fig. 12(f)). As more nodes are removed this trend is continued, with GLC remaining conservative and producing a better KLD (Fig. 12(g)), while MC becomes very inconsistent (Fig. 12(h)).

Unlike dense node removal, sparse GLC maintains graph sparsity and keeps elimination clique size small. This results in fast node removal on the order of tens of milliseconds per node [1, 2].

C. Evaluating GLC for Long-Term SLAM

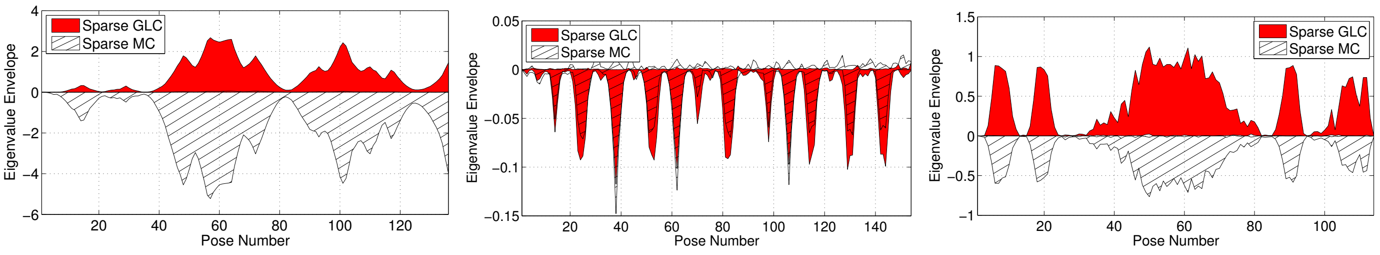
Having seen that GLC node removal provides an accurate way to remove nodes from a factor graph, we now wish to evaluate its effectiveness when integrated within a SLAM system. To do so we consider a multi-session scenario where



(a) EECS Sparse GLC (75.0%) (b) EECS Sparse GLC (75.0%, zoom) (c) EECS Sparse MC (75.0%) (d) EECS Sparse MC (75.0%, zoom)

(e) Intel Sparse GLC (33.3%) (f) Intel Sparse MC (33.3%) (g) Intel Sparse GLC (87.5%) (h) Intel Sparse MC (87.5%)

Fig. 12: Sample 3-σ uncertainty ellipses for the EECS graph with 75% node removal and for the Intel graph with 33.3% and 87.5% node removal using sparse GLC and MC. The true marginalization uncertainties are shown in cyan. Note that fewer red ellipses are plotted than cyan because fewer nodes remain in the graph after node removal. The percentage indicates the percentage of nodes that have been removed.



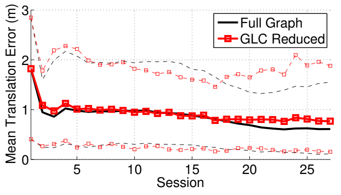
(a) Duderstadt (75.0% Removed) (b) EECS (75.0% Removed) (c) Intel (87.5% Removed)

Fig. 13: Comparison of marginal distributions between sparse GLC and sparse MC, for the Duderstadt, EECS, and Intel graphs. The range of eigenvalues of the difference between the covariances of the approximate-node-removal marginals and true marginals, $\mathrm{eig}(\Sigma_{ii}^{\mathrm{EST}} - \Sigma_{ii}^{\mathrm{TRUE}})$, is shown for both sparse GLC and sparse MC. In the ideal case the range will be zero, indicating perfect agreement between the approximate and the true marginals. Values larger than zero indicate conservative estimates while those less than zero indicate over-confidence. The results show that sparse GLC remains conservative while sparse MC is overconfident for the Duderstadt and Intel graphs. In the case of the EECS graph both methods produce estimates that are occasionally overconfident.

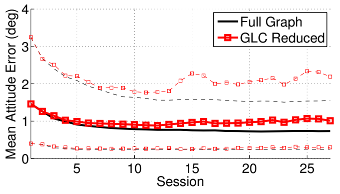
the robot repeatedly performs SLAM in discrete sessions. Under these conditions node removal can be performed between sessions as a batch operation. In an attempt to produce a graph that has a complexity dictated primarily by spatial extent and not by mapping duration, we remove spatially-redundant nodes based on a simple criteria that attempts to keep nodes with high node degree and nodes that have been recently added to the graph. This node removal scheme, and others including online methods that remove nodes as the robot explores, were originally proposed and evaluated in [2]. For long-term SLAM applications we focus on the CLT-based sparse-approximate version of GLC node removal as it is most appropriate in situations where many more nodes are removed than kept. Using the dense-exact version of GLC node removal in these circumstances would produce graphs with a very high node degree, and therefore, a high computationally complexity because of the loss of sparsity.

We performed experiments using data collected over the course of 15 months using a Segway robotic platform (Fig. 1). Data was collected over 27 trials (approximately bi-weekly) between January 8th, 2012 and April 5th, 2013 by manually driving the robot through the University of Michigan's North Campus. Data was collected both indoors and outdoors, at varying times of the day and in the presence of dynamic elements including people and cars. Additionally, the dataset contains several large construction projects. Each trial through the environment is on average $1.29\ \mathrm{h}$ in duration and $5.5\ \mathrm{km}$ in length totaling $34.9\ \mathrm{h}$ of data covering $147.4\ \mathrm{km}$ (Fig. 1). Graph constraints are derived from odometry, 3D LIDAR scan matching [39] and, when available, consumer-grade global positioning system (GPS). A ground-truth graph was created from all trajectories without node removal and with the addition of constraints from a highly accurate real-time kinematic (RTK) GPS system.

The reduced graph at the end of the last full run is compared with a full graph from which no nodes have been removed in

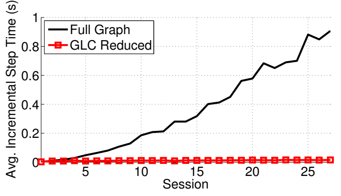


(a) Translation Error

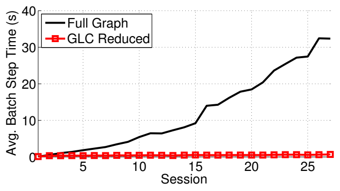


(b) Attitude Error

Fig. 14: Mean error for translation ($\sqrt{\delta_x^2+\delta_y^2+\delta_z^2}$) and attitude ($\sqrt{\delta_r^2+\delta_p^2+\delta_h^2}$) with respect to RTK-based ground-truth at the end of each mapping session for the full graph from which no nodes were removed, and the GLC reduced graph. 5% and 95% percentile bounds are denoted with dashed lines.



(a) Incremental Updates



(b) Batch Updates

Fig. 15: Mean CPU time for incremental and batch iSAM optimization update steps, and for GLC node removal, in seconds.

Fig. 1. In the bottom row, by scaling the z-axis according to time, we can clearly see the effects of node removal. We see that the most recent session is well connected to the previous session with some sparse connectivity to older nodes in the graph.

1) Reduced Graph Error and Computational Complexity: First, we consider the translation and attitude error of the reduced graph from the ground-truth. We include the full graph that was built without node removal as a baseline. We see the GLC reduced graph produces estimates with error similar to the full graph (Fig. 14). Though the graph produced with GLC node removal has a similar error to the full graph, it is vastly less computationally complex. We see in Fig. 16 that GLC node removal limits the number of nodes and factors to be essentially constant as only small additions to the spatial extent of the map are made after the first session, never exceeding 4,000 nodes or 15,000 factors. In comparison, the full graph grows linearly ending with over 46,000 nodes and 200,000 factors. We also see that the sparsity of the measurement Cholesky Factor, R, is higher than that of the full graph, but still quite sparse, with fill-in less than 0.4%.

As new nodes and factors are added to the graph, iSAM performs two different types of updates; an incremental update, where the solution is updated without relinearization, and a batch update, where the solution is relinearized and solved. In our experiments the batch optimization update was called every 50 incremental updates. In Fig. 15, we see that in the full graph, the computation time for incremental and batch update steps grows super-linearly, while for the GLC reduced graph they remain roughly constant. The time to remove a node using GLC is also relatively constant and on the order of 10 ms [2].

The total processing time for the 34.9 h of logged data, including graph optimization, node removal, data association and scan matching, took 58.7 h for the full graph. When using the proposed complexity management scheme the total processing times was reduced to 6.3 h, which is 5.5 times faster than real-time.

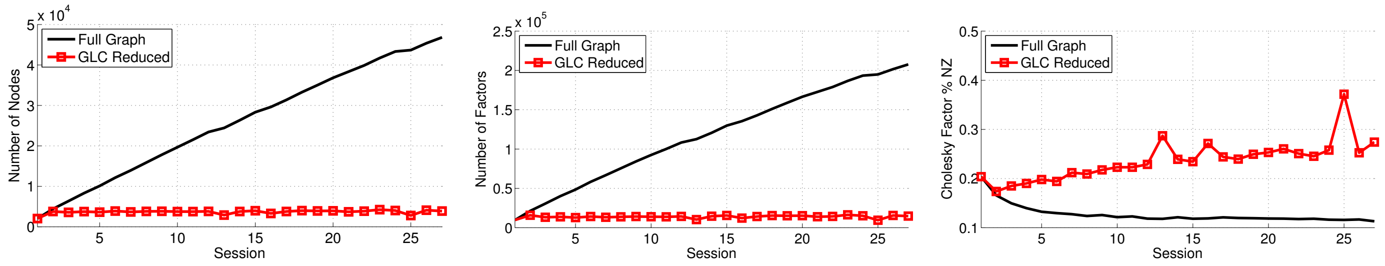
2) Distribution Comparison: In the previous experiment, because nodes are removed from the graph between each session the robot will make different data association decisions from those in the full graph, resulting in fundamentally different distributions. In order to isolate the effects of GLC, we wish to directly compare the distribution produced by repeatedly applying sparse-approximate GLC node removal to a full distribution derived using the exact same measurements, from which no nodes have been removed. This can be done by accumulating the measurements from each session in the GLC-reduced graph into one large graph. As in §IV-B, we also compare results with the sparse measurement composing method based upon CLT-guided node removal, as proposed in [18]. The results of this comparison are shown in Fig. 17. Here we see that repeatedly applying sparse approximate GLC node removal will produce a difference in the estimates from the full graph, though the difference remains low, both in terms of mean (Fig. 17(a) and (b)) and KLD (Fig. 17(c)).

By looking at the eigenvalues of the difference between the marginal covariances in the reduced and full graph, we can see that the GLC reduced graph produces a more accurate distribution (eigenvalue range closer to zero) than MC (Fig. 17(d)). In this case the range of eigenvalues for GLC is positive indicating a conservative estimate.

V. Discussion and Future Work

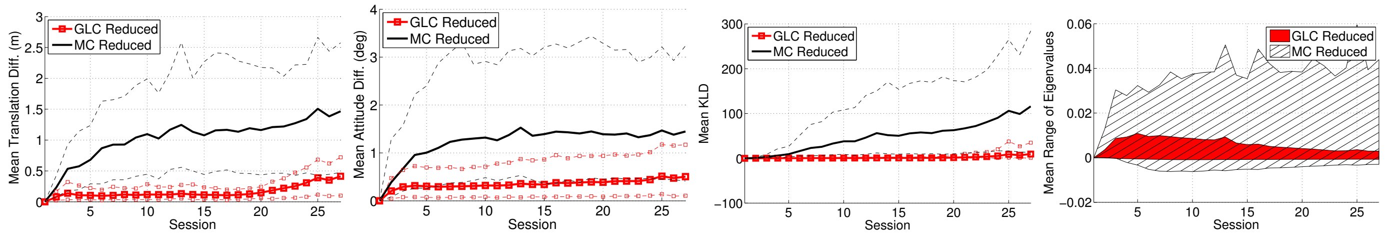
When considering the application of the proposed method, there are a few things to consider, some of which we hope to address in future work:

- When performing GLC, a good linearization point for the relative transforms within the elimination clique must exist. This affects when it is appropriate to remove nodes, especially if performing online node removal. The graph should be optimized as well as possible before node removal. Often it is desirable to remove well established or "mature" nodes from the graph, instead of nodes that have been recently instantiated and are highly uncertain. Note, however, that this is not a function of node age but rather whether the graph is sufficiently constrained and optimized to provides a good linearization point.
- Because the target information is often low rank, we use "pinning" to compute the mutual information when building the CLT and therefore, cannot guarantee that this yields a minimum KLD from the true distribution (though our experimental results show that we achieve a significantly lower KLD than other state-of-the-art methods).
- The CLT approximation itself is not guaranteed to be conservative and therefore sparse-approximate GLC node removal does not guarantee a conservative estimate. In fact, our results showed that CLT-based GLC sparse approximation can be either slightly conservative (Fig. 8 and Fig. 13(a) and (c)), or slightly over-confident



(a) Number of Nodes (b) Number of Factors (c) Graph Sparsity

Fig. 16: Long-term SLAM graph complexity. The full graph grows unbounded in both number of nodes and factors while the GLC-reduced graph remains roughly constant after the first session ((a) and (b)). The GLC-reduced graph remains quite sparse throughout the experiment, though slightly less so than the full graph (c).



(a) Translation Error (b) Attitude Error (c) KL Divergence (d) $\mathrm{eig}(\Sigma_{ii}^{\mathrm{GLC}} - \Sigma_{ii}^{\mathrm{TRUE}})$

Fig. 17: Long-term SLAM graph distribution comparison. Here, we compare the estimated distributions using GLC node removal with the estimated distributions using the same measurements but without node removal. Translation error (a) is defined as $\sqrt{\delta_x^2 + \delta_y^2 + \delta_z^2}$ and attitude error (b) as $\sqrt{\delta_r^2 + \delta_p^2 + \delta_h^2}$. The average KLD between resulting marginal covariances for each node are shown in (c). By looking at the eigenvalues of the difference between marginal covariances (d) we can see that the GLC reduced graph produces more accurate marginal uncertainties than MC. 5% and 95% percentile bounds are denoted with dashed lines.

(Fig. 13(b)). While our proposed GLC method avoids inconsistency pitfalls associated with measurement compounding, and accurately recreates the CLT, it may still be slightly overconfident if the CLT approximation cannot represent all of the true correlation within the clique. We have found experimentally that the CLT performs well on most graphs, and only results in noticeable overconfidence in graphs with large, dense cliques. In this regard, the methods proposed in [14], [24], and our recent work [38], which optimize the KLD of a sparse distribution while enforcing a consistency constraint, provide some insight into a way forward toward this end.

- When removing a set of nodes it is important to note that the order in which they are removed affects the resulting graph connectivity. Experimentally, we found that removing long chains of nodes sequentially sometimes produced large star shaped trees in the graph. To avoid this, sets of nodes were removed in a randomized order in all experiments. The variable elimination ordering problem [40] is well studied for dense node removal. The application and adaptation of existing variable elimination ordering strategies for node removal with sparse connectivity could further improve the performance of GLC-based complexity management schemes.

VI. Conclusions

We presented a factor-based method for node removal in a wide variety of SLAM graphs. This method can be used to alleviate some of the computational challenges in performing inference over long-term graphs by reducing the graph size and density. The proposed method is able to represent either exact marginalization, or a sparse approximation of the true marginalization, in a consistent manner over a heterogeneous collection of constraints. We experimentally evaluated the proposed method over multiple real-world SLAM graphs and showed that it outperformed other state-of-the-art methods in terms of Kullback-Leibler divergence. Additionally, the use of the proposed method in long-term complexity management schemes was experimentally validated.

References

[1] 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.
[2] ——, “Long-term simultaneous localization and mapping with generic linear constraint node removal,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Nov. 2013, pp. 1034–1041.
[3] F. Lu and E. Milios, “Globally consistent range scan alignment for environment mapping,” *Autonomous Robots*, vol. 4, pp. 333–349, 1997.
[4] S. Thrun and M. Montemerlo, “The graph SLAM algorithm with applications to large-scale mapping of urban structures,” *Int. J. Robot. Res.*, vol. 25, no. 5-6, pp. 403–429, 2006.
[5] F. Dellaert and M. Kaess, “Square root SAM: Simultaneous localization and mapping via square root information smoothing,” *Int. J. Robot. Res.*, vol. 25, no. 12, pp. 1181–1203, 2006.
[6] E. Olson, J. Leonard, and S. Teller, “Fast iterative alignment of pose graphs with poor initial estimates,” in *Proc. IEEE Int. Conf. Robot. and Automation*, Orlando, FL, USA, May 2006, pp. 2262–2269.
[7] R. M. Eustice, H. Singh, and J. J. Leonard, “Exactly sparse delayed-state filters for view-based SLAM,” *IEEE Trans. Robot.*, vol. 22, no. 6, pp. 1100–1114, 2006.
[8] K. Konolige and M. Agrawal, “FrameSLAM: From bundle adjustment to real-time visual mapping,” *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1066–1077, 2008.
[9] M. Kaess, A. Ranganathan, and F. Dellaert, “iSAM: Incremental smooth-

ing and mapping," *IEEE Trans. Robot.*, vol. 24, no. 6, pp. 1365–1378, 2008.

[10] V. Ila, J. M. Porta, and J. Andrade-Cetto, "Information-based compact pose SLAM," *IEEE Trans. Robot.*, vol. 26, no. 1, pp. 78–93, 2010.

[11] H. Johannsson, M. Kaess, M. Fallon, and J. J. Leonard, "Temporally scalable visual SLAM using a reduced pose graph," in *Proc. IEEE Int. Conf. Robot. and Automation*, Karlsruhe, Germany, May 2013, pp. 54–61.

[12] S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte, "Simultaneous localization and mapping with sparse extended information filters," *Int. J. Robot. Res.*, vol. 23, no. 7/8, pp. 693–716, 2004.

[13] M. R. Walter, R. M. Eustice, and J. J. Leonard, "Exactly sparse extended information filters for feature-based SLAM," *Int. J. Robot. Res.*, vol. 26, no. 4, pp. 335–359, 2007.

[14] J. Vial, H. Durrant-Whyte, and T. Bailey, "Conservative sparsification for efficient and consistent approximate estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, San Francisco, CA, USA, Sep. 2011, pp. 886–893.

[15] R. Eustice, M. Walter, and J. Leonard, "Sparse extended information filters: insights into sparsification," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Edmonton, Alberta, Canada, Aug. 2005, pp. 3281–3288.

[16] K. Konolige and J. Bowman, "Towards lifelong visual maps," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, St. Louis, MO, USA, Oct. 2009, pp. 1156–1163.

[17] E. Eade, P. Fong, and M. E. Munich, "Monocular graph SLAM with complexity reduction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Taipei, Taiwan, Oct. 2010, pp. 3017–3024.

[18] H. Kretzschmar and C. Stachniss, "Information-theoretic compression of pose graphs for laser-based SLAM," *Int. J. Robot. Res.*, vol. 31, pp. 1219–1230, 2012.

[19] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard, "Dynamic pose graph SLAM: Long-term mapping in low dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Vilamoura, Portugal, Oct. 2012, pp. 1871–1878.

[20] Y. Wang, R. Xiong, Q. Li, and S. Huang, "Kullback-leibler divergence based graph pruning in robotic feature mapping," in *Proc. European Conf. Mobile Robotics*, Barcelona, Spain, Sep. 2013, pp. 32–37.

[21] 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.

[22] J. Folkesson and H. Christensen, "Graphical SLAM—a self-correcting map," in *Proc. IEEE Int. Conf. Robot. and Automation*, New Orleans, LA, USA, April 2004, pp. 383–390.

[23] U. Frese, "Efficient 6-DOF SLAM with treemap as a generic backend," in *Proc. IEEE Int. Conf. Robot. and Automation*, Rome, Italy, Apr. 2007, pp. 4814—4819.

[24] G. Huang, M. Kaess, and J. J. Leonard, "Consistent sparsification for graph optimization," in *Proc. European Conf. Mobile Robotics*, Barcelona, Spain, Sep. 2013, pp. 150–157.

[25] U. Frese, "Treemap: an O(Log N) algorithm for simultaneous localization and mapping," in *Spatial Cognition IV*, C. Freksa, Ed. Springer Verlag, 2004.

[26] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. and Automation*, Shanghai, China, May 2011, pp. 3607–3613.

[27] M. Mazuran, T. G. Diego, S. Luciano, and W. Burgard, "Nonlinear graph sparsification for SLAM," in *Proc. Robot.: Sci. & Syst. Conf.*, Berkeley, CA, USA, Jul. 2014, pp. 1–8.

[28] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. and Automation*, Karlsruhe, Germany, May 2013, pp. 5200–5207.

[29] C. R. Rao and S. K. Mitra, *Generalized Inverse of Matrices and its Applications*. John Wiley & Sons, 1971.

[30] 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.

[31] P. Ozog and R. M. Eustice, "Toward long-term, automated ship hull inspection with visual SLAM, explicit surface optimization, and generic graph-sparsication," in *Proc. IEEE Int. Conf. Robot. and Automation*, Hong Kong, China, Jun. 2014, pp. 3832–3839.

[32] C. Chow and C. N. Liu, "Approximating discrete probability distributions with dependence trees," *IEEE Trans. on Info. Theory*, vol. 14, pp. 462–467, 1968.

[33] A. Davison, "Active search for real-time vision," in *Proc. IEEE Int. Conf. Comput. Vis.*, Beijing, China, Oct. 2005, pp. 66–73.

[34] T. P. Minka, "Inferring a Gaussian distribution," MIT Media Lab, Tech. Rep., 2001.

[35] 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, 2012.

[36] M. Kaess and F. Dellaert, "Covariance recovery from a square root information matrix for data association," *Robot. and Autonomous Syst.*, vol. 57, pp. 1198–1210, 2009.

[37] J. Neira and J. Tardos, "Data association in stochastic mapping using the joint compatibility test," *IEEE Trans. Robot. Autom.*, vol. 17, no. 6, pp. 890–897, 2001.

[38] N. Carlevaris-Bianco and R. M. Eustice, "Conservative edge sparsification for graph SLAM node removal," in *Proc. IEEE Int. Conf. Robot. and Automation*, Hong Kong, China, Jun. 2014, pp. 854–860.

[39] M. Magnusson, "The three-dimensional normal-distributions transform – an efficient representation for registration, surface analysis, and loop detection," Ph.D. dissertation, Örebro University, 2009, Örebro Studies in Technology 36.

[40] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Nicholas Carlevaris-Bianco (S'10) received the B.S. degree in Electrical Engineering from Grand Valley State University, Allendale, MI in 2007, and the M.S. degree in Electrical Engineering from the University of Michigan, Ann Arbor, MI, in 2011. Currently, he is pursuing the Ph.D. degree with the Department of Electrical Engineering, University of Michigan. His research interest include long-term mapping and navigation for autonomous robots.

Michael Kaess (S'02–M'08) received the Ph.D. (2008) and M.S. (2002) degrees in computer science from the Georgia Institute of Technology, Atlanta, GA. Currently, he is an Assistant Research Professor in the Robotics Institute at Carnegie Mellon University. Previously he has been a research scientist and postdoctoral associate in the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL). His research focuses on robot perception for navigation and autonomy.

Ryan M. Eustice (S'00–M'05–SM'10) received the B.S. degree in Mechanical Engineering from Michigan State University, East Lansing, MI in 1998, and the Ph.D. degree in Ocean Engineering from the Massachusetts Institute of Technology/Woods Hole Oceanographic Institution Joint Program, Woods Hole, MA, in 2005. Currently, he is an Associate Professor with the Department of Naval Architecture and Marine Engineering, University of Michigan, Ann Arbor, with joint appointments in the Department of Electrical Engineering and Computer Science, and in the Department of Mechanical Engineering. His research interests include autonomous navigation and mapping, computer vision and image processing, robot perception, and marine and mobile robotics.