2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Decoupled, Consistent Node Removal and Edge Sparsification for Graph-based SLAM

Kevin Eckenhoff, Liam Paull, and Guoquan Huang

Abstract— **Graph-based SLAM approaches have had success recently despite suffering from ever-increasing computational costs due to the need of optimizing over the entire robot trajectory. To address this issue, in this paper, we advocate the decoupling of marginalization (node removal) and sparsification (edge reduction) to allow for short-term retention of dense factors induced by marginalization while enabling us to spread the computation of these two operations over time. In particular, we analytically show that during marginalization, the correct choice of linearization points in constructing dense marginal factors is to use the relative (local), instead of global, state estimates in the Markov blanket of the marginalized node, which has lacked a general consensus in the literature. Furthermore, during sparsification, we determine an online sparse topology through sparsity-regularized convex optimization, which guides us to construct consistent sparse factors to best approximate the original dense factors across the Markov blanket. The proposed approach is tested extensively on both 2D and 3D public datasets and shown to perform competitively to the state-of-the-art algorithms.**

I. Introduction

In order for robots to persistently navigate and operate in unknown environments, it is essential to perform accurate localization and mapping in real time. To this end, graph-based optimization methods have recently become popular due to their increased accuracy as well as efficiency [1]–[3]. These approaches store the entire history of robot poses (and landmark positions in the case of feature-based SLAM) as nodes in the graph, while measurements between nodes are represented as edges (factors). At every time step, a batch maximum-a-posteriori (MAP) estimate is sought by formulating and solving an equivalent nonlinear least-squares (NLS) problem. Unfortunately, the problem size is unbounded due to the non-stop growth of the graph, which necessitates systematical reduction of the graph so as not to exceed what available resources permit.

In particular, two recent methods – Generic Linear Constraints (GLC) [4] and Nonlinear Factor Recovery (NFR) [5], [6] – were introduced to reduce graph size by marginalizing out a subset of nodes immediately followed by a sparsification of the resulting dense graph. This coupled scheme is unable to take advantage of all the information contained in the dense factors induced by marginalization, as they are immediately approximated through sparsification. Moreover, due to the fact that all the computations of both marginalization and sparsification are lumped together into one single time step, this coupling may not be computationally feasible for systems of limited processing power. Note also that during sparsification, both approaches [4], [5] offer (tree-based) sparse topologies, which may not be the best among all possible structures.

To mitigate the aforementioned issues, in this paper, building upon our prior work [7], we *decouple* the two computationally intensive processes, marginalization of nodes and sparsification of edges, by postponing sparsification to a later time after marginalization. By doing so, we are able to *relinearize* all the remaining measurements after marginalization and better utilize the accuracy of the dense factors inferred by marginalization. Splitting these two processes also allows us to spread the computational burden over time. Lastly, during sparsification, a sparsity-regularized convex optimization is formulated to determine *online* the sparse topology which will then guide the construction of resulting sparse factors that are to be used in future optimization. In particular, the main contributions of this work are the following:

- We perform marginalization and sparsification over a *subgraph* including *only* the factors incident to the corresponding marginalized nodes (i.e., the Markov blanket of the marginalized nodes) [8], rather than the whole graph as in our prior work [7]. During marginalization, we formulate the maximum likelihood estimation (MLE) problem in the frame of reference of one of the nodes in the subgraph, which yields the local, relative estimates that are optimal (up to linearization errors) and thus consistent. Note that these *local* state estimates, instead of the current *global* estimates as in [4], are used as linearization points for evaluating the induced marginal pdf (i.e., computing the corresponding mean and covariance/information matrix). This distribution provides prior information for the remaining nodes in subsequent optimization in the form of inferred nonlinear measurements in the global frame.
- We *decouple* sparsification from marginalization because not only can we keep the dense factors induced by marginalization in our graph for as long as desired (depending on the available computational resource) to improve estimation accuracy, but also spread the computational overhead over time. During sparsification, we first update the dense distribution using the intra-clique factors (i.e., the edges between remaining nodes in the Markov blanket). Note that these constraints are *excluded* from the dense factor calculations to allow for relinearization of these measurements in subsequent steps before sparsification takes place. We then formu-

K. Eckenhoff and G. Huang are with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, USA. Email: {keck,ghuang}@udel.edu

L. Paull is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. Email: lpaull@csail.mit.edu

late an ℓ_1-regularized convex optimization on the information matrix across the Markov blanket to find a sparse approximation. The *structure* of the resulting optimal sparse information matrix serves as the guidance for constructing a set of sparse inferred measurements that best capture the information contained in the Markov blanket and thus are used to replace the dense factors.

II. Related Work

To date, many algorithms are available to reduce the computational cost of graph-based SLAM. For example, incremental approaches such as iSAM [9] and iSAM2 [10] reuse previous computations of square root information matrix to improve efficiency. However, as graphs become denser and/or larger, they still suffer from increasing computational complexity. Other solutions process a subset of graph data in order to reduce the problem size. For instance, sliding window filters [11], [12] solve an optimization problem over a constant-size time window of states using only the measurements involved in that window. Keyframe-based approaches [13], [14] solve over a (heuristically) selected subgraph at each time step.

To reduce computational requirements for solving graph-based SLAM, focus has shifted to reducing the size of the graph. In particular, compact pose SLAM [15] adds only non-redundant nodes and highly informative loop-closing constraints to the graph. This method, however, simply discards unused poses and nodes, constituting a loss of information. Eade et al. [16] construct relative-pose measurements to remove nodes from the graph, but fail to account for the correlations between these new measurements. This arises because the new constraints are built geometrically from the discarded edges, which may cause the same information (i.e., same edges) to be used multiple times. Marginalization is a standard approach to remove nodes from a graph while retaining information contained in the discarded edges. Nerurkar et al. [17] marginalize out non-keyframes from which cost functions involving keyframes are derived. Since marginalization causes density in the graph, further approximation is employed to reduce this fill-in.

Recent research efforts seek to couple marginalization (node removal) and sparsification (edge reduction) in the same step to reduce computational complexity. Vial et al. [18] introduce a technique of consistent sparsification based on the conservative minimization of the Kullback-Leibler Divergence (KLD). However, this method does not recover individual factors, and therefore cannot be directly used in the sequential formulation. As briefly discussed in the preceding section, the GLC [4] method applies marginalization across the Markov blanket of the marginalized node but *with* intra-clique measurements, which yields n-nary linear constraints to approximate the blanket. However, these linear factors are evaluated at the current *global* state estimates as linearization points, which may result in inconsistent measurements. In contrast, the NFR approach [5], [6] allows for *any* type of inferred new factors (i.e., not only linear ones as in [4]). The information matrix of these factors is found by minimizing the (KLD) between the approximate and orignal marginal distributions. Besides using the global state estimates as linearization points in computing the mean and information of the new factors, the authors have also empirically tested the local state estimates as linearization points, which, however, has not been rigorously justified. Moreover, the sparse topologies such as tree-based structures used by both approaches [4]–[6] may not be optimal for other levels of sparsity. To address these issues, our proposed approach decouples sparsification from marginalization to better utilize the dense factors induced by marginalization as well as the intra-clique factors in the Markov blanket. We determine online a sparse structure based on sparsity-regularized convex optimization which guides us in constructing new, convservative sparse factors. In addition, we show that the proper way of choosing linearization points in this process is to use local state estimates.

III. Graph-based SLAM

The full SLAM problem is often formulated using the factor graph [2], where the edge (or constraint) between two nodes takes the following generic form:

$$\mathbf{z}_{ij} = \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \mathbf{n}_{ij} \tag{1}$$

where $\mathbf{h}_{ij}$ is a measurement function between the i-th and j-th nodes (states), $\mathbf{x}_i$ and $\mathbf{x}_j$, which maps the poses and/or positions into a measurement, $\mathbf{z}_{ij}$; and $\mathbf{n}_{ij}$ is zero-mean white Gaussian noise that corrupts the measurement, i.e., $\mathbf{n}_{ij} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}_{ij}^{-1})$. With this factor graph, we seek to solve for MLE of all the nodes in the graph:[1]

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \; p(\mathbf{z}|\mathbf{x}) = \arg\max_{\mathbf{x}} \prod_{(i,j)\in \text{supp}(\mathbf{z})} p(\mathbf{z}_{ij}|\mathbf{x}_i, \mathbf{x}_j) \tag{2}$$

where $\mathbf{z}$ is the stacked vector of all measurements, $\mathbf{x}$ is the set of all the nodes, and $\text{supp}(\mathbf{z})$ is the set of node pairs induced by $\mathbf{z}$. In the above expression, we have employed the common assumption that measurements are independent. With Gaussian noise, the above MLE problem (2) is equivalent to the following NLS problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \sum_{(i,j)\in \text{supp}(\mathbf{z})} \frac{1}{2} ||\mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j)||^2_{\mathbf{\Lambda}_{ij}} \tag{3}$$

where $||\mathbf{r}||^2_{\mathbf{\Lambda}} = \mathbf{r}^\top \mathbf{\Lambda} \mathbf{r}$ is the squared Mahalanobis distance (energy norm). To solve this problem, iterative algorithms such as Gauss-Newton are often used. In particular, at the k-th iteration, we wish to first solve for the error state $\delta\mathbf{x}$:

$$\delta\mathbf{x}^{(k)} = \arg\min_{\delta\mathbf{x}} \sum_{(i,j)\in \text{supp}(\mathbf{z})} \frac{1}{2} ||\mathbf{z}_{ij} - \mathbf{h}_{ij}(\hat{\mathbf{x}}_i^{(k)}, \hat{\mathbf{x}}_j^{(k)}) - \mathbf{H}_{ij}^{(k)} \delta\mathbf{x}||^2_{\mathbf{\Lambda}_{ij}} \tag{4}$$

where $\mathbf{H}_{ij}^{(k)} = \frac{\partial \mathbf{h}_{ij}}{\partial \mathbf{x}}|_{\hat{\mathbf{x}}^{(k)}}$ is the measurement Jacobian evaluated at the current state estimate. The solution to this problem is then used to update the state estimate: $\hat{\mathbf{x}}^{(k+1)} = \hat{\mathbf{x}}^{(k)} + \delta\mathbf{x}^{(k)}$. This process is repeated until convergence. It is important to note that, because all nodes are stored in the graph, the problem experiences unbounded growth of map size. This implies that a low-latency estimator may quickly

[1] Note that if there is a prior available, this SLAM problem becomes a batch MAP estimation problem [12].

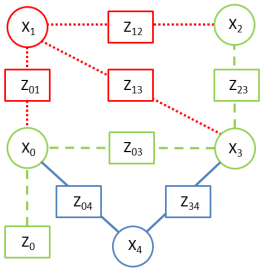


Fig. 1. A toy example to illustrate the node/edge partitions based on the Markov blanket of $\mathbf{x}_1$ that is to be marginalized. Thus, the nodes are partitioned as: $\mathbf{x}_m = \mathbf{x}_1$, $\mathbf{x}_b = \left[\mathbf{x}_0^T, \mathbf{x}_2^T, \mathbf{x}_3^T\right]^T$ and $\mathbf{x}_r = \mathbf{x}_4$. The constraints incident to $\mathbf{x}_1$ are $\mathbf{z}_m = \{\mathbf{z}_{01}, \mathbf{z}_{12}, \mathbf{z}_{13}\}$ shown in red dotted lines, the intra-clique constraints are $\mathbf{z}_c = \{\mathbf{z}_0, \mathbf{z}_{03}, \mathbf{z}_{23}\}$ in green dashed, and the remaining ones are $\mathbf{z}_r = \{\mathbf{z}_{04}, \mathbf{z}_{34}\}$ in blue solid.

become prohibitive. We thus seek to reduce the graph size by systematically removing nodes and sparsifying edges.

IV. Node Removal via Marginalization over Markov Blanket

In this section, we explain in detail how to remove nodes in a consistent manner by performing marginalization over the associated Markov blanket. In particular, we analytically show that the best choice of linearization points in this process is the *local* state estimates (that are attained by processing the local measurements only in the Markov blanket), which appears to be largely overlooked in the literature.

Suppose that we seek to marginalize out the node(s) denoted by $\mathbf{x}_m$ in order to reduce the size of the graph. Marginalization over all nodes is prohibitively expensive, and so we instead perform this operation over a much smaller subgraph. Specifically, we first construct the Markov blanket of the marginalized node $\mathbf{x}_m$, which is the smallest set of nodes that renders $\mathbf{x}_m$ conditionally independent of all other nodes in the graph [8]. The Markov blanket is an undirected graph consisting of all immediate neighboring nodes denoted by $\mathbf{x}_b$. We further denote by $\mathbf{x}_r$ the remaining nodes in the graph other than $\mathbf{x}_m$ and $\mathbf{x}_b$. Thus, we have: $\mathbf{x} = \begin{bmatrix}\mathbf{x}_m^T & \mathbf{x}_b^T & \mathbf{x}_r^T\end{bmatrix}^T$. We accordingly partition the measurement set $\mathbf{z}$ into: $\mathbf{z} = \{\mathbf{z}_b, \mathbf{z}_r\} = \{\mathbf{z}_m, \mathbf{z}_c, \mathbf{z}_r\}$, where $\mathbf{z}_b = \{\mathbf{z}_m, \mathbf{z}_c\}$ includes the all measurement constraints in the Markov blanket, and is further partitioned as the edges (constraints) $\mathbf{z}_m$ incident to $\mathbf{x}_m$, and the intra-cliques factors $\mathbf{z}_c$; and $\mathbf{z}_r$ represents all the remaining constraints. Fig. 1 visualizes the partitionings of nodes and edges.

Once the Markov blanket is built, we perform marginalization of $\mathbf{x}_m$ over $\mathbf{z}_m$ *only*, thus resulting in significant computational savings due to the smaller size of the Markov blanket than that of the entire graph. This process generates a prior distribution across the nodes in the Markov blanket, $\mathbf{x}_b$, for future optimization. In a Gaussian scenario (which typically is the case for SLAM), this pdf takes the form:

$$p(\mathbf{x}_b|\mathbf{z}_m) = \int_{\mathbf{x}_m} p(\mathbf{x}_b, \mathbf{x}_m|\mathbf{z}_m)\mathrm{d}\mathbf{x}_m \approx \mathcal{N}(\hat{\mathbf{x}}_b, \mathbf{\Lambda}_t^{-1}) \quad (5)$$

Note that this distribution encapsulates *all* the information contained in $\mathbf{z}_m$ about the nodes of the Markov blanket, thus providing all necessary prior information about $\mathbf{x}_b$ for future optimization – that is, this marginalization process results in *no* information loss for the given measurements except for linearization errors.

A. Prior Distribution of $\mathbf{x}_b$ after Marginalization

In order to determine the normal distribution (5) (i.e., the mean and covariance), we should solve the MAP estimation problem with respect to $\mathbf{x}_b$ using measurements $\mathbf{z}_m$, i.e., $\max p(\mathbf{x}_b|\mathbf{z}_m)$. However, since in general there is no prior for $\mathbf{x}_b$ and $\mathbf{x}_m$, we formulate and solve the following *local* MLE over the Markov blanket [see (3)]:

$$\{\hat{\mathbf{x}}_b, \hat{\mathbf{x}}_m\} = \arg\max_{\mathbf{x}_b, \mathbf{x}_m} p(\mathbf{z}_m|\mathbf{x}_b, \mathbf{x}_m) =$$
$$\arg\min_{\mathbf{x}_b, \mathbf{x}_m} \sum_{(i,j)\in\mathrm{supp}(\mathbf{z}_m)} \frac{1}{2}||\mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j)||^2_{\mathbf{\Lambda}_{ij}} \quad (6)$$

Note that the solution of the above problem renders the optimal value $\hat{\mathbf{x}}_b$ that has been calculated using *only* the measurements attached to $\mathbf{x}_m$ (i.e., $\mathbf{z}_m$). To obtain the information (or covariance) matrix of (5), we first notice that the information (Hessian) matrix of (6) is computed by:

$$\mathbf{\Lambda} = \sum_{(i,j)\in\mathrm{supp}(\mathbf{z}_m)} \mathbf{H}_{ij}^\top \mathbf{\Lambda}_{ij} \mathbf{H}_{ij} =: \begin{bmatrix}\mathbf{\Lambda}_{mm} & \mathbf{\Lambda}_{bm}^\top \\ \mathbf{\Lambda}_{bm} & \mathbf{\Lambda}_{bb}\end{bmatrix} \quad (7)$$

where the measurement Jacobians $\mathbf{H}_{ij}$ are evaluated at the *local* MLE estimates (see Lemma 4.1). The above information matrix is decomposed according to the dimensions of $\mathbf{x}_m$ and $\mathbf{x}_b$. Now, the target marginal information matrix $\mathbf{\Lambda}_t$ can be found via the Schur complement:

$$\mathbf{\Lambda}_t = \mathbf{\Lambda}_{bb} - \mathbf{\Lambda}_{bm}\mathbf{\Lambda}_{mm}^{-1}\mathbf{\Lambda}_{bm}^\top \quad (8)$$

Once the prior pdf of $\mathbf{x}_b$ (5) is determined, the MLE problem (2) is approximated by the following *NLS* problem:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}||\hat{\mathbf{x}}_b - \mathbf{x}_b||^2_{\mathbf{\Lambda}_t} + \sum_{(i,j)\in\mathrm{supp}(\mathbf{z}_r, \mathbf{z}_c)} \frac{1}{2}||\mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j)||^2_{\mathbf{\Lambda}_{ij}} \quad (9)$$

The prior distribution can therefore be thought of as a linear inferred factor which approximately captures the information contained in the discarded measurements $\mathbf{z}_m$ about the remaining states.

Lemma 4.1: If the mean $\hat{\mathbf{x}}_b$, measurement Jacobians $\mathbf{H}_{ij}$ (7), and thus the target information matrix $\mathbf{\Lambda}_t$ (8), are evaluated at the local MLE estimates (6), then the marginalized-NLS problem (9) best approximates the original non-marginalized MLE problem (3). If the global estimates (or indeed any other linearization point) is used, then (9) becomes suboptimal.

Proof: We show this from a cost function perspective. Based on the node and edge partitions (e.g., see Fig. 1), we decompose the cost function, $c(\mathbf{x})$, of the original non-marginalized MLE problem (3) into the costs associated with the factors attached to the marginalized nodes, and all other factors, respectively, i.e.,

$$c(\mathbf{x}) = c_m(\mathbf{x}_m, \mathbf{x}_b) + c_r(\mathbf{x}_b, \mathbf{x}_r) \quad (10)$$

Thus, we have:

$$\min_{\mathbf{x}} c(\mathbf{x}) = \min_{\mathbf{x}_b, \mathbf{x}_r}\left(\min_{\mathbf{x}_m} c(\mathbf{x}_m, \mathbf{x}_b, \mathbf{x}_r)\right) =$$
$$\min_{\mathbf{x}_b, \mathbf{x}_r}\left(c_r(\mathbf{x}_b, \mathbf{x}_r) + \min_{\mathbf{x}_m} c_m(\mathbf{x}_m, \mathbf{x}_b)\right) \quad (11)$$

When first solving the minimization of $c_m(\mathbf{x}_m, \mathbf{x}_b)$ [see (6)], due to the nonlinearity of measurements, the second-order Taylor-series approximation to $c_m(\mathbf{x}_m, \mathbf{x}_b)$ is employed:

$$c_m(\mathbf{x}_m, \mathbf{x}_b) \simeq c_m(\hat{\mathbf{x}}_m, \hat{\mathbf{x}}_b) + \mathbf{g}^T \begin{bmatrix} \mathbf{x}_m - \hat{\mathbf{x}}_m \\ \mathbf{x}_b - \hat{\mathbf{x}}_b \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_m - \hat{\mathbf{x}}_m \\ \mathbf{x}_b - \hat{\mathbf{x}}_b \end{bmatrix}^T \mathbf{\Lambda} \begin{bmatrix} \mathbf{x}_m - \hat{\mathbf{x}}_m \\ \mathbf{x}_b - \hat{\mathbf{x}}_b \end{bmatrix} \quad (12)$$

where the gradient vector $\mathbf{g} = \begin{bmatrix} \mathbf{g}_{mm} \\ \mathbf{g}_{mb} \end{bmatrix}$ and the Hessian matrix $\mathbf{\Lambda} = \begin{bmatrix} \mathbf{\Lambda}_{mm} & \mathbf{\Lambda}_{bm}^\top \\ \mathbf{\Lambda}_{bm} & \mathbf{\Lambda}_{bb} \end{bmatrix}$ are partitioned in conformity with the dimensions of $\mathbf{x}_m$ and $\mathbf{x}_b$, and are evaluated at the *linearization points* $\hat{\mathbf{x}}_m$ and $\hat{\mathbf{x}}_b$ [see (7)]. We note that since the cost function (12) is quadratic with respect to $\mathbf{x}_m$, its value can be obtained in terms of $\mathbf{x}_b$:

$$\mathbf{x}_m = \hat{\mathbf{x}}_m - \mathbf{\Lambda}_{mm}^{-1} \left(\mathbf{g}_{mm} + \mathbf{\Lambda}_{bm}^\top (\mathbf{x}_b - \hat{\mathbf{x}}_b) \right) \quad (13)$$

And substitution in (12) yields the minimum value of c_m:

$$\min_{\mathbf{x}_m} c_m(\mathbf{x}_m, \mathbf{x}_b) \simeq \zeta + \mathbf{g}_t^T (\mathbf{x}_b - \hat{\mathbf{x}}_b) + \frac{1}{2} (\mathbf{x}_b - \hat{\mathbf{x}}_b)^T \mathbf{\Lambda}_t (\mathbf{x}_b - \hat{\mathbf{x}}_b) \quad (14)$$

where ζ is a constant independent of $\mathbf{x}_b$ and $\mathbf{x}_m$, and

$$\mathbf{g}_t = \mathbf{g}_{mb} - \mathbf{\Lambda}_{bm} \mathbf{\Lambda}_{mm}^{-1} \mathbf{g}_{mm} \quad (15)$$

$$\mathbf{\Lambda}_t = \mathbf{\Lambda}_{bb} - \mathbf{\Lambda}_{bm} \mathbf{\Lambda}_{mm}^{-1} \mathbf{\Lambda}_{bm}^\top \quad (16)$$

With (14) and (11), the minimization of the cost function $c(\mathbf{x})$, i.e., the original MLE problem (3), is (approximately) equivalent to the minimization of the following cost function:

$$c_r'(\mathbf{x}_b, \mathbf{x}_r) = \mathbf{g}_t^T (\mathbf{x}_b - \hat{\mathbf{x}}_b) + \frac{1}{2} (\mathbf{x}_b - \hat{\mathbf{x}}_b)^T \mathbf{\Lambda}_t (\mathbf{x}_b - \hat{\mathbf{x}}_b) + \sum_{(i,j) \in \mathrm{supp}(\mathbf{z}_r, \mathbf{z}_c)} \frac{1}{2} ||\mathbf{z}_{ij} - \mathbf{h}_{ij}(\mathbf{x}_i, \mathbf{x}_j)||^2_{\mathbf{\Lambda}_{ij}} \quad (17)$$

If the linearization point used in computing the gradient and Hessian is the local minimum across the Markov blanket [i.e., the (sub-) optimal solution of the local MLE problem (6)], then the gradient term vanishes ($\mathbf{g}_t = \mathbf{0}$), and thus the minimization of (17) becomes equivalent to the NLS problem (9). If the *global* linearization point is used, then in general $\mathbf{g}_t \neq \mathbf{0}$, and thus the returned cost should be (17). This completes the proof. ■

B. Relative MLE Formulation

The measurements of the Markov blanket typically provides only *relative* information about the nodes, and thus the local MLE problem (6) will be under-constrained if the global state estimates are sought. In order to fully constrain the problem, as in our prior work [7], we reparametrize to solve for *relative* state estimates. This is achieved by (arbitrarily) choosing a node in the Markov blanket and then shifting the MLE problem (6) into its frame of reference.

To ease the ensuing derivations, we consider the toy example in Fig. 1, where the Markov blanket of $\mathbf{x}_1$ consists of nodes $\mathbf{x}_0$, $\mathbf{x}_2$, and $\mathbf{x}_3$, and reformulate the MLE problem (6) with respect to the local frame of reference of node $\mathbf{x}_0$:

$$\{{}^0\hat{\mathbf{x}}_i\}_{i=1}^3 = \arg \max_{\{{}^0\mathbf{x}_i\}_{i=1}^3} p({\mathbf{z}_m}|{}^0\mathbf{x}_1, {}^0\mathbf{x}_2, {}^0\mathbf{x}_3) \quad (18)$$

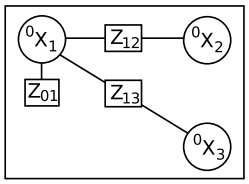


Fig. 2. The relative, local MLE formulation for the toy example of Fig. 1. By shifting the frame of reference into that of $\mathbf{x}_0$, the problem becomes fully constrained, allowing for the local MLE solution.

where ${}^0\mathbf{x}_i$ denotes the node $\mathbf{x}_i$ expressed in the frame of node $\mathbf{x}_0$. The topology of this relative problem is shown in Fig. 2. When performing marginalization of $\mathbf{x}_1$, the prior pdf, $\mathcal{N}\left(\begin{bmatrix} {}^0\hat{\mathbf{x}}_2 \\ {}^0\hat{\mathbf{x}}_3 \end{bmatrix}, {}^0\mathbf{\Lambda}_t^{-1} \right)$, can be determined similar to (6)-(8). In subsequent optimization, this distribution will contribute to the first term in (9) by $\frac{1}{2} \left|\left| \begin{bmatrix} {}^0\hat{\mathbf{x}}_2 \\ {}^0\hat{\mathbf{x}}_3 \end{bmatrix} - \begin{bmatrix} \mathbf{x}_2 \ominus \mathbf{x}_0 \\ \mathbf{x}_3 \ominus \mathbf{x}_0 \end{bmatrix} \right|\right|^2_{{}^0\mathbf{\Lambda}_t}$, where $\mathbf{x}_i \ominus \mathbf{x}_j$ refers to that $\mathbf{x}_i$ is transformed from the global frame to the local frame of $\mathbf{x}_j$ (i.e., ${}^j\mathbf{x}_i$).

C. Remarks

Note that because an accurate global linearization point may not be available, GLC [4] also shifts the frame of reference of the Markov blanket to that of an arbitrary node. However, since the inferred factors are constructed using the shifted *global* estimates of the nodes, based on Lemma 4.1, this may not be the best approximation of the original MLE problem, thus leading to suboptimal results. NFR uses local linearization points, but finds them in the global frame by fixing the estimate of a node. Note also that as evident from (6), the subgraph chosen for marginalization need only include the factors incident to the marginalized node – that is, $\mathbf{z}_m$ only, *without* the intra-clique factors $\mathbf{z}_c$. However, GLC [4] and NFR [6] choose to include these factors during marginalization for better sparsity, dismissing the ability to *relinearize* these measurements in subsequent optimization.

In addition, while we here focus on the case of full-state measurements, the same methodology is applicable to partial-state measurements such as bearing-only observations. This requires a different parametrization to achieve minimal representation of the local MLE (e.g., see [19]).

Lastly, although we have shown in our proof that gradient terms are ignored if (9) is used on the global linearization points, the full cost has been utilized in [12].

V. Edge Sparsification

Marginalization, while reducing the number of nodes in a graph, actually has an adverse effect on computation due to the addition of dense factors that destroy sparsity in the system. We must be able to *sparsify* our graph if our algorithm is to be tractable for large-scale problems. In this section we propose a method to explicitly solve for a sparse topology that will guide us to build new sparse factors to replace the dense measurements, rather than choosing a tree-based structure as in [4], [5].

A. Determination of Sparse Topology

Our proposed method selects a sparse information matrix, $\mathbf{\Omega}$, to approximate the local, marginal one, ${}^L\mathbf{\Lambda}_t =: \mathbf{\Sigma}^{-1}$, by

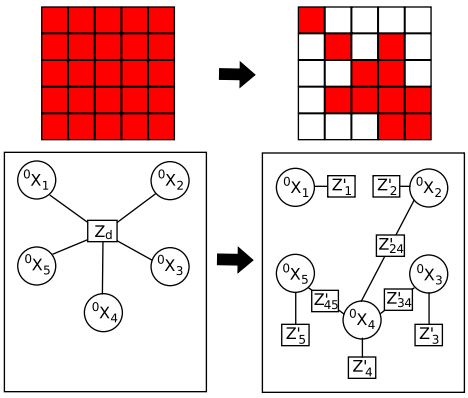


Fig. 3. Illustration of ℓ_1-regularized KLD minimization as a means for finding a sparse topology. First, the dense information is sparsified, yielding a matrix with much less connectivity. The structure of this information serves as a guide for building the topology of the desired sparse subgraph.

formulating the following ℓ_1-regularized KLD minimization (see [7]). Note that hereafter the left superscript "L" denotes the local, shifted frame of reference (see Section IV-B).

$$\min_{\mathbf{\Omega}} \ \langle \mathbf{\Omega}, \mathbf{\Sigma} \rangle - \log\det(\mathbf{\Omega}) + \lambda ||\mathbf{\Omega}||_1 \ \text{ s.t. } \mathbf{\Omega} \succeq \mathbf{0} \tag{19}$$

where $\langle *, * \rangle$ denotes the matrix inner product, and λ is a parameter which is used to control the sparsity of $\mathbf{\Omega}$. Optimization is performed subject to the constraint that the solution is positive semidefinite. Note that in our prior work [7], a similar ℓ_1-regularized minimization was employed, while being constrained by conservativeness of the desired sparse information matrix Ω (i.e. $\mathbf{\Omega} \preceq {}^L\mathbf{\Lambda}_t$). We do *not* enforce the consistency constraint here, because we only exploit the structure of the resulting information matrix $\mathbf{\Omega}$ to build the inferred, new factors, instead of the actual values of $\mathbf{\Omega}$, as in [7], for future optimization. Instead, consistency will be enforced when finding the actual sparse factors (see Section V-B). To solve (19), we employ an Alternating Direction Method of Multipliers (ADMM) method [20].

When the ADMM solver converges, the solution will be a sparse approximation of the target information matrix. Based on the fact that measurements between nodes introduce non-zero elements to corresponding entries of the information matrix, we conservatively search $\mathbf{\Omega}$ for off-diagonal sub-blocks with Frobenius norms above some threshold. If this criterion is met, an inferred measurement is added between the corresponding nodes. In addition, local prior measurements (connected to the origin node of the shifted frame of reference) are added onto each node of the graph to ensure that the diagonal elements can be captured. In this way, the structure of the approximate information matrix $\mathbf{\Omega}$ serves as a connectivity guide for our inferred subgraph (see Fig. 3).

It is obvious from the optimization problem (19) that the sparsity of the resulting topology is heavily dependent on the choice of sparsity parameter λ. In particular, we observe that off-diagonal block zeros tend not to appear if the highest magnitude entry in the corresponding covariance off-diagonal block is higher than the sparsity parameter. We therefore choose as a penalty parameter $\lambda = \alpha \overline{||\mathbf{\Sigma}_{ij}||}_\infty$ where α is a parameter used to control the sparsity of the solution, and $\overline{||\mathbf{\Sigma}_{ij}||}_\infty$ is the average infinity norm of the off-diagonal blocks in the true covariance. To limit the effect of large Markov blanket sizes, we choose $\lambda = \alpha \overline{||\mathbf{\Sigma}_{ij}||}_\infty n^\beta$, where n is the size of the Markov blanket in the local frame of reference, and β is a parameter used to control how much the problem size is penalized.

B. Construct Consistent Sparse Factors

Once the topology of our desired sparse subgraph is determined, we now construct sparse factors that not only obey the found topology but also best capture the original local target information ${}^L\mathbf{\Lambda}_t$. Specifically, suppose the inferred factors assume the following form:

$$\breve{\mathbf{z}}_{ij} = \breve{\mathbf{h}}_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \breve{\mathbf{n}}_{ij} \tag{20}$$

where $\breve{\mathbf{h}}(\cdot)$ is a measurement function that we are free to choose, and is often chosen to be the relative-pose model. In the above expression, $\breve{\mathbf{n}}_{ij} \sim \mathcal{N}(\mathbf{0}, \breve{\mathbf{\Lambda}}_{ij}^{-1})$ is the Gaussian noise, whose information matrix $\breve{\mathbf{\Lambda}}_{ij}$ needs to be determined.

Based on (20), we first construct the expected values of these inferred measurements and the corresponding Jacobians using the local relative state estimates, ${}^L\hat{\mathbf{x}}$, as the linearization points (see Section IV-B), i.e.,

$$\hat{\breve{\mathbf{z}}}_{ij} = \breve{\mathbf{h}}_{ij}({}^L\hat{\mathbf{x}}_i, {}^L\hat{\mathbf{x}}_j), \ \text{ and } \breve{\mathbf{H}}_{ij} = \frac{\partial \breve{\mathbf{h}}_{ij}}{\partial \mathbf{x}}|_{\mathbf{x}={}^L\hat{\mathbf{x}}} \tag{21}$$

To determine $\breve{\mathbf{\Lambda}}_{ij}$ for each of the new inferred measurements, it is desirable that all the new measurements do not contain extra information than the original dense factors, i.e, they should be conservative (consistent). To this end, similar to [6], we formulate an optimization problem minimizing the KLD between the true and approximated distribution, while enforcing a consistency constraint:

$$\min_{\breve{\mathbf{\Lambda}}} \ \langle \breve{\mathbf{H}}^T \breve{\mathbf{\Lambda}} \breve{\mathbf{H}}, \mathbf{\Sigma} \rangle - \log\det(\breve{\mathbf{H}}^T \breve{\mathbf{\Lambda}} \breve{\mathbf{H}}) \tag{22}$$

$$\text{s.t. } \ \breve{\mathbf{\Lambda}} \succeq \mathbf{0}, \ \breve{\mathbf{\Lambda}} \in \mathcal{X}, \ \text{and } \breve{\mathbf{H}}^\top \breve{\mathbf{\Lambda}} \breve{\mathbf{H}} \preceq \mathbf{\Sigma}^{-1} \tag{23}$$

where

$$\breve{\mathbf{H}} = \begin{bmatrix} \cdots, & \underbrace{\breve{\mathbf{H}}_{ij}^T}_{\kappa-\text{th block}}, & \cdots \end{bmatrix}^T, \quad \breve{\mathbf{\Lambda}} = \mathbf{Diag}\left(\cdots, \ \underbrace{\breve{\mathbf{\Lambda}}_{ij}}_{\kappa-\text{th block}}, \ \cdots \right)$$

are respectively the Jacobian and information matrices of the stacked measurements, and $\mathcal{X}$ is the set of block-diagonal matrices which ensures *uncorrelated* measurements. This solution is obtained by employing the interior point method [21]. To solve (22)-(23), the consistency constraint can be incorporated into the KLD minimization solver [6] by adding an associated log barrier function: $\phi(\breve{\mathbf{\Lambda}})_\rho = -\rho \text{logdet}(\mathbf{\Sigma}^{-1} - \breve{\mathbf{H}}^\top \mathbf{\Lambda} \breve{\mathbf{H}})$, where ρ is a penalty that is iteratively reduced to zero. Defining $\mathbf{\Psi} = \mathbf{\Sigma}^{-1} - \breve{\mathbf{H}}^\top \breve{\mathbf{\Lambda}} \breve{\mathbf{H}}$, this barrier's gradient and Hessian are given by [22]:

$$\mathbf{G} = \rho \breve{\mathbf{H}} (\mathbf{\Sigma}^{-1} - \breve{\mathbf{H}}^\top \breve{\mathbf{\Lambda}} \breve{\mathbf{H}})^{-1} \breve{\mathbf{H}}^\top \tag{24}$$

$$\mathbf{\Upsilon}_{kl} = \rho \breve{\mathbf{H}} \mathbf{\Psi}^{-1} \breve{\mathbf{H}}^\top \mathbf{J}_{kl} \breve{\mathbf{H}} \mathbf{\Psi}^{-1} \breve{\mathbf{H}}^\top \tag{25}$$

where $\mathbf{G}$ is the gradient of this constraint, $\mathbf{\Upsilon}_{kl}$ is the second partial derivative of the barrier function with respect to the (k,l) entry of $\breve{\mathbf{\Lambda}}$, and $\mathbf{J}_{kl}$ is the matrix that contains a one in the (k,l) position and zeroes everywhere else.

The solution to this problem will then contain no more information than the true distribution about the local, relative states, thereby preventing our method from being overconfident when performing sparsification. Note that, unlike [6] we do *not* project our information onto a lower-dimensional subspace, as the local, relative information matrix in our formulation is by construction full-rank. That is, the reparametrization automatically performs this projection. Once $\breve{\mathbf{\Lambda}}$ is found, the noise information matrix of each inferred measurement can be extracted from the solution's block diagonals.

C. Decouple Sparsification from Marginalization

In order to ensure the sparsity of the graph so as to reduce computational cost, while taking advantage of the better information provided by the dense factors induced by marginalization, we propose to *decouple* the edge sparsification from the node marginalization, i.e., postponing sparsification to a later time after marginalization. This decoupling allows for further control of the graph structure:

- Since the dense factors inferred during marginalization encapsulate all the information contained in the discarded measurements about the remaining nodes (see Section IV) and will be kept in the graph until the next sparsification takes place, we can reuse this information every time step before it gets sparsified.
- Since the intra-clique factors in the Markov blanket are excluded from the initial marginalization, we can relinearize these nonlinear measurements every time before sparsification, thus improving estimation accuracy.
- Staggering the marginalization and sparsification processes allows us to distribute their computation over time between steps. That is, we determine the dense factors during marginalization and then delay sparsification until computational power permits.

Specifically, after marginalization and when sparsification is needed, similar to the procedure in Section IV-B, we first perform the local relative optimization using *all* the factors in the Markov blanket (including intra-clique) to obtain the updated marginal distribution for sparsification, $\mathcal{N}({}^L\hat{\mathbf{x}}_{\oplus}|{}^L\mathbf{\Lambda}_{t\oplus}{}^{-1})$. Subsequently, we conduct edge sparsification as described in Section V-B to find new sparse factors which will replace the dense *and* intra-clique factors in the full graph during future optimization.

VI. Experimental Results

We evaluated our method, dubbed *Decoupled Marginalization and Sparsification* (DMS), on both 2D and 3D, real-world and simulated datasets, through comparisons to NFR that uses local state estimates as linearization points [6] and GLC [4]. The datasets considered are MIT Killian Court (2D real),[2] Manhattan3500 (2D synthetic), Sphere400 (3D synthetic) and Sphere2500 (3D synthetic).[3] Table I describes the specifications of these datasets. In the proposed DMS, we employed the iSAM solver [9] to perform batch optimization every time a new factor was added into the graph as well as after node removal. Note that in the DMS, dense factors were kept in the graph until the next node was marginalized, to prevent multiple dense factors from being stored, although a better removal strategy can be developed in the future. For GLC, the open-source implementation available in the iSAM v1.7 library was used. To have fair comparisons, we implemented NFR also within iSAM, rather than the g2o framework as provided by the authors. The performance metrics used was root mean squared errors (RMSE), instead of KLD as in [5], [6], since estimation accuracy often is the performance criterion of utmost importance for a given application.

TABLE I
Datasets and Specifications

Dataset	Type	# Node Removal	# Factors	Benchmark
MIT Killian	SE(2)	1294/1941	2191	Batch
Manhattan	SE(2)	1165/3500	5598	Truth
Sphere400	SE(3)	132/400	779	Batch
Sphere2500	SE(3)	832/2500	4949	Truth

TABLE II
Estimation accuracy comparison in dense formulation

Dataset/Method	Pos. RMSE (m)	Ori. RMSE (rad)
Killian/Dense-DMS	**0.15805**	**0.00155306**
Killian/Dense-GLC	0.518736	0.00904069
Manhattan/Dense-DMS	1.18873	0.0538883
Manhattan/Dense-GLC	**1.10766**	**0.0504965**
Sphere400/Dense-DMS	**0.135886**	**0.00211764**
Sphere400/Dense-GLC	0.243297	0.00878151
Sphere2500/Dense-DMS	**0.896463**	**0.0263427**
Sphere2500/Dense-GLC	1.04921	0.0326938

A. Dense Formulation without Sparsification

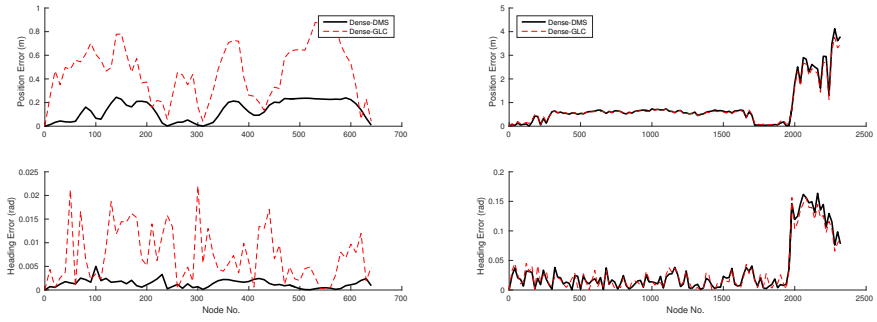
We first validated the proposed node removal via marginalization by considering the case without edge sparsification. In particular, we compared the proposed DMS to GLC, but not NFR, because, as reported in [6] the cliquey-dense formulation which is equivalent to GLC with global linearization points, tends to diverge if using local linearization points. As presented in Section IV, the proposed *Dense-DMS* calculates the dense factors with local linearization points and without including intra-clique factors. Table II shows the RMSEs over all the nodes, while Fig. 4 depicts the individual errors for each. Clearly, the proposed DMS generally outperforms its GLC counterpart, which is primarily attributed to that the high-accuracy dense factors induced by our marginalization. This motivates us to keep these factors in the graph as long as possible.

B. Sparse Formulation with Sparsification

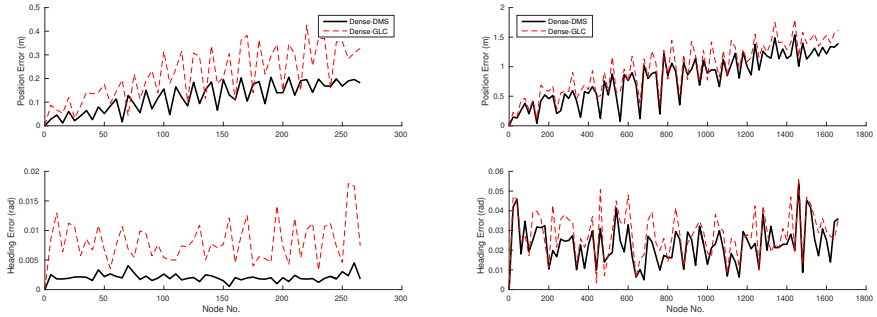
We next tested the complete algorithms (i.e., including edge sparsification). Note that as NFR [6] offers various sparse topologies, in this test we implemented the best subgraph topology, which is achieved by adding the next highest $(1-\gamma)(n-1)$ factors to the maximum spanning tree of size $\gamma(n-1)$, where n is the size of the Markov blanket

[2]This dataset can be found at: `http://kaspar.informatik.uni-freiburg.de/~slamEvaluation/datasets.php`

[3]These three datasets are available online at: `http://people.csail.mit.edu/kaess/isam`



(a) MIT Killian (b) Manhattan3500



(c) Sphere400 (d) Sphere2500

Fig. 4. Estimation errors for the dense formulations of DMS and GLC. It is clear that the proposed DMS tends to greatly outperform its GLC counterpart.

and $\gamma > 1$ is a proportionality factor. For the proposed DMS and its consistent version termed as *DMS-C*, we used the resource-aware sparsity parameter, $\alpha||\overline{\mathbf{\Sigma}_{ij}}||_{\infty} n^{\beta}$, with $\beta = 0.5$. This was chosen heuristically, and future work will further investigate the sparsity parameter choice. Note that since the accuracy of these methods heavily depends on the number of factors remaining in the graph, we tested DMS and NFR on various sparsity parameters, α and γ, and reported the runs which provided similar (but *not* identical) overall levels of sparsity. This implies that the comparisons presented here are *not* completely fair.

The RMSE results for the compared methods are shown in Table III and Fig. 5 plots the errors for each of the remaining nodes in the graph. Evidently, the proposed DMS approaches generally performed significantly better than GLC and attained slightly better accuracy (though by a small margin) than NFR. It is not surprising that GLC never provides the most accurate estimates, in part because it is being compared to the denser formulations, as well as it uses the global state estimates as linearization points in marginalization. These results validate the online sparse structure that is used in the proposed DMS, determined by ℓ_1-regularized KLD minimization, and can be effectively utilized to reduce graph density.

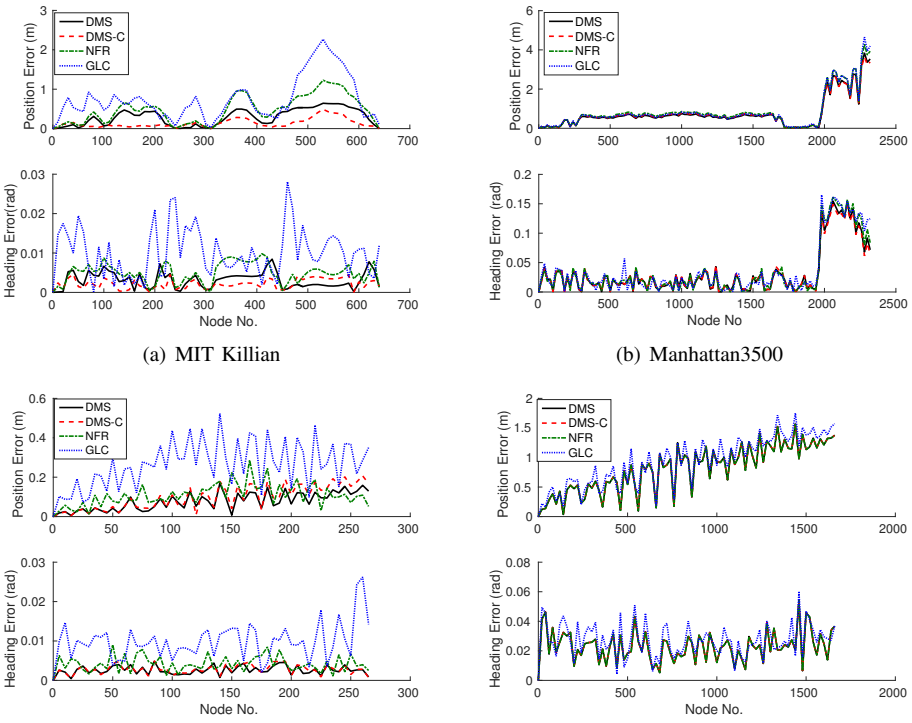
TABLE III

ESTIMATION ACCURACY COMPARISON IN THE SPARSE FORMULATION

Dataset/Method	Pos. RMSE (m)	Ori. RMSE (rad)	Factors
Killian/DMS	0.356361	0.00371119	1006
Killian/DMS-C	**0.175106**	**0.00261568**	1006
Killian/NFR	0.588046	0.00550779	997
Killian/GLC	0.917708	0.0117945	804
Manhattan/DMS	1.12313	0.0507679	4813
Manhattan/DMS-C	**1.11201**	**0.0492048**	4812
Manhattan/NFR	1.2478	0.0536323	4777
Manhattan/GLC	1.26928	0.0564418	4023
Sphere400/DMS	**0.089349**	**0.00315908**	780
Sphere400/DMS-C	0.114866	0.00322524	780
Sphere400/NFR	0.122146	0.00504577	774
Sphere400/GLC	0.288034	0.010465	649
Sphere2500/DMS	0.904645	0.0266831	5635
Sphere2500/DMS-C	0.904429	0.0266819	5635
Sphere2500/NFR	**0.901396**	**0.0266631**	5635
Sphere2500/GLC	1.01976	0.0319511	4112

VII. CONCLUSIONS

In this paper, we have introduced a decoupled, consistent marginalization and sparsification (DMS) approach for reducing the computational cost of graph-based SLAM to enable long-term operation. This decoupling allows for better use of accurate dense factors induced by marginalization and for spreading the computation of marginalization and sparsification between these two steps. In particular, we have shown that during node removal via marginalization, the proper choice of linearization points in constructing marginal, dense factors is to use the relative, local, instead of global, state estimates in the Markov blanket. Moreover, we have proposed to determine online a sparse topology through sparsity-regularized convex optimization. Based on this topology, consistent sparse factors are constructed to best approximate the original dense factors in the blanket. The proposed approach has been validated on both 2D and 3D public datasets and shown to outperform GLC [4], while providing competetive results to NFR [5], [6].



(a) MIT Killian

(b) Manhattan3500

(c) Sphere400

(d) Sphere2500

Fig. 5. Estimation errors for the sparse formulations of DMS, NFR, and GLC. Note that the proposed DMS uses the sparse topology induced online by the ℓ_1-regularized KLD minimization, and NFR employs the subgraph topology. Note also that in some of these plots, all the compared approaches perform very closely, which makes the corresponding lines difficult to distinguish.

REFERENCES

[1] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *Proc. of the IEEE International Conference on Robotics and Automation*, Orlando, FL, May 15–19, 2006, pp. 2262–2269.

[2] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, Dec. 2006.

[3] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[4] N. Carlevaris-Bianco, M. Kaess, and R. Eustice, "Generic node removal for factor-graph SLAM," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1371–1385, Dec. 2014.

[5] M. Mazuran, T. G. Diego, S. Luciano, and W. Burgard, "Nonlinear graph sparsification for SLAM," in *Proc. of Robotics: Science and Systems*, Berkeley, CA, Jul. 12–16, 2014.

[6] M. Mazuran, W. Burgard, and G. D. Tipaldi, "Nonlinear factor recovery for long-term SLAM," *International Journal of Robotics Research*, 2015. [Online]. Available: http://ijr.sagepub.com/content/early/2015/06/27/0278364915581629

[7] G. Huang, M. Kaess, and J. Leonard, "Consistent sparsification for graph optimization," in *Proc. of the European Conference on Mobile Robots*, Barcelona, Spain, Sep. 25–27, 2013, pp. 150–157.

[8] K. P. Murphy, *Machine Learning: A Probabilistic Pespective*. The MIT Press, 2012.

[9] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

[10] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *International Journal of Robotics Research*, vol. 31, pp. 217–236, Feb. 2012.

[11] G. Sibley, L. Matthies, and G. Sukhatme, "Sliding window filter with application to planetary landing," *Journal of Field Robotics*, vol. 27, no. 5, pp. 587–608, Sept./Oct. 2010.

[12] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "An observability constrained sliding window filter for SLAM," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, Sep. 25–30, 2011, pp. 65–72.

[13] K. Konolige, J. Bowman, J. D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua, "View-based maps," *International Journal of Robotics Research*, vol. 29, no. 8, pp. 941–957, Jul. 2010.

[14] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proc. of the IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nara, Japan, Nov. 13–16, 2007, pp. 225–234.

[15] V. Ila, J. Porta, and J. Andrade-Cetto, "Information-based compact pose SLAM," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 78–93, Feb. 2010.

[16] E. Eade, P. Fong, and M. Munich, "Monocular graph SLAM with complexity reduction," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, Oct. 18–22, 2010, pp. 3017–3024.

[17] E. Nerurkar, K. Wu, and S. Roumeliotis, "C-KLAM: Constrained keyframe-based localization and mapping," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, May 2014, pp. 3638–3643.

[18] J. Vial, H. Durrant-Whyte, and T. Bailey, "Conservative sparsification for efficient and consistent approximate estimation," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, Sep. 25–30, 2011, pp. 886 –893.

[19] G. Huang, K. Eckenhoff, and J. Leonard, "Optimal-state-constraint EKF for visual-inertial navigation," in *Proc. of the International Symposium on Robotics Research*, Sestri Levante, Italy, Sep. 12–15, 2015, (to appear).

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[21] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[22] K. Eckenhoff and G. Huang, "Decoupled, consistent node removal and edge sparsification for graph-based SLAM," University of Delaware, Tech. Rep. RPNG-2016-003, 2016, available: http://udel.edu/~ghuang/papers/tr_dms.pdf.