Optimizing Relative Transformations in Graph-Based SLAM

John Macdonald, Randal W. Beard

***Abstract*—Traditional back-end optimization in graph-based SLAM is focused on improving global pose estimates. The object of the optimization is to find the most likely configuration of global poses given fixed relative transformation constraints produced in the front-end. However, the relative transformations delivered by the front-end are themselves only estimates. In this paper we show how making relative transformations the focus of the back-end optimization leads to a novel algorithm that iteratively improves the joint estimate of global poses and relative transformations. We test the algorithm on front-end data from a 225 meter closed-loop trajectory. Our results show the algorithm provides the same global pose estimates in slightly less computational time when compared to g^{2}o, a state-of-the-art back-end optimization tool. But the optimized relative transformations produced by g^{2}o require several iterations before they become more accurate than their initial values. The algorithm presented in this paper offers advantages over traditional back-end optimization in at least two ways. It uses the covariance instead of the information form, allowing ready access to marginal covariance elements needed by the front-end. It also improves both global pose and relative transformation estimates at each iteration, making the end result of the optimization suitable to a wider variety of applications.**

I. Introduction

We seek to employ the graph-based representation of simultaneous localization and mapping (SLAM) as part of a flying system operating in unstructured and unknown indoor environments. Flying through such an environment leads to several immediate consequences. For example, an appropriate vehicle must be small and able to maneuver in cluttered spaces. A small and agile vehicle will need to produce a useful navigation solution quickly despite limits on payload and computational resources. Relaxing assumptions about structure in the environment motivates the use of information-rich sensors (e.g. stereo or RGB-D cameras), but these further increase demands on the constrained processing power.

The graphical representation of SLAM was first applied in [1]. As a part of graph-based SLAM, the so called "back-end" optimization [2] incorporates information gained from revisiting a location in the map to reduce the errors that have accumulated during exploration. The objective of that optimization is tied to the real-time navigation of the vehicle. For example, if the vehicle makes time critical decisions based on global pose estimates, the back-end should be able to quickly compute their mean and covariance. If, on the other hand, the vehicle navigates relative to saved images of the environment, the relative transformations between those images may be the most important output from the back-end.

The contribution of this paper is to present an alternative paradigm for addressing the back-end optimization problem. We seek to efficiently estimate the joint distribution over global poses *and* relative transformations by accounting for the independence and conditional independence properties inherent in the problem. This new approach gives essentially the same global pose estimates as the current state-of-the-art. It differs in that it operates in the covariance form, and it directly optimizes the relative transformations between saved images of the environment. Standard approaches can produce intermediate results that actually make estimates of the relative transformations worse. By estimating these variables directly we improve their estimates at each iteration of the algorithm.

In Section II we review related work representing current approaches to back-end optimization. In Section III we briefly review our front-end estimation and navigation techniques. Our front-end motivates our approach to back-end optimization that we present in Section IV. Results for a simple scenario presented in Section V highlight some of the strengths of this new paradigm and suggest likely paths for further development. We discuss those conclusions and plans for future work in Section VI.

II. Related Work

In 1997 Lu and Milios [1] proposed optimizing a graph of global poses as a robust solution to the SLAM problem. Their objective was, "to maintain all the local frames of [exteroceptive] data as well as the relative spatial relationships between local frames. ... Consistency is achieved by using all the spatial relations as constraints to solve for the data frame poses simultaneously." In other words, a "front-end" [2] system would determine the relative transformations between key poses where saved data was acquired in the globally referenced environment. By "relative" they meant that the transformations between global poses were defined in the reference frame of the pose from which they originated. Once relative transformations were estimated in the front-end, they proposed that those estimates be frozen and used in a back-end process to find the most likely arrangement of the global poses. The optimized global poses then aligned their associated data into a consistent map.

To further emphasize the nature of their approach, they stated that, "We treat relations [i.e. relative transformations] as primitives, but treat locations [i.e. global poses] as free variables. ... [W]e do not directly update the existing relations in the network when new observations are made. We simply add new relations to the network. All the relations are used as constraints to solve for the location variables which, in turn, define a set of updated and consistent relations. ... We do not deal with [i.e. optimize] the relations directly."

We note here that, like Lu and Milios, we tend to limit our discussion to graphs containing only relative transformations between global poses. Some of the papers cited below also optimize the position of global landmarks by accounting for the relative transformations between those landmarks and the poses from which they were observed. We will mention landmarks in our discussion where important, but we explain further in Section III that pose-to-landmark constraints can be subsumed by the front-end into pose-to-pose constraints using methods such as [3] and [4].

Several additional algorithms building on [1] have since been presented [5]–[19]. Optimization is achieved by minimizing the error measured by the squared Mahalanobis distance. Let $\boldsymbol{\rho}$ represent a vector of all of the global poses to be optimized, with individual poses designated by $\boldsymbol{\rho}_i$. We define $\boldsymbol{\tau}_i^j$ to be the relative transformation from $\boldsymbol{\rho}_i$ to $\boldsymbol{\rho}_j$. Let the probability distribution $\mathrm{p}(\boldsymbol{\tau}_i^j)$ represent the Gaussian distributed estimate of $\boldsymbol{\tau}_i^j$ with covariance matrix $\boldsymbol{\Sigma}_{i,j}$. Finally, let $h(\boldsymbol{\rho}_i, \boldsymbol{\rho}_j)$ denote a function that takes $\boldsymbol{\rho}_i$ and $\boldsymbol{\rho}_j$ as inputs and returns the relative transformation between them. Using this notation, the error metric to be minimized is written as

$$\epsilon(\boldsymbol{\rho}) = \sum \left(h\left(\boldsymbol{\rho}_i, \boldsymbol{\rho}_j\right) - \boldsymbol{\tau}_i^j\right)^\top \boldsymbol{\Sigma}_{i,j}^{-1} \left(h\left(\boldsymbol{\rho}_i, \boldsymbol{\rho}_j\right) - \boldsymbol{\tau}_i^j\right), \quad (1)$$

where $\epsilon(\boldsymbol{\rho})$ indicates that the scalar error, ϵ, is a function of all the global poses, $\boldsymbol{\rho}$.

Some early efforts at back-end optimization applied relaxation techniques to the problem. In one example [5] the authors draw an analogy between graph-based SLAM and a mechanical spring-mass system. They assert that global poses represent masses and that the fixed constraints, $\boldsymbol{\tau}_i^j$, from the front-end correspond to springs. Each constraint's uncertainty acts like a spring constant. They propose a type of gradient descent to sequentially adjust each pose based on the "forces" it experiences from adjacent poses.

The approach of [6], dubbed multi-level relaxation (MLR), follows in a similar vein. They primarily differ from [5] in that they propose the network be relaxed at different levels of resolution in order to make computation more efficient. As late as 2006 some authors [8] cited MLR as the current "state-of-the-art" in back-end optimization. See the citations in [6] for additional references to relaxation-based approaches.

More recent back-end optimization algorithms tend to take a different approach due to advances in direct methods for solving sparse linear systems [2]. Equation (1) can be approximated by replacing the nonlinear function $h\left(\boldsymbol{\rho}_i, \boldsymbol{\rho}_j\right)$ with its first order Taylor expansion. Taking the derivative, and setting the result equal to zero (see [2] for details) yields the normal equations

$$\mathbf{H}\boldsymbol{\rho}_{\boldsymbol{\Delta}} = \mathbf{d}, \quad (2)$$

where $\mathbf{H}$ is the information matrix associated with the probability distribution $\mathrm{p}(\boldsymbol{\rho})$, $\boldsymbol{\rho}_{\boldsymbol{\Delta}}$ is an incremental change in $\boldsymbol{\rho}$, and $\mathbf{d}$ is a constant vector.

The authors of [7], [8] use the observation that $\mathbf{H}$ is sparse for graph-based SLAM. Nonzero entries in the information matrix only occur along the block diagonal and in off-diagonal blocks corresponding to poses connected by a measured relative transformation. The authors then employ a variable elimination technique to reduce their initial graph to one containing fewer variables. Specifically, they remove all of the landmarks, incorporating the information from pose-to-landmark constraints into appropriate pose-to-pose constraints. The authors then solve the reduced system by matrix inversion or conjugate gradient descent.

The authors of [9]–[11] take the variable elimination technique further. They present a line of research that seeks to speed up the incremental optimization of $\boldsymbol{\rho}$ by making smart decisions about the order and method for eliminating variables from the graph. Instead of computing the information matrix, they focus attention on the Jacobian of $h\left(\boldsymbol{\rho}_i, \boldsymbol{\rho}_j\right)$ (and the similar function for landmarks) to improve accuracy and numerical stability [9]. Using QR-factorization they eliminate all of the variables in the graph to form the right triangular matrix $\mathbf{R}$, then solve the resulting structured, sparse linear system. The key to keeping this process efficient is keeping $\mathbf{R}$ as sparse as possible by choosing a good order in which variables are eliminated.

The authors of [9]–[11] also draw interesting connections between the matrices arising from linearization of (1) and probabilistic graphical models (Ch. 8 of [20]). Probabilistic graphical models include Bayesian networks, Markov random fields, and factor graphs. Their emphasis on graphical models culminates in [11] where the authors directly manipulate a tree similar to a Bayesian network [21] instead of the corresponding right triangular matrix $\mathbf{R}$. We also seek to draw insight from probabilistic graphical models and will discuss this further in Section IV and Section VI.

The work presented in [12]–[17] represents another thread in back-end optimization research. As usual, the methods presented in [12]–[17] iteratively linearize Equation (1) and adjust $\boldsymbol{\rho}$. In [12] the authors introduce a variant of stochastic gradient descent to make the iterative optimization robust to local minima. They also present interesting results that suggest Equation (1) as an error metric is "not an adequate measure of graph quality." We will return to this observation in Section IV.

In [13] and [14] the authors extend the work in [12], making it converge faster to a more accurate solution by organizing the global poses into a tree structure and distributing rotational error more effectively. The primary contribution of [15] is similar to [9]–[11] in that they find an efficient solution to the linear system used in iterative optimization by choosing an appropriate ordering for variable elimination. In a contemporary paper [16] the authors modify the typical iterative optimization by calculating state changes on a manifold. They also make use of a hierarchy of graph-based maps with varying degrees of resolution.

Finally, in "A General Framework for Graph Optimization" (dubbed g^{2}o) [17] the authors take many of the above innovations and package them in an efficient C++ implementation. Regarding their results they state, "We present evaluations carried out on a large set of real-world and simulated datasets; in all experiments g^{2}o offered a performance comparable with the state-of-the-art approaches and in several cases even outperformed them."

Of the foregoing papers, all but [14] treat the relative transformations as fixed constraints. In other words, after the front-end passes $\boldsymbol{\tau}_i^j$ to the back-end, that vector and its uncertainty are never altered. Recall that Lu and Milios said they "do not directly update the existing relations in the network when new observations are made. We simply add new relations to the network." This leads to a network that grows over time even if the robot remains within already explored territory. They authors of [14] allow the possibility of revisiting relative transformations, fusing new estimates with old ones. Their update of the transformations is equivalent to a Kalman update step where the measurement function is simply the identity function.

The extended Kalman filter (EKF) SLAM community has also produced a body of work relevant to our research. Several authors (e.g. [18], [19]; see also the references therein) divide the standard EKF SLAM map into a number of statistically independent local maps, or "sub-maps." Several factors motivate this approach to EKF SLAM. Consistency (as defined in Ch. 5 of [22]) is improved [23] since smaller uncertainty in the robot pose relative to the local map leads to smaller linearization error. It is also well known that traditional EKF SLAM becomes computationally intractable as the number of mapped features grows. Sub-maps help to alleviate this problem by bounding the size of the filter state within each sub-map. Such approaches then retain an estimate of the relative transformations between the sub-maps. The global map can then be obtained as above by using an iterative nonlinear optimization routine. In this sense, most work in EKF SLAM using sub-maps is similar to graph-based SLAM.

The work in [23] (and subsequent publications [24]–[26]) presents an exception to this analogy between EKF SLAM sub-mapping and graph-based SLAM. We discuss [23]–[26] in additional detail in Section V. We note here that [23] introduces a distinction from the rest of the sub-mapping literature. They do not treat the individual sub-maps as statistically independent. Rather, when sub-maps are to be optimized with information gained at loop closure, that information is propagated back through the network of sub-maps using the property of *conditional* independence.

The algorithms in [5]–[19] all have at their core a paradigm similar to [1]. The relative transformations $\boldsymbol{\tau}_i^j$ from global pose $\boldsymbol{\rho}_i$ to global pose $\boldsymbol{\rho}_j$ are estimated by a front-end system. Once passed to the back-end, the $\boldsymbol{\tau}_i^j$ are almost always treated as fixed constraints. The global poses are the focus of iterative optimization; the objective is to find the arrangement of global poses that best fits the fixed transformation estimates. Innovations over [1] have made this approach more efficient and more accurate, but the gist of the paradigm remains essentially the same.

We differ from the foregoing work on graph-based SLAM in that we seek to directly optimize the relative transformations in the back-end. Focusing on relative transformations leads to the same final solution as the preceding algorithms. However, for a simple example it produces noticeably more accurate intermediate results. Our focus on the relative transformations grows out of our front-end navigation concept. We will briefly highlight some relevant aspects of that front-end navigation approach before presenting our back-end concept in greater detail.

III. Front-End Philosophy and Design

Autonomous flight through a priori unknown, indoor environments is a growing topic of research, though to our knowledge only a few groups [27]–[32] have produced significant working prototypes. Multi-rotor helicopters are the platform of choice for this research. These vehicles have been identified as potential "game changers" [33] in robotics. We also adopt mutli-rotor helicopters in our work.

The advanced systems presented in [27]–[30] use small scanning laser range finders to sense the environment. Laser range finders produce accurate and frequent measurements that are easy to process, but they are inherently 2D sensors. To map a 3D environment, laser range finders require significant structure (vertical walls, piecewise constant floors, etc.). Stereo vision [31] and recently popular RGB-D cameras [32] are better suited to unstructured settings. However, data from these sensors require more computation, as evidenced by the system design in [31] or the use of an offboard computer in [32]. We use an RGB-D camera with the goal of performing all computation using the vehicle's limited resources.

To accommodate the vehicle's fast dynamics the authors of [27] advocate moving as many processes as possible out of the time-critical estimation and control. We also subscribe to this philosophy. Many authors represent the vehicle's current pose in globally metric coordinates. However, [27] states that complex 3D environments, "will likely require relaxing the need to maintain a purely metric representation of the environment and the state of the vehicle." We implement front-end navigation by making all estimates and objectives relative to saved images of the environment.

Relative navigation is especially relevant for indoor flight. During initial exploration the vehicle will move into unknown territory along traversable paths it can sense. Similarly, any interaction with the environment will be accomplished based on current sensor data. Plans to revisit an area should generally follow paths already explored due to the high likelihood of obstruction through unexplored areas. These are all relative navigation concepts.

However, globally metric information is still important for sophisticated autonomous behavior. For example, the authors of [11] point out that an accurate global map allows the vehicle to hypothesize whether globally adjacent locations might have a previously undetected pathway between them. Globally metric information should still be available to the vehicle, but the vehicle's immediate state estimation needs can be satisfied in the front-end using metric information referenced to the local surroundings

We implement relative navigation [34] using techniques similar to those described in [3], [4]. Our knowledge of the environment is embedded in saved RGB-D images. The vehicle observes its current relative pose by comparing a saved reference image with new images as they are captured. The current relative pose estimate can also incorporate IMU and altimeter measurements. As the previously saved image becomes inadequate (e.g. insufficient overlap with current images), the vehicle saves a new image and its relative transformation from the previous image. Navigation then continues relative to the new image. When the vehicle detects images saved at times in the more distant past, it also estimates and saves the relative transformation between that image and the most recently saved image (i.e. a loop closure estimate). The resulting chain of saved images and relative transformations define a map of the environment that is globally topological and locally metric.

In summary, we expect our small multi-rotor helicopter to navigate through unstructured settings using saved RGB-D images with the goal of full autonomy despite limited computing power. Our focus in the front-end is to provide time-critical estimates of the vehicle's metric pose relative to the most recently recognized saved image. However, we also seek to develop a globally metric map in the back-end optimization.

IV. Paradigm Shift

In this section we present our primary contribution, an alternative approach to back-end optimization. We first discuss some relevant background material related to Bayesian networks and manipulating Gaussian distributions. With that context we then present the details of our approach. We conclude this section by discussing similarities and differences between our work and that of [23]–[26].

A. Bayesian Network Concepts

A excellent introduction to Bayesian networks can be found in [20]. We have also found [35] to be a useful source for additional insights. We only mention a few concepts in this section and refer the reader to these and other sources for a more thorough description.

A Bayesian network (also called a Bayes or belief network) is a directed acyclic graph that represents the conditional dependencies among a collection of random variables. The random variables make up the nodes in the graph depicted by a labeled circle. A probabilistic dependence is indicated by a directed edge such that the node at the arrow's head is dependent on the node at the arrow's tail. To use the jargon, the variable at the head is referred to as the child node. The variable at the tail is called the parent node. The Asia Network in Figure 1 offers a canonical example and illustrates the concepts we introduce here.

The authors of [35] recommend that Bayesian networks be constructed so that the direction of an edge represent a causal relationship. While this is not technically necessary, they argue that doing so makes inference more intuitive. Causal relationships are manifest in Figure 1. Smoking, for example, has a causal influence on getting cancer or bronchitis.

It is also useful to mention here the concept of a Markov blanket. Formally defined, the Markov blanket for a given variable, x, is the set of its parents, children, and co-parents. More intuitively, we can think of a Markov blanket as the minimal set of nodes that isolates our belief about x from the rest of the network [20]. The distribution of x is conditionally independent of all other variables *given* the variables in its Markov blanket. In Figure 1, the Markov blanket for node 'C' consists of nodes 'S' (parent), 'or' (child), and 'T' (co-parent).

Finally, we introduce here the concept of a mediating variable. The authors of [35] describe mediating variables as,

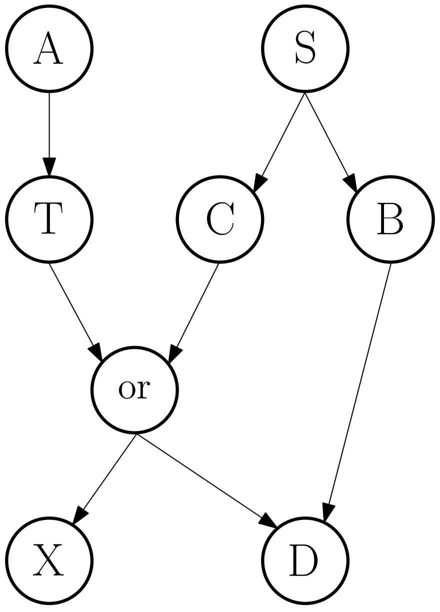

Fig. 1. The so-called Asia Network, a canonical example of a Bayesian network. The network models a physician's belief about the variables: A - the patient has recently been to Asia; S - the patient is a smoker; T - the patient has tuberculosis; C - the patient has cancer; B - the patient has bronchitis; X - the results of a patient's X-ray are abnormal; D - the patient exhibits dyspnoea (i.e. shortness of breath). The variable labeled 'or' is a mediating variable that captures the belief that the patient has either tuberculosis or cancer.

"variables for which posterior probabilities are not of immediate interest, but which play important roles for achieving correct conditional independence and dependence properties and/or efficient inference." In Figure 1 the 'or' variable is a mediating variable.

B. Updating Joint Gaussian Distributions

The ultimate description of a collection of random variables is the joint distribution over those variables. Conditional distributions are useful when decomposing the joint distribution, and marginal distributions are relevant, for example, when we receive outside information (i.e. a measurement) about a subset of the random variables. For use in subsequent sections, we will briefly present some derivations relating these types of distributions for jointly Gaussian random variables. For more detail, see [36].

Let a D-dimensional random vector $\mathbf{x}$ of jointly Gaussian variables be partitioned into two, disjoint sub-vectors such that $\mathbf{x} = \left[\mathbf{x}_1^\top, \mathbf{x}_2^\top\right]^\top$, where $\mathbf{x}_1$ is dimension D_1 and $\mathbf{x}_2$ is dimension D_2. Then the joint distribution $\mathrm{p}(\mathbf{x})$, with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, is partitioned such that

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix},$$
$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}.$$

Typical textbook derivations (e.g. [20], Chapter 2.3.1) define the conditional distribution $\mathrm{p}\left(\mathbf{x}_1 | \mathbf{x}_2\right)$, with mean $\boldsymbol{\mu}_{1|2}$ and covariance $\mathbf{\Sigma}_{1|2}$, such that

$$\boldsymbol{\mu}_{1|2} \triangleq \boldsymbol{\mu}_1 + \mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1}\left(\mathbf{x}_2 - \boldsymbol{\mu}_2\right), \tag{3}$$

$$\mathbf{\Sigma}_{1|2} \triangleq \mathbf{\Sigma}_{11} - \mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1}\mathbf{\Sigma}_{21}. \tag{4}$$

To simplify notation in the sequel we define

$$\mathbf{K} \triangleq \mathbf{\Sigma}_{12}\mathbf{\Sigma}_{22}^{-1}. \tag{5}$$

There are a few undesirable aspects of expressing the conditional distribution $\mathrm{p}\left(\mathbf{x}_1|\mathbf{x}_2\right)$ using (3) and (4). First, the mean vector $\boldsymbol{\mu}_{1|2}$ has dimension D_1. The marginal distribution $\mathrm{p}\left(\mathbf{x}_2\right)$ has a mean $\boldsymbol{\mu}_2$ of dimension D_2. To recover the joint distribution $\mathrm{p}\left(\mathbf{x}\right) = \mathrm{p}\left(\mathbf{x}_1|\mathbf{x}_2\right)\mathrm{p}\left(\mathbf{x}_2\right)$ would require that we sum exponents with different dimensions. It is also unattractive to leave the conditional distribution's functional dependence on $\mathbf{x}_2$ buried in the conditional mean.

We can rewrite the conditional distribution $\mathrm{p}\left(\mathbf{x}_1|\mathbf{x}_2\right)$ (see [36]) such that

$$\log\left(\mathrm{p}\left(\mathbf{x}_1|\mathbf{x}_2\right)\right) \propto \left(\mathbf{x} - \boldsymbol{\mu}\right)^\top \mathbf{A}\left(\mathbf{x} - \boldsymbol{\mu}\right), \tag{6}$$

where

$$\mathbf{A} \triangleq \begin{bmatrix} \mathbf{\Sigma}_{1|2}^{-1} & -\mathbf{\Sigma}_{1|2}^{-1}\mathbf{K} \\ -\mathbf{K}^\top\mathbf{\Sigma}_{1|2}^{-1} & \mathbf{K}^\top\mathbf{\Sigma}_{1|2}^{-1}\mathbf{K} \end{bmatrix}. \tag{7}$$

Note that the dimension and mean of the conditional exponent now correspond to the original joint distribution. We can similarly rewrite $\mathrm{p}\left(\mathbf{x}_2\right)$ such that

$$\log\left(\mathrm{p}\left(\mathbf{x}_2\right)\right) \propto \left(\mathbf{x} - \mathbf{b}\right)^\top \mathbf{B}\left(\mathbf{x} - \mathbf{b}\right), \tag{8}$$

where the D-dimensional vector $\mathbf{b}$ is defined as

$$\mathbf{b} \triangleq \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\mu}_2 \end{bmatrix},$$

and

$$\mathbf{B} \triangleq \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{22}^{-1} \end{bmatrix}.$$

In [36] we offer some additional observations about the relationship between the information matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{\Sigma}^{-1}$.

As mentioned above, the marginal distribution is important when incorporating new information from a measurement that is a function of a subset of variables from the full joint distribution. Let the measurement be a function of $\mathbf{x}_2$, and let $\mathrm{p}(\check{\mathbf{x}}_2)$ represent our belief about the updated states, with updated mean $\check{\boldsymbol{\mu}}_2$ and covariance $\check{\mathbf{\Sigma}}_{22}^{-1}$. We also define

$$\check{\mathbf{b}} \triangleq \begin{bmatrix} \mathbf{0} \\ \check{\boldsymbol{\mu}}_2 \end{bmatrix}$$

$$\check{\mathbf{B}} \triangleq \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \check{\mathbf{\Sigma}}_{22}^{-1} \end{bmatrix}$$

To propagate the new information contained in $\check{\mathbf{x}}_2$ into the remaining elements of $\mathbf{x}$ we recover the joint distribution from the conditional and the updated marginal distributions: $\mathrm{p}(\mathbf{x}) = \mathrm{p}(\mathbf{x}_1|\mathbf{x}_2)\mathrm{p}(\check{\mathbf{x}}_2)$. This product gives

$$\begin{aligned}\log\left(\mathrm{p}\left(\mathbf{x}\right)\right) &\propto \left(\mathbf{x} - \boldsymbol{\mu}\right)^\top\mathbf{A}(\mathbf{x} - \boldsymbol{\mu}) + (\mathbf{x} - \check{\mathbf{b}})^\top\check{\mathbf{B}}(\mathbf{x} - \check{\mathbf{b}}) \\ &\propto (\mathbf{x} - \overset{\star}{\boldsymbol{\mu}})^\top\overset{\star}{\mathbf{\Sigma}}^{-1}(\mathbf{x} - \overset{\star}{\boldsymbol{\mu}}),\end{aligned}$$

where the optimized joint covariance and mean are

$$\begin{aligned}\overset{\star}{\mathbf{\Sigma}} &\triangleq \left(\mathbf{A} + \check{\mathbf{B}}\right)^{-1} \\ &= \begin{bmatrix} \mathbf{\Sigma}_{1|2}^{-1} & -\mathbf{\Sigma}_{1|2}^{-1}\mathbf{K} \\ -\mathbf{K}^\top\mathbf{\Sigma}_{1|2}^{-1} & \check{\mathbf{\Sigma}}_{22}^{-1} + \mathbf{K}^\top\mathbf{\Sigma}_{1|2}^{-1}\mathbf{K} \end{bmatrix}^{-1}\end{aligned} \tag{9}$$

$$\overset{\star}{\boldsymbol{\mu}} \triangleq \overset{\star}{\mathbf{\Sigma}}\left(\mathbf{A}\boldsymbol{\mu} + \check{\mathbf{B}}\check{\mathbf{b}}\right) \tag{10}$$

The optimized covariance $\overset{\star}{\mathbf{\Sigma}}$ of the joint distribution can be recovered without inverting the information matrix as suggested by (9). Instead, following the derivation in [36], we find the optimized covariance and mean to be

$$\begin{aligned}\overset{\star}{\mathbf{\Sigma}} &= \begin{bmatrix} \overset{\star}{\mathbf{\Sigma}}_{11} & \overset{\star}{\mathbf{\Sigma}}_{12} \\ \overset{\star}{\mathbf{\Sigma}}_{21} & \overset{\star}{\mathbf{\Sigma}}_{22} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{\Sigma}_{11} - \mathbf{K}\left(\mathbf{\Sigma}_{22} - \check{\mathbf{\Sigma}}_{22}\right)\mathbf{K}^\top & \mathbf{K}\check{\mathbf{\Sigma}}_{22} \\ \check{\mathbf{\Sigma}}_{22}\mathbf{K}^\top & \check{\mathbf{\Sigma}}_{22} \end{bmatrix},\end{aligned} \tag{11}$$

$$\overset{\star}{\boldsymbol{\mu}} = \begin{bmatrix} \boldsymbol{\mu}_1 - \mathbf{K}\left(\boldsymbol{\mu}_2 - \check{\boldsymbol{\mu}}_2\right) \\ \check{\boldsymbol{\mu}}_2 \end{bmatrix}. \tag{12}$$

C. *Optimizing Relative Transformations*

Most back-end optimization routines seek to optimize the global poses by minimizing Equation (1). This function is parameterized by saved relative transformations considered to be "measured" in the front end and fixed in the back-end. Equation (1) compares the saved transformations to the relative transformations derived from the optimized global poses. The global poses are the focus of the optimization.

However, those saved relative transformations are themselves only estimates. As described in Section III, the saved estimates of the relative transformations are an amalgamation of several IMU, altimeter, and visual odometry measurements. There is no guarantee that one will arrive at a globally accurate map by making global pose estimates agree with error prone relative transformation estimates. This may be the reason why the authors of [12] argue that Equation (1) is "not an adequate measure of graph quality."

We also note that the set of all relative transformations between global poses fully characterizes the globally metric map just as much as the set of all the global poses. By this we mean that if all the relative transformations were known without error then the map would be completely defined. Since the relative transformations are the only thing we can actually observe, why not seek to further refine the transformation estimates in the back-end. Using a relative navigation scheme in the front-end makes this subtle shift in emphasis seem especially relevant. A focus on optimizing relative transformations is the underlying philosophy driving our approach to back-end optimization.

1) Direct Approach: For the remainder of this paper we will consider the simple case of a vehicle traveling around a long, rectangular hallway before returning to the origin to detect a single loop closure. This scenario allows us to examine some of the fundamental aspects of our approach. In Section VI we will discuss extending the algorithm to more complex map topologies.

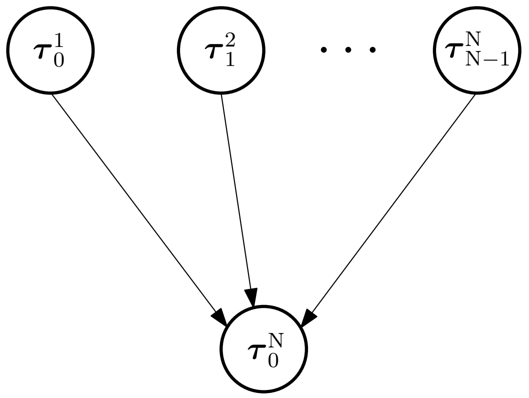

Fig. 2. A Bayesian network for a direct approach to optimizing the relative transformations. The nodes at the top represent odometry-like transformations between temporally consecutive saved images. The node at the bottom represents the single loop closure considered in this discussion. When evidence about τ_0^N becomes available through a loop closure measurement, all of the odometry-like measurements become correlated by the "explaining away" [20] phenomenon.

For this simple scenario, a direct approach to optimizing relative transformations might be modeled by the Bayesian network in Figure 2. The relative transformations represented in the top row exist between N temporally consecutive poses corresponding to images saved during exploration. Transformations between temporally consecutive poses are of the form τ_i^{i+1}. The transformation at the bottom of Figure 2 corresponds to the loop closure transformation from the origin to the pose of the last image in the trajectory.

When a new image of the environment is first saved, the vehicle begins navigation with respect to that image. The front end can estimate its change in position and heading with respect to that image *independent* of any other estimates of global poses or relative transformations. This is because the vehicle was certainly at the spot where the image was captured, no matter where that image is in the world or how the vehicle got there. This independence is reflected by the fact that no arrows enter the variables in the top row of Figure 2.

As the vehicle continues to explore, our prior belief about the relative transformations τ_i^{i+1} provides the only source of information about the relative transformation τ_0^N. This is reflected in the Bayesian network by the arrows pointing from all of the τ_i^{i+1} transformations into τ_0^N. However, at loop closure the vehicle measures its pose relative to the origin. This "evidence," to use the Bayesian network parlance, changes our belief about τ_0^N. For a Bayesian network like that of Figure 2, evidence on τ_0^N makes all of the τ_i^{i+1} transformations correlated [20].

We can also think about this scenario in terms of a covariance matrix for the joint distribution $p(\tau)$, where

$$\tau \triangleq \begin{bmatrix} \tau_0^1 \\ \tau_1^2 \\ \vdots \\ \tau_{N-1}^N \\ \tau_0^N \end{bmatrix}.$$

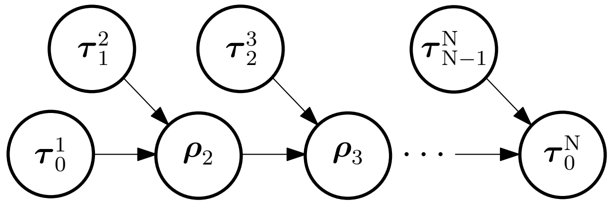

Fig. 3. A Bayesian network based on the idea of using global poses as mediating variables, thus leading to the joint distribution $p(\tau, \rho)$. Directed edges in the network are drawn to represent the physical reality that a relative transformation moves the vehicle from one global pose to another.

Before measuring a loop closure the covariance matrix has an upper left submatrix that is block diagonal because of the independence of the relative transformations τ_i^{i+1}. The last block row and column are dense because our prior belief about τ_0^N is a function of all the transformations preceding it. When a measurement of τ_0^N is applied to $p(\tau)$ via a Kalman update step, the cross-covariance elements in the last block row and column cause the remainder of the matrix to become dense also.

The correlations induced by the first loop closure measurement make it intractable to directly estimate the joint distribution over all relative transformations. To apply any future loop closure would require manipulating a large, dense covariance matrix. In a sense, this is analogous to naive EKF SLAM; using a single covariance matrix to keep track of correlations between all state elements is too computationally expensive.

2) Decomposing the Joint Distribution: Consider the relationship between a particular relative transformation τ_i^j and the rest of the relative transformations if the global poses ρ_i and ρ_j are *given*. Knowing these global poses ensures τ_i^j is always independent from the remaining relative transformations because τ_i^j is only defined by ρ_i and ρ_j. This conditional independence property is part of the justification [3] for the cost function given by Equation (1). The global poses ρ_i and ρ_j constitute a Markov blanket for τ_i^j.

This motivates us to reconsider our approach to back-end optimization by using the joint distribution $p(\tau, \rho)$. We construct the Bayesian network in Figure 3 following the guideline of assigning directed edges based on causal relationships. Relative transformations point into a global pose because they represent our belief about how the vehicle arrived there from the previous pose. We have made ρ_0 the arbitrary and certain global origin, therefore $\rho_1 = \tau_0^1$.

The remaining global poses can be considered mediating variables that help explain the "correct conditional independence ... properties" [35] between the relative transformations. In our simple scenario, if ρ_2 is given then our belief about τ_0^1 and τ_1^2 cannot be affected by any other relative transformation in the network. It is also evident from this Bayesian network that the Markov blanket of τ_2^3, for example, consists of ρ_2 (co-parent) and ρ_3 (child).

The Bayesian network in Figure 3 suggests the following decomposition of the joint distribution $p(\tau, \rho)$. To keep expressions compact we introduce the notation $\mathbf{x}_r$ to indicate the remaining variables in a random vector $\mathbf{x}$ that have not already been broken out in the preceding conditional distributions. Then the joint distribution can be written as

$$\begin{aligned}
\mathrm{p}\left(\boldsymbol{\tau},\boldsymbol{\rho}\right) &= \mathrm{p}\left(\boldsymbol{\tau}_0^1,\boldsymbol{\tau}_1^2|\boldsymbol{\tau}_\mathrm{r},\boldsymbol{\rho}_\mathrm{r}\right)\mathrm{p}\left(\boldsymbol{\tau}_\mathrm{r},\boldsymbol{\rho}_\mathrm{r}\right)\\
&= \mathrm{p}\left(\boldsymbol{\tau}_0^1,\boldsymbol{\tau}_1^2|\boldsymbol{\rho}_2\right)\mathrm{p}\left(\boldsymbol{\tau}_\mathrm{r},\boldsymbol{\rho}_\mathrm{r}\right)\\
&= \mathrm{p}\left(\boldsymbol{\tau}_0^1,\boldsymbol{\tau}_1^2|\boldsymbol{\rho}_2\right)\mathrm{p}\left(\boldsymbol{\tau}_2^3,\boldsymbol{\rho}_2|\boldsymbol{\tau}_\mathrm{r},\boldsymbol{\rho}_\mathrm{r}\right)\mathrm{p}\left(\boldsymbol{\tau}_\mathrm{r},\boldsymbol{\rho}_\mathrm{r}\right)\\
&= \mathrm{p}\left(\boldsymbol{\tau}_0^1,\boldsymbol{\tau}_1^2|\boldsymbol{\rho}_2\right)\mathrm{p}\left(\boldsymbol{\tau}_2^3,\boldsymbol{\rho}_2|\boldsymbol{\rho}_3\right)\mathrm{p}\left(\boldsymbol{\tau}_\mathrm{r},\boldsymbol{\rho}_\mathrm{r}\right)\\
&\;\;\vdots\\
&= \mathrm{p}\left(\boldsymbol{\tau}_0^1,\boldsymbol{\tau}_1^2|\boldsymbol{\rho}_2\right)\mathrm{p}\left(\boldsymbol{\tau}_2^3,\boldsymbol{\rho}_2|\boldsymbol{\rho}_3\right)\cdots\\
&\qquad\cdots\mathrm{p}\left(\boldsymbol{\tau}_{\mathrm{N}-1}^\mathrm{N},\boldsymbol{\rho}_{\mathrm{N}-1}|\boldsymbol{\tau}_0^\mathrm{N}\right)\mathrm{p}\left(\boldsymbol{\tau}_0^\mathrm{N}\right). \quad (13)
\end{aligned}$$

After completely decomposing $\mathrm{p}\left(\boldsymbol{\tau},\boldsymbol{\rho}\right)$ we have the marginal distribution $\mathrm{p}\left(\boldsymbol{\tau}_0^\mathrm{N}\right)$ at the end of (13). The loop closure measurement will be applied to this marginal distribution so that $\mathrm{p}\left(\check{\boldsymbol{\tau}}_0^\mathrm{N}\right)$ represents our updated belief about the relative transformation between the origin and the pose of the N^{th} saved image.

Using Equations (11) and (12) we now find the optimized distribution $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_{\mathrm{N}-1},\overset{\star}{\boldsymbol{\tau}}{}_{\mathrm{N}-1}^\mathrm{N},\overset{\star}{\boldsymbol{\tau}}{}_0^\mathrm{N}\right)$. We note that this is an efficient operation because the matrices involved are small. Once we have $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_{\mathrm{N}-1},\overset{\star}{\boldsymbol{\tau}}{}_{\mathrm{N}-1}^\mathrm{N},\overset{\star}{\boldsymbol{\tau}}{}_0^\mathrm{N}\right)$ we can trivially extract the marginal distribution $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_{\mathrm{N}-1}\right)$ and find $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_{\mathrm{N}-2},\overset{\star}{\boldsymbol{\tau}}{}_{\mathrm{N}-2}^{\mathrm{N}-1},\overset{\star}{\boldsymbol{\rho}}_{\mathrm{N}-1}\right)$ in the same manner. This pattern repeats back through the network until all variables have been updated with the loop closure information.

Algorithm 1 summarizes the back-end optimization process. Relative transformation estimates are produced in the front end. Those relative transformations can be composed in the back-end to find our prior belief about the joint distributions between relative transformations and global poses (Algorithm 1, lines 1 and 2). After a loop closure measurement is applied (line 3), the new information can be propagated back through the network of small joint distributions by repeated use of Equation (11) and (12) (lines 4 - 6). Since our goal is Back-End optimization of Relative Transformations, we will dub this approach BERT.

Algorithm 1: BERT using a modular representation of $\mathrm{p}\left(\boldsymbol{\tau},\boldsymbol{\rho}\right)$ for the single loop scenario.

```
  for (int $i = 1$; $i < \mathrm{N}$; $i$++) do
1     Compose $\boldsymbol{\rho}_i$ with $\boldsymbol{\tau}_i^{i+1}$ to find $\boldsymbol{\rho}_{i+1}$.
2     Calculate the joint covariance for $\mathrm{p}\left(\boldsymbol{\rho}_i,\boldsymbol{\tau}_i^{i+1},\boldsymbol{\rho}_{i+1}\right)$.
  end
3 Apply the loop closure measurement in a Kalman update
  to find $\mathrm{p}(\check{\boldsymbol{\tau}}_0^\mathrm{N})$.
4 Use Equation (11) and (12) to find $\mathrm{p}(\overset{\star}{\boldsymbol{\rho}}_{\mathrm{N}-1},\overset{\star}{\boldsymbol{\tau}}{}_{\mathrm{N}-1}^\mathrm{N},\overset{\star}{\boldsymbol{\tau}}{}_0^\mathrm{N})$
  where $\mathbf{x}_1$ corresponds to $\boldsymbol{\rho}_{\mathrm{N}-1}$ and $\boldsymbol{\tau}_{\mathrm{N}-1}^\mathrm{N}$, and $\mathbf{x}_2$
  corresponds to $\boldsymbol{\tau}_0^\mathrm{N}$.
  for (int $i = \mathrm{N}-1$; $i > 0$; $i$--) do
5     Extract the marginal distribution $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_i\right)$ from
      $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_i,\overset{\star}{\boldsymbol{\tau}}{}_i^{i+1},\overset{\star}{\boldsymbol{\rho}}_{i+1}\right)$
6     Find $\mathrm{p}\left(\overset{\star}{\boldsymbol{\rho}}_{i-1},\overset{\star}{\boldsymbol{\tau}}{}_{i-1}^{i},\overset{\star}{\boldsymbol{\rho}}_i\right)$ using
      Equations (11) and (12) where $\mathbf{x}_1$ corresponds to
      $\boldsymbol{\rho}_{i-1}$ and $\boldsymbol{\tau}_{i-1}^i$, and $\mathbf{x}_2$ corresponds to $\boldsymbol{\rho}_i$
  end
```

Small joint distributions of the form $\mathrm{p}(\boldsymbol{\rho}_i,\boldsymbol{\tau}_i^j,\boldsymbol{\rho}_j)$ are the important entity in BERT. We can use a collection of such distributions as a modular representation of the entire joint distribution, $\mathrm{p}(\boldsymbol{\tau},\boldsymbol{\rho})$. For example, in the simple scenario under consideration, our collection of small joint distributions is

$$\begin{gathered}
\mathrm{p}\left(\boldsymbol{\rho}_1,\boldsymbol{\tau}_1^2,\boldsymbol{\rho}_2\right)\\
\mathrm{p}\left(\boldsymbol{\rho}_2,\boldsymbol{\tau}_2^3,\boldsymbol{\rho}_3\right)\\
\mathrm{p}\left(\boldsymbol{\rho}_3,\boldsymbol{\tau}_3^4,\boldsymbol{\rho}_4\right)\\
\vdots\\
\mathrm{p}\left(\boldsymbol{\rho}_{\mathrm{N}-1},\boldsymbol{\tau}_{\mathrm{N}-1}^\mathrm{N},\boldsymbol{\rho}_\mathrm{N}\right)
\end{gathered}$$

We can update any of the individual variables within $\mathrm{p}(\boldsymbol{\rho}_i,\boldsymbol{\tau}_i^j,\boldsymbol{\rho}_j)$ and then propagate that information through the remaining small joint distributions based on Equations (11) and (12).

Notice that a given global pose variable can occur in multiple distributions. The belief about the global pose variables is identical in each small joint distribution by construction. The redundancy highlights the role of global poses as mediating variables. They serve to isolate our belief about each relative transformation, and they act as containers where information can be stored before propagating it through the rest of the network.

D. Comparison to CI Submap EKF-SLAM

We find some parallels and differences between BERT and the algorithms described in [23] and subsequent developments [24]–[26].

Similar to BERT, [23] divides the entire joint distribution into conditionally independent modules. In their case these modules are EKF SLAM sub-maps, primarily consisting of point features in the environment. They also show how duplicating some elements in multiple sub-maps allows the sub-maps to remain conditionally independent and efficiently share information gained at loop closure. During exploration the duplicated variables used in [23] consist of point feature estimates and vehicle pose estimates shared by temporally consecutive sub-maps.

We believe some important differences between [23] and our work arise from the way we represent the environment. In our concept of graph-based SLAM, we save key images of the environment that correspond to past vehicle poses. We do not reduce the raw images into a particular set of individual features. This removes feature position estimates from the back-end optimization, making it more efficient.

More importantly, because the vehicle was certainly at the pose represented by the saved image, reobserving the image is equivalent to reobserving the vehicle's actual pose at that previous point in time. In [25] the authors observe, "The price paid to maintain the conditional independence between submaps is some overhead in the size of the maps. We call overhead to *all those elements of a submap that cannot be observed from it, i.e., robot positions corresponding to*

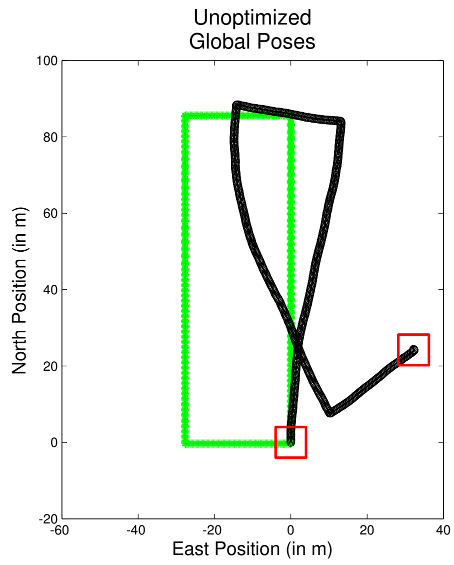

Fig. 4. True global poses in green (gray) verses the unoptimized estimate (black) based on composed odometry transformations. Red boxed regions are shown close up in Figures 5 & 6.

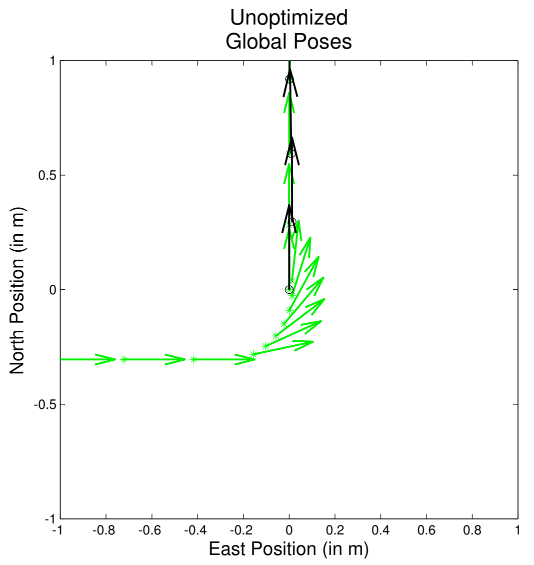

Fig. 5. A close up view of Figure 4 around the true beginning and end of the trajectory, where loop closure occurs. Arrows indicate the true (green/gray) and estimated (black) heading; the corresponding positions are marked (green/gray stars; black circles) at the base of each arrow.

the transitions between submaps and [additional] features included in the current submap because [they are needed to transmit new measurement information along the chain of intermediate submaps]," (emphasis added).

In [23] the authors use a simple, single loop closure scenario to illustrate their method just as we do here. Even in this simple case some feature estimates must be added as overhead to each of the sub-maps around the loop. Such is not the case with the approach we have presented.

We acknowledge that some non-trivial work remains on our part to establish whether we will need in the general case something equivalent to the overhead described in [23] and [25]. In [25] the authors are describing the extension of their algorithm to complex map topologies. Such an extension remains an item of future work for us which we discuss in Section VI. However, observing past poses in the form of saved images is a fundamental departure from feature-based maps.

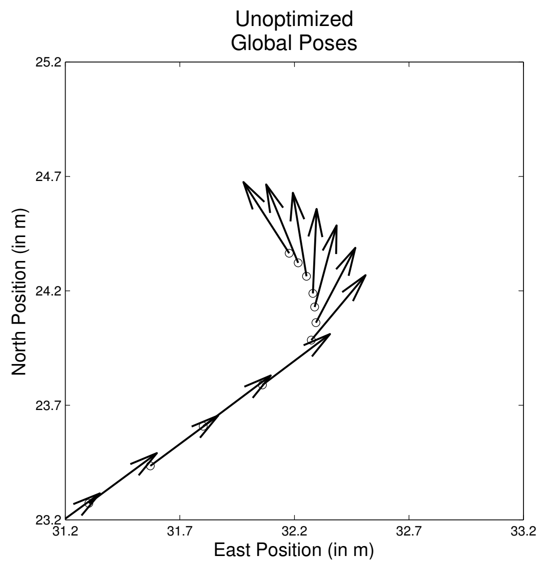

Fig. 6. A close up view of Figure 4 around the odometry-based estimate of the end of the trajectory. See also the caption on Figure 5

V. Results

A. Test Scenario

We have demonstrated in prior work [37], [38] that the roll and pitch of a multi-rotor vehicle can be estimated with high accuracy using an improved dynamic model in the front-end observer. It is also possible to use vision measurements in conjunction with frequent IMU and laser altimeter measurements to maintain accurate estimates of the vehicle's relative down position. Accordingly, we will confine our presentation of results to 2D optimization. Future work includes applying BERT to all six degrees of freedom.

For this test we generate front-end data using a Simulink simulation based on the dynamic model and state observers described in [37]. The vehicle flies around a rectangular hallway with a total trajectory length of about 225 meters. During exploration the vehicle designates new saved images about every 0.3 meters change in position or 10 degrees change in heading. This results in 735 relative transformations of type $\boldsymbol{\tau}_i^{i+1}$ that we will refer to as odometry transformations. In this experiment there is only a single loop closure transformation between the last pose and the origin.

The front-end saves the relative transformation estimates and their uncertainty for use in the back-end. The average error in the position change per transformation is 2.1 cm, or about 7% of the length of the transformation. The average error in the heading change of a transformation is 0.0115 radians (0.7 degrees). We note that the covariance matrix is not simply diagonal.

While these errors may not seem exceptionally high, the length of the trajectory allows for significant error to accumu-

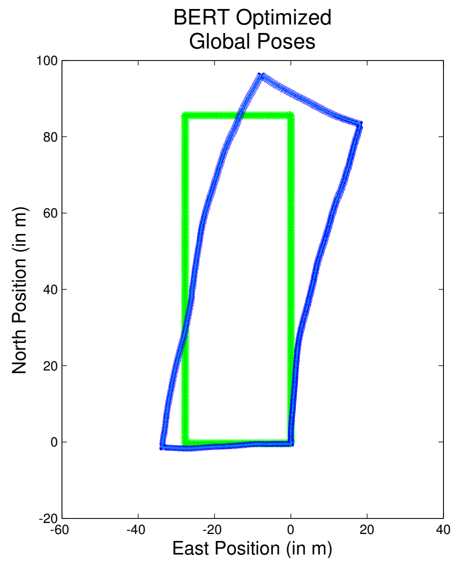

Fig. 7. True global poses in green (gray) verses estimates in blue (PUT SHADE OF GRAYSCALE VERSION HERE) from BERT as described in Algorithm 1. Not shown is the fact that one iteration of g^2o returns identical global poses.

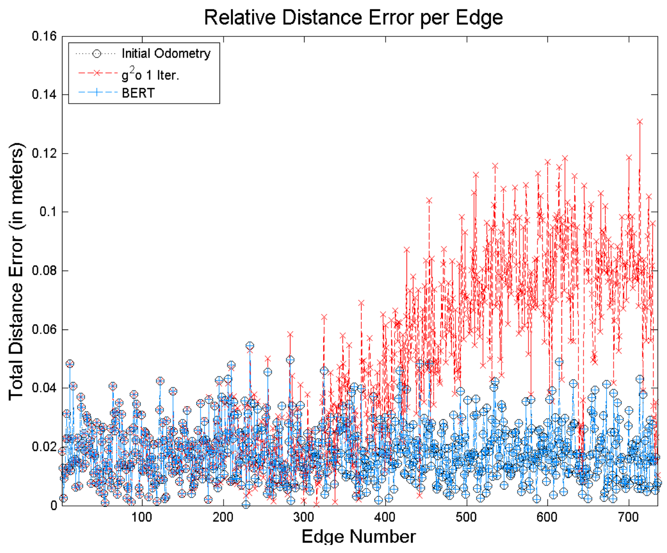

Fig. 8. The error in distance for each relative transformation estimate. We should expect optimized estimates to change only slightly from the originals because the 735 odometry transformations are being updated with information from a single loop closure. The important feature to note is the degradation toward the end of the trajectory caused by one iteration of g^2o.

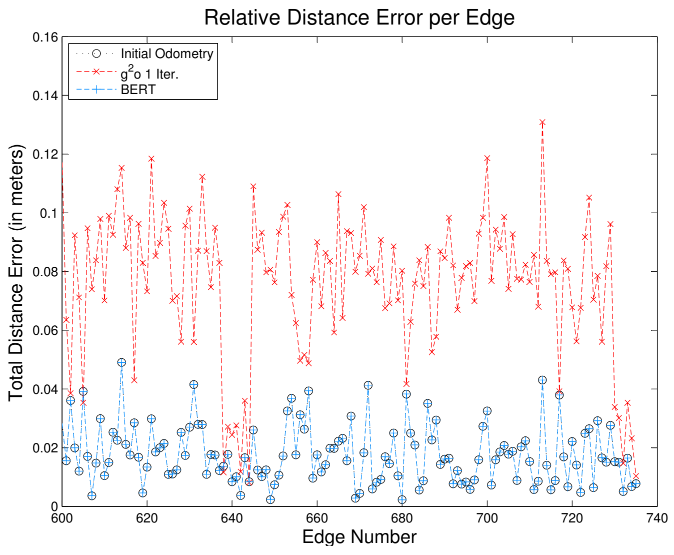

Fig. 9. A close up of Figure 8 highlighting the last 135 relative transformations.

late in the global pose estimates. Figure 4 shows an overhead view of the true global poses compared to the composition of unoptimized odometry transformations. Figures 5 and 6 show close up portions of Figure 4 to emphasize the error in the global pose estimate that has accumulated by the end of the trajectory.

We implement BERT using the Eigen C++ linear algebra library. We compare our results with those obtained using g^2o [17], which also relies on the Eigen library. The g^2o code is open-source and among the most recent work in back-end optimization. Also, as stated in Section II, the authors provide evidence that g^2o is at least as good as many other back-end optimization techniques both in accuracy and execution time.

B. A Single Iteration

We first apply BERT according to Algorithm 1 and compare our results to one iteration of g^2o. Figure 7 presents the optimized global poses achieved by our approach. These global poses are identical to the global poses returned by the first iteration of g^2o using a Gauss-Newton solver.

BERT runs in just 6.2 ms.[1] The first iteration of g^2o can takes about 25.5 ms. Of this time about 10.5 ms is spent numerically approximating the Jacobian of Equation (1). An analytical Jacobian should reduce the linearization time considerably. Subtracting all linearization time still makes the first iteration of g^2o around 9 ms longer than BERT. We assume that remaining difference is due to initialization g^2o requires as a more general software tool.

So BERT and the first iteration of g^2o produce the same global pose estimates in about the same time. However, the focus of BERT is optimization of the relative transformations.

[1]For timing results we use a 1.66 GHz Intel® Atom™ CPU (using a single thread) running Ubuntu 12.04. This computer is compatible with the size, weight, and power limitations of our vehicle.

Accordingly, we compare the effect BERT and g^2o have on the relative transformations. Like most back-end optimization algorithms, g^2o does not directly optimize transformations. Instead we must use the standard pose composition operator to find the new transformations that would be given by the optimized global poses g^2o provides. BERT returns the optimized relative transformations without extra computation.

Figure 8 plots the error in the estimated change of position in each relative transformation. A single iteration of g^2o causes the error to jump, especially near the end of the trajectory as shown in Figure 9. Before optimization the average error in this metric is about 2 cm. One iteration of g^2o causes that error to rise to about 8 cm (25% of an average transformation's

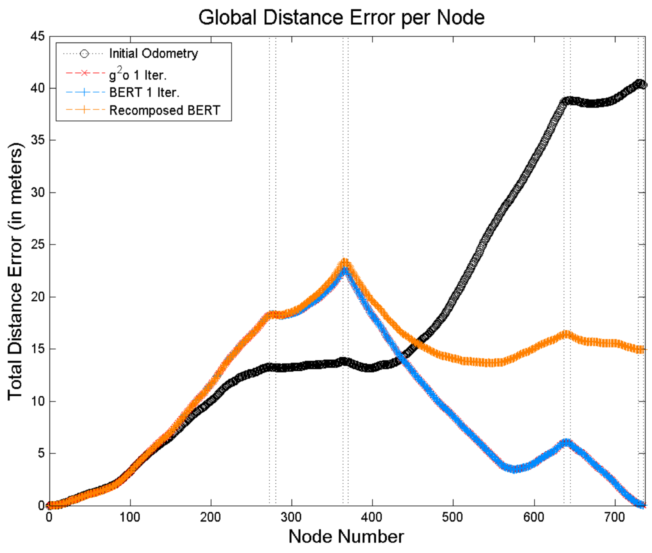

Fig. 10. This figure shows the error in global position estimates for each saved image along the trajectory after one iteration of g^{2}o and BERT. The vertical dashed lines mark poses occurring in the turns of the trajectory to facilitate comparisons with Figures 4 , 7 , and 12. The global poses returned by BERT and g^{2}o are identical. However, the global poses calculated by composing the relative transformations returned by BERT are significantly different. In particular, the global distance error at the end of the trajectory suggests the loop closure measurement might be reapplied to further improve the relative transformation estimates.

change in position) near the end of the trajectory.

We should expect 735 independent odometry transformations to be almost unchanged by a single loop closure measurement. Figures 8 and 9 show that BERT conforms to this intuition, producing relative transformations that are only slightly adjusted from the original front-end estimates. We have also observed that the relative transformations returned from BERT using Algorithm 1 are identical to those one would find using the direct approach to optimizing the transformations described in Section IV-C1. This bolsters our interpretation of global poses as mediating variables.

C. Multiple Iterations

A question about BERT naturally arises from the results presented so far. The relative transformation results from g^{2}o were found using the pose composition operator. In a manner of speaking, each $\boldsymbol{\tau}_i^j$ connects $\boldsymbol{\rho}_i$ to $\boldsymbol{\rho}_j$. Since BERT returns the same global pose estimates as one iteration of g^{2}o, what is significant about the different relative transformations returned by the two algorithms?

Standard back-end optimization techniques aim to find "the most likely configuration of the [global] poses given the [relative transformations delivered by the front-end]." [2] This differs fundamentally from BERT in that BERT returns an a posteriori estimate of the joint distribution $\mathrm{p}(\boldsymbol{\tau}, \boldsymbol{\rho})$. Standard back-ends use the pose composition operator to find new transformations implied by optimized poses. What then can we learn from the global poses one would calculate by composing the new relative transformations returned by BERT?

Figure 10 plots the error in position for each global pose along the trajectory. The equivalent global pose estimates from BERT using Algorithm 1 and from one iteration of g^{2}o show very little error in global position toward the end of the trajectory near the loop closure. This is intuitive given that a loop closure with the origin provides considerable information about the vehicle's true global pose.

However, if one were to recompose the relative transformation estimates returned by BERT, the error in global position at the end of the flight would still be significant (see again Figure 10). This observation motivates us to eliminate the global pose estimates obtained in the first application of BERT and repeat Algorithm 1. Doing so returns global pose and relative transformation estimates that are further refined.

Like g^{2}o and other traditional back-ends, BERT can be iterated until the solution converges. For the experiment in this paper, Figure 11 shows that BERT converges to a global pose solution that is slightly better than g^{2}o. Figure 12 gives a view of the BERT solution from the same overhead perspective as earlier plots. Figure 13 shows the evolution of the average global position error verses computation time.

We consider these results for global pose estimates to be useful, but the driving philosophy behind BERT is a focus on relative transformations. To that end, we conclude this section with some observations on the error in relative transformation estimates.

Equation (1), the squared Mahalanobis distance metric, measures the weighted deviation of optimized relative transformations from their initial values. Figure 14 (top) shows the evolution of Equation (1) per iteration of g^{2}o and BERT. Before the first iteration all of the deviation is concentrated in the difference between the measured and prior belief about the loop closure transformation. For g^{2}o, Equation (1) decreases monotonically with each iteration until it has converged to a final value. BERT converges to essentially the same value. However, BERT initially overshoots that value before settling into a steady state.

Figure 14 (top) also illustrates the error calculated by Equation (1) when the *true* relative transformations are compared to the original values. The result is two orders of magnitude higher than the minimum achieved by the optimization. The error surface defined by Equation (1) may not, in general, have a minimum value at the ideal solution.

The correct measure of map quality is the error calculated by comparing estimates to truth. Similar in form to Equation (1) we define the sum squared error

$$\bar{\epsilon}(\boldsymbol{\tau}) = \sum \left(\bar{\boldsymbol{\tau}}_i^j - \boldsymbol{\tau}_i^j\right)^\top \left(\bar{\boldsymbol{\tau}}_i^j - \boldsymbol{\tau}_i^j\right), \tag{14}$$

where $\bar{\boldsymbol{\tau}}_i^j$ indicates the true transformation from $\boldsymbol{\rho}_i$ to $\boldsymbol{\rho}_j$. Figure 14 (bottom) plots this metric for each iteration of g^{2}o and BERT. With g^{2}o, $\bar{\epsilon}(\boldsymbol{\tau})$ initially rises before settling into a new value that is slightly lower than before optimization. BERT, on the other hand, arrives at the same value of $\bar{\epsilon}(\boldsymbol{\tau})$ while making modest and monotonically decreasing changes.

With each iteration BERT improves the estimates of the global poses *and* the relative transformations. In the context of relative navigation, especially given limited computing power, we consider this a noteworthy feature. It may be sufficient during online operation to conserve computing resources by

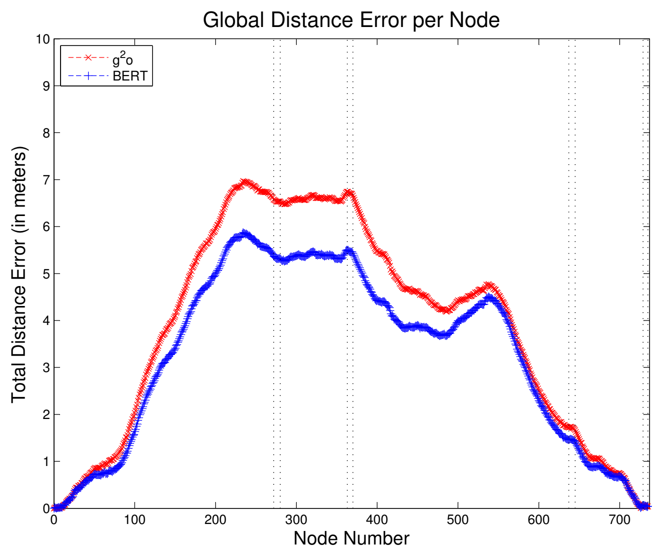

Fig. 11. This figure shows the error in global position estimates for each saved image along the trajectory after g^2o and BERT are run to convergence. See also the caption for Figure 10.

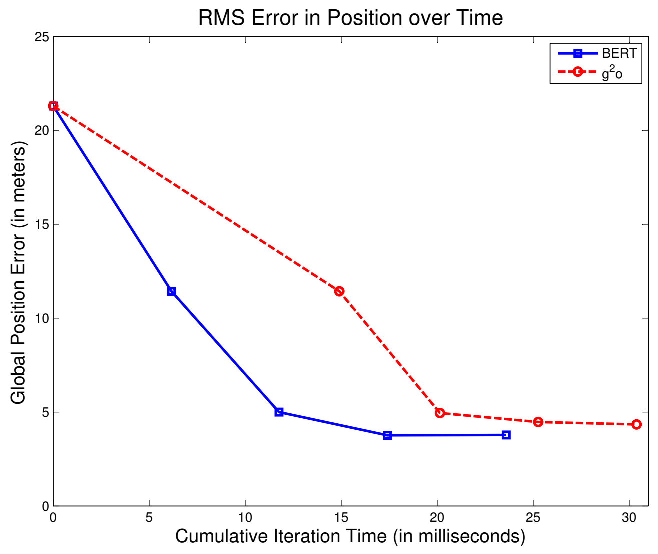

Fig. 13. RMS Error in global position estimates as a function of cumulative iteration time. Times shown for g^2o reflect subtracting all the linearization time out of each iteration; actual execution may be longer.

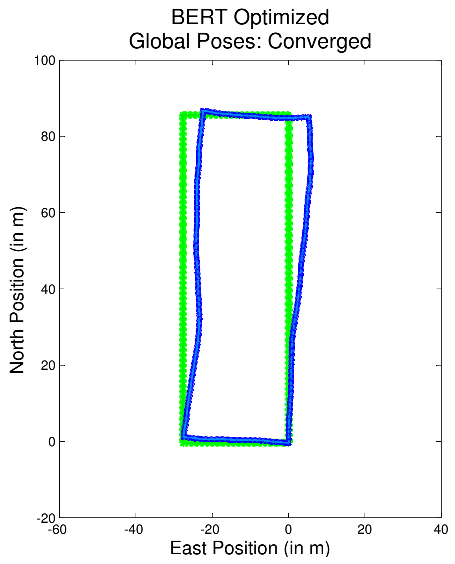

Fig. 12. True global poses in green (gray) verses estimates in blue (GRAYSCALE DESCRIPTION) from BERT run to convergence. Compare with Figures 4 and 7

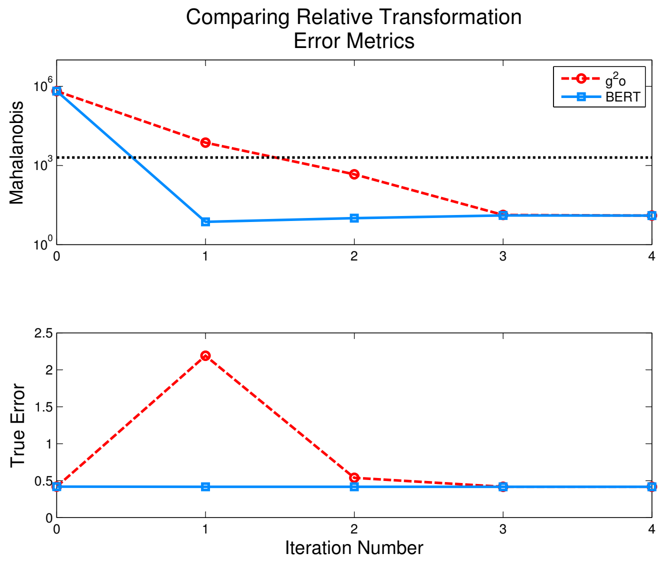

Fig. 14. Two error metrics for relative transformations and the performance of g^2o and BERT against those metrics. On top is the Mahalanobis distance (Equation (1)) measured with respect to the initial values of the estimated transformations. The horizontal dotted black line is the value of $\epsilon(\boldsymbol{\rho})$ when the true relative transformations are plugged into Equation (1). This top plot also suggests that BERT has something like a second order response to the impulse the system experiences at loop closure. On the bottom is the true sum squared error of the relative transformations. For the scenario under test the loop closure should only introduce a minor improvement in the relative transformations. Both methods reflect this in their final values, but g^2o requires three iterations before producing a reasonable result. BERT reduces the error in the relative transformations at each iteration without negative transient behavior.

running one iteration of BERT at a loop closure and waiting for additional loop closure measurements before optimizing further. Doing so with g^2o would be detrimental to the more immediately important relative transformation estimates.

VI. Conclusions and Future Work

A. Future Work

The most obvious item for future work is to extend BERT to general map topologies. A simple first step would be to retrace the loop in the current example, detecting the previously saved images along the way. The information from new estimates of each $\boldsymbol{\tau}_i^{i+1}$ can be incorporated into the map without any modification to the current approach.

To allow for arbitrary map topologies, we need to generate loop closure hypotheses between arbitrary global poses. The simple scenario presented above benefited from the fact that the origin involved in the loop closure was completely certain and independent. Therefore, we could generate a loop closure hypothesis without considering the correlation between the global poses involved. Our prior belief about arbitrary global poses before applying a loop closure should treat those poses as correlated. We expect that this will lead to additional computation in the composition steps represented by lines 1 - 2 in Algorithm 1, but we also expect the properties of conditional independence will keep the back-propagation of information (lines 5 - 6 of Algorithm 1) computationally neutral.

Applying BERT in general map topologies will likely benefit from considering additional formalisms to describe the problem. Back-end optimization is always posed in terms of estimating a random vector. However, this neglects the fact that relative transformations are temporally related. Considering the problem as a random process may lead to additional insights.

We also expect an effort to generalize BERT will benefit from using factor graphs. Our factor graphs will differ from those in [9]–[11] in that the relative transformations will be variables in the graph. Furthermore, the process described in Algorithm 1 seems to bear semantic resemblance to message passing algorithms such as those described in [20], Chapter 8.4. Factor graphs figure prominently in deriving such algorithms.

On the other hand, the lightweight computational burden of BERT might be useful in its current form as part of a visual odometry system. For example, we currently perform visual odometry by repeatedly comparing incoming frames to the last saved image of the environment. The comparisons produce measurements of the relative change in pose from that saved image. If we also compared temporally consecutive images we could save the relative transformations between them. The comparison to the saved image could then be treated like a global pose measurement, where the global poses are actually defined in the saved image's reference frame. It is common practice to smooth visual odometry estimates with, e.g., windowed bundle adjustment [39] to improve accuracy. We plan to investigate whether BERT can have a similar benefit at perhaps a lower computational cost.

Finally, we find the behavior in Figure 14 (top) interesting for an additional reason. Many authors briefly allude to an analogy between graph-based SLAM and a physical mass - spring system. Global poses are spoken of as masses and the saved relative transformations are described as springs. Equation (1) is said to describe the energy of the system configuration, and optimization adjusts the global poses (masses) until the network reaches a minimal energy configuration.

When we apply the loop closure measurement to the network it can be thought of like a step input to a second-order system. Initially the energy in the system responds to the step by overshooting its final value. As time (i.e. iteration) continues, the response settles down to a steady state with perhaps some ringing along the way. BERT exhibits this behavior, including some small ringing that is not obvious in Figure 14 (top) due to the logarithmic scale of the plot. To our knowledge, only [5] makes explicit use of the mass - spring system analogy to inform the development of their algorithm. In future work we expect to give closer attention to how we might draw insight from the analogy to further develop or explain BERT.

B. *Concluding Summary*

We recall here the context for our work. Our vehicle relies on the relative transformations for its front-end navigation. We also put a premium on computation time due to limited computational resources. We consider global pose estimates to be important but not time critical. The focus of our back-end optimization is to produce accurate relative transformations as efficiently as possible.

Our main contribution is an alternative approach to back-end optimization, BERT, that uses properties of conditional independence to efficiently estimate the joint distribution over relative transformations and global poses. We have illustrated in a simple scenario that BERT improves the estimates of the global poses *and* the relative transformations at each iteration of the algorithm. By comparison, a state-of-the-art back-end optimization tool produces similar global pose estimates at each iteration but requires multiple iterations to improve relative transformation estimates. This result is especially well-suited for our application, a small vehicle relying on relative transformations for its real-time navigation.

BERT offers other useful features. No extra computation is required to provide the optimized relative transformations, their associated marginal covariance matrices, or the marginal covariance matrices of the global poses. This latter measure of uncertainty can therefore be conveniently accessed if the front-end uses covariance gating to eliminate loop closure hypotheses. BERT also naturally admits global pose *measurements* when available, e.g. from intermittent GPS. Furthermore, BERT can be thought of as any-time algorithm, providing a useful result at any iteration and any stage of Algorithm 1. It is our hope that BERT will prove to be a useful alternative paradigm for back-end optimization in graph-based SLAM.

Acknowledgment

This research is supported through the U.S. Department of Defense SMART Scholarship program, `smart.asee.org`.

References

[1] F. Lu and E. Milios, "Globally Consistent Range Scan Alignment for Environment Mapping," *Autonomous Robots*, vol. 4, no. 4, pp. 333–349, 1997.

[2] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A Tutorial on Graph-Based SLAM," *IEEE Intelligent Transportation Systems Magazine*, pp. 31–43, Jan. 2010.

[3] K. Konolige and M. Agrawal, "FrameSLAM: From Bundle Adjustment to Real-Time Visual Mapping," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1066–1077, Oct. 2008.

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

[5] A. Howard, M. J. Mataric, and G. Sukhatme, "Relaxation on a Mesh : a Formalism for Generalized Localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 1055–1060.

[6] U. Frese, P. Larsson, and T. Duckett, "A Multilevel Relaxation Algorithm for Simultaneous Localization and Mapping," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 196–207, Apr. 2005.

[7] M. Montemerlo and S. Thrun, "Large-Scale Robotic 3-D Mapping of Urban Structures," in *International Symposium on Experimental Robotics*, 2004.

[8] S. Thrun and M. Montemerlo, "The GraphSLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 403–429, May 2006.

[9] 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, 2006.

[10] 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.

[11] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2 : Incremental Smoothing and Mapping Using the Bayes Tree," *International Journal of Robotics Research*, 2012.

[12] E. B. Olson, J. J. Leonard, and S. Teller, "Fast Iterative Alignment of Pose Graphs with Poor Initial Estimates," in *IEEE International Conference on Robotics and Automation*, no. May, 2006, pp. 2262–2269.

[13] G. Grisetti, D. Rizzini, C. Stachniss, E. B. Olson, and W. Burgard, "Online Constraint Network Optimization for Efficient Maximum Likelihood Map Learning," in *Proc. of the IEEE Int. Conf. on Robotics \& Automation (ICRA)*, 2008, pp. 1880–1885.

[14] G. Grisetti, C. Stachniss, and W. Burgard, "Nonlinear Constraint Network Optimization for Efficient Map Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 428–439, Sep. 2009.

[15] K. Konolige, G. Grisetti, R. Kummerle, W. Burgard, B. Limketkai, and R. Vincent, "Efficient Sparse Pose Adjustment for 2D mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Ieee, Oct. 2010, pp. 22–29.

[16] G. Grisetti, R. Kummerle, C. Stachniss, U. Frese, and C. Hertzberg, "Hierarchical Optimization on Manifolds for Online 2D and 3D Mapping," in *2010 IEEE International Conference on Robotics and Automation*, May 2010, pp. 273–278.

[17] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A General Framework for Graph Optimization," in *IEEE International Conference on Robotics and Automation*, 2011.

[18] M. Bosse, P. M. Newman, J. J. Leonard, M. Soika, W. Feiten, and S. Teller, "An Atlas framework for scalable mapping," *IEEE International Conference on Robotics and Automation*, pp. 1899–1906, 2003.

[19] C. Estrada, J. Neira, and J. D. Tardós, "Hierarchical SLAM : Real-Time Accurate Mapping of Large Environments," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 588–596, 2005.

[20] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006.

[21] M. Kaess, V. Ila, R. Roberts, and F. Dellaert, "The Bayes Tree : An Algorithmic Foundation for Probabilistic Robot Mapping," in *International Workshop on the Algorithmic Foundations of Robotics*, 2010.

[22] Y. Bar-Shalom, T. Kirubarajan, and X.-R. Li, *Estimation with Applications to Tracking and Navigation*. New York, NY, USA: John Wiley & Sons, Inc., 2002.

[23] P. Pinies and J. D. Tardós, "Large-Scale SLAM Building Conditionally Independent Local Maps: Application to Monocular Vision," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1094–1106, Oct. 2008.

[24] L. Paz, P. Pinies, J. D. Tardós, and J. Neira, "Large-Scale 6-DOF SLAM With Stereo-in-Hand," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 946–957, 2008.

[25] P. Pinies, L. M. Paz, and J. D. Tardós, "CI-Graph: An Efficient Approach for Large Scale SLAM," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 3913–3920.

[26] P. Pinies, L. M. Paz, D. Galvez-Lopez, and J. D. Tardós, "CI-Graph Simultaneous Localization and Mapping for Three-Dimensional Reconstruction of Large and Complex Environments Using a Multicamera System," *Journal of Field Robotics*, vol. 27, no. 5, pp. 561–586, 2010.

[27] A. Bachrach, S. Prentice, R. He, and N. Roy, "RANGE - Robust Autonomous Navigation in GPS-denied Environments," *Journal of Field Robotics*, vol. 28, no. 5, pp. 644–666, Sep. 2011.

[28] S. Shen, N. Michael, and V. Kumar, "Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 20–25.

[29] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a Navigation System for Autonomous Indoor Flying," in *2009 IEEE International Conference on Robotics and Automation*, no. Section III. Ieee, May 2009, pp. 2878–2883.

[30] ——, "A Fully Autonomous Indoor Quadrotor," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90–100, Feb. 2012.

[31] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "PIXHAWK : A System for Autonomous Flight using Onboard Computer Vision," in *IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 2992–2997.

[32] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Fox, and N. Roy, "Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera," in *International Symposium on Robotics Research*, 2011, pp. 1–16.

[33] V. Kumar and N. Michael, "Opportunities and Challenges with Autonomous Micro Aerial Vehicles," in *15th International Sympossium on Robotics Research*, 2011.

[34] R. Leishman, J. C. Macdonald, R. W. Beard, and T. McLain, "Relative Navigation and Control of a HexaKopter Rotorcraft," in *IEEE International Conference on Robotics and Automation*, 2012.

[35] U. B. Kjaerulff and A. L. Madsen, *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer, 2008.

[36] J. C. Macdonald, "Technical Note on Manipulating Gaussian Covariance Matrices," 2012. [Online]. Available: www.INeedAURL.byu.edu

[37] R. Leishman, J. C. Macdonald, R. W. Beard, and T. W. McLain, "Improved Use of Accelerometers in Estimating Quadrotor Attitude and Velocity," *IEEE Robotics & Automation Magazine (in review; preprint available from the authors)*.

[38] J. C. Macdonald, R. Leishman, R. W. Beard, and T. McLain, "Analysis of an Improved IMU-Based Observer for Multirotor Helicopters," *Journal of Intelligent & Robotic Systems (in review; preprint available from the authors)*.

[39] K. Konolige, M. Agrawal, and J. Sola, "Large Scale Visual Odometry for Rough Terrain," in *International Symposium on Robotics Research, November*, 2007.