Distributed Collaborative 3D Pose Estimation of Robots from Heterogeneous Relative Measurements: an Optimization on Manifold Approach

Joseph Knuth and Prabir Barooah

February 5, 2013

Abstract

We propose a distributed algorithm for estimating the 3-D pose (position and orientation) of multiple robots with respect to a common frame of reference when GPS is not available. This algorithm does not rely on the use of any maps, or the ability to recognize landmarks in the environment. Instead we assume that noisy *relative measurements* between pairs of robots are intermittently available, which can be any one, or combination, of the following: relative pose, relative orientation, relative position, relative bearing, and relative distance. The additional information about each robot's pose provided by these measurements are used to improve over self-localization estimates. The proposed method is similar to a pose-graph optimization algorithm in spirit: pose estimates are obtained by solving an optimization problem in the underlying Riemannian manifold $(SO(3)\times\mathbb{R}^3)^{n(k)}$. The proposed algorithm is directly applicable to 3-D pose estimation, can fuse heterogeneous measurement types, and can handle arbitrary time variation in the neighbor relationships among robots. Simulations show that the errors in the pose estimates obtained using this algorithm are significantly lower than what is achieved when robots estimate their pose without cooperation. Results from experiments with a pair of ground robots with vision-based sensors reinforce these findings. Further, simulations comparing the proposed algorithm with two state of the art existing collaborative localization algorithms identifies in what circumstances the proposed algorithm performs better then existing methods. Additionally, the question of trade-offs between cost (of obtaining a certain type of relative measurement) vs. benefit (improvement in localization accuracy) for the various types of relative measurements is considered.

1 Introduction

In recent years, interest in utilizing teams of autonomous mobile robots has grown rapidly. Multi-robot teams are beneficial in many ways. Utilizing a group of low cost robots may be more economical than risking a single, more costly robot. In search and rescue operations, a group of robots can cover a larger area than a single robot. In hazardous conditions, the innate redundancy of a group of robots may be necessary to prevent catastrophic loss of mission capability. Regardless of the application, localization is a crucial task for any autonomous mobile robot team.

Localization, the estimation of an autonomous robots global position and orientation, can be accomplished using a variety of sensors. Some of the more common sensors include Inertial Measurement Units (IMUs), vision based sensors (cameras, LIDARs), and Global Positioning System (GPS). Of the three, GPS is the only senor capable of providing global measurements of a robots position. However in many situations, GPS measurements may not be available, or may only be intermittently available. For example, a group of unmanned aerial vehicles (UAVs) operating in an urban environment may temporarily lose GPS measurements when the signal is blocked by large buildings. In such a situation, the global pose (position and orientation) can be obtained by integrating over the relative pose measurements obtained using IMUs or vision based sensors. This method of localization is referred to as *dead reckoning*.

Localization through dead reckoning can lead to a rapid growth in localization error (Olson et al., 2003; Knuth and Barooah, 2012a). When a team of robots is deployed, relative measurements between pairs of robots may be available. For instance, a robot equipped with a monocular camera can measure another robot's bearing with respect to itself whenever the other robot is in its field of view. These measurements, though noisy, provide additional information on the robots' poses that can be used to improve localization accuracy.

In this paper we propose a method for fusing measurements of various types to perform collaborative localization that improves localization accuracy over dead-reckoning. We assume all robots are equipped with proprioceptive sensors (cameras, IMUs, etc.) allowing each robot to measure its change in pose between time steps. We refer to these noisy measurements as *inter-time relative pose measurements*. In the absence of any other information, each robot can perform localization through dead reckoning using these noisy measurements. We further assume that each robot is equipped with exteroceptive sensors, so that noisy relative measurements between robots may become available intermittently. These measurements, which we call *inter-robot relative measurements*, can be one (or a combination) of the following: relative pose, relative orientation, relative position, relative bearing, and relative distance, between a pairs of robots. We provide a method to perform collaborative localization by fusing the inter-time and inter-robot relative measurements to obtain an estimate of the absolute pose of every robot.

We provide both a centralized and distributed algorithm for collaborative localization. In the centralized algorithm, all the measurements are assumed to be instantly available to a central processor. In the distributed algorithm, each robot performs localization using measurements from on-board sensors and information exchanged with neighboring

robots. The complexity of the computations performed by a robot is only a function of the number of its neighbors at any given time, not of the total number of robots. This makes the distributed algorithm scalable to arbitrarily large groups of robots. In addition, the communication requirements are small. At every update, a pair of neighboring robots needs to exchange only (i) the relative measurement between them and (ii) their current pose estimates.

Both simulations and experimental results (with a pair of P3-DX robots) are presented. In each case, the Monte-Carlo method was used to empirically obtain the bias and standard deviation of estimation error by conducting multiple trials. Results show that the proposed collaborative algorithm substantially improves upon dead-reckoning even when the team consists of a small number of robots. An examination of improvement vs. measurement type provides a basis for deciding whether the cost of the sensors required for obtaining certain types of measurements are commensurate with the localization accuracy they provide.

1.1 Related Work

Collaborative localization has been considered in the context of collaborative simultaneous localization and mapping (C-SLAM). Robots exchange local maps which are aligned and merged to improve robots' location estimates as well as to improve the maps; see (Ni et al., 2007; Andersson and Nygards, 2008; Indelman et al., 2011) and references therein. This requires the ability to identify common features in distinct maps generated by the robots. Recognizing common landmarks in distinct maps is often challenging. In addition, exchanging image data or maps between robots requires high bandwidth communication. A second body of work therefore considers the collaborative localization problem as one in which only relative measurements (of pose, position, orientation etc.) between pairs of robots are obtained and used to improve localization accuracy over self-localization. Our work falls into this category, which we term *collaborative localization from relative measurements*. Collaborative localization from relative measurements without map building is potentially useful to autonomous agents operating over large areas where collaborative mapping may be infeasible due to the large computation, memory, and communication requirements.

Theres is a large body of work on collaborative localization from relative measurements using a filtering-based approach, in both centralized and distributed settings. Extended Kalman Filter (EKF) - based formulation have been adopted in (Roumeliotis and Bekey, 2002; Martinelli et al., 2005; Panzieri et al.). Collaborative localization of two aircrafts from *indirect* relative measurement (observing common feature points on the ground) using the EKF has been proposed in (Melnyk et al., 2012). The papers (Fox et al., 2000; Rekleitis et al., 2002; Howard, 2011) propose particle-filter based methods for collaborative localization. (Leung et al., 2010b) proposes a Bayesian framework such that many filtering based methods can be applied within it. They propose decentralized algorithms to update a robot's belief of its own pose that is equivalent to the centralized belief whenever a certain Markovian property holds.

The pose-graph based methods (also known as network or view-based representations) have been popular and

successful in SLAM (see (Kummerle et al., 2011b) and references therein) and bundle adjustment (Sibley et al., 2009). A pose graph is a graph consisting of nodes that correspond to the unknown poses of the robots at various time instants and landmarks, and edges that correspond to observations between robot-landmark pairs. The best set of pose estimates are estimated by minimizing a cost function that quantifies how well a given set of poses predict the measurements. Under certain Gaussian assumptions on the measurement noise, the least-square solution obtained is the maximum likelihood (ML) estimate or of the poses. With additional statistical information on prior belief of the poses, again involving Gaussianity, the least square solution turns out to be the maximum a-posteriori estimate (MAP) as well. Pose graph optimization methods used in bundle adjustment and single robot SLAM can be directly applied to the multi-robot case. A centralized algorithm for ML collaborative localization through pose-graph optimization is presented in (Howard et al., 2002). Distributing the computations among the robots present additional challenges; (Kim et al., 2010) addresses them by using relative pose graphs. The MAP estimator for 2-D collaborative localization proposed in (Nerurkar et al., 2009) also belongs to this category. In (Knuth and Barooah, 2009) a pose-graph based algorithm for distributed collaborative localization in 3-D is proposed that uses a linearization of the relation between the pose measurements involving rotations.

When robots' absolute orientations are known, the problem of cooperative localization can be cast as a linear estimation problem (Sanderson, 1998; Barooah et al., 2010). In (Aragues et al., 2011), the authors use a two-phase approach to solve the 2-D collaborative localization problem where each phase is solved using linear estimation techniques: first the absolute orientations are estimated using relative orientation measurements, then these are used along with relative position measurements to obtain absolute position estimates, and finally an improved estimate of the absolute orientations are obtained from the position estimates.

1.2 Contributions

The method proposed here also belongs to the pose graph based approach: the estimation problem is formulated as an optimization defined by a graph, where nodes represent robot poses at various times and edges represent inter-time and inter-robot measurements. There is a subtle - but major - distinction between existing pose graph optimization methods, which we call Euclidean pose graph optimization (EPGO), and our proposed method. While existing methods use a vector space parameterization of orientation (such as the complex part of the unit quaternion representation) and then apply vector-space minimization techniques to search for the minima, we perform the optimization directly on the product Riemannian manifold in which the problem is naturally posed without relying on a specific parameterization. A gradient-descent method on the Riemannian manifold is then used for searching for the minima. The gradient descent algorithm is independent of the parameterization as well; any parameterization of the orientations can be used for numerical implementation without affecting the solution obtained. The advantages of doing so are discussed in

detail in Section 3 (see Remark 2). Simulations show that the proposed method based on Riemmanian manifold pose graph optimization, which we call RPGO, outperforms EPGO, the traditional pose graph optimization performed by vector-space based methods, when the measurement noise is large.

Two features of the problem of collaborative localization from relative measurements make it challenging: (i) the nonlinear relationship between the variables to be estimated (absolute positions and orientations) and measurements (relative position, orientation, bearing, distance etc.) and (ii) the non-commutativity of rotations in 3D. For Euclidean pose graph optimization in 3D, various tricks have to be employed to ensure that the result from vector-space optimization leads to accurate results; see (Grisetti et al., 2007) and references therein. Kalman filtering based approaches require, in addition to a specific vector-space parameterization of rotations, linearization of the non-linear relationships between variables and measurements. Indeed, most of the papers on collaborative localization consider the 2D case: a robot's pose is described by three scalars: x, y coordinate and an angle θ. Rotations in 2D commute and a vector-space parameterization does not lead to serious difficulties. Though many of these 2D algorithms can be extended to the 3-D case in principle, the extensions are far from straightforward. In fact, no details are provided for the extension to 3D case; all simulation and experimental results in (Sanderson, 1998; Roumeliotis and Bekey, 2002; Fox et al., 2000; Howard et al., 2002; Leung et al., 2010a; Aragues et al., 2011) are for the 2-D case. The belief propagation approaches of (Fox et al., 2000; Howard et al., 2002; Leung et al., 2010a), though developed without a specific parameterization, are feasible only if a vector-space parameterization of the orientations is used. Otherwise belief propagation of orientation estimates requires representation and computation of conditional probability density functions in $SO(3)$, a particularly challenging problem in itself. Some progress in fusion and propagation of pdfs in $SO(3)$ has been reported in (Wolfe et al., 2011; Long et al., 2012). In comparison with the KF-based methods described above, our method offers a distinct advantage. The linearization involved in the EKF requires a small angle approximation to hold at all times. Unless the time interval between two successive inter-vehicle measurements is extremely small, which is unlikely in many practical situations, the small angle approximation is violated, which is likely to lead to poor covariance updates and poor pose estimates. This is verified through simulations presented here.

This work, as compared to much of the earlier work on collaborative localization from relative measurements, differs in a number of novel ways. The proposed approach is (i) directly applicable to 3-D pose estimation, (ii) able to utilize heterogeneous measurement types (of the relative position, orientation, bearing, distance, or any combination thereof) between pairs of robots, and (iii) able to handle a time-varying neighbor relationship in the distributed setting. To the best of our knowledge, though earlier works have some of these features, no existing method has all of three features.

Ability to fuse heterogeneous relative measurements provides an important functionality. In particular, many of the papers mentioned above (Grisetti et al., 2007; Roumeliotis and Bekey, 2002; Fox et al., 2000; Howard et al., 2002; Aragues et al., 2011) require measurements of relative pose, i.e., position and orientation, between robot pairs.

While relative position can be obtained through stereo vision or laser range finders, obtaining relative orientation is quite challenging. It requires robots to be equipped with recognizable targets with known geometry; and even then, orientation measurements can be quite noisy. In fact, obtaining relative position measurements with stereo vision is also challenging due to small baseline enforced by the size of the robots. Laser range finders have limited range and therefore are not suitable to, say, UAVs where the distance between two UAVs may be large. However, even if only one camera of a robot sees a neighboring robot, a bearing measurement can be obtained. Or, radio frequency-based techniques such as TOA (time of arrival) measurements can be used to measure distance during wireless communication. With the ability to fuse all types of relative measurements, all available measurements can be utilized for improvement in localization accuracy. Though (Rekleitis et al., 2002; Martinelli et al., 2005) consider the case of heterogeneous measurements, their algorithms are limited to the 2-D case. Our work extends to 3-D the comparison between various measurement types initiated in (Rekleitis et al., 2002; Martinelli et al., 2005).

A third distinguishing contribution of the paper is the low communication requirements of the proposed algorithm. A crucial requirement in some of the prior work on collaborative localization is that of constant all-to-all communication among robots (Roumeliotis and Bekey, 2002; Howard et al., 2002; Caglioti et al., 2006; Martinelli et al., 2005; Nerurkar et al., 2009; Aragues et al., 2011). In contrast, the proposed distributed algorithm allows the number of neighbors of a robot (those it can exchange information with) to vary in an arbitrary manner with time. In the extreme case when the number of neighbors is always 0, the proposed algorithm simply degenerates into dead-reckoning.

Some preliminary aspects of the proposed method were reported in (Knuth and Barooah, 2012b). Compared to that paper, we make several novel contributions here. While the method in (Knuth and Barooah, 2012b) required all inter-robot measurements to be of the relative pose, here we extend the methodology to enable fusion of many types of measurements. As discussed above, this represents a significant extension. We also present detailed proofs of all the mathematical results, which were not present in (Knuth and Barooah, 2012b). Finally, here we include substantially more simulation and experimental results than in (Knuth and Barooah, 2012b).

The rest of the paper is organized as follows. The problem is precisely stated in Section 2. A centralized estimation scheme is described in Section 3, and a distributed algorithm that is inspired by the centralized scheme is described in Section 4. In Section 5 two existing state of the art algorithms are briefly reviewed. Simulation and experimental results with the proposed algorithms are presented in Sections 6 and 7, respectively. The paper concludes with a discussion in Section 8.

2 Problem statement

2.1 The Collaborative Localization Problem

Consider a group of r mobile robots indexed by $i = 1, \ldots, r$. Time is measured by a discrete counter $k = 0, 1, 2, \ldots$. Each robot is equipped with a local, rigidly attached frame of reference, that is, a coordinate system defined in the robot's local reference frame. We call the frame of reference attached to robot i at time k *frame* $i(k)$. Between any two frames of reference, say frame u and frame v, we denote the Euclidean transformation *from* v *to* u by $\mathbf{T}_{uv}$, where $\mathbf{T}_{uv}$ is an element of the special Euclidean group $SE(3)$. Specifically, if p_u is a point expressed in frame u and p_v is the same point expressed in frame v, then $p_u = \mathbf{T}_{uv} p_v$. We call $\mathbf{T}_{uv}$ the *relative pose* of frame v with respect to frame u.

Let frame 0 denote some fixed frame of reference that is common to all robots. The *absolute pose* of frame u is then given by the transformation $\mathbf{T}_{0u}$. We will often denote the absolute pose simply as $\mathbf{T}_u$. Robot i is said to be *localized* at time k when an estimate is known for the absolute pose of frame $i(k)$. We denote such an estimate by $\hat{\mathbf{T}}_{i(k)}$.

In this work, we consider the case when measurements of a robot's absolute pose, perhaps from measurements of a GPS and compass, are either not available or only rarely available. Instead, each robot is equipped with proprioceptive sensors such that, at every time k, the robot is able to obtain a relative pose measurement with respect to its previous pose. That is, a robot i at time k is able to measure the relative pose $\mathbf{T}_{i(k-1)i(k)}$. We refer to these measurements as *inter-time relative pose measurements*. Such measurements can be obtained with inertial sensors or vision based sensors Additionally, they need not be obtained from a sensor alone. Instead, a measurement could also be the estimate computed by fusing sensor measurements with predictions of the robot's motion from a dynamic/kinematic model.

In addition to the aforementioned proprioceptive sensors, each robot is equipped with exteroceptive sensors so that occasionally, a robot i is able to obtain a relative measurement of one or more other robots. We call these *inter-robot relative measurements*. If a robot i collects a measurement of robot j at time k, it can be one of the following:

- **Relative pose**: The Euclidean transformation from frame $j(k)$ to frame $i(k)$; denoted by the symbol $\mathbf{T}$.
- **Relative orientation**: The element of $SO(3)$ that describes the change in orientation from frame $j(k)$ to frame $i(k)$; denoted by the symbol $\mathbf{R}$.
- **Relative position**: The vector in $\mathbb{R}^3$ that describes the change in position between the frames $i(k)$ and $j(k)$, expressed in frame $i(k)$; denoted by the symbol $\mathbf{t}$.
- **Relative bearing**: The vector of unit length that points from frame $i(k)$ to frame $j(k)$, expressed in frame $i(k)$; denoted by the symbol $\boldsymbol{\tau}$.
- **Relative distance**: The distance between frame $i(k)$ and frame $j(k)$; denoted by the symbol δ.

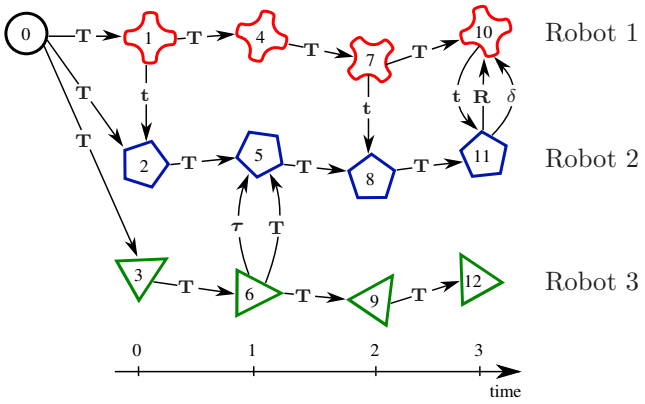


Figure 1: A time history of three robots up to time $k = 3$ with inter-time and inter-robot relative measurements. Each (robot, time) pair is labeled with the corresponding node index from $\mathcal{V}_0(3)$. Arrows indicate edges in $\mathcal{E}(3)$, i.e., relative measurements. Each edge is labeled to indicate the type of measurement. Each robot had GPS and compass measurements at the initial time $k = 0$. Thereafter, no other GPS or compass measurements were available.

Which pairs of robots will be able to obtain an inter-robot relative measurement will depend on many factors, including the kind of sensors they have on-board, the range of sensors, etc. An implicit assumption here is that a robot is able to uniquely identify another robot of which it obtains a relative measurement, so that there is no ambiguity on which pair of robots a relative measurement corresponds to.

The *collaborative localization problem* is the problem of estimating the pose of every robot at the current time k with respect to the common frame of reference by utilizing the inter-time and inter-robot measurements collected up to time k.

The situation above is best described in terms of a directed, time-varying, fully-labeled *graph* $\mathcal{G}(k) = (\mathcal{V}_0(k), \mathcal{E}(k), \ell(k))$, where nodes $\mathcal{V}_0(k)$ correspond to variables and edges $\mathcal{E}(k)$ to measurements, that shows how the noisy relative measurements relate to the absolute pose of each robot at every time step. The graph is defined as follows. For each robot $i \in \{1, \ldots, r\}$ and each time $k' \leq k$, a unique index (call it u) is assigned to the pair (i, k'). How this indexing is done is immaterial. The indices $\{1, \ldots, rk\}$ define a set $\mathcal{V}(k)$ that is a subset of the *node set* $\mathcal{V}_0(k)$ of the graph. We refer to the frame of reference attached to robot i at time k' as *frame* u, where u is the node assigned to the pair $(i,')$. We introduce another node, denoted by 0, that corresponds to the common frame of reference in which every robot's pose is to be determined. The node 0 is called the *grounded node*. The node set of the graph is then defined as $\mathcal{V}_0(k) := \mathcal{V}(k) \cup \{0\}$. The relative pose of frame u with respect to frame 0 is denoted simply by $\mathbf{T}_u$. We call the poses $\{\mathbf{T}_u\}_{u \in V(k)}$ the *node variables* of $\mathcal{G}(k)$.

The set of *directed edges* at time k, denoted $\mathcal{E}(k)$, corresponds to the noisy inter-time and inter-robot measurements collected up to time k. That is, suppose robot i is able to measure robot j's relative pose at time k', and let u, v

be the nodes corresponding to robots i, j at time k', respectively. Then for all $k \geq k'$, there exists a directed edge $e \in \mathcal{E}(k)$ corresponding to this measurement. Since robot i measures robot j, the edge e leaves node u and arrives at v, we denote this by $e \triangleright (u, v)$. Similarly, each inter-time relative pose measurements of a robot also creates an edge in the graph. To delineate the type of measurement, a label from the set $\{$ $\mathbf{T}$ (pose), $\mathbf{R}$ (orientation), $\mathbf{t}$ (position), $\boldsymbol{\tau}$ (bearing), δ (distance) $\}$ is attached to each edge. The map from the set of edges to the set of labels is denoted by $\ell(k)$. For each edge $e \in \mathcal{E}(k)$ where $e \triangleright (u, v)$, if $\ell(k)(e) = s$ that means there is a measurement of type s for frame v with respect to frame u, where $s \in \{\mathbf{T}, \mathbf{R}, \mathbf{t}, \boldsymbol{\tau}, \delta\}$. The noisy relative measurement associated with edge $e \triangleright (u, v)$ is denoted by $\hat{\mathbf{T}}_{u\,v}, \hat{\mathbf{R}}_{u\,v}, \hat{\mathbf{t}}_{u\,v}, \hat{\boldsymbol{\tau}}_{u\,v}$, or $\hat{\delta}_{u\,v}$ for $\ell(k)(e) = \mathbf{T}$ (pose), $\mathbf{R}$ (orientation), $\mathbf{t}$ (position), $\boldsymbol{\tau}$ (bearing), or δ (distance), respectively. A measurement of type $\mathbf{T}$ is really two measurements, one of type $\mathbf{R}$ and $\mathbf{t}$. We still use the nomenclature " measurement of type $\mathbf{T}$" for ease of comparison with prior work, since relative pose measurements are commonly considered in existing literature.

If the absolute pose in global GPS coordinates of at least one robot is known at time 0 through the use of a GPS and compass, then node 0 can be associated with a Terrestrial Reference Frame. When such measurements are not available, node 0 could correspond to the initial frame of reference of one of the robots. In either case, estimating the node variables is equivalent to determining the robots' poses with respect to frame of the grounded node 0. Without the grounded node, the problem of localization from relative measurements is indeterminate up to a rotation and a translation.

The graph $\mathcal{G}(k)$ is called the *measurement graph* at time k. Figure 1 shows an example of the graph corresponding to the measurements collected by 3 robots up to time index 3. Because each robot may be equipped with more than one sensor, multiple distinct edges may exist between a pair of nodes.

To ensure at least one estimate exists for every robot at each time k, we make the following assumption.

Assumption 1. *Each robot has access to an estimate of its absolute pose at time* 0.

Due to Assumption 1, an estimate of the pose of robot i at time time k (equivalently, the node variable $\mathbf{T}_u$, where node u corresponds to the pair (i, k)) can be computed by composing the inter-time relative pose measurements obtained by the robot i up to time k. This estimate is equivalent to robot i performing dead-reckoning. In practice, Assumption 1 holds if all robots have a GPS and compass measurement at time 0. If no GPS measurement is available, but each robot can obtain a relative pose measurement with respect to, say, robot 1, then again the assumption holds.

Often many more edges are present in the graph $\mathcal{G}(k)$ than those necessary to form a single estimate of the node variables; robots can benefit from collaborative localization in such a scenario. As an illustrative example, consider the scenario shown in Figure 1. The path $(0, 1, 4, 7, 10)$ provides the dead-reckoning estimate of robot 1 at time 3, or equivalently an estimate of $\mathbf{T}_{10}$. Similarly, the path $(0, 2, 5, 8, 11)$ provides as estimate for the node variable $\mathbf{T}_{11}$. Additionally, the three edges between nodes 10 and 11 corresponding to relative position, orientation, and distance

measurements all provide additional information about how $\mathbf{T}_{10}$ and $\mathbf{T}_{11}$ relate. Because there is noise in each relative measurement, by incorporating the information found in these three edges, we expect to get a better estimate of both $\mathbf{T}_{10}$ and $\mathbf{T}_{11}$. The goal of collaborative localization is to utilize all edges in the graph $\mathcal{G}(k)$ to improve over any estimates of the node variables that are obtained using only a single path. When the measurements are linearly related to the node variables, this can be accomplished by using the Best Linear Unbiased Estimator, as done in (Aragues et al., 2011; Barooah et al., 2010). In our case, the relationship between the measurements and node variables is nonlinear.

Remark 1. *The discussion above also shows that Assumption 1 is not necessary, but merely sufficient for localization. For robot i to be localized, all that is needed is that there exists a time k so that there is an undirected path from node u (where u corresponds to i, k) to node 0 such that the edges along this path are of type $\mathbf{T}$. In that case, an estimate of $\mathbf{t}_u$ can be obtained by concatenating the relative pose measurements along the path from 0 to u, and hence robot i is localized at time k. After that, even if no inter-robot relative measurements are available, robot i can perform dead reckoning. Assumption 1 is taken in order to simplify the presentation.*

2.2 The Distributed Collaborative Localization Problem

The problem stated above does not put any restriction on the access to the measurements collected up to time k. In particular, a method that assumes that all measurements collected by all the robots up to time k are instantly available to a central computer is allowed. However, when a large number of robots are involved, with communication accomplished through the use of low-bandwidth wireless links with limited range, such a centralized scheme is not feasible. In addition, retaining past measurements indefinitely will quickly exhaust both available memory and available processing capabilities of a centralized computing unit.

We now modify the problem by including constraints on communication and computation. In particular, a robot is now required to localize itself by using information that is available from on-board sensors and data it can collect from its *neighbors*. Two robots i and j are said to be neighbors at time k if they can communicate at time k.

The *distributed* collaborative localization problem is the following: each robot is to localize itself by using measurements collected by on-board sensors and information it can obtain by communicating with its neighbors. To simplify the development of the distributed algorithm, the following assumption is made:

Assumption 2. *If robot i can obtain a relative measurement of a robot j at time k, then i and j can communicate at that time.*

Though this assumption may seem strict, it can, in fact, always be satisfied: robot i simply drops any measurements involving j if it cannot communicate with j.

3 Centralized Collaborative Localization Algorithm

In this section we present a solution to the collaborative localization problem where all the relative measurements are instantly available to a central processor at each time k. To distinguish this algorithm from others pose-graph methods discussed in the sequel, we will refer to it as the Riemannian Pose Graph Optimization (RPGO) algorithm. The centralized solution naturally leads to a distributed scheme, which will be described in the next section.

3.1 Localization through optimization over a graph

Instead of addressing the problem of estimating the robots' current poses at time k, we examine the more general problem of estimating all the node variables $\{\mathbf{T}_u\}_{u\in\mathcal{V}(k)}$ of the measurement graph $\mathcal{G}(k)$, using the noisy relative measurements, $\{\hat{\mathbf{M}}_e\}_{e\in\mathcal{E}(k)}$, where fore each $(u,v)\triangleleft e\in E(k)$, $\hat{\mathbf{M}}_e$ is a measurement from node u to node v of one of the following: relative pose, orientation, position bearing, or distance.

In the following, we will interchangeably refer to both the absolute pose $\mathbf{T}_u$ and its corresponding orientation and position, $\mathbf{R}_u$ and $\mathbf{t}_u$ respectively, as node variables. Similarly, a pose measurement $\hat{\mathbf{T}}_{u\,v}$ will be identified with its corresponding orientation and position measurements, $\hat{\mathbf{R}}_{u\,v}$ and $\hat{\mathbf{t}}_{u\,v}$.

We pose the collaborative localization problem as an optimization of a cost function over the set of node variables, where the cost function measures how well a given set of absolute poses explains the noisy measurements collected up to time k. The initial condition for each node variable $\mathbf{T}_u=(\mathbf{R}_u,\mathbf{t}_u)$, $u\in\mathcal{V}(k)$ is given by the dead-reckoning estimate, whose existence is assured by Assumption 1.

Given a measurement $\hat{\mathbf{M}}_e$ between nodes u and v, let $c_e(\mathbf{R}_u,\mathbf{R}_v,\mathbf{t}_u,\mathbf{t}_v\,|\,\hat{\mathbf{M}}_e)\in\mathbb{R}_+$ denote the cost of the measurement $\hat{\mathbf{M}}_e$ as a function of the node variables. A suitable cost function must be chosen for each of the 4 measurements types (excluding pose measurements and instead considering a orientation/position pair) such that a higher cost indicates more disagreement between the observed measurements and the estimated node variables. This is done by considering the following fact. When the node variables and measurements are know exactly, each measurement can be given as a function of the node variables as follows:

$$\begin{aligned}\hat{\mathbf{R}}_{u\,v}&=\mathbf{R}_u^T\mathbf{R}_v & \hat{\boldsymbol{\tau}}_{u\,v}&=\mathbf{R}_u^T(\mathbf{t}_v-\mathbf{t}_u)/\|\mathbf{t}_v-\mathbf{t}_u\|\\ \hat{\mathbf{t}}_{u\,v}&=\mathbf{R}_u^T(\mathbf{t}_v-\mathbf{t}_u) & \hat{\delta}_{u\,v}&=\|\mathbf{t}_v-\mathbf{t}_u\|\end{aligned}\tag{1}$$

Since noise is present in the measurements, how much a noisy measurement differs from its value given as a function of the node variables provides a suitable cost.

In (1), $\hat{\mathbf{t}}_{u\,v}$, $\hat{\boldsymbol{\tau}}_{u\,v}$, and $\hat{\delta}_{u\,v}$ are all elements of a real vector space ($\mathbb{R}^3$ or $\mathbb{R}$) and so the standard Euclidean norms can be used to measure the distance between the noisy measurements and the equivalent node variable representations. In contrast, $\hat{\mathbf{R}}_{u\,v}$ and $\mathbf{R}_u^T\mathbf{R}_v$ are both elements of the manifold $SO(3)$ and are thus considered to be abstract operators and

not equated to any particular parameterization (such as rotation matrices). To this end, a suitable distance function on the manifold $SO(3)$ must chosen to measure their relative distance. Such a function is given by the Riemannian distance

$$d : SO(3) \times SO(3) \to \mathbb{R}_+, \quad (p, q) \mapsto \sqrt{-\frac{1}{2} \operatorname{Tr}\left(\log^2(p^T q)\right)} \tag{2}$$

where Tr denotes the trace function. More details on this distance function can be found in Appendix A.

The cost function at each time k is chosen as a sum of edge-costs over all edges (measurements):

$$f(\{\mathbf{T}_u\}_{u \in \mathcal{V}(k)}) := \sum_{(u,v) \lhd e \in \mathcal{E}(k)} c_e(\mathbf{R}_u, \mathbf{t}_u, \mathbf{R}_v, \mathbf{t}_v \,|\, \hat{\mathbf{M}}_e) \tag{3}$$

where $c_e(\mathbf{R}_u, \mathbf{t}_u, \mathbf{R}_v, \mathbf{t}_v \,|\, \hat{\mathbf{M}}_e)$, the cost for edge $e \triangleright (u, v)$, is given by

$$c_e(\mathbf{R}_u, \mathbf{t}_u, \mathbf{R}_v, \mathbf{t}_v) := \frac{1}{2} \times \begin{cases} d^2(\hat{\mathbf{R}}_{u\,v}, \mathbf{R}_u^T \mathbf{R}_v) + \|\hat{\mathbf{t}}_{u\,v} - \mathbf{R}_u^T(\mathbf{t}_v - \mathbf{t}_u)\|^2 & \text{if } \ell(k)(e) = \mathbf{T} \\ d^2(\hat{\mathbf{R}}_{u\,v}, \mathbf{R}_u^T \mathbf{R}_v) & \text{if } \ell(k)(e) = \mathbf{R} \\ \|\hat{\mathbf{t}}_{u\,v} - \mathbf{R}_u^T(\mathbf{t}_v - \mathbf{t}_u)\|^2 & \text{if } \ell(k)(e) = \mathbf{t} \\ \|\left(\hat{\boldsymbol{\tau}}_{u\,v} \|\mathbf{t}_v - \mathbf{t}_u\|\right) - \mathbf{R}_u^T(\mathbf{t}_v - \mathbf{t}_u)\|^2 & \text{if } \ell(k)(e) = \boldsymbol{\tau} \\ \|\left(\hat{\delta}_{u\,v} - \|\mathbf{t}_v - \mathbf{t}_u\|\right)\|^2 & \text{if } \ell(k)(e) = \delta \end{cases} \tag{4}$$

and $\|\cdot\|$ indicates the standard Euclidean norm on $\mathbb{R}^3$.

If the relative measurements were completely error free, the minimum value of the cost function would be 0. By minimizing the cost function, we expect to find an improved estimate for the absolute pose of each robot over what can be determined through dead reckoning alone. The cost function in (3) is similar to one proposed in (Tron and Vidal, 2009) for a static camera network; here the cost function changes with time.

3.2 Optimization over the product manifold

Finding the minimum of a function defined over a vector space has been studied extensively. However the function $f(\cdot)$ in (3) is defined on a curved surface, specifically, the product *Riemannian Manifold* $(SO(3) \times \mathbb{R}^3)^{n(k)}$ where $n(k) = |\mathcal{V}(k)|$, the cardinality of the set $\mathcal{V}(k)$. Though it is possible to parameterize the product manifold and thus utilize standard vector space optimization techniques, such a parameterization will inevitably introduce either an increase in dimensionality, and thus introduce constraints, or the parameterization will fail to be bijective over the entire domain. See (Yershova et al., 2010) for a discussion of the relevant parameterizations of $SO(3)$ and their associated problems.

In contrast, we desire a method that allows us to solve an unconstrained optimization problem in which the solution remains on the manifold during all intermediate steps. We accomplish this through use of a gradient descent algorithm on the product manifold.

Gradient descent in a Riemannian manifold is analogous to gradient descent in a vector space in the following sense. Given a smooth real valued function f defined on a manifold M, the gradient of f at $p \in M$, denoted $grad\, f(p)$, is a vector in the tangent space of M at p, which is denoted by T_pM. Just as in Euclidean space, $grad\, f(p)$ points in the direction of greatest rate of increase of f. An explicit expression for the gradient of the cost function (3) is provided in the next proposition; the proof is in Appendix C.

Proposition 1. *The gradient of the cost function shown in* (3) *at* $p = (\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_{n(k)}, \mathbf{t}_{n(k)}) \in \big(SO(3) \times \mathbb{R}^3\big)^{n(k)}$ *is*

$$grad\, f(p) = \big(grad\, f(\mathbf{R}_1), grad\, f(\mathbf{t}_1), \ldots, grad\, f(\mathbf{R}_{n(k)}), grad\, f(\mathbf{t}_{n(k)})\big),$$

where, for $u = 1, \ldots, n(k)$,

$$grad\, f(\mathbf{R}_u) = \sum_{e \in \mathcal{E}(k)} grad\, c_e(\mathbf{R}_u) \qquad grad\, f(\mathbf{t}_u) = \sum_{e \in \mathcal{E}(k)} grad\, c_e(\mathbf{t}_u)$$

The gradients of the edge cost c_e with respect to each manifold, $grad\, c_e(\mathbf{R}_u)$ and $grad\, c_e(\mathbf{t}_u)$ for all $u \in \mathcal{V}(k)$, are provided in Appendix D

Consider a function $f : S \to \mathbb{R}$ for some set S. When S is a vector space, the function S is minimized using gradient descent by iteratively updating an estimate $p_t \in S$ such that $p_{t+1} = p_t - \eta_t grad\, f(p_t)$ where $\eta_t \in \mathbb{R}_+$ is chosen so that $f(p_{t+1}) < f(p_t)$ (Boyd and Vandenberghe, 2004). An analogous update law can be defined when S is instead a Riemannian manifold. The key difference is that, in general, the addition operator is not defined on the manifold. Rather, the *parallel transport* map can be used to traverse the manifold in the direction of $-grad\, f$.

The parallel transport map on the product manifold $(SO(3) \times \mathbb{R}^3)^n(k)$ at a point

$$p := (\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_{n(k)}, \mathbf{t}_{n(k)}) \in (SO(3) \times \mathbb{R}^3)^{n(k)}, \tag{5}$$

denoted by $\exp_p$, is given by

$$\exp_p(\xi) = \big(\mathbf{R}_1 \exp(\mathbf{R}_1^T \xi_{\mathbf{R}_1}), \mathbf{t}_1 + \xi_{\mathbf{t}_1}, \ldots, \mathbf{R}_{n(k)} \exp(\mathbf{R}_{n(k)}^T \xi_{\mathbf{R}_{n(k)}}), \mathbf{t}_{n(k)} + \xi_{\mathbf{t}_{n(k)}}\big) \tag{6}$$

where $\xi = (\xi_{\mathbf{R}_1}, \xi_{\mathbf{t}_1}, \ldots, \xi_{\mathbf{R}_{n(k)}}, \xi_{\mathbf{t}_{n(k)}})$ is an element of the *tangent space* $T_p\big[(SO(3) \times \mathbb{R}^3)^{n(k)}\big] = T_{\mathbf{R}_1} SO(3) \times \cdots \times T_{\mathbf{t}_{n(k)}} \mathbb{R}^3$, and the $\exp(\cdot)$ function appearing in the right hand side of (6) is the map $\exp : \mathbb{L}(\mathbb{R}^3) \to \mathbb{L}(\mathbb{R}^3)$ defined by $exp(X) = \sum_{k=0}^{\infty} \frac{X^k}{k!}$ for $x \in \mathbb{L}(\mathbb{R}^3)$. The derivation of (6) is provided in Appendix B. The gradient descent law is

$$p_{t+1} = \exp_{p_t}(-\eta_t grad\, f(p_t)), \quad t = 0, 1, \ldots,$$

Algorithm 1: Centralized Riemannian Pose Graph Optimization

Input: $\mathcal{G}(k)$, all noisy measurements on $\mathcal{E}(k)$, an initial guess for each node variable $(\mathbf{R}_u, \mathbf{t}_u)$, $u \in \mathcal{V}(k)$.

Output: $\left\{(\hat{\mathbf{R}}_u, \hat{\mathbf{t}}_u)\right\}_{u\in\mathcal{V}(k)}$

$\hat{p} \to$ initial guess (p defined in (5));

repeat

  foreach $u \in \mathcal{V}(k)$ **do**

    Compute $grad\, f(\hat{\mathbf{R}}_u)$ and $grad\, f(\hat{\mathbf{t}}_u)$ for the cost f in (3) (as shown in Appendix D);

  end

  Determine $\eta^{(A)}$, the Armijo step size from (7) (with $\hat{p}$ for p_t);

  foreach $u \in \mathcal{V}(k)$ **do**

    $\hat{\mathbf{R}}_u \to \hat{\mathbf{R}}_u \exp\left(-\eta^{(A)} \hat{\mathbf{R}}_u^T grad\, f(\hat{\mathbf{R}}_u)\right)$;

    $\hat{\mathbf{t}}_u \to \hat{\mathbf{t}}_u - \eta^{(A)}\, grad\, f(\hat{\mathbf{t}}_u)$;

  end

until $\|grad\, f\left(\left\{(\hat{\mathbf{R}}_u, \hat{\mathbf{t}}_u)\right\}_{u\in\mathcal{V}(k)}\right)\| > \varepsilon$;

where $\eta_t \geq 0$ is the step size for iteration t. The parameter η_t is chosen as the *Armijo step size* $\eta_t^{(A)} = \beta^{N_t}\alpha$, where N_t is the smallest nonnegative integer such that

$$f(p_t) - f(exp_{p_t}(\beta^{N_t}\alpha\, grad\, f(p_t))) \geq \sigma\beta^{N_t}\alpha \|grad\, f(p_t)\|, \tag{7}$$

for scalar tuning parameters $\alpha > 0$, $\beta, \sigma \in (0, 1)$. The norm $\|\cdot\|$ defined on the vector space $T_p\left[(SO(3) \times \mathbb{R}^3)^n\right]$ is given by

$$\|(\xi_{\mathbf{R}_1}, \xi_{\mathbf{t}_1}, \ldots, \xi_{\mathbf{R}_{n(k)}}, \xi_{\mathbf{t}_{n(k)}})\|^2 = \sum_{u=1}^{n(k)} \left(\frac{1}{2}\,\mathrm{Tr}\left(\xi_{\mathbf{R}_u}^T \xi_{\mathbf{R}_u}\right) + \xi_{\mathbf{t}_u}^T \xi_{\mathbf{t}_u}\right).$$

which comes from the *Riemannian Metric*. More details on the Riemannian metric can be found in Appendix A. Theorem 4.3.1 in (Absil et al., 2008) grantees that the iterates p_t converges to a critical point of the cost function f defined in (3) as $t \to \infty$.

The pseudo-code of the centralized RPGO algorithm is given in Algorithm 1, where $\epsilon > 0$ is a user-specified accuracy threshold. Correctness of the RPGOalgorithm (convergence to a critical point of the cost function f in (3)) follows from (Absil et al., 2008, Theorem 4.3.1).

It should be noted that a chose of parameterization is necessary to explicitly compute the updates in the proposed gradient descent algorithm. However, because the gradient descent is carried out on the manifold, this chose does not impact the particular value of the node variables found at any step.

4 Distributed Collaborative Localization Algorithm

In this section we propose an algorithm for solving the distributed localization problem. The algorithm requires limited memory, processor power, and communication bandwidth for its execution. To distinguish this algorithm from the centralized RPGO algorithm presented in the previous section, we will refer to it as the Distributed Riemannian Pose Graph Optimization algorithm, or D-RPGO.

For each robot i, let $N_i^{(+)}(k)$ denote the set of all robots $j \in \{1, \dots, r\}$ such that, at time k, robot i can obtain a relative measurement with respect to j. Similarly, let $N_i^{(-)}(k)$ denote the set of all robots $j \in \{1, \dots, r\}$ such that, at time k, robot j can obtain a relative measurement with respect to i. The *neighbors* of robot i at time k are then given by the set $N_i(k) = N_i^{(+)}(k) \cup N_i^{(-)}(k)$. Due to Assumption 2, robot i can communicate with its neighbors $N_i(k)$ during time k.

Consider the *local measurement graph* $\mathcal{G}_i(k) = (\mathcal{V}_i(k), \mathcal{E}_i(k), \ell_i(k))$ of robot i, whose node set is simply the neighbors of i at time k along with the grounded node 0 and i itself: $\mathcal{V}_i(k) = N_i(k) \cup \{0, i\}$. The edges of $\mathcal{G}_i(k)$ correspond to the inter-robot measurements at time k between i and its neighbors, along with an edge $e \triangleright (0, j)$ for each $j \in \mathcal{V}_i(k)$ (see Figure 2 for an example). Each node in the local measurement graph $\mathcal{G}_i(k)$ is associated with an absolute pose of a robot at time k. No past poses belong to this graph. An edge $(p, q) \triangleleft e \in \mathcal{G}_i(k)$ (where $i = p$ or q) corresponds to an inter-robot relative measurement between robots p and q at time k. The additional edges $e \triangleright (0, j)$, $j \in \mathcal{V}_i(k)$ correspond to the "initial" estimate robot j's absolute pose at time k, denoted $\hat{\mathbf{T}}_{0\,j}(k)$. Each robot j obtains the estimate $\hat{\mathbf{T}}_{0\,j}(k)$ at time k by concatenating its pose estimate obtained at time $k-1$ with the noisy inter-time relative pose measurement describing its motion from $k-1$ to k. The estimate $\hat{\mathbf{T}}_{0\,j}(k)$ is then used as the measurement associated with edge $e \triangleright (0, j)$. The graph $\mathcal{G}_i(k)$ is a now a measurement graph since each edge has an associated noisy relative measurement. The edges $(0, j) \triangleleft e \in \mathcal{E}_i(k)$ for $j \in \mathcal{V}_i(k)$ ensures that Assumption 1 is satisfied for the local measurement graph $\mathcal{G}_i(k)$ for each i.

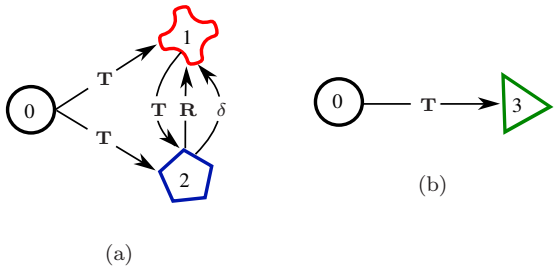


Figure 2: The local measurement graphs for the robots of Figure 1: (a) $\mathcal{G}_1(3)$ and $\mathcal{G}_2(3)$ (in this example they are the same) and (b) $\mathcal{G}_3(3)$.

The distributed algorithm works as follows. At each time k, every robot $i \in \{1, \dots, r\}$ forms an initial estimate

$\hat{\mathbf{T}}_{0\,i}(k)$ of its absolute pose as described above and obtains inter-robot relative measurements of each of its neighbors $j \in N_i^{(+)}(k)$. Robot i then transmits to each $j \in \mathcal{V}_i(k)$ its initial estimate of its absolute pose $\hat{\mathbf{T}}_{0\,i}(k)$, along with all inter-robot relative measurements between itself and j that it obtained at time k. Robot i receives in turn robot j's estimate of its current absolute pose $\hat{\mathbf{T}}_{0\,j}(k)$, along with relative measurements involving itself that j collected at that time. Robot i then executes Algorithm 1 on the local measurement graph $\mathcal{G}_i(k)$. The unknown node variables in this graph consist of its own pose and the poses of its neighbors (all at time k). After the computation, only the estimate of i's own current pose is retained in its local memory; all other computed values are discarded. Since the distributed algorithm is simply Algorithm 1 applied to a local measurement graph, it inherits the correctness property of Algorithm 1. Note that if robot i has no neighbors at time k, the distributed collaborative localization algorithm is equivalent to performing self-localization from inter-time relative measurements.

The local measurement graphs $\mathcal{G}_1(3)$, $\mathcal{G}_2(3)$ and $\mathcal{G}_3(3)$ corresponding to the example of Figure 1 are shown in Figure 2. At time $k = 3$, robot 3 can see no other robots, so it will update its absolute pose using the inter-time relative pose measurement alone, without the aid of any inter-robot relative measurements. Robots 2 and 1, in contrast, will use the relative measurements between them obtained at time $k = 3$ to update their pose estimates.

4.1 Communication requirements

In the distributed algorithm, all the past data is truncated and replaced by the current estimate. As a result, the size of the problem data for each robot i at time k (encoded in the local subgraph $\mathcal{G}_i(k)$), is bounded - and small - for all i and k. Specifically, for a robot i with N_i neighbors, the communication requirement of the distributed algorithm is $O(N_i)$. This can be seen as follows. A robot has to store current estimates of the node variables in its local measurement graph, and the number of nodes in that graph is $N_i + 1$. Each node variable can be represented by 6 numbers (3 for position and 3 for orientation). The robot also has to store the relative measurements between itself and its neighbors. The maximum memory requirement is when a robot is able to obtain measurements of all four type of relative measurements in the same time instant, (relative position, orientation, bearing, and distance). In that case, each neighbor causes a requirement of $3+3+2+1 = 9$ numbers to represent the inter-robot measurements (2 for orientation in 3D, and 1 for distance). Thus, the total memory (storage) requirement for robot i is at most $6(N_i + 1) + 9N_i$. At every time step, a pair of neighboring robots has to exchange (i) the relative measurement(s) between them and (ii) their current pose estimates: at most 15 numbers. If information is broadcast to all neighbors, a likely scenario with autonomous robots communicating wirelessly, a robot only has to transmit $15N_i$ numbers.

4.2 Computational Complexity

Just as the communications requirements scaled with the number of neighbors for a given robot i, so too does the computational complexity for a given robot's implementation of the $D-RPGO$ algorithm. Consider a robot i at time k, and let $N_i(k)$ indicate the number of neighbors for robot i at time k. The dimensionality of the optimization problem is then equal to N_i+1. Additionally, the number of operations necessary to find the gradient for robot i is $O(N_i(k)^3)$. This follows from the fact that the gradient is a $N_i(k)+1$ dimensional vector found by summing over the number of edges between neighbors, at most, $N_i(k)^2$.

5 Existing Methods of Collaborative Localization

In this section we compare the algorithm presented in section 3, referred to as the RPGO algorithm, with existing methods used for collaborative localization. Two state-of-the-art algorithms in particular are considered. The first algorithm we consider is based on standard pose graph optimization methods, see (Kummerle et al., 2011a) and references therein. The second algorithm considered utilizes an Indirect Extended Kalman filter (IEKF) to perform collaborative localization, similar to the method used in (Melnyk et al., 2012).

In section 6.3, simulations comparing the accuracy of both the standard pose graph algorithm, as well as the Indirect Extended Kalman filter with the algorithm presented in section 3 will be presented.

5.1 Pose Graph Optimization

The RPGO algorithm, first presented in section 3, is a special case of a larger class of algorithms called *pose graph optimization algorithms*. In contrast to the RPGO algorithm, which utilizes a cost function that is not dependent on any particular parameterization of $SO(3)$, standard pose graph optimization techniques require such a suitable parameterization to be used. In this section we consider one such popular parameterization given by the set of unit quaternions (Breckenridge, 1999). We refer to this algorithm as the Euclidean Pose Graph Optimization (EPGO) algorithm.

We again utilize the fully labeled, time varying graph presented in Section 2. For each node $u \in \mathcal{V}(k)$, let $\mathbf{q}_u$ denote the unit quaternion corresponding to the node variable $\mathbf{R}_u$. Similarly, given a orientation measurement $\hat{\mathbf{R}}_{u\,v}$, let $\hat{\mathbf{q}}_{u\,v}$ denote the corresponding unit quaternion.

Estimates for the node variables at time k are determined by minimizing the cost function

$$f(\{\mathbf{q}_u,\mathbf{t}_u\}_{u\in\mathcal{V}(k)}) := \sum_{(u,v)\lhd e\in\mathcal{E}(k)} g_e(\mathbf{q}_u,\mathbf{t}_u,\mathbf{q}_v,\mathbf{t}_v)^T P_e g_e(\mathbf{q}_u,\mathbf{t}_u,\mathbf{q}_v,\mathbf{t}_v) \tag{8}$$

where $P_e>0$ is a scaling matrix and $g_e(\mathbf{q}_u,\mathbf{t}_u,\mathbf{q}_v,\mathbf{t}_v)$ is a suitable *vector edge error* defined for each measurement

type. Though standard pose graph optimization is able to handle all measurement types considered thus far, for ease of exposition, only measurements of the relative position and orientation will be considered in the following discussion.

Given the unit quaternion parameterization, no canonical vector edge error exists. Instead, many choices are possible. One suitable choice, which is the vector edge error used in the subsequent comparisons, is given as follows:

$$g_e(\mathbf{q}_u, \mathbf{t}_u, \mathbf{q}_v, \mathbf{t}_v) = \begin{cases} \mathbf{q}_u^{-1} \otimes \mathbf{q}_v \otimes \hat{\mathbf{q}}_{u\,v}^{-1} - 1 & \text{if } \ell(k)(e) = \mathbf{R} \\ C(\mathbf{q}_u)^T(\mathbf{t}_v - \mathbf{t}_u) - \hat{\mathbf{t}}_{u\,v} & \text{if } \ell(k)(e) = \mathbf{t} \end{cases} \tag{9}$$

where $e \triangleright (u, v)$, $\otimes$ denotes quaternion multiplication (Breckenridge, 1999), and C denote the map that takes a unit quaternion to its corresponding 3×3 rotation matrix representation.

Many vector-space optimization algorithms can be used to search for the (9). One common choice is to use Levenberg-Marquadt, as was done in (Kummerle et al., 2011a). To implement Levenberg-Marquadt, a *minimal parameterization* is utilized. Given a unit quaternion $\mathbf{q}_u$, we denote the corresponding minimal parameterization as $\overline{\mathbf{q}_u}$ such that $\left[\overline{\mathbf{q}_u}^T \mathbf{q}_u^4\right]^T = \mathbf{q}_u^T$. In addition, a new operator $\boxplus : \mathrm{Dom}(\mathbf{q}_u) \times \mathrm{Dom}(\overline{\mathbf{q}_u}) \to \mathrm{Dom}(\mathbf{q}_u)$ is defined by

$$\mathbf{q}_u \boxplus \overline{\mathbf{q}_v} = \mathbf{q}_v \otimes \mathbf{q}_u. \tag{10}$$

To simplify the discussion, the parameterized node variables, minimally parameterized node variables, and vector edge error are all stacked as follows: $\mathbf{X} = \left[\mathbf{t}_1^T, \mathbf{q}_1^T, \ldots, \mathbf{t}_n^T, \mathbf{q}_n^T\right]^T$, $\mathbf{\Delta X} = [\mathbf{t}_{1'}^T, \overline{\mathbf{q}_{1'}}^T, \ldots, \mathbf{t}_{n'}^T, \overline{\mathbf{q}_{1'}}^T]^T$, $g = [g_1^T, \ldots, g_m^T]^T$, $P = \mathrm{diag}(P_1, \ldots P_m)$, where for all $i \in \mathcal{V}(k)$, $\mathbf{t}_{i'} \in \mathbb{R}^3$, $\overline{\mathbf{q}_{i'}} \in Dom(\overline{\mathbf{q}_i})$. Finally, we extend the $\boxplus$ operator to act on the new stacked state vectors as

$$\mathbf{X} \boxplus \mathbf{\Delta X} = [(\mathbf{X}_1 \boxplus \mathbf{\Delta X}_1)^T, \ldots, (\mathbf{X}_{2n} \boxplus \mathbf{\Delta X}_{2n})^T]^T$$

where the $\boxplus$ is defined for unit quaternions in (10) and reduces to the addition operator for position vectors. The Jacobian of the stacked vector error function g can now be computed as

$$J = \left[\frac{\partial g_i(\mathbf{X} \boxplus \mathbf{\Delta X})}{\partial \mathbf{\Delta X}_j}\Big|_{\mathbf{\Delta X}=0} \right]_{ij}$$

For the two measurement types considered, the Jacobian is explicitly given as follows. Consider an edge $e \triangleright (u, v)$ such that $\ell(e) = \mathbf{R}$. To simplify the notation, let $\mathbf{q} = \mathbf{q}_u^{-1}$ and $\mathbf{p} = \mathbf{q}_v \otimes \hat{\mathbf{q}}_{u\,v}^{-1}$. Then

$$\frac{\partial g_e(\mathbf{X} \boxplus \mathbf{\Delta X})}{\partial \Delta \mathbf{q}_h}\Big|_{\Delta \mathbf{X}=0} = \begin{bmatrix} p_4 \lfloor \overline{\mathbf{q}} \times \rfloor - p_4 q_4 id - q_4 \lfloor \overline{\mathbf{p}} \times \rfloor + \lfloor \overline{\mathbf{p}} \times \rfloor \lfloor \overline{\mathbf{q}} \times \rfloor \\ p_4 \overline{\mathbf{q}}^T + q_4 \overline{\mathbf{p}}^T - \overline{\mathbf{p}}^T \lfloor \overline{\mathbf{q}} \times \rfloor \end{bmatrix} I_{u,v}(h)$$

$$\frac{\partial g_e(\mathbf{X} \boxplus \mathbf{\Delta X})}{\partial \Delta \mathbf{t}_h}\Big|_{\mathbf{\Delta X}=0} = 0$$

Where $I_{u,v}(h) = 1$ if $h = u$, -1 if $h = v$, 0 otherwise, and $\lfloor \mathbf{v}\times \rfloor$ is the cross product matrix given by

$$\lfloor \bar{\mathbf{v}}\times \rfloor = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$$

for $\mathbf{v} = [v_1\, v_2\, v_3]^T$. Similarly, when $\ell(e) = \mathbf{t}$, we find

$$\left.\frac{\partial g_e(\mathbf{X} \boxplus \boldsymbol{\Delta}\mathbf{X})}{\partial \Delta \mathbf{q}_h}\right|_{\boldsymbol{\Delta}\mathbf{X}=0} = -2\lfloor (\mathbf{t}_v - \mathbf{t}_u)\times \rfloor I_u(h)$$
$$\left.\frac{\partial g_e(\mathbf{X} \boxplus \boldsymbol{\Delta}\mathbf{X})}{\partial \Delta \mathbf{t}_h}\right|_{\boldsymbol{\Delta}\mathbf{X}=0} = -I_{u,v}(h) C(\mathbf{q}_u)^T$$

Where I_u is the indicator function.

Levenberg-Marquadt is then carried out in the usual fashion, setting $H = J^T P J$, $b = -J^T P g(\mathbf{X})$. Fixing a time k, at each iteration of the Levenberg-Marquadt algorithm, the update state vector $\boldsymbol{\Delta}\mathbf{X}$ is given by solving the linear equation $(H + \lambda id)\boldsymbol{\Delta}\mathbf{X} = b$ for a suitably large λ. The current estimate of $\mathbf{X}$ is then updated by setting $\mathbf{X} \leftarrow \mathbf{X} \boxplus \boldsymbol{\Delta}\mathbf{X}$. This process is repeated until $\|\boldsymbol{\Delta}\mathbf{X}\|$ is suitably small.

Remark 2. *The orientations ($\mathbf{R}_{(\cdot)}$) that appear in our problem formations and through out the development of the RPGO algorithm are abstract rotation operators, or elements of $SO(3)$. No particular parameterization (rotation matrices, quaternions, ect.) is assumed during the development. In contrast, the cost function for the EPGO algorithm, (9) utilizes a specific parameterization in the form of unit quaternions. A different parameterization would necessarily lead to a different cost function, and possibly a different estimate delivered by the corresponding algorithm. The estimates found using the EPGO algorithm are a function of both the measurements and the chosen parameterization, rather then a function of the measurements alone.*

5.2 Indirect Extended Kalman Filter

We next consider an algorithm based on the Indirect Extended Kalman Filter (IEKF), similar to that used in (Melnyk et al., 2012). The two primary differences between the algorithm used in (Melnyk et al., 2012) and the IEKF algorithm given blow are as follows. **(i)** Inter-time relative measurements are of the relative pose, rather then measurements of the linear and angular velocities. This is motivated by the idea that measurements are captured by a camera, rather then an inertial measurement unit. **(ii)** The IEKF algorithm presented here utilizes measurements of the relative position between robots, where as the algorithm presented in (Melnyk et al., 2012) uses measurements of the bearing with respect to a time varying set of feature points. Under these two changes, the IEKF algorithm is able to function under the same circumstances as the RPGO algorithm presented in Section 3.

As with the EPGO algorithm, and in contrast to the RPGO algorithm, the IEKF algorithm requires a suitable parameterization of $SO(3)$ to be used. Unit quaternions are again chosen for this parameterization. To remain

consistent with the notation used in the literature, a rotation will be parameterized by its inverse quaternion. That is, given a node variable $\mathbf{R}_u$ where u maps to time k and robot i, the corresponding quaternion $\mathbf{q}_k^i$ is given by

$$\mathbf{q}_i^k = C(\mathbf{R}_u^T) \tag{11}$$

where C denotes the map that takes a unit quaternion to its corresponding 3×3 rotation matrix representation.

The previous definition introduces a new notation, disjoint from the rest of the paper, but necessary to properly describe the IEKF algorithm. For the remainder of this section, unless specifically noted otherwise, superscript notation will denote time and subscript will denote robot index.

The state vector for robot i at time k is given by

$$X_i^k = [(\mathbf{q}_i^k)^T \, (\mathbf{t}_i^k)^T]^T \tag{12}$$

where $\mathbf{t}_i^k$ is the position of robot i at time k and $\mathbf{q}_i^k$ is the orientation of robot i and time k as defined in (11). The full state vector X^k is given by stacking the state vector for each robot and $\hat{\mathbf{q}}_i^k$, $\hat{\mathbf{t}}_i^k$, $\hat{X_i^k}$, $\hat{X^k}$ denote estimates of the corresponding true states.

The *error state* vector for robot i at time k corresponding to (12) is given by

$$\tilde{X}_i^k = [(\boldsymbol{\delta\theta}_i^k)^T \, (\tilde{\mathbf{t}}_i^k)^T]^T \tag{13}$$

where $\boldsymbol{\delta\theta}_i^k$ is the angle-error vector defined by the error quaternion $\tilde{\mathbf{q}}_i^k = \mathbf{q}_i^k \otimes (\hat{\mathbf{q}}_i^k)^{-1} \approx [\frac{1}{2}(\boldsymbol{\delta\theta}_i^k)^T \; 1]^T$. This approximation is made by assuming that $d(id, C(\tilde{\mathbf{q}}_i^k))$ is sufficiently small. The state propagation model is chosen to correspond with the kinematic model when inter-time relative position and orientation measurements are available,

$$\mathbf{q}_i^{k+1} = (\mathbf{q}_i^{k,k+1})^{-1} \otimes \mathbf{q}_i^k \tag{14}$$

$$\mathbf{t}_i^{k+1} = \mathbf{t}_i^k + C(\mathbf{q}_i^k)^T \, \mathbf{t}_i^{k,k+1} \tag{15}$$

where $\mathbf{q}_i^{k,k+1}$, $\mathbf{t}_i^{k,k+1}$ denote the change in orientation and position respectively for robot i between time k and time $k+1$. The measured change in orientation and position for robot i is modeled as $\hat{\mathbf{q}}_i^{k,k+1} = \tilde{\mathbf{q}}_i^{k,k+1} \otimes (\mathbf{q}_i^{k,k+1})^{-1}$ and $\hat{\mathbf{t}}_i^{k,k+1} = \mathbf{t}_i^{k,k+1} + \tilde{\mathbf{t}}_i^{k,k+1}$ where $\tilde{\mathbf{q}}_i^{k,k+1} \approx [\frac{1}{2}(\boldsymbol{\delta\theta}_i^{k,k+1})^T \; 1]^T$. Here $\boldsymbol{\delta\theta}_i^{k,k+1}$ and $\tilde{\mathbf{t}}_i^{k,k+1}$ are assumed to be zero-mean white Gaussian noise processes. The linearized error state equation is then given by $\tilde{X}^{k+1} = F^k \tilde{X}^k + G^k[(\boldsymbol{\delta\theta}_1^{k,k+1})^T \, (\tilde{\mathbf{t}}_1^{k,k+1})^T \ldots (\boldsymbol{\delta\theta}_n^{k,k+1})^T \, (\tilde{\mathbf{t}}_n^{k,k+1})^T]^T$ for $F^k = \text{diag}(F_i^k)$, $G^k = \text{diag}(G_i^k)$ and

$$F_i^k = \begin{bmatrix} \mathbf{I}_3 & -C(\hat{\mathbf{q}}_i^k)^T \lfloor \hat{\mathbf{t}}_i^{k,k+1} \times \rfloor \\ 0 & C(\hat{\mathbf{q}}_i^{k,k+1})^T \end{bmatrix}$$

$$G_i^k = \begin{bmatrix} -C(\hat{\mathbf{q}}_i^k)^T & 0 \\ 0 & C(\hat{\mathbf{q}}_i^{k,k+1})^T \end{bmatrix}.$$

Where $\lfloor \mathbf{v}\times \rfloor$ is defined as in Section 5.1.

At each time step, the noisy measurements of the relative change in position and orientation are used to update the state estimate and covariance as

$$\hat{\mathbf{X}}_i^{k+1|k} = \begin{bmatrix} (\hat{\mathbf{q}}_i^{k,k+1})^{-1} \otimes \hat{\mathbf{q}}_i^k \\ \hat{\mathbf{t}}_i^k + C(\hat{\mathbf{q}}_i^k)^T \hat{\mathbf{t}}_i^{k,k+1} \end{bmatrix}$$

and

$$P^{k+1|k} = F^k P^k (F^k)^T + G^k Q^k (G^k)^T$$

where $Q^k = \mathrm{E}[[(\boldsymbol{\delta\theta}_1^{k,k+1})^T \ (\tilde{\mathbf{t}}_1^{k,k+1})^T \dots (\boldsymbol{\delta\theta}_n^{k,k+1})^T \ (\tilde{\mathbf{t}}_n^{k,k+1})^T]^T [(\boldsymbol{\delta\theta}_1^{k,k+1})^T \ (\tilde{\mathbf{t}}_1^{k,k+1})^T \dots (\boldsymbol{\delta\theta}_n^{k,k+1})^T \ (\tilde{\mathbf{t}}_n^{k,k+1})^T]]$. Here the superscript $k+1|k$ is used to indicate this is a temporary estimate of the state at time $k+1$ utilizing only the inter-robot relative measurements available up to time k. If no inter-robot measurements are available at time $k+1$, then $\hat{\mathbf{X}}_i^{k+1} = \hat{\mathbf{X}}_i^{k+1|k}$, $P^k = P^{k+1|k}$. When inter-robot measurements are available, they are utilized as follows. Given an edge $e_{ij} \in \mathcal{E}(k)$ corresponding to an inter-robot measurement at time k from robot i to j, the measurement is modeled by

$$\mathbf{z}_{e_{ij}}^k = C(\mathbf{q}_i^k)(\mathbf{t}_j^k - \mathbf{t}_i^k) + \zeta_{e_{ij}}$$

The noise $\zeta_{e_{ij}}$ is assumed to be a zero-mean, normally distributed with $\mathrm{E}[\zeta_{e_{ij}}(\zeta_{e_{ij}})^T] =: R^k$. The linearized error-measurement model is then given

$$\tilde{\mathbf{z}}_{e_{ij}}^k = \mathbf{z}_{e_{ij}}^k - \hat{\mathbf{z}}_{e_{ij}}^k \approx H_{e_{ij}}^k \tilde{\mathbf{X}}^k + \zeta_{e_{ij}}$$

where

$$\hat{\mathbf{z}}_{e_{ij}}^k := C(\hat{\mathbf{q}}_i^k)\left(\hat{\mathbf{t}}_j^k - \hat{\mathbf{t}}_i^k\right)$$
$$H_{e_{ij}}^k = \mathbf{e}_i^T \odot H_i^k + \mathbf{e}_j^T \odot H_j^k.$$

In the previous equation $\odot$ refers to the Kronecker product, $\mathbf{e}_i$ indicates the ith standard basis vector, and H_i^k, H_j^k are

$$H_i^k = \begin{bmatrix} -C(\hat{\mathbf{q}}_i^k) & \lfloor C(\hat{\mathbf{q}}_i^k)\left(\hat{\mathbf{t}}_j^k - \hat{\mathbf{t}}_i^k\right)\times \rfloor \end{bmatrix}$$
$$H_j^k = \begin{bmatrix} C(\hat{\mathbf{q}}_i^k) & 0 \end{bmatrix}.$$

Finally, given the set of measurements $\left\{\mathbf{z}_{e_1}^k, \cdots, \mathbf{z}_{e_m}^k\right\}$ acquired at time k, the total error-measurement $\tilde{\mathbf{z}}^k$ and error-measurement matrix H^k are given by stacking all the error-measurements $\tilde{\mathbf{z}}_{e_{ij}}^k$ and matrices $H_{e_{ij}}^k$ respectively.

The IEKF is then implemented as follows. Let $S^{k+1} = H^{k+1} P^{k+1|k} (H^{k+1})^T + R^{k+1}$. The Kalman gain is $K^{k+1} = P^{k+1|k}(H^{k+1})^T (S^{k+1})^{-1}$ and the update state is $\Delta\mathbf{X}^k = K^{k+1}(z^{k+1} - H^{k+1}\hat{\mathbf{X}}^{k+1|k})$. The IEKF state is

then updated to include the measurement z by

$$\hat{\mathbf{X}}^{k+1} = \hat{\mathbf{X}}^{k+1|k} \oplus \Delta\mathbf{X}^{k+1}$$

$$P^{k+1|k+1} = (I - K^{k+1}H^{k+1})P^{k+1|k}(I - K^{k+1}H^{k+1})^T + K^{k+1}R^{k+1}(K^{k+1})^T.$$

where $\oplus$ is defined component wise as follows.

$$\hat{\mathbf{X}}_i^{k+1|k} \oplus \Delta\mathbf{X}_i^{k+1} =: \begin{bmatrix} \hat{\mathbf{q}}_i^{k+1|k} \\ \hat{\mathbf{t}}_k^{k+1|k} \end{bmatrix} \oplus \begin{bmatrix} \Delta\theta_i^{k+1} \\ \Delta\mathbf{t}_i^{k+1} \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{q}_i^{k+1} \otimes \hat{\mathbf{q}}_i^{k+1|k} \\ \hat{\mathbf{t}}_i^{k+1|k} + \Delta\mathbf{t}_i^{k+1} \end{bmatrix}$$

where

$$\Delta\mathbf{q}_i^{k+1} = \begin{bmatrix} \frac{1}{2}(\Delta\theta_i^{k+1})^T & \left(1 - \frac{1}{4}\Delta\theta_i^{k+1}\right)^{1/2} \end{bmatrix}^T.$$

6 Simulation Results

In this section we present simulations comparing the centralized and distributed algorithms and study the effect of collaboration on localization accuracy using each of the inter-robot measurement types.

First, in section 6.1, we consider the case that all inter-robot relative measurements are of the relative pose. We examine the difference in localization accuracy between the centralized and distributed algorithms, as well as the effect of increasing number of robots on localization accuracy. In Section 6.2 we consider the heterogeneous measurement case in which the inter-time relative measurements are of relative pose, but the inter-robot relative measurements may be of relative pose, orientation, position, bearing, or distance. We examine the effects these various measurement types have on the accuracy of the location estimates.

To study the change in localization accuracy, we consider the following metrics. The *position (estimation) error* of robot i is $\mathbf{e}_i(k) := \hat{\mathbf{t}}_i(k) - \mathbf{t}_i(k)$, where $\mathbf{t}_i(k)$ is its global position at time k and $\hat{\mathbf{t}}_i(k)$ is the estimate of this position. The *bias* in the position estimation error of robot i is defined as $\|\,\mathrm{E}[\mathbf{e}_i(k)]\|$, where $\mathrm{E}[\,\cdot\,]$ denotes expectation and $\|\cdot\|$ refers to 2-norm, and the *standard deviation* as $\sqrt{\mathrm{Tr}\,(Cov(\mathbf{e}_i(k), \mathbf{e}_i(k)))}$, where $Cov(\,\cdot\,,\,\cdot\,)$ denotes covariance.

6.1 Centralized RPGO vs. DRPGO

To compare the centralized and distributed algorithms, we examine the localization of a group of 5 robots. All inter-robot relative measurements were of the relative pose. Each of the 5 robots travels along a distinct zig-zag path in 3-D, shown in Figure 3(a). Two robots could obtain relative pose measurements at time k if the Euclidean distance between them at that time was less than 7m. Due to assumption 2, communication between robots is possible between at least those pairs with a distance less than 7m. Furthermore, 25% of these potential measurements were dropped to simulate random failure. A plot of the number of neighbors of robot 1 over time is shown in Figure 3(b). The

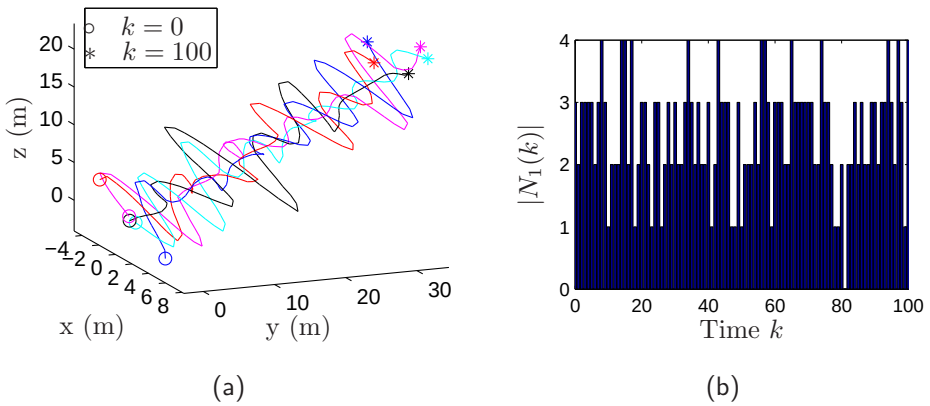


Figure 3: The (a) 3D trajectories for robots 1 through 5, used in all simulations and (b) the number of neighbors of robot 1 as a function of time.

orientation measurements for each relative pose (both inter-robot and inter-time) were corrupted by independent identically distributed (i.i.d.) unit quaternions drawn from a Von Mises-Fisher distribution (Mardia and Jupp, 2000) centered around the zero-rotation quaternion and with a concentration parameter of $10,000$. Noise in the relative translation measurements was simulated by adding i.i.d zero-mean normal random variables with covariance matrix $I_{3\times 3} \times 10^{-6}$.

Figure 4 shows the bias and standard deviation in position error of a single robot acting in a 5 robot team, estimated using a 100-iteration Monte Carlo simulation. The group of robots performed localizing using either the centralized or distributed collaborative localization algorithms with all inter-robot measurements being of the relative pose. The plots indicate that the improvement in localization accuracy with the distributed algorithm is quite close to that with the centralized algorithm. This is promising since the distributed algorithm is applicable to large teams of robots in highly dynamic scenarios that can lead to arbitrary time variation in neighbor relationships. The centralized algorithm is not applicable in a realistic setting, but it provides a measure of the best possible performance. From this point forward, we will only study the distributed algorithm.

We next conduct a series of Monte Carlo simulations with the distributed algorithm and compute means and variances of the resulting localization errors. One goal of this study is to verify that the accuracy improvements observed above are not a chance occurrence. Another goal is to examine how the localization accuracy varies with the number of robots in the team.

In each of the sample runs of the Monte Carlo simulation, the robots traveled the same paths - those shown in Figure 3(a). Measurement noise was introduced in the same manner as in the previous simulations, and varied randomly from run to run. Neighbor relations were again determined as described earlier, and kept the same from run to run to preclude that from being an additional source of randomness. Simulations for robot teams of size $1, 2, 3, 4$

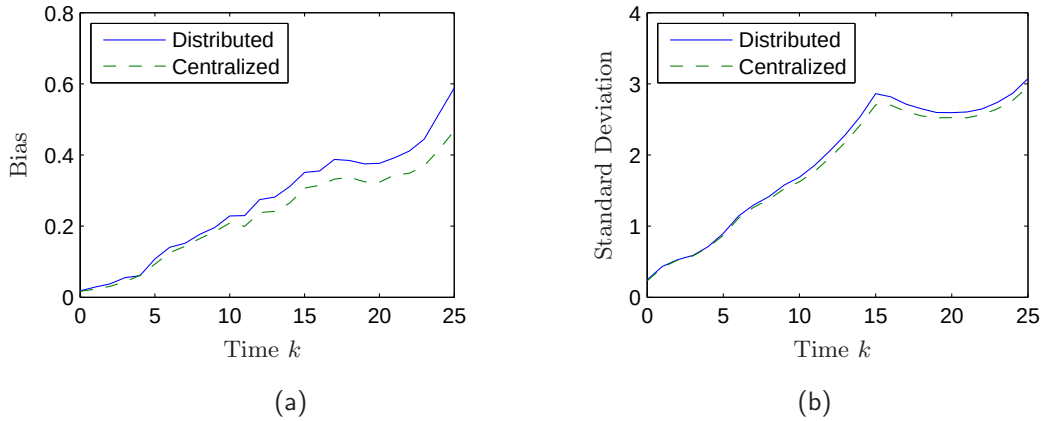


Figure 4: The (a) bias and (b) standard deviation in position error for a robot acting in a group of 5 localizing using either the centralized or distributed algorithm. This estimate was obtained using a 100-iteration Monte-Carlo experiment.

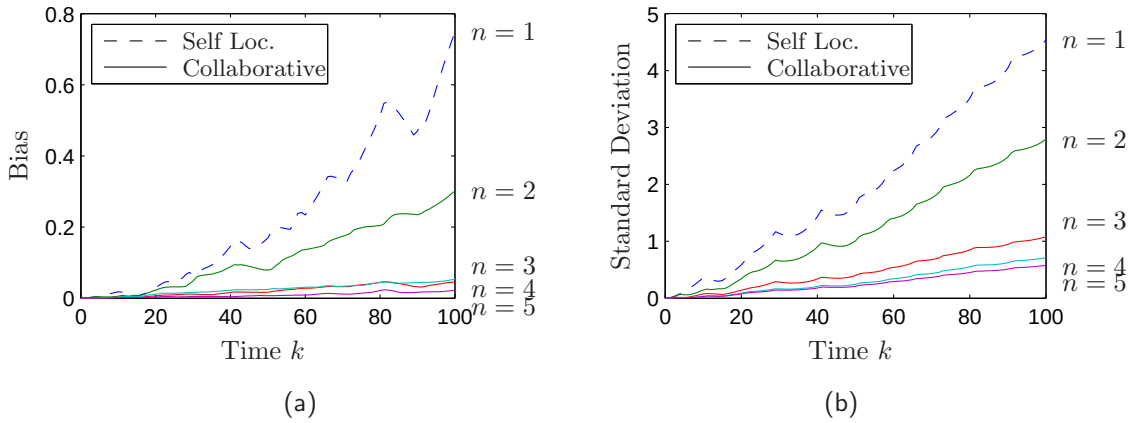


Figure 5: The (a) bias and (b) standard deviation in the position estimation error for robot 1 utilizing the distributed algorithm. The curves correspond to the number of robots in the group: 1, 2, 3, 4, or 5.

and 5 were carried out. In each case $1,000$ iterations were performed. When only one robot is present in the team, collaborative localization is equivalent to self-localization without the aid of any inter-robot relative pose measurement. As the number of robots in the team increases, the number of neighbors for a robot at any given time will tend to increase and so greater improvement in localization accuracy is expected.

The bias and standard deviation in the position estimation error $\mathbf{e}_i(k)$ for robot 1 ($i = 1$) are shown in Figure 5. Both bias and standard deviation show significant improvement with distributed collaborative localization over self-localization. This is evident even for a team of only two robots. As the number of robots in the team increases, the localization error of robot 1 decreases. The improvement in accuracy however, shows a diminishing return with increasing team size.

6.2 Effect of measurement type

We now perform simulations for the case when inter-robot relative measurements are of various types (pose, orientation, position, bearing, and distance). To examine the effect that measurement type has on localization accuracy, in each experiment, we let all inter-robot relative measurements be of the same type. Monte Carlo simulations were conducted in which we again consider 5 robots traveling in the zig-zag paths shown in Figure 3(a). Errors in the pose, orientation and position measurements are introduced as described in section 6.1. Noise in the bearing measurements is induced by rotating the true bearing through the application of a unit quaternion generated from an i.i.d. Von Mises-Fisher distribution. The noise in the distance measurements is normally distributed. The bias and standard deviation of position error for robot 1 are shown in Figure 6.

We see from the plots that improvement over self-localization occurs for all measurement types, with the exception of distance measurements. While distance measurements improve the standard deviation of the position estimates, they have little or no effect on the bias. That distance has little effect on the accuracy is consistent with the conclusions in the study (Martinelli et al., 2005) for 2-D localization. The fact that bearing measurements lead to higher improvement over distance or position measurements was also observed in (Martinelli et al., 2005) for the 2-D case. While the conclusions in (Martinelli et al., 2005) were based on single simulations, ours are based on Monte Carlo simulations.

Trade-offs between cost of sensors and the resulting benefit in localization can be analyzed from these empirically observed trends. Though full pose provides the most benefit to localization accuracy, it is clear that any of the considered inter-robot measurement types can be used to improve localization accuracy over dead-reckoning. In particular, after relative pose, relative position seems to be the most valuable types of inter-robot measurements, leading to significant reduction in both bias and variance over dead reckoning. This means that the cost of having sensors capable of measuring relative position (stereo vision, laser range finder, or monocular camera based bearing sensor along with a RF-based distance measurement) may very well be justified by the localization accuracy they lead to. On the other hand, it is also apparent that the improvement due to inter-robot bearing measurements is quite comparable to that due to inter-robot relative position measurements. Yet, only a single camera is necessary to measure the bearing, where as (in general) stereo vision is necessary to measure the full relative position, which is still quite prone to large errors unless large baseline stereo is used. Thus, given cost, payload, and reliability constraints, monocular cameras might be a better choice than stereo vision. These conclusions come with the caveat that they have been drawn from one set of Monte Carlo simulations; more extensive studies are needed to establish how general these trends are.

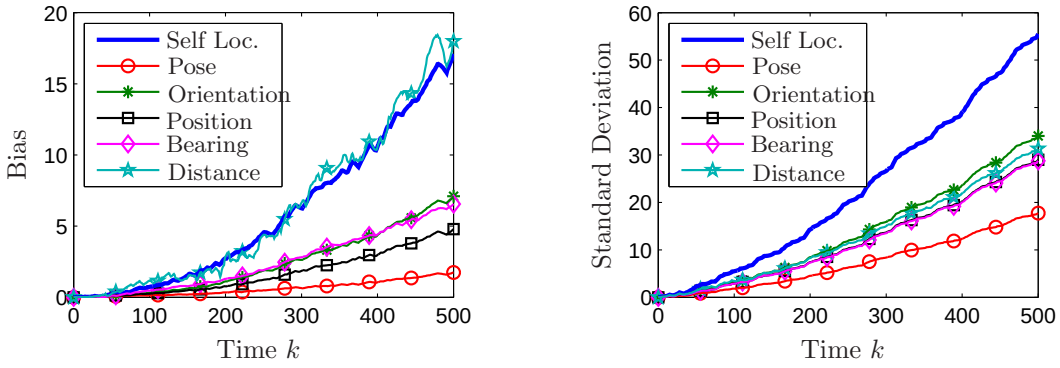


Figure 6: Simulation: The (a) bias and (b) standard deviation in position error for robot 1 when the distributed algorithm is applied to a group of 5 robots utilizing inter-robot relative measurements. The labels "Pose", "Orientation", "Position", "Bearing" and "Distance" indicate the group of robots used inter-robot measurements of the respective types to perform collaborative location using the distributed algorithm.

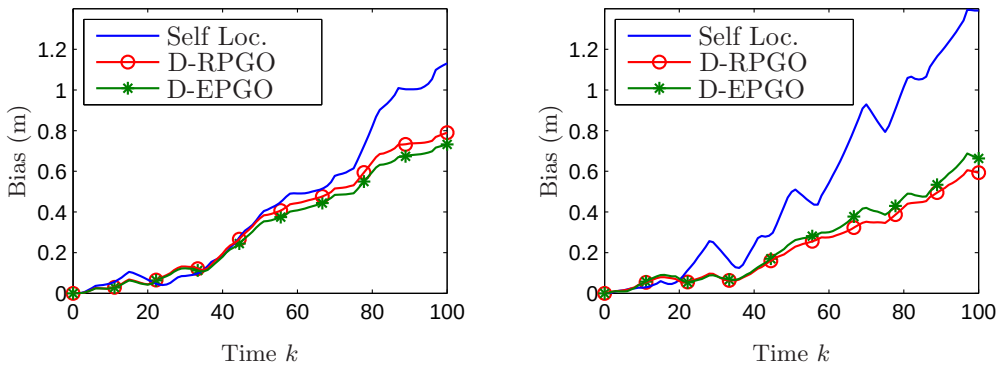
6.3 Comparison with alternate methods of distributed collaborative localization

We now present simulations that provide some insight into how the D-RPGO algorithm performs when compared with two state-of-the-art collaborative localization algorithms. All inter-robot relative measurements are of the relative pose for these simulations.

The first alternative we consider is the standard pose graph implementation presented in Section 5.1, referred to as the EPGO algorithm. To maintain comparability, we provide the same local measurement graph to both the D-RPGO algorithm as well as a EPGO algorithm, which we call the D-EPGO algorithm.

A group of 5 robots are simulated to move along the 3-D path described above. Error in the pose measurements were induced as in simulations in Section 6.2. Simulations were performed varying the concentration parameter K in the Von Mises-Fisher distribution from which the noisy rotations (quaternions) used to corrupt the inter-robot orientation measurements are drawn. These simulations show that, when K is very large, that is, the variance is very low, D-EPGO does very well, even outperforming the D-RPGO algorithm. However, when K is small, that is, the noise variance is large, D-RPGO outperforms D-EPGO. Due to a lack of space, only the results for $K = 10,000$ and $K = 100$ are shown; see Figure 7. It is clear that the proposed D-RPGO algorithm outperforms the D-EPGO algorithm when $K = 100$ (more noise), while the D-EPGO algorithm outperforms the D-RPGO algorithm when $K = 10,000$ (less noise). Though only the bias is recored in Figure 7, the standard deviation displayed a similar trend. We omit the corresponding plots due to a lack of space.

Next, we consider the IEKF algorithm presented in Section 5.2. A pair of robots is simulated traveling along distinct sinusoidal paths in 3-D space. Measurements are generated as described earlier. The trends observed from extensive

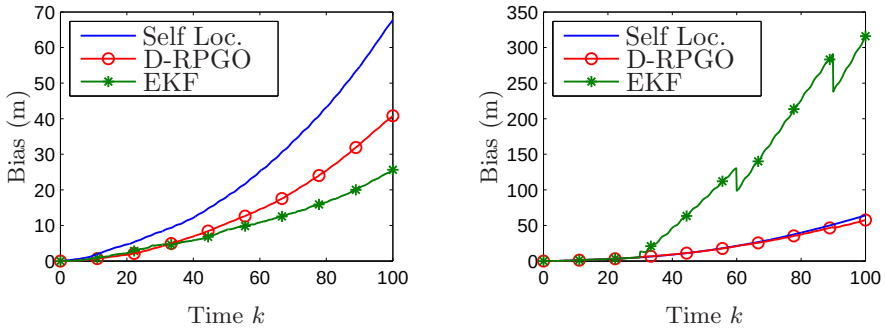


(a) Lose noise measurements (b) High noise measurements

Figure 7: Simulation: comparison between D-RPGO and D-EPGO algorithms when (a) inter-robot orientation measurements are less noisy ($k = 10,000$) and (b) inter-robot orientation measurements are more noisy ($k = 100$) where k is the concentration parameter for the Von-Mises Fisher distribution from which the noisy measurements are drawn. The bias in position estimation error of robot 1 (in a group of 5 robots utilizing noisy inter-robot relative pose measurements), computed with both algorithms, averaging over the same graph, is shown.

simulations can be summarized as follows. When the time interval between successive inter-robot measurements, call it ΔT, is small, the IEKF performs as well, or better than, the D-RPGO algorithm. However, when the time between measurements is large, the D-RPGO algorithm provides significantly better estimates of the robots' poses compared to the IEKF algorithm. Figure 8 provides numerical results for the case of a large ΔT (30 seconds in this example), when IEKF performs poorly, and small ΔT (0.1 seconds), when the IEKF performs well. How small ΔT has to be for IEKF to perform well depends on many factors, including the motion of the robots, noise in the measurements, etc. For the parameters used in the simulations mentioned above, ΔT has to be smaller than 0.1 sec for the IEKF to perform as well as the D-RPGO algorithm.

We believe the reason for this behavior of the IEKF is the error introduced by the linearization involved in covariance propagation. The linearized state equations rely on the assumption that the angle between the true and estimated orientation is very small. When the time interval between inter-robot measurements is sufficiently small, this approximation holds. In that case the error in the covariance matrix due to linearization is small enough that it does not out weigh the added benefit of using covariance information. However, the small angle approximation is violated for large time intervals, leading to quite poor covariance estimates, which in turn lead to poor pose estimates.



(a) $\Delta T = 0.1$ sec (b) $\Delta T = 30$ sec

Figure 8: Simulation: comparison between D-RPGO and IEKF algorithms with two different sampling periods for inter-robot measurements. The bias in position estimation error of robot 1 (in a group of 2 robots utilizing noisy inter-robot relative pose measurements), computed with both algorithms.

7 Experimental results

In this section we present results for experiments conducted with two Pioneer P3-DX robots; they are shown in Figure 9. Each robot was equipped with a calibrated monocular Prosillica EC 1020 camera and wheel odometers. Measurements from these sensors were fused to obtain the noisy inter-time relative pose measurements. Each robot is additionally equipped with a target allowing the on-bard cameras to measure the inter-robot relative pose by exploiting the known geometry of each target. The true pose of each robot was determined using an overhead camera capable of tracking each robot's target. The sensor suite was polled every 0.2 seconds with the noisy inter-robot relative pose measurements available at most, but not all, times.

All robots moved in straight lines with their paths approximately parallel. Six different pose estimates of the robots were obtained at each time. The first was a dead-reckoning estimate, obtained from the inter-time relative pose measurements alone. The remaining 5 estimates were obtained by using the distributed collaborative localization algorithm with the inter-robot noisy relative measurements being of full pose, orientation only, position only, bearing only, or distance only, respectively. These measurements were obtained by projecting the relative pose measurements. The resulting global position estimates, along with the true positions, for robot 1 are reported in Figure 10. Simulations presented in section 6 indicate that we should see a significant improvement in localization accuracy even in this small team and the experimental results are consistent with that conclusion. As in the simulations, distinct improvement in localization accuracy is seen when collaborative localization is performed irrespective of the type of inter-robot measurement though the improvement varied depending on the measurement type.

To show that the improvements seen in the previous experiments are not just a random occurrence, the previous experiment was repeated 24 times to produce an empirical estimate of the bias and standard deviation in position

Figure 9: Two Pioneer P3-DX robots equipped with cameras and targets. Robot 1 is shown on the left, while robot 2 is on the right.

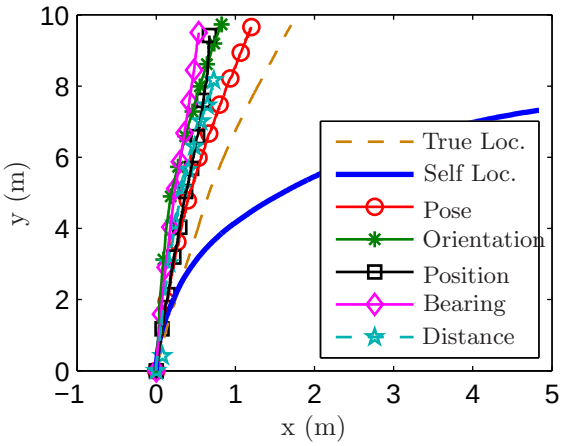


Figure 10: Experimental: A plot of the location of robot 1 in the overhead camera frame of reference when both robots move in a straight line. The true path (determined using the overhead camera), estimated path using self localization, and estimated path using the distributed collaborative localization algorithm are all reported. The various curves correspond to the type of inter-robot relative measurement used.

estimation error (by taking appropriate averages) for each type of inter-robot measurement. The results are reported in Figure 11. The experimental results again show that all measurement types lead to an increase in localization accuracy, with relative pose measurements leading to the maximum improvement, as expected. This is also the same trend that was observed in the simulations. In contrast to the simulations, here we see that distance measurements do lead to a non-negligible improvement in the bias of localization accuracy. As in the simulations, we see that both bearing and position measurements lead to similar improvement in the bias. However, in contrast to the simulations, in the experiments orientation measurements seem to improve the bias more than bearing or position measurements. The trend of the standard deviation improvement with orientation, position and orientation measurements are similar in both simulation and experiments. The most significant difference between the trends observed in simulations and experiments are in the standard deviation improvement between pose, orientation, and position measurements: they

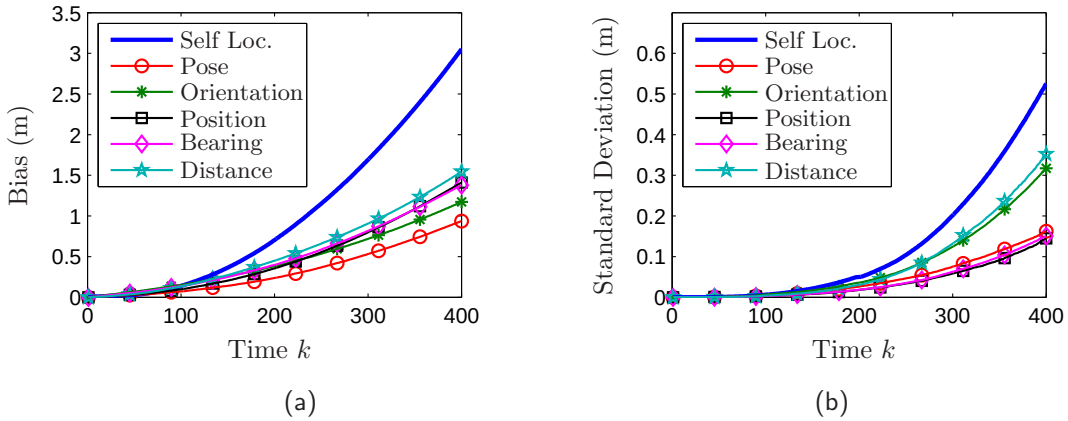


Figure 11: Experimental: The (a) bias and (b) standard deviation in position error for robot 1 with collaborative localization when both robots move in a straight line. The bias and standard deviation were empirically estimated by averaging over 24 repeated trials.

are much closer in the experiments than in the simulations. In fact, the standard deviation with position or orientation measurements seem to be a little smaller than that with pose measurements. We believe this is an artifact of the small number of experimental samples averaged to estimate the bias and standard deviation empirically, which limits the accuracy of the estimates.

In short, the experiments verify that the proposed algorithm for distributed collaborative localization leads to statistically significant improvement in localization accuracy even with a small number of robots. Several trends seen in the experiments about the relative merits of the different types of measurements are consistent with those in the simulations. However, there are a few noticeable differences as well. Though the exact cause of these differences is not clear, one should note that the simulations and experiments differ in a number of ways. The biggest difference is that the bias and standard deviations estimated from the experiments are likely to have higher error than in the simulations due to the small number of experimental samples, which in turn is due to the difficulty associated with conducting repeated experiments. The other significant difference comes from the measurement noise distributions. In the simulations, measurement noise was drawn from distributions that were somewhat arbitrarily chosen, while the noise distributions in the experiments are unknown. The third potential source of difference is the paths of the robots. In (Knuth and Barooah, 2012a), which examined the growth of dead-reckoning error, we showed that the path a robot traverses plays a crucial role in the bias and standard deviation of the localization error. In particular, the bias grows without bound if the robot moves in a straight line, but stays uniformly bounded by a constant if the robot stays inside a bounded region. Similar effects are likely in collaborative localization. Therefore, the difference between the paths used in the simulation and experiments might be another source of the difference in localization accuracy observed.

8 Summary

In this paper we introduce a novel distributed algorithm (D-RPGO) for estimating the 3-D pose of multiple robots when noisy inter-robot measurements of various types (relative pose, orientation, position, bearing, or distance) between pairs of robots are intermittently available. The distributed algorithm is inspired by a centralized algorithm for solving a least-squares type problem. The cost function is chosen to measure how well the estimates explain the relative measurements. A gradient-descent in a product Riemannian manifold is used to solve the optimization problem.

The proposed method is close in spirit to the pose graph optimization problems commonly uncounted in mapping and localization. The primary distinction is that our cost function is defined on a Riemannian manifold and is minimized on that surface without converting the problem to a vector space optimization problem. The estimator is thus naturally independent of the choice of parameterization of $SO(3)$ utilized to carry out the numerical computation. While this distinction may seem minor, results indicate it leads to performance benefit in some cases. In particular, the D-RPGO algorithm outperforms a distributed version of the traditional pose graph optimization (that uses a specific Euclidean parameterization of the robot orientations) in terms of accuracy when the noise in the inter-robot measurements is large.

It was found that the computation time of the D-RPGO algorithm is comparable to that of the D-EPGO algorithm in the simulations we performed. However, there are many ways to speed up the computations involved in Euclidean optimization that are not currently available for Riemannian optimizations. We suspect when applied to large graphs, Riemannian optimization will be slower than Euclidean optimization. Fortunately the measurement graphs that appear in distributed computation are small, since their size scales with the number of neighbors of a robot.

The IEKF has been a popular tool in past work on collaborative localization as well as mapping. Simulation comparison of the proposed D-RPGO algorithm with the IEKF showed that the IEKF performs poorly compared to the proposed D-RPGO algorithm unless the time interval between successive inter-robot relative measurements is quite small. This has important implications for many practical applications, since it is more likely that inter-robot measurements will be available infrequently. However, in those special cases when inter-robot measurements arrive frequently, the IEKF performs better then the D-RPGOalgorithm due to the consideration of the estimation covariance. One such scenario is the one considered in (Melnyk et al., 2012), where robots observe common feature points on the ground. In such a scenario the IEKF maybe preferable over the proposed method.

A limitation of the proposed method over the IEKF (and other filtering methods) is that the latter also provides a covariance estimate while the proposed method does not. A method to estimate the covariance, or some measure of estimation error, is an important future task. Although simulations reported here provide an indication of when the proposed method performs over existing state-of-the art methods, a more thorough study needs to be conducted to obtain a better understanding of the relative merits of the proposed method over alternatives. Future studies will also

address issues such as identification of the neighboring robots, and rejection of outliers in the relative measurements.

9 Acknowledgment

The authors would like to thank the University of Florida High Performance Computing Center for their support in carrying out the simulations presented in this paper.

10 Funding

This is work is partially supported by ARO grant DAAD19-03-D-0004

References

Absil, P., Mahony, R., and Sepulchre, R. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, 2008.

Andersson, L. A. A. and Nygards, J. C-SAM : Multi-robot SLAM using square root information smoothing. In *IEEE International Conference on Robotics and Automation*, Pasadena, California, USA, May 2008.

Aragues, R., Carlone, L., Calafiore, G., and Sagues, C. Multi-agent localization from noisy relative pose measurements. In *IEEE International Conference on Robotics and Automation*, Shanghai, China, Mar 2011.

Barooah, P., Russell, W. J., and Hespanha, J. P. Approximate distributed kalman filtering for cooperative multi-agent localization. In *International conference in Distributed Computing in Sensor Systems (DCOSS)*, Santa Barbara, CA, June 2010.

Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge Univ Pr, 2004.

Breckenridge, W. G. Quaternions-proposed standard conventions. Technical report, JPL, INTEROFFICE MEMORANDUM IOM 343-79-1199, 1999.

Caglioti, V., Citterio, A., and Fossati, A. Cooperative, distributed localization in multi-robot systems: a minimum-entropy approach. In *IEEE Workshop on Distributed Intelligent Systems*, Czech Republic, June 2006.

Fox, D., Burgard, W., Kruppa, H., and Thrun, S. A probabilistic approach to collaborative multi-robot localization. *Autonomous robots*, 8(3):325–344, 2000.

Gallot, S., Hulin, D., and LaFontaine, J. *Riemannian Geometry*. Springer, 3rd edition, 2004. ISBN 3540204938.

Grisetti, G., Grzonka, S., Stachniss, C., Pfaff, P., and Burgard, W. Efficient Estimation of Accurate Maximum Likelihood Maps in 3D. *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3472–3478, 2007.

Grove, K., Karcher, H., and Ruh, E. A. Jacobi fields and finsler metrics on compact lie groups with an application to differentiable pinching problems. *Mathematische Annalen*, 211:7–21, 1974.

Howard, A. Multi-robot Simultaneous Localization and Mapping using Particle Filters. *The International Journal of Robotics Research*, pages 1–11, Aug. 2011.

Howard, A., Matark, M., and Sukhatme, G. Localization for mobile robot teams using maximum likelihood estimation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, September 2002.

Indelman, V., Gurfil, P., Rivlin, E., and Rotstein, H. Distributed vision-aided cooperative localization and navigation based on three-view geometry. In *IEEE Aerospace Conference*, Big Sky, Montana, USA, March 2011.

Kim, B., Kaess, M., Fletcher, L., Leonard, J., Bachrach, A., Roy, N., and Teller, S. Multiple relative pose graphs for robust cooperative mapping. In *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, May 2010.

Knuth, J. and Barooah, P. Distributed collaborative localization of multiple vehicles from relative pose measurements. In *47th Annual Allerton Conference on Communication, Control, and Computing*, pages 314 –321, Allerton, IL, October 2009. doi: 10.1109/ALLERTON.2009.5394785.

Knuth, J. and Barooah, P. Error growth in position estimation from noisy relative pose measurements. *Robotics and Autonomous Systems*, 2012a. doi: 10.1016/j.robot.2012.11.001. first published online in Dec.

Knuth, J. and Barooah, P. Collaborative 3d localization of robots from relative pose measurements using gradient descent on manifolds. In *IEEE International Conference on Robotics and Automation*, St. Paul, Minnesota, May 2012b.

Kummerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. $g^2$0: a general framework for graph optimization. In *IEEE International Conference on Robotics and Automation*, pages 3607–3613, 2011a.

Kummerle, R., Grisetti, G., Strasdat, H., Konolige, K., and Burgard, W. g2o: A General Framework for Graph Optimization. *ICRA 2011*, pages 1–7, Feb. 2011b.

Leung, K. Y., Barfoot, T., and Liu, H. Decentralized localization of sparsely-communicating robot networks: A centralized-equivalent approach. *IEEE Transactions on Robotics*, 26(1):62 –77, 2010a.

Leung, K. Y. K., Barfoot, T. D., and Liu, H. H. T. Decentralized Localization of Sparsely-Communicating Robot Networks: A Centralized-Equivalent Approach. *IEEE TRANSACTIONS ON ROBOTICS*, 26(1):62–77, Feb. 2010b.

Long, A., Wolfe, K., Mashner, M., and Chirikjian, G. The banana distribution is gaussian: A localization study with exponential coordinates. In *Robotics: science and systems*, 2012.

Ma, Y., Košecká, J., and Sastry, S. Optimization criteria and geometric algorithms for motion and structure estimation. *International Journal of Computer Vision*, 44(3):219–249, 2001.

Mardia, K. V. and Jupp, P. E. *Directional Statistics*. Wiley series in probability and statistics. Wiley, 2000.

Martinelli, A., Pont, F., and Siegwart, R. Multi-robot localization using relative observations. In *IEEE International Conference on Robotics and Automation*, Barcelona, Spain, April 2005.

Melnyk, I. V., Hesch, J. A., and Roumeliotis, S. I. Cooperative Vision-aided Inertial Navigation Using Overlapping Views. *IEEE International conference on Robotics and Automation*, pages 1–8, 2012.

Nerurkar, E., Roumeliotis, S., and Martinelli, A. Distributed maximum a posteriori estimation for multi-robot cooperative localization. In *IEEE International Conference on Robotics and Automation*, pages 1402–1409, Kobe, Japan, May 2009.

Ni, K., Steedly, D., and Dellaert, F. Tectonic SAM: Exact, out-of-core, submap-based SLAM. In *IEEE International Conference on Robotics and Automation*, Roma, Italy, May 2007.

Olson, C. F., Matthies, L. H., Schoppers, M., and Maimone, M. W. Rover navigation using stereo ego-motion. *Robotics and Autonomous Systems*, 43(4):215–229, June 2003.

Panzieri, S., Pascucci, F., and Setola, R. 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2816–2821. IEEE. doi: 10.1109/IROS.2006.282065.

Rekleitis, I., Dudek, G., and Milios, E. Multi-robot cooperative localization: a study of trade-offs between efficiency and accuracy. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, EPFL, Switzerland, September 2002.

Roumeliotis, S. and Bekey, G. Distributed multirobot localization. *IEEE Transactions on Robotics and Automation*, 18(5):781 – 795, oct 2002.

Sanderson, C. A distributed algorithm for cooperative navigation among multiple mobile robots. *Advanced Robotics*, 12(4):335 – 349, 1998.

Sibley, G., Mei, C., Reid, I., and Newman, P. Adaptive Relative Bundle Adjustment. *Robotics Science and Systems*, pages 1–8, June 2009.

Tron, R. and Vidal, R. Distributed image-based 3-d localization of camera sensor networks. In *IEEE Conference on Decision and Control*, Shanghai, China, December 2009.

Wolfe, K. C., Mashner, M., and Chirikjian, G. S. Bayesian Fusion on Lie Groups. *Journal of Algebraic Statistics*, 2 (1):75–97, 2011.

Yershova, A., Jain, S., LaValle, S., and Mitchell, J. Generating uniform incremental grids on SO(3) using the hopf fibration. *The International Journal of Robotics Research*, 29(7), 2010.

Appendix

A Riemannian Manifolds

In this section we provide greater detail on the building blocks of Algorithm 1, including Riemannian manifolds. A full introduction to the study of Riemannian geometry is outside the scope of this paper; the interested reader is referred to (Gallot et al., 2004; Absil et al., 2008).

A *manifold* M of dimension d is a topological space that can locally be represented by $\mathbb{R}^d$. The set of all linear operators on the vector space $\mathbb{R}^3$ is denoted by $\mathbb{L}(\mathbb{R}^3)$. Elements of $\mathbb{L}(\mathbb{R}^3)$ can be represented by 3×3 matrices. We can then describe the manifold of 3-D rotations, denoted $SO(3)$, by the set $\left\{R \in \mathbb{L}(\mathbb{R}^3) : R^T R = id, det(R) = 1\right\}$. The symbol id denotes the identity operator. Given a point $p \in SO(3)$, the tangent space of $SO(3)$ at p, denoted $T_pSO(3)$ is given by

$$T_pSO(3) = \left\{p\hat{v} : \hat{v} \in \mathbb{L}(\mathbb{R}^3), \hat{v}^T = -\hat{v}\right\}.$$

A *Riemannian metric* on a manifold M is given by defining $\forall p \in M$ $g_p : T_pM \times T_pM \to \mathbb{R}$ such that

- $\forall p \in M$, g_p is an inner product
- given two vector fields X and Y on M, the map $g : M \to \mathbb{R}$, $p \mapsto g_p(X_p, Y_p)$ is smooth.

The Riemannian metric g gives rise to an inner-product norm on the tangent space T_pM for each $p \in M$. That is, for $\xi \in T_pM$, $\|\xi\|^2 = g_p(\xi, \xi)$. A *Riemannian manifold* is a manifold equipped with a Riemannian metric. The first Riemannian manifold we consider is $(SO(3), g)$ were g is the Riemannian metric given by

$$g_p(A, B) = \frac{1}{2} \operatorname{Tr}\left(A^T B\right) \tag{16}$$

for all $p \in SO(3)$, $A, B \in T_pSO(3)$. From this point on, when we refer to $SO(3)$ we mean this Riemannian manifold. To further simplify the notation, when the argument $p \in SO(3)$ is clear, we will denote $g_p(X_p, Y_p)$ by $g(X_p, Y_p)$.

For a Riemannian manifold, the analog to a straight line in Euclidean space is given by a *geodesic*. For two points p and q on a manifold M, a geodesic from p to q, denoted γ_{pq}, is a "shortest" path from p to q. More precisely, if we consider the set of all parameterized paths from p to q, given by $\Gamma = \{\gamma : [0,1] \to M : \gamma(0) = p, \gamma(1) = q\}$ then a geodesic γ_{pq} is given by a path that minimizes

$$\int_0^1 \|\gamma'(t)\| dt$$

over all paths in Γ, and $\gamma'(t)$ is the derivative of $\gamma(t)$ with respect to t, which is defined by the following relationship:

$$\gamma'(t_0) f = \frac{d(f(\gamma(t))}{dt}|_{t_0}$$

for all $f : M \to \mathbb{R}$. Note that the derivative $\gamma'(0)$ is an element of the tangent space $T_{\gamma(0)}M$. For the manifold $SO(3)$, the geodesic from p to q ($\in SO(3)$) is given by

$$\gamma_{pq}(t) = p \exp(tv), \ t \in [0,1]$$

where $v = \log(p^{-1}q)$, the map $\exp : \mathbb{L}(\mathbb{R}^3) \to \mathbb{L}(\mathbb{R}^3)$ is given by $\exp(X) = \sum_{k=0}^{\infty} \frac{X^k}{k!}$ and log is the inverse map of exp[1].

For an arbitrary Riemannian manifold, the Riemannian metric defines a corresponding distance function on the manifold as follow. For a Riemannian manifold M, the distance between two points $p, q \in M$ is given by

$$d_M(p,q) = \int_0^1 \|\gamma_{pq}(t)\| dt$$

In particular, for the manifold $SO(3)$,

$$d_{SO(3)}(p,q) = \sqrt{-\frac{1}{2} \operatorname{Tr}\left(\log^2(p^T q)\right)}.$$

The subscript $SO(3)$ on the distance function will be omitted when the arguments make it clear what manifold the distance refers to. The second manifold we consider is $(\mathbb{R}^3, < \cdot, \cdot >)$, standard Euclidean 3-space with the Riemannian metric given by the inner product on $\mathbb{R}^3$. Given points $p, q \in \mathbb{R}^3$ the (unique) geodesic connecting p and q is the straight line from p to q and $d(p,q) = \|p - q\| := < p - q, p - q >^{1/2}$, the standard Euclidean inner product norm.

In addition to the distance function, the Riemmanian metric also specifies a parallel transport function. Let M be an arbitrary Riemmanian manifold. For $p \in M$ and $\xi \in T_pM$, the *parallel transport* function, denoted $\exp_p$, finds a new point $q \in M$ given by moving along a geodesic beginning at p and with initial velocity ξ. More precisely,

$$q = \exp_p(\xi) = \gamma(1)$$

[1] This use of $\log(p)$ for $p \in SO(3)$ is only valid on the region where exp is a diffeomorphism, which it is at least on the open ball about $I \in SO(3)$ of radius π (Grove et al., 1974, Proposition 1.4).

where γ is a parameterized geodesic with $\gamma(0) = p$ and ξ being the tangent to $\gamma(t)$ at $t = 0$. In the case of $SO(3)$, $\exp_p(\xi) = p\exp(p^T\xi)$.

Finally, we consider the gradient for real value functions defined on the manifold. Given a Riemannian manifold M and a scalar, real-valued function $f : M \to \mathbb{R}$, the gradient of f at $p \in M$, denoted by $grad\ f(p)$, can be defined by the following relation:

$$g_p(grad\, f(p), \xi_p) = (f \circ \gamma)'(0) \quad \forall p \in M. \tag{17}$$

where $\gamma : [0, 1] \to M$ is any curve with $\gamma(0) = p$, and $(f \circ \gamma)'(0) := \frac{d(f\circ\gamma)(t)}{dt}|_0$. For an example of a gradient calculation that will be useful later, consider the following function

$$f_1 : SO(3) \to \mathbb{R},\ p \mapsto \frac{1}{2}d^2(p, q) \tag{18}$$

for some fixed $q \in SO(3)$. For an arbitrary $p \in SO(3)$, we consider the geodesic $\gamma_{pq}(t) = p\exp(tv)$ where $v = log(p^{-1}q)$. It can be shown that

$$(f_1 \circ \gamma)(t) = \frac{1}{2}(1 - t)^2 d^2(p, q)$$

and thus

$$(f_1 \circ \gamma)'(0) = -d^2(p, q) = \frac{1}{2}\,\mathrm{Tr}\left(v^2\right).$$

The corresponding tangent vector to $\gamma(t)$ at p is given by $\xi_p := \gamma'(0) = pv$. It follows from (16) that

$$g_p(grad\, f_1(p), \xi_p) = \frac{1}{2}\,\mathrm{Tr}\left((grad\, f_1(p))^T pv\right).$$

Applying (17) we have

$$grad\, f_1(p) = -pv = -p\log(p^{-1}q) \quad \forall p \in SO(3). \tag{19}$$

For a function $f : SO(3) \to \mathbb{R}$ the gradient at $\mathbf{R} \in SO(3)$ is given by (Ma et al., 2001):

$$grad\, f(\mathbf{R}) = f_{\mathbf{R}} - \mathbf{R}f_{\mathbf{R}}^T\mathbf{R} \tag{20}$$

where $f_{\mathbf{R}} \in L(\mathbb{R}^3)$ is the linear operator whose matrix representation (using the canonical basis vectors for $\mathbb{R}^3$) is given by: $(f_{\mathbf{R}})_{ij} = \frac{\partial f}{\partial \mathbf{R}_{ij}}$, where $\mathbf{R}_{ij}$ represents the (i, j)-th entry of the 3×3 matrix representation of $\mathbf{R}$. Gradient calculation for a function $f : \mathbb{R}^3 \to \mathbb{R}$ is straightforward.

Apart from $SO(3)$ and $\mathbb{R}^3$, the other Riemannian manifold that is useful in this study is $SE(3)$. Instead of $SE(3)$, however, we consider the equivalent manifold $SO(3) \times \mathbb{R}^3$. There exists a natural way to define the Riemannian metric on this manifold, based on the Riemannian metrics in $SO(3)$ and $\mathbb{R}^3$ so that we can deal with geodesics and gradients. This has to do with the fact that $SO(3) \times \mathbb{R}^3$ is a product manifold, which is the topic of the next section.

B Product Manifold

Let $\left\{(M_i, g^{(i)})_{i=1}^n\right\}$ be a set of n Riemannian manifolds. We define the *product Riemannian manifold* (M, g) as follows:

$$\begin{aligned} M &:= M_1 \times \cdots \times M_n \\ g_p(\xi, \zeta) &:= \sum_{i=1}^{n} g_{p_i}^{(i)}(\xi_i, \zeta_i) \end{aligned} \tag{21}$$

for all $p = (p_1, \ldots, p_n) \in M$ and all $\xi = (\xi_1, \ldots, \xi_n),\ \zeta = (\zeta_1, \ldots, \zeta_n) \in T_pM$. We first restrict our considerations to product Riemannian manifolds of the form $M = M_1 \times M_2$ (i.e. $n = 2$). The extension to any finite combination is straight forward. The following lemma will be useful in the sequel. The proof of the lemma can be found in (Ma et al., 2001).

Lemma 1 (Ma et al. (2001))**.** *Let $M = M_1 \times M_2$ be a product Riemannian manifold as defined in (21) and consider a smooth function $f : M \to \mathbb{R}$. Then for any parameterized path $\gamma = (\gamma_1, \gamma_2) : [0, 1] \to M$,*

$$\frac{d}{dt} f(\gamma(t))|_{t_0} = \frac{d}{dt} f(\gamma_1(t), \gamma_2(t_0))\Big|_{t_0} + \frac{d}{dt} f(\gamma_1(t_0), \gamma_2(t))\Big|_{t_0}$$

where $t_0 \in [0.1]$.

Theorem 1. *Let $M = M_1 \times M_2$ be a product Riemannian manifold, as defined in (21). Consider a smooth function $f : M \to \mathbb{R}$ and for all $(p, q) \in M$ ($p \in M_1$, $q \in M_2$), define*

$$f_1^q : M_1 \to \mathbb{R},\ \ a \mapsto f(a, q)$$

$$f_2^p : M_2 \to \mathbb{R},\ \ b \mapsto f(p, b).$$

Then for all $(p, q) \in M$

$$grad\, f(p, q) = (grad\, f_1^q(p), grad\, f_2^p(q))\,.$$

Proof. Fix $(p, q) \in M$ and consider an arbitrary tangent vector $\xi = (\xi_1, \xi_2) \in T_{(p,q)}M$. Choose a parameterized path $(\gamma_1, \gamma_2) = \gamma : [0, 1] \to M$ such that ξ_1 is tangent to $\gamma_1(t)$ at $p = \gamma_1(t_0)$ and ξ_2 is tangent to $\gamma_2(t)$ at $q = \gamma_2(t_0)$ (and thus ξ is tangent to $\gamma(t)$ at $(p, q) = \gamma(t_0)$). Using lemma 1, we have

$$(f \circ \gamma)'(t_0) = \frac{d}{dt}(f_1^q \circ \gamma_1)|_{t_0} + \frac{d}{dt}(f_2^p \circ \gamma_2)|_{t_0}.$$

From this relation and (17), we find that

$$g(grad\, f(p, q), \xi) = g^{(1)}(grad\, f_1^q(p), \xi_1) + g^{(2)}(grad\, f_2^p(q), \xi_2). \tag{22}$$

Let $grad\, f(p, q) = (A, B) \in T_{(p,q)M}$ for some unknown $A \in T_pM,\ B \in T_qM$. Using the definition of g given in (21), we can rewrite (22) as

$$g^{(1)}(A, \xi_1) + g^{(2)}(B, \xi_2) = g^{(1)}(grad\, f_1^q(p), \xi_1) + g^{(2)}(grad\, f_2^p(q), \xi_2).$$

Since this equality holds for all $\xi_1 \in T_pM$ and all $\xi_2 \in T_qM$, we have $A = grad\, f_1^q(p)$, $B = grad\, f_2^p(q)$, which completes the proof. □

As the number of manifolds increases, the notation $grad\, f_1^q(p)$ becomes more cumbersome. We will therefore simplify the notation by writing $grad\, f(p)$ (to mean $grad\, f_1^q(p)$) whenever the manifold the gradient found with respect to is made obvious by the argument p and q is clear from context.

The following corollary is a direct consequence of Theorem 1.

Corollary 1. *Given a set of n Riemannian manifolds $\{(M_i, g_i)\}_{i=1}^n$, if the product Riemannian metric on the product manifold (M, g), where $M = M_1 \times \cdots \times M_n$, is defined as*

$$g_p(X, Y) = g_1(X_1, Y_1) + \cdots + g_n(X_n, Y_n)$$

for $p = (p_1, \ldots, p_n) \in M$ and $X = (X_1, \ldots, X_n), Y \in T_pM = T_{p_1}M_1 \times \ldots T_{p_n}M_n$, then

$$grad\, f(p) = (grad\, f(p_1), \ldots, grad\, f(p_n)).$$

We next consider what geodesics look like on the product manifold.

Theorem 2. *If M is a product Riemannian manifold as defined in (21), then a geodesic between $p = (p_1, p_2)$ and $q = (q_1, q_2) \in M$ can be obtained by the product of geodesics between p_1 and q_1, and p_2 and q_2, on M_1 and M_2, respectively.*

Proof. Let $\gamma_{pq}, \gamma_{p_1q_1}, \gamma_{p_2q_2}$ denote the geodesics on M, M_1, and M_2 respectively. By definition, $\gamma_{pq} =: (\gamma^{(1)}, \gamma^{(2)})$ is the path that minimizes

$$\gamma_{pq} = \arg\min_{\gamma \in \Gamma_{pq}} \int_0^1 \|\gamma'(t)\|^2 dt$$

where Γ_{pq} denotes the set of all paths on M from p to q and, for $\gamma \in \Gamma_{pg}$, $\gamma(t)$ denotes a parameterization of path γ such that $\gamma(0) = p$ and $\gamma(1) = q$. Using the definition of the inner-product norm on the product manifold, we see that

$$\begin{aligned}
\gamma_{pq} &= \arg\min_{(\gamma^{(1)}, \gamma^{(2)}) \in \Gamma_{pq}} \int_0^1 (\|\gamma^{(1)'}(t)\|^2 + \|\gamma^{(2)'}(t)\|^2) dt \\
&= \left(\arg\min_{\gamma \in \Gamma_{p_1q_1}} \int_0^1 \|\gamma'(t)\|^2 dt,\ \arg\min_{\gamma \in \Gamma_{p_2q_2}} \int_0^1 \|\gamma'(t)\|^2 dt \right) \\
&= (\gamma_{p_1q_1}, \gamma_{p_2q_2}).
\end{aligned}$$

□

That parallel transport on M is given by parallel transport on each individual M_i immediately follows. Thus proving (6).

C Proof of Theorem 1

proof. The cost function (3) is a function from $\mathcal{M}$ to $\mathbb{R}$ where $\mathcal{M}$ is the product manifold

$$\mathcal{M} := \left(SO(3) \times \mathbb{R}^3\right)^n, \text{ where } n := |\mathcal{V}(k)|.$$

We define the Riemannian metric on this product manifold as follows. For $p = (\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_n, \mathbf{t}_n) \in \mathcal{M}$ and $X, Y \in T_p\mathcal{M}$, where $X \in T_p\mathcal{M}$ is expanded as $(X_{\mathbf{R}_1}, X_{\mathbf{t}_1}, \ldots, X_{\mathbf{R}_n}, X_{\mathbf{t}_n})$,

$$g_p(X, Y) := \sum_{i \in n} \left(g_{\mathbf{R}_i}(X_{\mathbf{R}_i}, Y_{\mathbf{R}_i}) + < X_{\mathbf{t}_i}, Y_{\mathbf{t}_i} > \right)$$

This meets the conditions of Corollary 1 and so for $p = (\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_n, \mathbf{t}_n) \in \mathcal{M}$, we have

$$grad\, f(p) = (grad\, f(\mathbf{R}_1), grad\, f(\mathbf{t}_1), \ldots, grad\, f(\mathbf{R}_h), grad\, f(\mathbf{t}_h)).$$

All that remains is to determine $grad\, f(\mathbf{R}_u)$ and $grad\, f(\mathbf{t}_u)$ for $u = 1, \ldots, n$. Using linearity of the gradient operator, we have

$$grad\ f(\mathbf{R}_u) = \sum_{e \in \mathcal{E}(k)} grad\ c_e(\mathbf{R}_u), \text{ and } grad\ f(\mathbf{t}_u) = \sum_{e \in \mathcal{E}(k)} grad\ c_e(\mathbf{t}_u).$$

First define $f_2 := \frac{1}{2} d^2(\mathbf{R}_u^T \mathbf{R}_v, \hat{\mathbf{R}}_{u\,v})$, $f_3 := \frac{1}{2} \|\hat{\mathbf{t}}_{u\,v} - \mathbf{R}_u^T(\mathbf{t}_v - \mathbf{t}_u)\|^2$, so that for $e \triangleright e(u, v)$, $c_e = f_2 + f_3$. For $h = u$ or $h = v$, we therefore have

$$grad\ c_e(\mathbf{R}_h) = grad\ f_2(\mathbf{R}_h) + grad\ f_3(\mathbf{R}_h) \tag{23}$$

$$grad\ c_e(\mathbf{t}_h) = 0 + grad\ f_3(\mathbf{t}_h). \tag{24}$$

The Riemannian distance $d(p, q)$ is bi-invariant, meaning that for arbitrary $R, \bar{R} \in SO(3)$, $d(p, q) = d(Rp\bar{R}, Rq\bar{R})$. Using bi-invariance, we obtain $d^2(\mathbf{R}_u^T \mathbf{R}_v, \hat{\mathbf{R}}_{u\,v}) = d^2(\mathbf{R}_v, \mathbf{R}_u \hat{\mathbf{R}}_{u\,v})$, which implies $grad\ f_2(\mathbf{R}_v) = grad\ f_1(\mathbf{R}_v | q)$ where $f_1 : SO(3) \to \mathbb{R}$ is given by (18) and $q = \mathbf{R}_u \hat{\mathbf{R}}_{u\,v}$. Applying formula (19), we obtain

$$grad\ f_2(\mathbf{R}_v) = -\mathbf{R}_v \log(\mathbf{R}_v^T \mathbf{R}_u \hat{\mathbf{R}}_{u\,v}).$$

The expression for $grad\ f_2(\mathbf{R}_u)$ is similarly obtained; by thinking of f_2 as a function of $\mathbf{R}_u$ and keeping $\mathbf{R}_v, \hat{\mathbf{R}}_{u\,v}$ fixed. These gradients are compactly represented as:

$$grad\, f_2(\mathbf{R}_h) = \begin{cases} -\mathbf{R}_h \log\left(\mathbf{R}_h^T\, \mathbf{R}_v\, \hat{\mathbf{R}}_{u\,v}^T\right) & \text{if } h = u \\ -\mathbf{R}_h \log\left(\mathbf{R}_h^T\, \mathbf{R}_u\, \hat{\mathbf{R}}_{u\,v}\right) & \text{if } h = v \\ 0 & \text{otherwise} \end{cases}$$

The gradient of the function $f_3 : SO(3) \to \mathbb{R}$ (thinking of f_3 only as a function of $\mathbf{R}_u$ while $\mathbf{t}_u, \mathbf{t}_v, \hat{\mathbf{t}}_{u\,v}$ are fixed) can be obtained using (20) with straightfrward but tedious calculations; which turn out to be $grad\, f_3(\mathbf{R}_u) = -(\mathbf{t}_v - \mathbf{t}_u)\hat{\mathbf{t}}_{u\,v}^T +$

$\mathbf{R}_u \hat{\mathbf{t}}_{u\,v} (\mathbf{t}_v - \mathbf{t}_u)^T \mathbf{R}_u$. The gradient of $f_3 : \mathbb{R}^3 \to \mathbb{R}$ is simply $(\frac{\partial f_3(x)}{\partial x})^T$, which is

$$grad\, f_3(\mathbf{t}_h) = \begin{cases} \big(\mathbf{t}_h + \mathbf{R}_h\, \hat{\mathbf{t}}_{u\,v} - \mathbf{t}_v\big) & \text{if } h = u \\ \big(\mathbf{t}_h - \mathbf{R}_u\, \hat{\mathbf{t}}_{u\,v} - \mathbf{t}_u\big) & \text{if } h = v \\ 0 & \text{o.w.} \end{cases}$$

These formulae completely specify the gradient of the edge cost, $grad\ c_e(\,\cdot\,)$ in (23). The expressions for $grad\ f(\mathbf{R}_u)$ and $grad\ f(\mathbf{t}_u)$ for $u = 1, \ldots, n$ that are provided in the theorem are obtained simply by adding the components of the gradients that are derived above. □

D Gradient of the Cost Function (3) for Heterogeneous Measurements

As in the proof of Proposition 1, the gradient of the function f in (3) at a point $p = (\mathbf{R}_1, \mathbf{t}_1, \ldots, \mathbf{R}_n, \mathbf{t}_n) \in \big(SO(3) \times \mathbb{R}^3\big)^n$ is given by

$$grad\, f(p) = \sum_{e \in \mathcal{E}(k)} grad\, c_e(p) =: (grad\, f(\mathbf{R}_1), \ldots, grad\, f(\mathbf{t}_n)) \tag{25}$$

where $grad\, g_e(p)$ is the gradient of the edge cost function for edge $e = (u, n)$. Finding the gradient of the cost function (3) then reduces to finding the gradients of the edge costs c_e (specified in (4)) for each edge $e \in \mathcal{E}(k)$.

It follows from Corollary 1 that the gradient of the cost function c_e in (4) for the edge $e = (u, v) \in \mathcal{E}(k)$ is

$$grad\ c_e(p) = \Big(grad\ c_e(\mathbf{R}_1), grad\ c_e(\mathbf{t}_1), \ldots, grad\ c_e(\mathbf{R}_n), grad\ c_e(\mathbf{t}_n)\Big)$$

For $\ell(k)(e) = \mathbf{T}$ (pose), the gradients have already been computed in the previous section, which can be compactly represented as $(e \triangleright (u, v))$

$$grad\, c_e(\mathbf{R}_h) = \begin{cases} -2\mathbf{R}_h \Big(\log(\mathbf{R}_h^T \mathbf{R}_v \hat{\mathbf{R}}_{u\,v}^T) \\ \quad + \mathbf{R}_h^T (\mathbf{t}_v - \mathbf{t}_u) \hat{\mathbf{t}}_{u\,v}^T - \hat{\mathbf{t}}_{u\,v} (\mathbf{t}_v - \mathbf{t}_u)^T \mathbf{R}_h \Big) & \text{if } h = u \\ -2\mathbf{R}_h \log(\mathbf{R}_h^T \mathbf{R}_u \hat{\mathbf{R}}_{u\,v}) & \text{if } h = v \\ 0 & \text{o.w.} \end{cases}$$

$$grad\ c_e(\mathbf{t}_h) = 2 I_{uv}(h)(\mathbf{t}_u + \mathbf{R}_u \hat{\mathbf{t}}_{u\,v} - \mathbf{t}_j)$$

where $I_{uv}(h) = 1$ if $h = u$, -1 if $h = v$ and 0 otherwise. If $\ell(k)(e) = \mathbf{R}$ (orientation) or $\ell(k)(e) = \mathbf{t}$ (position), we have

the following expressions for the gradient from the previous section. If $\ell(k)(e) = \mathbf{R}$,

$$grad\, c_e(\mathbf{R}_h) = \begin{cases} -2\mathbf{R}_h\Big(\log(\mathbf{R}_h^T \mathbf{R}_v \hat{\mathbf{R}}_{u\,v}^T) & \text{if } h = u \\ -2\mathbf{R}_h\Big(\log(\mathbf{R}_h^T \mathbf{R}_u \hat{\mathbf{R}}_{u\,v}) & \text{if } h = v \\ 0 & \text{otherwise} \end{cases}$$

$$grad\ c_e(\mathbf{t}_h) = 0.$$

and if $\ell(k)(e) = \mathbf{t}$, then

$$grad\, c_e(\mathbf{R}_h) = \begin{cases} -2\mathbf{R}_h\Big(\mathbf{R}_h^T(\mathbf{t}_v - \mathbf{t}_u)\hat{\mathbf{t}}_{u\,v}^T - \hat{\mathbf{t}}_{u\,v}(\mathbf{t}_v - \mathbf{t}_u)^T\mathbf{R}_h\Big) & \text{if } h = u \\ 0 & \text{otherwise} \end{cases}$$

$$grad\ c_e(\mathbf{t}_h) = 2I_{uv}(h)(\mathbf{t}_u + \mathbf{R}_u\hat{\mathbf{t}}_{u\,v} - \mathbf{t}_v).$$

If $\ell(k)(e) = \boldsymbol{\tau}$ (bearing) or $\ell(k)(e) = \delta$ (distance), the gradient $grad\ c_e(\mathbf{R}_h)$ can be computed by using formula (20). The gradient $grad\ c_e(\mathbf{t}_h)$ is obtained by differentiation. We obtain, for $\ell(k)(e) = \boldsymbol{\tau}$ (bearing)

$$grad\ c_e(\mathbf{R}_h) = \begin{cases} -2\mathbf{R}_h\Big(\mathbf{R}_h^T(\mathbf{t}_v - \mathbf{t}_h)\hat{\boldsymbol{\tau}}_{u\,v}^T\|\mathbf{t}_u - \mathbf{t}_v\| - \hat{\boldsymbol{\tau}}_{u\,v}\|\mathbf{t}_v - \mathbf{t}_u\|(\mathbf{t}_v - \mathbf{t}_u)^T\mathbf{R}_h\Big) & \text{if } h = u \\ 0 & \text{otherwise} \end{cases}$$

$$grad\ c_e(\mathbf{t}_h) = -4I_{uv}(h)[(\mathbf{t}_v - \mathbf{t}_u) - \|\mathbf{t}_v - \mathbf{t}_u\|\mathbf{R}_u\hat{\boldsymbol{\tau}}_{u\,v}]$$

and if $\ell(k)(e) = \delta$ (dist)

$$grad\, c_e(\mathbf{R}_h) = 0$$

$$grad\, c_e(\mathbf{t}_h) = -2I_{uv}(h)\frac{(\hat{\delta}_{u\,v} - \|\mathbf{t}_v - \mathbf{t}_u\|)}{\|\mathbf{t}_v - \mathbf{t}_u\|}(\mathbf{t}_v - \mathbf{t}_u).$$

The gradient $grad\ f(p)$ can now be computed by using these formulas.