Article

The International Journal of Robotics Research
2024, Vol. 43(14) 2245–2273

DOI: 10.1177/02783649241256970
journals.sagepub.com/home/ijr
Sage

Group-*k* consistent measurement set maximization via maximum clique over *k*-uniform hypergraphs for robust multi-robot map merging

Brendon Forsgren[1], Michael Kaess[2], Ram Vasudevan[3], Timothy W. McLain[1] and Joshua G. Mangelson[4]

Abstract

*This paper unifies the theory of consistent-set maximization for robust outlier detection in a simultaneous localization and mapping framework. We first describe the notion of pairwise consistency before discussing how a consistency graph can be formed by evaluating pairs of measurements for consistency. Finding the largest set of consistent measurements is transformed into an instance of the maximum clique problem and can be solved relatively quickly using existing maximum-clique solvers. We then generalize our algorithm to check consistency on a group-*k *basis by using a generalized notion of consistency and using generalized graphs. We also present modified maximum clique algorithms that function over generalized graphs to find the set of measurements that is internally group-*k *consistent. We address the exponential nature of group-*k *consistency and present methods that can substantially decrease the number of necessary checks performed when evaluating consistency. We extend our prior work to perform data association, and to multi-agent systems in both simulation and hardware, and provide a comparison with other state-of-the-art methods.*

Keywords

Localization, localization and mapping, mapping, multi-robot SLAM

Received 4 November 2023; Revised 5 March 2024; Accepted 16 April 2024
Senior Editor: Jose Luis Blanco
Associate Editor: Junyi Geng

1. Introduction

Multi-agent simultaneous localization and mapping (SLAM) refers to the problem of estimating a map of the environment by fusing the measurements collected by multiple robots as they navigate through that environment. For the estimated map to be accurate, both the local trajectories of the vehicles and the relative offsets (translation and orientation) between the trajectories need to be estimated.

In SLAM, the estimation problem is often modeled using a factor graph containing pose and landmark node variables, and factor nodes that encode the relationship between poses and landmarks. A special case of the SLAM problem, called pose graph SLAM, eliminates the landmark nodes and only estimates the vehicle trajectory. We often formulate the problem as the maximum likelihood estimation (MLE) of the time-discretized robot trajectory given odometric and loop-closure measurements as described by Cadena et al. (2016). Assuming independence and additive Gaussian noise in the measurement and process models, the problem becomes a nonlinear, weighted-least-squares problem that can be solved quickly using available solvers like those presented by Kaess et al. (2008), Kümmerle et al. (2011), and Agarwal et al. (2023).

In multi-agent SLAM, multiple vehicles are used to map the environment, resulting in increased scalability and efficiency in the mapping process. However, in addition to estimating the local map, the vehicles must also estimate their relative pose to accurately combine their maps. Generating inter-vehicle measurements is a process that is often susceptible to perceptual aliasing and can be inaccurate. Identifying poor inter-vehicle measurements is a

[1]Department of Mechanical Engineering, Brigham Young University, Provo, UT, USA
[2]Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA
[3]Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI, USA
[4]Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT, USA

Corresponding author:
Brendon Forsgren, Department of Mechanical Engineering, Brigham Young University, 350 Engineering Building, Provo, UT 84602-0002, USA.
Email: brendon5@byu.edu

challenging problem given the lack of a single odometry backbone and potentially no prior information on the initial configuration of the vehicles, as shown by Pfingsthorn and Birk (2016). In this paper, we first present a technique to identify poor inter-vehicle measurements for full-degree-of-freedom constraints between vehicles. We then generalize this technique to enable its use with low-degree-of-freedom measurements.

Rather than attempt to classify measurements as inliers and outliers, we find the largest consistent set of inter-robot relative-pose measurements. In our prior conference paper (Mangelson et al., 2018),[1] the problem is formulated as a combinatorial optimization problem that seeks to find the largest set of pairwise-consistent measurements. We then show that this problem can be transformed into an instance of the maximum-clique problem, that existing algorithms can be used to find the optimal solution for moderately sized problems, and that heuristic-based methods exist that often find the optimal solution for larger numbers of measurements. This proposed method is then evaluated on both simulated and real-world data showing that the technique outperforms existing robust SLAM algorithms in selecting consistent measurements and estimating the merged maps. These contributions are included in Section 4, Section 5, and Section 11.

Our second conference paper (Forsgren et al., 2022)[2] generalizes the concept of pairwise consistency to group-k consistency for scenarios, such as range-based SLAM, where pairwise consistency is insufficient to characterize the consistency of a set of measurements. We show that by using a generalized graph and modifying known maximum-clique algorithms to function over generalized graphs, we can robustly reject outliers in scenarios where pairwise consistency fails. The generalized method was evaluated on simulated data and showed that enforcing group-k consistency outperforms enforcing pairwise consistency. These results are discussed in Sections 6 to 8, 9.2, and 12.

This paper builds on our two prior conference publications and makes the following contributions:

1. We develop a framework that takes advantage of the hierarchical structure of consistency to decrease the number of consistency checks needed when building the generalized graph online. (Section 7)
2. We evaluate GkCM on hardware data recorded by an unmanned underwater vehicle in a range-only SLAM scenario and compare with other outlier-rejection algorithms (Section 12).
3. We propose a consistency function that can be used in vision-based multi-agent pose graph optimization problems. We verify this consistency function on both simulated and hardware data (Sections 9.1 and 13).
4. We compare our maximum clique algorithms over hypergraphs with other recently developed algorithms (Section 8, Section 12, and Section 13).
5. We release our implemented algorithms including a mechanism to evaluate consistency in sets of k measurements and a parallelized implementation of our maximum clique algorithms over generalized graphs (https://bitbucket.org/jmangelson/gkcm/src/master/).

The remainder of this paper is organized as follows. In Section 2, related work is discussed. In Section 3, the general formulation of the multi-robot pose graph SLAM problem is presented. Pairwise consistency maximization (PCM) is presented in Sections 4 and 5. The generalized algorithm, Group-k consistency maximization (GkCM), is presented in Sections 6 to 8. A brief discussion of our target applications and the consistency checks used to evaluate consistency are presented in Section 9. An evaluation of our maximum clique algorithms over generalized graphs is found in Section 10, followed by the evaluation of PCM in Section 11 and GkCM in Sections 12 and 13. Finally, in Section 14, we conclude.

2. Related work

The ability to identify and remove outlier measurements is important to many robotics and computer-vision applications. Given the sensitivity of nonlinear least-squares optimization to poor information, there has been a significant amount of effort dedicated toward developing methods to detect and remove outlier measurements from the optimization problem.

The random sample consensus (RANSAC) algorithm in Hartley and Zisserman (2003) is popular in the computer-vision community and detects outliers by fitting models to random subsets of the data and counting the number of inliers that belong to each model. The RANSAC algorithm struggles in scenarios where there is no unique model of the underlying data, such as in multi-agent SLAM, or when the outlier ratio is so large that no accurate model of the data can be found. Recent work by Sun (2021) has improved the RANSAC algorithm by adding a compatibility score between the random samples. The new technique, called RANSIC, shows improved performance in high-outlier regimes but will still struggle in multi-agent SLAM scenarios because no unique model exists that can classify the measurements as inliers and outliers. A technique called VODRAC introduced by Hu and Sun (2023) also improves on the RANSAC and RANSIC algorithms by using a two-point sampling strategy combined with a weight-based voting strategy that speeds up the consensus maximization and is robust in 99% outlier regimes.

Other approaches use the concept of M-estimation. These techniques attempt to detect the presence of outliers during the optimization process and use a robust cost function to decrease their influence in the weighted nonlinear least-squares problem. Sünderhauf and Protzel (2012) use switchable constraints, which introduces a switchable error factor that can be turned off if the residual error becomes too high. Dynamic covariance scaling (DCS), introduced by Agarwal et al. (2013), generalizes the switchable constraints method by increasing the covariance

matrix associated with measurements that have a high residual error, essentially smoothing the transition to turning a constraint off. Yang et al. (2020a) introduce graduated non-convexity (GNC), a technique that first solves a convex approximation of the original problem and iteratively solves less-convex approximations until the original problem is solved. The max-mixtures technique presented by Olson and Agarwal (2013) uses mixtures of Gaussians to model various data modes and can detect outliers in real time. Each of these methods was designed for a single-agent system and assumes a trusted odometry backbone is present. To apply these systems successfully in multi-agent scenarios would require a good initialization of the relative pose between agents which is not always available. Expectation maximization techniques are used by Dong et al. (2015) and Carlone et al. (2014) to detect outliers among inter-robot measurements for multi-agent systems but the technique still requires an initial guess of the relative pose between agents. Most recently Yang and Carlone (2022) introduce a method called STRIDE that reformulates the estimation problem using standard robust cost functions as a polynomial optimization problem. Their method is certifiably optimal and works with up to 90% of the measurements being outliers but does not run in real-time.

Carlone et al. (2014) noted that classifying measurements as inliers or outliers is an unobservable task. In light of this, the focus of research has changed from classifying measurements as inliers and outliers to identifying the largest consistent or compatible set of measurements. Joint compatibility branch and bound (JCBB), first introduced by Neira and Tardós (2001), is a method that searches for the largest jointly compatible set. However, utilizing JCBB in multi-robot mapping problems can be difficult because it requires solving the graph for a combinatorial number of measurement combinations to evaluate the likelihood of each measurement given each combination of the other measurements.

Single-cluster spectral graph partitioning (SCGP), used by Olson et al. (2005), identifies an inlier set by thresholding the eigenvector associated with the largest eigenvalue of the adjacency matrix of the underlying consistency graph. SCGP has successfully been applied to pose SLAM (Olson 2009) as well as range-only SLAM (Olson et al. 2005). A similar algorithm called CLIPPER thresholds the eigenvector associated with the largest eigenvalue of the compatibility graph's affinity matrix and shows that they can identify inliers in 99% outlier regimes. CLIPPER has successfully been applied in point-cloud, line-cloud, and plane-registration problems (Lusk et al. 2021; Lusk and How 2022).

In Mangelson et al. (2018) we introduce PCM which also generates a consistency graph, but transforms the problem into an instance of the maximum-clique problem by showing that the maximum clique represents the largest pairwise consistent set. Graph-based Maximum Consensus Estimation for Point Cloud Registration (GMCR) (Gentner et al. 2023) builds on PCM and decouples the scale, rotation, and translation estimation into three separate problems. GMCR utilizes a translation invariant and a rotation invariant that are used to generate three separate consistency graphs (one for scale, rotation, and translation). The maximum clique of each of these graphs is found and used to estimate the respective quantity. TEASER++, introduced by Yang et al. (2020b), also decouples the scale, rotation, and translation estimation but formulates the problem as a truncated least-squares problem that is certifiably optimal. TEASER++ utilizes maximum clique methods after the scale estimation step to prune the number of outliers present and decrease the problem size before performing the rotation and translation estimation. These algorithms are designed and tested for point cloud registration while our proposed algorithm is tested in multiple applications and can easily be applied to problems not tested here. Recent work done by Zhang et al. (2023) utilizes multiple maximal cliques, instead of just the maximum clique, when identifying correspondences in point cloud registration. The use of maximal cliques resulted in more correct correspondences being identified in point cloud registration problems but also resulted in more error in the estimated rotation and translation between point clouds than when the maximum clique was used.

All consistency-based methods described previously are constrained to evaluate consistency on a pairwise basis. Shi et al. (2021) generalize the evaluation of consistency to groups of k measurements and uses the maximum k-core of an embedded consistency graph to quickly approximate the maximum clique in an algorithm called ROBIN. Our prior work (Forsgren et al. 2022) introduces GkCM which evaluates measurements for consistency in groups of $k \geq 2$ and generates a generalized consistency graph where edges in the graph connect k nodes. The effectiveness of GkCM was demonstrated in range-based SLAM which requires evaluating groups of 4 measurements for consistency. We also modified the maximum-clique algorithms presented by Pattabiraman et al. (2015) to find the maximum clique of a generalized graph given that no other maximum-clique algorithm over a generalized graph existed. Since then, Shi et al. (2022) have presented a mixed-integer linear program (MILP) that will find the maximum clique of a generalized graph.

Our contributions in this paper are a generalization of the work done by Mangelson et al. (2018), and an extension of the work by Forsgren et al. (2022). Specifically, we demonstrate the effectiveness of the proposed GkCM algorithm in multi-agent SLAM scenarios and propose a new consistency function for multi-agent visual SLAM applications that requires evaluating consistency in groups of three. We present heuristics that alleviate the computational burden of the GkCM algorithm and compare our generalized maximum clique algorithm with another maximum clique algorithm over generalized graphs.

3. Problem formulation

In our factor-graph formulation of SLAM, we denote time-discretized versions of the robot trajectory by $\mathbf{x}_i \in \mathrm{SE}(2)$ or

SE(3). The factors in the graph are derived from the measurements observed by the robot and penalize estimates of the trajectory that make the observed measurement unlikely. We denote measurements that relate the variables $\mathbf{x}_i$ and $\mathbf{x}_j$ by $\mathbf{z}_{ij}$ and call them odometric measurements if i and j are consecutive and loop-closure measurements if i and j are non-consecutive in time. The goal of pose graph SLAM is, then, to estimate the most likely value of each pose variable $\mathbf{x}_i$ given the measurements $\mathbf{z}_{ij}$. We can formulate the single-robot pose graph SLAM problem as the MLE

$$\widehat{\mathbf{X}} = \underset{\mathbf{X}}{\operatorname{argmax}}\ P(\mathbf{Z}|\mathbf{X}). \tag{1}$$

where $\mathbf{X}$ is the set of all pose variables $\mathbf{x}_i$, and $\mathbf{Z}$ is the set of all relative pose measurements $\mathbf{z}_{ij}$.

In multi-robot SLAM, we also need to estimate the relative transformation between the local coordinate frames of the respective robots. We adopt the method presented by Kim et al. (2010), which proposes the use of an anchor node for each trajectory that encodes the pose of the vehicle's local coordinate frame with respect to some global reference frame. We denote the homogeneous transformation matrix representing this offset by T_a^g and represent measurements relating cross-trajectory poses by $\mathbf{z}_{ij}^{ab}$, where a and b are robot IDs and i and j respectively denote which poses on robots a and b are being related. T_a^g is an element of SE(2) or SE(3). $\mathbf{z}_{ij}^{ab}$ is also often an element of SE(2) or SE(3) but can be a function of this transformation in general.

In the case of two robots, the SLAM estimation problem becomes

$$\widehat{\mathbf{X}}, \widehat{\mathbf{T}}^g = \underset{\mathbf{X}, \mathbf{T}^g}{\operatorname{argmax}}\ P\left(\mathbf{Z}^a, \mathbf{Z}^b, \mathbf{Z}^{ab} \middle| \mathbf{X}, \mathbf{T}^g\right), \tag{2}$$

where $\mathbf{X}$ now represents the trajectories of both robots, $\mathbf{Z}^{ab}$ represents the set of all cross-trajectory measurements, $\mathbf{Z}^r$ represents the set of measurements local to robot r, and $\mathbf{T}^g = \{\mathbf{T}_a^g, \mathbf{T}_b^g\}$. This problem can be treated as weighted, nonlinear least squares and can be solved efficiently using an array of specialized optimization libraries.

Existing methods can handle outlier measurements in the local measurement sets $\mathbf{Z}^a$ and $\mathbf{Z}^b$, but not in the inter-robot set $\mathbf{Z}^{ab}$ since no prior estimate of the initial transformation between coordinate frames of the robots exists in general. The focus of this paper is on selecting a subset of the measurements in the inter-robot set $\mathbf{Z}^{ab}$ that can be trusted. The next sections outline our approach to accomplish this.

4. Pairwise consistent measurement set maximization

In this section, we first define a novel notion of consistency and then we use that notion to formulate the selection of inter-robot loop closure measurements as a combinatorial optimization problem that finds the largest consistent set.

4.1. Pairwise consistency

Directly determining if a measurement is an inlier or outlier from the graph itself is unobservable as shown by Carlone et al. (2014). Thus, instead of trying to classify inlier versus outlier, we attempt to determine the maximum subset of measurements that are internally pairwise consistent:

Definition 1. *A set of measurements $\tilde{\mathbf{Z}}$ is **pairwise internally consistent** with respect to a consistency metric C and the threshold γ if*

$$C\left(\mathbf{z}_i, \mathbf{z}_j\right) \leq \gamma, \quad \forall \quad \mathbf{z}_i, \mathbf{z}_j \in \tilde{\mathbf{Z}} \tag{3}$$

where C is a function measuring the consistency of measurements $\mathbf{z}_i$ and $\mathbf{z}_j$, and γ is chosen a priori.

This definition of consistency requires that every measurement in the set be consistent with every other measurement in the set with respect to C and γ. The details of the function C are problem-dependent and will be left to the user to determine. We describe our target application and the associated consistency function in Section 9.1.

It should also be noted that pairwise consistency does not necessarily signify full joint consistency. A set of measurements can be pairwise internally consistent but not jointly consistent. However, checking full joint consistency is an exponential operation and requires possibly checking every combination of measurements to evaluate their consistency (Neira and Tardós 2001). Finding the maximum-cardinality pairwise-consistent set is also exponential, but by formulating the problem in this way, we can leverage a body of literature on the maximum-clique problem in graph theory that can find or estimate the solution efficiently. In practice, we observed that testing for pairwise consistency was restrictive enough to filter inconsistent measurements from typical pose graphs with full degree-of-freedom measurements.

4.2. The maximal cardinality pairwise consistent set

Having this definition of pairwise internal consistency allows us to restrict our algorithm to only consider sets of measurements that are pairwise internally consistent; however, due to perceptual aliasing, we may end up with multiple subsets that are pairwise internally consistent. We need to find a way to select between these possible subsets.

The underlying assumption of our method is based on the following two initial assumptions:

Assumption 1. *The pose graphs are derived from multiple robots or the same robot in multiple sessions exploring the same environment.*

Assumption 2. *The inter-robot measurements are derived from observations of that environment and the system used to derive them is not biased toward selecting incorrect measurements over correct ones.*

These assumptions fit a large number of multi-robot mapping situations and are reasonable even in perceptually aliased environments whenever a place recognition system does not systematically select the perceptually aliased measurement over the correct ones.

If the above conditions are met then the following can also be safely assumed:

Assumption 3. *As the number of inter-robot measurements increases, the number of measurements in the correct consistent subset will grow larger than those in the perceptually aliased consistent subsets.*

Our goal is, then, to efficiently find the largest consistent subset of $\mathbf{Z}^{ab}$, which we denote by $\mathbf{Z}^*$.

To formalize this, we introduce a binary switch variable, s_u, for each constraint in the set $\mathbf{Z}^{ab}$ and let s_u take on the value 1 if the measurement is contained in the chosen subset, and 0 otherwise. Note that there is a single s_u for each measurement $\mathbf{z}_{ij}^{ab} \in \mathbf{Z}^{ab}$; however, for simplicity of notation, we now re-number them with the single index u and denote the corresponding measurement $\mathbf{z}_{ij}^{ab}$ by $\mathbf{z}_u$. Letting $\mathbf{S}$ be the vector containing all s_u, our goal is to find the solution, $\mathbf{S}^*$, to the following optimization problem

$$\begin{aligned} \mathbf{S}^* &= \underset{\mathbf{S}\in\{0,1\}^m}{\operatorname{argmax}} \ \|\mathbf{S}\|_0 \\ \text{s.t.} &||\epsilon_{uv}||_{\Sigma_{uv}} \ s_u s_v \leq \gamma \ \forall \ u, v, \end{aligned} \tag{4}$$

where m is the number of measurements in $\mathbf{Z}^{ab}$, $\mathbf{z}_u$ is the measurement corresponding to s_u, ϵ_{uv} is the associated error term corresponding to measurements $\mathbf{z}_u$ and $\mathbf{z}_v$, and Σ_{uv} is the covariance matrix associated with the error ϵ_{uv}. We refer to this as the PCM problem.

Once found, we can use $\mathbf{S}^*$ to index into $\mathbf{Z}^{ab}$ and get $\mathbf{Z}^*$. This consistent subset of the measurements can then be plugged into any of the existing nonlinear least-squares-based solvers to merge the individual robot maps into a common reference frame. In the next section, we show how this problem can be reformulated into an equivalent problem that has been well studied.

5. Solving PCM via maximum clique over consistency graphs

In this section, we describe how to solve the PCM problem. The goal of PCM is to determine the largest subset of the measurements $\mathbf{Z}^{ab}$ that are pairwise internally consistent. This pairwise consistency is enforced by the n^2 constraints listed in equation (4) where n is the number of measurements in $\mathbf{Z}^{ab}$. It is important to note that the norm on the left-hand side of the constraints does not contain any of the decision variables s_i. These distance measures can be calculated in pre-processing and combined into a matrix of consistency measures $\mathbf{Q}$, where each element $[\mathbf{Q}]_{uv} = q_{uv} = ||\epsilon_{uv}||_{\Sigma_{uv}}$, corresponds to the consistency of measurement $\mathbf{z}_u$ and $\mathbf{z}_v$. This process is depicted in steps B and C in Figure 1.

We will now introduce the concept of a consistency graph.

Definition 2. *A* ***consistency graph*** *is a graph $G = \{V, \mathcal{E}\}$ where each vertex $v \in V$ represents a measurement and each edge $e \in \mathcal{E}$ denotes consistency of the vertices it connects.*

We can transform the matrix of consistency measures $\mathbf{Q}$ into the adjacency matrix for a consistency graph if we threshold it by γ and make it symmetric by requiring that both q_{uv} and q_{vu} be less than or equal to γ to insert an edge into the graph. An example adjacency matrix and a consistency graph are shown in step D of Figure 1.

A *clique* in graph theory is defined as a subset of vertices in which every pair of vertices has an edge between them and the *maximum clique* is the largest such subset of nodes in the graph. A clique of the consistency graph corresponds to a *pairwise internally consistent set* of measurements because every measurement is pairwise consistent with every other measurement in the set. Thus, the solution to the problem defined in equation (4) is the maximum clique of the consistency graph (see step E of Figure 1).

In graph theory, the problem of finding the maximum clique for a given graph is called the maximum clique problem and is an NP-hard problem (Wu and Hao 2015). Zuckerman (2006) and Feige et al. (1991) show that the maximum clique problem is also hard to approximate, meaning that finding a solution arbitrarily close to the true solution is also NP-hard. Dozens of potential solutions have been proposed, each of which can be classified as either an exact or a heuristic algorithm. All of the exact algorithms are exponential in complexity and are usually based on branch and bound, while the heuristic algorithms often try to exploit some type of structure in the problem, making them faster, but not guaranteeing the optimal solution (see Wu and Hao 2015).

In 2015, Pattabiraman et al. (2015) proposed a method that aggressively prunes the search tree and can find maximum-clique solutions for large sparse graphs relatively quickly. They present both an exact algorithm as well as a heuristic version that can be used when the exact algorithm becomes intractable. Though our method could theoretically use any one of the proposed maximum clique algorithms, we selected the one proposed by Pattabiraman et al. (2015) because of its simplicity, parallelizability, and open-source implementation.

6. Group-*k* consistency maximization

In this section, we generalize the notion of consistency to sets of $k > 2$ measurements and use this generalized definition to formulate a combinatorial optimization problem.

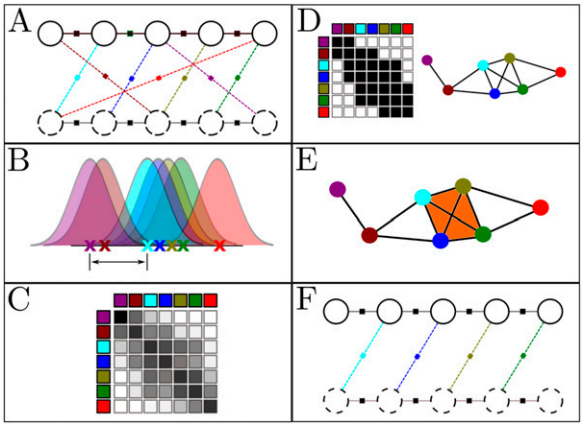


Figure 1. An illustration of the Pairwise Consistency Maximization (PCM) algorithm for selecting consistent inter-map loop closures measurements. (a) Given two independently derived pose graphs (shown in dotted and solid circles in step A) and a set of potential loop closures between them (shown by colored, dotted lines), our goal is to determine which of these inter-robot loop closures should be trusted. (b) Using a consistency metric such as Mahalanobis distance, we calculate the consistency of each pairwise combination of measurements. (c) We store these pairwise consistency values in a matrix where each element corresponds to the consistency of a pair of measurements. (d) We can transform this matrix into the adjacency matrix for a *consistency graph* by thresholding the consistency and making it symmetric using the maximum consistency when associated elements across the diagonal have differing consistency values. Each node in this graph represents a measurement and edges denote consistency between measurements. Cliques in this graph are *pairwise internally consistent sets*. (e) Finding the maximum clique represents finding the largest pairwise internally consistent set. (f) After determining the largest consistent set, we can robustly merge the two pose graphs using only the consistent inter-map loop closures, allowing us to reject false measurements.

While maximizing pairwise consistency in Mangelson et al. (2018) outperformed other existing robust SLAM methods, pairwise consistency is not always a sufficient constraint to identify outlier measurements. For example, a set of three range measurements may all intersect in a pairwise manner even if the set of measurements do not intersect at a common point, indicating that they are pairwise consistent but not group-3 consistent.

As currently framed, the consistency function defined in equation (3) is constrained to take only two measurements as its input. In some scenarios, such as with the range measurements described above, it may be useful to define a consistency function that depends on more than two measurements.

6.1. Group-k consistency

To handle the situation where consistency should be enforced in groups of greater than two measurements we now define a novel notion of *group-k internally consistent sets*.

Definition 3. *A set of measurements* $\tilde{\mathbf{Z}}$ *is* ***group-k internally consistent*** *with respect to a consistency metric* C *and the threshold* γ *if*

$$C(\{\mathbf{z}_o, \ldots, \mathbf{z}_k\}) \leq \gamma, \ \forall \quad \{\mathbf{z}_0, \ldots, \mathbf{z}_k\} \in \mathcal{P}_k(\tilde{\mathbf{Z}}) \tag{5}$$

where C *is a function measuring the consistency of the set of measurements* $\{\mathbf{z}_o, \ldots, \mathbf{z}_k\}$, $\mathcal{P}_k(\tilde{\mathbf{Z}})$ *is the set of all permutations of* $\tilde{\mathbf{Z}}$ *with cardinality* k, *and* γ *is chosen* a priori.

This definition of consistency requires that every combination of measurements of size k be consistent with respect to C and γ. We note that the n-invariant introduced by Shi et al. (2021) is a special case of our generalized consistency function that does not depend on the relative transformation between poses where measurements were taken. The appropriate choice of consistency function is problem-dependent and therefore left to the user to determine. However, we define our target applications and their associated consistency functions in Section 9. As with pairwise consistency, establishing group-k consistency does not guarantee full joint consistency. We settle for checking group-k consistency and use it as an approximation for joint consistency to keep the problem tractable.

It is worth noting that certain consistency checks may contain degenerate cases where the consistency may not be evaluated. One such example occurs when using trilateration with range measurements to localize a feature in the environment. If two or more of the poses where two range measurements are identical there will not be a unique location for the feature that is consistent with the range measurements. In scenarios where degeneracies are possible, a test to detect a degeneracy can be designed. If the test indicates the poses are in a degenerate configuration, one or more of the measurements can be stored in a buffer whose consistency with the maximum clique can be tested later. If a degeneracy is still present, then the consistency of the measurement must be tested another way, or the measurement be labeled inconsistent. In practice, we found that degenerate configurations did not present an issue because of noise inherent in our target applications.

6.2. Group-k consistency maximization

Analogous to pairwise consistency defined by Mangelson et al. (2018), we now want to find the largest subset of measurements that is internally *group-k consistent*. We use the same assumptions described in Section 4.2.

As in PCM, our goal is to find the largest consistent subset of $\mathbf{Z}$. We accomplish this by introducing a binary switch variable s_u for each measurement in $\mathbf{Z}$ and let s_u be 1 if the measurement is contained in the chosen subset and 0 otherwise. Letting $\mathbf{S}$ be the vector containing all s_u, our goal is to find $\mathbf{S}^*$ to the following optimization problem

$$\begin{aligned} \mathbf{S}^* = & \underset{\mathbf{S}\in\{0,1\}^m}{\operatorname{argmax}} \ \|\mathbf{S}\|_0 \\ \text{s.t. } & C(\{\mathbf{z}_0,\ldots,\mathbf{z}_k\})\ s_0\cdots s_k \le \gamma \\ & \forall\{\mathbf{z}_0,\ldots,\mathbf{z}_k\}\in\mathcal{P}_k(\mathbf{Z}) \end{aligned} \tag{6}$$

where m is the number of measurements in $\mathbf{Z}$ and $\mathbf{z}_u$ is the measurement corresponding to s_u. We refer to this problem as the Group-k consistency maximization, or GkCM, problem. This problem is a generalization of PCM, and for $k = 2$ they become identical.

6.3. Solving group-k consistency maximization

As with PCM, we can solve the GkCM problem by finding the maximum clique of a consistency graph. However, because we want to find the largest subset that is group-k internally consistent, we need to operate over generalized graphs. In graph theory, a *k-uniform hypergraph* (or *generalized graph*), G, is defined as a set of vertices V and a set of k-tuples of those vertices $\mathcal{E}$ (Bollobás, 1965). Each k-tuple is referred to as an edge and a *clique*, within this context, is a subgraph of G where every possible edge exists in $\mathcal{E}$. We now introduce the concept of a *generalized consistency graph*:

> **Definition 4.** *A **generalized consistency graph** is a generalized graph $G = \{V, \mathcal{E}\}$ with k-tuple edges, where each vertex $v \in V$ represents a measurement and each edge $e \in \mathcal{E}$ denotes consistency of the vertices it connects. An example of a generalized graph and its maximum clique are shown in* Figure 2.

Solving equation (6) is equivalent to finding the maximum clique of a generalized consistency graph and consists of the following two steps: Building the generalized consistency graph and finding the maximum clique. The next two sections explain these processes in more detail.

7. Building the generalized consistency graph

The graph is built by creating a vertex for each measurement and performing the relevant consistency checks to determine what edges should be added. If the graph is created all at once, there are $\binom{m}{k}$ checks to perform. If the graph is being built incrementally by checking the consistency of a newly added measurement with those already in the graph then the number of checks is $\binom{m-1}{k-1}$. Due to the number of checks to be performed, it is important that the consistency function in equation (5) be computationally efficient. Note that all the checks are independent, allowing for the computation and graph construction to be parallelized on a CPU or GPU to decrease the time to perform the necessary checks. Additionally, memory usage is not significantly affected by the choice to perform the consistency checks incrementally or in batch.

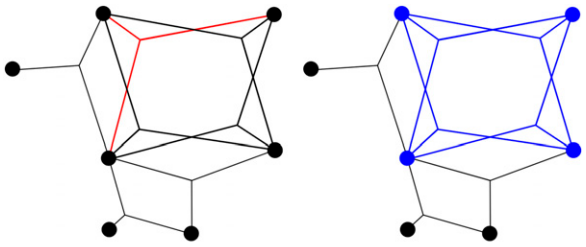


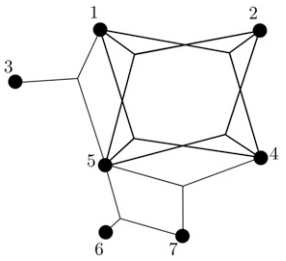
Figure 2. An example of a generalized consistency graph with edges made of 3-tuples. (a) Highlights that each edge denotes consistency of three measurements. (b) Highlights the maximum clique of the generalized graph in blue. (a) Generalized graph. (b) Maximum clique.

As a consequence of the combinatorial growth in the number of checks, we need to develop a method that can substantially reduce the number of checks to be computed. We utilize the fact that a set of measurements that are group-k consistent will almost always be group-$(k-1)$ consistent, assuming that the group-k and group-$(k-1)$ consistency functions are similar. Using this observation, we can first perform $\binom{m}{k-1}$ checks and record which combinations of measurements passed the consistency check. Then, when performing the group-k check we only need to check combinations for which all groups of $(k-1)$ measurements passed their respective check. The process can be done starting with group-2 checks and building up to the final group-k check. We show in Section 10 that this can significantly decrease time spent performing consistency checks. This hierarchy-based heuristic can also be used when constructing the graph incrementally.

An example where this heuristic is useful is when determining the consistency of range measurements to a static landmark. Three range measurements are consistent if they all intersect at the same point. However, if any pair of the three measurements do not intersect then any group-3 consistency check containing that pair of measurements will also fail. Thus, when processing measurements, we can check the $\binom{m}{2}$ pairwise intersections of measurements and do only the group-3 checks on combinations where consistency is possible instead of the total $\binom{m}{3}$ checks.

8. Finding the maximum clique of a generalized graph

Once the graph has been built, we can find the largest consistent set by finding the maximum clique of the graph. The PCM algorithm used the exact and heuristic methods presented by Pattabiraman et al. (2015) but these algorithms were not designed for generalized graphs and used only a single thread. Here, we generalize these algorithms to



$d(v_3) = 2$
$N(v_3) = \{1, 5\}$
$E(v_3) = \{\{1, 5\}\}$

$d(v_4) = 4$
$N(v_4) = \{1, 2, 5, 7\}$
$E(v_4) = \{\{1, 2\}, \{1, 5\}, \{2, 5\}, \{5, 7\}\}$

Figure 3. Examples of the degree, neighborhood, and edge set definitions for generalized graphs.

function over k-uniform hypergraphs and provide a parallelized implementation of their algorithms.

We start by defining relevant notation. We denote the n vertices of the graph $G = (V, \mathcal{E})$ as $\{v_1, \ldots, v_n\}$. Each vertex has a neighborhood $N(v_i)$, that is the set of vertices connected to that vertex by at least one edge. The degree of v_i, $d(v_i)$, is the number of vertices in its neighborhood. We also define an edge set, $E(v_i)$, for each vertex consisting of a set of $(k-1)$-tuples of vertices. The edge set is derived from the set of k-tuples in $\mathcal{E}$ containing the given vertex by removing the given vertex from each edge. Figure 3 shows an example of these values for a given graph.

A set of vertices, U, that would increase the clique size by one is found (MaxClique line 11) from the set of edges R that a valid candidate vertex must have (MaxClique line 7). The Clique function then recursively iterates through potential cliques and updates R and U (Clique lines 13, 16). The clique is tracked with S and a check is performed to see if $S > S_{\max}$ where $S_{\max}$ is replaced with S if the check passes. The process is repeated for each vertex in the graph (MaxClique line 3). The exact algorithm evaluates all possible cliques, and as such, the time complexity of the exact algorithm is exponential in the worst case.

The heuristic algorithm, Algorithm 2, has a similar structure to the exact algorithm but uses a greedy search to find a potential maximum clique more quickly. For each node with a degree greater than the size of the current maximum clique (MaxCliqueHeu line 4), the algorithm selects a clique of size k who has the greatest number of connections in $E(v_i)$ (MaxCliqueHeu line 5). This is done by summing the number of connections each node in $N(v_i)$ has in $E(v_i)$ and selecting the edge $e \in E(v_i)$ with the sum total of connections. If the selected clique can potentially be made larger than $S_{\max}$, then a greedy search selects nodes based on the largest number of connections in $E(v_i)$ (CliqueHeu line 5). The generalized heuristic algorithm presented in

Algorithm 1 Exact Algorithm for Finding the Maximum Clique of a k-Uniform Hypergraph.
Input: Graph $G = (V, \mathcal{E})$, **Output:** Maximum Clique S_{max}

```
1:  function MaxClique(G = (V, E))
2:      S_max ← ∅
3:      for i = 1 to n do
4:          if d(v_i) + 1 ≥ |S_max| then
5:              for each e ∈ E(v_i) do
6:                  S ← e ∪ v_i; U ← ∅
7:                  R ← CombinationsOfSize(S, k − 1)
8:                  for each v_j ∈ N(v_i) do
9:                      if j > i then
10:                         if d(v_j) + 1 ≥ |S_max| then
11:                             if R ⊂ E(v_j) then
12:                                 U ← U ∪ {v_j}
13:                 Clique(G, R, S, U)
```

```
1:  function Clique(G = (V, E), R, S, U)
2:      if U = ∅ then
3:          if |S| > |S_max| then
4:              S_max ← S
5:      while |U| > 0 do
6:          if |S| + |U| ≤ |S_max| then
7:              return
8:          Select any vertex u from U
9:          U ← U \ {u}; S_rec ← S ∪ {u}
10:         N'(u) := {w|w ∈ N(u) and d(w) ≥ |S_max|}
11:         U_rec ← ∅; R_rec ← R
12:         for each p ∈ CombinationsOfSize(S, k − 2) do
13:             R_rec ← R_rec ∪ {p ∪ {u}}
14:         for each q ∈ U ∩ N'(u) do
15:             if R_rec ⊂ E(q) then
16:                 U_rec ← U_rec ∪ {q}
17:         Clique(G, R_rec, S_rec, U_rec)
```

8.1. Algorithm overview

The generalized exact and heuristic algorithms presented in Algorithm 1 and Algorithm 2 respectively are similar in structure to the algorithms by Pattabiraman et al. (2015) but require additional checks to guarantee a valid clique is found since the algorithms now operate over generalized graphs.

The exact algorithm, Algorithm 1, begins with a vertex v and finds cliques of size k that contain v (MaxClique line 5). Algorithm 2 has the same complexity of $O(n\Delta^2)$ as the original algorithm presented by Pattabiraman et al. (2015) despite the modifications made to operate on generalized graphs.

Both algorithms are guaranteed to find a valid clique and can be easily parallelized by using multiple threads to simultaneously evaluate each iteration of the loop on line 3 of MaxClique and MaxCliqueHeu. This significantly decreases the runtime of the algorithm. Our released C++ implementation allows the user to specify the number of threads to be used.

Algorithm 2 Heuristic Algorithm for Finding the Maximum Clique of a k-Uniform Hypergraph.
Input: Graph $G = (V, \mathcal{E})$, **Output:** Potential Maximum Clique S_{max}

```
function MaxCliqueHeu(G = (V, E))
    S_max ← ∅
    for i = 1 to n do
        if d(v_i) + 1 ≥ |S_max| then
            Select e ∈ E(v_i) with max connect. in E(v_i)
            S ← e ∪ v_i; U ← ∅
            R ← CombinationsOfSize(S, k − 1)
            for each v_j ∈ N(v_i) do
                if d(v_j) + 1 ≥ |S_max| then
                    if R ⊂ E(v_j) then
                        U ← U ∪ {v_j}
            if |S| + |U| > |S_max| then
                CliqueHeu(G, R, S, U)
```

```
function CliqueHeu(G = (V, E), R, S, U)
    if U = ∅ then
        if |S| > |S_max| then
            S_max ← S
    Select a vertex u ∈ U with max connect. in E(v_i)
    U ← U \ {u}; S_rec ← S ∪ {u}
    N'(u) := {w | w ∈ N(u) and d(w) ≥ |S_max|}
    U_rec ← ∅; R_rec ← R
    for each p ∈ CombinationsOfSize(S, k − 2) do
        R_rec ← R_rec ∪ {p ∪ {u}}
    for each q ∈ U ∩ N'(u) do
        if R_rec ⊂ E(q) then
            U_rec ← U_rec ∪ {q}
    CliqueHeu(G, R_rec, S_rec, U_rec)
```

A heuristic was introduced by Chang et al. (2021) to avoid computing the maximum clique from scratch in incremental scenarios. When a new measurement is received the maximum clique will either remain unchanged or a larger clique will exist. A more efficient search can be performed by only searching for cliques that contain the new measurement and comparing the largest clique with that measurement to the current maximum clique. We implement this heuristic in our maximum clique algorithms over generalized graphs so that G*k*CM can be performed in both batch and incremental scenarios.

9. Target applications and associated consistency checks

We introduced the notion of a consistency check in equation (5) and noted that the specifics of a consistency check will be dependent on the measurement and the application. This section is dedicated to describing our target applications with the appropriate consistency checks.

9.1. Multi-agent pose graph optimization

Our first application is multi-agent pose graph optimization (PGO). Pose graph optimization is a special case of the SLAM problem where only the vehicle trajectory is estimated. This is accomplished by condensing measurements to environmental features into other inter-pose constraints. This can be accomplished using a variety of sensors and we target the use of LIDAR and camera sensors in our experiments.

9.1.1. Multi-agent LIDAR PGO. LIDAR-based PGO uses a rotating laser-rangefinder to generate a point cloud that describes the distance and direction of objects in the surrounding environment. The rotation and translation between point clouds can be estimated using point cloud registration algorithms like GICP (Segal et al., 2009). When the two point clouds used in the registration algorithm are sequential, an odometry constraint is formed, otherwise it is a loop closure constraint. Point clouds between different robots can be compared and when a match is detected and aligned, an inter-vehicle constraint is formed that describes the relative rotation and translation between the two agents. When two different inter-robot constraints have been obtained, a loop can be formed by concatenating the odometry of each robot with the inter-vehicle constraints. Consistency can be evaluated by traversing the loop formed using two inter-vehicle constraints and checking if the resulting transformation is the identity. In our experimental validation we used the $k = 2$, or pairwise, consistency metric given by

$$C\left(\boldsymbol{z}_{ik}^{ab}, \boldsymbol{z}_{jl}^{ab}\right) = \left|\left|\left(\ominus \boldsymbol{z}_{ik}^{ab}\right) \oplus \widehat{\boldsymbol{x}}_{ij}^{a} \oplus \boldsymbol{z}_{jl}^{ab} \oplus \widehat{\boldsymbol{x}}_{lk}^{b}\right|\right|_{\Sigma} \tag{7}$$

$$\triangleq \left|\left|\epsilon_{ikjl}\right|\right|_{\Sigma_{ikjl}}, \tag{8}$$

which is based on the metric used by Olson (2009). In equation (7) we have adopted the notation of Smith et al. (1990) to denote pose composition using $\oplus$ and inversion using $\ominus$, $\|\cdot\|_{\Sigma}$ signifies the Mahalanobis distance, and the variables $\widehat{\boldsymbol{x}}_{ij}^{a}$ and $\widehat{\boldsymbol{x}}_{lk}^{b}$ are the current relative pose estimates of the associated poses corresponding to inter-robot measurements $\boldsymbol{z}_{ik}^{ab}$ and $\boldsymbol{z}_{jl}^{ab}$.

This choice of metric is useful because it is both easy to compute and follows a chi-squared distribution, giving us a strategy to select the threshold γ without knowledge of the specific dataset. The composition inside the norm of equation (7) evaluates the pose transformation around a loop and should evaluate to the identity transformation in the case of no noise (see Olson, 2009). With Gaussian noise, this normalized squared error follows a chi-squared distribution with degree of freedom equal to the number of degrees of freedom in our state variable. By setting γ accordingly, we can determine if the measurements $\boldsymbol{z}_{ik}^{ab}$ and $\boldsymbol{z}_{jl}^{ab}$ are consistent with one another.

9.1.2. Multi-agent visual PGO. Camera sensors can also be used in PGO problems. The rotation and direction of translation between two images of the same location can be estimated using bundle adjustment or decomposing the essential matrix. Note that without outside information like

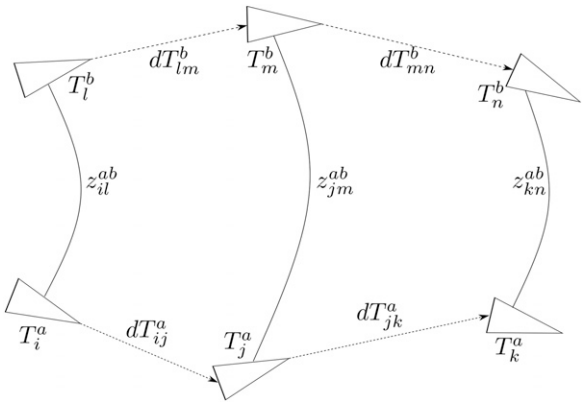


Figure 4. A visual of the setup used in the consistency check for a multi-agent visual PGO scenario. There are two agents each with odometry and several inter-vehicle measurements as outlined in equation (9). All measurements are expressed in frame of robot a.

the baseline between the two images, the scale of the translation cannot be recovered. It is possible to use an IMU to estimate the scale and recover a full relative pose constraint but such information is typically not present in multi-agent scenarios resulting in a low-degree-of-freedom constraint that takes the following form

$$z_{ij}^{ab} = \left(\alpha, \epsilon, R_{ij}^{ab}\right) \tag{9}$$

where R_{ij}^{ab} expresses the rotation of agent b at pose j in the frame of agent a at pose i, and α and ϵ are the azimuth and elevation angles describing the direction of pose j with respect to pose i. Lacking the scale factor causes the measurement to lose a degree of freedom over the full relative pose transformation and means that a pairwise check is no longer sufficient to check the consistency of the inter-vehicle measurements. We provide a visual reference of the setup in Figure 4 showing the relationship of the poses and measurements between the two agents. The $k = 3$ check, $C(z_{il}^{ab}, z_{jm}^{ab}, z_{kn}^{ab})$ has two parts: the first part is similar to the consistency check in equation (7) except that we trace a loop using only the rotations as shown here

$$\begin{aligned} C\left(z_{il}^{ab}, z_{jm}^{ab}\right)_R &= \left|\left|\widehat{R}_{ij}^a \oplus R_{jm}^{ab} \oplus \left(\ominus \widehat{R}_{lm}^b\right) \oplus \left(\ominus R_{il}^{ab}\right)\right|\right|_\Sigma \\ C\left(z_{il}^{ab}, z_{jm}^{ab}\right)_R &\leq \gamma_R. \end{aligned} \tag{10}$$

This portion is repeated for all combinations of two measurements in the group of three. If this check passes, then we proceed with the second part of the consistency check, which verifies that the azimuth and elevation angles are consistent as required by

$$\begin{aligned} C\left(z_{il}^{ab}, z_{jm}^{ab}, z_{kn}^{ab}\right)_d &= \left|\left|h\left(\mathbf{X}_{ijk}^a, \mathbf{X}_{lmn}^b, \mathbf{Z}_{il,jm,kn}^{ab}\right) - \begin{pmatrix}\alpha \\ \epsilon\end{pmatrix}\right|\right|_\Sigma \\ C\left(z_{il}^{ab}, z_{jm}^{ab}, z_{kn}^{ab}\right)_d &\leq \gamma_d, \end{aligned} \tag{11}$$

where the function $h(\mathbf{X}_{ijk}^a, \mathbf{X}_{lmn}^b, \mathbf{Z}_{il,jm,kn}^{ab})$ calculates the expected azimuth and elevation angle using the poses on agents a and b associated with the measurements in $\mathbf{Z}$. The function h first uses two of the measurements in $\mathbf{Z}$ to estimate the scale in the direction of translation by solving a linear least-squares problem given by

$$\begin{aligned} A &= \begin{pmatrix} \bar{t}_{il} & R_{ij}^a \bar{t}_{jm} \end{pmatrix} \\ \mathbf{b} &= -\left(t_{ij}^a - R_{ij}^a R_{jm}^{ab} R_{ml}^b t_{lm}^b\right) \\ \bar{t} &= \begin{pmatrix} \cos(\alpha)\cos(\epsilon) & \sin(\alpha)\cos(\epsilon) & \cos(\epsilon) \end{pmatrix}^T \\ \mathrm{s} &= \begin{pmatrix} s_{il} \\ s_{jm} \end{pmatrix} = A^{-1}\mathbf{b} \end{aligned} \tag{12}$$

where $\bar{t}$ is the unit vector denoting the direction indicated by the azimuth and elevation angles of a particular measurement, t_{ij}^a denotes the position of j with respect to i for agent a, $\mathbf{s}$ is a two vector denoting the scale on the two measurements used, and s_{il} is the scale on the measurement between poses i and l. While testing this function, we found that it was sensitive to the errors in the rotations in the poses and measurements that would often result in a negative scale. To alleviate this, we first perform an initialization step using the single-loop technique described by Carlone et al. (2015). We elected to use the single-loop technique because it has an algebraic solution and is less computationally intense than methods such as chordal initialization that produce accurate initializations in more complex pose graphs.

Once the scale has been recovered, the relative pose and uncertainty between the agents is estimated by applying the scale

$$\begin{aligned} T_{il}^{ab} &= \begin{pmatrix} R_{il}^{ab} & s_{il}\bar{t}_{il}^{ab} \\ 0_{1x3} & 1 \end{pmatrix} \\ \Sigma_T &= H\Sigma_z H^T \end{aligned}$$

where Σ_T is the covariance matrix on the full relative pose, Σ_z is the covariance on the measurement and H is the Jacobian of the function that applies the scale. The poses involved in the third measurement, which we will call T_k^a and T_n^b, can be expressed in a common reference frame as

$$\begin{aligned} T_k^a &= T_k^a \\ T_n^b &= T_{il}^{ab} \oplus T_{ln}^b. \end{aligned}$$

With this information, the expected azimuth and elevation angles are calculated as

$$\begin{aligned} dT &= \left(\ominus T_k^a\right) \oplus T_n^b \\ \alpha &= \operatorname{atan2}(dT.y, dT.x) \\ \epsilon &= \operatorname{atan2}\left(dT.z, \sqrt{dT.x^2 + dT.y^2}\right). \end{aligned} \tag{13}$$

The covariance on dT is calculated using methods described by Mangelson et al. (2020), and the covariance on the

expected azimuth and elevation angles are found by further pre- and post-multiplying by the Jacobian matrix of their respective functions. This process is repeated three times so that each measurement can be validated.

9.1.3. Degenerate configurations. In scenarios such as when two poses lie exactly on top of each other, the relative pose cannot be recovered because the matrix in equation (12) becomes singular, resulting in a degeneracy. Should such a degeneracy occur, solutions such as those discussed in Section 6 may be used. In practice, however, such degeneracies never arose due to the noise inherent in the problem.

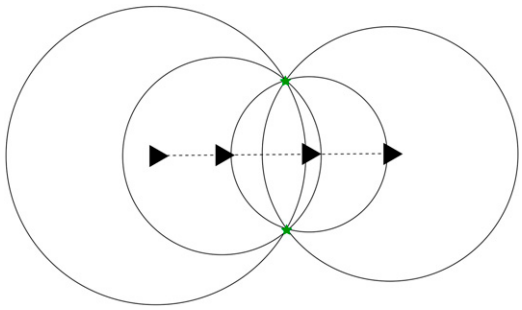


Figure 5. Degenerate pose configuration where range measurements do not result in a unique landmark location.

9.2. Range-based SLAM

In range-based SLAM, a robot explores its environment and takes distance measurements to beacons or landmarks in the environment. This is commonly done with underwater vehicles using acoustic pulses. After several distance measurements to a single beacon have been acquired, the position of the beacon can be estimated and used to correct the odometry of the robot. The number of required measurements depends on whether a 2D or 3D position is being estimated. Our experiments target a 2D scenario and use a $k = 4$ consistency check given by

$$C(\mathbf{z}_{ai}, \mathbf{z}_{bi}, \mathbf{z}_{ci}, \mathbf{z}_{di}) = \left\| h\left(\mathbf{X}_{abcd}, \mathbf{Z}^i_{abc}\right) - \mathbf{z}_{di} \right\|_\Sigma \leq \gamma \qquad (14)$$

where $\mathbf{z}_{di}$ is a range measurement from pose d to beacon i, $\mathbf{X}_{abcd}$ is a tuple of poses $\mathbf{x}_a$, $\mathbf{x}_b$, $\mathbf{x}_c$, and $\mathbf{x}_d$, and $\mathbf{Z}^i_{abc}$ is a tuple of range measurements from poses $\mathbf{x}_a$, $\mathbf{x}_b$, and $\mathbf{x}_c$ to beacon i. The value γ is a threshold value and the function $h(\mathbf{X}_{abcd}, \mathbf{R}^i_{abc})$ is a measurement model defined as

$$h\left(\mathbf{X}_{abcd}, \mathbf{R}^i_{abc}\right) = \left\| \mathrm{l}\left(\mathbf{X}_{abc}, \mathbf{R}^i_{abc}\right) - \mathbf{p}_d \right\|_2 \qquad (15)$$

where $\mathbf{X}_{abc}$ is a tuple of poses $\mathbf{x}_a$, $\mathbf{x}_b$, and $\mathbf{x}_c$, and $\mathbf{p}_i$ is the position of pose i. The function $\mathrm{l}(\mathbf{X}_{abc}, \mathbf{R}^i_{abc})$ is a trilateration function that depends on the input poses and the associated range measurements and returns an estimate of the beacon's location denoted $\mathbf{b}$. The covariance, Σ, is a function of the covariances on the measurements $\mathbf{z}$, and the poses $\mathbf{x}$. The joint covariance, Σ_j, of the poses and beacon location is calculated by forming the measurement Jacobian of a factor graph and using methods described by Kaess and Dellaert (2009). Once the joint covariance has been obtained the covariance, Σ, is calculated as $\Sigma = H\Sigma_T H^T + R_{z_d}$ where $H = \begin{pmatrix} \frac{\partial h}{\partial \mathbf{x}} & \frac{\partial h}{\partial \mathbf{b}} \end{pmatrix}$, $\Sigma_T = blockdiag(\Sigma_j, \Sigma_{z_d})$, and R_{z_d} is the measurement uncertainty on z_d.

The metric checks that the range to the intersection point of three range measurements matches the range of the fourth measurement at some confidence level. The check is done four times for a given set of four measurements, where each permutation of three measurements is used to localize the beacon. As with the consistency functions in Section 9.1, this consistency function follows a χ^2 distribution, meaning that the threshold value γ can be easily chosen without knowledge of the data. Given the combinatorial nature of group consistency, the trilateration algorithm needs to be fast and accurate. The algorithm described by Zhou (2011) fits these criteria and presents a closed-form algorithm that performs comparably to an iterative nonlinear optimization approach but without the need for an initial guess or an iterative solver.

Since the measurements to multiple beacons are often being collected, we make the following simplifying assumption when applying G*k*CM to range-based SLAM.

Assumption 4. *The data association for a given measurement is known meaning that inliers can be identified individually for each beacon. We will relax this assumption later in one of our experiments.*

9.2.1. Degenerate configurations. Since our consistency check defined in equation (14) uses a trilateration algorithm, we need to discuss the scenarios where trilateration fails to provide a unique solution. The first case is where the poses are collinear as shown in Figure 5, and the second is when two of the three poses occupy the same position. The trilateration algorithm by Zhou (2011) is robust to such configurations and can return two estimates for the beacon's location. The consistency check in equation (14) can pass if either estimate is deemed consistent.

10. Evaluation of generalized graph operations

This section marks the beginning of our experimental validation. We first begin by evaluating our algorithms over generalized graphs, followed by an evaluation of PCM using the multi-agent LIDAR PGO consistency check described in Section 9.1. We next evaluate G*k*CM in a single agent range-based SLAM scenario. Lastly, we evaluate G*k*CM in a multi-agent visual PGO application.

We carried out several experiments to evaluate the effectiveness of Algorithm 1, Algorithm 2, and our hierarchy-based approach to evaluating consistency. We also provide a comparison of Algorithms 1 and 2 with the MILP algorithm in Shi et al. (2022) and the maximum *k*-core algorithm in Shi et al. (2021) which, to our knowledge, are the only other algorithms that find or approximate the maximum clique of a generalized graph.

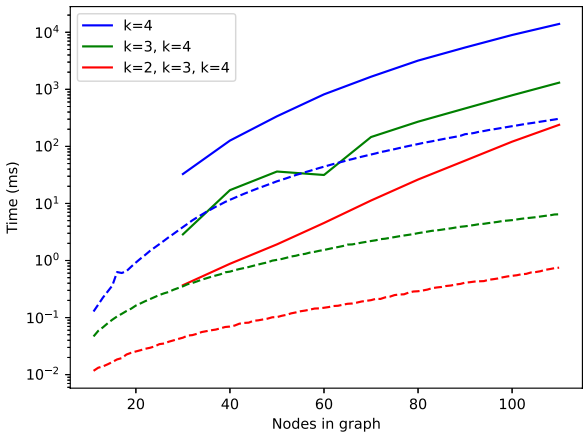


Figure 6. Time to complete all consistency checks in both batch and incremental scenarios using different hierarchies of checks. Batch results are represented by solid lines while incremental results are dashed. Note the log scale on the time axis.

10.1. Hierarchy of consistency evaluation

In this first experiment, we first evaluate the effectiveness of using a hierarchy of consistency checks to decrease the time required to perform all the required checks in a group-4 scenario. For a given number of measurements m, we randomly picked $m/10$ measurements that would be in the maximum clique meaning that these measurements are always consistent with each other. We also randomly identified other groups of measurements that are consistent with each other so that the consistency graph that would be formed would have 20% of the total edges present in the graph. Once all the measurements were generated we evaluated the runtime to check the consistency of all measurements in both a batch and incremental manner. We performed three different evaluations. The first was using just a group-4 check, the second was a group-3 check followed by a group-4 check, and the last was a group-2 check followed by a group-3 check followed by a group-4 check.

We first tested the efficacy of this heuristic in a batch scenario where all measurements are processed after the data has been collected. For a given 4-uniform hypergraph size we generated a graph with n nodes as described previously. We varied the number of measurements between 30 and 110, and tested in increments of 10, measurements recording the time required to process all checks. We repeated the test 100 times and averaged the time to perform all checks. As can be seen in Figure 6, for each hierarchy used we achieved about an order of magnitude speed up for this setup. Using just a group-4 check required an average of 13.9 s to perform the checks for all 110 measurements and we reduced that time to an average of 0.244 s when group-2 and group-3 checks were used to filter out inconsistent combinations.

We also tested the hierarchy of checks in incremental scenarios. The setup for the incremental scenario was the same as the batch scenario except that we started with 10 measurements and measured the time required to process all the required checks as each new measurement came in. We saw similar speed gains as those found in the batch scenario, decreasing the required time from 305 ms to 0.73 ms. The entirety of the results for this experiment can be seen in Figure 6.

We note that this approach is most effective in high outlier regimes where the vast majority of the measurements will not be consistent with one another. In scenarios where the outlier ratio is low, using the proposed hierarchy approach provides no benefit and can even result in increased memory usage since the consistent sets of the previous hierarchy must be stored and increased run times because few combinations will be filtered out. As such we note that the efficacy of this technique will be situation dependent. This technique also shows one advantage of our consistency function formulation over the use of invariants as used in several other works (Shi et al., 2021; Lusk et al., 2021; Gentner et al., 2023) since a lower-order invariant may not exist for a given application.

10.2. Timing comparison of maximum clique algorithms

In this second experiment, we evaluate the runtime characteristics of the proposed maximum clique algorithms over generalized graphs in Algorithms 1 and 2 as well as the MILP algorithm (Shi et al., 2022) and the maximum k-core method (Shi et al., 2021).

We begin by outlining the method for finding the maximum k-core and describing the MILP problem. A k-core of a graph G is the largest subgraph of G such that every vertex in G has a degree of at least k. The maximum k-core is the k-core with maximum k for G or where the $(k+1)$-core is the empty set. We base our maximum k-core algorithm on the work by Matula and Beck (1983) which presents a linear-time algorithm. The k-core algorithm used by Shi et al. (2021) finds the k-core of a 2-uniform hypergraph by embedding a k-uniform hypergraph into a 2-uniform hypergraph. For example, an edge in a 3-uniform hypergraph that connects nodes a, b, and c, would form three edges in a 2-uniform graph. These edges would connect nodes a and b, a and c, and b and c. While Shi et al. (2021) note that the maximum k-core often provides a good approximation of the maximum clique, we note that this is for the maximum clique of a 2-uniform hypergraph. We will show later that this does not always hold for k-uniform hypergraphs.

The MILP algorithm developed in Shi et al. (2022) is defined as follows. Given a k-uniform hypergraph $G(V, \mathcal{E})$ where $|V| = N$ the maximum clique can be found by solving the following MILP given by

$$\begin{aligned} \max_{\mathbf{b}\in\{0,1\}^N} \quad & \sum_{i=1}^{N} b_i \\ \text{s.t.} \quad & \sum_{i\in\mathcal{M}} b_i \leq k-1, \forall \mathcal{M} \subset V, |\mathcal{M}| = k, \mathcal{M} \notin \mathcal{E}. \end{aligned} \tag{16}$$

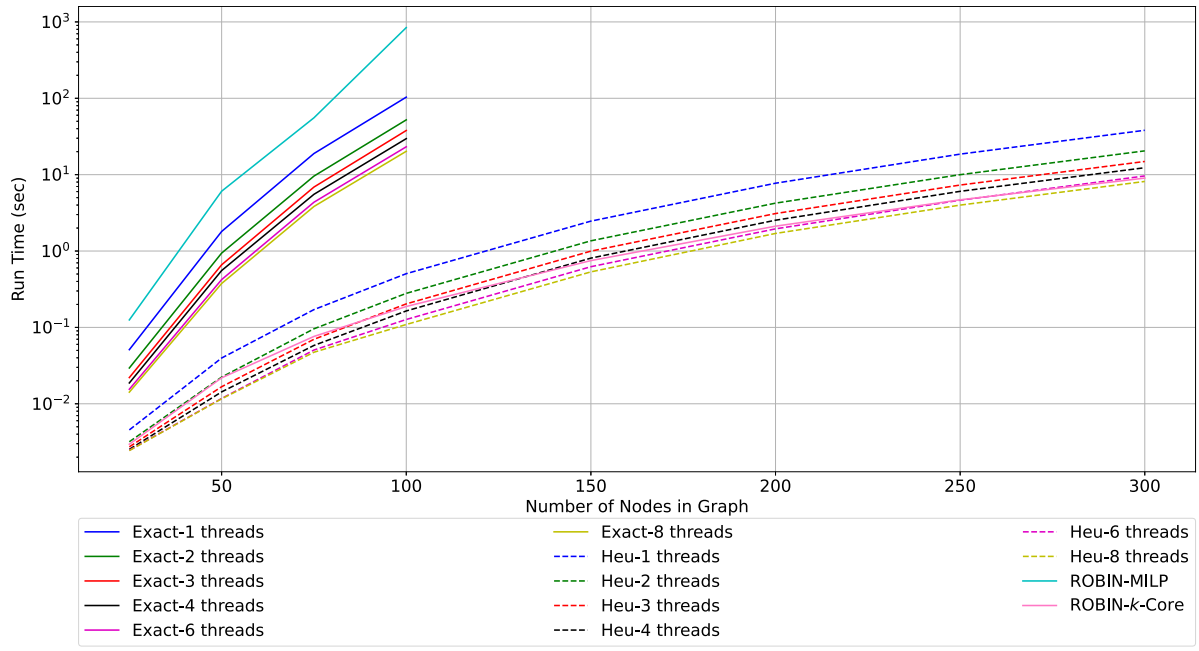


Figure 7. Average runtime for generalized maximum clique algorithms proposed in this paper. We also compare against other generalized maximum clique algorithms in Shi et al. (2022, 2021). This includes both the time to evaluate the necessary data structures such as neighborhoods/edge sets and the time to estimate the maximum clique. Using eight threads, the heuristic algorithm was able to find the maximum clique of a graph with 250 nodes in a few seconds.

The algorithm seeks to maximize the number of vertices in the set $\boldsymbol{b}$ subject to the constraint that for every potential edge $\mathcal{M}$ that does not appear in $\mathcal{E}$, the sum of the variables b_i in that edge must be less than k. We implement this MILP algorithm in C++ using the Gurobi solver Gurobi (2017).

In this experiment, we randomly generated 3-uniform hypergraphs with various node counts ranging from 25 vertices to 300 vertices. Each graph contained all the edges necessary to contain a maximum clique of cardinality ten and additional randomly selected edges to meet a specified graph density. While the run times of the algorithms are dependent on the density of the graph, for this experiment, we chose to hold the density of the graph constant at 0.1 such that approximately ten percent of all potential edges were contained in the graph. We generated 100 sample graphs for each number of nodes and used all four of the algorithms listed above to estimate the maximum clique of each graph and measured the average runtime for each. Figure 7 shows the results of this experiment using various numbers of threads ranging from one to eight. The exact algorithm and the MILP algorithm were only used for graphs with a total number of nodes of 100 or less because of the exponential nature of the algorithms.

As can be seen in Figure 7 the MILP algorithm in Shi et al. (2022) is the slowest algorithm followed by the exact algorithm in Algorithm 1. This is to be expected since both algorithms are guaranteed to find the maximum clique and do not use any heuristics or approximations that would cause them to find a suboptimal solution. Both the k-core and heuristic algorithm (Algorithm 2) run significantly faster, with the heuristic maximum-clique algorithm using eight threads being the fastest in this test case. This is primarily due to the graph density. As noted above, the performance of both Algorithm 1 and Algorithm 2 are largely dependent on the graph's density and that run time will increase as the density increases. We suspect that as the graph density grows the runtime for Algorithm 2 would exceed that of the maximum k-core algorithm. Additionally, if we look at the rate of

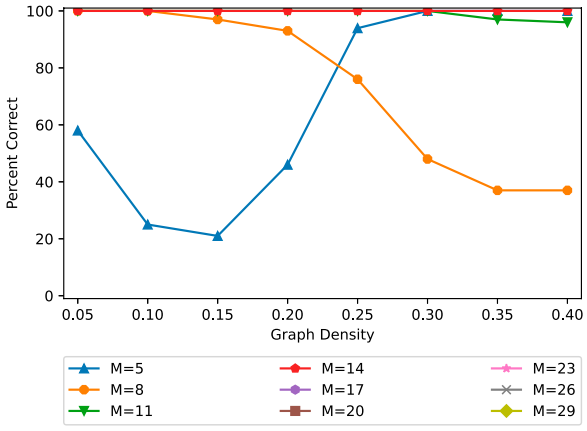


Figure 8. Evaluation of the heuristic algorithm proposed in Algorithm 2. Individual lines denote the cardinality of the maximum clique (where M is the size of the maximum clique) inserted into the graph. The horizontal axis denotes the density of edges in the graph and the vertical axis denotes the percentage of test cases where the algorithm returned a clique of the correct cardinality. The heuristic algorithm returned cliques of the correct size 100% of the time for the graphs with max clique size of 14, 17, 20, 23, 26, and 29.

increase for the k-core and heuristic algorithms, the k-core algorithm seems to scale slightly better with the number of vertices in the graph. For larger graphs than those used in this experiment, we expect that the k-core algorithm will outperform Algorithm 2 in terms of speed.

10.3. Heuristic evaluation

In the third experiment, we again randomly generated 3-uniform hypergraphs as described previously, however, in this case, we varied the density of the graph and the size of the inserted clique, while holding the total number of nodes at 100. For each graph, we used the MaxCliqueHeu algorithm to estimate the maximum clique and then evaluated whether or not the algorithm was successful in finding a clique of the same size as the clique we inserted. We again generated 100 sample graphs for each combination of inserted-clique size and graph density. Figure 8 plots the summarized results. If the algorithm happened to return a maximum clique larger than the inserted clique, then the associated sample was dropped because the inserted clique was not the maximum clique.

This experiment shows that the size of the maximum clique and the success rate of the proposed heuristic algorithm are correlated. In addition, it shows that, with the exception of the case when the inserted clique was small (cardinality 5), the density of the graph and the success rate are inversely correlated. As such, the heuristic seems to perform best when the size of the maximum clique is large or when the connectivity of the graph is relatively sparse.

Prior work states that the maximum k-core provides a good approximation of the maximum clique of a graph. We took this opportunity to evaluate the maximum k-core algorithm used in ROBIN (Shi et al. (2021)). In this experiment, we found that the k-core almost always contained the maximum clique of the generalized graph. However, further examination showed that the k-core was larger than the maximum clique by dozens of nodes. This translates to the maximum k-core including dozens of measurements in the consistent set that should not be used in a SLAM application. Our application experiments later show that the k-core can, at times, provide a good approximation of the maximum clique of a generalized graph and produce usable maps as a result. In general, the maximum k-core does not always provide a good approximation of the maximum clique of a generalized graph, and exploring the cases when a maximum k-core can approximate the maximum clique of a generalized graph is an interesting avenue for future work.

11. Pairwise consistency maximization evaluation

In this section, we evaluate the performance of PCM on a variety of synthetic and real-world datasets. For comparison, we implemented single cluster graph partitioning (SCGP) (Olson et al., 2005), dynamic covariance scaling (DCS) (Agarwal et al., 2013), and random sample consensus (RANSAC) (Hartley and Zisserman, 2003).

We implemented SCGP as described in Olson et al. (2005), except for using an off-the-shelf eigenfactorization library as opposed to the power method for simplicity. We implemented DCS as described in the original paper with $\phi = 5$. We implemented RANSAC by iteratively selecting a single, random inter-map measurement and evaluating the likelihood of the other measurements given the model estimated from the sampled measurement. Because the processing time for this evaluation is so low (given that the Mahalanobis distance evaluations were performed in preprocessing), we exhaustively iterate through all the measurements and evaluate the likelihood of the other measurements with respect to it in turn. We then return the set of measurements that are likely given the sampled point with the largest support. As explained in Section 11.2, RANSAC is especially sensitive to the likelihood threshold and does not check pairwise consistency.

For PCM, we present results using the exact-maximum-clique algorithm (PCM-Exact), as well as the heuristic algorithm (PCM-HeuPatt) as explained by Pattabiraman et al. (2015).

11.1. Simulated 1D world

First, we simulated a one-dimensional world where the robot has a single state variable, x, and receives measurements that are direct observations of that state. We simulate inlier measurements by drawing multiple samples from a Gaussian distribution with a fixed variance and mean x. We simulate both random and perceptually aliased outliers by drawing multiple samples from a single Gaussian with fixed mean and variance and several others from individual Gaussians with random means and variances. We assume the variances are known and are used when computing Mahalanobis distance.

11.1.1. Comparison with combinatorial. For this first experiment, we compare how well PCM-Exact and PCM-HeuPatt approximated the combinatorial gold standard in equation (4). We generated 100,000 sample worlds. On each of these samples, we estimated the pairwise consistent set using the combinatorial solution as well as PCM-Exact, PCM-HeuPatt, SCGP, and RANSAC.

Figure 9 shows a comparison between these four methods with respect to the combinatorial solution. Both PCM methods enforce the consistency of the returned measurements. PCM-Exact returns the same number of points as the combinatorial solution 100% of the time, while PCM-HeuPatt returns the same number of points 98.97% of the time. SCGP varies significantly in both the number of points returned and the consistency of those measurements. RANSAC also sometimes returns more or less points than the combinatorial solution and also fails to enforce measurement consistency. Interestingly, RANSAC is especially dependent on a threshold value. The threshold value for

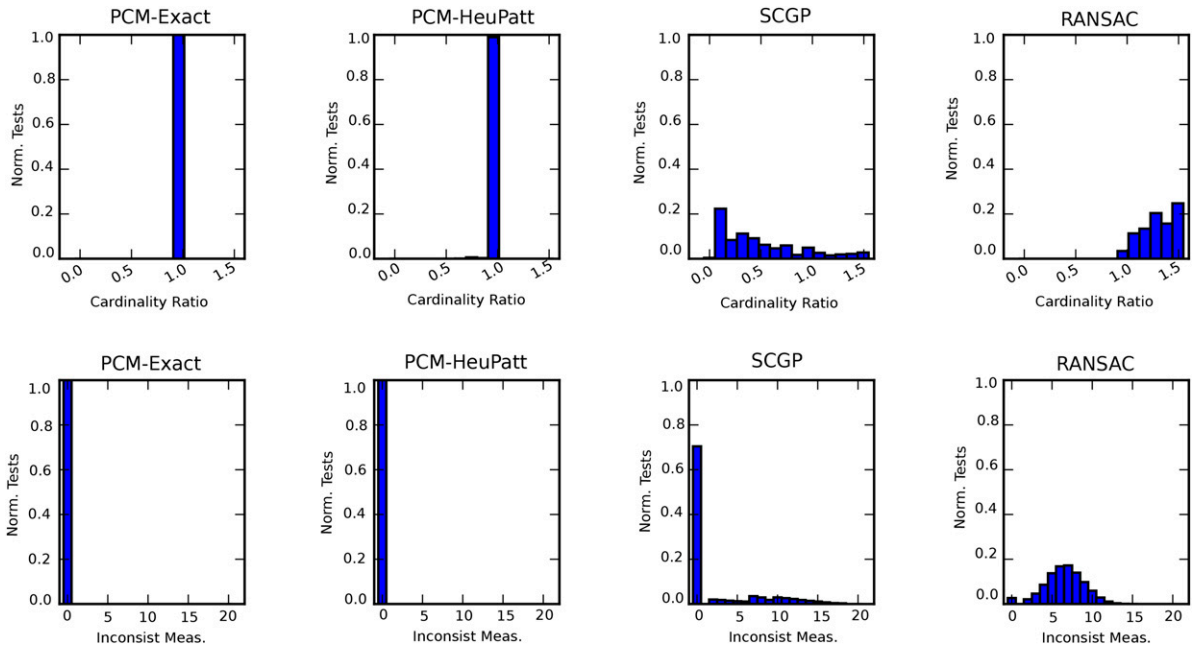


Figure 9. Histograms that evaluate how well PCM, SCGP (Olson et al., 2005), and RANSAC (Hartley and Zisserman, 2003) approximate the combinatorial maximum pairwise consistent set in equation (4). The first row of histogram plots shows the size of the measurement set as compared to the maximum consistent set size. The second row of histograms shows the number of inconsistent pairs returned with respect to the set γ threshold on Mahalanobis distance.

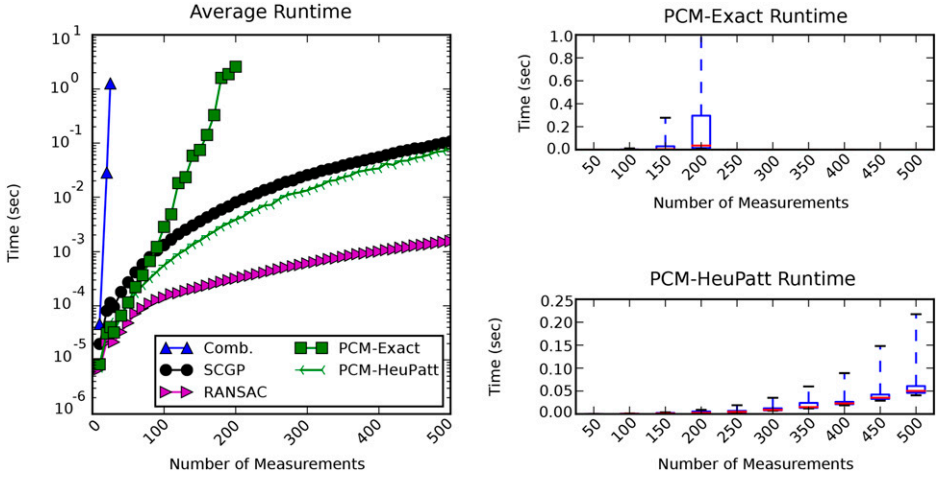


Figure 10. A plot of the evaluation times of the different methods versus the number of measurements being tested. The combinatorial solution takes exponential time and PCM-Exact takes exponential time in the worst case, while the other methods are polynomial in the number of measurements. (This excludes the time to estimate the distance matrix **Q**, which is required for all methods).

RANSAC is centered around a single point and thus is not the same as the threshold value for PCM. If the value is set too high, the number of inconsistent measurements increases. If it is set too low, the total number of returned measurements decreases below the optimal. In Figure 9, RANSAC's threshold is set arbitrarily to show a single snapshot.

11.1.2. Timing comparison. We also used this 1D world to evaluate the timing characteristics of the different algorithms. To test this, we generated 500 sample worlds each for an increasing number of measurement points. The results are shown in Figure 10. Note, these timing results can be significantly improved through parallelization.

11.2. Synthetic 2D comparison

To test our method's accuracy and consistency on a full SLAM dataset, we took a portion of the City10000 dataset released with iSAM (Kaess et al., 2008) and split it to form

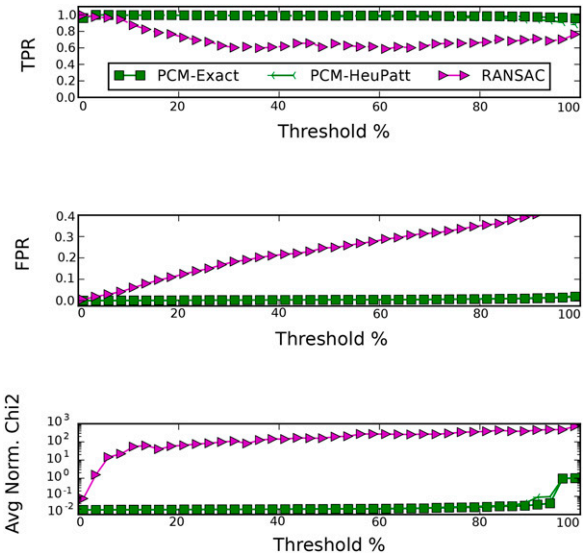


Figure 11. The true positive rate (TPR = TP/(TP + FN)), false positive rate (FPR = FP/(FP + TN)), and average normalized χ^2 value of PCM-Exact, PCM-Heu, and RANSAC versus the threshold value γ. The TPR and FPR can be thought of as the probability of getting a true positive or a false positive. The χ^2 value should be close to zero if the measurements in the graph are consistent.

two separate robot trajectories. After removing all factors connecting the two graphs, we generated 81 different versions of this dataset by randomly selecting a subset of the true loop closures between the two graphs to be used as inliers, as well as randomly adding outlier loop closures to the graph. As before, some of the outliers are internally consistent to simulate perceptual aliasing and some are generated randomly with random mean and covariance. In this experiment, the number of inlier loop closures was 15, there were two groups of five perceptual aliased outliers, and the number of random outliers was 90.

11.2.1. Parameter sweep. Because RANSAC is significantly dependent on the threshold value set, we ran a parameter sweep for the likelihood threshold over all 81 datasets. Figure 11 summarizes this experiment. The true positive rate (TPR) and false positive rate (FPR) of PCM are relatively unaffected by the choice of the threshold parameter as long as it is less than about 85%. RANSAC, on the other hand, has a different FPR for each threshold selected and never has an FPR of zero. This is because PCM conservatively evaluates the consistency of each measurement and determines the consistency of a group of measurements as a whole, while RANSAC selects the largest set of measurements that are likely given a single randomly selected measurement. The last plot shows the average normalized chi-squared value of the residual for the entire graph after solving with the selected factors. This value should be close to zero if the graph is consistent.

The results show that PCM does significantly better at restricting the set of measurements to those that are consistent with one another, decreasing the likelihood of getting a false measurement. This is essential because of the extreme susceptibility of SLAM to false loop closures. The results for PCM-HeuPatt are also almost indistinguishable from those of PCM-Exact.

11.2.2. Accuracy Analysis. To evaluate the accuracy of PCM, we compared its performance on all 81 datasets to SCGP, RANSAC (using the two second to lowest thresholds from Figure 11), and dynamic covariance scaling (Agarwal et al., 2013) or DCS. γ for both PCM-Exact and PCM-HeuPatt was set so that it corresponded to the equivalent of 11% likelihood.

Table 1 gives an overall summary of the results. We used the mean squared error (MSE) of the trajectory of the two graphs (with respect to the no-outlier case), the residual, and the normalized χ^2 value of the nonlinear least squares solver as metrics to evaluate the solution accuracy. The rotation MSE was calculated via $\epsilon_{\text{rot}} = 1/n\sum_i \left|\left|\log(R_{i\_\text{true}}^{\top} R_{i\_\text{est}})\right|\right|_F$ over each pose i and the translation MSE was calculated in the normal manner. For this experiment, all MSE were calculated with respect to the absolute trajectory value.

PCM has the lowest trajectory MSE, and DCS has the lowest residual. Note that DCS also has the highest trajectory MSE, which is as expected. DCS seeks to minimize the least-squares residual error and depends on a good initialization to determine what measurements are consistent enough to not be turned off. Without this initialization, inter-map factors have large residuals and are turned off by the DCS algorithm.

Once given the matrix $\mathbf{Q}$, RANSAC and both PCM methods take about the same amount of time to find the consistency set. The average time to estimate the Mahalanobis distances without the use of analytical Jacobians, parallelization, and incremental updates was 70.8 s.

Figure 12 shows example plots of the estimated maps. Both SCGP and RANSAC have trouble disabling all inconsistent measurements. PCM-HeuPatt accurately approximates PCM-Exact and both PCM methods do well at disabling inconsistent measurements. When PCM does accept measurements not generated from the true distribution, they are still consistent with the uncertainty of the local graphs.

11.3. Real-world pose graph SLAM

We evaluate PCM on the 3D University of Michigan North Campus Long-Term Vision and LiDAR Dataset (NCLT) released by Carlevaris-Bianco et al. (2015). The NCLT dataset was collected using a Segway robot equipped with a LiDAR and Microstrain IMU, along with a variety of other sensors. There are 27 sessions in all with an average length of 5.5 km per session.

For our experiment, we took two sessions collected about 2 weeks apart, removed the first third of one and the last third of the other, and then generated potential loop-closure

Table 1. Results from using DCS, SCGP, RANSAC(with two different thresholds), and PCM to robustly merge maps generated from a synthetic city dataset. These results are a summary of runs on 81 different generated datasets. We evaluated the mean-squared error (MSE) of the two graphs with respect to the non-outlier case (NO-OUT). The worst results for each metric are shown in red, the best are shown in blue, and the second best shown in bold.

	Trans. MSE (m^2)		Rot. MSE		Residual		Inliers		χ^2		Eval Time (sec)
	Avg	Std	Avg	Std	Avg	Std	TPR	FPR	Avg	Std	Avg
NO-OUT	0.0	0.0	0.0	0.0	32.3	0.117	1.0	0.0	N/A	N/A	N/A
DCS	1.83e5	1.19e6	4.17	3.29	**31.7**	**0.507**	0.0	**0.0**	**0.0130**	**< 1.00e-3**	N/A
SCGP	623	1280	0.648	1.54	2.37e5	8.94e5	0.668	0.0510	96.7	364	6.00e-3
RANSAC-1%	**5.69**	**22.0**	**9.00e-3**	**4.00e-2**	185	588	**0.998**	6.00e-3	0.0760	0.239	**< 1.00e-3**
RANSAC-3.5%	183	636	0.236	0.791	3810	18500	0.974	0.0190	1.55	7.53	**< 1.00e-3**
PCM-Exact-11%	**0.276**	**1.54**	**< 1.00e-3**	**3.00e-3**	**45.1**	**105**	**0.997**	**1.00e-3**	**0.0180**	**0.0430**	**< 1.00e-3**
PCM-HeuPatt-11%	**0.276**	**1.54**	**< 1.00e-3**	**3.00e-3**	**45.1**	**105**	**0.997**	**1.00e-3**	**0.0180**	**0.0430**	**< 1.00e-3**

measurements between the two graphs by aligning every fourth scan on each graph using GICP (Segal et al., 2009) and selecting the match with the lowest cost function. We then labeled these registrations as inliers and outliers by thresholding the translation and rotation mean-squared error of the estimated pose transformations with respect to the ground-truth poses for the dataset derived by performing pose graph optimization on all 27 sessions. Finally, to increase the difficulty of the dataset, we removed all but one sixteenth of the measurements labeled as inliers from the graph, resulting in a graph with ten inliers and 98 outliers.

In this experiment, we compare PCM-HeuPatt with DCS, SCGP, and RANSAC. Table 2 provides a comparison of results and Figure 13 shows the estimated maps. The MSE was calculated using the same method as in the prior section, however, in this test we calculated trajectory and map relative pose error separately. The trajectory MSE calculates the error in the estimated relative pose between consecutive nodes allowing us to evaluate graph correctness, while the relative map pose MSE evaluates the offset between the maps.

PCM results in the graph with the best trajectory MSE and the best translational MSE for the relative pose of the two graphs and results in a consistent graph regardless of threshold. It also detects all of the inlier measurements as well as three of the measurements labeled as outliers. DCS once again disables all measurements. SCGP results in an accurate graph but only enables three of the inlier measurements, and finally, RANSAC (for both of the lowest thresholds tried) enables all inliers and several outliers, and results in an inconsistent graph regardless of the threshold selected.

We repeated this experiment but varied the threshold value, γ, used in PCM and RANSAC and recorded the normalized χ^2 error of the resulting optimization which can be seen in Figure 14. As can be seen, PCM maintains a consistent normalized $\chi^2 < 1$ across all threshold values indicating that it chooses a set of measurements that are consistent with each other and the odometry data. The χ^2 values produced using RANSAC vary with the threshold value and are often large indicating that RANSAC was not able to identify a consistent set of measurements.

Note that while in this experiment PCM admits more false positives than in the last experiment, the measurements it accepts are consistent with the inlier measurements and local trajectories even though they were labeled as outliers (Figure 14). Notice that PCM has a better MSE for the relative map pose than the no outlier (NO-OUT) version of the graph. This suggests that by maximizing the consistent set, PCM is selecting measurements that are actually inliers but were mislabeled as outliers when compared to the ground truth. After verification, this turned out to be the case.

It is also important to note that although SCGP results in an accurate graph for this dataset, as shown in the earlier experiment, this does not occur in all cases. In addition, if it fails to select the maximum consistent set of measurements, this can be catastrophic in the case of perceptual aliasing.

12. Range-based SLAM evaluation

In this section, we evaluate the performance of G*k*CM on several synthetic and real datasets where a robot is exploring and taking range measurements to static beacons. We compare the results of G*k*CM, using both the exact (G*k*CM-Exact) and heuristic (G*k*CM-HeuPatt) maximum clique algorithms, to the results of PCM, where the consistency check for PCM is the check used by Olson et al. (2005), and both MILP and k-core variants of ROBIN.

12.1. Simulated 2D world

First, we simulate a two-dimensional world where a robot navigates in the plane. We simulate three different trajectories, (Manhattan world, circular, and a straight line) along with range measurements to static beacons placed randomly in the world. Gaussian noise was added to all range measurements and a portion of the measurements were corrupted to simulate outlier measurements. Half of the corrupted measurements were generated in clusters of size 5, and the other half as single random measurements using a

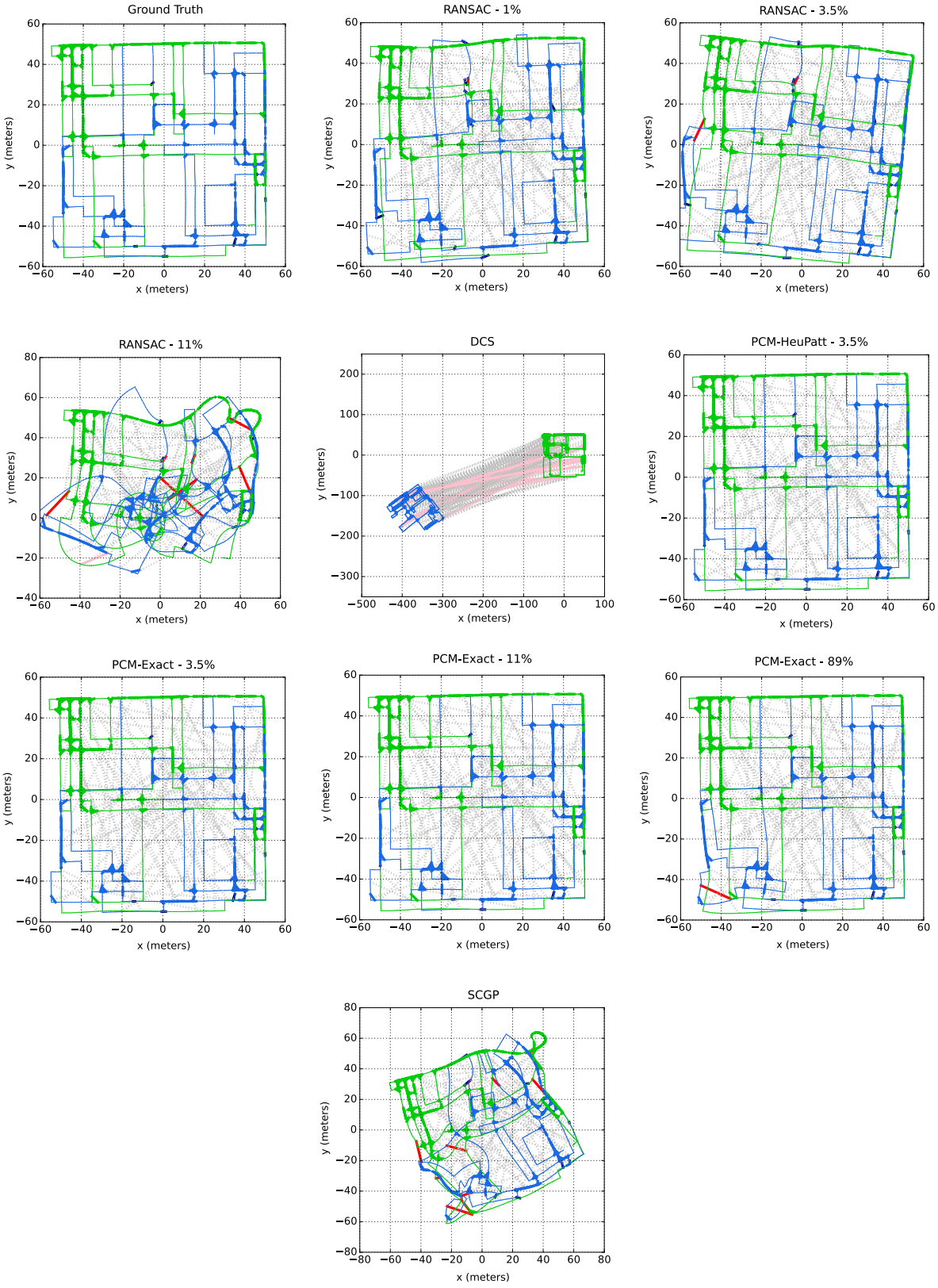


Figure 12. Example plots of the maps estimated by PCM-Exact, PCM-HeuPatt, RANSAC, DCS, and SCGP for one of the generated city datasets. Correctly labeled inlier factors are shown in bold dark blue with correctly disabled outliers shown as dotted gray. Accepted outliers are shown in bold red with disabled inliers shown in pink.

Table 2. Results from using DCS, SCGP, RANSAC(with two different thresholds), and PCM to robustly merge segments extracted From two sessions of the NCLT dataset. NO-OUT corresponds to a version with none of the measurements labeled as outliers. We evaluated the mean squared error (MSE) of the two graphs with respect to the ground truth. The worst results for each metric are shown in red and the best are shown in blue.

	Rel. Pose MSE		Traj. MSE		Residual	Inliers		χ^2	Evaluation
	Trans. (m^2)	Rot.	Trans. (m^2)	Rot.	Error	TP	FP	Value	Time (sec)
NO-OUT	455	0.0308	0.0501	5.00e-4	765	10	0	0.343	N/A
DCS	2.07e5	0.715	0.0502	**5.00e-4**	**724**	0	**0**	**0.257**	N/A
SCGP	522	**0.0162**	0.0502	**5.00e-4**	748	3	**0**	0.342	2.10e-3
RANSAC - 1%	1240	0.0697	0.104	1.50e-3	4230	**10**	6	1.86	$<$ **1.00e-4**
RANSAC - 3.5%	13500	17.4	0.115	4.00e-3	7460	**10**	7	3.28	1.00e-4
PCM-HeuPatt	**386**	0.0245	**0.0501**	**5.00e-4**	817	**10**	3	0.364	1.00e-4

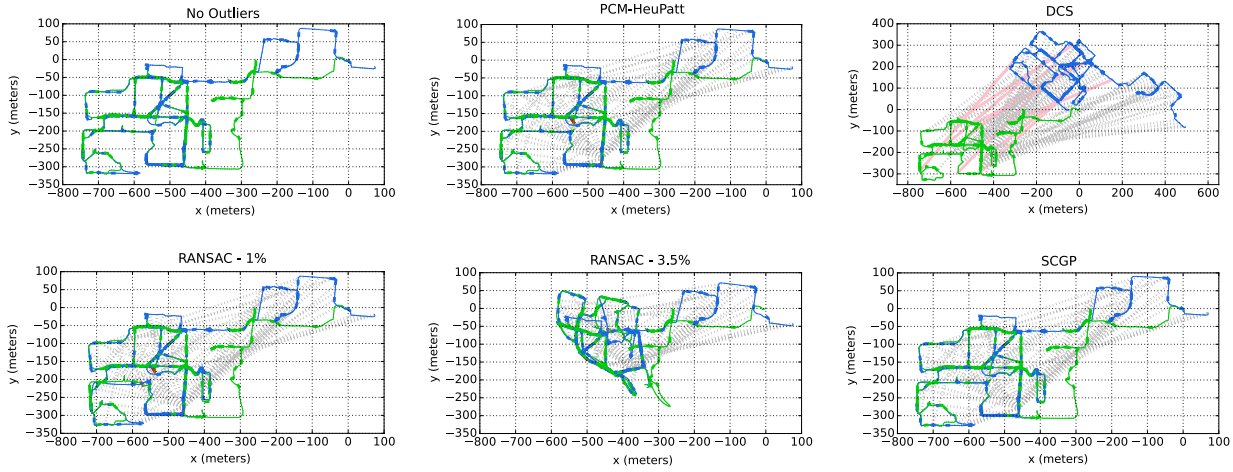


Figure 13. Plots of the trajectories of two partial sessions of the NCLT dataset as estimated by PCM-HeuPatt, RANSAC, DCS, and SCGP.

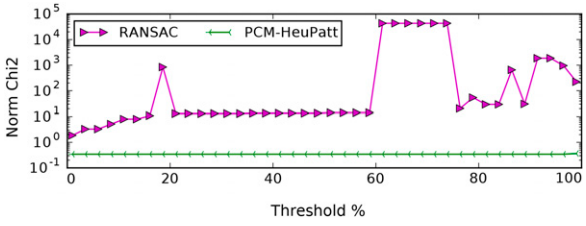


Figure 14. The normalized χ^2 value of the resulting NCLT graph versus the threshold value γ for PCM-HeuPatt and RANSAC. The χ^2 value for RANSAC is never below one signifying that the selected factors are probabilistically inconsistent, while the χ^2 value for PCM is relatively constant and below 1.0 regardless of threshold.

Gaussian distribution with a random mean and a known variance. We assume that the variances of the range measurements are known and that these variances are used when performing the consistency check. The simulation was run multiple times varying values such as the trajectory and beacon locations, and statistics were recorded for comparison.

12.1.1. Monte Carlo experiment. This first example was done to show how G*k*CM performs in situations with large percentages of outliers. In this experiment, a trajectory of 75 poses was simulated with measurements being taken at each pose and 60 of the measurements were corrupted to be outliers. G*k*CM was used to identify consistent measurements that were used to solve the range-based SLAM problem in equation (1) using GTSAM (Kaess et al., 2012). The experiment averaged statistics over 100 runs and results are shown in Table 3 where we report the mean and standard deviation for each metric. Figure 15 shows visual examples of the maps produced using the set of selected measurements by each method except G*k*CM-Exact because it is identical to the result in ROBIN-MILP.

As can be seen, G*k*CM-Exact and ROBIN-MILP perform the best across most metrics followed closely by G*k*CM-HeuPatt. It is expected that G*k*CM-Exact and ROBIN-MILP would perform equivalently because they are

Table 3. Statistics comparing G*k*CM to other methods in the Monte Carlo experiment. Best results are in **bold**.

	Trans. RMSE (m)		Rot. RMSE (rad)		Beacon error (m)		Residual		Inliers		χ^2			Evaluation time (s)
	Avg	Std	Avg	Std	Avg	Std	Avg	Std	TPR	FPR	Avg	Std	Median	Avg
G*k*CM-HeuPatt	1.15	1.23	0.107	0.0702	**20.9**	**37.2**	290	502	0.793	**3.66e-3**	1.42	2.45	**0.443**	0.401
G*k*CM-exact	**1.11**	**0.95**	**0.104**	**0.0644**	21.1	37.5	**279**	**461**	0.794	**3.66e-3**	**1.36**	**2.25**	**0.443**	832
ROBIN-MILP	**1.11**	**0.95**	**0.104**	**0.0644**	21.1	37.5	**279**	**461**	0.794	**3.66e-3**	**1.36**	**2.25**	**0.443**	1450
ROBIN-*k*-core	9.62	12.6	0.782	0.825	45.4	59.4	3.81e7	7.19e6	0.903	0.102	11500	20500	30.8	0.0222
PCM	2.75	4.63	0.224	0.204	24.0	39.0	10700	16900	**0.976**	0.0151	46.3	72.8	6.24	**9.16e-3**

both guaranteed to find the exact maximum clique. G*k*CM-HeuPatt uses a heuristic to help find the maximum clique faster but can find a suboptimal clique. This accounts for the small differences in accuracy between the G*k*CM-HeuPatt and the exact maximum clique methods. We also note that G*k*CM-HeuPatt runs significantly faster than either of the exact maximum clique methods while sacrificing minimal accuracy. Both PCM and the *k*-core methods ran faster than G*k*CM-HeuPatt, but the results for these methods are not accurate because their consistent sets contain outlier measurements. The only other statistic where either G*k*CM-Exact, ROBIN-MILP, or G*k*CM-HeuPatt was not the best method was the true positive rate (TPR). In this case, both PCM and ROBIN-*k*-core did better at including more inliers in their set of consistent measurements. However, we argue that since the presence of even a single outlier can have serious negative effects on the map quality, the false positive rate (FPR) is a better metric to evaluate each algorithm and G*k*CM-HeuPatt, G*k*CM-Exact, and ROBIN-MILP had the lowest FPR. The normalized χ^2 entry in the table tells us how well the state estimates fit the data that has been selected and ideally $\chi^2 < 3.84$ indicating that the data fits within a 95% confidence interval. The χ^2 values for G*k*CM-HeuPatt and ROBIN-MILP both meet this threshold while those for PCM and ROBIN-*k*-core are far larger, indicating inconsistencies in the measurements.

The inaccurate results for PCM were expected because PCM enforces group-2 consistency instead of group-4 consistency. For range-based SLAM, group-2 consistency is much less strict and only requires that two range measurements have a common intersection point. The group-4 consistency check used by the other methods is much more strict and requires that all four measurements share the same intersection point within a reasonable amount of uncertainty. Given that the ROBIN-*k*-core algorithm uses this stricter group-4 consistency check, we found it surprising that the resulting map was inaccurate. We knew that the data used to construct the *k*-core graph was the same data used to construct the generalized graphs used in the G*k*CM and ROBIN-MILP algorithms yet these other algorithms were able to reject virtually all of the outlier measurements. Further inspection of the *k*-core and the maximum cliques showed that the *k*-core included most, if not all, of the measurements found in the cliques produced by the exact maximum clique algorithms used in G*k*CM-Exact and ROBIN-MILP. This finding leads us to theorize that the embedding of a 4-uniform hypergraph into a 2-uniform hypergraph does not guarantee that a similar enough graph structure will exist for the maximum *k*-core to approximate the maximum clique of the original graph.

Additionally, we wished to know at what ratio of outliers to inliers the performance of G*k*CM begins to drop off. To measure this we simulated robot odometry for 100 poses and corrupted the measurements taken to a beacon with enough outliers to achieve a desired outlier ratio. We ran the set of measurements through G*k*CM and observed if the

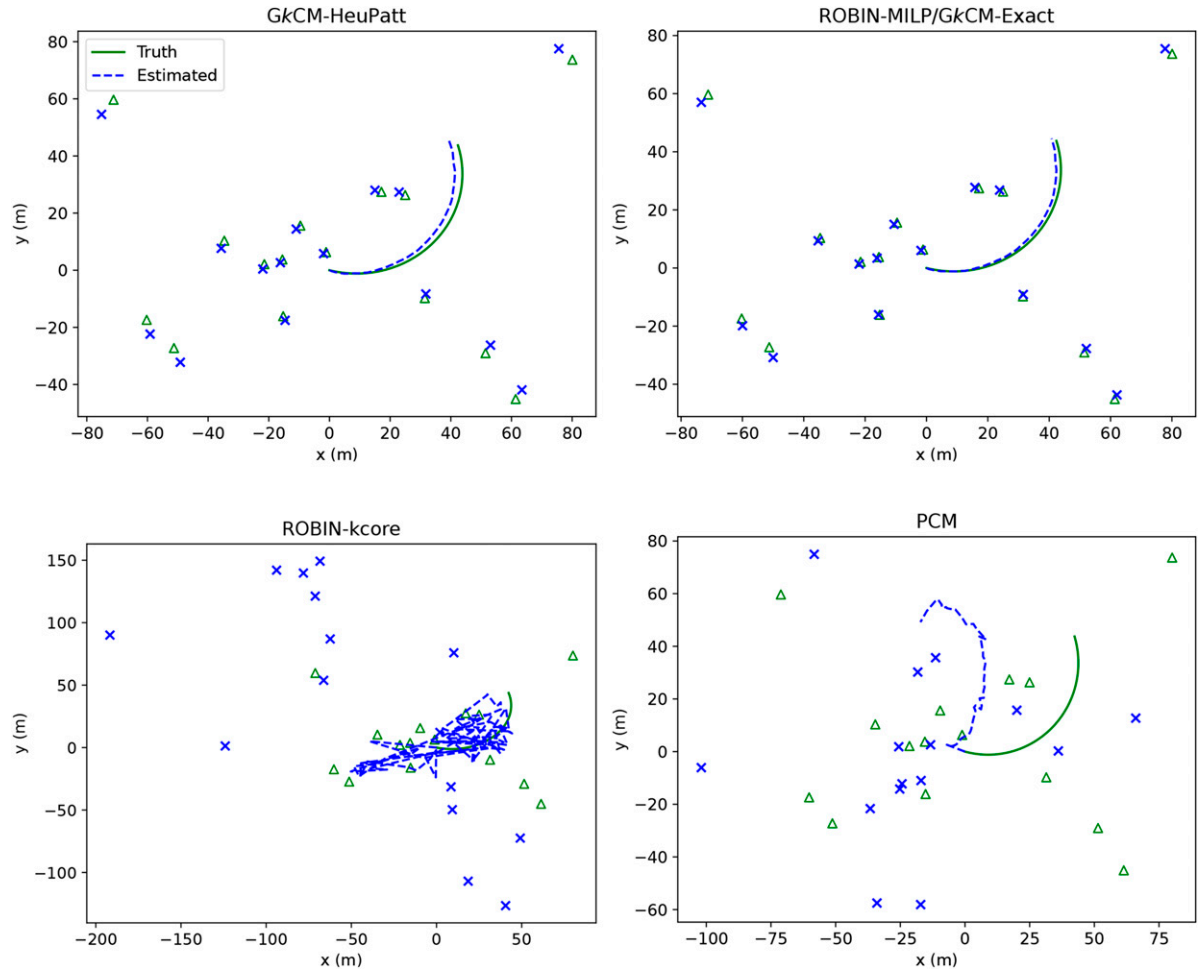


Figure 15. Results of a single run of the Monte Carlo experiment where 80% of the 75 measurements considered are outliers. The blue dashed lines and x denote the estimated trajectory and beacon locations. The green line and triangles denote the true trajectory and beacon locations.

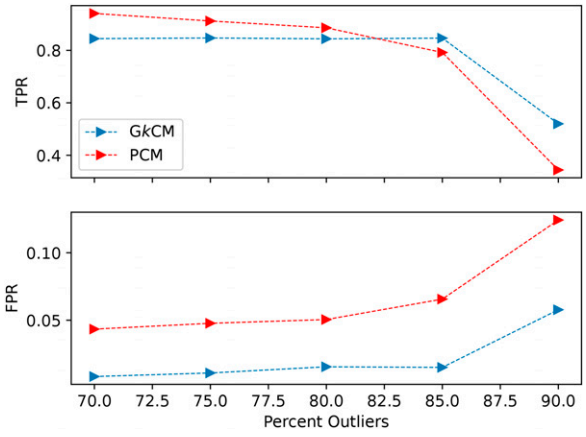


Figure 16. Results showing the normalized TPR ($TP/(TP + FN)$) and FPR $FP/(FP + TN)$ by varying the number of outliers for a fixed trajectory.

selected set of consistent measurements matched the ground truth. Using the same robot odometry, this was done with several different outlier percentages ranging from 70% to 90% outliers. The process was repeated for multiple trajectories and the true/false positive rates for each outlier percentage were recorded. For computation purposes, this was only run for G*k*CM-HeuPatt and results can be seen in Figure 16.

The figure shows that the true and false positive rates for G*k*CM-HeuPatt are fairly constant until about 85% of the measurements are outliers. For PCM however, the true positive rate decreases with the number of outliers, and the false positive rate increases. These results are expected because as more outliers are present, it is more likely that either an outlier clique will form or that an outlier measurement will intersect with the inlier set with a pairwise basis showing the need for group-*k* consistency.

12.2. Hardware experiment

This experiment evaluates the ability of G*k*CM to reject outliers in range-based SLAM using an underwater vehicle and acoustic ranging to beacons of an unknown location. We use the dataset collected by Olson et al. (2006), where they use SCGP to detect and remove outlier range measurements. The data collected uses four beacons placed

Table 4. Results for the χ^2 value, residual, and landmark error for the hardware experiment. Best results are in bold. DNF indicates the algorithm did not finish.

	χ^2	Residual	LM error (m)	Evaluation time (s)
G*k*CM-HeuPatt	**5.42e-3**	**41.1**	**4.73**	151
G*k*CM-exact	**5.42e-3**	**41.1**	**4.73**	55800
ROBIN-MILP	DNF	DNF	DNF	DNF
ROBIN-*k*-core	51.4	3.94e5	166	36.1
PCM	4.63	3.51e4	90.0	**22.3**
SCGP	0.456	3450	64.4	N/A

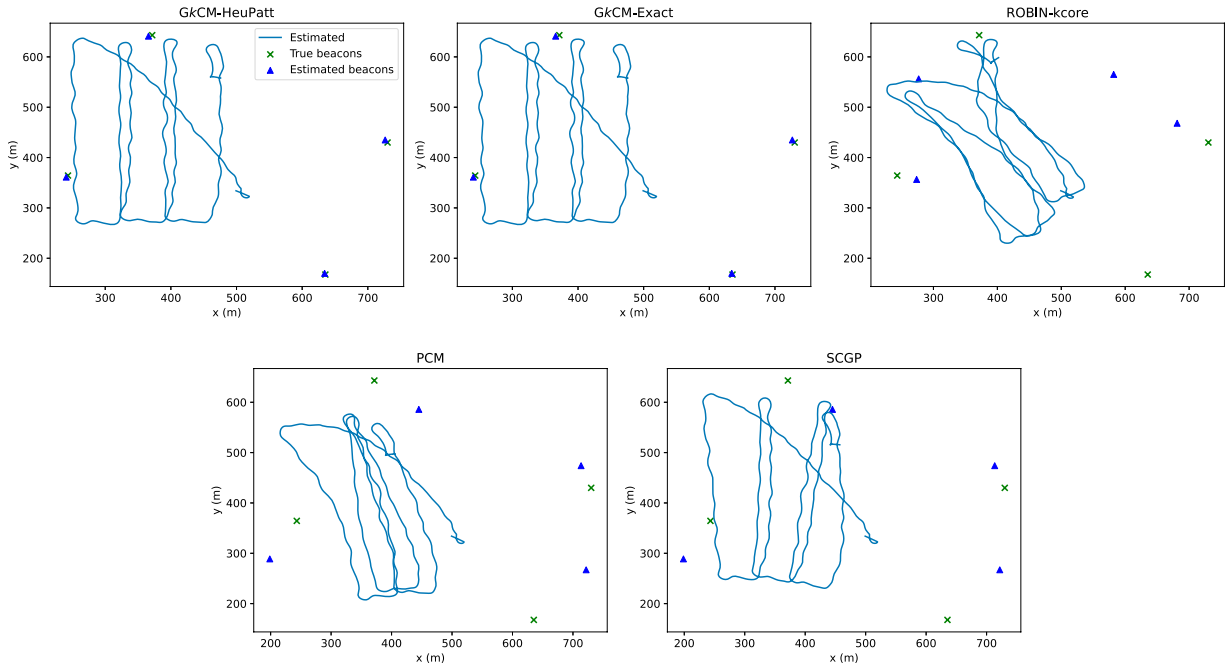


Figure 17. Example plots of the maps estimated by G*k*CM, PCM, MILP and *k*-core variants of ROBIN, and SCGP on the data collected in Olson et al. (2006). No truth data for the vehicle is available but the true beacon locations are denoted by a green x and the estimated locations by a blue triangle.

around the area of exploration. The underwater vehicle collects between 400 and 600 measurements to each beacon over the course of the experiment. Due to the exponential runtime of the maximum clique algorithms and the factorial increase in the number of consistency checks, we down-sample the number of measurements to 100 measurements to each beacon. Given that outliers are present in the data, we randomly selected 80 outlier measurements and 20 inlier measurements as classified by the SCGP algorithm. This was done to ensure that there is an inlier clique in the data while showing that we can reject outliers in high-outlier regimes.

We compare the results of G*k*CM with the results produced by SCGP, PCM, and with ROBIN using both the MILP and *k*-core algorithms. A summary of our results can be found in Table 4 and visual results can be seen in Figure 17. As can be seen, both G*k*CM-Exact and G*k*CM-HeuPatt attained the highest accuracy but G*k*CM-HeuPatt was able to find the maximum clique for each beacon in about 2.5 min while G*k*CM-Exact required about 15 h to find the same clique. PCM and ROBIN-*k*-core had faster evaluation times but produced much less accurate results. The evaluation time for SCGP is not listed because the set of consistent measurements selected by SCGP was provided in the dataset, but SCGP also produced less accurate results than the G*k*CM methods. Lastly, ROBIN-MILP was used to find the maximum clique but the program was terminated after running for 2 days. Had ROBIN-MILP been allowed to finish, we would expect the results to be identical to G*k*CM-Exact since these algorithms are guaranteed to find the same consistent set of measurements.

We note that the accuracy of PCM and SCGP varied with the subset of measurements used in the test. However, in all the tests we ran they never performed better than either G*k*CM method, although they occasionally achieved similar performance and a qualitative comparison between these algorithms and G*k*CM can be seen in Figure 17. The least accurate algorithm ended up being the ROBIN-*k*-core

technique. This is surprising considering that the algorithm uses a group-4 consistency metric and that PCM produced usable results in many instances by only enforcing group-2 consistency. We again compared the maximum k-core with the maximum cliques found using the exact maximum clique algorithms in GkCM-Exact and ROBIN-MILP and found that the maximum k-core contained the entire maximum clique produced by these other algorithms in every instance. This supports our theory that the k-core of an embedded generalized graph is not guaranteed to maintain a similar enough structure to provide a good approximation of the maximum clique of the original graph.

12.3. Data association

In this experiment, we remove the assumption that the correspondence between a range measurement and its beacon is known. To accomplish this, we modified both the exact and heuristic algorithms to track the n largest cliques where n is the number of beacons in the environment assuming the number of beacons is known. Since each clique corresponds to consistent measurements that belong to a unique beacon, we enforce the constraint that a measurement cannot appear in more than one clique.

This experiment was run on a short trajectory of 30 poses where five measurements were received at each pose (one to each beacon). As such 150 measurements are being considered by the GkCM algorithm. Results were averaged over 81 different trials. Visual results can be seen in Figure 18 while statistics are in Table 5. GkCM correctly identifies the five cliques corresponding to the different beacons and outperforms PCM in all the metrics. We do not provide a comparison with the other algorithms since they are unable to track multiple cliques simultaneously.

12.4. Tuning experiment

We showed that changing the threshold value, γ, did not significantly impact the results of the PCM algorithm. Due to enforced group consistency, as opposed to pairwise, we designed an experiment to test if GkCM has a similar property. We accomplished this by fixing a robot trajectory of 50 poses and the associated measurements and running GkCM multiple times with a different value for γ each time. The measurements contained 40 outliers that were generated as described previously. We averaged the χ^2 value and the true and false positive rates over multiple runs. Figure 19 shows how the above values vary with the consistency threshold for both GkCM and PCM.

As can be seen, GkCM performs better than PCM in both the normalized χ^2 and false positive rate, which is more important in our application than the true positive rate. The results indicate that the performance of GkCM varies more with the threshold γ than PCM, especially at low and high confidence thresholds. As such, we recommend that confidence values be used from the $50-90\%$ confidence range where performance was less variable with the confidence threshold.

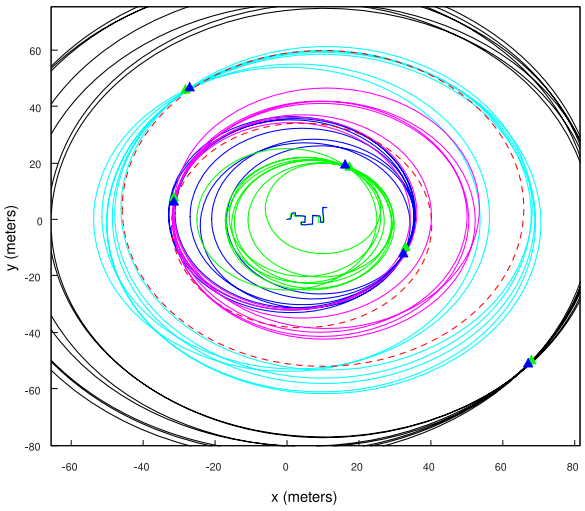


Figure 18. Results of GkCM for performing data association and outlier rejection. Each clique found is shown in a different color. Measurements labeled as outliers included in the maximum clique are red dashed lines.

12.5. Incremental update

In this last experiment, we evaluate the incremental heuristic described by Chang et al. (2021) since their experiments only evaluated the heuristic for a k-uniform hypergraph where $k = 2$. For this experiment, we generate a trajectory of 100 poses and measurements, and at each step, we evaluate how long both an incremental and batch update take. Updates include performing consistency checks and finding the maximum clique. We record the runtime for the graph size and average statistics over multiple runs. We plot the runtime against the size of the graph in Figure 20.

As can be seen, the incremental update with this heuristic provides similar benefits for GkCM as it does for PCM. On average, for a graph of 100 nodes with 80 outliers, it takes a batch solution over 40 s to solve for the maximum clique while it takes only 3 s for the incremental update. These findings validate the results presented by Chang et al. (2021) and also allow for GkCM to be run closer to real-time.

13. Multi-agent visual PGO evaluation

This section describes the experimental setup and the results obtained for each of the experiments evaluating the efficacy of GkCM for multi-agent visual PGO applications. In each experiment, we compare our results with the pairwise consistency approach in PCM and with both the MILP and k-core variants of ROBIN.

Table 5. Statistics for G*k*CM and PCM in Data Association experiment. Best results are in bold.

	Translational RMSE (m)		Rotational RMSE (rad)		Beacon error (m)		Residual		Inliers		χ^2		
	Avg	Std	Avg	Std	Avg	Std	Avg	Std	TPR	FPR	Avg	Std	Median
G*k*CM-HeuPatt	**0.626**	**0.565**	**0.247**	**0.0629**	**34.1**	52.1	**72.1**	**88.7**	0.837	**0.008**	**1.05**	**1.26**	**0.306**
PCM	3.15	3.61	0.452	0.222	40.3	**51.0**	1010	1040	**0.95**	0.0166	13.9	14.2	29.3

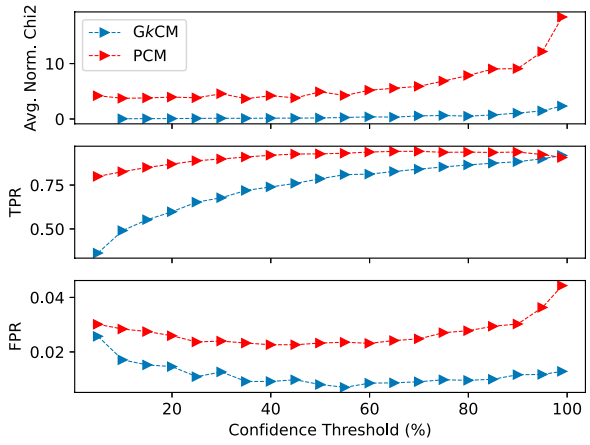


Figure 19. Results showing the normalized χ^2 value, TPR, and FPR by varying the consistency threshold value, γ, for a fixed trajectory.

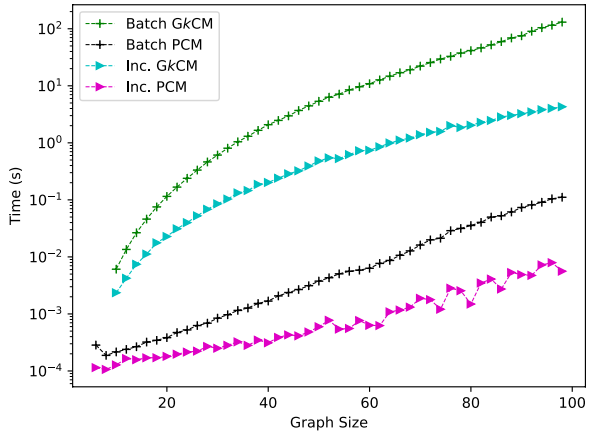


Figure 20. Timing data for both batch and incremental updates for G*k*CM and PCM. This includes the time to perform the relevant consistency checks and the new maximum clique. Note the log-scale on the vertical axis.

13.1. Simulation results

We first evaluated our algorithm in simulation. We randomly initialized two agents in a predetermined area and let the agents move about in a Manhattan world scenario. After generating the path, we searched for poses on each robot within half a meter of each other to generate an inter-robot measurement. For computational purposes, we selected 100 measurements to be used, and if 100 measurements were not generated we reran the simulation until 100 measurements were present. We then ran the G*k*CM, PCM, and ROBIN (using both the MILP and *k*-core clique solvers) to find the largest inlier set. Statistics such as the TPR, FPR, RMSE, and the χ^2 value of the resulting PGO routine were recorded and averaged over 100 runs. To initialize the relative pose, we use chordal initialization (Carlone et al., 2015) to generate rotation estimates. To initialize the translation, we picked two measurements from the set of inlier measurements and estimated the scale by solving equation (12) and applying the scale to the unit vector created by the azimuth and elevation angles.

As can be seen from the results, all of the methods performed fairly equally across all the metrics except for evaluation time. PCM ran the fastest followed closely by G*k*CM-HeuPatt. ROBIN-*k*-core and G*k*CM-Exact were next with all four of these methods finding the maximum clique in less than a second. The slowest algorithm was the ROBIN-MILP which required almost a minute on average to find the maximum clique. Differences in the residual and normalized χ^2 statistics were also observed and we believe the large differences in these fields are a result of different initial conditions. We could not guarantee that the same measurements were used to initialize the scale on the translation vector between the two agents since the inlier cliques were not guaranteed to be the same. Often the cliques had five or six measurements in common with one or two that varied between them. Due to the nature of least-squares methods, this caused the scale to vary and resulted in the maps converging to slightly different solutions. We note that this is a problem with the initialization of the graph, which is not the focus of this research. Each of these methods was able to reject nearly all of the outlier measurements, noting that PCM and ROBIN-*k*-core also found more inlier measurements in their consistent sets. Visual results for a single trial can be seen in Figure 21 while statistics are in Table 6. Note that ROBIN-MILP and G*k*CM-Exact share a figure because they produce the exact same results.

We note that in this experiment the ROBIN-*k*-core algorithm performed nearly as accurately as the G*k*CM and ROBIN-MILP algorithms. As with the range-based SLAM experiments we compared the maximum *k*-core with the maximum cliques produced by G*k*CM-Exact and ROBIN-MILP and found that the maximum *k*-core contained the entire maximum clique produced by these other algorithms. There were usually a few other

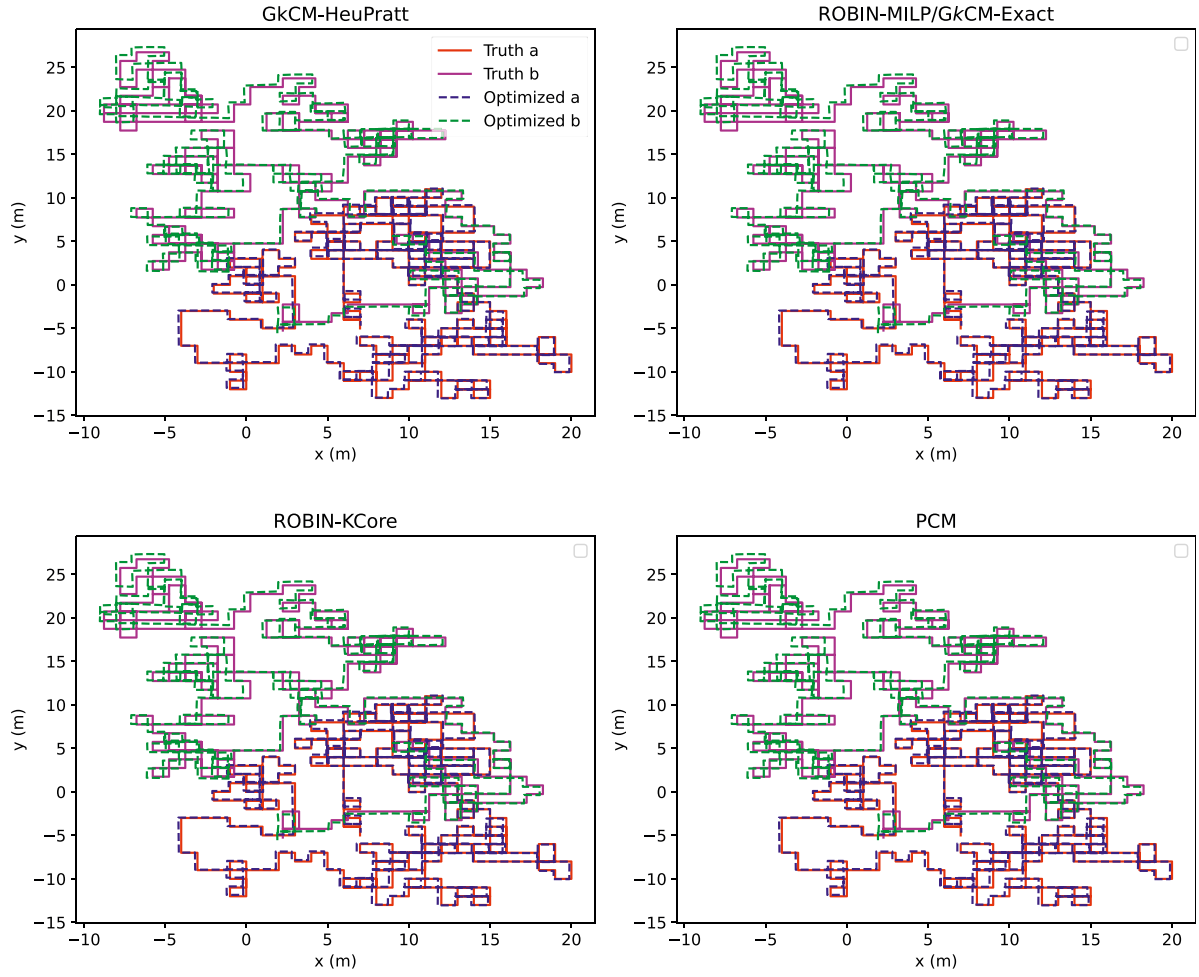


Figure 21. Results for each of the algorithms in the multi-agent visual PGO simulation. The red and blue solid lines denote the true trajectories of each vehicle while the dashed lines denote the estimated trajectories.

Table 6. Statistics comparing G*k*CM to other methods in the multi-agent simulation experiment. Best results are in bold.

	Trans. error (m)		Rot. error (rad)		Residual		Inliers		χ^2			Evaluation time (ms)	
	Avg	Std	Avg	Std	Avg	Std	TPR	FPR	Avg	Std	Median	Avg	Std
G*k*CM-HeuPatt	**1.26**	**1.87**	9.29e-3	**6.43e-3**	**8920**	**40200**	0.741	**2.22e-4**	**2.95**	**13.3**	1.07e-2	0.701	**0.540**
G*k*CM-Exact	1.34	2.18	9.945e-3	6.87e-3	11400	48800	0.741	**2.22e-4**	3.77	16.1	**1.04e-2**	6.08	5.41
Robin-MILP	1.34	2.18	9.945e-3	6.87e-3	11400	48800	0.741	**2.22e-4**	3.77	16.1	**1.04e-2**	55600	23600
Robin-*k*-core	1.41	2.05	**8.55e-3**	8.23e-3	11900	41300	**0.991**	**2.22e-4**	3.91	13.6	1.51e-2	5.05	0.863
PCM	1.37	2.00	8.61e-3	8.20e-3	11300	40700	0.985	**2.22e-4**	3.73	13.4	1.39e-2	**0.428**	0.727

measurements included in the *k*-core and these measurements were usually inlier measurements that the other algorithms had missed. This experiment shows that it is possible that embedding a generalized graph into a 2-uniform graph can maintain a similar enough graph structure so that the maximum *k*-core approximates the maximum clique of the original graph. Exploring conditions when this may be true is beyond the scope of this paper but is an interesting avenue for future work.

13.2. Hardware results

Having shown that the G*k*CM algorithm is effective at rejecting outliers in a simulation environment, we now seek to validate our algorithm using hardware data. In this experiment, we used the NCLT dataset presented by Carlevaris-Bianco et al. (2015) because it provides data of a robot exploring the same area over multiple sessions, and has ground truth data for large portions of the dataset

measured using RTK GPS techniques. We selected two sessions (Feb. 2, 2012, and Feb. 4, 2012) that occurred in the same season and around the same time of day to facilitate place recognition, which was done using the OpenFABMap library developed by Glover et al. (2012). We trained OpenFABMap using data collected in one session and created matches to places in the second session. Due to the size of the dataset, we elected to use only one of the cameras onboard the robot and used every 50th image.

With this setup, we generated 267 matches of which 25 were identified as true matches while the others were labeled as outliers due to perceptual aliasing. We estimated the relative pose between the two images by decomposing the essential matrix using OpenCV, which was then refined through bundle adjustment using GTSAM Dellaert (2012). From this relative pose, we generated the measurement in equation (9) using equation (13) and transformed the covariance of the relative pose to the measurement covariance using the Jacobian of equation (13). Once the measurements were generated, we then checked the consistency of all the measurements and found the maximum clique using each of the methods discussed previously. The measurements found in the maximum clique, along with the odometry data for both sessions, were then used in a multi-agent pose-graph optimization algorithm to combine the maps made during each session.

A visual representation of our results can be found in Figure 22 while the statistics are in Table 7. As can be seen, both the G*k*CM methods can successfully filter out the outlier measurements and fuse the maps and have the best performance across all metrics except evaluation time in Table 7. The methods produce identical results because they found the same maximum clique, however, G*k*CM-HeuPatt was able to find the maximum clique much faster than G*k*CM-Exact. The former required almost 6 min while the latter required about five and a half days. ROBIN-MILP was terminated before it finished finding the maximum clique, but because it is guaranteed to find the maximum clique it would produce results equivalent to G*k*CM-Exact. Both the *k*-core and PCM methods found their consistent set of measurements faster, but their sets contained outlier measurements that corrupted their estimated results and rendered the map unusable. As noted before, the primary shortcomings of these methods are that PCM enforces consistency on a pairwise basis, and ROBIN-*k*-core operates over a 2-uniform hypergraph instead of a 3-uniform hypergraph.

The results produced by the G*k*CM-HeuPatt and G*k*CM-Exact methods successfully merge the maps with some error in the offset in the origins of the two maps. The map of the second agent (shown by the greed dotted line in Figure 22) should lie nearly on top of the blue dotted line but often is

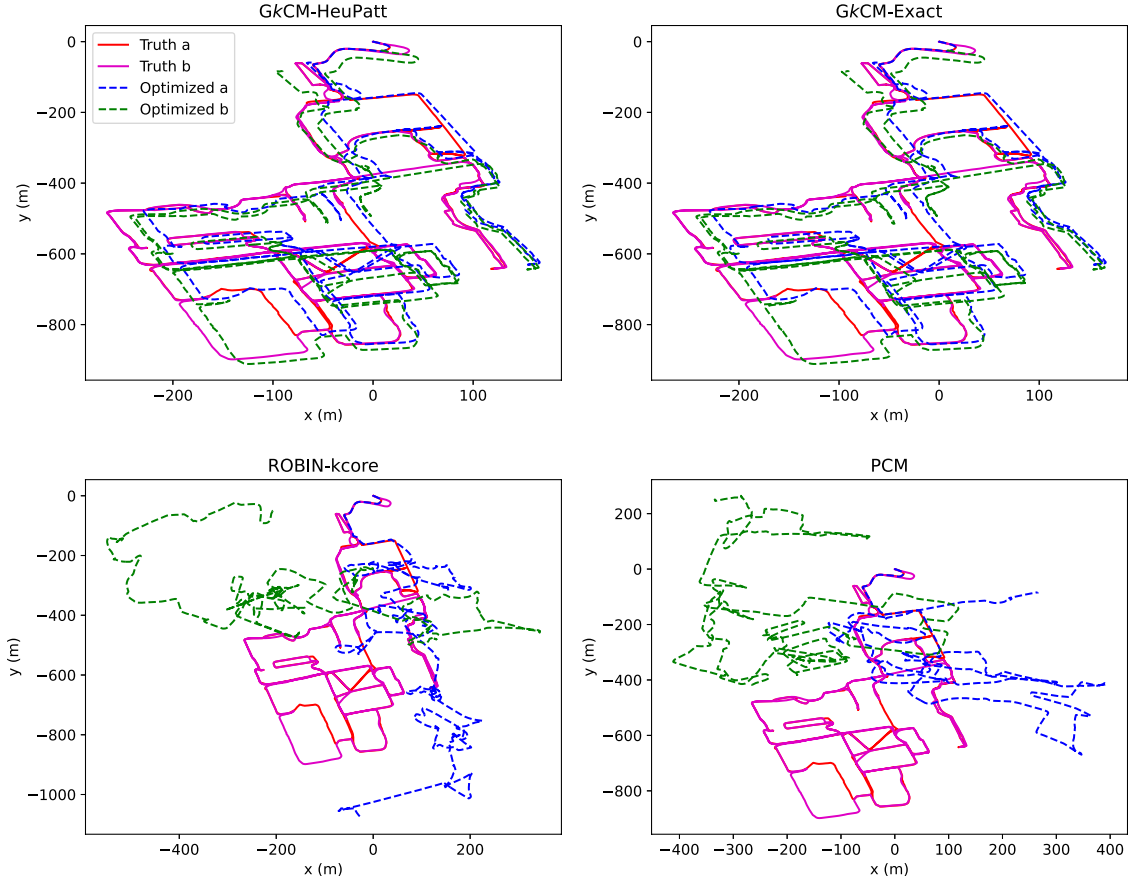


Figure 22. Results for the hardware experiment. The solid red and purple lines are the true robot trajectories generated using GPS, Lidar, and odometry. The dotted lines are the estimated trajectories using odometry and inter-vehicle measurements from a camera.

Table 7. Numerical results for the multi-agent hardware experiment. Best results are in bold. DNF indicates the algorithm did not finish.

	Trans. error (m)	Rot. error (rad)	Residual	χ^2	Evaluation time (s)
G*k*CM-HeuPatt	**25.3**	**4.27e-2**	**1.99e4**	**0.125**	348
G*k*CM-Exact	**25.3**	**4.27e-2**	**1.99e4**	**0.125**	4.75e5
ROBIN-MILP	DNF	DNF	DNF	DNF	DNF
ROBIN-*k*-core	311	0.985	8.84e6	55.4	**3.32**
PCM	354.5	0.692	2.24e5	1.40	41.4

shifted down and to the left. We believe this level of error to be acceptable for several reasons. The first is that much less information was used when fusing the maps in our estimated results than was used in generating the ground truth. The ground truth data fused RTK-GPS and Lidar scans with the odometry from all 12 sessions where data was collected. Lidar scans were aligned to generate constraints both in a single session and between sessions. This allowed many full-degree-of-freedom constraints to be generated within each session and between the different sessions. Comparatively, our solution used only a subset of the visual information provided and only used this information to generate constraints between the two sessions. No loop closure constraints were generated when the robot visited a location it had visited before in the same session. Additionally, using only a single camera increases the difficulty of the problem because the scale is not observable in the inter-session constraints. This problem does not arise when comparing Lidar scans.

Lastly, it was not anticipated that the simulation experiments and hardware experiments produced such different results. In the simulation, all methods had a fairly similar level of accuracy, but ROBIN-*k*-core and PCM were less accurate in the hardware experiment. We attribute this to the difference in the length of the trajectories used in the simulation and hardware experiments. The simulated experiments had a trajectory consisting of about 500 poses per robot while the two sessions in the NCLT dataset had over 23,000 and 29,000 poses respectively. This means that the distance between inter-vehicle measurements in the NCLT dataset is longer and that the accumulated relative pose uncertainty in the odometry between measurements was larger. This larger uncertainty caused more consistency checks to pass and resulted in a denser consistency graph. The result of this is that more measurements were labeled as consistent when evaluating the NCLT dataset. PCM did not have the benefit of the direction information to filter out more outlier measurements like G*k*CM. ROBIN-*k*-core had the benefit of the direction information but the denser graph but could not approximate the maximum clique of the group-3 consistency graph.

14. Conclusion

In this paper, we presented a unification of the theory of consistency and its applications to outlier rejection in multi-agent robotic systems. We first present the PCM algorithm and show that it can successfully reject outlier loop closures in multi-agent systems when other common techniques fail to do so. We then generalize the PCM algorithm and introduce the G*k*CM algorithm that enforces consistency in groups of k measurements. This work introduced two algorithms that can find the maximum clique of a generalized graph that are used to find the largest set of consistent measurements. We provide a comparison of our maximum-clique algorithms with another maximum-clique algorithm over generalized graphs and show that our algorithms can find the maximum clique faster. Enforcing group consistency enables G*k*CM to enforce consistency for low-degree-of-freedom measurements and we showed that G*k*CM outperforms techniques that enforce pairwise consistency in both range-based SLAM and multi-agent visual PGO applications. Lastly, we presented techniques that alleviate the exponential nature of both evaluating consistency and finding the maximum clique, and released an open-source implementation of our library for future use by the research community.

Acknowledgments

We would like to thank Allan Papalia and John Leonard at the Marine Robotics Group for providing the range-based SLAM dataset for us to use in evaluating our algorithms.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The authors disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was funded by the Office of Naval Research award N00014-21-1-2435, the NAVSEA Panama City NEEC Grant N00174-23-1-0005, and the Center for Autonomous Air Mobitlity and Sensing (CAAMS), a National Science Foundation Industry University Cooperative Research Center (IUCRC) under NSF award No. IIP-2139551 along with significant contributions from the AFRL Munitions Directorate and other CAAMS industry members.

ORCID iD

Brendon Forsgren https://orcid.org/0000-0003-0440-2608

Notes

1. © 2018 IEEE. Reprinted, with permission, from Pairwise Consistent Measurement Set Maximization for Robust Multi-Robot Map Merging. 2018 IEEE International Conference on Robotics and Automation (ICRA).
2. © 2022 IEEE. Reprinted, with permission, from Group-*k* consistent measurement set maximization for robust outlier detection. 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

References

Agarwal P, Tipaldi GD, Spinello L, et al. (2013) Robust map optimization using dynamic covariance scaling. In: Proceedings of the IEEE international conference on robotics and automation. Karlsruhe, Germany, pp. 62–69. IEEE.

Agarwal S, Mierle K and Team TCS (2023) Ceres solver. URL https://github.com/ceres-solver/ceres-solver.

Bollobás B (1965) On generalized graphs. *Acta Mathematica Academiae Scientiarum Hungaricae* 16(3-4): 447–452.

Cadena C, Carlone L, Carrillo H, et al. (2016) Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. *IEEE Transactions on Robotics* 32(6): 1309–1332.

Carlevaris-Bianco N, Ushani AK and Eustice RM (2015) University of Michigan North Campus long-term vision and Lidar dataset. *The International Journal of Robotics Research* 35(9): 1023–1035.

Carlone L, Censi A and Dellaert F (2014) Selecting good measurements via ℓ-1 relaxation: a convex approach for robust estimation over graphs. In: 2014 IEEE/RSJ international conference on intelligent robots and systems, pp. 2667–2674. IEEE.

Carlone L, Tron R, Daniilidis K, et al. (2015) Initialization techniques for 3D SLAM: a survey on rotation estimation and its use in pose graph optimization. In: Proceedings of the IEEE international conference on robotics and automation, pp. 4597–4604. IEEE.

Chang Y, Tian Y, How JP, et al. (2021) Kimera-multi: a system for distributed multi-robot metric-semantic simultaneous localization and mapping. In: Proceedings of the IEEE international conference on robotics and automation, pp. 11210–11218. IEEE.

Dellaert F (2012) *Factor Graphs and GTSAM: A Hands-On Introduction*. Technical report, Georgia Institute of Technology.

Dong J, Nelson E, Indelman V, et al. (2015) Distributed real-time cooperative localization and mapping using an uncertainty-aware expectation maximization approach. In: Proceedings of the IEEE International conference on robotics and automation. Seattle, Washington, USA, pp. 5807–5814. IEEE.

Feige U, Goldwasser S, Lovász L, et al. (1991) Approximating clique is almost NP-complete. In: Proceedings of the IEEE annual symposium on foundations of computer science. Kazimierz Dolny, Poland, pp. 2–12. IEEE.

Forsgren B, Vasudevan R, Kaess M, et al. (2022) Group-k consistent measurement set maximization for robust outlier detection. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, pp. 4849–4856. IEEE.

Gentner M, Murali PK and Kaboli M (2023) *GMCR: graph-Based Maximum Consensus Estimation for Point Cloud Registration*. arXiv preprint arXiv:2303.04032.

Glover A, Maddern W, Warren M, et al. (2012) OpenFABMAP: an open-source toolbox for appearance-based loop closure detection. In: Proceedings of the IEEE international conference on robotics and automation, pp. 4730–4735. IEEE.

Gurobi O (2017) Gurobi optimizer. https://www.gurobi.com/.

Hartley R and Zisserman A (2003) *Multiple View Geometry in Computer Vision*. Cambridge, UK: Cambridge University Press.

Hu E and Sun L (2023) VODRAC: efficient and robust correspondence-based point cloud registration with extreme outlier ratios. *Journal of King Saud University - Computer and Information Sciences* 35(1): 38–55.

Kaess M and Dellaert F (2009) Covariance recovery from a square root information matrix for data association. *Robotics and Autonomous Systems* 57(12): 1198–1210.

Kaess M, Ranganathan A and Dellaert F (2008) iSAM: incremental smoothing and mapping. *IEEE Transactions on Robotics* 24(6): 1365–1378.

Kaess M, Johannsson H, Roberts R, et al. (2012) iSAM2: incremental smoothing and mapping using the Bayes tree. *The International Journal of Robotics Research* 31(2): 216–235.

Kim B, Kaess M, Fletcher L, et al. (2010) Multiple relative pose graphs for robust cooperative mapping. In: Proceedings of the IEEE international conference on robotics and automation, Anchorage, Alaska, USA, pp. 3185–3192. IEEE.

Kümmerle R, Grisetti G, Strasdat H, et al. (2011) g2o: a general framework for graph optimization. In: Proceedings of the IEEE international conference on robotics and automation, Shanghai, China, pp. 3607–3613. IEEE.

Lusk PC and How JP (2022) Global data association for SLAM with 3D Grassmannian manifold objects. In: Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, 4463–4470. IEEE.

Lusk PC, Fathian K and How JP (2021) Clipper: a graph-theoretic framework for robust data association. In: Proceedings of the IEEE international conference on robotics and automation, pp. 13828–13834. IEEE.

Mangelson JG, Dominic D, Eustice RM, et al. (2018) Pairwise consistent measurement set maximization for robust multi-robot map merging. In: Proceedings of the IEEE international conference on robotics and automation, pp. 2916–2923. IEEE.

Mangelson JG, Ghaffari M, Vasudevan R, et al. (2020) Characterizing the uncertainty of jointly distributed poses in the Lie algebra. *IEEE Transactions on Robotics* 36(5): 1371–1388.

Matula DW and Beck LL (1983) Smallest-last ordering and clustering and graph coloring algorithms. *Journal of the ACM (JACM)* 30(3): 417–427.

Neira J and Tardós JD (2001) Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation* 17(6): 890–897.

Olson E (2009) Recognizing places using spectrally clustered local matches. *Robotics and Autonomous Systems* 57(12): 1157–1172.

Olson E and Agarwal P (2013) Inference on networks of mixtures for robust robot mapping. *The International Journal of Robotics Research* 32(7): 826–840.

Olson E, Walter MR, Teller SJ, et al. (2005) Single-cluster spectral graph partitioning for robotics applications. In: Proceedings of the robotics: science & systems conference. Cambridge, Massachusetts, USA.

Olson E, Leonard JJ and Teller S (2006) Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering* 31(4): 949–958.

Pattabiraman B, Patwary MMA, Gebremedhin AH, et al. (2015) Fast algorithms for the maximum clique problem on massive graphs with applications to overlapping community detection. *Internet Mathematics* 11(4-5): 421–448. DOI: 10.1080/15427951.2014.986778.

Pfingsthorn M and Birk A (2016) Generalized graph SLAM: solving local and global ambiguities through multimodal and hyperedge constraints. *The International Journal of Robotics Research* 35(6): 601–630.

Segal A, Haehnel D and Thrun S (2009) Generalized-ICP. In: Proceedings of the robotics: science & systems conference. Seattle, Washington, USA.

Shi J, Yang H and Carlone L (2021) ROBIN: a graph-theoretic approach to reject outliers in robust estimation using invariants. In: 2021 IEEE International conference on robotics and automation (ICRA), pp. 13820–13827. IEEE.

Shi J, Yang H and Carlone L (2022) *Optimal and Robust Category-Level Perception: Object Pose and Shape Estimation from 2D and 3D Semantic Keypoints*. arXiv preprint arXiv: 2206.12498.

Smith R, Self M and Cheeseman P (1990) Estimating uncertain spatial relationships in robotics. In: Autonomous Robots, pp. 167–193.

Sun L (2021) RANSIC: fast and highly robust estimation for rotation search and point cloud registration using invariant compatibility. *IEEE Robotics and Automation Letters* 7(1): 143–150.

Sünderhauf N and Protzel P (2012) Towards a robust back-end for pose graph SLAM. In: Proceedings of the IEEE international conference on robotics and automation. St. Paul, Minnesota, USA, pp. 1254–1261. IEEE.

Wu Q and Hao JK (2015) A review on algorithms for maximum clique problems. *European Journal of Operational Research* 242(3): 693–709.

Yang H and Carlone L (2022) Certifiably optimal outlier-robust geometric perception: semidefinite relaxations and scalable global optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45(3): 2816–2834.

Yang H, Antonante P, Tzoumas V, et al. (2020a) Graduated non-convexity for robust spatial perception: from non-minimal solvers to global outlier rejection. *IEEE Robotics and Automation Letters* 5(2): 1127–1134.

Yang H, Shi J and Carlone L (2020b) Teaser: fast and certifiable point cloud registration. *IEEE Transactions on Robotics* 37(2): 314–333.

Zhang X, Yang J, Zhang S, et al. (2023) 3d registration with maximal cliques. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 17745–17754. IEEE.

Zhou Y (2011) A closed-form algorithm for the least-squares trilateration problem. *Robotica* 29(3): 375–389.

Zuckerman D (2006) Linear degree extractors and the inapproximability of max clique and chromatic number. In: Proceedings of the ACM annual symposium on the theory of computing. Seattle, Washington, USA, pp. 681–690. ACM.