DDF-SAM 2.0: Consistent Distributed Smoothing and Mapping

Alexander Cunningham, Vadim Indelman and Frank Dellaert

***Abstract*— This paper presents an consistent decentralized data fusion approach for robust multi-robot SLAM in dangerous, unknown environments. The DDF-SAM 2.0 approach extends our previous work by combining local and neighborhood information in a single, consistent augmented local map, without the overly conservative approach to avoiding information double-counting in the previous DDF-SAM algorithm. We introduce the anti-factor as a means to subtract information in graphical SLAM systems, and illustrate its use to both replace information in an incremental solver and to cancel out neighborhood information from shared summarized maps. This paper presents and compares three summarization techniques, with two exact approaches and an approximation. We evaluated the proposed system in a synthetic example and show the augmented local system and the associated summarization technique do not double-count information, while keeping performance tractable.**

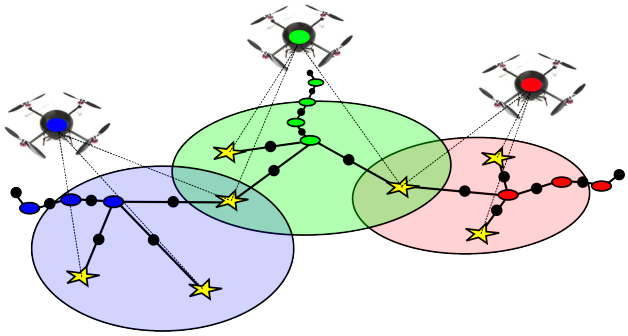

Fig. 1. Example multi-robot SLAM scenario, showing three robots observing landmarks (stars), with some landmarks in common. Also illustrated is factor graph with measurements (edges with filled black circles) and the trajectory for each platform.

I. Introduction

A key enabling technology for robust multi-robot teams capable of operation in challenging environments is a perception system capable of fusing information in a decentralized manner. The SLAM community has produced substantial improvements in the quality of robot mapping techniques, enabling efficient, real-time mapping using a variety of sensors in single-robot cases. This paper addresses the application of these state-of-the-art SLAM techniques to fleets of small, cheap robots to enable robust multi-robot perception, extending the sensor horizon of any individual robot. Fig. 1 shows a simple three-robot mapping scenario with commonly observed map landmarks motivating this approach.

In our previous work on decentralized robot perception [1], [2], we introduced DDF-SAM (denoted as DDF-SAM 1.0 in this paper), a SLAM paradigm extending Smoothing and Mapping (SAM) [3] techniques to the Decentralized Data Fusion (DDF) problem [4] by robots performing local SLAM and sharing a subset of map variables with neighbors in the form of *summarized maps*. The key component of the approach is the sharing of summarized densities between robots, allowing for application-specific choices of transmitted information that users can throttle to match the available communication bandwidth between platforms.

However, DDF-SAM 1.0 has two key shortcomings: 1) an overly conservative approach for avoiding information double-counting, and 2) reliance on a batch marginalization approach for map summarization. The method employed for avoiding information double-counting enforced a strict separation between a robot's local map and it's neighborhood map comprised of data from neighboring robots - an approach that avoids double-counting at the expense of each robot maintaining two separate, but incomplete maps of its environment. Furthermore, the batch summarization technique used to marginalize non-shared variables from the local map performs a computationally expensive and separate factorization of the entire local system, which increases in complexity over time and limits scalability. We examine alternate approaches for summarization, as well as the impact these approaches have on performance in the full system.

To address these shortcomings, we introduce DDF-SAM 2.0 with the following contributions towards effective decentralized perception: 1) the *augmented local system* as a replacement for the separate local and neighborhood maps maintained in DDF-SAM 1.0 to provide a single, consistent map on each robot blending local and neighborhood information, 2) the *anti-factor* as a tool to avoid double-counting of neighborhood information by down-dating estimates, and 3) the derivation and evaluation of three summarization techniques applicable to incremental solving techniques, using both exact marginalization and approximations.

II. Related Work

While this paper focuses on a graphical-model formulation of the multi-robot SLAM problem, there is a long history of related work in distributed inference. As is common throughout much of the mapping and estimation community, many techniques employ filtering approaches to perform inference. Roumeliotis et al. [5] consider an extended Kalman filter (EKF) framework and perform a decentralized calculation of the augmented covariance matrix between all the robots in the group assuming relative pose measurements. Nerurkar

Authors are with the Georgia Institute of Technology, Atlanta, GA contactable at acunning, indelman, frank@cc.gatech.edu. This work was partially funded through the Micro Autonomous Systems and Technology (MAST) Alliance, with sponsorship from Army Research Labs (ARL).

et al. [6] develop a MAP estimator with a distributed data-allocation scheme to simultaneously process and update local data while incorporating inter-robot relative measurements. Bahr et al. [7] introduced a method for consistent cooperative localization with a bank of filters for tracking the origins of measurements to prevent double-counting. Indelman et al. [8] used a graph-based approach to calculate the correlation terms for consistent information fusion in the EKF framework. Another method for consistent information fusion is covariance intersection [9], previously applied to single-robot SLAM [10] with straightforward extensions to multi-robot SLAM. Thrun et. al, presented an extension of the sparse extended information filter for multi-robot scenarios [11], [12], which actively removed information to ensure sparseness at the cost of approximation.

In addition to filtering approaches for multi-robot SLAM, there are a variety of centralized graph-SLAM techniques, such as C-SAM [13], which combines observations from multiple robots in a single factor graph. Centralized graph-SLAM techniques allow for easier extension of single-robot SLAM algorithms, such as iSAM [14] applied to multi-robot mapping via pose-graphs [15], but centralizing inference makes scalability to large robot systems with unreliable communication difficult, especially when performing inference online. Bailey et al. [16] proposed a cooperative localization technique using relative range-bearing measurements between the robots, extending previous work in the context of airborne SLAM [17]. Several works fuse information from multiple local maps into a global map without updating the local maps with the global solution [18], [19].

III. Problem Formulation

The goal for decentralized SLAM is to extend the sensor horizon of individual robots in a decentralized robot team with the inclusion of *neighborhood* information. The primary characteristic of a decentralized system is that individual nodes communicate with only those nodes in their local area, or neighborhood. In a SLAM context, this means there is no global map at a central server, only neighborhood estimates on each robot. Let N_r be the set of robots that communicate with r, such that $r \notin N_r$, and let N_r' be an augmented neighborhood $N_r' \triangleq \{N_r, r\}$. In practical scenarios, this is a time-varying set due to intermittent communication, resulting in a dynamic network. Robust system operation within this dynamic network topology and under limited computation comprise the key requirements for a DDF algorithm.

One key challenge in DDF systems is avoiding overconfidence due to *double-counting* measurements as information propagates through the network. Consider the following example with three robots A, B, and C: robot A shares local information with robot B, then robot B incorporates this information into its own estimate and shares information with robot C. Robot C incorporates the information from robot B (which contains information from robot A), and finally shares its new estimate with robot A. By completing this cycle in the network, robot A has now received information derived from both its own local estimates (as was shared with B), and from local measurements by robots B and C. If robot A treats the message from robot C as independent of its own measurements, it will double-count its local measurements, leading to overconfidence in its estimate. It is possible to avoid this problem by enforcing non-cyclic constraints on network topology [20], but this reduces the resilience of the system to communication and node failures.

In addition to the typical poses X and landmarks L of a typical SLAM formulation, we additionally denote each set to include the observing robot r. Let X^r denote poses $\{x_i^r\}$, L^r be the set of observed fixed environmental landmarks $\{l_j^r\}$ by the robot r. For convenience, let $\Theta^r \triangleq \{X^r, L^r\}$. As an extension of this superscript notation for the associated robot, we denote multi-robot quantities replacing a single robot r with a set, such as N_r', such that $\Theta^{N_r'}$ represents the union of all θ within r and in neighboring maps. Denote the set $\{z_m^r\}$ of sensor observations by robot by as Z^r, where for each z_m^r there exists a sparse generative prediction model $h(\Theta_m^r) = \bar{z}_m^r$, where Θ_m^r is exactly the subset of involved involved variables (such as a single pose and landmark for a single landmark sighting) and $\bar{z}_m^r$ is the predicted measurement. As is typical in SLAM formulations, we assume measurements are corrupted by zero-mean Gaussian noise. This paper assumes known data associations between measurements and landmarks are for both single robot platforms and between robot platforms; for a treatments of data associations between platforms, see [2], [21].

Individually, each robot r in a team of robots is capable of performing full SLAM locally, which we can augment with map information from robots in the neighborhood N_r of r to a) improve the certainty on the landmarks L^rdirectly observed by r and b) incorporate neighborhood landmarks L^{N_r} into the map of r.

The following sections describe the core representations of probabilistic inference for SLAM, introducing factor graphs, the variable elimination algorithm, and the Bayes tree as a partially solved form of a factor graph after elimination. For a more detailed treatment, see [3], [22].

A. Inference in SLAM

As in a single-robot SLAM scenario, we use a *factor graph* to represent the inference problem. Let $\phi(\Theta)$ be a non-negative, real-valued function of these variables that factors over subsets $\Theta_i \subseteq \Theta$, as in

$$\phi(\Theta^r) = \prod_i \phi_i^r(\Theta_i^r)$$

then a factor graph is a bipartite graph that expresses this factorization [24]. Formally, the factor graph $G \triangleq (\Phi, \Theta, \mathcal{E})$ has two types of nodes: *factors* $\phi_i \in \Phi$ and *variables* $\theta_j \in \Theta$. An edge $e_{ij} \in \mathcal{E}$ signifies that a variable $\theta_j \in \Theta_i$.

For single robot we calculate the maximum a posteriori estimate over the full likelihood $p(\Theta^r|Z^r)$ using the factorization, which each factor is parameterized on its corresponding measurement $\phi_i^r(\Theta_i^r; z_i^r) \propto p(\Theta_i^r|z_i^r)$. We

cast inference as the optimization problem

$$\Theta^{r*} \triangleq \arg\max_{\Theta^r} \prod_i \phi_i^r(\Theta_i^r; z_i^r)$$

and then reformulate the cost function in negative log-likelihood form to generate an additive cost function, which allows us to treat the problem as a large scale minimization problem with loss function

$$L\left(\Theta^r\right) \triangleq \sum_i \phi_i^r(\Theta_i^r; z_i^r) \propto -\log \prod_i p\left(\Theta_i^r | z_i^r\right).$$

With this loss function we can solve for the full solution Θ^r in batch using non-linear optimization techniques, such as Levenberg–Marquardt, by successively linearizing and applying the delta computed from solving a linear system. Each iteration of the non-linear solver involves linearization of the original system about the current estimate $\hat{\Theta}^r$, solving for the delta vector Δ which is then used for updating the linearization point $\hat{\Theta}^r$.

In order to make online inference tractable, we will more closely examine the linear solver

B. Variable Elimination

The *elimination* algorithm is a way to factorize a factor graph into a Bayes net of the form

$$p\left(\Theta\right) = p\left(\theta_1 | S_1\right) p\left(\theta_2 | S_2\right) \ldots p\left(\theta_n\right) \tag{1}$$

where S_j denotes an assignment to the *separator* $\mathcal{S}(\theta_j)$ of variable θ_j, defined as the set of variables on which θ_j is conditioned, after elimination. The elimination algorithm closely follows the notation in [25], although using a different narrative where the end-result is a Bayes net. Let $\Phi_{j:n} \triangleq \phi(\theta_j, \ldots, \theta_n)$ denote a partially eliminated factor graph. The algorithm proceeds by eliminating one variable θ_j at a time, starting with the complete factor graph $\Phi_{1:n}$. Eliminating a single variable θ_j yields a single conditional $p\left(\theta_j | S_j\right)$, as well as a reduced factor graph $\Phi_{j+1:n}$ on the remaining variables. After all variables have been eliminated, we obtain a Bayes net with the factorization in (1). The computational cost of this algorithm depends strongly on the order in which we eliminate variables, in which orderings yielding small separator set sizes reduce the cost of each elimination operation.

We use partial elimination, in which we create a factor graph $\Phi_{j:n}$ on a smaller set of variables, as a means to create a joint density over a specific set of variables. To create a factor graph on a set of variables Θ_A, choose an ordering such that $\Theta \setminus \Theta_A$ appear first.

C. Incremental Solving with the Bayes Tree

With a linear system factorized through elimination, we can exploit the structure of the resulting Bayes net to allow for incremental solving [14] of a *single solver*, in which we update only sections of an existing Bayes net as we add new factors to the system. In this paper, much of the focus will be on using this solver efficiently in a multi-robot context to avoid expensive batch elimination steps.

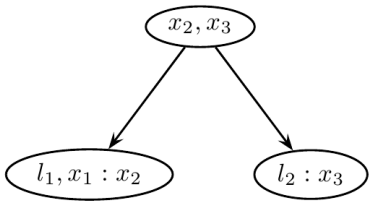

Fig. 2. Small example Bayes tree, showing a root clique with two variables, and two leaf cliques.

The structure of the Bayes net in an incremental solver has as a more general structure that we can exploit: the Bayes tree [22], [23] is a tree-structured graphical model that defines a factored density over cliques of variables. Each node in the tree defines a conditional density conditioned on its parent, in the much same way as a Bayes net. However, unlike Bayes nets, a Bayes tree is always acyclic, and its nodes can have variables in common. The formal definition of the Bayes tree is given below. Given a set of variables Θ, a *Bayes tree* $\mathcal{T} = (\mathcal{C}, \mathcal{E})$ is a rooted tree whose nodes represents *cliques* $C_i \subset \Theta$, while its structure defines the following joint density $P(\Theta)$ on the variables Θ:

$$P(\Theta) = \prod_i P(F_i | S_i) \tag{2}$$

Here the *separator* S_i is defined as the intersection $C_i \cap \Pi_i$ of a clique C_i and its parent clique Π_i, the *frontal variables* F_i are the remaining variables in C_i, i.e., $F_i \triangleq C_i \setminus S_i$, and $P(F_i | S_i)$ is a conditional density on F_i given S_i. An example is shown in Figure 2, where we write $C_i = F_i : S_i$ for a clique C_i.

IV. DDF-SAM 1.0

The DDF-SAM 1.0 architecture [1] has three main parts: a) Each robot r updates its own local map with incoming measurements Z^r from onboard sensors, b) *summarizes* the local map to form a density $\tilde{\Phi}^r\left(\Theta^r\right)$ on exactly the variables to share (usually chosen to be locally observed L^r), and then distributes this density its neighbors; and c) maintains a neighborhood map by fusing all available summarized maps, including the local summarized map, into a single estimate over variables shared between robots.

In this approach, we avoid double-counting by maintaining two separate maps on each robot r: the local map $p\left(X^r, L^r | Z^r\right)$ containing all locally available measurements, and the neighborhood map $p\left(L^{N_r'} | Z^{N_r'}\right)$, which is the fusion center for received summarized maps. The *only* connection between these maps occurs as we summarize the local map into $p\left(L^r | Z^r\right)$ and add this summarization to the neighborhood map. At no point does any information propagate from the neighborhood map back to the local map, which ensures that summarized information is completely independent of any neighboring robots, and that no information is double-counted.

To summarize the local map, we use the partial elimination approach described in Sec. III-B to eliminate X^r from

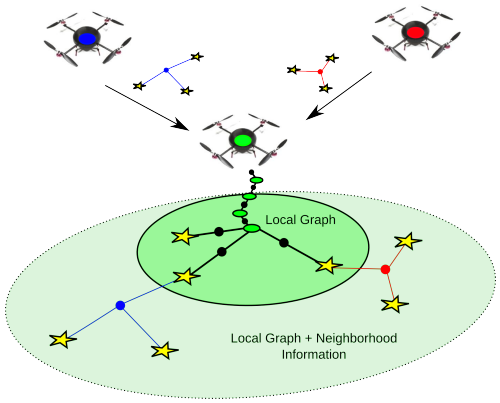

Fig. 3. The augmented local system corresponding to the scenario in Fig. 1, focusing on the center robot receiving summarized maps from the two neighboring robots. Note the expanded region of coverage through the addition of summarized neighborhood information.

$p\left(X^r, L^r | Z^r\right)$, yielding a summarized map $\tilde{\Phi}^r\left(\Theta^r\right) \triangleq p\left(L^r | Z^r\right)$. To compute this summarization, DDF-SAM 1.0 uses *batch summarization*, in which we eliminate the entire local system with a new ordering, which is expensive, and occurs at every summarization interval.

We construct the neighborhood map for each robot r by caching all received summarized maps from neighboring robots and the local robot, such that the neighborhood map

$$p\left(L^{N'_r} | Z^{N'_r}\right) = \prod_i^{N'_r} p\left(L^i | Z^i\right) \tag{3}$$

treats each received summarized map (including the summarization of the local robot's map) identically. Upon receiving new summarized maps, we rebuild the entire graph and solve in batch, which becomes increasingly expensive as the robots explore more area.

While this two-map structure succeeds in avoiding information double-counting by means of preventing any neighborhood information from being incorporated with the local system, this approach is overly conservative and inefficient. In addition, we cannot use neighborhood information to improve the trajectory estimate X^r, and are left with possibly inconsistent maps of landmarks L^r and $L^{N'_r}$.

V. DDF-SAM 2.0

This paper contributes DDF-SAM 2.0, an approach to preventing the double-counting of information, even while combining both local and neighborhood information into a single incremental Bayes tree solver, resulting in an *Augmented Local Graph*, as shown in Fig. 3. This approach enables each robot to maintain a consistent SLAM solution with an extended effective sensor horizon provided by map information from neighboring robots.

It is a simple extension of DDF-SAM 1.0 to assemble a single graph containing both the local density and summarized maps from neighboring robots, and maintain a single incremental solver, but difficulty arises when summarizing this augmented local system to share with other robots. The key insight enabling the augmented local system comes from additive nature of information when incorporating new measurements: not only is it possible to add information to a system, but to *subtract* information as well.

In the following we present an approach for consistent information fusion, i.e. without double-counting, while summarizing information and incorporating the new summarized information into augmented local maps. To that end we introduce the *anti-factor* to down-date factorized estimates. In addition, we can reduce the number of redundant solving operations on each platform by reusing the Bayes tree computed during local updates.

We separate the key operations in this approach into an update step, in which we add new factors from either local measurements or from a new summarized map from another robot, and a summarization step, in which we produce density on a subset of available variables for sharing with neighboring robots. First, we will derive the anti-factor for linear systems.

A. Anti-factor Down-dating

Assume a robot has a factor graph that corresponds to the following system, linearized about the current estimate Θ:

$$A_{1,n}\Delta = b_{1,n} \tag{4}$$

where $A_{1,n}$ and $b_{1,n}$ are defined as

$$\begin{aligned} A_{1,n} &\doteq \begin{bmatrix} A_1^T & \dots & A_r^T & \dots & A_n^T \end{bmatrix}^T \\ b_{1,n} &\doteq \begin{bmatrix} b_1^T & \dots & b_r^T & \dots & b_n^T \end{bmatrix}^T \end{aligned}$$

with A_i and b_i being the Jacobian and the right-hand-side term of the ith factor. The system (4) can be re-written as

$$\sum_{i=1}^{n} A_i^T A_i \Delta = \sum_{i=1}^{n} A_i^T b_i.$$

In practice, we apply factorization techniques to exploit the system sparseness and factorize the Hessian $H_n \doteq \sum_{i=1}^{n} A_i^T A_i$ as $H_n = A_{1,n}^T A_{1,n} = R_n^T R_n$, with R_n being an upper triangular matrix. The system (4) can therefore be written as

$$R_n \Delta = d_n$$

where $d_n \doteq \left(R_n^T\right)^{-1} A_{1,n}^T b_{1,n}$.

Assume the factor that corresponds to the measurement Jacobian A_r and the right-hand-side vector b_r represents information that was obtained from some other robot. This factor should be canceled out before the current robot shares information with that robot. One can readily observe that the factor (A_r, b_r) is canceled out by introducing an anti-factor that corresponds to the following operations

$$H_{n+1} = R_{n+1}^T R_{n+1} = H_n - A_r^T A_r$$

$$d_{n+1} = \left(R_{n+1}^T\right)^{-1} \left(R_n^T d_n - A_r^T b_r\right)$$

Another way to look at it is to consider the term $\|A_r\Delta - b_r\|^2$ that the factor (A_r, b_r) contributes to the cost function. This term can also be written as $\Delta^T A_r^T A_r \Delta -$

$2\Delta^T A_r^T b_r + b_r^T b_r$ and adding the same expression with a negative sign to the cost function will cancel out the factor (A_r, b_r).

In further descriptions, we will denote an anti-factor using inverse notation, such that for a set of factors Φ, the corresponding anti-factor will be denoted Φ^{-1}.

B. Local and Neighborhood Updates

Updating the solver with local factors remains essentially the same as in a single-robot iSAM case, in which we simply add new factors to the existing graph and update the Bayes tree solution.

While adding a new neighborhood system is an operation that consists of adding a new set of factors to the underlying linear solver on a robot, because summarized maps actually *replace* information from previous timesteps, we use anti-factors to cancel the previously added information. Let a be a neighboring robot, sending a summarized graph $\tilde{\Phi}_k^a\left(L_k^a\right)$ at time k, and we wist to integrate this graph into the existing augmented local graph of robot r. Let $\Phi_k^{N_r'}\left(L_k^{N_r} \bigcup \Theta_k^r\right)$ denote an augmented local system, operating on the union of neighborhood variables $L_k^{N_r}$ with the set of all locally observed variables Θ_k^r. If we were to eliminate this system to form a Bayes tree in an incremental solver, we can incorporate a summarized map $\tilde{\Phi}_k^a\left(L_k^a\right)$ from robot a into the augmented local graph exactly as if it were a set of new local measurements. However, when r receives a new summarized map $\tilde{\Phi}_{k+1}^a\left(L_{k+1}^a\right)$, there is substantial shared information with the summarized map previously incorporated. In DDF-SAM 1.0, we solved this problem by reconstructing the entire graph from only the latest cached summarized maps, but this results in a large batch optimization procedure to integrate a new summarized map.

With a down-dating procedure, however, we can subtract out the information from the old summarized map from time k and then add the new summarized map from $k+1$. In order to add a new summarized graph while removing double-counted information, we instead add $\left(\tilde{\Phi}_k^a\left(L_k^a\right)\right)^{-1} \bigcup \tilde{\Phi}_{k+1}^a\left(L_{k+1}^a\right)$, in which we add the anti-factor opposite of the summarized map from time k and the new map from time $k+1$ simultaneously. From the perspective of updating an incremental solver, this is no different an operation than adding new local factors.

C. Summarization

A key component of the DDF-SAM approach is the use of *summarization*, in which each robot shares information on only a subset of variables with its neighbors. Choosing this set of variables to share with neighboring robots allows for throttling the amount of data transmitted between robots to those variables most likely to be useful. As previously, for clarity of description, we will assume only locally observed landmarks are to be shared.

While DDF-SAM 1.0 used batch elimination approach to create an exact joint density on the shared variables, other summarization approaches exist, both through exact algorithms and approximations. In addition to batch, we introduce an alternative approach for computing an exact summarized map through reordering of the Bayes tree, and an approach for efficiently computing a naive Bayes approximation from the Bayes tree.

1) Schur Complement Reordering: We can address the problem of summarizing the augmented local system without recomputing the entire system by periodically reordering the system to place exactly L^r at the root of the Bayes tree, which allows us to extract the summarized map directly from the Bayes tree without computation. In this case, we factorize the system into $p\left(\Theta^r\right) = p\left(X^r, L^{N_r} | L^r\right) p\left(L^r\right)$, such that we can extract $p\left(L^r\right)$ directly from the root of the tree. It should be noted that this approach, like batch summarization, is an exact marginalization operation, so no information is lost in the process.

For this approach, we assume that the Bayes tree solver for a robot r executes reordering at every summarization interval, and during local and neighborhood updates uses the standard Bayes tree incremental procedure. The computational efficacy of this approach will depend on the density of the graph.

2) Naive Bayes Approximate: Because exact marginalization can become expensive, particularly in a densely connected graph, we consider an approximate summarization technique. In using approximate techniques, in order to maintain consistent solutions throughout the robot network, we must ensure that any approximation is more conservative. For its interesting implications on the structure of the resulting Bayes tree, we examine the so-called "Naive Bayes," which approximates an arbitrary distribution $p(x, y, z)$ as independent marginals $p(x)p(y)p(z)$. Thrun et al [12], shows informally that the naive Bayes approximation is always conservative

$$p(x, y, z) \succeq p(x)p(y)p(z)$$

since

$$p(x, y, z) = p(x|y, z)p(y|z)p(z)$$

and conditioning on a variable can only increase information, $p(y|z)$ is more informative than $p(y)$, and $p(x|y, z)$ is more informative than $p(x)$. As such, we can construct a summarized map out of the marginals of each saved variable.

The default approach for computing variable marginals in a graphical system is to linearize and reorder, such that the target variable θ_j is at the end of the ordering, and then eliminate the entire system. This is an expensive operation for a single marginal, as it amounts to the same computational process as batch summarization, but must also be repeated for each variable. However, we can exploit the structure of the Bayes tree to cache intermediate computations, which allows us to only eliminate small portions of the tree. We can compute a marginal $p(C_i)$, in which C_i contains θ_j, and then eliminate $p(C_i)$ to yield the marginal θ_j.

The key is using the recursive structure of the tree to compute only the marginals $p(S_i)$ on the separator sets S_i, via

$$p(S_i) = \int_{\neg S_i} p(F_\pi | S_\pi) p(S_\pi) \tag{5}$$

where we informally used $\Pi_i = F_\pi : S_\pi$. To avoid redundant computation, we cache the separator marginals $p(S_\pi)$ so each is only computed once. We can then compute the clique marginals $p(C_i)$ by

$$p(C_i) = p(F_i | S_i) p(S_i)$$

Fine-grained marginals on an individual variable θ_j can always be computed by

$$p(\theta_j) = \int_{\neg \theta_j} p(F_i)$$

for the (unique) clique C_i where θ_j is a frontal variable $\theta_j \in F_i$, and where the frontal marginals $p(F_i)$ can be computed using with clique or separator marginals:

$$p(F_i) = \int_{S_i} p(C_i) = p(F_i | S_i) \int_{S_i} p(S_i)$$

If the marginal for more than one variable in F_i is required, it is beneficial to cache the calculation of $p(F_i)$.

A key characteristic of the resulting summarized map is that rather than being a densely connected joint, as results from exact approaches, the naive Bayes summarized map is completely disconnected. While this has more obvious implications on the size of messages between robots, the neighborhood update step for robots receiving this summarized map will be much sparser, such that updating the Bayes tree will be less computationally expensive.

D. Down-dating Summarizations

While it is possible to use the existing Bayes tree to compute a summarization, the measurements from the neighboring robots N_r are still present in the system, we can cancel out this neighborhood information through the use of anti-factors. The summarized graph $\tilde{\Phi}_k^r(L_k^r)$ sent by robot r becomes the union of factors from the joint $p\left(L_k^r | Z_k^{N_r^t}\right)$ on L_k^r, and the set of all antifactors from cached neighborhood maps $\bigcup_{a \in N_r} \left(\tilde{\Phi}_k^a(L_k^a)\right)^{-1}$. While this results in adding additional factors to a summarized graph before transmission, it should be noted that linear factors in Hessian form can be combined additively, ensuring the size of summarized maps does not necessarily increase with neighborhood size. Because we have exactly canceled out the information contributions from *all neighboring robots* N_r before transmission, we ensure that no information will be double-counted as these summarized maps propagate between robots.

VI. Evaluation

In order to evaluate exact cancellation of information during inference, we simplified the scenario to remove the additional complexities induced by nonlinearities, resulting in a nearly-linear system. In this case, we assume each robot chooses the same linearization point for each newly observed variable. The impact of the GPS measurements is more subtle, in that by constraining the position of each pose in an absolute coordinate frame, we ensure that rotational drift on each platform cannot result in large changes to the solution in the event of a loop closure. The motivation for these simplifications in the simulated system is to focus on ensuring exact removal of double-counted information, and will address the more general case, incorporating re-linearization and separate decentralized variable initialization, in future work.

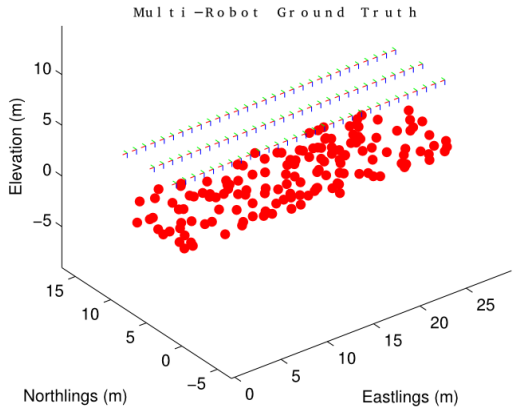

Fig. 4. Evaluation scenario ground truth with three flying robots over a field of randomly generated landmarks.

The implementation of both DDF-SAM 2.0 and an DDF-SAM 1.0 (used as a control) uses the GTSAM C++ library for the underlying graphical inference algorithms, with simulation and visualization provided through a Matlab wrapper. To compare summarization approaches, at each summarization interval, the system performs both the DDF-SAM 2.0 Bayes tree summarization, making use of anti-factors, and the full batch summarization approach of DDF-SAM 1.0. To evaluate the effect on map quality of using the augmented local map of DDF-SAM 2.0 over the split map representation of DDF-SAM 1.0, we examined the precision of estimates for both local and neighborhood maps.

We evaluated this new DDF-SAM 2.0 approach in a simulated scenario designed to represent a small team of quadrotor helicopters flying outdoors over an area of interest. In this system, we model three platforms, each equipped with a downwards facing camera, an inertial measurement unit and a GPS system. The camera runs an image-based feature detector, which produces labeled landmark detections. As a simplifying assumption, the simulation provides known data associations, both at the local and neighborhood level, for each landmark.

The scenario includes randomly generated terrain consisting of a volume of landmarks $30\,m$ by $10\,m$ by $0.5\,m$, with 150 landmarks uniformly distributed throughout. The robot trajectories represented forward flight at an altitude of $6\,m$ above the landmarks, at a simulated velocity of $1\,^m/_s$ over an interval of $30\,sec$.

VII. Results

Example plots comparing the augmented local system appear in Fig. 5 and neighborhood-only solution in Fig. 6.

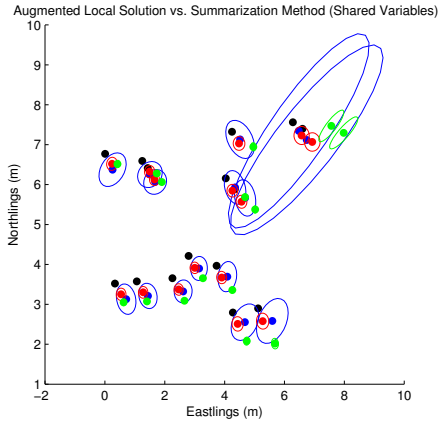

Fig. 5. DDF-SAM 2.0 Augmented Local System plotted at 4 seconds into simulation with landmark covariances plotted, with comparison to ground truth (black), augmented local systems with batch summarization (red) and naive Bayes summarization (green), and a pure local solution (blue). This image shows only variables with both local and neighborhood information.

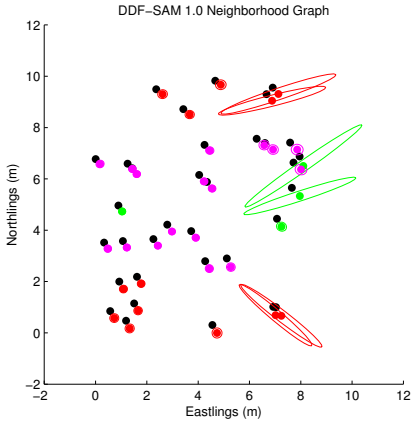

Fig. 6. DDF-SAM 1.0 neighborhood-only map 2D plots with covariances tracking the center robot in the simulated scenario, with comparison to ground truth. Black denotes ground truth, green landmarks are purely local landmarks, magenta landmarks overlap with neighbors, and red landmarks are only observed through neighbors.

These plots were extracted after a small number of poses to ensure covariance ellipses are readily visible, and are projected into a 2D plane for clarity of presentation. In these cases, we compare the effect of approximate summarization with exact batch summarization and the pure local solution on variables shared between robots.

Fig. 5 compares the effect of the exactness of summarization approach on augmented local map solution, with ground truth and the pure local solution as control comparisons. In this case, we show the result on a single robot with two neighbors, with the augmented local system created from exact batch summarization and with naive Bayes approximate summarization. The solution and covariance ellipses for both summarization types are similar, and both are tighter than the pure local solutions, indicating that the local estimate for a robot improves with the integration of neighborhood factors. Note that the local solution plotted is exactly the DDF-SAM 1.0 local map solution, which clearly shows that augmenting the local map also improves estimation precision.

Comparing the uncertainty estimates for the augmented local systems in Fig. 5 and the DDF-SAM 1.0 equivalent pure-neighborhood plot in Fig. 6 provides qualitative insight into consistency in DDF-SAM 2.0. Note that the marginal covariance ellipses for variables in the neighborhood-only system, which is guaranteed consistent by construction, are the same as the local augmented system. As a qualitative evaluation of consistency between the systems, the marginal covariance ellipses be the same in each case, so long as no information is double-counted or ignored.

In addition to qualitative measures of consistency, we computed a numerical score for the information present in individual summarized maps, defined as the trace of the information matrix of each summarized map. If anti-factors down-date summarized maps correctly, all summarization techniques that directly exploit the Bayes tree should be no more certain than the control batch summarization technique.

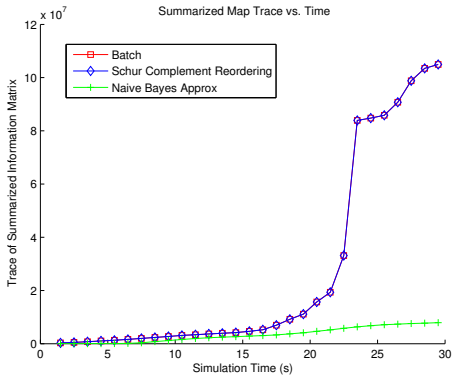

Fig. 7. Quantitative measurement of information in summarized maps - the trace of the information matrix for each summarization type. The difference between batch summarization and Schur complement summarization is within numerical error.

Fig. 7 shows this plot over time, indicating that the naive Bayes approach is a conservative estimate for the full joint density on the landmarks.

To evaluate computational impact of different summarization approaches, we present the timings for local updates, summarization and neighborhood updates in Fig. 8. In this case, we separated the timing plots for local and neighborhood updates because we expect neighborhood updates to become more expensive over time due to the increasing size of summarized maps as robots explore.

VIII. Conclusions and Future Work

In this paper we presented a consistent decentralized SLAM that extends the previous method by combining local maps and information shared by other robots within communication range into augmented maps. We introduced anti-factor as a tool to avoid double-counting by down-dating and systematically analyzing its application for incrementally maintaining local augmented systems. In addition to the original batch summarization approach, we presented an

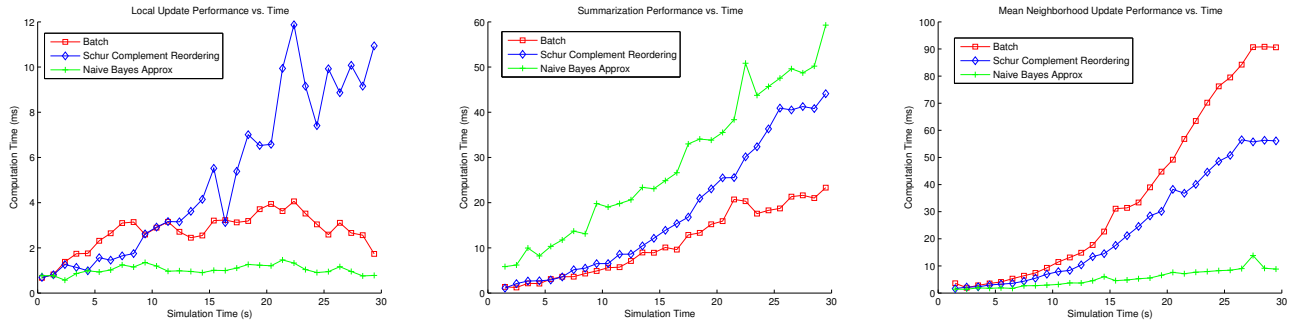

Fig. 8. Timing performance for updating a local augmented system with incoming local measurements (left), computing the summarized map (center), and updating the local augmented system with summarized maps from neighboring robots (measured as mean time to add an incoming summarized map). Each operation is evaluated when using batch summarization (equivalent to DDF-SAM 1.0), Schur complement reordering summarization and the naive Bayes approximate reordering.

alternate exact summarization approach operating on the Bayes tree directly, and a naive Bayes approximate summarization algorithm. We presented an evaluation in a synthetic, mostly linear scenario demonstrating that the DDF-SAM 2.0 algorithm can produce consistent estimates that exactly avoid information double-counting. The results of this experiment show that when summarizing a local augmented system, we successfully subtract exactly the information that is double-counted to yield summarizations that have equivalent uncertainty to a batch summarization process.

In future work, we will relax the core assumptions made by the evaluation presented in this paper by incorporating re-linearization into the the approach, as well as handling separate linearization points between robot platforms. We are also investigating means to approximate or directly compute a joint over a subset of variables directly from the Bayes tree efficiently. As an additional future goal, we will examine means to introduce incremental communication, such that robots no longer need to share all of their landmarks at once. By addressing these core concerns, we can demonstrate a system capable of mapping in unknown environments.

REFERENCES

[1] A. Cunningham, M. Paluri, and F. Dellaert, "DDF-SAM: Fully distributed slam using constrained factor graphs," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[2] A. Cunningham, K. Wurm, W. Burgard, and F. Dellaert, "Fully distributed scalable smoothing and mapping with robust multi-robot data association," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2012.

[3] F. Dellaert and M. Kaess, "Square Root SAM: Simultaneous localization and mapping via square root information smoothing," *Intl. J. of Robotics Research*, vol. 25, pp. 1181–1203, Dec 2006.

[4] H. Durrant-Whyte and M. Stevens, "Data fusion in decentralized sensing networks," in *4th Intl. Conf. on Information Fusion*, 2001.

[5] S. Roumeliotis and G. Bekey, "Distributed multi-robot localization," *IEEE Trans. Robot. Automat.*, August 2002. To Appear.

[6] E. Nerurkar, S. Roumeliotis, and A. Martinelli, "Distributed maximum a posteriori estimation for multi-robot cooperative localization," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pp. 1402–1409, May 2009.

[7] A. Bahr, M. Walter, and J. Leonard, "Consistent cooperative localization," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pp. 3415–3422, May 2009.

[8] V. Indelman, P. Gurfil, E. Rivlin, and H. Rotstein, "Graph-based distributed cooperative navigation for a general multi-robot measurement model," *Intl. J. of Robotics Research*, vol. 31, August 2012.

[9] S. Julier and J. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *American Control Conference*, pp. 2369–73, 1997.

[10] S. Julier and J. K. Uhlmann, "Using covariance intersection for slam," *Robotics and Autonomous Systems*, vol. 55, pp. 3–20, Jan. 2007.

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

[12] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT press, Cambridge, MA, 2005.

[13] L. Andersson and J. Nygards, "C-SAM : Multi-robot SLAM using square root information smoothing," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2008.

[14] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. Robotics*, vol. 24, pp. 1365–1378, Dec 2008.

[15] B. Kim, M. Kaess, L. Fletcher, J. Leonard, A. Bachrach, N. Roy, and S. Teller, "Multiple relative pose graphs for robust cooperative mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Anchorage, Alaska), pp. 3185–3192, May 2010.

[16] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte, "Decentralised cooperative localisation for heterogeneous teams of mobile robots," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2011.

[17] M. Bryson and S. Sukkarieh, "Architectures for cooperative airborne simultaneous localisation and mapping," *Journal of Intelligent and Robotic Systems, Special Issue on Airborne SLAM*, vol. 55, pp. 267–297, June 2009.

[18] R. Aragues, J.Cortes, and C. Sagues, "Distributed consensus on robot networks for dynamically merging feature-based maps," *IEEE Transactions on Robotics*, 2012.

[19] S. Williams, G. Dissanayake, and H. Durrant-Whyte, "Towards multi-vehicle simultaneous localisation and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2002.

[20] S. E. Webster, L. L. Whitcomb, and R. M. Eustice, "Preliminary results in decentralized estimation for single-beacon acoustic underwater navigation," in *Robotics: Science and Systems (RSS)*, (Zaragoza, Spain), June 2010.

[21] R. Aragues, E. Montijano, and C. Sagues, "Consistent data association in multi-robot systems with limited communications," in *Robotics: Science and Systems (RSS)*, (Zaragoza, Spain), June 2010.

[22] M. Kaess, V. Ila, R. Roberts, and F. Dellaert, "The Bayes tree: An algorithmic foundation for probabilistic robot mapping," in *Intl. Workshop on the Algorithmic Foundations of Robotics*, Dec 2010.

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

[24] F. Kschischang, B. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, February 2001.

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