Robust Optimization of Factor Graphs by using Condensed Measurements

Giorgio Grisetti Rainer Kümmerle Kai Ni

***Abstract*— Popular problems in robotics and computer vision like simultaneous localization and mapping (SLAM) or structure from motion (SfM) require to solve a least-squares problem that can be effectively represented by factor graphs. The chance to find the global minimum of such problems depends on both the initial guess and the non-linearity of the sensor models. In this paper we propose an approach to determine an approximation of the original problem that has a larger convergence basin. To this end, we employ a divide-and-conquer approach that exploits the structure of the factor graph. Our approach has been validated on real-world and simulated experiments and is able to succeed in finding the global minimum in situations where other state-of-the-art methods fail.**

I. Introduction

Several problems in autonomous robotics and computer vision, like simultaneous localization and mapping (SLAM) or structure from motion (SfM), require the solution of a least-squares problem that exhibits a strong locality, which results in the sparse structure of the corresponding factor graph. Variables in these problems are correlated when they are temporally or spatially close. The temporal locality is the consequence of the sequential data acquisition, while the spatial locality derives from the limited range of the sensors.

A common way to solve these least-squares problems is exploiting iterative methods like Gauss-Newton or Levenberg-Marquardt, which progressively refine an initial guess until convergence. If this guess is out of the convergence basin of the algorithms, the iterative optimization may not converge to the global minimum. The convergence basin, in turn depends on the shape of the sensor models used to construct the factors of the least-squares problem. The more irregular/non-linear these error functions are, the tighter the convergence basin is.

In this paper, we discuss an approach to determine a good initial guess for least-squares problems arising from SLAM or SfM. The key idea of our approach is to partition the input factor graph into small locally connected sub-graphs that represent sub-problems. These sub-problems can be solved robustly and efficiently, but their solutions cannot be combined together in a straightforward manner. To this end, from each partial solution we construct a simple factor graph that constrains the relative positions of the variables in the solution. These sub-graphs represent an convex approximation of the original ones which exhibit a larger convergence basin. The factors in these graphs incorporate the information contained in the sub-graph from which they were generated, thus we refer to them as *condensed* measurements. To find the global layout of the partial solutions in the space, we find the minimum of the union of all translated sub-graphs. As this translated problem is more convex, we have increased the chance of finding the correct minimum, which represents the initial guess. Our method can be seen as a generalization of other divide-and-conquer methods like HOG-Man [1] or T-SAM [2]. Compared to HOG-Man our approach allows to deal with factor graphs whose variables are not only robot poses, but arbitrary elements like landmarks and system parameters. T-SAM is orthogonal to our approach since it mostly focuses on the solution of the linear sub-problem and does not address the non-linear aspects in the general case. We tested our approach on a wide set of real-world and simulated experiments acquired with different 2D and 3D sensors. Furthermore, we performed a statistical comparative analysis. Our approach succeeded in solving problems where other state-of-the-art methods failed. Our system is built as an extension to g^{2}o [3], an open source generic factor graph optimization package. Thus, it can be straightforwardly applied to all the instances of problems handled by g^{2}o with minimal effort.

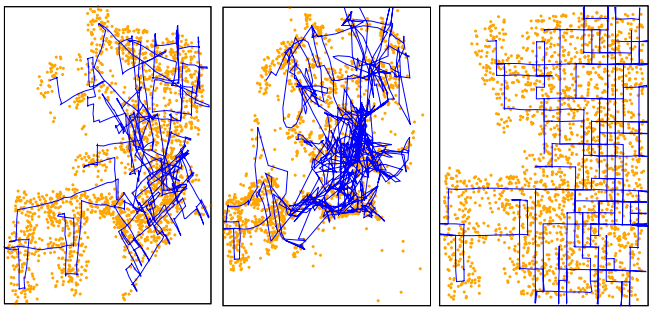

Fig. 1. A robot equipped with a stereo camera is simulated in a Manhattan world. The trajectory is drawn in blue whereas the features are depicted in orange. Left: The initial guess is computed by composing the odometry of the robot. Middle: Running a standard iterative Levenberg-Marquardt algorithm yields a sub-optimal estimate. Right: Our approach converges to the global minimum.

Figure 1 shows a motivating example of our approach where we simulated a robot with a stereo camera driving in a Manhattan maze. A standard iterative Levenberg-Marquardt algorithm converges to a local minimum. However, our approach estimates the correct solution.

II. Related Work

In the past, graph optimization problems have been studied intensively in the area of robotics and computer vision. One seminal work is that of Lu and Milios [4] where the

G. Grisetti and R. Kümmerle are with the University of Freiburg. G. Grisetti is also with Sapienza, University of Rome. Kai Ni is with Microsoft.

relative motion between two scans was measured by scan-matching and the resulting graph was optimized by iterative linearization. At that time optimization of the graph was regarded as too time-consuming for real-time performance. Due to recent advancements in the development of direct linear solvers (e.g., [5]), graph-based SLAM has re-gained popularity, and a large variety of different approaches to optimize pose graphs have been proposed. For example, Howard *et al.* [6] apply relaxation to build a map. Duckett *et al.* [7] propose the usage of Gauss-Seidel relaxation to minimize the error in the network of constraints. Frese *et al.* [8] introduced multi-level relaxation (MLR), a variant of Gauss-Seidel relaxation that applies the relaxation at different levels of resolution. Olson *et al.* [9] suggested a gradient descent approach to optimize pose-graphs (i.e., graphs whose variables only consist robot poses). Later, Grisetti *et al.* [10] extended this approach by applying a tree-based parameterization that increases the convergence speed. Both approaches are robust to the initial guess and rather easy to implement. However, their application is restricted to graphs whose nodes represent only isometries and do not support different types of variables. Furthermore, they assume that the covariance is roughly spherical and thus have difficulties in optimizing pose-graphs, where some constraints have covariances with null spaces or substantial differences in the eigenvalues. Assuming diagonal covariances Carlone *et al.* [11] recently demonstrated how to obtain a linear approximation independent of the initial guess. However, their method is limited to 2D pose-graphs.

In presence of Gaussian sensor noise, inferencing in a factor graph corresponds to solving a nonlinear least-squares problem. Typically this is done by forming a linear system around the current state, solving, and iterating. One promising technique for solving the linear system is preconditioned conjugate gradient (PCG), which was used by Konolige [12] as well as Montemerlo and Thrun [13] as an efficient solver for large sparse pose constraint systems.

More recently, Dellaert and colleagues suggested a system called $\sqrt{\text{SAM}}$ [14] implemented using sparse direct linear solvers [5]. Kaess *et al.* [15] introduced a variant of this called iSAM that is able to update the linear matrix associated with the nonlinear least-squares problem. Konolige *et al.* [16] showed how to construct the linear matrix efficiently by exploiting the typical sparse structure of the linear system. However, the latter approach is restricted to 2D pose-graphs.

In computer vision, Sparse Bundle Adjustment [17] is a nonlinear least-squares method that takes advantage of the sparsity of the Jacobian pattern between points and camera poses. Recently, there have been several systems [18], [19] that advance similar concepts of sparse linear solvers and efficient factorizations of the system of linear equations ($\sim$100M sparse matrix elements). Additionally, several approaches to Visual SLAM [20], [21] employ a similar concept like our approach. Those systems generate a pose-graph representation out of the camera measurements obtained at key frame locations. They are, however, restricted to generate pair-wise camera constraints for the pose-graph layer.

Hierarchical extensions to these approaches have been proposed by Ni *et al.* [2] (TSAM) and Grisetti *et al.* [1] (HOG-Man) that employ a divide-and-conquer strategy to partition the original large problem into smaller ones and then utilize these partial solutions to construct a global sparse problem. The solution of the global problem is the arrangement of the partial solutions (local maps) in the space. In particular [2] utilizes nested dissection and an efficient parameterization to efficiently solve large off-core problems involving arbitrary types of variables. Conversely, [1] encapsulates the relations between sub-problems in nonlinear constraints connecting them, but it is restricted to operate on pose-graphs. The approach presented in this paper is orthogonal to TSAM and extends it by expressing the solution of the sub-problems by using more general measurement functions that *condense* most of the relevant information in the local solution. These error functions are user defined, so they can be chosen to be as smooth as possible, *regardless* of the factors in the input problem. The smoothness of the condensed measurements, makes our approach more robust to wrong initial guesses. Our method also extends HOG-Man, since it operates on arbitrary factor graphs and is not restricted to pose-graphs.

III. Least Squares Optimization of Factor Graphs on a Manifold

Minimizing a factor graph consists in solving the following equation:

$$F(\mathbf{x}) = \sum_{k \in \mathcal{C}} \underbrace{\mathbf{e}_k(\boldsymbol{x}_k, \mathbf{z}_k)^T \boldsymbol{\Omega}_k \mathbf{e}_k(\boldsymbol{x}_k, \mathbf{z}_k)}_{F_k} \quad (1)$$

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} F(\mathbf{x}). \quad (2)$$

Here

- $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the state vector where each $\mathbf{x}_i$ represents one generic state variable. For instance, in SLAM these variables can model robot positions or the pose of landmarks.
- F_k: is a factor that models a measurement depending on a subset $\boldsymbol{x}_k = (\mathbf{x}_{k_1}, \ldots, \mathbf{x}_{k_q}) \subset (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ of the state variables.
- $\mathbf{z}_k$ is a measurement that depends on the state variables in $\boldsymbol{x}_k$. Given a configuration of $\boldsymbol{x}_k$, a prediction $\hat{\mathbf{z}}_k = \mathbf{h}_k(\boldsymbol{x}_k)$ of the measurement can be computed by a sensor model $\mathbf{h}_k(\boldsymbol{x}_k)$. The uncertainty of the measurement is modeled by its information matrix $\boldsymbol{\Omega}_k^z$.
- $\mathbf{e}_k(\boldsymbol{x}_k, \mathbf{z}_k)$ is an error function that computes the distance vector between the prediction $\hat{\mathbf{z}}_k$ and a real measurement $\mathbf{z}_k$. This error is $\mathbf{0}$ when the prediction obtained by mapping the states $\boldsymbol{x}_k$ to the measurements is equal to the real measurement: $\mathbf{z}_k = \mathbf{h}_k(\boldsymbol{x}_k)$. A straightforward error function is the vector difference: $\mathbf{e}(\boldsymbol{x}_k, \mathbf{z}_k) = \mathbf{h}_k(\boldsymbol{x}_k) - \mathbf{z}_k$, but other choices are possible.
- $\boldsymbol{\Omega}_k$ is the information matrix that models the uncertainty of the *error*. It depends on the measurement function and on the measurement uncertainty represented by $\boldsymbol{\Omega}_k^z$. If the error function is the vector difference between prediction and measurement, $\boldsymbol{\Omega}_k = \boldsymbol{\Omega}_k^z$.

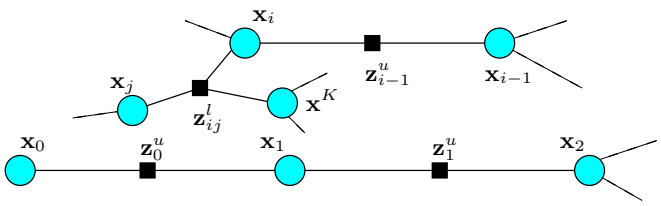

Fig. 2. Here we illustrate a portion of a SLAM problem with unknown parameters represented as a factor graph. The round nodes represent state variables, while the square nodes represent the factors. $\mathbf{x}_{0:n}$ denotes the robot poses, and $\mathbf{x}^K$ denotes the unknown calibration parameters of a sensor on the robot. The factors are depicted as black squares and arise either from odometry measurements $\mathbf{z}^u_{0:n}$ or from environment measurements $\mathbf{z}^l_{ij}$ which relate pairs of robot locations $\mathbf{x}_i$ and $\mathbf{x}_j$ and calibration parameters $\mathbf{x}^K$.

To simplify the notation, in the rest of this paper we will encode the measurement in the indexes of the error functions:

$$\mathbf{e}_k(\boldsymbol{x}_k, \mathbf{z}_k) \overset{\text{def.}}{=} \mathbf{e}_k(\boldsymbol{x}_k) \overset{\text{def.}}{=} \mathbf{e}_k(\mathbf{x}). \tag{3}$$

A problem in this form can be effectively represented by a factor graph. A factor graph is a bipartite graph where each node represents either a variable node $\mathbf{x}_i$ or a factor node F_k between a subset $\boldsymbol{x}_k$ of state variables involved in the k^{th} constraint. A factor is connected by edges to all nodes in the subset $\boldsymbol{x}_k$. Figure 2 shows an example of a mapping between a factor graph and an objective function.

If a good initial guess $\breve{\mathbf{x}}$ of the parameters is known, a numerical solution of Eq. 2 can be obtained by the Gauss-Newton (GN) or Levenberg-Marquardt (LM) algorithms. In the remainder, we will refer to these approaches as direct methods. In this section we review two common tricks that make least-squares minimization more robust when the measurements $\mathbf{z}_k$ or the state variables $\boldsymbol{x}_k$ span over non-Euclidean manifold spaces.

As proposed in [22], we can define the following two operators:

- $\boxplus : \mathbf{x}_1 \times \mathbf{x}_2 \rightarrow \mathbf{x}$, that applies the perturbation $\mathbf{x}_2$ to $\mathbf{x}_1$.
- $\boxminus : \mathbf{x}_1 \times \mathbf{x}_2 \rightarrow \mathbf{x}$, that computes the perturbation $\mathbf{x}$ that transforms $\mathbf{x}_1$ to $\mathbf{x}_2$.

Since the increments computed by iterative methods are usually rather small, their minimal representation is far from the singularities. Accordingly, when operating with non-Euclidean measurement spaces, it is convenient to use the $\boxminus$ operator instead of the regular minus to compute the error. More in detail, let $\hat{\mathbf{z}}_k = \mathbf{h}(\boldsymbol{x}_k)$ be a measurement function, the error function

$$\mathbf{e}_k(\boldsymbol{x}_k) = \hat{\mathbf{z}}_k \boxminus \mathbf{z}_k \tag{4}$$

usually has a smoother behavior than the vector difference $\hat{\mathbf{z}}_k - \mathbf{z}_k$. In Eq. 4 the difference is first computed in the original non-Euclidean domain, and then it is converted to a vector form. This difference is supposed to be small, thus its minimal form is far from singularities. Conversely, the vector difference first computes the minimal representation of values potentially far from the origin (and thus close to the singularities), and then computes the difference. This can result in high values of the error norm that come from little displacements in the orientation.

Similarly, we can use the $\boxplus$ operator to write down the equation of the error under a small perturbation of the state variables around a linearization point $\breve{\mathbf{x}}$ as:

$$\begin{aligned} \mathbf{e}_k(\breve{\boldsymbol{x}}_k \boxplus \boldsymbol{\Delta x}_k) &= \mathbf{e}_k(\breve{\mathbf{x}} \boxplus \boldsymbol{\Delta x}) && (5)\\ &\simeq \mathbf{e}_k + \mathbf{J}_k \boldsymbol{\Delta x}. && (6) \end{aligned}$$

Here the Jacobian is $\mathbf{J}_k = \frac{\partial \mathbf{e}_k(\breve{\mathbf{x}} \boxplus \boldsymbol{\Delta x})}{\partial \boldsymbol{\Delta x}}\Big|_{\boldsymbol{\Delta x}=\mathbf{0}}$. The advantage of using the $\boxplus$ operator instead of the more common $+$ is that $\boxplus$ takes care of handling the singularities and the error function in $\boldsymbol{\Delta x}$ has a smoother profile. Clearly, in case the measurements or the state variables are Euclidean, the $\boxplus$ and the $\boxminus$ become a regular vector addition and subtraction.

Substituting Eq. 6 in Eq. 1 we obtain a quadratic form that approximates the error given the perturbations of the variables:

$$F(\breve{\mathbf{x}} \boxplus \boldsymbol{\Delta x}) = \mathrm{c} + 2\mathbf{b}\boldsymbol{\Delta x} + \boldsymbol{\Delta x}^T \mathbf{H} \boldsymbol{\Delta x} \tag{7}$$

Here $c = \sum_k \mathbf{e}_k^T \boldsymbol{\Omega}_k \mathbf{e}_k$ is a constant term, $\mathbf{b} = \sum_k \mathbf{J}_k^T \boldsymbol{\Omega}_k \mathbf{e}_k$ is the coefficient vector and $\mathbf{H} = \sum_k \mathbf{J}_k^T \boldsymbol{\Omega}_k \mathbf{J}_k$, is the information matrix of the system.

The quadratic form in Eq. 7 can be minimized in $\boldsymbol{\Delta x}$ by solving the linear system

$$(\mathbf{H} + \lambda \mathbf{I})\, \boldsymbol{\Delta x}^* = -\mathbf{b}. \tag{8}$$

Here λ is a damping factor used to render the system positive definite and to control the convergence. $\mathbf{H}$ is sparse by construction, having non-zeros only between variables appearing in a factor. This allows us to solve Eq. 8 with efficient approaches like sparse Cholesky factorization or Preconditioned Conjugate Gradients (PCG). The linearized solution is then obtained by applying the perturbation to the initial guess via $\boxplus$

$$\mathbf{x}^* = \breve{\mathbf{x}} \boxplus \boldsymbol{\Delta x}^*. \tag{9}$$

Direct methods iterate the linearization in Eq. 7, the solution in Eq. 8, and the update step in Eq. 9, updating the damping factor and executing backup steps to achieve monotonic convergence.

IV. Considerations about SLAM-like problems

In a large class of problems, e.g., SLAM or SfM, the variables in the factor graph represent spatial entities that are either poses of the moving sensor (laser range finders, cameras, etc) or positions of the observed entities (landmarks, scans, local maps and so on). The factors correspond to the measurements that depend on either single agent positions (like GPS, magnetic field, or attitude), temporally subsequent agent positions (like odometry, velocity, or acceleration), or they arise by the observation of a map element from a certain position of the observer. Usually, each measurement involves a set of variables that are spatially close. This results in a *local connectivity* of the graph: since variables are related to spatial entities and only variables that are within a certain range between each other are connected.

In the presence of devices having a highly nonlinear model, like a bearing only sensor or a monocular camera,

and in absence of a good initial guess, direct methods can fail. Observing the same scene with more informative sensors that are, for instance, capable of detecting also the range of a landmark or the depth of a point increases the chances of success for direct methods. Thus, direct approaches applied to SLAM-like problems are *sensitive to the sensor model*: the used sensor model has a great impact on the profile of the error function, and thus on the chances of finding the global minimum. Clearly, the sensor model is a characteristic of the problem and cannot be arbitrarily changed. However, once we have a consistent local solution for a portion of the problem, we can formulate another problem having a similar solution that uses less non-linear sensor models. It is in general convenient to use in this stage sensor models that have the smoothest possible error profile and that allow to observe the highest possible number of dimensions of the involved state variables.

For short trajectories the open loop estimate obtained by using incremental approaches (wheel odometry, visual odometry, integration of the accelerometers) is sufficient to obtain a good solution despite using less informative measures (like bearing only data). Thus, direct approaches work well on *small size* problems.

V. Robust Optimization of Factor Graphs using Condensed Measurements

In this section, we illustrate our approach based on the three SLAM features highlighted above. We provide a graphical explanation through Figure 3 that shows how our approach works on a simple landmark-based SLAM problem. In this case, we restricted ourselves to a factor graph involving only binary factors and in the figure we highlight only the nodes that represent variables. The binary factors are represented by the edges, while triangles denote robot poses and circles landmark locations.

Our approach partitions the original problem into small chunks based on the trajectory of the vehicle. This is illustrated by the dotted-dashed line in Fig 3a. Each of these chunks form a small factor graph describing a portion of the problem. Because of the local connectivity, each factor will capture a small contiguous portion of the environment that can be seen as a local map. These local maps interact with each other through variables belonging to more than a single local map. These *shared* variables are illustrated in red.

Since these sub-graphs are small, we can obtain a reasonable solution for each of these problems by using a direct method. In absence of global measurements, these local maps are free to float in the space. Thus, to determine a unique solution, we need to "fix" some variables. In the remainder of this section, we will refer to these variables as the *origin* (gauge) of the local maps and are illustrated in dark blue in Fig. 3c. Having a solution for a local map means that we know a Gaussian approximation of each variable within the local map, relative to its origin. This is true *regardless* the sensors that have been used to determine this solution. As an example we can obtain the $x-y$ location of landmarks in a

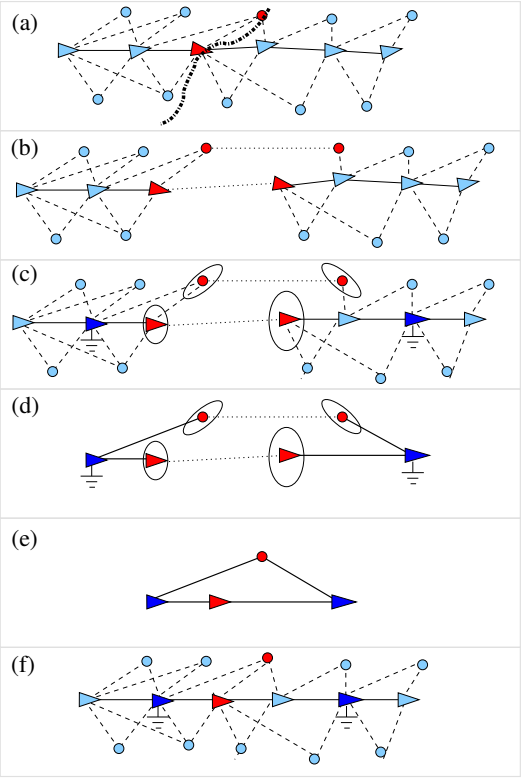

Fig. 3. The overview of our optimization procedure on a simple landmark-based SLAM problem. Here we illustrate the factor graph by highlighting only the variables. The factors denote binary measurements and are encoded in the edges. (a),(b) We partition the problem into sub-graphs. The shared variables are in red, and dotted lines show the corresponding variables in different partitions. (c) We solve these problems independently with respect to their origins (dark blue), and we determine the marginal covariances of the shared variables. (d) We compute condensed factors connecting each shared variable to its origin. (e) We solve the complete problem on the condensed factors to determine the layout of the local maps (f).

SLAM problem even if our robot is equipped with bearing-only sensors, because multiple observations are fused by the SLAM algorithm to obtain the solution. This step is described in detail in Section V-A.

Once we have a solution for the local maps, we seek for a global alignment that satisfies all the equality constraints induced by the shared variables. These constraints are depicted with dotted lines in figures 3b, 3c, and 3d. Approaching this problem by initializing the original factor graph with the local solutions can fail, because these local solutions will be destroyed during the global optimization. We then reduce the problem of determining a global initial guess to finding a global alignment of the local maps and of the shared variables, while attempting to preserve the structures of the local maps computed before.

To this end, we replace the factor graph of each local map with another and simpler one whose solution approximates the original local solution. In constructing this factor graph, we utilize smooth sensor models that can fully qualify each variable within the local map with respect to its origin. We construct the reduced problem by considering the variables in the origin along with all the shared variables of a local map. We then add a factor between each shared variable

and the origin, as illustrated in Figure 3d. This factor is computed from a measurement function that depends on the type of the variable: we should obviously describe differently measurements of a robot pose and measurements of a landmark. The mean and the information matrix of the measurement are computed by projecting the marginal covariance of a state variable in the measurement space via the unscented transform [23]. This step is discussed in detail in Section V-B.

This procedure converts the factor graph of a local map, which can have an arbitrary topology into a star topology. The center of the star is the origin of the local map and the other nodes are the shared variables. The non-shared variables are not considered in this stage, since they do not directly concur to determine a global alignment. The shared variables are connected to the origin by factors that are generated based on the solution of the local map. We call these factors *condensed* factors, since they summarize the relationship between a variable and the origin of the local map by considering *all* the measurements when optimizing the local map.

Once we "condensed" the local maps, we assemble an approximation of the original global factor graph by combining all the newly computed factor graphs into a new sparser factor graph, whose solution is a global configuration of the origins and of the shared variables. This is illustrated in Figure 3e. Furthermore, since the sensor models are smoother than the original ones, the new problem will have a larger convergence basin and direct approaches are likely to work. Having found an approximated solution for the origins of the local maps and of the shared variables, we can determine a good initial guess by arranging the local maps computed at the beginning of the procedure accordingly, as shown in Figure 3f. At this point, an optimization which considers the original factors can further refine the approximated solution.

A. Constructing and Solving the Local Maps

The procedure outlined above requires to first partition the input into small sub-graphs that lead to local maps. Since the local maps need to be merged in a later step, we should prevent the information stored in the original problem from being used multiple times. The relations between variables are modeled by the factors of the graph. Hence, it is sufficient to partition the *factors* of the input problem into different local maps. Conversely, the variables can be replicated; a variable that appears in more than one local map becomes a shared variable, that is a vertex separator of the original factor graph.

Additionally, we want these local maps to admit unique solutions, which means that the resulting linear system of Eq. 8 should be fully determined, once we fix the vertex in the origin. In general this is a challenging problem, however, since we assume that we have odometry, we can construct a solvable sub-graph by selecting a contiguous segment of the robot trajectory (around five meters in our experiments). Subsequently, we consider all landmarks that have been seen from within the trajectory portion. Depending on the type of sensors, the landmarks can be fully observable or not. For instance, to determine the position of a landmark observed with a bearing only sensor we need two observations from two different robot positions. If we are able to measure also the range, a single observation is sufficient. Based on the odometry guess we attempt to determine the position of all observed landmarks. The landmarks whose position cannot be determined from within the local map are discarded together with their measurements. In this way, we obtain a set of factors that lead to a fully specified problem. Note that certain landmarks can be discarded from all local maps, since their position cannot be determined in any of them. This results in ignoring some of the information when approximating the initial guess. However, this information will be recovered in the final refinement stage of the algorithm where we employ a direct method on the original factors starting from the initial guess computed by our approach.

Once we have partitioned the graph into sub-problems, we solve each of them independently by using the direct methods described in Section III. We can then compute the marginal covariances of the separators by applying an algorithm based on dynamic programming described by Kaess *et al.* [24] from the corresponding blocks of $\mathbf{H}^{-1}$. To solve the local maps and to compute the marginal covariances, we use the open-source g^{2}o package [3]. Note that, due to our manifold formulation, we obtain the marginal covariances of the increments $\mathbf{\Delta x}_i$ and not of the state variables $\mathbf{x}_i$.

B. Computing Condensed Factors

In this section, we describe how to compute a new set of factors that relate the separators $\{\mathbf{x}_i\}$ and the origin $\mathbf{x}_g$, given the marginal covariances $\{\mathbf{\Sigma}_i\}$ previously computed.

We define a family of measurement functions

$$\mathbf{h}^{\text{typeOf}(\mathbf{x}_i)}(\mathbf{x}_g, \mathbf{x}_i) \overset{\text{def.}}{=} \mathbf{h}(\mathbf{x}_g, \mathbf{x}_i) \tag{10}$$

that depend on the *type* of the separators. In our SLAM examples we have two types of variables: the landmark poses that are represented as vectors in $\Re^2$ or $\Re^3$ and the robot poses that belong to either $SE2$ or $SE3$. The origin of the local maps is always selected to be a robot pose. The "virtual sensors" in our case are:

$$\mathbf{h}_i(\mathbf{x}_g, \mathbf{x}_i) = \begin{cases} \text{toVec}(\mathbf{X}_g^{-1} \cdot \mathbf{X}_i) & \text{if}(\text{typeOf}(\mathbf{x}_i) == \text{pose}) \\ \mathbf{X}_g^{-1} \cdot \mathbf{x}_i & \text{if}(\text{typeOf}(\mathbf{x}_i) == \text{landmark}) \end{cases}$$

Here, $\mathbf{X}_*$ denotes the homogeneous matrix of the transformation $\mathbf{x}_*$. Intuitively, the measurement between two poses is the relative movement between one and the other. The measurement between a landmark and a pose is the projection of the landmark in the frame of the observing pose. However, the above choices are not unique and in the general case, the user can select any measurement function that is invertible in $\mathbf{x}_i$. We selected these measurement functions to compute our condensed factors because the experiments demonstrated that they behave better than other models.

Once we know which particular measurement function to select for each separator, we need to determine the factors connecting the origin and the separators:

$$F_i = \mathbf{e}_i(\mathbf{x}_g, \mathbf{x}_i)^T \mathbf{\Omega}_i \mathbf{e}_i(\mathbf{x}_g, \mathbf{x}_i). \tag{11}$$

To this end, we recall Eq. 4 that relates measurement function and error vector through the $\boxminus$ operator: $\mathbf{e}_i(\mathbf{x}_g, \mathbf{x}_i) = \mathbf{h}(\mathbf{x}_g, \mathbf{x}_i) \boxminus \mathbf{z}_i$. The error function depends on the (known) measurement function $\mathbf{h}_i(\cdot)$ and on the unknown measurement $\mathbf{z}_i$. Since the error should be $\mathbf{0}$ at the current solution of the local map, the measurement vector at the equilibrium is:

$$\mathbf{z}_i = \mathbf{h}(\mathbf{x}_g^*, \mathbf{x}_i^*), \tag{12}$$

where $\mathbf{x}_g^*$ and $\mathbf{x}_i^*$ are the actual values of the origin and of a separator after solving the sub-problem.

To qualify the factors in Eq. 11 we still have to compute the information matrix $\mathbf{\Omega}_i$. Since the origin node is fixed, its covariance matrix is zero. Thus, only the marginal covariance of $\mathbf{x}_i$ contributes in determining $\mathbf{\Omega}_i$. The procedure outlined in the previous section gives us the covariance matrices of the increments $\mathbf{\Delta x}_i$. Hence, we need to remap them through the error function. To this end, we rewrite the error function, highlighting the contribution of the increments:

$$\mathbf{e}_i(\mathbf{x}_g, \mathbf{x}_i \boxplus \mathbf{\Delta x}_i) = \mathbf{h}(\mathbf{x}_g, \mathbf{x}_i \boxplus \mathbf{\Delta x}_i) \boxminus \mathbf{z}_i. \tag{13}$$

We then remap the marginal covariance of $\mathbf{\Delta x}_i$ by using the unscented transform [23]. We extract a set of sigma points $\{\sigma^k_{\mathbf{\Delta x}_i}\}$ from the marginal covariance $\mathbf{\Sigma}_{\mathbf{\Delta x}_i}$ of the increments $\mathbf{\Delta x}_i$ and we remap them through Eq. 13 as follows:

$$\sigma^k = \mathbf{e}_i(\mathbf{x}_g^*, \mathbf{x}_i^* \boxplus \sigma^k_{\mathbf{\Delta x}_i}). \tag{14}$$

We then compute $\mathbf{\Omega}_i$ by inverting the covariance matrix reconstructed from the projected sigma points.

The procedure outlined above allows us to determine the new factors used to describe a local map in a compact manner at a higher level of abstraction. The new factors are computed after considering the solution of a full portion of the problem and model the relationships between the origin of a local map and the separators.

VI. Experiments

We validated our approach on real-world data and performed extensive statistical tests on simulated data. On all datasets we compare our approach with the Levenberg-Marquardt (LM) implementation in the g^{2}o package. All results have been validated by both visual inspection and comparing the errors of the final solution. Real-world experiments provide evidence on the real applicability of the results. While the ground truth of simulated datasets allow us to characterize the behavior of the approaches in a more detailed way.

A. Real World Experiments

The first experiment that we describe is done on the popular Victoria-Park dataset. It was acquired with a car equipped with a laser range finder and odometer. Point landmarks are the trees in the park and are observed through laser scans. A feature extraction algorithm reports the $x-y$ location of the detected trees, in the laser frame. We will refer to this sensor model as the "Cartesian" dataset. From this dataset, we constructed a bearing only dataset, where we replaced the cartesian observations of a landmark with the corresponding bearing measurement. As shown in Figure 4a, the noise in the odometry is relatively high.

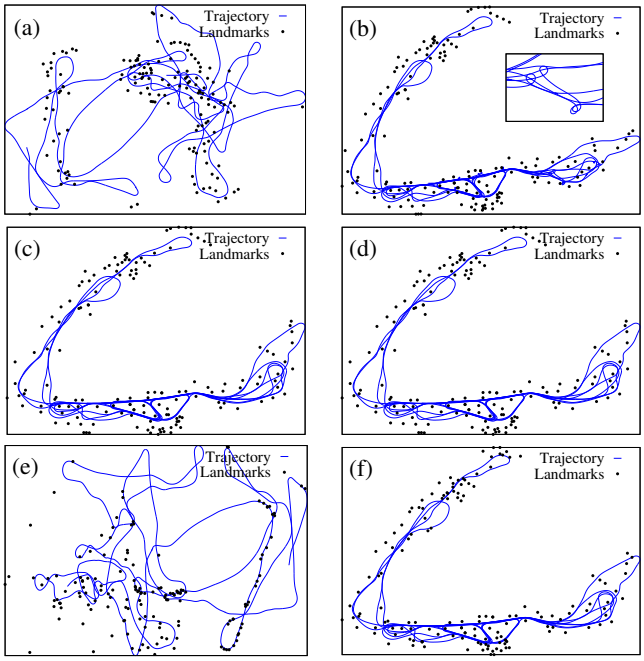

Fig. 4. The Victoria-Park dataset: the landmarks are shown as little dots, and blue curves are the robot trajectories. The initial guess is shown in (a). By using a Cartesian sensor model the direct approaches fail in batch optimization, $F = 30607$; (b). the direct approaches succeed when run incrementally, $F = 389$; (c). Our approach succeeds; (d). The bearing only dataset cannot be optimized with direct approaches neither incrementally (e) nor in the batch mode; (f) The correct map is obtained by our method.

Processing the whole Cartesian dataset with direct approaches does not give the correct solution (Fig. 4b), which can however be obtained by running the direct approaches incrementally, after inserting every 50 sequential odometry measurements (Fig. 4c). However, our approach always finds the correct solution (Fig. 4d). The solution is the same as the one computed by the direct approaches run incrementally.

The bearing-only dataset cannot be solved by direct approaches when run either in batch mode or incrementally (Fig. 4e), due to the high nonlinearities in the sensor model. Conversely, our approach succeeds (Fig. 4f).

In the second experiment, we describe the results of processing a 3D dataset acquired at the Freiburg University campus with a mobile robot equipped with a Bumblebee stereo camera. From each frame, we extracted visual features along with disparity and constructed one large bundle adjustment problem enriched with odometry information. The results of the experiments are illustrated in Figure 5. For this dataset, we considered two initial guesses: one obtained by optimizing the pose graph constructed by densely matching pair-wise observations (thus pretty accurate), and one based only on the wheel odometry. We processed this dataset with direct methods both batch and incrementally. Direct approaches always succeeded in finding the optimal solution when initialized with the good guess, while they failed in all cases starting from the bad guess. Our approach succeeded

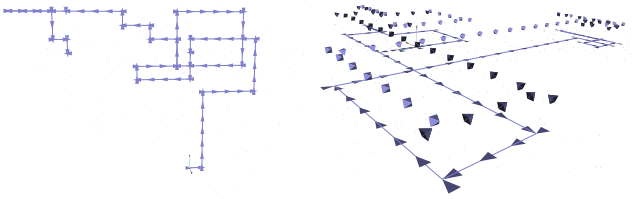

Fig. 6. Two simulated datasets, in which points represent point features, while blue lines indicate the trajectory of the robot. Left: the top view of a 2D dataset. Right: the perspective view of a 3D dataset.

in creating the local maps and determining a good initial alignment.

B. Simulated Experiments

We generated a set of synthetic 2D and 3D datasets by simulating a robot moving in a grid world and sensing point landmarks in its neighborhood. In all cases, we created a synthetic world by placing a set of landmarks in the environment and letting the robot move for increasingly long trajectories along a simulated Manhattan world. The same synthetic world was used to create different datasets: one for each sensor setup. Thus, in these experiments we tested: different trajectory lengths and different sensor modalities. Figure 6 shows the ground truth of two synthetic datasets used in our experiments.

In 2D we used point landmarks and simulated both a Cartesian sensor, similar to the one used in the Victoria Park experiment and a bearing-only sensor. In 3D we used an ideal Cartesian sensor capable of measuring the position of a landmark in the reference frame of the observer (this can model a 3D laser), a depth sensor, which measures the "depth" of points in the image plane (this can model RGBD cameras like the Kinect) and finally we used a disparity sensor suitable to model stereo cameras.

Table I shows the characteristic of the different dataset in terms of number of poses, landmarks and factors in the graph and sensors used to perceive the landmarks. For each dataset we report the value of the error $F(\cdot)$ at the minimum found by the algorithms for the different datasets. F_{id} is the theoretical minimum value obtained by running LM using the ground truth as initial guess. F_{init} represents the error of the initial guess derived from the odometry. F_{LM} is the error of the solution obtained by running 100 iterations of the LM algorithm in the g^{2}o package and F_{cond} using our approach to determine the initial guess and then running 10 iterations of standard LM. We marked in bold the cases of wrong convergence. The reader might observe that in some cases our approach did not reach the absolute minimum, but this is due to the fact that we limited the LM iterations to 10. We verified that running 100 iterations of LM results in reaching the theoretical minimum in our simulated experiments. Additionally, we observed a substantial speedup in using the condensed measurements. For a large problem of 5001 poses computing the solution with LM takes approximately 18 minutes against 9 minutes of running the condensed approach plus 10 iterations of LM. In total the generation of the condensed factors and the solution of the sparse problem took less than 4 minutes on a Core 2 Duo 2.6 Ghz using one single core. The column "# cond" reports the number of condensed factors in the global sparse problem, while the column F_{sparse} reports the initial and final error of the optimization of this sparse problem. The significant reduction of the error is possible because the constructed problem is more convex than the input one. For problems having a very small size, no condensed factors are generated. This is the case of the 2D dataset with 11 pose variables, where the solution is computed with standard LM.

As it can be seen from the table, the larger the problem becomes, the harder is for LM to converge, while our method always finds the correct minimum. Also the non-linearity of the sensor has a great effect on the convergence. In 2D, when we use a bearing only sensor and the number of constraints, increases LM fails. In 3D using a Cartesian sensor leads to the correct solution for standard approaches, while the same approach fails when using a depth or a disparity model, the former being better than the latter. Our method succeeds in all cases.

VII. Conclusions

We presented a novel approach for optimizing factor graphs obtained from SLAM or SfM problems. The algorithm is robust to noisy initial guesses and highly nonlinear sensor models. The key idea is to construct an approximation of the original problem having a larger convergence basin by computing condensed measurements from partial solutions, to determine a good initial guess. Our approach can solve problems that cannot be handled by other state-of-the-art methods. In the future, we plan to exploit the divide-and-conquer strategy of our method to take advantage of modern parallel CPU and GPU architectures.

References

[1] G. Grisetti, R. Kümmerle, C. Stachniss, U. Frese, and C. Hertzberg, "Hierarchical optimization on manifolds for online 2D and 3D mapping," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2010.

[2] K. Ni and F. Dellaert, "Multi-level submap based SLAM using nested dissection," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[3] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A general framework for graph optimization," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2011.

[4] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous Robots*, vol. 4, pp. 333–349, 1997.

[5] T. A. Davis, *Direct Methods for Sparse Linear Systems*. SIAM, 2006, part of the SIAM Book Series on the Fundamentals of Algorithms.

[6] A. Howard, M. Matarić, and G. Sukhatme, "Relaxation on a mesh: a formalism for generalized localization," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2001.

[7] T. Duckett, S. Marsland, and J. Shapiro, "Fast, on-line learning of globally consistent maps," *Autonomous Robots*, vol. 12, no. 3, pp. 287 – 300, 2002.

[8] U. Frese, P. Larsson, and T. Duckett, "A multilevel relaxation algorithm for simultaneous localisation and mapping," *IEEE Transactions on Robotics*, vol. 21, no. 2, pp. 1–12, 2005.

[9] E. Olson, J. Leonard, and S. Teller, "Fast iterative optimization of pose graphs with poor initial estimates," in *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2006, pp. 2262–2269.

[10] G. Grisetti, C. Stachniss, and W. Burgard, "Non-linear constraint network optimization for efficient map learning," *IEEE Trans. on Intelligent Transportation Systems*, 2009.

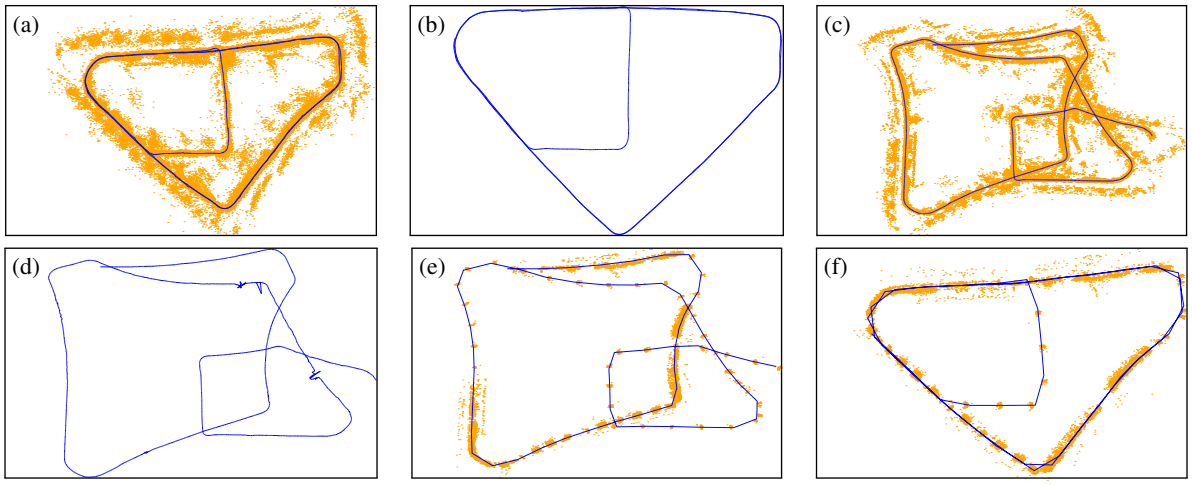

Fig. 5. The top view of our Visual SLAM dataset: the point features are displayed as little dots, the trajectory of the robot is depicted in blue. (a) The initial guess obtained by the SLAM algorithm. (b) The trajectory of the robot obtained by running LM on the good guess. (c) The noisy initial guess obtained by composing the odometry. (d) The result of the same algorithm using the noisy initial guess is even worse than the input. (e) The layout of the condensed problem solved by our approach after being initialized with the noisy initial guess. (f) The final result after the global alignment.

TABLE I

SUMMARY OF THE SIMULATED EXPERIMENTS

Dataset	# poses	# landm.	# factors	sensor	F_{init}	F_{ideal}	F_{lm}	F_{cond}	# cond. factors	F_{sparse} (init/final)
2D	11	403	1229	bearing	$3.351{\cdot}10^{7}$	533.837	555.927	593.097	-	- / -
				cartesian	24883	1656	1656	1656	-	- / -
	101	1102	10223	bearing	$4.825{\cdot}10^{8}$	8088.46	8088.46	8121.86	1031	208938 / 3845.72
				cartesian	367552	17982	17982	17982	1328	6778.08 / 594.761
	1001	2000	105399	bearing	$7.056{\cdot}10^{9}$	101483	101483	101483	15218	$2.681{\cdot}10^{6}$ / 43868.3
				cartesian	$1.267{\cdot}10^{9}$	205351	205351	205351	17348	$2.202{\cdot}10^{6}$ / 4482.67
	5001	2000	534688	bearing	$7.666{\cdot}10^{10}$	528335	$\mathbf{2.226 \cdot 10^{7}}$	528339	74437	$5.031{\cdot}10^{8}$ / 248863
				cartesian	$1.792{\cdot}10^{10}$	$1.056{\cdot}10^{6}$	$1.056{\cdot}10^{6}$	$1.056{\cdot}10^{6}$	85693	$7.792{\cdot}10^{7}$ / 26300
3D	11	175	226	depth	5339.7	123.436	123.436	123.436	16	8.69811 / 2.25886
				disparity	7947.67	126.067	126.067	126.067	16	11.6747 / 2.80877
		177	232	cartesian	2550.35	129.623	129.623	129.623	30	64.0997 / 16.4429
	101	708	1809	depth	$9.490{\cdot}10^{7}$	3008.07	3008.07	3008.07	1207	9452.95 / 330.039
				disparity	$1.308{\cdot}10^{8}$	3006.33	**183734**	3006.33	1207	12263 / 423.092
		710	1868	cartesian	553392	3226.68	3226.68	3226.68	1219	22150 / 937.877
	1001	3875	19267	depth	$5.477{\cdot}10^{10}$	43646.9	$\mathbf{3.018 \cdot 10^{7}}$	43646.9	14882	$1.631{\cdot}10^{7}$ / 5845.47
				disparity	$3.148{\cdot}10^{12}$	43551.8	$\mathbf{1.167 \cdot 10^{7}}$	43551.8	14882	$2.22{\cdot}10^{7}$ / 7361.63
			19806	cartesian	$3.636{\cdot}10^{8}$	45162.9	45162.9	45162.9	15040	$1.899{\cdot}10^{7}$ / 13897.1
	5001	4922	96659	depth	$1.5684{\cdot}10^{14}$	261146	$\mathbf{2.010 \cdot 10^{9}}$	261146	79142	$5.967{\cdot}10^{8}$ / 35200.3
				disparity	$4.518{\cdot}10^{12}$	261364	$\mathbf{3.239 \cdot 10^{8}}$	261371	79142	$6.304{\cdot}10^{8}$ / 44369.1
			99332	cartesian	$8.086{\cdot}10^{9}$	268993	268993	268993	79985	$4.818{\cdot}10^{8}$ / 81782.2

[11] L. Carlone, R. Aragues, J. Castellanos, and B. Bona, "A linear approximation for graph-based simultaneous localization and mapping," in *Proc. of Robotics: Science and Systems (RSS)*, 2011.

[12] K. Konolige, "Large-scale map-making," in *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2004.

[13] M. Montemerlo and S. Thrun, "Large-scale robotic 3-d mapping of urban structures," in *Proc. of the Int. Symposium on Experimental Robotics (ISER)*, 2004.

[14] F. Dellaert and M. Kaess, "Square Root SAM: Simultaneous localization and mapping via square root information smoothing," *Int. Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1204, Dec 2006.

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

[16] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent, "Sparse pose adjustment for 2d mapping," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[17] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzibbon, "Bundle adjustment - a modern synthesis," in *Vision Algorithms: Theory and Practice*, ser. LNCS. Springer Verlag, 2000, pp. 298–375.

[18] K. Konolige, "Sparse sparse bundle adjustment," in *Proc. of the British Machine Vision Conference (BMVC)*, 2010.

[19] Y. Jeong, D. Nister, D. Steedly, R. Szeliski, and I. Kweon, "Pushing the envelope of modern methods for bundle adjustment," in *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition (CVPR)*, 2010.

[20] J. Lim, J.-M. Frahm, and M. Pollefeys, "Online environment mapping," in *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition (CVPR)*, 2011.

[21] H. Strasdat, A. Davison, J. Montiel, and K. Konolige, "Double window optimisation for constant time visual SLAM," in *Proc. of the Int. Conf. on Computer Vision (ICCV)*, 2011.

[22] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds," *Information Fusion*, 2011.

[23] S. Julier, "The scaled unscented transformation," in *American Control Conference, 2002. Proceedings of the 2002*, vol. 6, 2002, pp. 4555 – 4559 vol.6.

[24] M. Kaess and F. Dellaert, "Covariance recovery from a square root information matrix for data association," *Journal of Robotics and Autonomous Systems, RAS*, vol. 57, pp. 1198–1210, Dec 2009.