KI – Künstliche Intelligenz manuscript No.
(will be inserted by the editor)

Lifelong Map Learning for Graph-based SLAM in Static Environments

Henrik Kretzschmar · **Giorgio Grisetti** · **Cyrill Stachniss**

Received: date / Accepted: date

Abstract In this paper, we address the problem of lifelong map learning in static environments with mobile robots using the graph-based formulation of the simultaneous localization and mapping problem. The pose graph, which stores the poses of the robot and spatial constraints between them, is the central data structure in graph-based SLAM. The size of the pose graph has a direct influence on the runtime and the memory complexity of the SLAM system and typically grows over time. A robot that performs lifelong mapping in a bounded environment has to limit the memory and computational complexity of its mapping system. We present a novel approach to prune the pose graph so that it only grows when the robot acquires relevant new information about the environment in terms of expected information gain. As a result, our approach scales with the size of the environment and not with the length of the trajectory, which is an important prerequisite for lifelong map learning. The experiments presented in this paper illustrate the properties of our method using real robots.

Keywords SLAM · Mapping · Expected Information Gain

1 Introduction

Maps of the environment are needed for a wide range of robotic applications, including transportation tasks and many service robotic applications. Therefore, learning maps is regarded as one of the fundamental problems in mobile robotics. In the last two decades, several effective approaches for learning maps have been developed. The graph-based formulation of the simultaneous localization and mapping (SLAM) problem models the poses of the robot as nodes in a graph. Spatial constraints between poses resulting from observations or from odometry are encoded in the edges between the nodes. Graph-based approaches such as [6, 9, 19], which are probably the most efficient techniques at the moment, typically marginalize out the features (or local grid maps) and reduce the mapping problem to trajectory estimation without prior map knowledge. Therefore, the underlying graph structure is often called the pose graph.

The majority of the approaches, however, assumes that map learning is carried out as a preprocessing step and that the robot later on uses the model for tasks such as localization and path planning. A robot that is constantly updating the map of its environment has to address the so-called lifelong SLAM problem. This problem cannot be handled well by most graph-based techniques since the complexity of these approaches grows with the length of the trajectory. As a result, the memory as well as the computational requirements grow over time and therefore these methods cannot be applied to lifelong SLAM.

The contribution of this paper is a novel approach that enables graph-based SLAM approaches to operate in the context of lifelong map learning in static scenes. Our approach is orthogonal to the underlying graph-based mapping technique and applies an entropy-driven strategy to prune the pose graph while minimizing the loss of information. This becomes especially important when re-traversing already mapped areas. As a result, our approach scales with the size of the environment and not with the length of the trajectory. It should be noted that not only long-term mapping systems benefit from our method. Even traditional mapping systems are able to compute a map faster since less resources are claimed and less comparisons between observations are needed to solve the data association problem. We furthermore illustrate that the resulting grid maps are less blurred compared to the maps built without our approach.

H. Kretzschmar · G. Grisetti · C. Stachniss
University of Freiburg, Dept. of Computer Science,
Georges-Koehler-Allee 79, D-79110 Freiburg, Germany
E-mail: {kretzsch, grisetti, stachnis}@informatik.uni-freiburg.de

2 Related Work

There is a large variety of SLAM approaches available in the robotics community. Common techniques apply extended and unscented Kalman filters [11, 13], sparse extended information filters [3, 23], particle filters [8, 15], and graph-based, least squares error minimization approaches [6, 9, 14, 19].

Graph-based SLAM approaches to estimate maximum-likelihood maps are often regarded as the most effective means to reduce the error in the pose graph. Lu and Milios [14] were the first to refine a map by globally optimizing the system of equations to reduce the error introduced by constraints. Since then, a large variety of approaches for minimizing the error in the constraint network have been proposed. Duckett et al. [2] use Gauss-Seidel relaxation. The multi-level relaxation (MLR) approach by Frese et al. [6] apply relaxation at different spatial resolutions. Given a good initial guess, it yields very accurate maps particularly in flat environments. Folkesson and Christensen [5] define an energy function for each node and try to minimize it. Thrun and Montemerlo [22] apply variable elimination techniques to reduce the dimensionality of the optimization problem. Olson et al. [19] presented a fast and accurate optimization approach which is based on the stochastic gradient descent (SGD). Compared to approaches such as MLR, it still converges from a worse initial guess. Based on Olson's optimization algorithm, Grisetti et al. [9] proposed a different parameterization of the nodes in the graph. The tree-based parameterization yields a significant boost in performance. In addition to that, the approach can deal with arbitrary graph topologies. The approach presented in this paper is built upon the work by Grisetti et al. [9].

Most graph-based approaches available today do not provide means to efficiently prune the pose graph, that has to be corrected by the underlying optimization framework. Most approaches can only add new nodes or apply a rather simple decision whether to add a new node to the pose graph or not (such as the question of how spatially close a node is to an existing one). However, there are some notable exceptions: Folkesson and Christensen [5] combine nodes into so-called star nodes which then define rigid local submaps. The method applies delayed linearization of local subsets of the graph, permanently combining a set of nodes in a relative frame. Related to that, Konolige and Agrawal [12] subsample nodes for the global optimization and correct the other nodes locally after global optimization.

Other authors considered the problem of updating a map upon changes in the environment. For example, Biber and Duckett [1] propose an approach to update an existing model of the environment. They use five maps on different time scales and incorporate new information by forgetting old information. Related to that, Stachniss and Burgard [20] learn clusters of local map models to identify typical states of the environment. Both approaches focus on modeling changes in the environment but do not address the full SLAM problem since they require an initial map to operate. The approach presented in this paper further improves the node reduction techniques for graph-based SLAM by estimating how much the new observation will change the map. It explicitely considers the expected information gain of observations to decide whether the corresponding nodes should be removed from the graph or not.

In the remainder of this paper, we will first introduce the basic concept of pose graph optimization originally presented in [9]. In Section 4, we describe our contribution to information-driven node reduction. In Section 5, we finally presents our experimental evaluation.

3 Map Learning using Pose Graphs

The graph-based formulation of the SLAM problem models the poses of the robot as nodes in a graph (a so-called pose graph). Spatial constraints between poses resulting from observations or from odometry are encoded in the edges between the nodes. Most approaches to graph-based SLAM focus on estimating the most-likely configuration of the nodes and are therefore referred to as maximum-likelihood (ML) techniques. The approach presented in this paper also applies an optimization framework that belongs to this class of methods.

3.1 Problem Formulation

The goal of graph-based ML mapping algorithms is to find the configuration of the nodes that maximizes the likelihood of the observations. Let $\mathbf{x} = (x_1 \ \cdots \ x_n)^T$ be a vector of parameters which describes a configuration of the nodes. Let δ_{ji} and Ω_{ji} be respectively the mean and the information matrix of an observation of node j seen from node i. Let $f_{ji}(\mathbf{x})$ be a function that computes a zero noise observation according to the current configuration of the nodes j and i.

Given a constraint between node j and node i, we can define the error e_{ji} introduced by the constraint as

$$e_{ji}(\mathbf{x}) = f_{ji}(\mathbf{x}) - \delta_{ji} \tag{1}$$

as well as the residual $r_{ji} = -e_{ji}(\mathbf{x})$. Let $\mathscr{C}$ be the set of pairs of indices for which a constraint δ exists. The goal of a ML approach is to find the configuration of the nodes that minimizes the negative log likelihood of the observations. Assuming the constraints to be independent, this can be written as

$$\mathbf{x}^* = \operatorname*{argmin}_{\mathbf{x}} \sum_{\langle j,i \rangle \in \mathscr{C}} r_{ji}(\mathbf{x})^T \Omega_{ji} r_{ji}(\mathbf{x}). \tag{2}$$

3.2 Map Optimization

To solve Eq. (2), different techniques can be applied. Our work applies the approach of Grisetti et al. [9] which is an extention of the work of Olson et al. [19]. Olson et al. [19] propose to use a variant of the preconditioned stochastic gradient descent (SGD) to compute the most likely configuration of the nodes in the network. The approach minimizes Eq. (2) by iteratively selecting a constraint and by moving the nodes of the pose graph in order to decrease the error introduced by the selected constraint. Compared to the standard formulation of gradient descent, the constraints are not optimized as a whole but individually. The nodes are updated according to the following equation:

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \lambda \cdot \mathbf{H}^{-1} J_{ji}^T \Omega_{ji} r_{ji} \tag{3}$$

Here, $\mathbf{x}$ is the set of variables describing the locations of the poses in the network and $\mathbf{H}^{-1}$ is a preconditioning matrix. J_{ji} is the Jacobian of f_{ji}, Ω_{ji} is the information matrix capturing the uncertainty of the observation, r_{ji} is the residual, and λ is the learning rate which decreases with the iteration. For a detailed explanation of Eq. (3), we refer the reader to [9] or [19].

In practice, the algorithm decomposes the overall problem into many smaller problems by optimizing subsets of nodes, one subset for each constraint. Whenever a solution for one of these subproblems is found, the network is updated accordingly. Obviously, updating the constraints one after each other can have antagonistic effects on the corresponding subsets of variables. To avoid infinite oscillations, one uses the learning rate λ to reduce the fraction of the residual which is used for updating the variables. This makes the solutions of the different sub-problems converge asymptotically to an equilibrium point which is the solution reported by the algorithm.

3.3 Tree Parameterization for Efficient Map Optimization

The poses $\mathbf{p} = \{p_1, \ldots, p_n\}$ of the nodes define the configuration of the graph. The poses can be described by a vector of *parameters* $\mathbf{x}$ such that a bidirectional mapping between $\mathbf{p}$ and $\mathbf{x}$ exists. The parameterization defines the subset of variables that are modified when updating a constraint in SGD. An efficient way of parameterizing the nodes is to use a tree. To obtain that tree, we compute a spanning tree from the pose graph. Given such a tree, one can define the parameterization for a node as

$$x_i = p_i - p_{\text{parent}(i)}, \tag{4}$$

where $p_{\text{parent}(i)}$ refers to the parent of node i in the spanning tree. As shown in [9] this approach can dramatically speed up the convergence rate compared to the method of Olson et al. [19].

The technique described so far is typically executed as a batch process but there exists also an incremental variant [10] which performs the optimization only on the portions of the graph which are affected by the introduction of new constraints. It is thus able to re-use the previously generated solutions to compute the new one.

3.4 The SLAM Front-end

The approach briefly described above only focuses on correcting a pose graph *given* all constraints and is often referred to as the SLAM back-end. In contrast to that, the SLAM front-end aims at extracting the constraints from sensor data. In this paper, we build our work upon the SLAM front-end described in the Ph.D. thesis of Olson [17]. We refer to this technique as "without graph reduction". The front-end generates a new node every time the robot travels a minimum distance. Every node is labeled with the laser-scan acquired at that position. Constraints between subsequent nodes are generated by pairwise scan-matching. Every time a new node is added, the front-end seeks for loop closures with other nodes. It therefore approximates the conditional covariances of all nodes with respect to the newly added node using the technique described in [25]. Once these covariances are computed, the approach selects a set of candidate loop closures using the χ^2 test. The corresponding edges are then created by aligning the current observations with the ones stored in each candidate loop closing node determined by the previous step. In addition to that, the front-end applies an outlier rejection technique based on spectral clustering [18] to reduce the risk of wrong matches.

The contribution of this paper is a technique that "sits" between the SLAM front-end and the optimizer, the SLAM back-end, in order to enable a robot to perform lifelong map learning in static worlds. It allows for removing redundant nodes by considering the expected information gain of the observations. As we will show in the remainder of this paper, our method allows for efficient map learning especially in the context of frequent re-traversals of previously mapped areas. In addition to that, we illustrate that this technique improves the map quality when learning grid maps and that it generates sharp boundaries between free and occupied spaces.

4 Our Approach to Lifelong Map Learning

In the context of lifelong map learning, a robot cannot add new nodes to the graph whenever it is re-entering already visited terrain. The key idea of our approach is to prune the graph structure to limit the number of nodes. Most of the

existing approaches to graph reduction simply consider the position of a potential new node and do not integrate it into the graph if it is spatially close to an existing node [4, 12]. In this paper, we propose a different, information-driven approach to node reduction. In contrast to considering poses only, we estimate the expected amount of information which an observation contributes to the belief of the robot.

Most robotics systems that apply graph-based mapping approaches eventually convert the pose graph into another data structure to represent the environment for tasks such as path planning. In this context, popular models are occupancy grid maps [16, 24] and feature maps. Therefore, we consider the effects of the node reduction technique on the resulting occupancy grid map.

4.1 Information-Theoretic Node Reduction

Our approach uses the expected information gain, which is defined as the expected reduction of uncertainty in the belief of the robot caused by an observation. Entropy H is a general measure for uncertainty in the belief over a random variable x. For a discrete probability distribution, entropy is given by

$$H\left(p(x)\right) = -\sum_{x} p(x)\log_2 p(x). \tag{5}$$

In theory, the entropy has to be computed over the full SLAM posterior, thus taking into account the pose and the map uncertainty. In our case, however, we aim at discarding observations when *re-traversing known areas*. If the robot performs a re-traversal, it has already identified loop-closing constraints that caused a pose correction carried out by the optimization framework. Therefore, the nodes which are discarded typically have a minor effect on the pose uncertainty itself which is why the relevant part of the uncertainty is given by the map uncertainty. To put this in other words, the optimization framework is a maximum likelihood estimator and its estimate is the node arrangement. Given this arrangement, the map can be computed directly and thus the entropy as well. For a grid map, the entropy is given by

$$\begin{aligned} H(p(m)) &= \sum_{c\in m} H(p(c)) \qquad (6)\\ &= -\sum_{c\in m}\Big(p(c)\log_2 p(c) + p(\neg c)\log_2 p(\neg c)\Big), \end{aligned}$$

where c refers to the individual cells of the grid map m, and $p(c)$ refers to the estimated probability that the cell c is occupied [21]. Based on the entropy, we can define the information gain of an observation z_i with respect to a grid cell c as

$$I_c(z_i) = H\Big(p(c \mid z_{1:t} \setminus \{z_i\})\Big) - H\Big(p(c \mid z_{1:t})\Big), \tag{7}$$

where $z_{1:t}$ refers to the set of all laser measurements in the graph which have not yet been removed. Consequently, for a grid cell c, the expected information gain of an observation z_i is given by

$$\begin{aligned} \mathbb{E}[I_c(z_i)] = {} & I_c(z_i = \text{occupied})\, p(c \mid z_{1:t} \setminus \{z_i\}) \qquad (8)\\ & + I_c(z_i = \text{free})\, p(\neg c \mid z_{1:t} \setminus \{z_i\}). \qquad (9) \end{aligned}$$

Finally, summing over all grid cells yields the expected information gain of an observation z_i with respect to the entire grid map:

$$\mathbb{E}[I(z_i)] = \sum_{c\in m} \mathbb{E}[I_c(z_i)] \tag{10}$$

A key idea of our approach is to discard all observations and the corresponding nodes in the graph whose expected information gain is smaller than a threshold. Note that our approach does not make its decision for the most recent observation only. In contrast, it also considers old observations for removal, i. e., in each step, our algorithm considers the expected information gain of all measurements $z_{1:t}$. In practice, however, the sensor of the robot only has a limited range. We therefore compute the expected information gain only for those nodes which were recorded in the vicinity of the current robot position. We remove observations until the one with the lowest expected information gain has a value greater than the threshold.

In this section, we described how to identify nodes in the graph corresponding to measurements that can be discarded minimizing the loss of relevant information. In the next section, we describe how to prune the graph while preserving most of the information encoded in the edges.

4.2 Updating the Graph

To remove a node i from the graph structure while preserving the global behavior of the error function, we need to augment the graph with edges between each pair of neighbors of the node i. This is a well known effect of the marginalization of Gaussian distributions.

When removing a node which has k neighbors, the total number of edges can grow by up to $\frac{1}{2}k(k-1)-k$. This introduces a complexity which is not suited for lifelong map learning. An illustration of the graph update is depicted in the left and middle illustration of Figure 1.

Therefore, we propose an alternative way of removing a node which is an approximate solution but which does not increase the size of the graph. Let i be the node to be removed and N be the set of neighbors of i. The key idea of our method is to select one node $j \in N$ and merge the information of all edges connecting i into existing as well as new edges connecting j. This results in removing all k edges at i

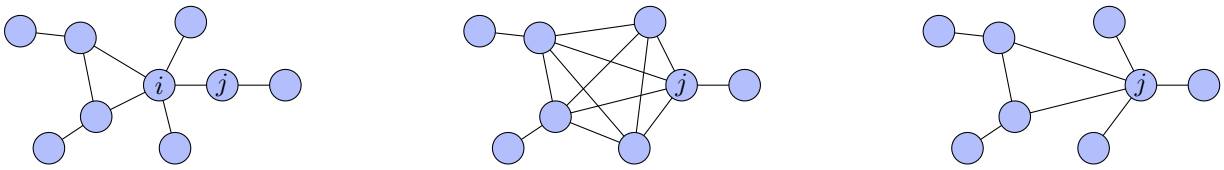

Fig. 1 Left: Graph built according to the observations. Middle: Exactly marginalized but densely connected graph after removing node i. Right: Approximate solution obtained by collapsing the edges at i into node j.

and in adding $(k-1)$ edges between j and $N\backslash\{j\}$. If this results in two edges connecting the same two nodes j and one of its neighbors, these edges can directly be merged (see also Figure 1, right). Consequently, the overall number of edges always decreases at least by 1.

Merging the information encoded in the edges can be done in a straight forward manner since the edges encode Gaussian constraints. Thus, concatenating two constraints with means δ_{ji} and δ_{kj} and information matrices Ω_{ji} and Ω_{kj} is done by

$$\delta_{ki} = \delta_{ji} \oplus \delta_{kj} \tag{11}$$

$$\Omega_{ki} = \left(\Omega_{ji}^{-1} + \Omega_{kj}^{\prime -1}\right)^{-1}, \tag{12}$$

where $\Omega_{kj}^{\prime -1}$ is the covariance matrix obtained by applying the transformation δ_{ji} to Ω_{kj}^{-1}.

Similar to that, merging two constraints δ_{ij}^1 and δ_{ij}^2 between nodes j and i is done by

$$\delta_{ji} = \Omega_{ji}^{-1}(\Omega_{ji}^{(1)}\delta_{ji}^{(1)} + \Omega_{ji}^{(2)}\delta_{ji}^{(2)}) \tag{13}$$

$$\Omega_{ji} = \Omega_{ji}^{(1)} + \Omega_{ji}^{(2)}. \tag{14}$$

To collapse a node i into one of its neighbors, one could select, in theory, an arbitrary node $j \in N$. However, the selection of the node j can have a significant influence on the resulting map that will be obtained. The reason for that is the underlying optimization framework. Most existing approaches assume Gaussian observations (the edges represent Gaussians) although this assumption may not hold in practice. In addition to that, some optimization systems assume roughly spherical covariances to exhibit maximum performance. Thus, it is desirable to avoid long edges to limit the effect of linearization errors. Our approach therefore considers all neighbors $j \in N$ and selects the one such that the sum of the lengths of all edges in the resulting graph is minimized:

$$j^* = \operatorname*{argmin}_{j \in N} \sum_{e \in \mathcal{G}(i \to j)} length(e), \tag{15}$$

In Eq. (15), $\mathcal{G}(i \to j)$ refers to the graph that results when collapsing node i into node j. Furthermore, $length(e)$ refers to the norm of the translational part of e. Note that in practice, only the edges at i and those at j need to be involved in the computation and not the entire graph.

4.3 Gamma Index

To evaluate how densely connected a pruned pose graph is in practice, we will evaluate real world data sets using the so-called gamma index [7]. The gamma index (γ) is a measure of connectivity of a graph. It is defined as the ratio between the number of existing edges and the maximum number of possible edges. It is given by

$$\gamma = \frac{e}{\frac{1}{2}v(v-1)}, \tag{16}$$

where e is the number of edges and v is the number of nodes in the graph. The gamma index varies from 0 to 1, where 0 means that the nodes are not connected at all and 1 means that the graph is complete.

As we will show in the experiments, our approach leads to a small and more or less constant gamma index, below 0.01 (in all our datasets). In contrast to that, the sound pruning strategy tends towards much higher gamma values.

5 Experimental Evaluation

We carried out the experimental evaluation of this work at the University of Freiburg using a real ActivMedia Pioneer-2 robot equipped with a SICK laser range finder and running CARMEN. In addition to that, we considered a dataset that is frequently used in the robotics community to evaluate SLAM algorithms, namely the Intel Research dataset provided by Dirk Hähnel.

5.1 Runtime

The experiments in this section are designed to show that our approach to informed graph pruning is well suited for robot mapping – for lifelong map learning as well as for standard SLAM. In the first set of experiments, we present an analysis of how the graph structure and thus the runtime increases when a robot constantly re-traverses already mapped areas. We compare the results of a graph-based optimization approach [9] in combination with a re-implementation of Olson's SLAM front-end [17] to our novel method (using the same front-end and back-end).

In the experiment presented in Figure 2, the robot was constantly re-visiting already known areas. It traveled forth

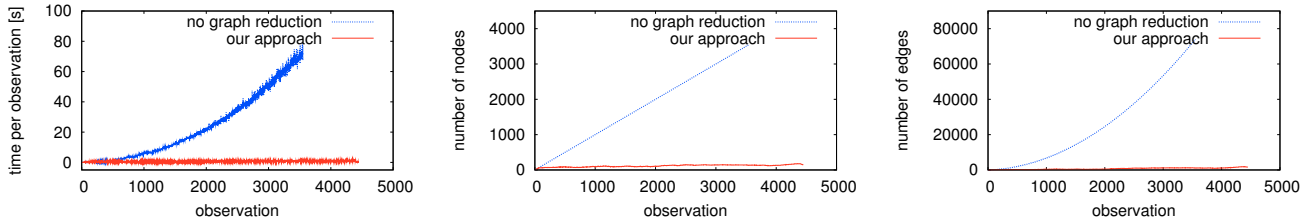

Fig. 2 Typical results of an experiment in which the robot moves in already visited areas. Left: Runtime per observation for the standard approach (blue) and for our method (red). Middle and right: Number of nodes and edges in the graph for both methods. Due to time reasons, the experiment for the standard approach was aborted after around 3500 observations.

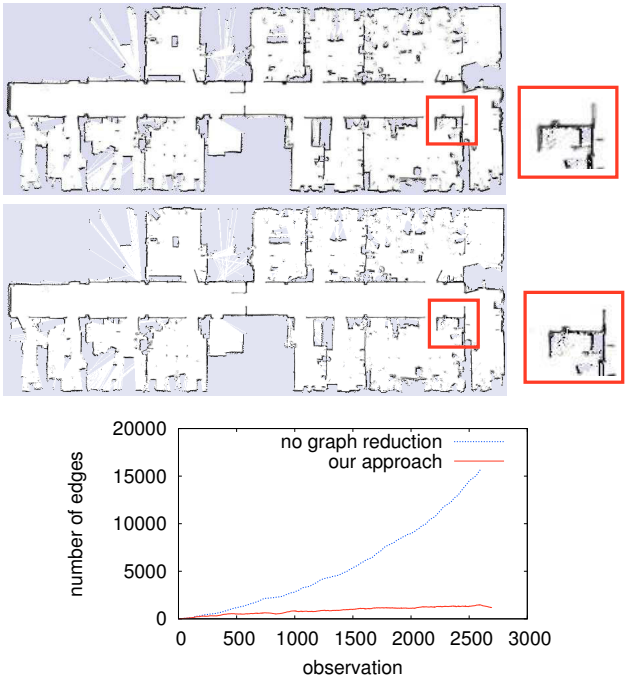

Fig. 3 Results obtained by a robot moving at Freiburg University, building 079, repeatedly visiting the individual rooms and the corridor. Top map image: standard approach without graph pruning. Bottom map image: our approach.

and back 100 times in an approximately 20 m long corridor in our lab environment. This behavior leads to a runtime explosion for the standard approach. Please note that this is not caused by the underlying optimization framework (which is executed in a few milliseconds) but by the SLAM front-end that looks for constraints between the nodes in the graph considering all previously recorded scans. In contrast to that, our approach keeps the number of nodes in the graph more or less constant and thus avoids the runtime explosion.

In an additional experiment, the robot repeatedly visited different rooms and the corridor in our lab. Figure 3 shows the resulting maps as well as the effect of the graph sparsification. In sum, this experiment yields similar results than in the previous experiment.

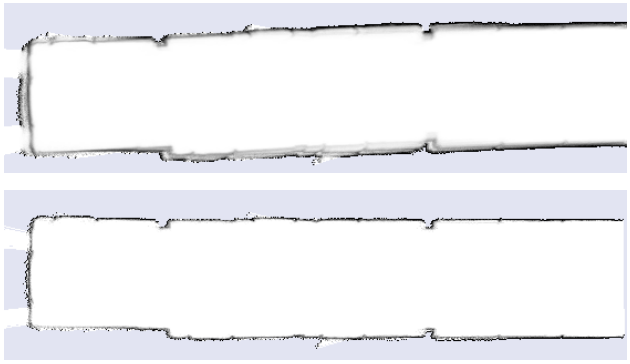

Fig. 4 Robot moving 100 times forth and back in the corridor. Standard grid-based mapping approaches (top) tend to generate thick and blurred walls whereas our graph sparsification (bottom) does not suffer from this issue.

5.2 Improved Grid Map Quality by Information-driven Graph Sparsification

The second experiment is designed to illustrate that our approach has a positive influence on the quality of the resulting grid map. In contrast to feature-based approaches, grid maps have one significant disadvantage when it comes to lifelong map learning. Whenever a robot re-enters a known region and uses scan-matching, the chance of making a small alignment error is nonzero. After the first error, the probability of making further errors increases since the map the robot aligns its observation with already has a (small) error. In the long run, this is likely to lead to divergence or at least to artificially thick walls and obstacles.

This effect, however, is significantly reduced when applying our graph sparsification technique since scans are only maintained as long as they provide relevant information, otherwise, they are discarded. To illustrate this effect, consider Figure 4. The top image shows the result of 100 corridor traversals without graph sparsification. The thick and blurred walls as a result of the misaligned poses are clearly visible. In contrast to that, the result obtained by our approach does not suffer from this problem (bottom image). The same effect can also be observed in the magnified areas in Figure 3.

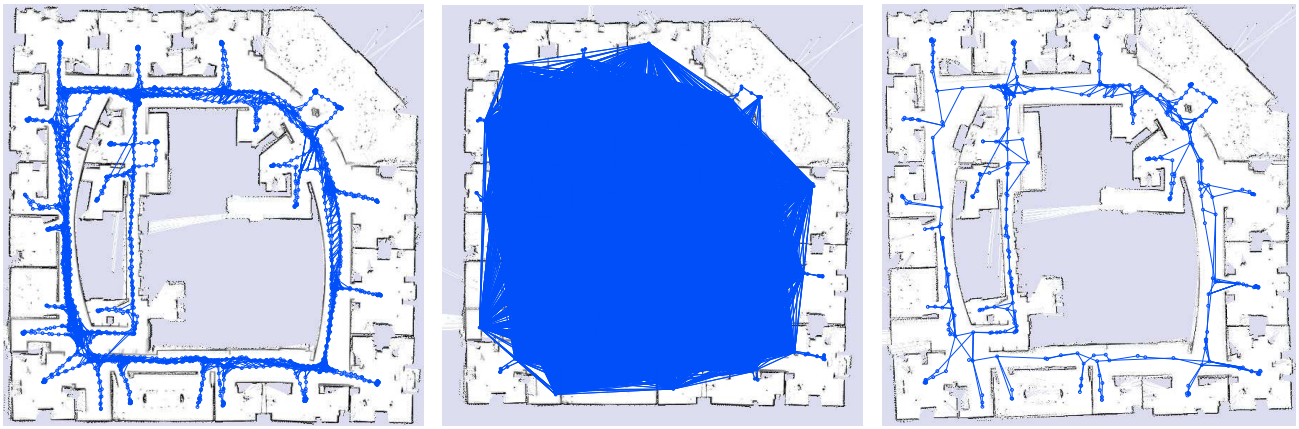

Fig. 6 Map and graph obtained from the Intel Research Lab dataset by using the standard (left) as well as by full marginalization (middle) and by using our approach (right). Standard approach: 1802 nodes, 3916 edges, full marginalization: 349 nodes, 13052 edges, our approach with approximate marginalization: 354 nodes, 559 edges.

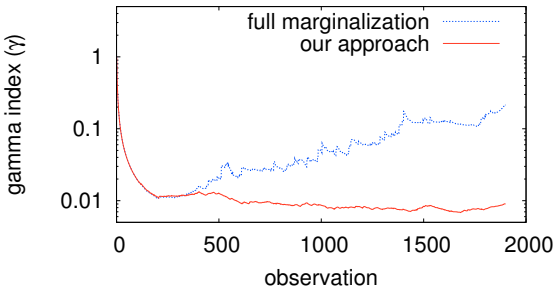

Fig. 5 Evolution of the gamma index (Intel Research Lab).

5.3 Approximate Graph Update

We furthermore compared the effects of our approximate graph update routine versus full marginalization of the nodes (see Figure 1 for an illustration). As discussed in Section 4.2, our node removal technique is guaranteed to also decrease the number of edges in the graph. In contrast to this, full marginalization typically leads to densely connected graphs.

This effect can be observed in real world data such as the Intel Research Lab, see Figure 5. This figure shows the gamma indices of both graphs. As can be seen, the graph structure obtained with our approach is and stays comparably sparse. The corresponding graphs as well as the graph obtained by the standard approach are depicted in Figure 6.

This sparsity achieved by our approach has two advantages: First, the underlying optimization method depends linearly on the number of edges in the graph. Thus, having less edges results in a faster optimization. Second, after multiple node removals using full marginalization, it is likely that also spatially distant nodes are connected via an edge. As mentioned in Section 5.2, these long distant edges can be suboptimal for the underlying optimization engine (as this is the case for [9]).

6 Conclusion

In this paper, we presented a novel approach that allows for lifelong map learning in static scenes. It is designed for mobile robots that use a graph-based framework to solve the simultaneous localization and mapping problem. By considering the expected information gain of observations, our method removes redundant information from the graph and in this way keeps the size of the pose-constraint network constant as long as the robot traverses already mapped areas. We introduce an approximate way to prune the graph structure that enables us to limit the complexity and allows for highly efficient robotic map learning. The approach has been implemented and thoroughly tested with real robot data. We provided real world experiments and considered standard benchmark datasets used in the SLAM community to illustrate the advantages of our methods.

Note that even though this paper describes only 2D experiments generated based on a 2D implementation of the work, the extension to 3D should be straightforward. Given a local 3D grid (or a more efficient representation such as an octree), the entropy and thus the expected information gain can be computed in the same way. Furthermore, the approximate marginalization is directly applicable to any kind of constraint network. Therefore, we believe that the approach can be directly applied to 3D data even though we have not done this so far.

Acknowledgements We would like to thank Dirk Hähnel for providing the Intel Research Lab dataset. This work has partly been supported by the German Research Foundation (DFG) under contract number SFB/TR-8 and by the European Commission under contract number FP7-ICT-231888-EUROPA.

References

1. P. Biber and T. Duckett. Dynamic maps for long-term operation of mobile service robots. In *Proc. of Robotics: Science and Systems (RSS)*, pages 17–24, 2005.
2. T. Duckett, S. Marsland, and J. Shapiro. Fast, on-line learning of globally consistent maps. *Autonomous Robots*, 12(3):287 – 300, 2002.
3. R. Eustice, H. Singh, and J. Leonard. Exactly sparse delayed-state filters. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2428–2435, 2005.
4. R. Eustice, H. Singh, and J. Leonard. Exactly Sparse Delayed-State Filters for View-Based SLAM. *IEEE Transactions on Robotics*, 22(6):1100–1114, 2006.
5. J. Folkesson and H. Christensen. Graphical slam - a self-correcting map. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2004.
6. U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Transactions on Robotics*, 21(2):1–12, 2005.
7. W. L. Garrison and D. F. Marble. A prolegomenon to the forecasting of transportation devel., 1965.
8. G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.
9. G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proc. of Robotics: Science and Systems (RSS)*, 2007.
10. G. Grisetti, D. L. Rizzini, C. Stachniss, E. Olson, and W. Burgard. Online constraint network optimization for efficient maximum likelihood map learning. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2008.
11. S. Julier, J. Uhlmann, and H. Durrant-Whyte. A new approach for filtering nonlinear systems. In *Proc. of the American Control Conference*, pages 1628–1632, 1995.
12. K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008.
13. J. Leonard and H. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(4):376–382, 1991.
14. F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.
15. M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 1985–1991, 2003.
16. H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 116–121, St. Louis, MO, USA, 1985.
17. E. Olson. *Robust and Efficient Robotic Mapping*. PhD thesis, MIT, Cambridge, MA, USA, June 2008.
18. E. Olson, M. Walter, J. Leonard, and S. Teller. Single cluster graph partitioning for robotics applications. In *Proceedings of Robotics Science and Systems*, pages 265–272, 2005.
19. 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)*, pages 2262–2269, 2006.
20. C. Stachniss and W. Burgard. Mobile robot mapping and localization in non-static environments. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, pages 1324–1329, 2005.
21. C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, pages 65–72, Cambridge, MA, USA, 2005.
22. S. Thrun and M. Montemerlo. The graph SLAM algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403, 2006.
23. S. Thrun, Y. Liu, D. Koller, A. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Int. Journal of Robotics Research*, 23(7/8):693–716, 2004.
24. S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, USA, 2005.
25. G. D. Tipaldi, G. Grisetti, and W. Burgard. Approximated covariance estimation in graphical approaches to slam. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.