Simultaneous Localization and Mapping (SLAM) with Multimodal Probability Distributions

Max Pfingsthorn and Andreas Birk

The International Journal of Robotics Research published online 8 October 2012
DOI: 10.1177/0278364912461540

The International Journal of
Robotics Research
0(0) 1–29

DOI: 10.1177/0278364912461540
ijr.sagepub.com

Simultaneous localization and mapping with multimodal probability distributions

Max Pfingsthorn and Andreas Birk

Abstract

Simultaneous Localization and Mapping (SLAM) has focused on noisy but unique data associations resulting in linear Gaussian uncertainty models. However, a unique decision is often not possible using only local information, giving rise to ambiguities that have to be resolved globally during optimization. To solve this problem, the pose graph data structure is extended here by multimodal constraints modeled by mixtures of Gaussians (MoG). Furthermore, optimization methods for this novel formulation are introduced, namely (a) robust iteratively reweighted least squares, and (b) Prefilter Stochastic Gradient Descent (SGD) where a preprocessing step determines globally consistent modes before applying SGD. In addition, a variant of the Prefilter method (b) is introduced in form of (c) Prefilter Levenberg–Marquardt. The methods are compared with traditional state-of-the-art optimization methods including (d) Stochastic Gradient Descent and (e) Levenberg–Marquardt as well as (f) Particle filter SLAM and with (g) an optimal exhaustive algorithm. Experiments show that ambiguities significantly impact state-of-the-art methods, and that the novel Prefilter methods (b) and (c) perform best. This is further substantiated with experiments using real-world data. To this end, a method to generate MoG constraints from a plane-based registration algorithm is introduced and used for 3D SLAM under ambiguities.

Keywords

SLAM, pose graph, registration, ambiguity, multimodal distribution, mixture of Gaussians

1. Introduction

Any approach to Simultaneous Localization and Mapping (SLAM) clearly requires some form of registration in the widest sense, i.e. the estimation of spatial relations between sensor readings of the environment at different locations. Examples include the wide range of techniques for scan matching or the spatial association of landmarks. In graph-based methods (Lu and Milios, 1997; Golfarelli et al., 2001; Dellaert, 2005; Frese et al., 2005; Olson et al., 2006; Grisetti et al., 2007c, 2010; Konolige et al., 2010), which are discussed in more detail in Section 1.2, these spatial relations are represented in edges. Existing approaches to graph-based SLAM are very efficient but they suffer from what we believe to be a fundamental drawback, namely the assumption that the registration represented in each edge gives a noisy but in principle correct and unique solution. More precisely, the spatial relation is assumed to be drawn from a unimodal distribution and it is modeled as such in subsequent probabilistic processing.

When the robot moves from A to B and acquires sensor data s_A and s_B in these locations, there are many possible reasons why a registration of s_A and s_B may not lead to a unique solution. One fundamental issue is ambiguity in the environment. Consider, for example, a hallway with a repetitive pattern of doors or lights on the ceiling or a corridor intersection. Any form of processing of the observed sensor data can only resolve the robot motion up to the distance to the nearest door in the hallway case, but it would be impossible to estimate which door seen in the previous observation is the currently nearest one. Similarly, it is possible to know the relative position to a corridor entrance in the symmetric corridor intersection case, but not which exact corridor entrance it is. Thus, in these ambiguous cases, the resulting probability distribution contains a discrete number of rather pronounced local maxima, one per possible robot motion. A traditional sensor data registration method

Jacobs University Bremen, School of Engineering and Science, Bremen, Germany

Corresponding author:
Max Pfingsthorn, Jacobs University Bremen, School of Engineering and Science, Campus Ring 1, D-28759 Bremen, Germany.
Email: m.pfingsthorn@jacobs-university.de

may either just randomly choose one of these modes and report it, or report a result with an exaggerated uncertainty. In these examples of translational and rotational ambiguity, it is impossible to compute a single accurate motion estimate by only considering data within the observation pair being registered. Other possible sources for ambiguities include the presence of changes in the environment over time, occlusions, limited sensor range, or limitations in the concrete registration methods themselves. An illustrative example of a registration process with ambiguous sensor data is presented in Figure 1, showing the corridor intersection case from above.

It is hence desirable to use multimodal distributions in order to model these cases in SLAM. Each mode in such a distribution represents a candidate spatial transformation the robot may have undergone. Please note that in this paper, we use the word 'multimodal' to refer to a probability density function with multiple modes (local maxima), and not to the use of multiple sensor modalities.

One may argue that in these ambiguous cases, particle-filter-based SLAM (Murphy, 1999; Doucet et al., 2000; Hahnel et al., 2003; Montemerlo and Thrun, 2003; Grisetti et al., 2007b) is the ideal solution as it theoretically can handle arbitrarily shaped distributions. Although this is correct in theory, it is much more difficult in practice, as also substantiated by experiments presented in Section 3. The main challenge is to limit the amount of particles to a reasonable number to achieve good efficiency while also maintaining good coverage of the probability density (Frese, 2006; Grisetti et al., 2007b). Typical numbers of particles used in the SLAM literature range from a few dozen to a few hundreds; examples include 20 (Fairfield et al., 2006; Tomono, 2007), 30–40 (Stachniss et al., 2005), 20–160 (Welle et al., 2010), 50 (Kuemmerle et al., 2009), up to 100 (Grisetti et al., 2005), 100–200 (Marks et al., 2009), 200 (Schroeter and Gross, 2008), 250 (Koenig et al., 2008), 400 (Barkby et al., 2009), and 500 particles (Elinas et al., 2006; Fairfield et al., 2007). Recent work also attempts to reduce the number of required particles or, conversely, to achieve better results with the same number of particles. P-SLAM (Chang et al., 2006, 2007) for example is designed to predict environmental structures to work with less particles. One of the most interesting contributions along this direction is Grisetti et al. (2007b), where large maps are build with at most 80 particles. This is achieved by using scan matching to compute the proposal distribution, which again introduces the assumption of a unique registration result and thus biases the potentially multimodal particle distribution towards unimodality.

Stachniss et al. (2007) identified the problem of multimodality in particle-filter-based SLAM and briefly investigated it. Their results clearly show that multimodal registration results, called proposal distributions in particle filter terms, do occur and can have a negative impact on the final mapping result if not taken into account properly. In the work of Stachniss et al., up to 100 particles are needed with up to 6% multimodal proposal distributions over the whole trajectory. However, these potentially multimodal proposal distributions are generated using a unimodal odometry model as a reference and subsequent hill-climbing for scan matching. Thus, the results gathered by Stachniss et al. show that multimodality due to locally ambiguous structure is even a problem worth considering when taking odometry into account.

The number of particles used in the filter is usually seen in the context of finding correspondences to previously visited locations, also referred to as loop closing: 'At least one particle of the set must already close that loop by chance, and either many particles are needed or there will be gaps in a loop already closed' (Frese, 2006). Please note that the number of particles needed to represent a multimodal distribution of pose transformations is a different issue. Each mode represents a possible choice which way the robot may have moved. Having to make many of these choices in a row, and keeping track of all possible trajectories, leads to a combinatorial explosion. Experimental results presented in Section 3 indicate that 10,000 particles are not enough to handle this situation consistently, even when there are just a few bimodal or trimodal registrations.

One may ask whether the handling of multimodal registrations is an issue at all. If graph-based SLAM cannot represent it by design and particle filter SLAM is supposedly not suited for it in practice, then the obvious question is why are there so many reports of successful applications of the two families of approaches in the literature. The answer is simple. Spatial relations from registration results are usually combined with motion estimates from odometry. This kind of sensor fusion helps to compensate for possible ambiguities in one sensor modality with a unique, but quite uncertain, estimate in another one. Consider for example a featureless hallway with a repetitive pattern as described before, or a symmetric star crossing as motivated in Figure 1. Odometry can approximately predict the correct part of the pattern or help to disambiguate the symmetries in the appearance of the crossing. It can also play a significant role by weighting candidate transformations even if it is very noisy. However, robots are increasingly used in applications where good motion estimates are inherently difficult or require costly equipment. Examples include aerial and underwater vehicles, as well as land robots operating on difficult terrain. However, Stachniss et al. (2007) also showed that even when good unimodal odometry is available and the robot operates in an indoor setting, multimodal motion estimates still occur and should be explicitly dealt with. In addition, SLAM has been predominantly investigated so far in static environments. However, dynamic changes in the scene, which cannot be avoided for many real-life applications, inherently lead to structural ambiguities. An example here would be moving cars and rearranged furniture or other equipment. Last but not least, multimodal effects may be relatively rare but even a single bimodal registration where the wrong choice is made can cause a severe distortion of

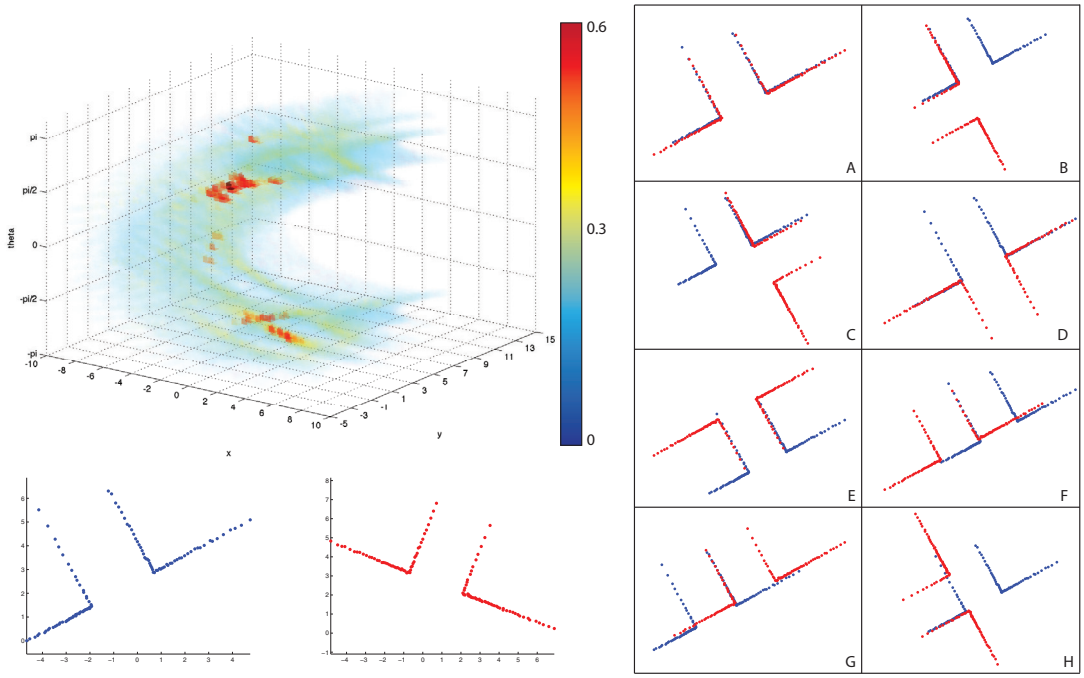

Fig. 1. Structural ambiguity in the environment can be one source of registration errors, here for example in form of the crossing of two corridors. Two scans (bottom left) are to be registered; one scan contains the corridor to the North while the other scan includes the East corridor after the robot turned to the right. Using a simple correlation-based method to match the two scans, multiple maxima are immediately visible in the resulting parameter space (top left). Some registration results corresponding to the local maxima are shown on the right, ordered by quality. However, in this case, *A* is not the globally correct result, but *C* is.

the map in form of an effect known as brokenness (Birk, 2010).

There are three main contributions presented in this article:

- First, the problem of ambiguous registration results leading to multimodal distributions of local spatial relations is addressed. This problem cannot be tackled with standard graph-based SLAM algorithms by design as they assume unimodal distributions. Furthermore, it is shown that particle filter SLAM, which in theory can handle this problem, is at least severely challenged by it, especially when a good odometry estimate is not available to mitigate the effects of ambiguities.
- Second, an extension of the pose graph data structure is introduced in form of multimodal edge constraints modeled by mixtures of Gaussians (MoG), which is complemented by suited optimization methods. These methods are (a) robust iteratively reweighted least squares and (b) Prefilter Stochastic Gradient Descent (SGD) where a preprocessing step is used to determine globally consistent modes before applying SGD (Olson et al., 2006). In addition, a variant of method b) is introduced in form of (c) Prefilter Levenberg–Marquardt (LM).
- Third, the benefits of MoG pose graph optimization with the novel prefilter methods are further substantiated with real-world experiments. It is demonstrated how the MoG pose graph optimization can be used with a concrete registration method and that it leads to superior results. Plane-based registration is used here as an example for 3D mapping with two different data sets.

The rest of this article is structured as follows. The remainder of this section will introduce maximum likelihood SLAM with pose graphs as well as previous work in this area, and present the extension to multimodal pose graphs. In Section 2, several methods to optimize multimodal pose graphs are described. Experimental results on several synthetic graphs are presented in Section 3. Section 4 focuses on experiments with real-world data, including a method to generate multimodal results from a plane-based registration algorithm. Finally, Section 5 concludes the article.

1.1. Maximum likelihood SLAM

One very popular formulation of the SLAM problem is probabilistic. Given a sequence of sensor readings $z_{1:t}$ and control inputs (such as motor voltage) $u_{1:t}$ until time t, compute the most probable map m and trajectory $x_{1:t}$:

$$x^*_{1:t}, m^* = \underset{x_{1:t},m}{\operatorname{argmax}} \left[p(x_{1:t}, m | z_{1:t}, u_{1:t}) \right]. \quad (1)$$

After Thrun et al. (2005), this probability is factored as

$$p(x_{1:t}, m | z_{1:t}, u_{1:t}) = p(x_{1:t} | z_{1:t}, u_{1:t}) p(m | x_{1:t}, z_{1:t}). \quad (2)$$

It is thus possible to estimate the trajectory separately from the map.

Graph-based maximum likelihood SLAM methods reduce the above full SLAM problem to a localization problem in order to compute good estimates for the trajectory $x_{1:t}$;

$$x^*_{1:t} = \underset{x_{1:t}}{\operatorname{argmax}} \left[p(x_{1:t}|z_{1:t}, u_{1:t}) \right]. \tag{3}$$

Deriving the map m^* from the optimal trajectory $x^*_{1:t}$ is trivial.

Formally, a pose graph is an undirected graph $G = (V, E)$ consisting of vertices V and edges E. The vertices $v_i \in V$ denote poses where the robot obtained sensor observations z_i. A pose estimate x_i is also associated with the vertex and thus is a tuple $v_i = (x_i, z_i)$. In addition to the vertices it connects, each edge $e_k \in E$ contains a constraint c_k on the pose estimates of the associated vertices, thus $e_k = (v_i, v_j, c_k)$. While the graph itself is undirected, the edge has to declare a sort of observation direction, the direction in which the constraint was generated. In case the edge is traversed in reverse direction, the constraint c must be inverted. What exactly that entails is up to the representation of the constraint.

We consider the general case, where one such constraint between vertices v_i and v_j is of the form

$$p(x_j \ominus x_i | z_i, z_j, u_{i:j}) \tag{4}$$

$$\stackrel{\text{def.}}{=} p(t_i^j | z_i, z_j, u_{i:j}) \tag{5}$$

where $\ominus$ is the pose difference operator (Smith et al., 1990), which produces the relative transformation t_i^j from the coordinate frame at v_i (namely x_i) to the frame at v_j (x_j), and $u_{i:j}$ is the sequence of control inputs between the two. The constraint is generated, for example, by odometry or a sensor data registration algorithm using z_i and z_j. In general, the constraint c stored in the graph will have parameters that depend on z_i, z_j, and $u_{i:j}$. Thus, we will write

$$p(t_i^j | z_i, z_j, u_{i:j}) = p(t_i^j | c_k) \quad \text{for constraint } k. \tag{6}$$

It is now possible to express the probability of the robot trajectory as a function of the constraints:

$$p(x_{1:t}|z_{1:t}, u_{1:t}) = \prod_i \prod_j p(x_j \ominus x_i | z_i, z_j, u_{i:j}) \tag{7}$$

$$p(x_{1:t}|G) = \prod_{(v_i, v_j, c_k) \in E} p(x_j \ominus x_i | c_k). \tag{8}$$

As not all observations are directly related to each other, the specific graph of constraints is used to formulate the probability in a more intuitive way.

Previously, the constraint probability density (Equation (6)) was exclusively modeled with a single multivariate normal distribution. This allowed to formulate very efficient optimization algorithms to solve Equation (3).

1.2. Previous work in pose graph optimization

In the past, the constraint probability density functions have been modeled as multivariate normal distributions as follows:

$$p(t_i^j | c_k) = \frac{1}{|2\pi \Sigma_k|^{1/2}} e^{-\frac{1}{2}(t_i^j \ominus \mu_k)^{\mathrm{T}} \Sigma_k^{-1} (t_i^j \ominus \mu_k)}. \tag{9}$$

Here, μ_k is the mean of the transformation estimate for the constraint c_k, and Σ_k is the corresponding covariance matrix. Note that $|2\pi \Sigma_k| = (2\pi)^d |\Sigma_k|$ if $\Sigma_k \in \mathbb{R}^{d \times d}$ (Petersen and Pedersen, 2008).

In the context of optimization, the logarithm of the full trajectory probability (Equation (8)) is most useful due to its numerical stability and ease of computation:

$$\ln p(t_i^j | c_k) = -\frac{1}{2} \ln (|2\pi \Sigma_k|) - \frac{1}{2} (t_i^j \ominus \mu_k)^{\mathrm{T}} \Sigma_k^{-1} (t_i^j \ominus \mu_k) \tag{10}$$

$$\ln p(x_{1:t}|G) = -\frac{1}{2} \sum_{(v_i, v_j, c_k) \in E} \ln (|2\pi \Sigma_k|) - \frac{1}{2} \sum_{(v_i, v_j, c_k) \in E} (t_i^j \ominus \mu_k)^{\mathrm{T}} \Sigma_k^{-1} (t_i^j \ominus \mu_k). \tag{11}$$

By neglecting the constant terms, this is turned into a cost function equivalent to the sum of squared Mahalanobis distances over all constraints.

$$cost(x_{1:t}|G) = \sum_{(v_i, v_j, c_k) \in E} (t_i^j \ominus \mu_k)^{\mathrm{T}} \Sigma_k^{-1} (t_i^j \ominus \mu_k). \tag{12}$$

This cost function was first described by Lu and Milios (Lu and Milios, 1997).

Up to now, almost all pose graph optimization methods have used the presented approach. Only the specific method to find the global minimum of Equation (12) varies between the known methods.

In their early seminal work, Lu and Milios (1997) solved this optimization problem by iteratively linearizing Equation (12) around the current trajectory estimate, stacking all constraint equations in a constraint matrix, and subsequently solving the resulting linear system. This procedure results in incremental improvements to the complete trajectory at each iteration and is closely related to Newton's method. However, the larger the pose graph, the larger the constraint matrix that needs to be inverted. Later work, such as that of Borrmann et al. (2008), extends this approach to 3D and sparse constraint matrices, which speeds up the method.

Golfarelli et al. (2001) introduced a kinetic perspective on pose graph optimization by relating it to a truss structure with springs. More severe linearization artifacts as in Lu and Milios's work occur in the conversion to a truss structure. In fact, rotational springs are represented as linear

springs attached to a lever in order to be able to represent them in classical structural analysis terminology. As a result, the employed method does not work well with nonlinearities introduced by rotational degrees of freedom, yet still requires solving a large linear system.

Frese et al. (2005) employed a multi-scale representation of the graph, a practice borrowed from the partial differential equation literature. Here, a variant of Gauss–Seidel relaxation is applied to the graph at different levels of subsampling for only a few iterations. Results are projected up and down between levels to achieve fast global convergence. Owing to its iterative nature, the method relinearizes the cost function at each step which corrects the pose of one vertex at a time. The method thus properly takes nonlinearities into account.

Dellaert's $\sqrt{\text{SAM}}$ (Dellaert, 2005) takes a Bayesian network perspective on the SLAM problem. The robot's trajectory is modeled as a Markov chain, while the landmarks and the measurements relating them to the robot poses correlate multiple poses in the trajectory. Since the robot has a limited field of view, the resulting linear system is sparse and can be efficiently solved by specialized reordering and sparse Cholesky methods. Dellaert also introduced the use of general least-squares methods for SLAM.

Olson et al. (2006) published a very efficient and popular method called stochastic gradient descent. The main contribution lies in the reinterpreted state space of the pose graph. Instead of using a set of global poses, as in the previously described methods, Olson uses an incremental state, where pose x_t is the sum of all preceding pose increments $\sum_1^t \delta x_i$ on a trajectory. In this state representation, following the gradient of a single constraint has corrective side effects on all poses $x_{i:j}$ in between the two poses x_i and x_j connected by the constraint. Thus, the method is able to rapidly converge to a good global estimate. This method has been extended to a tree representation of the incremental state space (Grisetti et al., 2007c) and to 3D pose graphs (Grisetti et al., 2007a). An online variant also exists (Olson et al., 2007; Grisetti et al., 2008).

More classical nonlinear least-squares methods, such as Gauss–Newton and LM, were used by Dellaert (2005), Hertzberg (2008), Grisetti et al. (2010), Konolige et al. (2010), and Kümmerle et al. (2011). Grisetti et al. (2010) reduce the iteration complexity by optimizing the graph at different scales, much like the approach by Frese et al. (2005). Hertzberg's, Grisetti et al.'s, and Kümmerle et al.'s methods (Hertzberg, 2008; Grisetti et al., 2010; Kümmerle et al., 2011) implicitly linearize the trajectory estimate by replacing the pose difference and compound operators by linearized variants. This allows to update poses in the trajectory (or landmark locations) easily and consistently. Kümmerle et al. (2011) as well as Hertzberg (2008) aim at a general framework for nonlinear least-squares optimization applied to pose graphs that is easily extensible to different vertex types (e.g. full pose, landmark location, etc.), yet computationally efficient. Konolige et al. (2010) rather follow a classical approach and mainly focus on assembling the sparse linear subproblem in the LM method efficiently.

Sünderhauf and Protzel (2012) add an interesting extension to the work of both Dellaert (2005) and Kümmerle et al. (2011), namely switch variables that are part of the optimization and can switch off constraints. The marked advantage of such an approach is that false loop closures or misled registration results, i.e. spurious edges in the graph, can be removed during the optimization.

1.3. The MoG pose graph

Particle and linear Gaussian distributions are at opposite ends of the density complexity spectrum. Particle distributions are very general, but usually hard to compute and memory intensive. Linear Gaussians are easy to compute and use, but are limited in their descriptive power. A distribution family which is both more descriptive than single linear Gaussians and less computationally complex than particle distributions is hence an interesting option for modeling the density in Equation (6).

In this article, we explore the possibility of using a MoG in Equation (6). This has several advantages. Most importantly, a mixture can represent multimodal densities which may arise due to ambiguities in the sensor data registration. In addition, a mixture can approximate non-Gaussian and nonlinear distributions with a single mode as well, e.g. generated by odometry error models (Thrun et al., 2005).

The usual reason quoted for using single Gaussians in SLAM and state estimation in general is the central limit theorem (Feller, 1945). It states that if arbitrarily distributed random variables are added many times, e.g. through pose composition, the distribution of the resulting random variable is Gaussian in the limit. However, since one constraint is not directly combined with other constraints (it may be for the purpose of the final estimation and optimization algorithm, but not conceptually) this theorem does not apply. Thus, too much important information is lost when using overly simplified Gaussian distributions already in the constraints.

From a practical perspective as motivated above it is also of interest to be able to represent multimodal registration results. Consider again the example shown in Figure 1 with the two scans containing two different corridors at a crossing. The global optimum of the registration by cross-correlation is A, while the correct result would be C. As shown experimentally in Section 3, standard graph-based methods are extremely sensitive to wrong motion estimates which give rise to inconsistent constraints in the graph. This occurs when only the most prominent registration result is taken into account (A in the example), even if such situations only occur very rarely in the complete mapping process. The remedy is to represent all local optima of the registration that are also likely candidates of being the correct solution, thus covering all feasible motions and

allowing the optimization method to infer the correct one. Note that according local optima can be easily identified in not only cross-correlation-based scan-matching but also in various other registration methods. It is for example straightforward to extend existing spectral (Pfingsthorn et al., 2010) or random sampling consensus (RANSAC)-based (Fischler and Bolles, 1981) registration methods to produce a MoG as the registration result. For example, in the case of the iFMI spectral registration method (Pfingsthorn et al., 2010), the result is a sort of histogram of potential registration results. It is only a matter of fitting a MoG to the histogram, by using e.g. expectation maximization. In RANSAC-based methods, the only change needed is to keep track of the N most congruent samples instead of just the best one. In iterative methods such as iterative closest point (ICP) (Besl and McKay, 1992), local optima can for example be identified by perturbing the starting conditions, as in (Stachniss et al., 2007). A specific example is given in Section 4.1, which describes how MoG registration results are generated with a plane-based 3D registration method (Pathak et al., 2010c) used in the experiments with real-world data.

It is important to note that there is a significant difference between global and local data association, and hence in the ambiguity inherently present in each of these problems. We define global data association to be what is more commonly called loop detection or place recognition. Here, the main problem is to find observations made some time ago that match the current one. This should generally lead to the generation of a new edge between the current vertex and another, much older, vertex generated during a previous visit to the current location.

On the other hand, we define local data association to be the search for true correspondences in two consecutive observations. In this case, it is very likely that the two observations do match, but ambiguities may occur due to the environment structure, occlusions, or dynamic changes in the environment that are not decidable given only these two observations. We define this kind of ambiguity as local ambiguity. This case is very efficiently modeled by using mixtures in edge constraints as proposed in this article, and it is possible to resolve these only by considering the map as a whole.

Note that global data association is orthogonal to local data association. It is possible to detect loops without knowing the exact and unique transformation between the stored and current observations. More precisely, global data association involves a probability mass function over all loop hypotheses, i.e. the existence of feasible edges, explicitly including the null hypothesis of the non-existence of a loop at the current vertex. For example, Sünderhauf and Protzel (2012) specifically allow for a single loop-closure hypothesis, i.e. the existence of an edge, and the null hypothesis, i.e. the non-existence. Local data association involves a probability mass function (the component weights) over all possible transformation hypotheses between two observations. Both have to be estimated, and both may coexist at the same time.

Consistency checks such as single-cluster spectral graph partitioning (SCGP) by Olson (2009) can be used to solve the global data association problem. However, Olson explicitly rejects locally ambiguous results in the loop detection step. If a good multimodal local data association method was used, this would not be necessary. In previous work, Olson also demonstrated how to register two laser scans with SCGP (Olson et al., 2005), which may be applicable to the local data association problem. However, as the potential ambiguity is an inherent part of the observation itself, no algorithm will be able to solve a truly ambiguous registration using only local information. In addition, if SCGP was to be used to resolve local ambiguities, i.e. choosing one of the local motion hypotheses for all edges, it would require a polynomial-time pairwise compatibility metric which does not exist in the case of multimodal pose graph, at least not in the form described in Olson (2009). As there potentially exist exponentially many paths between two vertices depending on which combination of components are chosen on the way, computing the consistency of a single loop already has exponential cost.

Thus, a global method is indeed needed to solve the local data association problem if ambiguities occur. However, this method should not only deal with mutually exclusive choices to solve local ambiguities, but at the same time also allow more versatile and nonlinear probability densities. The most natural and elegant solution is thus to combine the global optimization step with the global solution to the local data association problem.

The change in parameterization of the density in Equation (6) has far-reaching consequences for the optimization of the resulting MoG pose graph. A general mixture model is formulated as follows (Titterington et al., 1985):

$$p(x) = \sum_{m=1}^{M} p(m)\, p(x|m) = \sum_{m=1}^{M} \pi_m p(x|m) \qquad (13)$$

where $\sum_{m=1}^{M} \pi_m = 1$ and $\pi_m \in (0, 1]$. We make the explicit extension that a single component weight π_m is 1 if and only if the mixture contains only one component. While technically not being a mixture, this allows constraints to include the classical case of single Gaussian distributions as described above. In our case, each component density $p(x|m)$ is a multivariate Gaussian with mean μ_m and covariance Σ_m as defined in Equation (9). Therefore, each constraint $c = (\{\pi_m\}, \{\mu_m\}, \{\Sigma_m\}, M_k)$. Also, we denote the number of components on constraint k as M_k. Thus, the joint probability of a MoG pose graph is

$$p(x_{1:t}|G) = \prod_{(v_i, v_j, c_k) \in E} \sum_{m=1}^{M_k} \pi_m p(t_i^j | \mu_m, \Sigma_m). \qquad (14)$$

Unfortunately, the log probability is not as easily computed as in the unimodal case due to the sum of the mixture:

$$\ln p(x_{1:t}|G) = \sum_{(v_i,v_j,c_k)\in E} \ln \left[\sum_{m=1}^{M_k} \pi_m p(t_i^j|\mu_m, \Sigma_m) \right]. \quad (15)$$

However, since a MoG constraint is rather the exception in a MoG pose graph, this form is still desirable since it is compatible with the more numerous single Gaussian for which Equation (10) still holds if $M_k = 1$.

$$\ln p(x_{1:t}|G) = \sum_{(v_i,v_j,c_k)\in E} \cdot \begin{cases} \ln \left[\sum_{m=1}^{M_k} \pi_m p(t_i^j|\mu_m, \Sigma_m) \right], & \text{if } M_k \neq 1 \text{ or} \\ -\frac{1}{2} \ln(|2\pi\Sigma_1|) - \frac{1}{2}(t_i^j \ominus \mu_1)^{\mathrm{T}} \Sigma_1^{-1} (t_i^j \ominus \mu_1). \end{cases} \quad (16)$$

Here, it is not possible to neglect the constant terms of the Gaussian distribution since the relative scaling of the component covariances in the mixture matters.

Owing to the significant reformulation of constraints, previous optimization approaches are only partially applicable with a less restrictive distribution family and local ambiguity. Many assumptions are violated when moving from a single Gaussian to a mixture.

2. MoG pose graph optimization

2.1. Particle filter

As particle filters for SLAM (Thrun et al., 2005) can deal with arbitrary distributions, they can also be considered as an option for optimizing MoG pose graphs. Classical particle filters estimate the trajectory of the robot over time. At each step, hypotheses about the new robot pose are sampled from a proposal distribution, such as an odometry model or registration result. During the resampling phase, only good particles, i.e. ones with a high probability, are drawn into the new particle population for the next step.

A single particle iteration thus depends on two steps: A way to draw samples from a proposal distribution, and a way to evaluate the importance of a single particle. Both is relatively straightforward when dealing with a pose graph containing MoG.

Sampling from a constraint probability density which is a MoG consists of two steps. First, a component is chosen according to the mixture probabilities π_m (Equation (13)). Then, a sample is drawn from the selected component distribution.

Evaluating a particle's importance is equivalent to evaluating the pose graph joint probability as described in Equation (8). This gives rise to the particle-based optimization algorithm (Algorithm 1), which is very similar to that described by Stachniss et al. (2007).

Instead of following the robot trajectory, which might be suboptimal for estimation, our particle-based optimization approach samples all edges that form a spanning tree from the first vertex in the graph v_1. This is easily implemented using a breadth-first traversal, neglecting edges that connect to already visited vertices. Such a traversal guarantees the shortest paths to all vertices, which in turn means that the least amount of sampled transformations have to be compounded to estimate global vertex poses. Therefore, a breadth-first traversal minimizes the number of particles needed for consistent sampling.

Algorithm 1: Particle-based optimization of a MoG pose graph.

Input: MoG PoseGraph G, maximum number of particles N, minimum effective number of Particles N_{min} (usually $\frac{N}{2}$)
Output: optimized vertex poses $x_{1:t}$
initialize particle set $x_1^i = 0,\ i = 1 \ldots N$;
initialize particle weights $w^i = \frac{1}{N},\ i = 1 \ldots N$;
for *all breadth-first edges from* v_1, $(v_a, v_b, c) \in E$ **do**
  if *pose b already set* **then**
    continue;
  end
  $y = x$;
  for $i = 1 \ldots N$ **do**
    sample $t \sim p(x_k|c_k)$;
    $y_b^{[i]} = x_a^{[i]} \oplus t$;
    $w^{[i]} = \prod_{(v_a,v_b,c)\in E:isset(x_a)\vee isset(x_b)} p(y_b^{[i]} \ominus y_a^{[i]}|c)$;
  end
  normalize particle weights w;
  if $\frac{1}{\sum_i^N (w^{[i]})^2} > N_{min}$ **then**
    continue;
  end
  for $i = 1 \ldots N$ **do**
    draw $n \propto w^{[n]}$ using the low variance sampler from Thrun et al. (2005);
    $x^{[i]} = y^{[n]}$;
  end
end
$i^* = \operatorname{argmax}_i w^{[i]}$;
$x_{1:t} = x_{1:t}^{[i^*]}$;

For each spanning tree edge, the underlying constraint density is sampled N times, once per particle. The pose estimate for the newly discovered vertex is formed by compounding the sampled transformation with the particle's estimate of the current vertex pose. Then, the particle probability is given by evaluating all edge constraints which connect vertices where the pose estimate is already set. Only if the effective number of particles, computed as

$$N_{eff} = \frac{1}{\sum_i^N (w^{[i]})^2} \quad (17)$$

is less than the desired effective number of particles N_{min} (usually half the number of particles N), the resampling

step draws N particles from the temporary set of particles, with replacement. Here the low variance sampling algorithm from Thrun et al. (2005) is used. When all spanning tree edges have been processed, the best particle is chosen as the optimization result.

The main drawback of this method is particle exhaustion. A significant amount of particles is needed to consistently sample mixture densities with more than one component, many more than typically used in particle-filter-based SLAM methods. This is especially the case with deep graphs, i.e. where the spanning tree is deep, or when MoG constraints are encountered early in the breadth-first traversal.

The second drawback is that all pose estimates are discretized at the time of sampling the constraint distribution. Thus, the best particle after running the optimization only approaches the global maximum of the joint probability as $N \rightarrow \infty$. Even if sufficiently many particles are used to consistently sample the underlying pose graph, the best sample only approximates the global maximum.

2.2. Reduction methods

It is possible to divide the multimodal pose graph optimization problem into two separate subproblems. First, find a unimodal replacement density for all edge mixtures. Then use a known optimization method, e.g. any of those described in Section 1.2, to optimize the resulting unimodal pose graph in a second step. This is denoted as the reduction approach, which is interesting as it allows to use well-established methods from the literature in the second step. However, the first subproblem is not trivial.

The reduction approach is related to least trimmed squares (LTS), a robust least-squares formulation. The general idea behind LTS (Rousseeuw and Leroy, 2005) is to make a binary decision which terms of the quadratic function to keep and which to drop as outliers. If the binary decision can identify outliers reliably, then this method works very well. Unfortunately, there is no universal way of computing these decisions. Therefore, a few different approaches to solve this problem are presented below.

2.2.1. Basic methods: Exhaustive, Max, and Multi-Edge The most obvious and naïve solution is to exhaustively try all component combinations to generate a number of unimodal pose graphs, or candidate decisions on which superfluous modes to drop in the first step. Each candidate is optimized and if the final negative log probability as evaluated in the original MoG pose graph is less than the previously recorded minimum, this result is kept. Such an exhaustive search is hence optimal in the sense that it finds the best possible solution that can be achieved with the given optimization method used in the second step. In addition, it converges to exactly the same result as if the MoG pose graph contained no ambiguity to start with, and thus represents a canonical baseline for comparison. However, it also takes a very long time to complete as it involves a combinatorial explosion. In the experimental results below, this method for the first step is referred to as Exhaustive.

Another possibility for the first step of finding a unimodal replacement is to only use the component of each mixture with the largest weight. This corresponds to neglecting the multimodality altogether and mirrors what happens in the classical case where only unimodal densities are considered, i.e. in state-of-the-art graph SLAM methods. In the experiments below, this filtering method is referred to as Max.

Furthermore, given that the underlying optimization method can process it, it is also possible to treat each component as a separate edge. Note that this results in a graph with more than one constraint between two vertices, also known as a multigraph. This can be considered as a case where no filtering is done at all. However, it does not change the cost function as described in Equation (12) in itself; it only adds more terms. A possible underlying method using a robust cost function, an approximation to the one from Equation (12), may still converge to a good solution. Since all components are regarded as separate edges, and thus form a multi-edge, the method is referred to as Multi-Edge.

2.2.2. Prefilter The most optimal reduction method would compute the choice of components that the exhaustive method uses without having to try all others. The prefilter method described in this section aims to approximate this choice as quickly as possible by employing ideas from the particle method. Specifically, the method assigns global vertex poses and uses a simple heuristic to select globally consistent components for further optimization.

The important insight here is that it is not necessary to traverse the complete graph in order to assign global poses to each vertex. Only a spanning tree has to be traversed, as in the particle filter implementation described above. Thus, to reduce the number of possible global poses for each vertex during this traversal, the minimum spanning tree with respect to the number of components on each edge constraint is used. Specifically, the edge weight $w(e)$ is not uniform (as in Grisetti et al., 2007c), but is equal to the number of components, so $w(e_k) = M_k$. This in effect chooses the least ambiguous spanning tree in order to track the least amount of combinations needed to assign global poses. Edges with high ambiguity are skipped if possible and non-ambiguous edges are preferred. In addition, the specific edge distributions are not randomly sampled as in the particle filter method, only the component means are used. This significantly speeds up the method with respect to a full particle filter as described in Section 2.1.

Algorithm 2 shows how to incrementally follow minimum spanning tree edges using Prim's algorithm (Cormen et al., 2001) and to assign multiple global poses to each vertex. In order to trade off time and memory requirements versus accuracy, only the most probable N assignment sets

Algorithm 2: An algorithm to find a good approximation to the globally correct choices of components.

Input: MoG PoseGraph G
Input: maximum number of hypotheses N
Output: $\mathbf{X}$: a set of N sets of vertex poses $X = \{x_i\}$
$\mathbf{X} = \{\{x_1\}\}$;
$V_{used} = \{v_1\}$;
$E_{used} = \emptyset$;
initialize priority queue P to sort by number of components;
for *all adjacent edges e of* v_1 **do**
  enqueue(P, e);
  $E_{used} = E_{used} \bigcup e$;
end
while P *not empty* **do**
  e = dequeue(P);
  (v_{start}, v_{end}) = vertices(e);
  for *all adjacent edges* $e_{adj} \notin E_{used}$ *to* v_{start} *or* v_{end} **do**
    enqueue(P,e_{adj});
    $E_{used} = E_{used} \bigcup e_{adj}$;
  end
  if $v_{start} \notin V_{used}$ **then**
    $v_{new} = v_{start}$;
    $v_{old} = v_{end}$;
    let *means* contain the inverse of all means (spatial transformations) of the mixture in e;
  else
    $v_{new} = v_{end}$;
    $v_{old} = v_{start}$;
    let *means* contain all means (spatial transformations) of the mixture in e;
  end
  for *each hypothesis* $X \in \mathbf{X}$ **do**
    x_{old} = pose of v_{old} in X;
    $\mathbf{X} = \mathbf{X} \setminus X$;
    for *each mean in means* **do**
      $\mathbf{X} = \mathbf{X} \bigcup \{X \bigcup (x_{old} \oplus mean)\}$;
    end
  end
  if $|\mathbf{X}| > N$ **then**
    sort $\mathbf{X}$ by joint probability of assigned vertex poses per hypothesis;
    truncate $\mathbf{X}$ to contain only the N most probable elements;
  end
end

are kept after each iteration. The algorithm is deterministic, it produces the same sets of global pose assignments given the same parameters. This is especially useful as it allows for a fast and consistent filtering of congruent pose estimates in the MoG pose graph.

Only the global pose assignment set that is most probable, i.e. the best ranked set after algorithm (Algorithm 2), is used to select the component of each edge mixture for further optimization. For each edge, the pose difference $t_i^j = x_j \ominus x_i$ between the two connected vertices v_i and v_j is computed based on their assigned global poses from this set. Then the component which assigns the highest weighted probability, its net contribution to the full mixture probability, to the pose difference is chosen. Concretely, the selected component is

$$m^* = \underset{m}{\operatorname{argmax}} \left[\pi_m p(t_i^j | m) \right] \tag{18}$$

or, equivalently,

$$m^* = \underset{m}{\operatorname{argmax}} \left[2 \ln \left(\frac{\pi_m}{|2\pi \Sigma_m|} \right) - (t_i^j \ominus \mu_m)^{\mathrm{T}} \Sigma_m^{-1} (t_i^j \ominus \mu_m) \right]. \tag{19}$$

This method is denoted as Prefilter.

Also other approaches were considered and tested, including:

- random selection of components during each iteration;
- greedy selection of most probable components during each iteration;
- greedy selection of closest components based on residual during each iteration;
- random selection of components, subsequent optimization, and finally reporting the best result (RANSAC like).

However, each of these more heuristic methods did not perform well, especially in comparison with the above Prefilter method, and they are thus not included in the discussion here for space constraints.

2.2.3. Optimizing the resulting unimodal pose graph For the second step of optimizing the resulting unimodal pose graph, two state-of-the-art methods are used.

First of all, an improved version of Olson's SGD (Olson et al., 2006) as implemented in the popular TORO library (Grisetti et al., 2007c) is used. An important detail of this implementation, and SGD in general, is that it uses a relative parameterization of the vertex poses, such that any vertex pose is computed as the sum of relative pose increments from the start of the trajectory to itself. In particular, TORO uses a spanning tree for this relative parameterization in order to make better use of the underlying graph structure. Only the pose increments of each vertex relative to its parent in the spanning tree are changed during optimization. This allows good global convergence because the actual global vertex poses change significantly when small increments are applied to the relative poses, and SGD is thus less likely to find local minima. However, the optimization may also oscillate significantly, which is managed by a step size schedule which reduces the step length every iteration. In addition, due to speed considerations, a diagonal approximation of the Hessian is used in SGD, which may reduce convergence speed and accuracy.

Furthermore, a sparse LM method as implemented in the g^2o library (Kümmerle et al., 2011) is employed. LM is a very well known least-squares method, and is guaranteed to converge, but only to the nearest optimum. The method can not jump out of one basin of convergence to another, as SGD can, which makes it even more important to find a good initial guess. In contrast to SGD, LM uses the full hessian information, and thus also off-diagonal elements of the constraint covariances, which is expected to make it converge faster. The g^2o library implements a robust least squares formulation, namely iteratively reweighted least squares (IRLS), and is thus the only method usable for the Multi-Edge approach. In this method, each constraint is assigned a weight in each iteration dependent on the current residual. The g^2o library utilizes the Huber cost function (Huber, 1973), which is quadratic close to zero, and linear with larger residuals, but still smooth and convex. In theory, this should allow the LM method to behave well even in the presence of outliers such as inconsistent constraints in the graph. The robust LM formulation is denoted with LM in all methods below that use it as second step, except in the exhaustive case where no outliers due to the incorrect selection of modes are expected.

In the following, all reduction methods are denoted with the according prefix for the first step and the related postfix for the second step. For example, Exhaustive SGD refers to exhaustive filtering as a first step for finding a unimodal replacement in combination with SGD as subsequent unimodal optimization in the second step. Accordingly, Exhaustive LM is the combination of exhaustive filtering with LM optimization.

Thus, the complete list of reduction methods includes

- Exhaustive SGD and Exhaustive LM;
- Max SGD and Max LM;
- Multi-Edge LM;
- as well as Prefilter SGD and Prefilter LM.

3. Experiments with synthetic data

The discussion of experimental results is split into two distinct parts. This section focuses on a systematic evaluation and characterization of the proposed optimization methods using synthetic MoG pose graphs to motivate and analyze the generic aspects involved in multimodal graph SLAM. The synthetic data sets have the important advantage that the amount of multimodality in the constraints can be easily controlled and varied. In particular, it will be shown that standard SLAM methods, in particular traditional graph-based and particle filter SLAM methods, are challenged by ambiguous registration results. Furthermore, MoG pose graphs that model these ambiguities in constraints with multimodal distributions together with Prefilter SGD/LM optimization are a very promising alternative for this case. Ground truth data is used to quantify the quality of the final map using the state squared error (*SSE*) metric by Olson (2008), see also Appendix B.

Section 4 then describes the application of MoG pose graphs in the context of a concrete registration method to generate 3D maps. There, real-world data sets are used to further substantiate that MoG pose graphs with Prefilter SGD/LM optimization produces significantly better results than standard graph-based and particle filter SLAM methods.

3.1. Results of systematic evaluation with varying multimodality

Synthetic pose graphs were generated to test the presented methods in depth. A total of 110 different random pose graphs are used, broken up into 11 separate cases with varying complexity with respect to the amount of multimodality in the constraints. In all cases, multimodal constraints are assumed to be an exception, i.e. most edges contain a standard Gaussian and only a few contain a mixture. Also, the number of modes in case of mixtures is quite small. As will be shown, even these relatively small amounts of multimodality have significant effects. All 110 graphs are generated from the same environment as shown in Figure 2. The exact algorithm to generate the graphs is included in Appendix A.

The degree of multimodality plays a significant role in the shape of the log probability function. As this effect is mainly due to the combinatorial properties of pose composition in the graph with multiple components, we can define the following complexity metric:

$$C(G) = \log_2\left(\prod_{(v_i,v_j,c_k)\in E} M_k\right) = \sum_{(v_i,v_j,c_k)\in E} \log_2(M_k). \quad (20)$$

For example, a graph with a single two-component edge and all other edges being unimodal would have a multimodal complexity of $C(G) = 1$. In condition 8, a graph with five three-component edges has a complexity of $C(G) = \log_2(3^5) = 7.92$. Similarly, a graph without any MoG constraints would have a $C(G) = 0$. The different cases considered for the experiments in this article are shown in Table 1.

Each method described in Section 2 was used to optimize each generated pose graph. As traditional optimization methods used for comparison with the standard unimodal case, the Max SGD and Max LM methods are assumed not to know about multimodal constraints, and thus are initialized with a breadth-first assignment of poses using the maximum component for each edge. Prefilter SGD and Prefilter LM, as well as Multi-Edge LM, are initialized with the most probable set of global vertex poses produced with the prefilter step described in Section 2.2.2. Here, $N = 200$ was chosen for the good trade-off between computation speed ($\approx$ 2 ms) and accuracy. Only the best result of these was used as the initial condition for optimization and to select modes as described above. The Particle method did not need initialization, and since it is not deterministic it was run 10

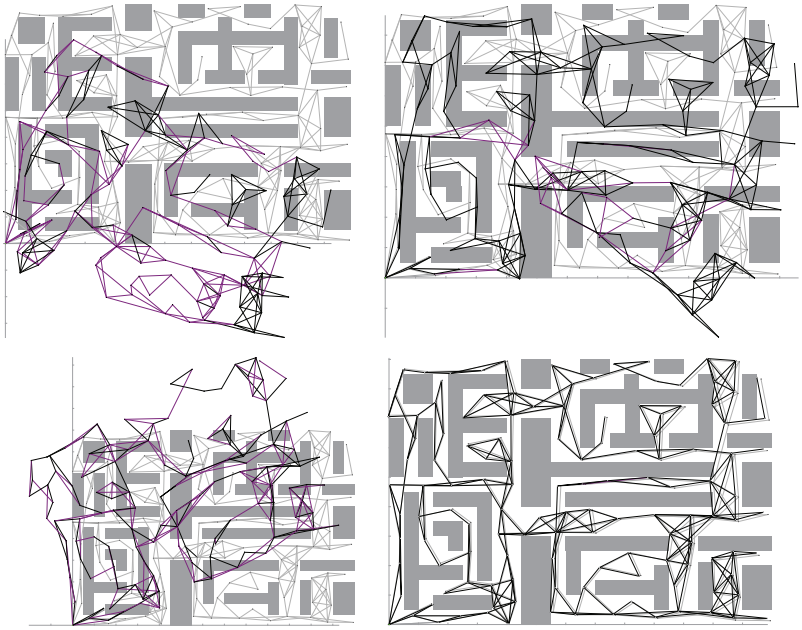

Fig. 2. An example multimodal pose graph of complexity $C(G) = 4$, meaning it contains four two-component mixtures, all other edge distributions are unimodal Gaussians. Top: Result of the Max SGD and Max LM methods, respectively. Edges that are assigned a low probability transformation are shown in dark gray/magenta. The ground truth is shown in light gray in the background. Note that this method is the standard up to now. Bottom left: Result of the Particle method with 10,000 particles. Bottom right: Optimization result of the Prefilter SGD method.

Table 1. The 11 conditions used in the experiments with their different amounts of multimodal edges and their degree of multimodality $C(G)$. The overall percentage of multimodal edges (MM%) is also shown. On the right, the minimum Euclidean and squared Mahalanobis distances between two components from the same mixture are also shown. On average, components were located 189.045 units from each other, with an average Mahalanobis distance of 20,825.7.

Condition	$C(G)$	Number of edges with X modes				MM %	Minimum distance	Minimum Mahalanobis
		$X = 1$	$X = 2$	$X = 3$	$X = 4$			
1	1	255	1	0	0	0.4	38.996	509.816
2	2	254	2	0	0	0.8	15.909	160.058
3	3	253	3	0	0	1.2	70.701	1917.990
4	4	252	4	0	0	1.6	29.372	428.785
5	8	248	8	0	0	3.2	21.000	416.107
6	16	240	16	0	0	6.4	5.428	9.109
7	32	224	32	0	0	12.8	12.322	134.981
8	7.92	251	0	5	0	2.0	6.957	180.949
9	8	252	0	0	4	1.6	5.419	98.743
10	15.92	244	6	5	1	4.8	25.285	362.238
11	31.85	232	12	10	2	9.6	6.052	60.956

times per graph. All other methods are deterministic and thus did not need additional trials.

Table 2 summarizes the number of trials that delivered a result within five times the residual of the Exhaustive SGD method, which is a canonical comparison basis as it represents the best achievable result by all methods described in this paper. Exhaustive LM did not perform quite as well on these graphs, most probably due to the differences in the underlying SGD and LM optimization methods (see Section 2.2.3). Both methods are initialized with the same graph and starting conditions. LM is very sensitive to the starting conditions, while SGD is slightly more robust in that respect. LM was run without the robust cost function for the exhaustive case, since no outliers are expected due to small errors in the reduction process. At large, the differences between LM and SGD in the exhaustive case are coincidental and not of concern in this paper. They are to show what is achievable with either method, and mainly used as a comparison basis with the reduction heuristics discussed below.

From this brief summary, it is clear that the standard graph-based methods Max SGD and Max LM do not work well with multimodal registration results, even if there are only very few of them. Max LM can sometimes achieve reasonable results in a few relatively simple conditions,

Table 2. Successes by method and condition (Table 1) in terms of the number of trials that produced a result within 5 times the SSE_{xy} and SSE_{θ} error of the exhaustive SGD method. The particle method used 10,000 particles.

Condition	Max LM	Max SGD	Particle	Multi-Edge LM	Prefilter LM	Prefilter SGD
1	40%	60%	0%	40%	80%	100%
2	0%	40%	0%	0%	0%	100%
3	30%	0%	0%	0%	60%	100%
4	40%	0%	0%	0%	60%	100%
5	20%	0%	0%	0%	40%	100%
6	0%	0%	0%	0%	50%	100%
7	0%	0%	0%	0%	70%	90%
8	20%	0%	0%	0%	70%	100%
9	20%	0%	0%	0%	60%	100%
10	0%	0%	0%	0%	70%	100%
11	0%	0%	0%	0%	50%	100%

mostly because of its robust cost function. Max SGD only achieves very few good results in two simplest conditions. With increasing complexity, the Max SGD and Max LM methods do not achieve any reasonable results anymore. It is also clear that the Particle method often fails to converge, especially with very multimodal graphs, even though an extremely large number of 10,000 particles was used.

More surprising is the complete failure of the Multi-Edge LM method. It appears that instead of converging to a single mode per constraint, it converged to some configuration in between multiple modes. Thus, the final distance to ground truth is quite large.

The performance of the Prefilter SGD method proposed here is much better and more consistent. It achieves a very accurate result in nearly all cases. An illustrative example result of the Max SGD/LM and Prefilter SGD methods is shown in Figure 2.

Figure 3 shows the distribution of *SSE* residual errors after optimization for each method by condition. Since these distributions can be very skewed, the five-number summary (minimum, lower quartile, median, upper quartile, and maximum) was used to better represent the performance of each method in these figures. The figure shows that Prefilter SGD's performance is almost identical to the Exhaustive SGD method, which obviously requires substantially more computation time. Here, it is also clear that Exhaustive LM and Prefilter LM converge to nearly the same result as well. The difference of the final results is rather caused by the differences between the underlying SGD and LM optimization methods, not the quality of the Prefilter approach.

The Multi-Edge LM method takes all components into account, but not their weights. Unfortunately, the conceptually interesting Multi-Edge LM method presents similarly bad results as the Max SGD/LM methods. In the lower complexity graphs it performs well, but it diverges significantly already at a complexity of 8 (condition 5) or more.

The high similarity of results achieved by the Prefilter and exhaustive methods, when used with the same underlying optimization method, shows that the Prefilter step as described in Section 2.2.2 allows a very close approximation of the combination of modes used by the exhaustive method in significantly less time. While the exhaustive search requires exponential time in the graph complexity, i.e. $O(2^{C(G)})$, the filtering is done in $O(|V|)$. Since the optimal combination is only approximated, the filtering can report a wrong assignment, which happened for example

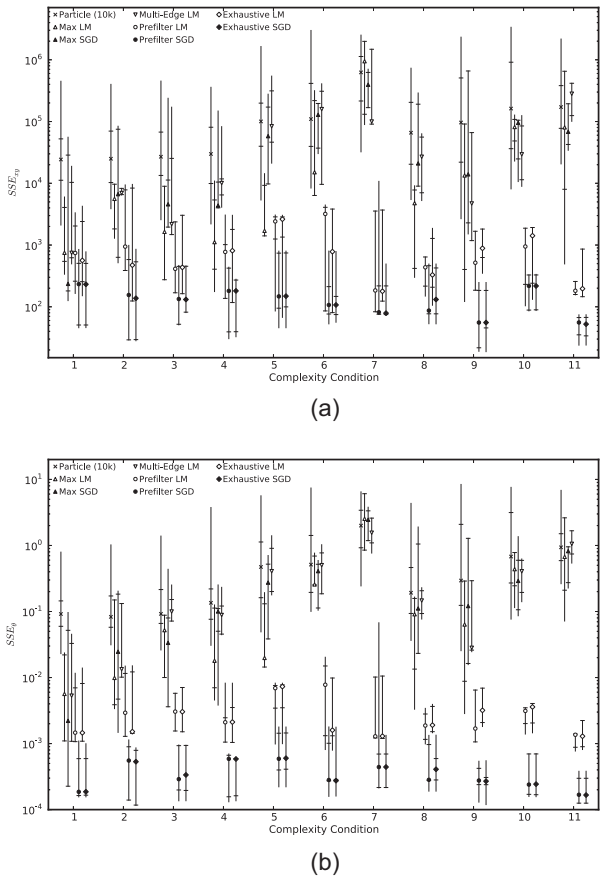

Fig. 3. Performance of each method by condition. Reported figures are the minimum, lower quartile, median, upper quartile, and maximum of the residual *SSE* errors relative to ground truth. Smaller is better. The marker location specifies the median, a vertical line is drawn between the minimum and maximum, and horizontal line ticks indicate the quartiles. In some cases, the median marker may occlude the quartile marks. Note the logarithmic scale on the *y*-axis. Here SSE_{xy} is shown in the top graph, the bottom shows SSE_{θ}.

Table 3. Runtimes (means and standard deviations) in seconds by method and condition. The experiments were run on an Intel Core i7-2720QM, 2.2 GHz, 8 GB RAM. All implementations were done in C++. The Particle method used 10,000 particles.

Condition	Exhaustive LM	Exhaustive SGD	Max LM	Max LM	Multi-Edge LM	Particle	Prefilter LM	Prefilter SGD
1	0.0333	0.0659	0.0262	0.0298	0.0452	2.2399	0.0231	0.0392
	σ0.0056	σ0.0022	σ0.0112	σ0.0010	σ0.0307	σ0.0591	σ0.0083	σ0.0262
2	0.1447	0.1390	0.0350	0.0322	0.0306	2.2329	0.0224	0.0299
	σ0.0969	σ0.0025	σ0.0174	σ0.0031	σ0.0208	σ0.0466	σ0.0072	σ0.0005
3	0.2544	0.2755	0.0553	0.0303	0.0522	2.5432	0.0269	0.0309
	σ0.1393	σ0.0091	σ0.0568	σ0.0009	σ0.0328	σ0.4449	σ0.0081	σ0.0021
4	0.4504	0.5494	0.0475	0.0310	0.0691	2.4625	0.0356	0.0309
	σ0.2660	σ0.0153	σ0.0380	σ0.0023	σ0.0533	σ0.3710	σ0.0138	σ0.0023
5	18.5471	8.9363	0.0227	0.0303	0.0762	2.8486	0.0245	0.0301
	σ8.5286	σ0.2580	σ0.0058	σ0.0013	σ0.0416	σ0.4083	σ0.0098	σ0.0012
6	$\approx 10^{3.7}$	$\approx 10^{3.3}$	0.0376	0.0297	0.0533	2.6239	0.0322	0.0308
	–	–	σ0.0352	σ0.0012	σ0.0435	σ0.4432	σ0.0084	σ0.0027
7	$\approx 10^{8.5}$	$\approx 10^{8.2}$	0.0719	0.0298	0.0423	2.8482	0.0354	0.0305
	–	–	σ0.0274	σ0.0009	σ0.0304	σ0.3115	σ0.0111	σ0.0021
8	18.0294	4.1744	0.0202	0.0310	0.0534	2.5284	0.0205	0.0298
	σ8.7641	σ4.1132	σ0.0079	σ0.0021	σ0.0293	σ0.3911	σ0.0070	σ0.0008
9	12.4474	4.4003	0.0465	0.0305	0.0790	2.8559	0.0385	0.0304
	σ8.7379	σ4.3367	σ0.0352	σ0.0017	σ0.0316	σ0.3266	σ0.0167	σ0.0014
10	$\approx 10^{3.7}$	$\approx 10^{3.7}$	0.0738	0.0306	0.0402	2.7521	0.0289	0.0304
	–	–	σ0.0500	σ0.0005	σ0.0330	σ0.3410	σ0.0063	σ0.0015
11	$\approx 10^{8.4}$	$\approx 10^{8.4}$	0.0287	0.0313	0.0295	2.5556	0.0276	0.0316
	–	–	σ0.0166	σ0.0016	σ0.0209	σ0.3424	σ0.0076	σ0.0017

in condition 7 where especially Prefilter SGD produced a larger variance of results (see both Table 2 and Figure 3).

The Particle method did not converge to a good result consistently, even with 10,000 particles. It appears that the globally best particles are assigned too little probability early on in the filtering process and are thus filtered out during the resampling phase. However, even further increasing the number of particles significantly did not show much improvement; neither did increasing or lowering the minimum effective particle count N_{min} in algorithm 1.

Another interesting observation is that the residual error to ground truth of the Max SGD and Max LM methods, as shown in Figure 3, actually increases exponentially with the graph complexity $C(G)$ (note the logarithmic scale). The more complex the graph, the worse the approximation that Max SGD/LM relies on.

Table 3 shows measured runtimes by method and condition. The Max SGD/LM, Multi-Edge LM, and Prefilter SGD/LM methods have nearly constant runtimes, as their computational cost is dominated by the underlying SGD or LM implementation, which in turn depends only on the size of the graph.

The filtering of components only requires around 1 ms for the specific size of the graphs studied here, which results in a very small increase of computation time in the Prefilter SGD/LM methods. Some runtimes of Exhaustive SGD and Exhaustive LM were not recorded, as the exponential factor would have made the experiment infeasible, e.g. individual trials in conditions 7 and 11 would have taken almost 10 years to complete. However, because the ground truth is known, it was still possible to compute the ideal results by only considering the correct combination.

Table 4. Additional trials of Prefilter SGD to investigate its asymptotical behavior. Note the exponential complexity. The mean runtime is shown in seconds. Means and standard deviations of the final *SSE* error are reported. As in Table 2, successful trials are reported as a percentage within five times the *SSE* error of Exhaustive SGD.

$C(G)$	32	64	128	256	512
Runtime (s)	0.03	0.05	0.93	44.32	232.22
Successes	100%	80%	10%	20%	0%
SSE_{xy}	51.07	535.46	3,250.20	5,255.02	155,257.52
	σ16.86	σ840.28	σ3,654.21	σ5,607.53	σ348,952.17
SSE_{θ}	0.00	0.03	0.08	0.22	0.91
	σ0.00	σ0.07	σ0.05	σ0.17	σ1.34

Note that Table 4 shows a few additional results to provide some idea of the asymptotical behavior of Prefilter SGD. More graphs of the same size as those described in Table 1 were generated with up to 11 modes per edge and a complexity of up to $C(G) = 512$. In the most complex graph, only a single edge was left with a unimodal Gaussian constraint. The table shows the predicted combinatorial explosion.

3.2. *The influence of good odometry estimates*

We believe that good odometry estimates are the single most important factor that allowed traditional methods to succeed so far. In order to substantiate this postulate, and also to compare with the results achieved by Stachniss et al.

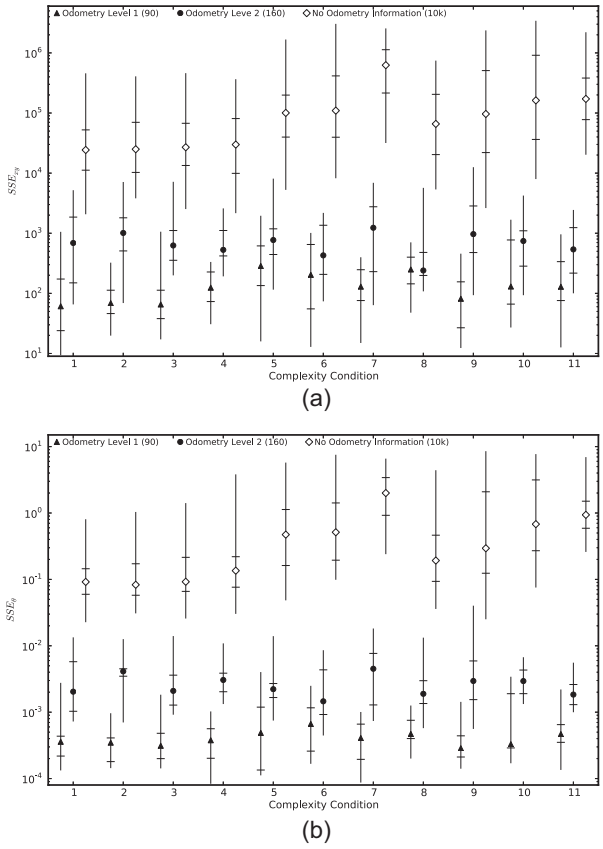

Fig. 4. Performance of the Particle method by condition and different levels of fused odometry. Reported figures are the minimum, lower quartile, median, upper quartile, and maximum. The marker location specifies the median, a line is drawn between the minimum and maximum, and line ticks indicate the quartiles. Note the logarithmic scale on the y-axis. Here SSE_{xy} is shown in the top graph, the bottom shows SSE_{θ}. Also, note the different number of particles used for the three cases (90, 160, and 10,000).

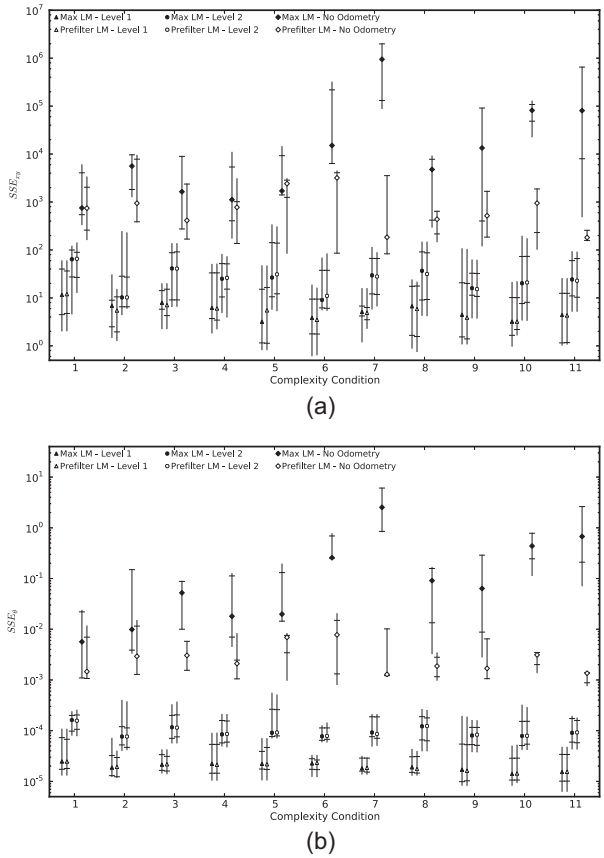

Fig. 5. Performance of the Max LM and Prefilter LM methods by condition and different levels of fused odometry. Reported figures are the minimum, lower quartile, median, upper quartile, and maximum. The marker location specifies the median, a line is drawn between the minimum and maximum, and line ticks indicate the quartiles. Note the logarithmic scale on the y-axis. Here SSE_{xy} is shown in the top graph, the bottom shows SSE_{θ}.

(2007), a number of trials were run with odometry estimates fused with the MoG pose graphs from the previous results.

Odometry with two separate noise levels was used to illustrate their effect in the context of the particle method. The odometry noise added to the ground truth transformation was computed as follows:

$$\sigma_{xy} = v_{xy} \cdot d \tag{21}$$

$$\sigma_{\theta} = v_{\theta} \cdot \max(.4\pi, a + .00001d) \tag{22}$$

$$odo_i^j = x_j \ominus x_i \oplus \mathcal{N}(0, diag(\sigma_{xy}, \sigma_{xy}, \sigma_{\theta})), \tag{23}$$

where d is the ground truth distance and a is the absolute ground truth rotation angle. In the first noise level (level 1), $v_{xy} = 0.01$, and $v_{\theta} = 0.002$. In the second noise level, $v_{xy} = 0.04$, and $v_{\theta} = 0.004$. For each noise level, one odometry estimate per edge was generated and fused with the corresponding edge using the update equations described in Appendix C.

Figure 4 shows the performance of the Particle method on these fused graphs, as well as the unfused graphs for comparison. For the first noise level, the number of particles was set to 90. A total of 160 particles were used for the second noise level. A lower number of particles was needed with odometry than without since the fusion process severely discounted modes far away from the odometry result and thus reduced multimodality. These results are comparable with those of Stachniss et al. (2007).

It is clear to see that the particle method does not converge to a good solution without odometry information, and in fact presents a very large variance in all computed solutions. The particle method cases with fused odometry show a much reduced variance and better results, especially in the rotation error. This is to be expected due to the corrective effects of odometry. Note that the recovered trajectory was of good quality in the cases with odometry while using a relatively reasonable number of particles, replicating and validating the findings of Stachniss et al. (2007). In addition, it is clear from the data that with increasing multimodality of the graph, the performance of the particle filter, even with fused odometry, decreases.

Figure 5 shows the performance of the Max LM and Prefilter LM methods on the fused graphs. One main

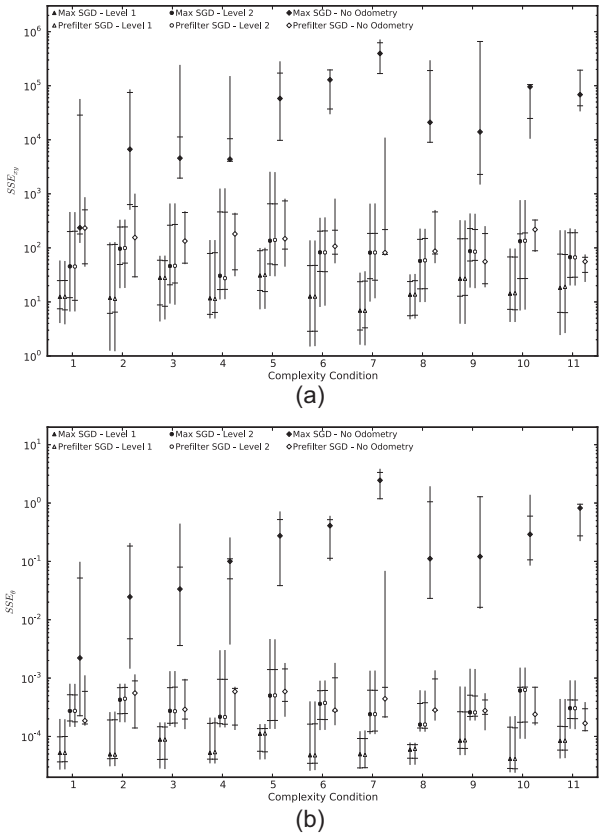

Fig. 6. Performance of the Max SGD and Prefilter SGD methods by condition and different levels of fused odometry. Reported figures are the minimum, lower quartile, median, upper quartile, and maximum. The marker location specifies the median, a line is drawn between the minimum and maximum, and line ticks indicate the quartiles. Note the logarithmic scale on the *y*-axis. Here SSE_{xy} is shown in the top graph, the bottom shows SSE_θ.

observation is that with odometry, the two methods perform virtually identically. The strong effect of the odometry estimate on the MoG weights, even with very noisy odometry, allows the Max LM method to choose the right component. However, as odometry noise increases, the fused component means are further away from ground truth, and both methods converge to a worse result. Much the same can be seen in Figure 6, showing the performance of the Max SGD and Prefilter SGD methods. The reweighting effect of the odometry estimates is present here as well.

Table 5 shows the times required to run the Particle method on the fused pose graphs. The Particle method exhibits a rather constant, but large, time requirement, linear with the number of particles. Naturally, the lower number of particles required in the cases with odometry decreases the runtimes accordingly. In addition, less components have to be sampled, depending on the fusion result. The time required for the graph-based methods remained the same as in the cases without odometry discussed above as the size of the graphs did not change. Table 6 shows the performance of the Particle, Max LM, Prefilter LM, Max SGD, and Prefilter SGD methods on the graphs with odometry information and, as a reference, on those without as well. In general, the result achieved with odometry of any level is much better.

Table 5. Runtimes (means and standard deviations) in seconds for the particle method for different levels of odometry applied to the initial pose graph. The experiments were run on an Intel Core i7-2720QM, 2.2 GHz, 8 GB RAM.

Condition	Level 1 (90)	Level 2 (160)	None (10,000)
1	0.0197	0.0335	2.2399
	σ0.0013	σ0.0014	σ0.0591
2	0.0195	0.0331	2.2329
	σ0.0007	σ0.0013	σ0.0466
3	0.0199	0.0333	2.5432
	σ0.0011	σ0.0013	σ0.4449
4	0.0199	0.0332	2.4625
	σ0.0008	σ0.0011	σ0.3710
5	0.0197	0.0335	2.8486
	σ0.0009	σ0.0014	σ0.4083
6	0.0202	0.0340	2.6239
	σ0.0010	σ0.0017	σ0.4432
7	0.0202	0.0336	2.8482
	σ0.0008	σ0.0013	σ0.3115
8	0.0195	0.0337	2.5284
	σ0.0008	σ0.0015	σ0.3911
9	0.0198	0.0336	2.8559
	σ0.0009	σ0.0014	σ0.3266
10	0.0198	0.0336	2.7521
	σ0.0010	σ0.0017	σ0.3410
11	0.0199	0.0335	2.5556
	σ0.0009	σ0.0017	σ0.3424

4. Application to plane-based registration and real-world data sets

4.1. Multimodal estimates from plane-based registration

It is important to note that the specific method used to generate MoG motion estimates or registration results is separate from the general theoretical framework introduced above. Much like in the traditional unimodal case, the choice of sensors or of the registration method is generally irrelevant to the pose graph data structure and the optimization method used in the SLAM system.

Since traditional registration methods usually report the uncertainty of the single registration result as a linear Gaussian, some changes are needed to generate MoG constraints. Many possible methods come to mind. For example, there is the option of using any iterative method such as ICP (Besl and McKay, 1992) that may converge to different local minima given different perturbed initial guesses, much like the computation of the proposal distribution of Stachniss et al. (2007). Also, many registration methods already generate a list of ranked results, which can be employed in a canonical way for generating a combined MoG result. A concrete example for the second case is presented in this section, namely the extension of plane based registration algorithm to generate multimodal constraints.

Table 6. Successes by the Particle, Max LM, Prefilter LM, Max SGD, and Prefilter SGD methods (as in Table 2). Odometry information with different noise levels (L1 and L2) was merged with the pose graphs before the methods were applied. 'None' means no odometry was fused, as with the other methods summarized in Table 2. The unfused results are shown for comparison.

Condition	Particle (90/160/10,000)			Max LM			Prefilter LM			Max SGD			Prefilter SGD		
	L1	L2	None	L1	L2	None	L1	L2	None	L1	L2	None	L1	L2	None
1	80%	80%	0%	100%	100%	40%	100%	100%	80%	100%	90%	60%	100%	90%	100%
2	100%	30%	0%	100%	100%	0%	100%	100%	0%	100%	100%	40%	100%	100%	100%
3	80%	50%	0%	100%	100%	30%	100%	100%	60%	100%	90%	0%	100%	90%	100%
4	90%	60%	0%	100%	100%	40%	100%	100%	60%	100%	80%	0%	100%	80%	100%
5	80%	50%	0%	100%	100%	20%	100%	100%	40%	100%	80%	0%	100%	80%	100%
6	60%	70%	0%	100%	100%	0%	100%	100%	50%	100%	100%	0%	100%	100%	100%
7	100%	40%	0%	100%	100%	0%	100%	100%	70%	100%	90%	0%	100%	90%	90%
8	90%	80%	0%	100%	100%	20%	100%	100%	70%	100%	100%	0%	100%	100%	100%
9	90%	30%	0%	100%	100%	20%	100%	100%	60%	90%	80%	0%	90%	90%	100%
10	90%	70%	0%	100%	100%	0%	100%	100%	70%	100%	90%	0%	100%	90%	100%
11	70%	10%	0%	100%	100%	0%	100%	100%	50%	100%	100%	0%	100%	100%	100%

The plane matching algorithm introduced in Pathak et al. (2010c) is based on large planar surface patches extracted from range scans. Concretely, plane matching uses an algorithm called minimally uncertain maximal consensus (MUMC) to determine the unknown plane correspondences through maximizing geometric consistency by minimizing the uncertainty volume in configuration space. These correspondences give rise to a least-squares transformation estimate that respects the plane parameter uncertainties computed during plane extraction. The method also includes closed form expressions for the final transformation covariances.

Plane matching is a very recent method, but it has already been used successfully in several applications where only very limited overlap occurs between scans (Pathak et al., 2010b,a,d). Nevertheless, multimodal results can also be observed here. MUMC produces a ranked list of candidate transformations where the top-ranked result is often the correct one. As long as some parts of the data overlap, the correct transformation usually does appear in the list, but not necessarily as the highest ranked one.

Within the main processing loop of the plane-based registration algorithm from Pathak et al., new hypotheses are created during each execution and appended to a list (Pathak et al., 2010c, Algorithm 2). Instead of choosing only the best result $\bar{\omega}$, the list $\mathcal{W}$ of all potential results is post-processed here to deliver a small amount of good results to incorporate in multimodal constraints. The list $\mathcal{W}$ consists of the tuples ω_i, where each tuple contains a possible registration result and some accompanying information and metrics. In the following, the notation $\omega.a$ is used to denote a member a of a tuple ω. Specifically, each tuple ω in $\mathcal{W}$ contains the following:

1. the translation vector ${}^{\ell}_{r}\mathbf{t}$;
2. the rotation quaternion ${}^{\ell}_{r}\check{\mathbf{q}}$;
3. the translation covariance ${}^{\ell}_{r}\mathbf{C}_{\mathbf{tt}}$;
4. the rotation covariance ${}^{\ell}_{r}\mathbf{C}_{\check{\mathbf{q}}\check{\mathbf{q}}}$;
5. the uncertainty volume α;
6. the plane correspondence overlap metric o_p.

Note that the same notation as in Craig (2005) is used to describe the respective coordinate frames of the plane clouds as left (denoted l) and right (r). See also Pathak et al. (2010c) for a more detailed description.

The overlap metric o_p is computed as follows:

$$o_p = \frac{\#\Gamma}{\#{}^{r}\mathcal{P}} \tag{24}$$

where $\#\Gamma$ is the number of used plane correspondences, and $\#{}^{r}\mathcal{P}$ is the number of planes in the right plane cloud (Pathak et al., 2010c). In effect, this quantifies how much of the complete scene described by planes was used to compute the registration result. The more planes are available for matching, the more should be used as correspondences in a successful match. However, since we expect ambiguous results also due to low overlap, a high threshold for o_p only prevents coincidental results in very cluttered scenes. As a side effect of the explicit inclusion of multiple registration results, the overlap o_p is allowed to be significantly lower, yielding more results that would otherwise have been discarded. This allows the inclusion of locally not very likely registration results that may be globally correct.

In some cases, these parameter settings produce up to 200 results in the list $\mathcal{W}$, many of which were either very similar to each other due to minor variations in the set of correspondences used, or simply very unlikely. To reduce the number of redundant results and to exclude very unlikely ones, this list is processed with Algorithm 3.

Several parameters are used to quickly discard solutions in algorithm 3:

1. $\mathcal{O}_p$ dictates the minimum overlap allowed overall for a solution to be considered.
2. $\mathcal{L}_{unc}$ expresses the maximum uncertainty volume α relative to the least uncertain solution in $\mathcal{W}$. So, the actual maximum uncertainty volume is $\alpha_{max} = \mathcal{L}_{unc} \cdot \omega_{min}.\alpha$.
3. $\mathcal{O}_{min}$ is the minimum overlap relative to the least uncertain solution in $\mathcal{W}$. So the actual minimum overlap is $o_{min} = \mathcal{O}_{min} \cdot \omega_{min}.o_p$.

Algorithm 3: Post-processing of the complete list of potential solutions $\mathcal{W}$.

input : A list of all consistent registrations $\mathcal{W}$ from Algorithm 2 of Pathak et al. (2010c)
output: The reduced list of potential solutions $\mathcal{W}^*$

Initialize $\mathcal{W}^* = \emptyset$
Sort the elements $\omega_i \in \mathcal{W}$ by uncertainty volume of the solution $\omega_i.\alpha$.
Set ω_{min} to the first $\omega_i \in \mathcal{W}$ where the maximum eigenvalue of $\omega_i.{}^{\ell}_{r}\mathbf{C}_{\check{\mathbf{q}}\check{\mathbf{q}}}$ and $\omega_i.{}^{\ell}_{r}\mathbf{C}_{\mathbf{tt}}$ is less than λ_{max}.
if *no such* ω_{min} *exists* **then**
  return
end
Set $\mathcal{W}^* = \{\omega_{min}\}$
Set $\alpha_{max} = \mathcal{L}_{unc} \cdot \omega_{min}.\alpha$.
Set $o_{min} = \mathcal{O}_{min} \cdot \omega_{min}.o_p$.
for $\forall \omega_i \in \mathcal{W}$ **do**
  if $\omega_i.o_p < \mathcal{O}_p$ **or** $\omega_i.o_p < o_{min}$ **or** *max eigenvalue of* $\omega_i.{}^{\ell}_{r}\mathbf{C}_{\check{\mathbf{q}}\check{\mathbf{q}}}$ *or* $\omega_i.{}^{\ell}_{r}\mathbf{C}_{\mathbf{tt}} > \lambda_{max}$ **or** $\omega_i.\alpha > \alpha_{max}$ **then**
    continue
  end
  Set $j_{rep} = -1$
  for $\forall \omega^*_j \in \mathcal{W}^*$ **do**
    if $||\omega_i.{}^{\ell}_{r}\mathbf{t} - \omega^*_j.{}^{\ell}_{r}\mathbf{t}|| < \mathcal{D}_{min}$ **or** $\omega_i.{}^{\ell}_{r}\check{\mathbf{q}} \oslash \omega^*_j.{}^{\ell}_{r}\check{\mathbf{q}} < \mathcal{R}_{min}$ **then**
      $j_{rep} = j$
      break;
    end
  end
  if $j_{rep} == -1$ **then**
    Append $\mathcal{W}^* \leftarrow \mathcal{W}^* \cup \{\omega_i\}$
  end
  else
    This ω_i may replace $\omega^*_{j_{rep}}$.
    if $\omega_i.o_p > \omega^*_{j_{rep}}.o_p$ **and** $\omega_i.o_p < \mathcal{O}_{max}$ **and** $\omega_i.\alpha < \omega^*_{j_{rep}}.\alpha \cdot \mathcal{L}_{rep}$ **and** $||\omega_i.{}^{\ell}_{r}\mathbf{t} - \omega^*_j.{}^{\ell}_{r}\mathbf{t}|| < \mathcal{D}_{max}$ **and** $\omega_i.{}^{\ell}_{r}\check{\mathbf{q}} \oslash \omega^*_j.{}^{\ell}_{r}\check{\mathbf{q}} < \mathcal{R}_{max}$ **then**
      Replace $\omega^*_{j_{rep}}$ with ω_i
    end
  end
end

4. $\mathcal{D}_{min}$ is the minimum Euclidean distance between accepted solutions.
5. $\mathcal{R}_{min}$ is the minimum angular distance between accepted solutions, which is computed as the absolute angle value of the rotation between two solutions in the axis-angle notation ($\oslash$ in the algorithm).
6. λ_{max} is the maximum covariance eigenvalue allowed.

In addition, some parameters are needed to determine that a slightly less likely solution (based on the uncertainty volume) may replace another more likely one. This is needed if a very likely solution, that only has very few correspondences, is close to but not as accurate as a slightly less likely solution that used many more correspondences:

1. $\mathcal{L}_{rep}$ expresses the maximum relative uncertainty volume the replacement is allowed to have.
2. $\mathcal{O}_{max}$ is the maximum overlap that allows a potential solution to be considered to be replaced.
3. $\mathcal{D}_{max}$ is the maximum Euclidean distance to the replacement candidate.
4. $\mathcal{R}_{max}$ is the maximum angular distance to the replacement candidate.

Each result left in $\mathcal{W}^*$ after post-processing is used as a component in the final MoG for a constraint in the graph. Several ways to compute a good mixture weight for each solution were tried, however the specific method did not seem to make a difference in practice. In the experiments below, the weights are close to uniform, but their ordering keep the original ordering as returned by the processing algorithm above.

4.2. Bremen city center

The first experiment with real-world data is based on 13 scans that were recorded with a Riegl VZ-400 in the center of Bremen, Germany. Each point cloud consists of between

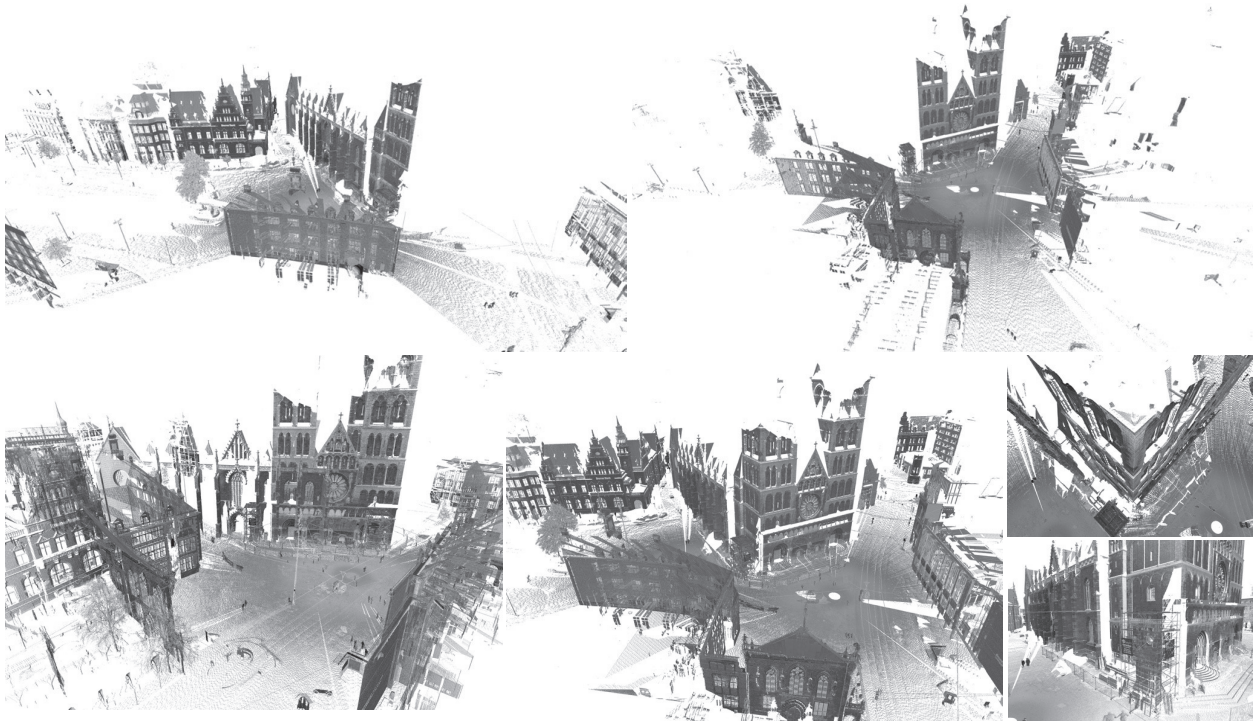

Fig. 7. Above: Scans 1 (right) and 2 (left) of the Bremen City data set. Below: The effects of ambiguity due to occlusion on plane-registration of the scan pair. The left image shows the most likely result as reported by the plane registration method. The second most likely registration result, shown on the right, is the globally correct but less certain one (see the covariance determinants in the first row of Table 8). Detail views of where the scans meet at the church tower are shown in the right column. Note that no odometry exists to disambiguate the results.

Table 7. Plane matching parameters used for the Bremen City data set.

$\mathcal{O}_p$	0.1	λ_{max}	800000
$\mathcal{L}_{unc}$	e^{15}	$\mathcal{L}_{rep}$	e^{3}
$\mathcal{O}_{min}$	0.667	$\mathcal{O}_{max}$	0.24
$\mathcal{D}_{min}$	5	$\mathcal{D}_{max}$	0.2
$\mathcal{R}_{min}$	0.065	$\mathcal{R}_{max}$	0.005

15 and 20 million points with reflectance information. The scanner was mounted on a tripod without a mobile base, thus no odometry information is available. However, markers were placed in the environment beforehand to allow for a comparison with the 'gold standard' for geodetic applications, i.e. registration with artificial markers in the Riegl software which requires additional manual assistance in the process such as confirming or re-selecting correspondences. This registration based on artificial markers can also be used to seed methods that need a good initial guess, e.g. for ICP-based methods such as 6D-SLAM (Borrmann et al., 2008). Note that no initial guess, i.e. no initial marker-based registration, no motion estimates, no GPS, or anything similar is used in the results presented here, other than as a comparison.

Translations between the different scanning locations were quite large, up to 50 m. As mentioned, 6D-SLAM requires the marker based Riegl registration as an initial guess to successfully run on this data set. Plane registration, in contrast, performs very well already without any initial motion estimates (Pathak et al., 2010d). On most pairs, the plane registration results are very close to the marker-based registration, and even appear to be more precise under close visual inspection of the point cloud data. But the method also fails on several scan pairs. It can be noted that these pairs tend to have very large occlusions and very low overlap, and hence display quite some ambiguities in terms of possible plane correspondences. Furthermore, it can be noted that the correct solutions for these pairs are among the top candidates in the MUMC ranking.

One example where occlusion results in diminished overlap and thus to ambiguous registration results is shown in Figure 7. The plane matching parameters used are shown in Table 7. The prefilter step again used $N = 200$ samples in the filtering process, as in the synthetic case above.

Using the multimodal plane registration method as described in Section 4.1, the 13 scans and the registrations between them gave rise to a graph with a total of 13 vertices and 23 edges, of which 7 are multimodal. This MoG pose graph has a complexity $C(G) = 7.58496$. Table 8 gives an

Table 8. The multimodal edges in the Bremen City map. The left column shows the edge in question, the other columns show the list of modes in the order reported by the plane-based registration. Each mode is shown in form of the estimated translation $T(\cdot)$ and rotation $R(\cdot)$ as well as the determinant of the covariance $det(C)$ associated with it. Furthermore, each globally correct mode is highlighted in gray.

Pair	#1	#2	#3
$1 \rightarrow 2$	$T = (8.22, -4.88, 0.45)$ $R = (2.03, 0.69, -68.12)$ $\det(C) = 2.38 \times^{-23}$	$T = (-1.67, 41.22, -0.26)$ $R = (2.30, -0.02, -157.07)$ $\det(C) = 6.97 \times^{-19}$	–
$5 \rightarrow 6$	$T = (-22.21, 0.14, 0.27)$ $R = (0.78, 0.77, 72.69)$ $\det(C) = 4.94 \times^{-26}$	$T = (-22.14, 0.24, 6.67)$ $R = (0.64, 0.76, 72.69)$ $\det(C) = 2.08 \times^{-23}$	–
$6 \rightarrow 7$	$T = (-21.30, 5.89, 0.19)$ $R = (-1.41, 1.10, -132.24)$ $\det(C) = 7.03 \times^{-22}$	$T = (0.85, 4.52, -0.08)$ $R = (-2.31, 0.58, -42.07)$ $\det(C) = 6.70 \times^{-16}$	–
$1 \rightarrow 12$	$T = (-47.18, -48.63, -1.49)$ $R =, (-1.11, -0.13, 139.99)$ $\det(C) = 2.26 \times^{-21}$	$T = (3.13, 2.22, 0.10)$ $R = (0.14, -0.80, 50.63)$ $\det(C) = 6.92 \times^{-20}$	–
$8 \rightarrow 10$	$T = (40.83, 15.20, -1.25)$ $R = (1.54, -3.78, -176.13)$ $\det(C) = 6.89 \times^{-16}$	$T = (1.15, 16.00, 0.04)$ $R = (0.06, -1.22, -113.78)$ $\det(C) = 2.33 \times^{-13}$	$T = (-8.52, 12.75, 0.21)$ $R = (2.14, -1.13, 10.13)$ $\det(C) = 1.05 \times^{-12}$
$9 \rightarrow 11$	$T = (-20.42, 49.43, -1.27)$ $R = (1.47, 0.90, 28.85)$ $\det(C) = 1.54 \times^{-11}$	$T = (7.73, 32.00, -0.44)$ $R = (2.34, 0.78, 124.47)$ $\det(C) = 1.14 \times^{-8}$	–
$9 \rightarrow 12$	$T = (2.64, 3.50, -0.08)$ $R = (-0.49, -1.19, -53.71)$ $\det(C) = 5.83 \times^{-19}$	$T = (-41.38, 61.60, -1.75)$ $R = (-0.87, -0.87, -50.20)$ $\det(C) = 2.86 \times^{-13}$	–

overview over the seven multimodal edges, especially their modes in terms of the estimated transformations in the form of translations T and rotations R as well as the determinant of the covariances $\det(C)$ associated with them. The smaller the determinant, the more certain the candidate registration result. Using the settings described in Table 7, at most three modes per edge were encountered. For each pair shown in Table 8, the correct mode is indicated by gray shading. Note that four times the optimum is correct while three times the second best choice is the proper solution. Also, higher-ranked components than the globally correct ones are all supposedly more certain, showing the need to take less certain registration results into account to achieve global convergence. Furthermore, note that the modes tend to be quite far apart, i.e. there tend to be significantly different spatial transformations associated with them. This is only partially an effect of the post-processing applied to the MUMC results as it allows a minimum distance of 5 m between components (see Table 7), while distances between the reported components are much larger. Choosing the wrong mode has hence strong impacts on SLAM, which cannot be easily repaired with additional loop closures.

Figures 8 and 9 show the final maps after optimization using both the traditional unimodal Max SGD method and the Prefilter SGD/LM methods. The results from Prefilter LM and Prefilter SGD are virtually indistinguishable with almost the same residual error (see Table 9). The planar representation along with the pose graph structure is shown in Figure 8. Figure 9 shows a high-quality point cloud rendering using the mapped reflectance value as color. It is quite obvious that the traditional Max SGD/LM methods fail while Prefilter SGD as well as Prefilter LM are able to find good map estimates.

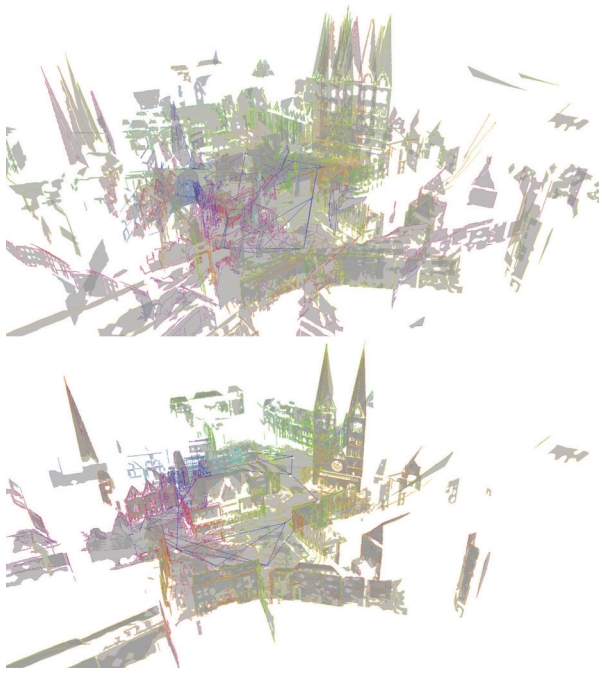

Fig. 8. Final maps in plane patch representation after optimization with the traditional unimodal Max SGD method (top) and the multimodal Prefilter LM (bottom). The planes used for matching as well as the graph structure is shown. The log probability of the traditional result is -1.26933×10^9, while the log probability of the multimodal result is -1.01384×10^5.

Table 9. Runtimes in seconds and result quality for the traditional Max SGD/LM, Multi-Edge LM, and Prefilter SGD/LM on the Bremen City data set. Recorded on a Core i7-2720QM, 2.2 GHz with 8 GB of RAM. The *SSE* metric (see Appendix B) was computed relative to the de-facto ground truth transformations given by the marker based registration. Max SGD/LM were initialized with a breath-first traversal of the graph, the rest of the methods were initialized with the Prefilter method described in Section 2.2.2.

Method	Initialization	Optimization	Log probability	SSE_{xyz}	$SSE_{\rho\theta\phi}$
Max SGD	0.000226	0.006497	-1.26001×10^9	3.65999×10^9	7.92405×10^{-1}
Max LM	0.000231	0.001963	-3.56238×10^9	6.63873×10^9	1.11384×10^0
Multi-edge LM	0.000276	0.022238	-1.45329×10^{10}	2.65199×10^9	1.61360×10^0
Prefilter SGD	0.000279	0.006288	-1.30253×10^5	1.02629×10^5	3.19938×10^{-5}
Prefilter LM	0.000269	0.003624	-1.01384×10^5	1.02628×10^5	3.30090×10^{-5}

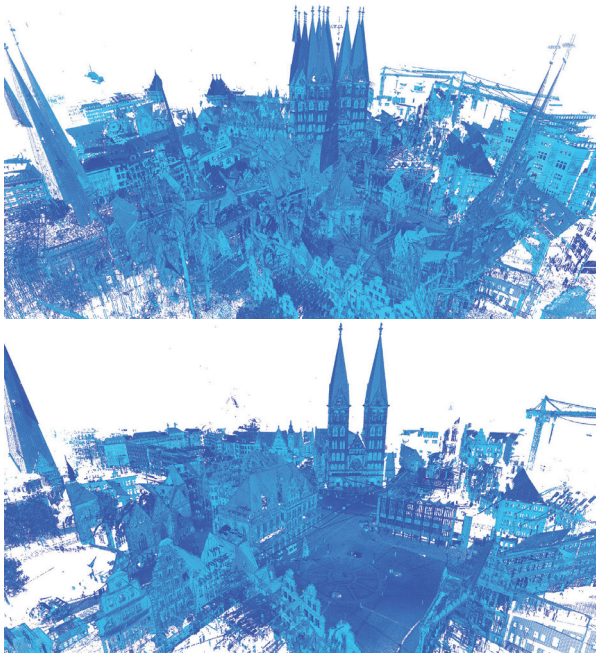

Fig. 9. Mapping results in point cloud representation using the traditional unimodal Max SGD method (top), and the multimodal Prefilter LM method (bottom). Laser reflectance values are used for assigning grayscale values (coded with the Jet colormap in color). See also Extension 1 for an animated view of these maps.

Two comparisons to ground truth are made, once using the marker-based registration, and once by superimposing the map on aerial imagery from Google Earth. Table 9 shows runtimes as well as the quality of the results based on final log probability and an error metric relative to the marker-based transformations. Figure 10 shows the final multimodal map computed by Prefilter LM in relation to the Google Earth imagery.

The values in Table 9 clearly show that for negligible computational overhead, Prefilter LM and Prefilter SGD arrive at significantly better solutions. Not only is the resulting map more than four orders of magnitude more likely given the edge constraints (here, the full log joint probability is used for both methods), but the result is also much closer to the de-facto ground truth given by the marker-based registration. In addition, it aligns well with the aerial imagery. Note that the units are in millimeters, so an SSE_{xy} value of 102,628 for both prefilter SGD and prefilter LM is very small (around 30 cm mean square error per position). The SSE_{xy} achieved by max SGD of 3.58956×10^9 on the other hand is still very big, around 60 m.

The Particle method was also applied to this data set. Table 10 shows the performance given different particle counts over 100 trials. The resulting log probability stabilizes with more than 800 particles, and many more would be needed to achieve a better result. While the performance is not bad, much better in fact than the standard Max SGD/LM methods, a comparable result to Prefilter SGD/LM cannot be achieved. In addition, the required computation time is an order of magnitude larger.

4.3. Hannover Fair Hall

In a second experiment with real-world data, a set of 22 scans was recorded in Hall 22 at Hannover Fair exhibition grounds during evening hours with a nodding 2D Sick S300 laser scanner actuated with a cheap and rather inaccurate servo. The set was recorded during the RoboCup German Open 2009 competition, and many screens, booths, and competition fields are visible in the data. People were occasionally moving through the scans, introducing additional noise; due to the rather slow motion of the scanner, they appear as blurry blobs.A further challenging aspect of this data set is that rather large translations and rotations occurred between scans, and thus the overlap between scans is often very small. The average translation between two scans is around 5 m, with a maximum sensor range of 30 m. Note that odometry of the mobile robot base is in theory available, however, it is very imprecise as the robot uses tracked locomotion over a mixture of carpets and slippery floors.

The plane matching parameters used are shown in Table 11. As above, the prefilter step used $N = 200$ samples in the filtering process.

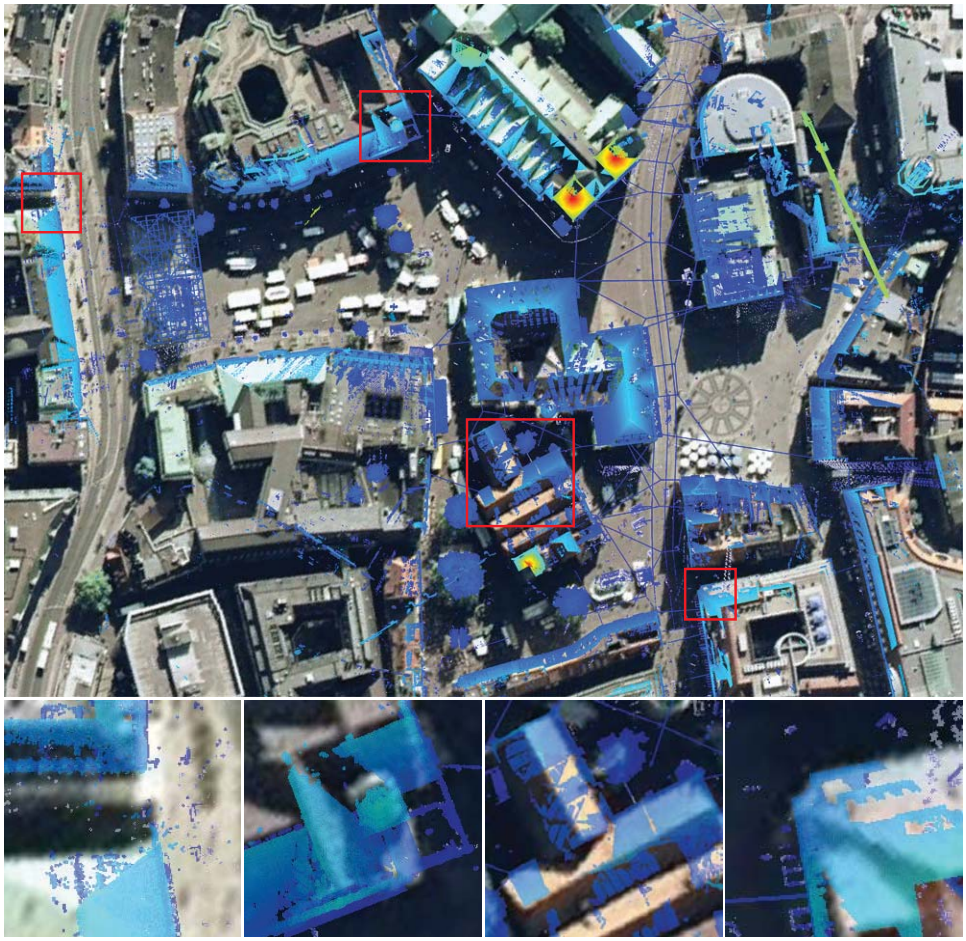

Fig. 10. Final map after optimization with Prefilter LM overlaid on aerial imagery of Bremen city center from Google Earth. Note that due to the height of the buildings, some parallax exists in the aerial image, and some ground features (fountains, small trees), as well as some high structures (cathedral tower) do not match exactly. The image shows the down-projected map at the general roof level. The height is used to assign grayscale values (coded with the Jet colormap in color).

Table 10. Runtimes in seconds and result quality for different numbers of particles of the Particle method on the Bremen City data set. Owing to the non-deterministic nature of the Particle method, 100 trials were run for each particle count and summarized by the mean and standard deviation. The data was recorded on a Core i7-2720GM, 2.2 GHz with 8 GB of RAM. The *SSE* metric (see Appendix B) was computed relative to the de-facto ground truth transformations given by the marker-based registration.

Number of Particles	Time	Log probability	SSE_{xyz}	$SSE_{\rho\theta\phi}$
25	0.001554	-1.28219×10^{8}	2.850298×10^{8}	0.024349
	$\sigma 2.05189 \times 10^{-4}$	$\sigma 6.140699 \times 10^{8}$	$\sigma 9.643072 \times 10^{8}$	$\sigma 0.157559$
50	0.002379	-1.43149×10^{6}	9.127352×10^{6}	5.061603×10^{-5}
	$\sigma 1.57122 \times 10^{-4}$	$\sigma 1.117032 \times 10^{7}$	$\sigma 7.872734 \times 10^{7}$	$\sigma 6.387661 \times 10^{-5}$
100	0.004469	-2.98506×10^{5}	5.3707584×10^{5}	4.427961×10^{-5}
	$\sigma 2.72404 \times 10^{-4}$	$\sigma 2.333723 \times 10^{4}$	$\sigma 1.314959 \times 10^{6}$	$\sigma 8.960579 \times 10^{-7}$
200	0.008602	-2.90968×10^{5}	2.8931014×10^{5}	4.442851×10^{-5}
	$\sigma 4.38217 \times 10^{-4}$	$\sigma 1.073785 \times 10^{4}$	$\sigma 5.761819 \times 10^{5}$	$\sigma 6.917717 \times 10^{-7}$
400	0.016455	-2.86378×10^{5}	2.2318785×10^{5}	4.439304×10^{-5}
	$\sigma 4.87811 \times 10^{-4}$	$\sigma 4.710047 \times 10^{3}$	$\sigma 5.652077 \times 10^{4}$	$\sigma 6.021706 \times 10^{-7}$
800	0.032582	-2.84246×10^{5}	2.2788351×10^{5}	4.436361×10^{-5}
	$\sigma 8.26252 \times 10^{-4}$	$\sigma 4.490705 \times 10^{3}$	$\sigma 5.825175 \times 10^{4}$	$\sigma 6.852595 \times 10^{-7}$

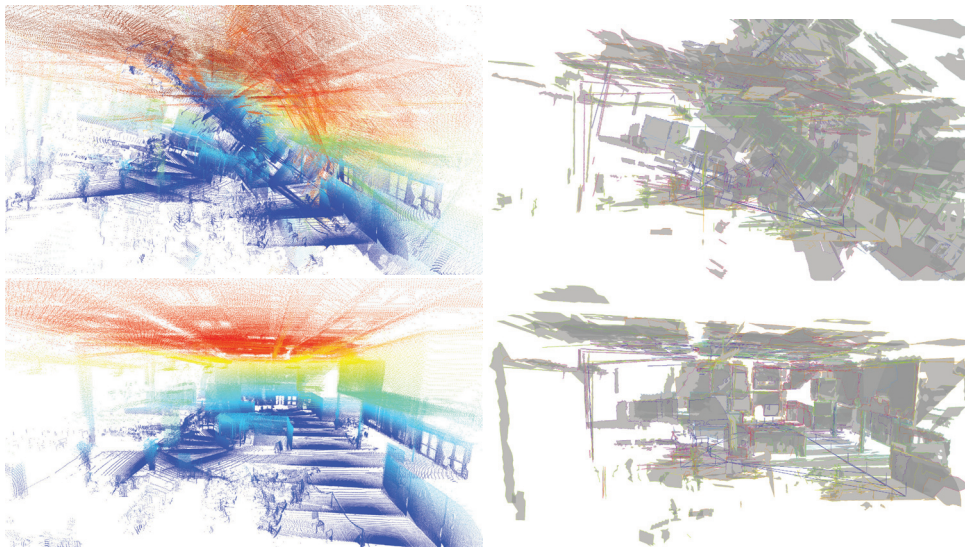

Fig. 11. Hannover Fair map. Top: Traditional Max SGD method using only the locally most likely registration result. Bottom: Result of the Prefilter LM method. The local z coordinate was used to assign grayscale values (coded with the Jet colormap in color). See also Extension 1 for an animated view of these maps.

Table 11. Plane matching parameters used for the Hannover Fair data set.

$\mathcal{O}_p$	0.1	λ_{max}	800,000
$\mathcal{L}_{unc}$	10^{22}	$\mathcal{L}_{rep}$	10^3
$\mathcal{O}_{min}$	0.667	$\mathcal{O}_{max}$	0.24
$\mathcal{D}_{min}$	2	$\mathcal{D}_{max}$	0.5
$\mathcal{R}_{min}$	0.065	$\mathcal{R}_{max}$	0.035

The final map consists of 23 vertices and 53 edges. Tables 14 and 15 show all 21 edges containing multimodal constraints. At most five modes were detected per edge, and in the worst case, the correct mode is actually the fifth one. The resulting MoG pose graph complexity is $C(G) = 30.4167$.

Note that in relation to the modes encountered in the Bremen City data set (Table 8), these modes are spatially closer to each other. This is due to a much more cluttered scene with many parallel planes (e.g. partitions, screens) and noise (e.g. by moving people and jitter in the servo control) which generate ambiguities that tend to have a negative impact on the registration method. Despite the relative spatial closeness, the wrong optima of the plane registration are so far away from the proper solution that the standard Max SGD and Max LM methods fail (see Figure 11).

Table 12 shows the achieved results of the Particle method with different numbers of particles. Since the graph was rather small, and the method is stochastic, 100 trials were run to produce these numbers. Naturally, the computation time increases linearly with the amount of particles used. Also, the log probability of the final MoG pose graph configuration increases when more particles are used. Around 1,600 particles are needed to find a good result reliably. At first glance, this may seem less than for the synthetic graphs discussed above, but note the large difference in graph size. Still, Prefilter SGD and Prefilter LM converge to a better result than the Particle method, and they do this at least one order of magnitude faster.

Table 13 shows the runtime and map quality comparisons between the traditional Max SGD/LM methods, the Multi-Edge LM method, and the Prefilter SGD/LM methods. The five methods do not differ significantly in their runtime, but produce very different results. The Prefilter SGD/LM methods arrive at a map that is almost four orders of magnitudes more likely given the log joint probability than the traditional methods.

Figure 11 show the two resulting maps. Quite obviously, the traditional Max SGD method failed, resulting in vastly misplaced observation poses and visibly inaccurate map parts. The same holds for the Max LM method which also produces obviously distorted maps. Both the Prefilter SGD and Prefilter LM methods, however, converge to nearly the same very good final result, even though the joint probability of the results is quite low. This may be explained by the many multimodal constraints that actually assign a low weight to the correct mode. Still, it seems that the information from other constraints reduces the likelihood of other optima sufficiently. This data set also illustrates that the modes do not have to be significantly far apart from each other for the Prefilter step to work.

5. Conclusion

In this article, a way to represent local ambiguities in pose graph SLAM has been introduced. Concretely, MoG are

Table 12. Runtimes in seconds and the final log probability achieved by the Particle method using different particle counts (labeled # above) on the Hannover Fair data set. Due to the non-determinism of the Particle method, 100 trials were run per count. The data was recorded on a Core i7-2720QM 2.2GHz with 8GB of RAM.

Number of particles	Time	Log probability
100	0.010071	-5.285083×10^{8}
	$\sigma 2.831905 \times 10^{-4}$	$\sigma 8.736545 \times 10^{8}$
200	0.019808	-1.923713×10^{8}
	$\sigma 3.68352 \times 10^{-4}$	$\sigma 5.632018 \times 10^{8}$
400	0.037288	-1.060568×10^{7}
	$\sigma 6.222948 \times 10^{-4}$	$\sigma 1.027818 \times 10^{7}$
800	0.072761	-5.349669×10^{6}
	$\sigma 1.710345 \times 10^{-3}$	$\sigma 4.192651 \times 10^{7}$
1600	0.147081	-3.769561×10^{6}
	$\sigma 6.048332 \times 10^{-3}$	$\sigma 1.698311 \times 10^{6}$
3200	0.295176	-3.353374×10^{6}
	$\sigma 1.216805 \times 10^{-2}$	$\sigma 1.632804 \times 10^{6}$
6400	0.579942	-3.269732×10^{6}
	$\sigma 2.648059 \times 10^{-2}$	$\sigma 6.91194 \times 10^{4}$
12800	1.213969	-3.244765×10^{6}
	$\sigma 7.467676 \times 10^{-2}$	$\sigma 4.488209 \times 10^{4}$

Table 13. Runtimes in seconds and results for the traditional Max SGD/LM, Multi-Edge LM, and Prefilter SGD/LM, recorded on a Core i7-2720QM, 2.2 GHz with 8 GB of RAM. Hannover Fair data set. Max SGD/LM were initialized with a breadth-first traversal of the graph, the rest of the methods were initialized with the poses computed by the Prefilter method described in Section 2.2.2.

Method	Initialization	Optimization	Final log probability
Max SGD	0.000457	0.014856	-2.54952×10^{9}
Max LM	0.000462	0.048428	-1.20496×10^{8}
Multi-edge LM	0.000312	0.059520	-2.33928×10^{10}
Prefilter SGD	0.000353	0.015385	-2.36122×10^{6}
Prefilter LM	0.000304	0.064116	-7.76502×10^{5}

used here for this purpose. This representation allows for much more complex constraint probability density functions, yet is easy to handle analytically and numerically. Such complex constraint functions occur, for example, if ambiguities in the environment give rise to multiple possible registration results. In addition, MoG can also be used to model nonlinear uncertainty models, e.g. from odometry, more accurately.

Furthermore, it is shown with experimental data that in the case of ambiguous registration results, state-of-the-art methods in the form of particle filters and graph based methods do not work well. In theory, particle filters can handle arbitrary probability distributions, including MoG, in the registration results. However, even with relatively few ambiguities and an extremely large number of particles, namely 10,000, the performance in terms of accuracy and consistency is very limited. This is shown with experiments on synthetic data where the degree of multimodality can easily be controlled. Existing graph-based methods cannot handle multimodal registrations without modifications. More precisely, they correspond to the case of using only the component of each mixture with the largest weight. When using this strategy for optimization on a MoG pose graph, the experimental results show that the performance in terms of accuracy is also very limited. Further experiments with real data sets show the same effect.

A naïve solution is to enumerate all component combinations to generate all possible variations of unimodal pose graphs based on a MoG pose graph. Each candidate graph can then be optimized with a standard method and the best optimization result among all candidates is used as the final result. This exhaustive search will most likely find a very good solution, as also confirmed in the experiments, but it also takes a very long time to complete as it involves a combinatorial explosion. In some examples presented above, optimization with this approach would take almost 10 years.

However, a promising method for MoG pose graph SLAM has also been introduced in this article. The Prefilter method was introduced, which converts the MoG pose graph into a unimodal one and subsequently uses a state-of-the-art pose graph optimization method, such as LM or SGD. The Prefilter method incrementally follows minimum spanning tree edges and assigns multiple global poses to each vertex to find a good global combination of modes. The edge cost used in the minimum spanning tree is the number of modes of its constraint, thus minimizing ambiguity during traversal. This greedy heuristic is shown to be very fast and robust in the presented experiments, especially compared with the standard methods.

Funding

The research leading to the presented results was supported in part by the European Community's Seventh Framework Programme (grant agreement number 231378, 'Cooperative Cognitive Control for Autonomous Underwater Vehicles (Co^3-AUVs)'; grant agreement number 270350, 'Cognitive Robot for Automation Logistics Processes (RobLog)'; and grant agreement number 288704, 'Marine robotic system of self-organizing, logically linked physical nodes (MORPH)').

Acknowledgments

We would like to thank Jan Elseberg, Dorit Borrmann, and Andreas Nüchter for providing the Bremen City Center data set, and Sören Schwertfeger as well as Jann Poppinga for providing the Hannover Fair data set.

Table 14. Multimodal edges in the Hannover Fair map, continued in Table 15. Again, the left column shows the edge containing the modes on the right. Up to five modes per edge were detected. The globally correct mode is highlighted.

Pair	#1	#2	#3	#4	#5
3 → 4	$T = (3.94, 0.02, 0.17)$ $R = (-0.29, 0.32, 3.38)$ $\det(C) = 1.83 \times 10^{-21}$	$T = (3.87, 0.00, 0.74)$ $R = (-0.17, 4.56, 3.59)$ $\det(C) = 5.32 \times 10^{-14}$	$T = (2.04, -0.04, 0.03)$ $R = (0.08, 1.48, -15.49)$ $\det(C) = 3.43 \times 10^{-13}$	–	–
8 → 9	$T = (3.23, -7.09, 7.11)$ $R = (-89.05, 20.77, -2.88)$ $\det(C) = 1.55 \times 10^{-24}$	$T = (4.17, 0.13, 0.23)$ $R = (-1.05, -0.88, 21.01)$ $\det(C) = 3.50 \times 10^{-23}$	$T = (4.34, 5.22, 2.11)$ $R = (83.89, -22.27, 90.23)$ $\det(C) = 9.93 \times 10^{-17}$	$T = (3.56, 0.08, 2.36)$ $R = (0.48, 3.18, 20.43)$ $\det(C) = 3.23 \times 10^{-16}$	–
18 → 19	$T = (2.62, 0.65, 0.04)$ $R = (-0.33, -0.02, 12.09)$ $\det(C) = 1.81 \times 10^{-20}$	$T = (1.45, 2.96, -0.05)$ $R = (-0.90, -0.30, 12.56)$ $\det(C) = 3.41 \times 10^{-15}$	$T = (0.75, 0.49, 0.01)$ $R = (-0.07, 0.01, 8.04)$ $\det(C) = 1.83 \times 10^{-14}$	–	–
19 → 20	$T = (1.32, 0.91, 0.03)$ $R = (-0.65, -0.36, 17.24)$ $\det(C) = 5.06 \times 10^{-17}$	$T = (3.33, 1.31, 0.19)$ $R = (-0.39, 0.72, 19.52)$ $\det(C) = 2.13 \times 10^{-15}$	–	–	–
22 → 23	$T = (4.30, -9.80, 0.22)$ $R = (-3.34, 0.48, 32.70)$ $\det(C) = 1.50 \times 10^{-14}$	$T = (2.25, -2.53, 0.06)$ $R = (1.07, 1.74, -44.70)$ $\det(C) = 2.54 \times 10^{-14}$	$T = (-0.38, 0.43, 0.09)$ $R = (1.56, -0.39, -27.46)$ $\det(C) = 1.53 \times 10^{-8}$	$T = (-0.46, 0.41, 0.09)$ $R = (1.77, 0.37, -42.38)$ $\det(C) = 2.60 \times 10^{-7}$	–
23 → 24	$T = (0.71, -0.36, -0.07)$ $R = (0.11, -0.02, -5.77)$ $\det(C) = 4.33 \times 10^{-19}$	$T = (2.89, -0.64, 0.09)$ $R = (0.16, -0.25, -5.50)$ $\det(C) = 9.49 \times 10^{-16}$	$T = (-4.96, -13.59, -0.51)$ $R = (0.51, -1.17, -9.87)$ $\det(C) = 7.30 \times 10^{-13}$	$T = (1.33, -0.32, 2.16)$ $R = (0.93, 15.27, -4.79)$ $\det(C) = 1.85 \times 10^{-12}$	–
24 → 25	$T = (4.20, 0.11, 0.23)$ $R = (-0.31, 0.17, -1.63)$ $\det(C) = 1.67 \times 10^{-11}$	$T = (-2.60, 0.70, -0.49)$ $R = (1.17, -4.77, -0.88)$ $\det(C) = 1.94 \times 10^{-11}$	–	–	–
2 → 4	$T = (0.35, -0.14, 0.04)$ $R = (-0.33, 0.06, -0.23)$ $\det(C) = 1.40 \times 10^{-19}$	$T = (5.65, 0.79, 2.30)$ $R = (-93.61, 3.16, 175.76)$ $\det(C) = 3.24 \times 10^{-16}$	$T = (7.03, -0.18, 0.33)$ $R = (-0.59, 0.14, -0.37)$ $\det(C) = 3.12 \times 10^{-14}$	–	–
5 → 7	$T = (8.61, -0.33, 0.43)$ $R = (0.74, -1.65, -3.50)$ $\det(C) = 3.58 \times 10^{-20}$	$T = (-6.01, 1.35, 2.14)$ $R = (91.82, -5.18, -2.01)$ $\det(C) = 3.29 \times 10^{-19}$	–	–	–
7 → 10	$T = (8.71, 0.32, 0.15)$ $R = (-1.26, 0.80, 22.18)$ $\det(C) = 1.38 \times 10^{-21}$	$T = (0.24, 0.52, 0.06)$ $R = (-2.45, -3.22, 21.72)$ $\det(C) = 8.54 \times 10^{-15}$	–	–	–
7 → 11	$T = (14.61, 7.57, 2.49)$ $R = (-92.61, -39.57, 179.09)$ $\det(C) = 3.25 \times 10^{-21}$	$T = (-3.70, -8.97, -0.64)$ $R = (-91.18, 46.23, -1.69)$ $\det(C) = 1.97 \times 10^{-20}$	$T = (10.39, 1.83, 0.30)$ $R = (-1.72, 0.94, 47.72)$ $\det(C) = 2.93 \times 10^{-20}$	$T = (0.69, 5.97, 0.28)$ $R = (-1.04, 0.28, -42.12)$ $\det(C) = 9.08 \times 10^{-17}$	–

Table 15. Multimodal edges in the Hannover Fair map, continued from Table 14. Again, the left column shows the edge containing the modes on the right. Up to five modes per edge were detected. The globally correct mode is highlighted.

Pair	#1	#2	#3	#4	#5
7 → 12	$T = (9.96, 3.36, 0.12)$ $R = (-2.49, 1.71, 102.66)$ $\det(C) = 2.65 \times 10^{-21}$	$T = (-5.07, 0.40, 3.58)$ $R = (99.93, 75.13, 101.31)$ $\det(C) = 5.12 \times 10^{-12}$	$T = (2.16, 6.37, 0.27)$ $R = (-3.61, 2.02, 13.55)$ $\det(C) = 4.58 \times 10^{-13}$	–	–
7 → 14	$T = (4.38, 5.74, 0.10)$ $R = (-1.55, 2.03, 166.28)$ $\det(C) = 7.35 \times 10^{-24}$	$T = (1.05, 1.97, -3.94)$ $R = (91.64, 10.44, 1.92)$ $\det(C) = 4.40 \times 10^{-18}$	-	-	-
8 → 10	$T = (0.75, 0.25, 0.03)$ $R = (-1.53, -1.18, 19.53)$ $\det(C) = 1.69 \times 10^{-24}$	$T = (3.92, 0.23, 0.26)$ $R = (-1.24, 0.07, 20.85)$ $\det(C) = 6.60 \times 10^{-22}$	$T = (3.97, 3.35, 5.72)$ $R = (-91.80, -66.75, -91.24)$ $\det(C) = 2.26 \times 10^{-14}$	–	–
8 → 12	$T = (2.24, -0.33, 10.86)$ $R = (175.90, -2.60, 165.37)$ $\det(C) = 2.67 \times 10^{-20}$	$T = (5.39, 1.43, 5.19)$ $R = (-81.72, -73.25, 169.04)$ $\det(C) = 6.64 \times 10^{-16}$	$T = (5.60, 3.07, 0.32)$ $R = (-3.65, 1.95, 102.15)$ $\det(C) = 2.10 \times 10^{-16}$	$T = (2.58, 10.60, 0.47)$ $R = (-0.69, 4.28, -168.06)$ $\det(C) = 4.73 \times 10^{-16}$	–
8 → 13	$T = (3.71, 4.71, 0.25)$ $R = (-2.28, 1.79, 134.30)$ $\det(C) = 1.99 \times 10^{-23}$	$T = (0.23, 1.48, 12.04)$ $R = (-93.50, 46.82, 173.12)$ $\det(C) = 2.54 \times 10^{-19}$	$T = (1.23, 8.75, 0.30)$ $R = (1.40, 3.87, -136.78)$ $\det(C) = 1.74 \times 10^{-16}$	–	–
9 → 11	$T = (2.13, 0.84, 0.11)$ $R = (-0.24, -0.61, 26.36)$ $\det(C) = 9.19 \times 10^{-24}$	$T = (0.64, 1.62, 2.13)$ $R = (-92.31, -39.37, -112.83)$ $\det(C) = 7.96 \times 10^{-21}$	$T = (-2.76, 2.65, 0.27)$ $R = (-0.84, -1.62, 26.93)$ $\det(C) = 1.40 \times 10^{-16}$	$T = (3.69, 4.56, 0.29)$ $R = (-0.01, -0.35, 25.79)$ $\det(C) = 4.42 \times 10^{-15}$	–
10 → 12	$T = (2.30, 2.30, 0.13)$ $R = (-3.48, 0.88, 80.71)$ $\det(C) = 5.22 \times 10^{-25}$	$T = (0.99, 2.51, -2.96)$ $R = (89.03, -17.37, 71.29)$ $\det(C) = 4.24 \times 10^{-16}$	–	–	–
10 → 13	$T = (1.68, 4.31, -0.95)$ $R = (87.78, -48.40, 71.85)$ $\det(C) = 6.54 \times 10^{-26}$	$T = (4.87, 6.26, 3.10)$ $R = (92.29, 40.49, 70.00)$ $\det(C) = 5.90 \times 10^{-25}$	$T = (1.25, 4.42, 0.13)$ $R = (-3.20, 2.38, 112.68)$ $\det(C) = 5.41 \times 10^{-21}$	$T = (1.23, 4.37, 2.29)$ $R = (-3.15, 2.02, 112.82)$ $\det(C) = 1.93 \times 10^{-20}$	–
19 → 21	$T = (3.20, 0.01, 0.13)$ $R = (-2.07, -0.69, 60.96)$ $\det(C) = 6.39 \times 10^{-19}$	$T = (3.73, -2.25, 0.23)$ $R = (-1.46, -1.29, 60.83)$ $\det(C) = 1.69 \times 10^{-17}$	$T = (5.52, 3.09, 0.26)$ $R = (-2.33, 0.20, 60.02)$ $\det(C) = 2.25 \times 10^{-17}$	–	–
19 → 23	$T = (8.63, 1.56, -1.16)$ $R = (-2.39, 1.73, -150.53)$ $\det(C) = 1.86 \times 10^{-17}$	$T = (7.57, -0.97, 0.11)$ $R = (0.30, 1.24, -72.06)$ $\det(C) = 1.48 \times 10^{-15}$	$T = (6.99, 1.47, 0.16)$ $R = (1.08, 1.41, -72.35)$ $\det(C) = 8.66 \times 10^{-15}$	$T = (8.58, 1.57, 0.23)$ $R = (1.69, 5.52, -150.08)$ $\det(C) = 3.59 \times 10^{-12}$	$T = (0.49, 0.12, 0.07)$ $R = (-0.09, 1.75, -34.80)$ $\det(C) = 9.17 \times 10^{-9}$

References

Anderson BDO and Moore JB (1979) *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall.

Barkby S, Williams S, Pizarro O and Jakuba M (2009) An efficient approach to bathymetric SLAM. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009 (IROS 2009)*, pp. 219–224.

Besl PJ and McKay ND (1992) A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14: 239–256.

Birk A (2010) A quantitative assessment of structural errors in grid maps. *Autonomous Robots* 28: 187–196.

Borrmann D, Elseberg J, Lingemann K, Nüchter A and Hertzberg J (2008) Globally consistent 3D mapping with scan matching. *Robotics and Autonomous Systems* 56: 130–142.

Burgard W, Stachniss C, Grisetti G, et al. (2009) A comparison of slam algorithms based on a graph of relations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009 (IROS 2009)*, pp. 2089–2095.

Chang H, Lee C, Lu Y-H and Hu Y (2006) Simultaneous localization and mapping with environmental structure prediction. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006 (ICRA 2006)*, pp. 4069–4074.

Chang H, Lee C, Lu Y-H and Hu Y (2007) P-SLAM: Simultaneous localization and mapping with environmental-structure prediction. *IEEE Transactions on Robotics* 23: 281–293.

Cormen R, Leiserson C, Rivest R and Stein C (2001) *Introduction to algorithms*. Cambridge, MA: The MIT Press.

Craig JJ (2005) *Introduction to Robotics — Mechanics and Control*. Englewood Cliffs, NJ: Prentice-Hall.

Dellaert F (2005) Square root SAM In *Proceedings of Robotics: Science and Systems*, Cambridge, MA.

Doucet A, Freitas ND, Murphy KP and Russell SJ (2000) *Rao–Blackwellised Particle Filtering for Dynamic Bayesian Networks*. San Mateo, CA: Morgan Kaufmann, pp. 176–183.

Elinas P, Sim R and Little J (2006) /SPL SIGMA/SLAM: stereo vision SLAM using the Rao–Blackwellised particle filter and a novel mixture proposal distribution. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006 (ICRA 2006)*, pp. 1564–1570.

Fairfield N, Kantor G and Wettergreen D (2006) Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment. In *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3575–3580.

Fairfield N, Kantor GA and Wettergreen D (2007) Real-time SLAM with octree evidence grids for exploration in underwater tunnels. *Journal of Field Robotics* 24(1–2): 3–21.

Feller W (1945) The fundamental limit theorems in probability. *Bulletin of the American Mathematical Society* 51: 800–832.

Fischler MA and Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Graphics and Image Processing* 24: 381–395.

Frese U (2006) A discussion of simultaneous localization and mapping. *Autonomous Robots* 20: 25–42.

Frese U, Larsson P and Duckett T (2005) A multilevel relaxation algorithm for simultaneous localization and mapping. *IEEE Transactions on Robotics* 21: 196–207.

Golfarelli M, Maio D and Rizzi S (2001) Correction of dead-reckoning errors in map building for mobile robots. *IEEE Transactions on Robotics and Automation* 17: 37–47.

Grisetti G, Grzonka S, Stachniss C, Pfaff P and Burgard W (2007a). Efficient estimation of accurate maximum likelihood maps in 3D. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007 (IROS 2007)*, pp. 3472–3478.

Grisetti G, Kümmerle R, Stachniss C, Frese U and Hertzberg C (2010). Hierarchical optimization on manifolds for online 2D and 3D mapping. In *IEEE International Conference on Robotics and Automation, 2010 (ICRA '10)*, pp. 273–278.

Grisetti G, Rizzini D, Stachniss C, Olson E and Burgard W (2008) Online constraint network optimization for efficient maximum likelihood map learning. In *IEEE International Conference on Robotics and Automation, 2008 (ICRA 2008)*, pp. 1880–1885.

Grisetti G, Stachniss C and Burgard W (2005) Improving grid-based SLAM with Rao–Blackwellized particle filters by adaptive proposals and selective resampling. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Grisetti G, Stachniss C and Burgard W (2007b) Improved techniques for grid mapping with Rao–Blackwellized particle filters. *IEEE Transactions on Robotics* 23: 34–46.

Grisetti G, Stachniss C, Grzonka S and Burgard W (2007c). A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA.

Hahnel D, Burgard W, Fox D and Thrun S (2003) An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003 (IROS 2003)*, vol. 1, pp. 206–211.

Hertzberg C (2008) *A Framework for Sparse, Non-Linear Least Squares Problems on Manifolds*. Master's thesis, University of Bremen.

Huber PJ (1973) Robust regression: Asymptotics, conjectures and Monte Carlo. *The Annals of Statistics* 1: 799–821.

Koenig A, Kessler J and Gross H-M (2008) A graph matching technique for an appearance-based, visual SLAM-approach using Rao–Blackwellized particle filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008 (IROS 2008)*, pp. 1576–1581.

Konolige K, Grisetti G, Kümmerle R, Burgard W, Limketkai B and Vincent R (2010) Efficient sparse pose adjustment for 2D mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010 (IROS 2010)*.

Kuemmerle R, Steder B, Dornhege C, et al. (2009) On measuring the accuracy of SLAM algorithms. *Autonomous Robots* 27: 387–407.

Kümmerle R, Grisetti G, Strasdat H, Konolige K and Burgard W (2011) G2o: A general framework for graph optimization. In *IEEE International Conference on Robotics and Automation (ICRA), 2011*, pp. 3607–3613.

Lu F and Milios E (1997) Globally consistent range scan alignment for environment mapping. *Autonomous Robots* 4: 333–349.

Marks TK, Howard A, Bajracharya M, Cottrell GW and Matthies LH (2009) Gamma-SLAM: Visual SLAM in unstructured environments using variance grid maps. *Journal of Field Robotics* 26: 26–51.

Montemerlo M and Thrun S (2003) Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proceedings IEEE International Conference on Robotics and Automation, 2003 (ICRA '03)*, vol. 2, pp. 1985–1991.

Murphy K (1999) Bayesian map learning in dynamic environments. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 1015–1021.

Olson E (2008) *Robust and Efficient Robotic Mapping*. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Olson E (2009) Recognizing places using spectrally clustered local matches. In *Robotics and Autonomous Systems*.

Olson E, Leonard J and Teller S (2006) Fast iterative alignment of pose graphs with poor initial estimates. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006 (ICRA 2006)*, pp. 2262–2269.

Olson E, Leonard J and Teller S (2007) Spatially-adaptive learning rates for online incremental SLAM. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA.

Olson E, Walter M, Leonard J and Teller S (2005) Single cluster graph partitioning for robotics applications. In *Proceedings of Robotics Science and Systems*, pp. 265–272.

Pathak K, Birk A and Vaskevicius N (2010a) Plane-based registration of sonar data for underwater 3D mapping. In *International Conference on Intelligent Robots and Systems (IROS)*, pp. 4880–4885.

Pathak K, Birk A, Vaskevicius N, Pfingsthorn M, Schwertfeger S and Poppinga J (2010b) Online 3D SLAM by registration of large planar surface segments and closed form pose-graph relaxation. *Journal of Field Robotics* 27: 52–84.

Pathak K, Birk A, Vaskevicius N and Poppinga J (2010c) Fast registration based on noisy planes with unknown correspondences for 3D mapping. *IEEE Transactions on Robotics* 26(2): 1–18.

Pathak K, Borrmann D, Elseberg J, Vaskevicius N, Birk A and Nuchter A (2010d) Evaluation of the robustness of planar-patches based 3D-registration using marker-based ground-truth in an outdoor urban scenario. In *International Conference on Intelligent Robots and Systems (IROS)*, pp. 5725–5730.

Petersen KB and Pedersen MS (2008) The matrix cookbook, Version 20081110. http://www2.imm.dtu.dk/pubdb/p.php?3274.

Pfingsthorn M, Birk A, Schwertfeger S, Bülow H and Pathak K (2010) Maximum likelihood mapping with spectral image registration. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation, 2010 (ICRA 2010)*.

Rousseeuw PJ and Leroy AM (2005) *Robust Regression and Outlier Detection*. New York: John Wiley & Sons, Inc.

Schroeter C and Gross H-M (2008) A sensor-independent approach to RBPF SLAM - map match SLAM applied to visual mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008 (IROS 2008)*, pp. 2078–2083.

Smith R, Self M and Cheeseman P (1990) Estimating uncertain spatial relationships in robotics. In Cox IJ and Wilfon GT (eds), *Autonomous Robot Vehicles*. New York: Springer-Verlag, pp. 167–193.

Stachniss C, Grisetti G and Burgard W (2005) Recovering particle diversity in a Rao–Blackwellized particle filter for SLAM after actively closing loops. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005 (ICRA 2005)*, pp. 655–660.

Stachniss C, Grisetti G, Burgard W and Roy N (2007) Analyzing Gaussian proposal distributions for mapping with Rao–Blackwellized particle filters. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007 (IROS 2007)*, pp. 3485–3490.

Sünderhauf N and Protzel P (2012) Towards a robust back-end for pose graph SLAM. In *IEEE International Conference on Robotics and Automation, 2012 (ICRA '12)*, St Paul, MN.

Thrun S, Burgard W and Fox D (2005) *Probabilistic Robotics*. Cambridge, MA: The MIT Press.

Titterington D, Smith A and Makov U (1985) *Statistical Analysis of Finite Mixture Distributions*. New York: John Wiley & Sons, Inc.

Tomono M (2007) Monocular slam using a Rao–Blackwellised particle filter with exhaustive pose space search. In *2007 IEEE International Conference on Robotics and Automation*, pp. 2421–2426.

Welle J, Schulz D, Bachran T and Cremers A (2010) Optimization techniques for laser-based 3D particle filter SLAM. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3525–3530.

Appendix A: The algorithm to generate the synthetic MoG pose graphs

The random MoG pose graphs used in Section 3.1 for a systematic evaluation of the different methods are generated with algorithm 4. In the experiments, multimodal registration results are assumed to be an exception, i.e. most edges contain a standard Gaussian and only a few contain a mixture. In addition, the number of modes is assumed to be small. Note that the exact number of multimodal edges with their exact number of modes is provided as input to the algorithm, thus the amount of multimodality can be exactly controlled.

Algorithm 4 simply ensures that there is a proper mode per edge, i.e. that there is one in principle correct (but noisy) registration, by checking that the related spatial transformation does not pass through a wall such that there is a line of sight between the two related vertices. The Gaussian noise for the proper first constraint per edge is initialized with covariance $\Sigma = diag(1 + 0.05|x|, 1 + 0.05|y|, 0.01 + 0.01|\theta|)$. So, in Algorithm 4 $\sigma_{xy} = 0.05$ and $\sigma_\theta = 0.01$, as well as $\delta_{xy} = 1.0$ and $\delta_\theta = 0.01$. Here, x, y, and θ denote the relative ground truth pose of the target vertex relative to the base vertex of the constraint. If an additional constraint is added, a random weight between 0.01 and 1 is generated; this is followed by a renormalization of all weights.

Furthermore, the algorithm ensures that the edges have a reasonable length, i.e. that no registrations are assumed between far away places. All 110 synthetic graphs are sampled from the environment shown in Figure 2. This environment spans an area of 1,300 by 900 units. Each edge is approximately 100 units long on average. Each graph consists of 128 vertices and 256 edges. Any edge is between 75 and 230 units long by design.

Algorithm 4: Algorithm to generate the multimodal graphs.

Input: Minimum and maximum vertex distances d_- and d_+
Input: Number of vertices and edges N_V and N_E
Input: For each number of components $m = 2 \dots M$, the number of desired edges containing this number of components N_m
Input: Translation variance factor σ_{xy} and rotation variance factor σ_θ
Input: Translation variance offset δ_{xy} and rotation variance offset δ_θ
Output: MoG Pose Graph G

$e_x = (1, 0, 0)$
$e_y = (0, 1, 0)$
$e_\theta = (0, 0, 1)$
while $|V| < N_V$ **do**
  Sample a pose in free space x with random orientation
  if $\exists v_i \in V$: *the distance from x to x_i is between d_- and d_+ and the line between them does not intersect an obstacle* **then**
    $t_{gt} = x \ominus x_i$
    $\Sigma = diag(\delta_{xy} + \sigma_{xy}||t_{gt}^{\mathrm{T}} e_x||, \delta_{xy} + \sigma_{xy}||t_{gt}^{\mathrm{T}} e_y||, \delta_\theta + \sigma_\theta||t_{gt}^{\mathrm{T}} e_\theta||)$
    $\mu = t_{gt} \oplus \epsilon\, \mathcal{N}(\mathbf{0}, \Sigma)$
    $v_{new} = (x,)$
    $V = V \cup v_{new}$
    $E = E \cup (v_i, v_{new}, (\{1\}, \{\mu\}, \{\Sigma\}))$
  end
end
while $|E| < N_E$ **do**
  Select random vertex $v_i \in V$
  for *all vertices v_j not connected to v_i* **do**
    if *the distance from x_i to x_j is between d_- and d_+ or there is no line of sight between the two* **then**
      continue
    end
    $t_{gt} = x_j \ominus x_i$
    $\Sigma = diag(\delta_{xy} + \sigma_{xy}||t_{gt}^{\mathrm{T}} e_x||, \delta_{xy} + \sigma_{xy}||t_{gt}^{\mathrm{T}} e_y||, \delta_\theta + \sigma_\theta||t_{gt}^{\mathrm{T}} e_\theta||)$
    $\mu = t_{gt} \oplus \epsilon\, \mathcal{N}(\mathbf{0}, \Sigma)$
    $E = E \cup (v_i, v_j, (\{1\}, \{\mu\}, \{\Sigma\}))$
  end
end
for $\forall m = 2 \dots M$ **do**
  for $\forall n = 1 \dots N_m$ **do**
    Select random edge $e_i \in E$ which has only one component
    $(v_a, v_b, c) = e_i$
    $(\pi, \mu, \Sigma) = c$
    while $|c| < m$ **do**
      Sample a pose in free space x with random orientation
      if *distance from x_a to x is not between d_- and d_+ or there is no line of sight between them* **then**
        continue
      end
      $\mu = x \ominus x_a$
      $\Sigma = diag(\sigma_{xy}||\mu^{\mathrm{T}} e_x||, \sigma_{xy}||\mu^{\mathrm{T}} e_y||, \sigma_\theta||\mu^{\mathrm{T}} e_\theta||)$
      $c = c$
    end
  end
end

Note that all 'ground truth' modes, i.e. those which represent an in principle correct but just noisy spatial transformation, are generated in the same way as all other components in the mixtures. This reflects the idea of local ambiguity: any of the components in a mixture are potential registration results.

Appendix B: A quality metric for experimental map evaluation

Olson (2008) describes a map error metric *SSE* where the mean squared error between ground truth and the estimated vertex poses is computed, once for translation (SSE_{xy}) and once for rotation (SSE_θ).

$$t_i^{gt} = x_i \ominus gt_i \tag{25}$$

$$SSE_{xy} = |V|^{-1} \sum_{v_i \in V} (e_1^{\mathrm{T}} t_i^{gt})^2 + (e_2^{\mathrm{T}} t_i^{gt})^2 \tag{26}$$

$$SSE_\theta = |V|^{-1} \sum_{v_i \in V} (e_3^{\mathrm{T}} t_i^{gt})^2 \tag{27}$$

where gt_i is the ground truth pose of vertex i, x_i is its estimate, and $e_n \in \mathbb{R}^3$ is a canonical unit vector in the nth dimension (i.e. $e_1 = (1, 0, 0)^{\mathrm{T}}$). This metric strongly emphasizes the absolute difference to ground truth poses and thus punishes global errors severely. Other error metrics, such as those introduced by Burgard et al. (2009), rather focus on local errors between two vertices in the graph and are more lenient. Any method that does well in the *SSE* metric can be expected to also receive high scores in the other less stringent metrics. Hence, *SSE* is used here.

Appendix C: Fusing mixtures of Gaussians for odometry

In order to study the effects of odometry in Section 3.1, a method to fuse a MoG (the possibly ambiguous registration result), with a single Gaussian estimate (the odometry estimate) is needed. This is done using the measurement update equations of the Gaussian sum filter (Anderson and Moore, 1979, p. 214). As usual, the mean and covariance of a single component m are updated as follows:

$$\mu_m = \bar{\mu}_m + K_m(\mu_{odo} - \bar{\mu}_m), \tag{28}$$

$$\Sigma_m = \bar{\Sigma}_m - K_m \bar{\Sigma}_m, \tag{29}$$

$$K_m = \bar{\Sigma}_m(\bar{\Sigma}_m + \Sigma_{odo})^{-1}. \tag{30}$$

This is done for each component. Finally, the fused weights are

$$\pi_m \propto \bar{\pi}_m p(\bar{\mu}_m | \mu_{odo}, \bar{\Sigma}_m + \Sigma_{odo}), \tag{31}$$

where $\bar{\mu}_m$, $\bar{\Sigma}_m$, and $\bar{\pi}_m$ are the component parameters before fusion.

The weight update has the result that components that are far away from the odometry estimate are weighted significantly less. In the results described in Section 3.2, a component is dropped from the fused mixture if its weight drops below 0.005 after normalization. After one or more components have been dropped from the mixture, the remaining weights are normalized again. The approximation error can be made arbitrarily small by selecting a lower threshold.

Appendix D: Index to multimedia extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	3D animation of the distribution in Figure 1 and the final maps generated by max SGD and prefilter LM of the real-world data sets as discussed in Section 4.2 and 4.3. See also Figures 9 and 11.