

# Speeding Up Rao-Blackwellized SLAM

Giorgio Grisetti<sup>\*†</sup> Gian Diego Tipaldi<sup>†</sup> Cyrill Stachniss<sup>\*</sup> Wolfram Burgard<sup>\*</sup> Daniele Nardi<sup>†</sup>

<sup>\*</sup>University of Freiburg, Department of Computer Science, D-79110 Freiburg, Germany

<sup>†</sup>Dipartimento Informatica e Sistemistica, Università “La Sapienza”, I-00198 Rome, Italy

**Abstract**—Recently, Rao-Blackwellized particle filters have become a popular tool to solve the simultaneous localization and mapping problem. This technique applies a particle filter in which each particle carries an individual map of the environment. Accordingly, a key issue is to reduce the number of particles and/or to make use of compact map representations. This paper presents an approximative but highly efficient approach to mapping with Rao-Blackwellized particle filters. Moreover, it provides a compact map model. A key advantage is that the individual particles can share large parts of the model of the environment. Furthermore, they are able to re-use an already computed proposal distribution. Both techniques substantially speed up the overall process and reduce the memory requirements. Experimental results obtained with mobile robots in large-scale indoor environments and based on published, standard datasets illustrate the advantages of our methods over previous Rao-Blackwellized mapping approaches.

## I. I

Learning maps is a fundamental task of mobile robots and a lot of researchers focused on this problem. In the literature, the mobile robot mapping problem is often referred to as the *simultaneous localization and mapping (SLAM)* problem [3, 7, 8, 9, 13, 14, 15, 20]. In general, SLAM is a complex problem because for learning a map the robot requires a good pose estimate while at the same time a consistent map is needed to localize a robot. This dependency between the pose and the map estimate makes the SLAM problem hard and requires to search for a solution in a high-dimensional space.

Murphy, Doucet, and colleagues [15, 2] introduced Rao-Blackwellized particle filters (RBPFs) as an effective means to solve the SLAM problem. The main problem of the Rao-Blackwellized approaches is their complexity, measured in terms of the number of particles required to learn an accurate map. Reducing this quantity is one of the major challenges for this family of algorithms.

The contribution of this paper is a technique that reduces the computational and the memory requirements in the context of Rao-Blackwellized mapping. In this way, it becomes feasible to maintain a comparably large set of particles online. This is achieved by enabling a subset of samples to share large parts of the map and to use the same proposal distribution. Our system allows a standard laptop computer to perform all computations necessary to learn accurate maps with more than one thousand samples online.

This paper is organized as follows. After the discussion of related work, we briefly introduce Rao-Blackwellized mapping. We then describe our technique for efficiently drawing particles from a proposal distribution. After this, we present

our map representation. Finally, we show experiments illustrating the improvements of our approach to Rao-Blackwellized mapping.

## II. R W

Solutions to the SLAM problem can be classified according to their underlying estimation technique. The most popular approaches are Extended Kalman filters (EKF), maximum likelihood techniques, sparse extended information filters (SEIFs), and Rao Blackwellized particle filters (RBPFs). The effectiveness of the EKF comes from the fact that it estimates the fully correlated posterior over landmark positions and robot poses [10, 17]. Its weakness lies in the strong assumptions regarding the motion model and the sensor noise. Moreover, the landmarks are assumed to be uniquely identifiable. There exist techniques [16] to deal with unknown data association in the SLAM context. However, if certain assumptions are violated the filter is likely to diverge [6].

An alternative approach is to use a maximum likelihood algorithm that computes a map by constructing a network of relations. The relations represent the spatial constraints between the poses of the robot [8, 12].

Thrun *et al.* [20] proposed a SEIF method which uses the inverse of the covariance matrix. In this way, measurements can be integrated efficiently. Eustice *et al.* [5] presented an improved technique to accurately compute the error-bounds within the SEIF framework and thus reduces the risk of becoming overly confident.

In [15, 2], Rao-Blackwellized particle filters have been introduced as an effective means to solve the SLAM problem. Each particle in a RBPF represents a potential trajectory of the robot and a map of the environment. The framework has been subsequently extended by Montemerlo *et al.* [13, 14] for approaching the SLAM problem with landmarks. To learn accurate grid maps, RBPFs have been used by Eliazar and Parr [3] and Hähnel *et al.* [9]. Whereas the first work describes an efficient map representation, the second one presents an improved motion model that reduces the number of required particles. A combination of the approach of Hähnel *et al.* and Montemerlo *et al.* as been presented by Grisetti *et al.* [7], which extends the ideas of FastSLAM-2 to the grid map case. We present in this paper an approximative solution to Rao-Blackwellized mapping which describes how to draw particles and how to represent maps so that the system can be executed significantly faster and needs less memory resources.

Lisien *et al.* [11] realized a hierarchical map model in the context of SLAM and reported that this improves loop-closing. Bosse *et al.* [1] describe a generic framework for

SLAM in large-scale environments. They use a graph structure of local maps with relative coordinate frames similar to the work described in [4]. This approach is able to reduce the complexity and at the same time it can better deal with linearization problems in the context of EKF-SLAM. Our approach is related to this framework since we also use local maps attached to a graph structure to model the environment. However, our motivation to use such a map representation is to allow multiple particles to share a map.

The contribution of the paper is a computational and memory efficient Rao-Blackwellized particle filter for SLAM. Our approach allows the robot to efficiently determine the proposal distributions to sample the next generation of particles in an approximative manner. Additionally, we present a compact map model in which multiple particles share a map. This enables us to maintain substantially more samples with less memory and computational requirements compared to state-of-the-art Rao-Blackwellized mapping approaches.

### III. R -B M

RBPFs for SLAM are used to estimate the posterior  $p(x_{1:t}, m \mid z_{1:t}, u_{1:t-1})$  about the trajectory  $x_{1:t}$  of the robot and the map  $m$  of the environment given the observations  $z_{1:t}$  and odometry measurements  $u_{1:t-1}$ . Its key idea is to separate the estimation of the trajectory of the robot from the map estimation process

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

This can be done efficiently, since the posterior over maps  $p(m \mid x_{1:t}, z_{1:t})$  can be computed analytically given the knowledge of  $x_{1:t}$  and  $z_{1:t}$ . Computing the posterior  $p(x_{1:t} \mid z_{1:t}, u_{1:t-1})$  is similar to the localization problem, since only the trajectory of the vehicle needs to be estimated. This is done using a particle filter which incrementally processes the observations and the odometry readings. The overall process can be summarized by the following four steps:

- 1) *Sampling*: The next generation of particles is obtained from the current generation by sampling from a so-called proposal distribution.
- 2) *Importance Weighting*: An individual importance weight is assigned to each particle according to the most recent observation, the pose estimate, and the map associated with this particle.
- 3) *Resampling*: Particles with a low importance weight are typically replaced by samples with a high weight. This step is necessary since only a finite number of particles is used to approximate a continuous distribution.
- 4) *Map Estimation*: The map of each particle is updated based on pose represented by that particle.

Several authors proposed optimizations to Rao-Blackwellized mapping. They either presented compact map representations [3] to deal with large particle sets or accurate proposal distributions [7, 9, 13] in order to keep the number of samples small.

In this section, we present our approach to Rao-Blackwellized mapping which is able to handle large particle sets while reducing the memory and computational requirements. Our implementation is based on the open-source implementation [18] of the mapping system of Grisetti *et al.* [7]. The mayor drawback of this approach lies in its complexity. It runs online only for small particle sets. This is due to an informed but expensive to compute proposal distribution which is determined for each particle individually. Furthermore, each particle maintains a full grid map.

In the context of Rao-Blackwellized particle filters for SLAM, the proposal is used to model the relative movement of the vehicle under uncertainty. In most situations, this uncertainty is similar for all samples within one movement. It therefore makes sense to use the same uncertainty to propagate the particles as long as they appear to represent similar state hypotheses. In this section, we derive a way to sample multiple particles from the same proposal. As a result, the time consuming computation of the proposal distribution can be carried out for a few particles that are representatives for groups of similar samples.

Furthermore, local maps which are represented in a robot-centered coordinate frame look similar for many particles. We therefore present a compact map model in which multiple particles can share a map. Instead of storing an individual map, each sample maintains only a set of reference frames for the different local maps. This substantially reduces the memory requirements of the mapping algorithm.

#### A. Different Situations During Mapping

Before we derive our new proposal distributions, we start with a brief analysis of the behavior of a RBPF. One can distinguish three different types of situations during mapping:

- The robot is moving through *unknown* areas,
- is moving through *well-known* areas, or
- is *closing a loop*.

In each of those situations, the filter behaves differently. Whenever the robot is moving through unknown terrain, the trajectory uncertainty grows. This is due to the fact that the errors are accumulated along the trajectory. The resulting uncertainty can only be bounded by observations which cover a (partially) known region.

In the second case, a map of the surroundings of the robot is known and in this way the SLAM problem turns into a localization problem which is typically easier to handle. Whenever the robot is closing a loop, the particle cloud is often widely spread. By reentering known areas, the filter can typically determine which particles are consistent with their own map and which are not. Such a situation leads to an unbalanced distribution of particle weights. The next resampling action then eliminates a series of unlikely hypotheses.

For each of these three situations, we will present a proposal distribution that needs to be computed only for a small set of representatives rather than for all particles. For the beginning, let us assume that

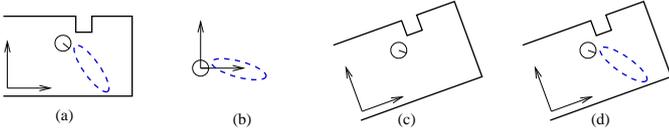


Fig. 1. Image (a) depicts the pose hypothesis of a particle, its local map, and the computed proposal which is represented by the blue/dashed ellipse. Image (b) illustrates the proposal distribution represented in the ego-centric reference frame of that sample. Image (c) shows a second particle and its map. By carrying out a coordinate transform, the proposal of the first particle can be used by the second particle as long as their maps are locally similar (d).

- 1) the current situation is known, which means that the robot can determine whether it is moving through unknown terrain, within a known area, or is closing a loop,
- 2) the corresponding local maps of two samples are similar if considered in a particle-centered reference frame. In the following, we refer to this property as *local similarity* of the maps,
- 3) an accurate algorithm for pose tracking is used and the observations are affected by a limited sensor noise.

### B. Computing the Proposal for Unknown Terrain

When moving through unknown areas, most parts of the map are irrelevant for computing the proposal distribution. Only a local map around the current pose is therefore taken into account. This map, called  $\tilde{m}_{t-1}^{(i)}$ , refers to the local map of particle  $i$  with respect to the pose  $x_{t-1}^{(i)}$  of that particle at time step  $t-1$ . In the surroundings of the robot, we can approximate

$$p(x_t | m_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}) \approx p(x_t | \tilde{m}_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}). \quad (2)$$

Under Assumption 2, which requires that the maps of particle  $i$  and  $j$  are locally similar, we can write

$$\tilde{m}_{t-1}^{(i)} \ominus x_{t-1}^{(i)} \approx \tilde{m}_{t-1}^{(j)} \ominus x_{t-1}^{(j)}. \quad (3)$$

Here  $\oplus$  and  $\ominus$  are the standard pose compounding operators (see [12]). E.g.,  $a \ominus b$  is an operator that translates all the points in the domain of the function  $a$  so that the new origin of the domain of  $a$  is  $b$  and  $\oplus$  is its inverse.

We observed that the proposal distributions for different particles are similar if transformed to an ego-centric reference frame

$$p(x_t | \tilde{m}_{t-1}^{(j)}, x_{t-1}^{(j)}, z_t, u_{t-1}) \ominus x_{t-1}^{(j)} \approx p(x_t | \tilde{m}_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}) \ominus x_{t-1}^{(i)}. \quad (4)$$

As a result, we can determine the proposal for a particle  $j$  by computing the proposal in the reference frame of particle  $i$  and translating it to the reference frame of particle  $j$

$$p(x_t | \tilde{m}_{t-1}^{(j)}, x_{t-1}^{(j)}, z_t, u_{t-1}) \approx x_{t-1}^{(j)} \oplus (p(x_t | \tilde{m}_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}) \ominus x_{t-1}^{(i)}). \quad (5)$$

This computation is illustrated in Figure 1. Eq. (5) shows how to transform a proposal between particles while the robot moves through unknown terrain. The complex proposal computation needs to be performed only once and can then be translated to the reference frame of the other particles.

### C. Computing the Proposal for Already Visited Areas

Whenever the robot moves through known areas, each particle stays localized in its own map according to Assumption 3. To update the new pose of each particle while the robot moves, we maximize the likelihood of the observation around the pose predicted by odometry

$$x_t^{(i)} = \operatorname{argmax}_{x_t} p(x_t | \tilde{m}_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}). \quad (6)$$

Analog to Eq. (3)-(5), we can express the proposal of particle  $j$  using the one of particle  $i$ . The only difference is that we do not apply the  $\oplus$  and  $\ominus$  operators based on the poses of the samples. Instead, the operators are applied based on the particle dependent reference frames  $l^{(i)}$  and  $l^{(j)}$  of the local maps. These reference frames were established when previously mapping the terrain. This results in

$$p(x_t | \tilde{m}_{t-1}^{(j)}, x_{t-1}^{(j)}, z_t, u_{t-1}) \approx l^{(j)} \oplus (p(x_t | \tilde{m}_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}) \ominus l^{(i)}). \quad (7)$$

Combining Eq. (6) and Eq. (7) leads to

$$x_t^{(j)} = \operatorname{argmax}_{x_t} p(x_t | \tilde{m}_{t-1}^{(j)}, x_{t-1}^{(j)}, z_t, u_{t-1}) \quad (8)$$

$$\approx l^{(j)} \oplus (\underbrace{\operatorname{argmax}_{x_t} p(x_t | \tilde{m}_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_{t-1}) \ominus l^{(i)}}_{x_t^{(i)}}) \quad (9)$$

$$= l^{(j)} \oplus (x_t^{(i)} \ominus l^{(i)}). \quad (10)$$

Under the Assumptions 2 and 3, we can estimate the poses of all samples according to Eq. (10). In this way, the complex computation of an informed proposal needs to be done only once. When the robot is in one of the two situations described above, the computation of the importance weights is done as proposed in [7] except that we have to evaluate the weights only once.

### D. Computing the Proposal When Closing a Loop

In contrast to the two situations described before, the computation of the proposal is more complex in case of a loop-closure. This is due to the fact that Assumption 2 (local similarity) is typically violated even for subsets of particles. This fact can be illustrated by supposing a widely spread cloud of particles when closing a loop. The different samples re-enter the previously mapped terrain at different locations. This results in different hypotheses about the topology of the environment and definitively violates Assumption 2. Dealing with such a situation, requires additional effort in the estimation process.

Let us start with the informed proposal considering all sensor observations  $z_{1:t}$  and the most recent odometry reading  $u_{t-1}$ . The proposal can be factorized as

$$p(x_t | z_{1:t}, x_{1:t-1}^{(i)}, u_{t-1}) = \eta p(z_t | z_{1:t-1}, x_{1:t-1}^{(i)}) p(x_t | x_{t-1}^{(i)}, u_{t-1}) \quad (11)$$

$$= \eta p(z_t | x_t, m_{t-1}^{(i)}) p(x_t | x_{t-1}^{(i)}, u_{t-1}), \quad (12)$$

where  $\eta$  is a normalizer resulting from Bayes' rule.

Whenever a particle  $i$  closes a loop, we consider that its map  $m_{t-1}^{(i)}$  consists of two components. The first one is a local map  $m_{\text{local}}^{(i)}$ , which has no overlap with the previously seen area and does not affect the loop closure. Secondly, a loop map  $m_{\text{loop}}^{(i)}$  which models a previously mapped part of the environment re-visited after moving through unknown terrain for a long period of time.

$$p(z_t | x_t, m_{t-1}^{(i)}) = p(z_t | x_t, m_{\text{local}}^{(i)}, m_{\text{loop}}^{(i)}) \quad (13)$$

Under the assumption that these two maps are disjoint, it is possible to choose a likelihood function that allows us to apply the following factorization

$$p(z_t | x_t, m_{\text{local}}^{(i)}, m_{\text{loop}}^{(i)}) \propto p(z_t | x_t, m_{\text{local}}^{(i)})p(z_t | x_t, m_{\text{loop}}^{(i)}) \quad (14)$$

Notice that the computation of the proposal in case of a loop-closure is more expensive than in the two other situations. Fortunately, loop-closing situations occur rarely. The robot has to travel through unknown and eventually known terrain for a comparably long period of time before a loop-closure can occur.

According to the importance sampling principle, the particle weights are given by

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{p(x_t^{(i)} | z_t, x_{t-1}^{(i)}, m_{\text{local}}^{(i)}, m_{\text{loop}}^{(i)}, u_{t-1})}{p(x_t^{(i)} | z_t, x_{t-1}^{(i)}, m_{\text{local}}^{(i)}, u_{t-1})} \quad (15)$$

$$= w_{t-1}^{(i)} \frac{\eta_1^{(i)} p(z_t | x_t^{(i)}, m_{\text{local}}^{(i)}) p(z_t | x_t^{(i)}, m_{\text{loop}}^{(i)})}{\eta_2^{(i)} p(z_t | x_t^{(i)}, m_{\text{local}}^{(i)})} \quad (16)$$

$$= w_{t-1}^{(i)} p(z_t | x_t^{(i)}, m_{\text{loop}}^{(i)}) \frac{\eta_1^{(i)}}{\eta_2^{(i)}}, \quad (17)$$

where  $\eta_1$  and  $\eta_2$  are normalization factors resulting from Bayes' rule.

### E. Approximative Importance Weight Computation

Eq. (17) tells us how to update the particle weights in case of a loop closure. Unfortunately, the computation of the normalizing factors  $\eta_1$  and  $\eta_2$  cannot be done efficiently. Therefore, in our current implementation, the weights are evaluated according to the raw observation model based on the loop map  $m_{\text{loop}}$

$$w_t^{(i)} \approx w_{t-1}^{(i)} p(z_t | x_t^{(i)}, m_{\text{loop}}^{(i)}) \quad (18)$$

rather than according to Eq. (17). This means that we ignore the ratio of the normalizing factors  $\eta_1$  and  $\eta_2$  and approximate the importance weights when closing a loop. This is significantly faster to compute and as we will demonstrate in the experiments, the approximation error is small.

V. A S E , L S ,  
P T

All of the derivations made in the previous section require the robot to know whether it is moving through unknown terrain, through a previously mapped area, or is currently closing a loop (Assumption 1). Here, we describe how to distinguish the different cases. Detecting the first two situations

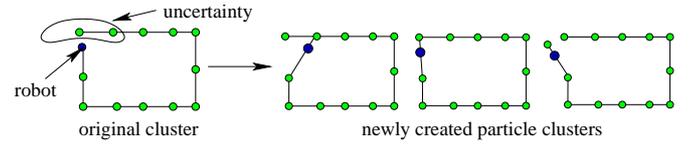


Fig. 2. The left image depicts a cluster while the robot is approaching a loop-closure. The shown particle cluster splits up into three different clusters (topology hypotheses) as depicted in the right image.

can be done in a straightforward way by comparing the area covered by the current observation given the particle pose and the map constructed so far.

More difficult is to decide whether or not the robot is closing a loop. To make this decision, we apply the approach proposed by Stachniss *et al.* [19] in the context of exploration with active loop-closing. This approach uses a dual representation consisting of a grid map and a topologic map that models the trajectory of the vehicle. By comparing both representations, one is able to accurately determine whether or not a robot is closing a loop.

Assumption 2 (local similarity) typically holds only up to the first loop closure but is then violated. By explicitly modeling the different topological hypotheses, it is still possible to represent the posterior in an appropriate way. To achieve local similarity, we introduce the notation of a *particle cluster* which describes a subset of particles for which the assumption of local similarity between maps holds. Ambiguities in the posterior can then be modeled using multiple particle clusters. Such clusters are obtained by grouping similar samples so that the maps within one cluster represent the same topology.

In the following, we explain how to represent such a set of samples and how to split up a particle cluster in case the assumption of local similarity is violated.

In our current system, we represent a map as a set of local maps also called patches. A global map for a given particle can be obtained by specifying the location of each patch within a global reference frame. Each sample therefore has to store only a list of reference frames  $l_n^{(i)}$  for the patches. In this way, the individual patches  $\mathcal{P}_1, \dots, \mathcal{P}_N$  need to be stored only once per cluster. The map of particle  $i$  can be computed by

$$m^{(i)} = \bigcup_n l_n^{(i)} \oplus \mathcal{P}_n. \quad (19)$$

Within one particle cluster, the local maps of each particle fulfill the assumption of local similarity. Therefore, they can share their patches. This results in a more compact representation compared to storing individual grid maps. In our current implementation, we used a graph structure where each node is a reference to the corresponding patch. To actually implement this representation, we store for each particle the state vector  $s_t^{(i)}$

$$s_t^{(i)} = \langle \underbrace{x_t^{(i)}}_{\text{robot pose}}, \underbrace{k}_{\text{cluster ID}}, \underbrace{l_1^{(i)}, \dots, l_{N_k}^{(i)}}_{\text{patches locations}} \rangle. \quad (20)$$

Each cluster  $C_k$  is represented by

$$C_k = \langle \underbrace{\mathcal{P}_1, \dots, \mathcal{P}_{N_k}}_{\text{pointer to patches}}, \underbrace{\{e_{l,m}\}}_{\text{graph edges}} \rangle. \quad (21)$$

Note that the number  $N_k$  of patches does not grow with the length of trajectory traveled by the robot. It grows with the

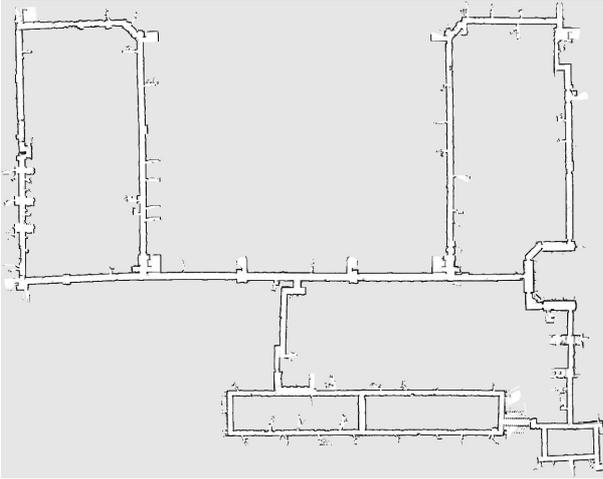


Fig. 3. Learned map of the MIT Killian Court using our approach.

number of relevant patches which is related to the size of the environment.

In the beginning of the mapping process, we start with a single cluster, but after closing a loop, multiple topology hypotheses typically occur. Whenever a topological hypothesis represented by a particle cluster needs to be split up, one has to determine which particle belongs to which topological hypothesis. In our current implementation, we cluster the samples according to their Euclidian distance to the different nodes in their own graph structure of reference frames. For each particle, we determine the list of nodes in the field of view of that sample. We order this list according to the Euclidian distance from the pose represented by the sample to the corresponding node. Then, a cluster is given by the samples which have the same list of nodes. An example which illustrates how new clusters are created in case of a loop-closure is depicted in Figure 2.

Throughout our experiments, we observed that multiple particle clusters are created when closing a loop. The actual number ranges up to 50. However, after a few iterations only a small number of cluster (up to 5) typically survive.

Note that it might be possible to represent each cluster by an EKF and not by particles like we do. However, in this case one would have to deal with linearization problems and Gaussian uncertainty. Furthermore, our approach uses grid maps and does not rely on predefined feature extractors like typical EKF approaches do.

To fulfill Assumption 3, we apply an incremental scan alignment technique based on laser range finder data. The experiments presented in this paper indicate that this setup/implementation is sufficient to satisfy the three assumptions. As a result, we obtain a mapping system which provides highly accurate maps in a fast and memory efficient manner.

## VI. E

In this section, we present experiments based on real robot datasets which are commonly used within the SLAM community. In the first experiment, we corrected several log files using our approach. Figure 3 depicts the resulting map of the MIT Killian Court. This is a challenging dataset, since it is a large (it took 2.5h to record this log file) and it contains

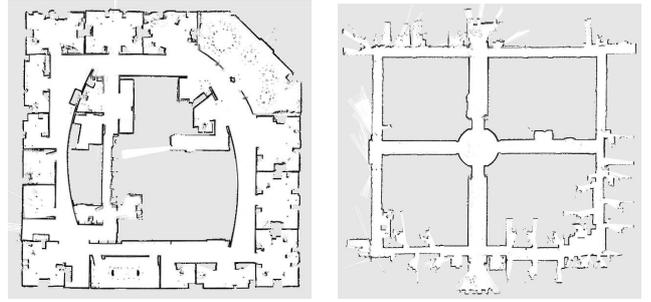


Fig. 4. The left image depict the Intel Research Lab and the right one the Austin ACES building at the University of Texas.

TABLE I

	C	MIT	
	PC	1.3 GH CPU.	
	#particles	execution time	max. memory
our approach	2,000	51 min	210 MB
our approach	1,000	41 min	180 MB
our approach	500	30 min	165 MB
RBPF of [18]	150	(memory swapping)	2.9 GB
RBPF of [18]	80	300 min	1.5 GB
RBPF of [18]	50	190 min	1 GB

several nested loops which can lead to particle depletion. As shown in this figure, the map does not show inconsistencies like for example double walls. Comparable results have been obtained using the Intel Research Lab and the Austin ACES dataset which are both depicted in Figure 4.

The second experiment is designed to show the advantages of our approach compared to a Rao-Blackwellized mapper without our optimizations. For this comparison, we used the open-source mapper provided in [18]. We compared the overall time, needed to correct the MIT Killian Court dataset and the memory used to store the maps. This was done using a (comparably slow) PC with a 1.3 GHz CPU and 1.5 GB RAM. The results of both mapping approaches are summarized in Table I. In our current implementation, the filter update for *each cluster* takes in average 20ms when moving through known terrain and 200ms when moving through unknown terrain. When actually closing a loop, *each particle* requires approximately 2ms of execution time while the check for the closure takes around 0.3ms per sample.

Since the approximated proposal is not as accurate as the original one, we need more particles to achieve the same robustness in filter convergence and quality of the resulting maps. However, we can maintain more than one order of magnitude more particles while requiring less runtime and memory. In all our experiments, this sufficiently accounted for the less accurately drawn samples.

The savings on runtime are mainly caused by transforming an already computed proposal distribution so that it can be used for several particles instead of computing it from scratch each time. The memory savings are due to the fact that all particles within a cluster can share a single map model. Furthermore, the memory usage and runtime of our approach grows much slower when increasing the number of particles. The reason is that the complexity of our filter grows mainly with the number of topological hypotheses (particle clusters) which need to be maintained and not directly with the number

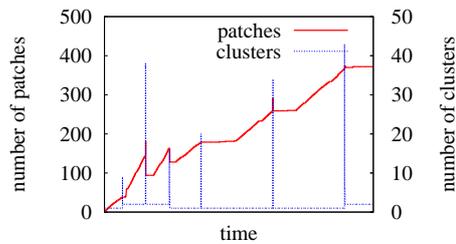


Fig. 5. This plot depicts the number of patches in the memory and the number of clusters over time for the MIT dataset using 1500 particles.

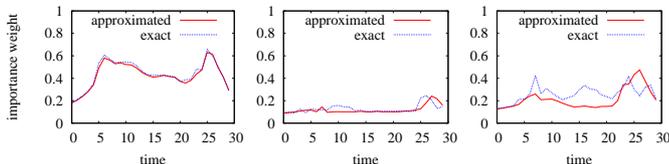


Fig. 6. Difference in the particle weights caused the approximative computation for three different samples during a loop closure. The left and middle image show typical results, the right one depicts the one of the worst results during our experiments.

of samples. Notice that the *maximum* memory usage shown of our approach is much higher than the typical one. There exist a few peaks in the memory usage which arise from a loop closure where several clusters are temporarily created but deleted after a few steps (compare Figure 5). The typical memory usage is around 20% of the maximum usage.

Figure 5 depicts the number of patches that need to be stored and the number of clusters during the estimation process of the MIT dataset with 1,500 particles. As can be seen, the number of clusters is typically small until a loop closure occurs. At this point, the number of clusters increases. However, after a short period of time most of the clusters vanish.

The last experiment evaluates the error introduced by our approximative importance weight computation when closing a loop. We ignore the normalization factors to achieve a faster estimation. We analyzed the loop-closing actions and in most situations the approximation error was small. Figure 6 depicts the differences between the sound computation and our approximation for three different particles during a loop closure. For a more quantitative evaluation between both methods, we computed the KL-divergence (KLD) between the distribution of the importance weights in both cases. It turned out, that the average KLD was only 0.02 (a KLD of 0 means that the distributions are equal and the higher the value the more different are the distributions). Substantiated by the good approximation quality, we ignore the evaluation of  $\eta_1$  and  $\eta_2$  when computing the particle importance weight.

### VII. C

In this paper, we presented efficient optimizations for Rao-Blackwellized SLAM on grid maps. We are able to update the complex posterior requiring substantially less resources by performing the computations only for a set of representatives instead of for all particles. We extended a state-of-the-art mapping system in a way that the computation of the proposal distribution is significantly faster and needs only a fraction of the memory resources. The key idea is that clusters of particles can share a compact map representation as well as an informed proposal distribution to draw the next generation of particles.

With our optimizations, we are able to maintain more than one order of magnitude more samples and at the same time require less memory and computational resources compared to other state-of-the-art Rao-Blackwellized mapping techniques. This increase in number of particles we are able to maintain compensates for the errors introduced by our approximations.

Our approach has been implemented, tested, and evaluated based on real robots and standard log files used within the SLAM community to demonstrate the accuracy as well as the benefits of our system.

### R

- [1] M. Bosse, P.M. Newman, J.J. Leonard, and S. Teller. An atlas framework for scalable mapping. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, Taipei, Taiwan, 2003.
- [2] A. Doucet, J.F.G. de Freitas, K. Murphy, and S. Russel. Rao-Blackwellized particle filtering for dynamic bayesian networks. In *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2000.
- [3] A. Eliazar and R. Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003.
- [4] C. Estrada, J. Neira, and J.D. Tardós. Hierarchical slam: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596, 2005.
- [5] R. Eustice, M. Walter, and J.J. Leonard. Sparse extended information filters: Insights into sparsification. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 641–648, 2005.
- [6] U. Frese and G. Hirzinger. Simultaneous localization and mapping - a discussion. In *Proc. of the IJCAI Workshop on Reasoning with Uncertainty in Robotics*, pages 17–26, Seattle, WA, USA, 2001.
- [7] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with Rao-Blackwellized particle filters by adaptive proposals and selective resampling. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2443–2448, Barcelona, Spain, 2005.
- [8] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc. of the IEEE Int. Symp. on Computational Intelligence in Robotics & Automation (CIRA)*, pages 318–325, 1999.
- [9] D. Hähnel, W. Burgard, D. Fox, and S. Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2003.
- [10] J.J. Leonard and H.F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(4):376–382, 1991.
- [11] B. Lisen, D. Silver D. Morales, G. Kantor, I.M. Rekleitis, and H. Choset. Hierarchical simultaneous localization and mapping. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2003.
- [12] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Journal of Autonomous Robots*, 4:333–349, 1997.
- [13] M. Montemerlo, S. Thrun D. Koller, and B. Wegbreit. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proc. of the Int. Conf. on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, 2003.
- [14] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to simultaneous localization and mapping. In *Proc. of the National Conference on Artificial Intelligence (AAAI)*, 2002.
- [15] K. Murphy. Bayesian map learning in dynamic environments. In *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*, 1999.
- [16] J. Neira and J.D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17(6), 2001.
- [17] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. Cox and G. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer Verlag, 1990.
- [18] C. Stachniss and G. Grisetti. Mapping results obtained with Rao-Blackwellized particle filters. <http://www.informatik.uni-freiburg.de/~stachnis/research/rbpfmapper/>, 2004.
- [19] C. Stachniss, D. Hähnel, W. Burgard, and G. Grisetti. On actively closing loops in grid-based fastslam. *Advanced Robotics*, 19:1059–1080, 2005.
- [20] S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Int. Journal of Robotics Research*, 23(7/8), 2004.