in: Proc. 52nd Int. Scientific Colloquium (IWK), 2007, Technische Universität Ilmenau, Germany, Vol. I, pp. 445-450, Universitätsverlag 2007

C. Schröter, H.-M. Gross

Efficient Gridmaps for SLAM with Rao-Blackwellized Particle Filters

Abstract

Simultaneous localization and mapping (SLAM) has been an important field of research in the robotics community in recent years. A successful class of SLAM algorithms are Rao-Blackwellized Particle Filters (RBPF), where the particles approximate the pose belief distribution, while each particle contains a separate map. So far, RBPF with landmark based environment representations as well as gridmaps have been shown to work. Existing gridmap approaches typically used laser range scanners, because the high accuracy of that sensor keeps the state uncertainty low and allows for efficient solutions. In this paper, we present a combination of our previous work on map-matching with RBPF, which enable us to solve the SLAM problem also with low-resolution sonar range sensors. Furthermore, we introduce a simple and fast but very efficient shared representation of gridmaps which reduces the memory cost overhead caused by inherent redundancy between the particles.

1 Introduction and Related Work

In order to navigate autonomously, a basic requirement for a mobile robot is the ability to build a map of the environment. Because mapping depends on a good estimate of the robot's pose w.r.t. the environment, while localization needs a consistent map, the localization and mapping problems are coupled in applications where an unknown area has to be explored without an external position reference like GPS. The term Simultaneous Localization And Mapping (SLAM) has been coined for this problem [1]. SLAM can be seen as a generalization of the map building problem, as it describes the objective of aquiring a map of the environment without assuming any additional position information apart from those that can be derived from the mapping process itself.

There are two main criteria that can be used to categorize existing SLAM techniques: the kind of model used to describe the robot and environment state and the algorithm that is utilized to estimate the state belief.

In many SLAM approaches, the map representation is assumed to be a vector of point-like feature positions [2], also called landmarks. The attractiveness of feature/landmark-based representations for SLAM lies in their compactness. However, they rely on *a priori* knowledge about the structure of the environment to identify and distinguish potential landmarks. Furthermore, a data association problem arises from the need to robustly recognize landmarks. In contrast to landmark representations, gridmaps [3] do not make assumptions about specific features to be observable in the environment. They can represent arbitrary environment structures with nearly unlimited detail. However, they require a large amount of memory.

An effective means of handling the high-dimensionality in the SLAM problem has been introduced in the form of the Rao-Blackwellized Particle Filter (RBPF): in this approach the state space is partitioned into the pose and map state. A particle filter approximates the pose belief distribution of the robot, while each particle contains a map which represents the model of the environment, assuming the pose estimation of that specific particle to be correct.

Our aim here is to use a RBPF for grid mapping using no other sensory input than robot odometry and low-resolution sonar range scans. Since this requires a relatively large number of particles, we have to emphasize the efficient representation of the maps carried by the particles. To this purpose we present a short analysis of map redundancy between particles and a map storing scheme that exploits that redundancy in order to save memory.

The rest of the paper is organized as follows: We give a short introduction to the RBPF approach for SLAM in the next section. Section 3 will explain the specific details of our Sonar-SLAM implementation, while section 4 deals with the shared gridmap representation. Experiments with real robot data are presented and discussed in section 5, section 6 closes with a short summary and outlook.

2 Rao-Blackwellized Particle Filter for SLAM

As already described before, the complexity of the SLAM problem arises from the very high-dimensional state space, consisting of the variables describing the robot pose and the variables describing the environment state. In the case of gridmaps, the map alone usually contains a few thousands up to several million cells, each of which corresponding to a state variable. Obviously, a full posterior over the state is extremely costly to estimate. The idea of the RBPF in application to SLAM is to use a particle filter to estimate the robot trajectory distribution $p(x_{1:t}|z_{1:t}, u_{0:t})$ given the sequence of odometry measurements $u_{0:t}$ and environment observations $z_{1:t}$. This trajectory estimate is then used to estimate the desired distribution over map and trajectory:

$$p(x_{1:t}, m|z_{1:t}, u_{1:t}) = p(m|x_{1:t}, z_{1:t})p(x_{1:t}|z_{1:t}, u_{0:t}) \tag{1}$$

The particle filter works analogous to Monte-Carlo-Localization [7], except that instead of one given map each particle contains a separate map. To calculate the importance weights for $p(x_{1:t})$, each particle uses its own map. The map, in return, is built from the estimated trajectory of that corresponding particle. The effect is that a number of hypothesis maps are built, each corresponding to a possible trajectory. Importance weighting is performed with the weight for particle i following

$$w^{(i)} \simeq \frac{p(x_t^{(i)}|z_{1:t}, u_{0:t})}{\pi(x_t^{(i)}|z_{1:t}, u_{0:t})} \tag{2}$$

Here, $\pi(x_t^{(i)})$ denotes the proposal distribution. Typically, the motion model is used to generate the proposal distribution from the last particle generation (again, in analogy to localization), in which case the weight formula simplifies to

$$w^i \simeq p(z_t|x_t^{(i)}, m^{(i)}) \tag{3}$$

By repeatedly calculating importance weights followed by resampling to adapt the particle distribution to the estimated distribution, particles are preferred whose maps match new observations best, therefore the most likely map is selected.

3 Sonar Grid SLAM

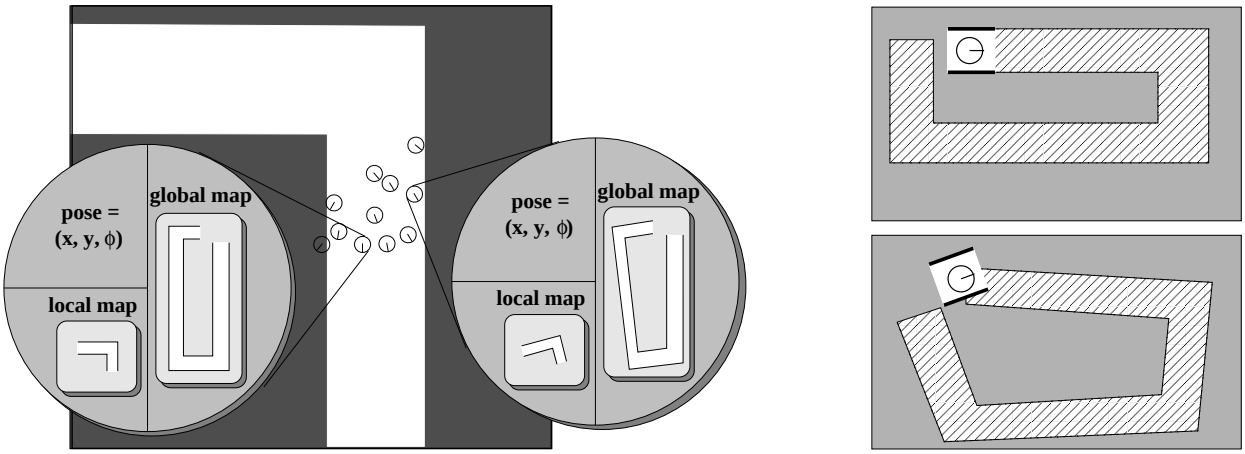

Figure 1: Left - Data representation overview: The particles model the distribution of the robot pose belief. Each particle carries a full map of the environment, which is a combination of the full particle trajectory and the sonar range measurements, and a local map, which only contains the most recent measurements. Right - Map matching: For the upper particle the local map (clean white) is aligned to the global map (hatched) very well, while for the lower particle, which does not contain a position belief consistent with the environment, the local map conflicts with the global map. This situation would result in a higher weight for the upper particle.

The base of our Sonar SLAM approach is a particle filter, where each particle contains a pose estimate as well as a map estimate. Without loss of generality we can assume the robot to start mapping at position (0,0,0). While the robot moves, the particles move as well, according to the odometry readings and the probabilistic odometry motion model, which describes the uncertainty in the actual robot motion. Due to this uncertainty, the motion model contains a stochastic component, which effects in the particles spreading out and generating slightly different trajectories. Additionally, during motion the robot observes the environment by means of sonar range sensors. A map update is triggered frequently (approx. every 0.2m). In that map update, each particle adds the new environment observation to its own map, at its own estimated current position. Since the position estimates of the particles are slightly different, the maps differ as well (Fig. 1).

In order to determine the likeliness of a map hypothesis, we need to calculate particle weights by comparing expected and sensed measurement.

We already presented a way of comparing expectation and observation from sonar range sensors in a previous work on mapping [6]. There, we proposed an approach we called map matching: a local map was built from only the most recent sonar measurements and the resulting local map was matched against the global map to find the most likely position w.r.t. that global map. In order to be able to use map matching, each particle must not only know its global map, but also a local map. We exclude the most recent range measurements from the global map, and use those measurements for the local map. That way, global and local map are built from different data and we avoid comparing certain measurements against themselves. The local map can either be rebuilt from the pose and scan queues for each weight calculation or be persistent in the particle by just adding every new scan and forgetting old scans. Making the local map persistent is more efficient but less flexible.

The calculation of the match value between the local and global map is quite simple: For each occupied cell in the local map the occupancy value of the corresponding cell in the global map is tested. If the global map cell also is occupied, that cell contributes with a value of $+1$. If the global map cell is free, it contributes with a value of -1. Cells with unknown or undecided occupancy do not contribute. That way, the match value is positive if local and global map are very similar, and it is negative if many objects exist in the local map where there is free space in the global map. To obtain the actual particle weight $match^{(i)}$, an exponential function is applied as follows:

$$w^{(i)} = e^{\frac{match^{(i)}}{f}} \tag{4}$$

with f being a free parameter to influence the spread in the particle weights and therefore the speed of convergence.

4 Shared Gridmaps

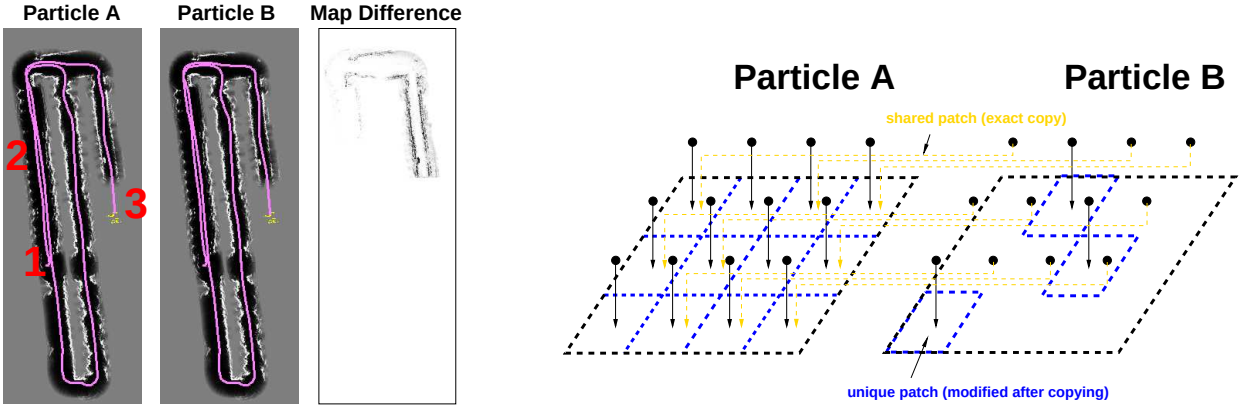

Figure 2: Left - Analysis: The robot started at position 1, closed the loop and moved onward to position 3. Particle B was generated as a copy of A during resampling approx. at position 2. Therefore, the major part of the map is identical between particles A and B. Right - Improved map representation: Particle A contains a map which consists of a number of separate patches. Particle B is created as a copy of A - the map of B consists of references to the patches in A. Only when A or B modifies a certain patch, it creates a real copy, so Particle A and B then have a separate instance of that patch.

A major problem with using gridmaps in RBPF is the memory cost: In a naive implementation, the number of cell values to be stored would be the product of grid size and particle number. However, the maps of the individual particles are not completely independent: In the resampling as part of the observation update, particles with low weights are deleted and replaced by copies of particles with higher weights. This results in multiple identical copies of the same map. Afterwards, each of the particles will modify its respective map differently, according to the path assumed through the probabilistic motion model: The copies will not remain identical, but it is important to notice the changes often only affect a small area of the already aquired map (see Fig. 2). The idea to save wasting memory for redundant information therefore is to split up the map into smaller patches and share those patches across the particles. When a particle A is cloned, each "copy" of a map patch belonging to the clone particle B is just a reference to the original patch. Only when either A or B modify a map patch later, a real copy is created in the local memory of the respective particle.

The effect of this representation is that the memory cost is not determined by the map area, but by the size of path loops. As long as a loop is not closed yet, particles are diverging and many path hypotheses are maintained. When the loop gets closed, only the best particles survive, and new particles are generated as copies of those few best fitting hypotheses. While a loop is open, each particle holds an own independent map of that loop, but when it is closed, only few unique maps of that specific loop (the best fitting ones) continue to exist. Therefore, the "residual" memory cost is determined by the entire map area (the sum of all loops) and nearly independent of the particle number, while the peak memory cost is determined by particle number and maximum length of a single loop.

5 Experiments

To test our approach we built maps of a home store which is the regular test environment for our navigation algorithms. This environment is very well suited for our proposed SLAM approach as it essentially consists of a large number of small circles of hallways (50 to 100 m loop length). Fig. 3 shows the resulting map and the the overall memory usage for all particles over time. The data shows that our SLAM approach using map matching and shared gridmaps builds a consistent map with a bounded amount of memory. Only robot odometry and sonar range sensors were used in those experiments.

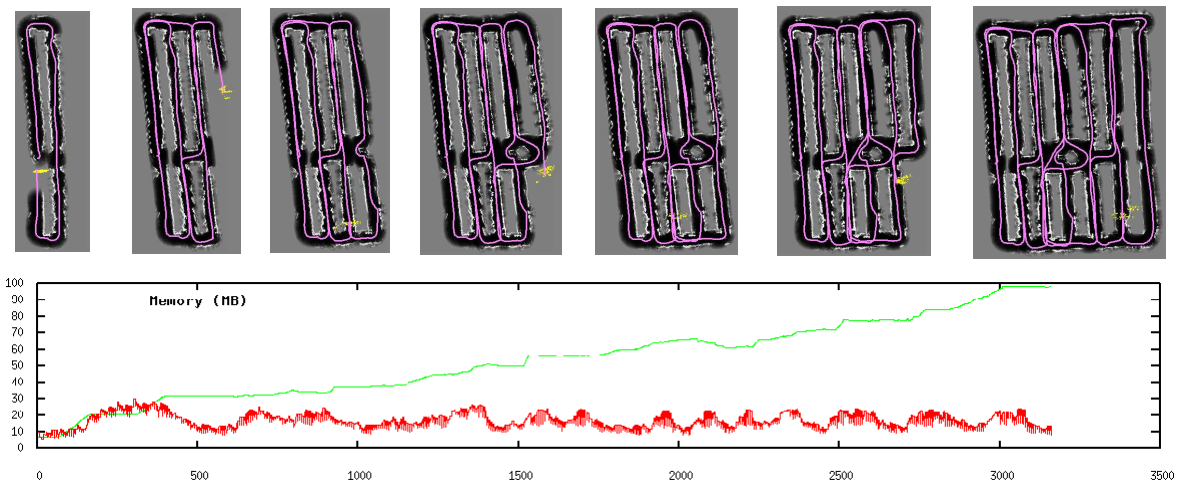

Figure 3: Row 1: Several steps of mapping (500 particles): Yellow dots denote the particle positions, path (magenta line) and map for one selected particle are shown.
Row 2: Map memory cost for plain gridmaps (green) and shared maps (red, see section 4). It is clearly visible that the memory for plain maps is growing monotonously, while for the shared maps the cost collapses with each loop closure.

6 Summary & Outlook

We presented an implementation of RBPF with gridmaps which is able to solve the SLAM problem with low-resolution sensors such as sonar range finders. Furthermore, we introduced a shared map representation for particle filters which effectively makes the maximum memory cost depend on the loop size instead of the overall map size. Experiments show that our approach is well suited for large-scale environments consisting of many loops.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*, The MIT Press, 2005.

[2] R. Smith, M. Self, and P. Cheeseman, "A stochastic map for uncertain spatial relationships," in *4th International Symposium on Robotic Research*. 1987, MIT Press.

[3] H. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI Magazine*, vol. 9, no. 2, pp. 61–77, 1988.

[4] K. P. Murphy, "Bayesian map learning in dynamic environments," in *Advances in Neural Information Processing Systems 12 (NIPS99)*, 1999, pp. 1015–1021.

[5] D. Haehnel, 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. IROS-2003*, pp. 206–211.

[6] C. Schroeter, H.-J. Boehme, and H.-M. Gross, "Robust map building for an autonomous robot using low-cost sensors," in *Proc. SMC-2004*, pp. 5398 – 5403.

[7] D. Fox, W. Burgard, F. Dellaert and S. Thrun, "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots" in *Proc. AAAI Natl. Conf. on Artifical Intelligence*, 1999, pp. 5398 – 5403.

[8] C. Schroeter, A. Koenig, H.-J. Boehme, and H.-M. Gross, "Multi-sensor Monte-Carlo-Localization combining omnivision and sonar range sensors," in *Proc. ECMR-2005*, pp. 164–169.

Author Information:

Christof Schröter, Horst-Michael Gross Department of Neuroinformatics and Cognitive Robotics, Faculty of Computer Science and Automation,
Ilmenau Technical University, PO Box 10 05 65, 98684 Ilmenau
Tel: +49 3677 69 1306
Fax: +49 3677 69 1665
E-mail: christof.schroeter@tu-ilmenau.de