



On actively closing loops in grid-based FastSLAM

Cyrill Stachniss , Dirk Hähnel , Wolfram Burgard & Giorgio Grisetti


To cite this article: Cyrill Stachniss , Dirk Hähnel , Wolfram Burgard & Giorgio Grisetti (2005)
On actively closing loops in grid-based FastSLAM, Advanced Robotics, 19:10, 1059-1079, DOI:
[10.1163/156855305774662181](https://doi.org/10.1163/156855305774662181)

To link to this article: <https://doi.org/10.1163/156855305774662181>



Published online: 02 Apr 2012.



Submit your article to this journal 



Article views: 108



View related articles 



Citing articles: 32 View citing articles 

On actively closing loops in grid-based FastSLAM

CYRILL STACHNISS^{1,*}, DIRK HÄHNEL^{1,2}, WOLFRAM BURGARD¹
and GIORGIO GRISETTI^{1,3}

¹ University of Freiburg, Department of Computer Science, 79110 Freiburg, Germany

² Intel Research Seattle, 1100 NE 45th Street, Seattle, WA 98105, USA

³ Dipartimento Informatica e Sistemistica, Università 'La Sapienza', 00198 Rome, Italy

Received 1 March 2005; accepted 23 May 2005

Abstract—Acquiring models of the environment belongs to the fundamental tasks of mobile robots. In the past, several researchers have focused on the problem of simultaneous localization and mapping (SLAM). Classical SLAM approaches are passive in the sense that they only process the perceived sensor data and do not influence the motion of the mobile robot. In this paper, we present a novel integrated approach that combines autonomous exploration with simultaneous localization and mapping. Our method uses a grid-based version of the FastSLAM algorithm and considers at each point in time actions to actively close loops during exploration. By re-entering already visited areas, the robot reduces its localization error and in this way learns more accurate maps. Experimental results presented in this paper illustrate the advantage of our method over previous approaches that lack the ability to actively close loops.

Keywords: Exploration; active loop-closure; place re-visiting; re-localization; FastSLAM.

1. INTRODUCTION

In general, the task of acquiring models of unknown environments requires the solution of three subtasks, which are mapping, localization and motion control. Mapping is the problem of integrating the information gathered with the robot's sensors into a given representation. Localization is the problem of estimating the position of the robot. Finally, the control problem involves the question of how to steer a vehicle in order to efficiently guide it to a desired location or along a planned trajectory.

The diagram in Fig. 1 depicts also the overlapping areas of these three tasks. Simultaneous localization and mapping (SLAM), is the problem of building a map based on pose estimations while simultaneously localizing the robot within the map

*To whom correspondence should be addressed. E-mail: stachnis@informatik.uni-freiburg.de

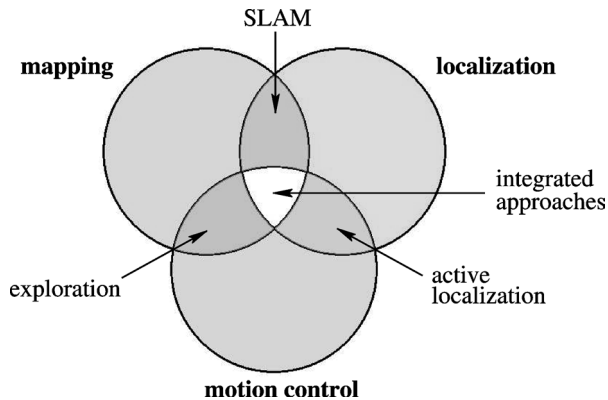


Figure 1. Subtasks that need to be solved by a robot in order to acquire accurate models of the environment [1]. The overlapping areas represent combinations of these subtasks.

constructed so far. Active localization seeks to guide the robot to locations within the map to improve the pose estimate. In contrast to this, exploration approaches focus on guiding the robot efficiently through the environment in order to build a map. The center area of the diagram represents the so-called integrated approaches which address mapping, localization and motion control simultaneously.

A naive approach to realize an integrated technique which solves all three tasks simultaneously could be to combine a SLAM algorithm with an exploration procedure. Since exploration strategies often try to cover unknown terrain as fast as possible, they avoid repeated visits to known areas. This strategy, however, is suboptimal in the context of the SLAM problem because the robot typically needs to re-visit places to localize itself again. A good pose estimate is necessary to make the correct data association, i.e. to determine if the current measurements fit into the map built so far. If the robot uses an exploration strategy that avoids multiple visits to the same place, the probability of making the correct associations is reduced. This indicates that combinations of exploration strategies and SLAM algorithms should consider whether it is worth re-entering already covered spaces or to explore new terrain. It can be expected that a system, which takes this decision into account, can improve the quality of the resulting map.

Figure 2 gives an example that illustrates why an integrated approach performing active place re-visiting provides better results than approaches that do not consider re-entering known terrain during the exploration phase. In the situation shown in the left-hand image, the robot traversed the loop just once. The robot was not able to correctly determine the angle between the loop and the straight corridor because it did not collect enough data to accurately localize itself. The second map shown in the right-hand image has been obtained with the approach described in this paper after the robot traveled twice around the loop before entering the corridor. As can be seen, this reduces the orientation error from approximately 7° (left image) to 1° (right image). This example illustrates that the capability to actively close

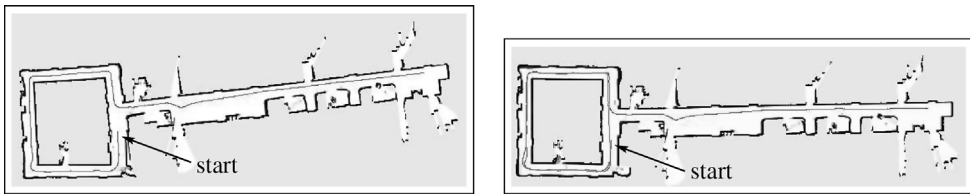


Figure 2. Two maps obtained from real world data acquired at Sieg Hall, University of Washington. The left-hand image depicts an experiment in which the robot traversed the loop only once before it entered the long corridor. As can be seen, the robot was unable to correctly close the loop, which led to an error of 7° in the orientation of the horizontal corridor. In the case in which the robot re-visited the loop, the orientation error was reduced to 1° (see right-hand image).

loops during exploration allows the robot to reduce its pose uncertainty during exploration and thus to learn more accurate maps.

The contribution of this paper is an integrated algorithm for generating trajectories to actively close loops during SLAM and exploration. Our algorithm uses a grid-based version of the FastSLAM algorithm and maintains a Rao-Blackwellized particle filter to estimate the map and the trajectory of the robot. Our approach explicitly takes into account the uncertainty about the pose of the robot during the exploration task. Additionally, it reduces the risk that the robot becomes overly confident in its pose when actively closing loops, which is a typical problem of particle filters in this context. As a result, we obtain more accurate maps compared to combinations of SLAM with greedy exploration.

This paper is organized as follows. The next section discusses related work and in Section 3 we then explain the idea of grid-based FastSLAM, the mapping algorithm used throughout this work. In Section 4, we present our integrated exploration technique. We describe how to take into account the pose uncertainty and how to actively close loops. Finally, Section 5 presents experiments carried out on real robots as well as in simulation.

2. RELATED WORK

Several previous approaches to SLAM and mobile robot exploration are related to our work. In the context of exploration, most of the techniques presented so far focus on generating motion commands that minimize the time needed to cover the whole terrain [2–5]. Other methods seek to optimize the view-points of the robot to maximize the expected information gain and to minimize the uncertainty of the robot about grid cells [6, 7]. Most of these techniques, however, assume that the location of the robot is known during exploration. In the area of SLAM, the vast majority of papers focus on the aspect of state estimation as well as belief representation and update [8–15]. These techniques, however, are passive and only consume incoming sensor data without explicitly generating controls.

Recently, some techniques have been proposed which actively control the robot during SLAM. For example, Makarenko *et al.* [1] as well as Bourgault *et al.* [16]

extracted landmarks out of laser range scans and used an Extended Kalman Filter to solve the SLAM problem. Furthermore, they introduced a utility function which trades off the cost of exploring new terrain with the utility of selected positions with respect to a potential reduction of uncertainty. The approaches are similar to the work done by Feder *et al.* [17] who considered local decisions to improve the pose estimate during mapping. Sim *et al.* [18] presented an approach in which the robot follows a parametric curve to explore the environment and considers place-revisiting actions if the pose uncertainty gets too high. These four techniques integrate the uncertainty in the pose estimate of the robot into the decision process of where to move next. However, they rely on the fact that the environment contains landmarks that can be uniquely determined during mapping.

In contrast to this, the approach presented in this paper makes no assumptions about distinguishable landmarks in the environment. It uses raw laser range scans to compute accurate grid maps. It considers the utility of re-entering known parts of the environment and following an encountered loop to reduce the uncertainty of the robot in its pose. In this way, the resulting maps become highly accurate.

3. GRID-BASED FASTSLAM

To estimate the map of the environment, we use a highly efficient variant of the FastSLAM algorithm [13], which itself is an extension of the Rao-Blackwellized particle filter for simultaneous localization and mapping proposed by Murphy *et al.* [14]. FastSLAM uses a set of weighted particles to represent the full posterior $p(x_{1:t}, m \mid z_{1:t}, u_{0:t-1})$ about the map m of the environment and the trajectory $x_{1:t}$ of the robot given a sequence of observations $z_{1:t}$ up to time t and the odometry measurements $u_{0:t-1}$. The key idea of the Rao-Blackwellized particle filter for SLAM is to separate the estimation of the trajectory of the robot from the map estimation process:

$$p(x_{1:t}, m \mid z_{1:t}, u_{0:t-1}) = p(x_{1:t} \mid z_{1:t}, u_{0:t-1})p(m \mid x_{1:t}, z_{1:t}, u_{0:t-1}) \quad (1)$$

$$= p(x_{1:t} \mid z_{1:t}, u_{0:t-1})p(m \mid x_{1:t}, z_{1:t}), \quad (2)$$

where (2) is obtained from (1) by assuming that m is independent of the odometry measurements $u_{0:t-1}$ given all the poses $x_{1:t}$ of the robot and the corresponding observations $z_{1:t}$.

To estimate the first term of (2), i.e. the posterior $p(x_{1:t} \mid z_{1:t}, u_{0:t-1})$, FastSLAM uses a particle filter. This filter estimates the trajectory of the robot based on the odometry information and the observed laser range data similar to Monte-Carlo localization [19].

Estimating the full posterior about the map and poses of the robot can be done efficiently since the quantity $p(m \mid x_{1:t}, z_{1:t})$ can be computed analytically once $x_{1:t}$ and $z_{1:t}$ are known. By assuming that laser range finders provide very accurate range observations and given one knows the poses of the robot while obtaining these observations, a grid map can be directly computed using ray-casting operations.

As a result, each of the samples represents a possible trajectory of the robot and additionally maintains an individual map which is updated upon ‘mapping with known poses’ [20] according to its own pose estimate.

One of the most common particle filtering algorithms is the Sampling Importance Resampling (SIR) filter. FastSLAM incrementally processes the observations and the odometry readings as they are available, and updates the set of samples representing the posterior about the map and the trajectory of the vehicle. The overall process can be summarized by the following four steps:

- (i) Sampling. The next generation of particles is obtained from the current generation, by sampling from a proposal distribution π . The motion model of the robot is often used as the proposal.
- (ii) Importance Weighting. An individual importance weight ω is assigned to each particle. The weights account for the fact that the proposal distribution π in general is not equal to the true distribution of successor states. In our filter, the weight of each particle is proportional to the likelihood $p(z_t | m, x_t)$ of the most recent observation given the map m associated with this particle and the corresponding pose x_t .
- (iii) Resampling. Particles with a low importance weight ω are typically replaced by samples with a high weight. This step is necessary since only a finite number of particles are used to approximate a continuous distribution. Furthermore, resampling allows to apply a particle filter in situations in which the true distribution differs from the proposal.
- (iv) Map estimating. For each particle, the corresponding map estimate is updated based on the obtained observation and the pose represented by that particle.

The FastSLAM algorithm used throughout this paper computes grid maps. It applies a scan-matching procedure to compute highly accurate odometry data and uses this corrected odometry in the prediction step of the particle filter [12]. In this way, the number of particles can be reduced so that even maps of large environments can be constructed online. An example for such a filter is illustrated in Fig. 3. It depicts three particles with the individually estimated trajectories and the maps updated according to the estimated trajectory. In the depicted situation, the robot closed a loop and the different particles produced different maps. Particle 1 generated a quite accurate pose estimate, whereas particles 3 yields big alignments errors. Therefore, particle 1 will get a much higher importance weight compared to particle 3. The weight of particle 2 will be between the weight of particle 1 and 3 because its alignment error is smaller than the one of particle 3, but bigger than the one of particle 1.

Note that adapting the map discretization leads to only minor changes in the resulting trajectory. Generally, the finer the resolution, the more precise the pose estimate and therefore the resulting map. However, our observation is that as long as the grid resolution is reasonable small, it mainly influences the execution time

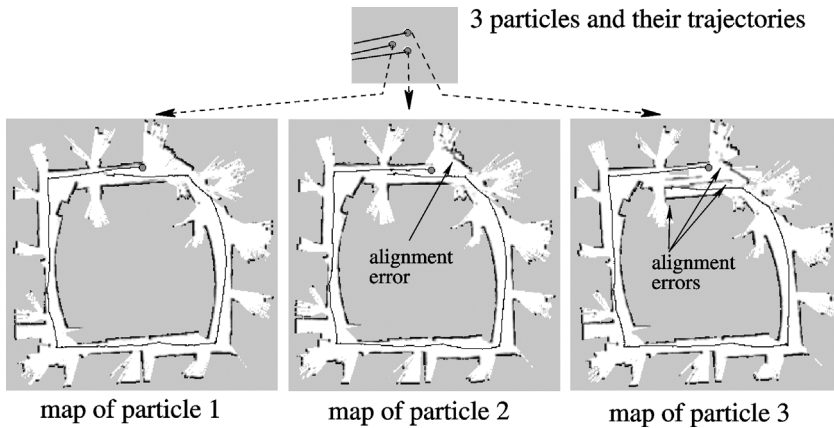


Figure 3. Example for three particles used within FastSLAM to represent $p(x_{1:t}, m \mid z_{1:t}, u_{0:t-1})$. Each particle estimates the trajectory of the robot and maintains an individual map which is updated according to the estimated trajectory.

of the algorithm and the memory requirements. Typically, the grid resolution is set values between 5 and 15 cm.

In the following section, we describe how to actively close loops during exploration in order to obtain accurate grid maps using the FastSLAM algorithm.

4. EXPLORATION WITH ACTIVE LOOP-CLOSING FOR FASTSLAM

During FastSLAM, whenever the robot explores new terrain, all samples have more or less the same importance weight, since the most recent measurement is typically consistent with the part of the map constructed from the immediately preceding observations. As a result, the uncertainty of the particle filter increases. As soon as it re-enters known terrain, however, the maps of some particles are consistent with the current measurement and some are not. Accordingly the weights of the samples differ largely. Due to the resampling step, the uncertainty about the pose of the robot usually decreases. One typical example is shown in Fig. 4. In the two left images, the robot explores new terrain and the uncertainty of the sample set increases. In the right image, the robot travels through known terrain and unlikely particles have vanished.

Note that this effect is much smaller if the robot just moves backward a few meters to re-visit previously scanned areas. This is because each map associated with a particle is generally locally consistent. Inconsistencies mostly arise when the robot re-enters areas explored some time ago. Therefore, visiting places seen further back in the history has a stronger effect on the differences between the importance weights and typically also on the reduction of uncertainty compared to places recently observed.

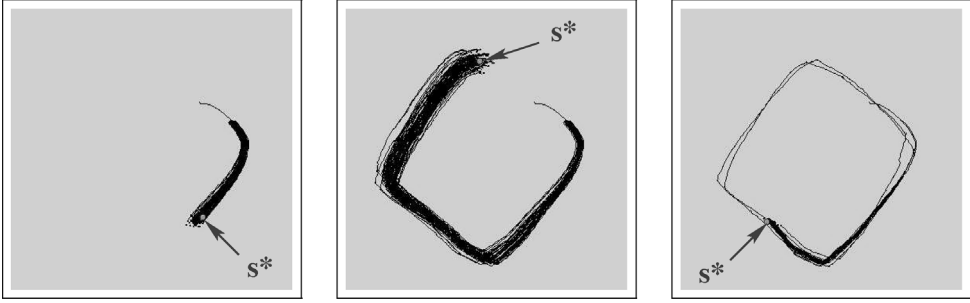


Figure 4. Evolution of a particle set and the map of the most likely particle s^* at three different time steps. In the two left images, the vehicle traveled through unknown terrain, so that the uncertainty increased. In the right image, the robot re-entered known terrain so that samples representing unlikely trajectories vanished.

4.1. Detecting opportunities to close loops

The key idea of our approach is to identify opportunities for closing loops during terrain acquisition. Here, closing a loop means actively re-entering the known terrain and following a previously traversed path. To determine whether there exists a possibility to close a loop we consider two different representations of the environment. In our current system, we associate with particle s an occupancy grid map $m^{[s]}$ and a topological map $\mathcal{G}^{[s]}$, both of which are updated while the robot is performing the exploration task. In the topological map $\mathcal{G}^{[s]}$, the vertices represent positions visited by the robot. The edges represent the trajectory corresponding to the particle s . To construct the topological map, we initialize it with one node corresponding to the starting location of the robot. Let $x_t^{[s]}$ be the pose of particle s at the current time step t . We add a new node at $x_t^{[s]}$ to $\mathcal{G}^{[s]}$ if the distance between $x_t^{[s]}$ and all other nodes in $\mathcal{G}^{[s]}$ exceeds a threshold c (here set to 2.5 m) or if none of the other nodes in $\mathcal{G}^{[s]}$ is visible from $x_t^{[s]}$:

$$\forall n \in \text{nodes}(\mathcal{G}^{[s]}) : [\text{dist}_{m^{[s]}}(x_t^{[s]}, n) > c \vee \text{not_visible}_{m^{[s]}}(x_t^{[s]}, n)]. \quad (3)$$

Whenever a new node is created, we also add an edge from this node to the most recently visited node. To determine whether or not a node is visible from another node, we perform a ray-casting operation in the occupancy grid $m^{[s]}$.

Figure 5 depicts such a graph for one particular particle during different phases of an exploration task. In each image, the topological map $\mathcal{G}^{[s]}$ is printed on top of the metric map $m^{[s]}$. To motivate the idea of our approach, we would like to refer the reader to the left image of this figure. Here, the robot was almost closing a loop. This can be detected by the fact that the length of the shortest path between the current pose of the robot and previously visited locations in the topological map $\mathcal{G}^{[s]}$ was large, whereas it was small in the grid-map $m^{[s]}$.

Thus, to determine whether or not a loop can be closed, we compute for each sample s the set $\mathcal{I}(s)$ of positions of interest which contains all nodes that are close

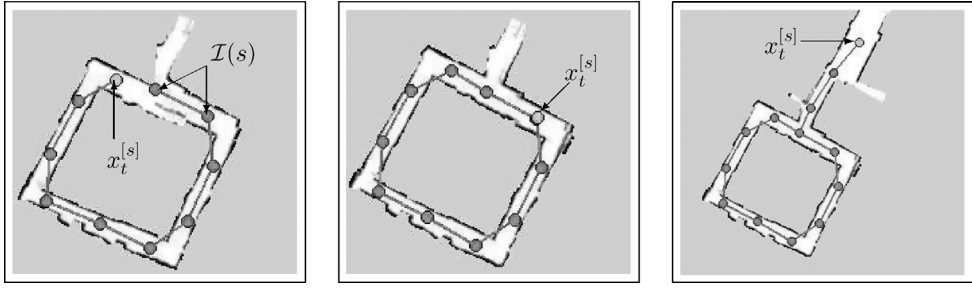


Figure 5. The red dots and lines in these three image represent the nodes and edges of $\mathcal{G}^{[s]}$. In the left image, $\mathcal{I}(s)$ contained two nodes (indicated by the arrows) and in the middle image the robot closed the loop until the pose uncertainty is reduced. After this, it continued with the acquisition of unknown terrain (right image).

to the current pose $x_t^{[s]}$ of particle s based on the grid map $m^{[s]}$, but are far away given the topological map $\mathcal{G}^{[s]}$ of s :

$$\mathcal{I}(s) = \{x_{t'}^{[s]} \in \text{nodes}(\mathcal{G}^{[s]}) \mid \text{dist}_{m^{[s]}}(x_{t'}^{[s]}, x_t^{[s]}) < c_1 \wedge \text{dist}_{\mathcal{G}^{[s]}}(x_{t'}^{[s]}, x_t^{[s]}) > c_2\}. \quad (4)$$

Here, $\text{dist}_{\mathcal{M}}(x_1, x_2)$ is the length of the shortest path from x_1 to x_2 given the representation \mathcal{M} . The distance between two nodes in $\mathcal{G}^{[s]}$ is given by the length of the shortest path between both nodes, whereas the length of a path is computed by the sum over the lengths of the traversed edges. Depending on the number of nodes in $\mathcal{I}(s)$, this distance information can be efficiently computed using either the A^* algorithm or Dijkstra's algorithm. The terms c_1 and c_2 are constants that must satisfy the constraint $c_1 < c_2$. In our current implementation, the values of these constants are $c_1 = 6$ m and $c_2 = 20$ m.

If $\mathcal{I}(s) \neq \emptyset$, there exist so-called shortcuts from the current pose $x_t^{[s]}$ represented by the corresponding particle to the positions in $\mathcal{I}(s)$. These shortcuts represent edges that would close a loop in the topological map $\mathcal{G}^{[s]}$. The left image of Fig. 5 illustrates a situation in which a robot encounters the opportunity to close a loop since $\mathcal{I}(s)$ contains two nodes which is indicated by two arrows. The key idea of our approach is to use such shortcuts whenever the uncertainty of the robot in its pose becomes too large. The robot then re-visits portions of the previously explored area and in this way reduces the uncertainty in its position.

To determine the most likely movement allowing the robot to follow a previous path of a loop, one in principle has to integrate over all particles and consider all potential outcomes of that particular action. Since this would be too time consuming for online processing, we consider only the particle s^* with the highest accumulated logarithmic importance weight:

$$s^* = \underset{s}{\operatorname{argmax}} \sum_{i=1}^t \log p(z_i^{m^{[s]}}, x_i^{[s]}). \quad (5)$$

If $\mathcal{I}(s^*) \neq \emptyset$, we choose the node x_{t_e} from $\mathcal{I}(s^*)$ which is closest to $x_t^{[s^*]}$:

$$x_{t_e} = \underset{x \in \mathcal{I}(s^*)}{\operatorname{argmin}} \operatorname{dist}_{m^{[s^*]}}(x_t^{[s^*]}, x). \quad (6)$$

In the sequel, x_{t_e} is denoted as the entry point at which the robot has the possibility to close a loop. t_e corresponds to the last time the robot was at the node x_{t_e} .

Note that it can happen that the particle s^* is far away from representing the correct pose estimate. In such a situation, the robot sometimes computes a path which cannot be traversed in practice. The robot will follow this path until it recognizes that it is blocked and then will abort the loop-closing behavior.

4.2. Stopping the loop-closing process

To determine whether or not the robot should activate the loop-closing behavior, our system constantly monitors the uncertainty $\mathcal{H}(t)$ about the robot's pose at the current time step. The necessary condition for starting the loop-closing process is the existence of an entry point x_{t_e} and that $\mathcal{H}(t)$ exceeds a given threshold. Once the loop-closing process has been activated, the robot approaches x_{t_e} and then follows the path taken after previously arriving at x_{t_e} . During this process, the uncertainty in the pose of the vehicle typically decreases because the robot is able to localize itself in the map built so far and unlikely particles vanish.

Furthermore, we have to define a criterion for deciding when the robot actually has to stop following a loop. A first attempt could be to introduce a threshold and to simply stop the trajectory-following behavior as soon as the uncertainty becomes smaller than a given threshold. This criterion, however, can be problematic, especially in the case of nested loops. Suppose the robot encounters the opportunity to close a loop that is nested within an outer and so far unclosed loop. If it eliminates all of its uncertainty by repeatedly traversing the inner loop, particles necessary to close the outer loop may vanish. As a result, the filter diverges and the robot fails to build a correct map (compare Fig. 6).

To remedy this so-called particle depletion problem [21], we introduce a constraint on the uncertainty of the robot. Let $\mathcal{H}(t_e)$ denote the uncertainty of the posterior when the robot visited the entry point last time. Then the new constraint allows the robot to re-traverse the loop only as long as its current uncertainty $\mathcal{H}(t)$ exceeds $\mathcal{H}(t_e)$. If the constraint is violated, the robot resumes its terrain acquisition (exploration) process. This constraint is designed to reduce the risk of depleting relevant particles during the loop-closing process. The idea is that by observing the area within the loop, the robot does not obtain any information about the world outside the loop. As a result, the robot cannot reduce the uncertainty $\mathcal{H}(t)$ in its current posterior below its uncertainty $\mathcal{H}(t_e)$ when entering the loop since $\mathcal{H}(t_e)$ is the uncertainty stemming from the world outside the loop.

To better illustrate the importance of this constraint, consider the following example. A robot moves from place A to place B and then repeatedly observes B.

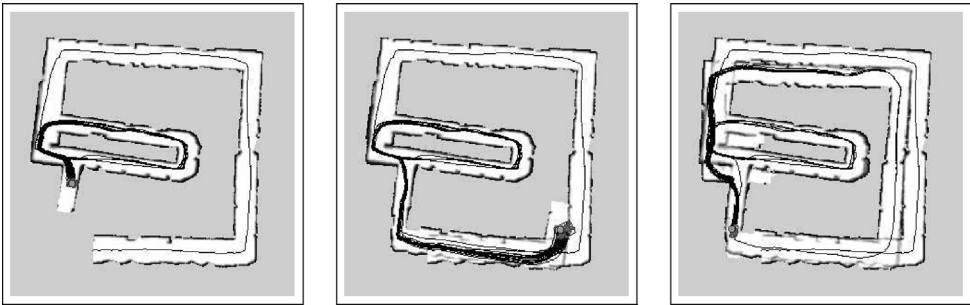


Figure 6. The particle depletion problem. A robot traveled through the inner loop several times (left image). After this, the diversity of hypotheses about the trajectory outside the inner loop had decreased too much (middle image) and the robot was unable to close the outer loop correctly (right image).

While it is mapping B, it does not get any further information about A. Since each particle represents a whole trajectory of the robot, hypotheses representing ambiguities about A will also vanish when reducing potential uncertainties about B. Our constraint reduces the risk of depleting particles representing ambiguities about A by aborting the loop-closing behavior at B as soon as the uncertainty drops below the uncertainty stemming from A.

Finally, we have to describe how we actually measure the uncertainty in the position estimate. The typical way of measuring the uncertainty of a posterior is to use the entropy. To compute the entropy of a posterior represented by particles, one typically uses a multi-dimensional grid representing the possible (discretized) states. Each cell a in this grid stores a probability which is given by the sum of the normalized weights of the samples corresponding to that cell. The entropy is then computed by summing up $-p(a) \log p(a)$ of each cell a in that grid.

In the case of multi-modal distributions, however, the entropy does not consider the distance between the different modes. As a result, a set of k different pose hypotheses which are located close by each other leads to the same entropy value as the situation in which k hypotheses are randomly distributed over the environment. The resulting maps, however, would look similar in the first case, but quite different in the second case. In our experiments, we figured out that we obtained better results if we use the volume expanded by the samples instead of the entropy of the posterior. We therefore calculated the pose uncertainty by determining the volume of the oriented bounding box around the particle cloud. A good approximation of the minimal oriented bounding box can be obtained efficiently by a principal component analysis.

Note that the loop-closing process is also aborted after a robot traveled for a long period of time in the same loop in order to avoid a (theoretically possible) endless loop-closing behavior. In all our experiments, however, this problem has never been encountered.

4.3. Reducing the exploration time

The experiments presented later on in this paper demonstrate that our uncertainty-based stopping criterion is an effective way to reduce the risk of particle depletion. However, it can happen that newly acquired sensor data acquired during loop-closing do not provide a lot of new information for the robot. Moving through such terrain leads to an increased exploration time. Therefore, it would be more efficient to abort the loop-closing in situations in which the new sensor data does not help to identify unlikely hypotheses.

To estimate how well the current set of N particle represents the true posterior, Liu [22] introduced the effective number of particles N_{eff} (also called effective sample size):

$$N_{\text{eff}}(t) = \frac{1}{\sum_{s=1}^N (\omega_t^{[s]})^2}. \quad (7)$$

The idea behind this measure is to determine the variance in the importance weights of the particles. Liu uses N_{eff} to resample in an intelligent way, but it is also very useful in the context of active loop-closing. We monitor the change of N_{eff} over time, which allows us to analyze how the newly acquired information affects the filter. If N_{eff} stays constant, the new information does not help to identify unlikely hypotheses represented by the individual particles. In that case, the variance in the importance weights of the particles does not change over time. If, in contrast, the value of N_{eff} decreases over time, the new information is used to identify that some particles are less likely than others. This is exactly the criterion we need to decide whether or not the loop-closing should be aborted in order to keep the exploration time small. As long as new information helps to identify unlikely particles, we follow the loop. As soon as the observations do not provide any new knowledge about the environment for a period of k time steps, we continue to explore new terrain.

Note that this criterion is optional and is not essential for a successful loop-closing strategy. It can directly be used if the underlying FastSLAM approach applies an adaptive resampling technique (if no adaptive resampling is used, one needs to monitor the relative change in N_{eff} after integrating each measurement, because after each resampling step the weights of all particles are set to $1/N$. As a result, N_{eff} is always equal to the number N of particles and the variance therefore is zero). More details on Rao-Blackwellized mapping using adaptive resampling have been published by Grisetti *et al.* [23]. In the experimental section of this paper, we will demonstrate that N_{eff} is a useful criterion in the context of active loop-closing and show how it behaves during exploration.

As long as the robot is localized well enough or no loop can be closed, we use a frontier-based exploration strategy [2] to choose target points for the robot. A frontier is any known and unoccupied cell that is an immediate neighbor of

an unknown, unexplored cell [5]. By extracting frontiers from a given grid map, one can easily determine potential target locations which lead the robot to so far unknown terrain. To select one of these frontier cells as the next goal location, a common way is to determine the travel cost to each frontier cell. This cost can be computed using Dijkstra's algorithm or a value iteration technique like done in Ref. [2]. It has been shown by Koenig and Tovey [3] that for a single robot this strategy yields reasonable short trajectories compared to the theoretically optimal solution. In our current system, we determine frontiers based on the map of the particle s^* .

A precise formulation of the loop-closing strategy is given by Algorithm 1. In our implementation, this algorithm runs as a background process that is able to interrupt the frontier-based exploration procedure.

```

1: Compute  $\mathcal{I}(s^*)$ 
2: if  $\mathcal{I}(s^*) \neq \emptyset$  then begin
3:    $\mathcal{H} \leftarrow \mathcal{H}(t_e)$ 
4:    $\text{path} \leftarrow x_t^{[s^*]} \cdot \text{shortest\_path}_{\mathcal{G}[s^*]}(x_{t_e}, x_t^{[s^*]})$ 
5:   while  $\mathcal{H}(t) > \mathcal{H} \wedge \text{var}(N_{\text{eff}}(n-k), \dots, N_{\text{eff}}(n)) > \epsilon$  do
6:     robot_follow(path)
7: end

```

Algorithm 1. The loop-closing algorithm

4.4. Handling multiple nested loops

Note that our loop-closing technique can also handle multiple nested loops. During the loop-closing process, the robot follows its previously taken trajectory to re-localize. It does not leave this trajectory until the termination criterion (see line 5 in Algorithm 1) is fulfilled. Therefore, it never starts a new loop-closing process before the current one is completed. A typical example with multiple nested loops is illustrated in Fig. 7. In the situation depicted in the left-hand picture, the robot starts with the loop-closing process for the inner loop. After completing the most inner loop, it moves to the second inner one and again starts the loop-closing process. Since our algorithm considers the uncertainty at the entry point, it keeps enough variance in the filter to also close the outer loop correctly. In general, the quality of the solution and whether or not the overall process succeeds depends on the number of particles used. Since determining this quantity is an open research problem, the number of particles has to be defined by the user in our current system.

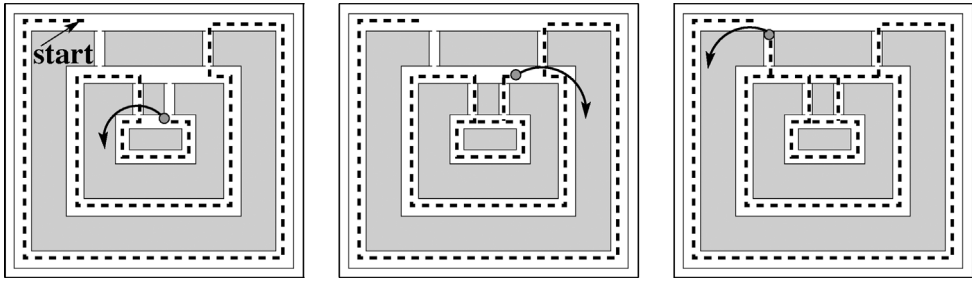


Figure 7. Active loop-closing of multiple nested loops.

5. EXPERIMENTS

Our approach has been implemented and evaluated in a series of real world and simulation experiments. For the real world experiments we used an iRobot B21r robot and an ActivMedia Pioneer II robot. Both are equipped with a SICK laser range finder. For the simulation experiments we used the real-time simulator of the Carnegie Mellon Robot Navigation Toolkit [24].

The experiments described in this section are designed to illustrate that our approach can be used to actively learn accurate maps of large indoor environments. Furthermore, they demonstrate that our integrated approach yields better results than an approach without active loop-closing. Additionally, we analyze how the active termination of the loop-closure influences the result of the mapping process.

5.1. Real-world exploration

The first experiment was carried out to illustrate that our current system can effectively control a mobile robot to actively close loops during exploration. To perform this experiment, we used a Pioneer II robot to explore the main lobby of the Department for Computer Science at the University of Freiburg. The size of this environment is 51×18 m. Figure 8 depicts the final result obtained by a completely autonomous exploration run using our active loop-closing technique. It also depicts the trajectory of the robot, which has an overall length of 280 m. The robot decided 4 times to re-enter a previously visited loop in order to reduce the uncertainty in its pose. Figure 8 also shows the corresponding entry points as well as the positions where the robot left the loops ('exit points'). In this experiment, the FastSLAM routine used 250 particles. As can be seen, the resulting map is quite accurate.

5.2. Active loop-closing versus frontier-based exploration

The second experiment should illustrate the difference in approaches that do not consider loop-closing actions. We used real robot data obtained with a B21r robot in the Sieg Hall at the University of Washington. As can be seen from the motivating example in the Introduction (Fig. 2), the robot traversed the loop twice during map building. To eliminate the influence of measurement noise and different movements

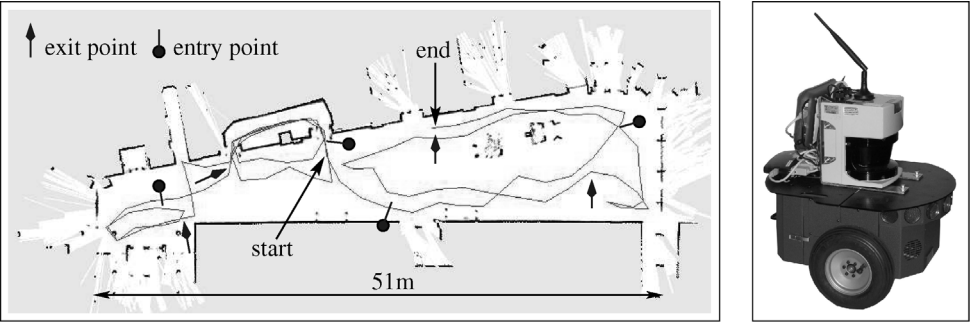


Figure 8. The left image shows the resulting map of an exploration experiment in the entrance hall of the Department for Computer Science at the University of Freiburg. It was carried out using a Pioneer II robot equipped with a laser range scanner (see right image). Also plotted is the path of the robot as well as entry and exit points where the robot started and stopped the active loop-closing process.

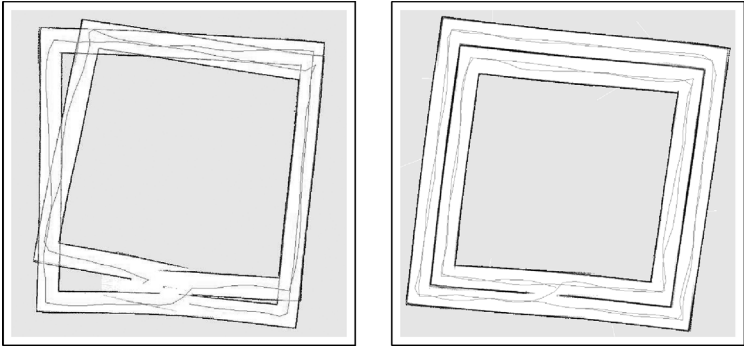


Figure 9. An environment with two large loops. The outer loop has a length of over 220 m. The left image show the resulting map of a trajectory in which the robot drove through the loops only once. In the second run, the robot visited every loop twice and obtained a highly accurate map (see right image).

of the robot, we removed the data corresponding to one loop traversal from the recorded data file and used this data as input to our FastSLAM algorithm. In this way, we simulated the behavior of a greedy exploration strategy which forces the robot to directly enter the corridor after returning to the starting location in the loop. As can be seen from the same figure, an approach that does not actively re-enter the loop fails to correctly estimate the angle between the loop and the corridor which should be oriented horizontally in that figure. Whereas the angular error was 7° with the standard approach, it was only 1° in the case where the robot traversed the loop twice. Both maps corresponded to the particle with the highest accumulated importance factor.

A further experiment that illustrates the advantage of place re-visiting is shown in Fig. 9. The environment used in this simulation run is 80×80 m and contains two large nested loops with nearly featureless corridors. The left image shows the result of the frontier-based approach which traversed each loop only once. Since the robot

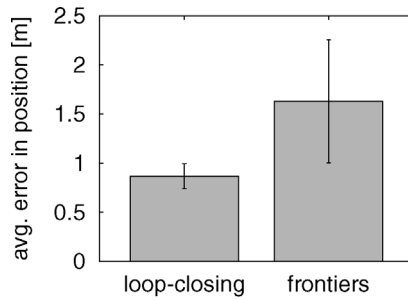


Figure 10. Comparison of our loop-closing strategy with a pure frontier-based exploration technique. The left bar in this graph plots the average error in the pose of the robot obtained with our loop-closing strategy. The right one shows the average error when a frontier-based approach was used. As can be seen, our technique significantly reduces the distances between the estimated positions and the ground truth (confidence intervals do not overlap).

is not able to correct the accumulated pose error, the resulting map contains large inconsistencies and two of the corridors are mapped onto each other. Our approach, in contrast, first re-visits the outer loop before entering the inner one (see right-hand image). Accordingly, the resulting map is quite accurate.

5.3. A quantitative analysis

To quantitatively evaluate the advantage of the loop-closing behavior, we performed a series of simulation experiments in an environment similar to Sieg Hall. We performed 20 experiments — 10 with active loop-closing and 10 without. After completing the exploration task, we measured the average error in the relative distances between positions lying on the resulting estimated trajectory and the ground truth provided by the simulator. The results are depicted in Fig. 10. As can be seen, the active loop-closing behavior significantly reduces the error in the position of the robot.

5.4. Importance of the termination criterion

In this experiment, we analyze the importance of the constraint that terminates the active loop-closing behavior as soon as the current uncertainty $\mathcal{H}(t)$ of the belief drops under the uncertainty $\mathcal{H}(t_e)$ of the posterior when the robot was at the entry point the last time.

In this simulated experiment, the robot had to explore an environment which contains two nested loops, as depicted in Fig. 11d. In the first case, we simply used a constant threshold to determine whether or not the loop-closing behavior should be stopped. In the second case, we applied the additional constraint that the uncertainty should not become smaller than $\mathcal{H}(t_e)$.

Figure 6 shows the map of the particle with the highest accumulated importance weight obtained with our algorithm using a constant threshold instead of consider-

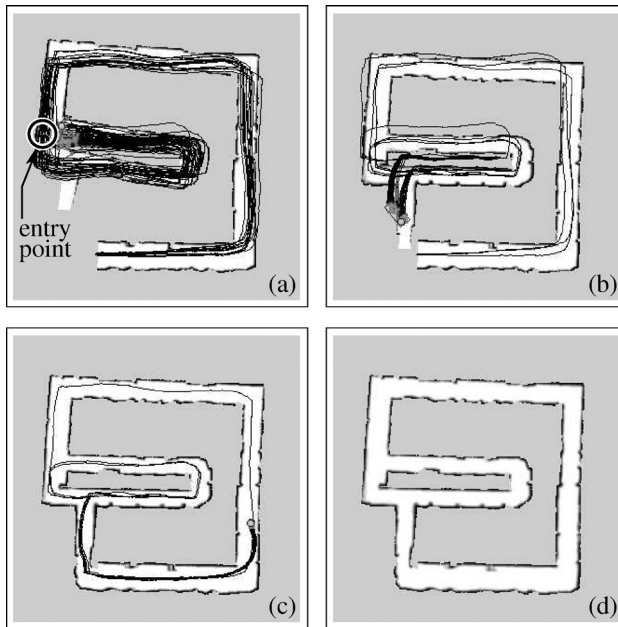


Figure 11. In image (a), the robot detected an opportunity to close a loop. It traversed parts of the inner loop as long as its uncertainty exceeds the uncertainty $\mathcal{H}(t_e)$ of the posterior when the robot was at the entry point and started the loop-closing process. The robot then turned back and left the loop (b) so that enough hypotheses survived to correctly close the outer loop (c) and (d). In contrast, a system considering only a constant threshold criterion fails to map the environment correctly, as depicted in Fig. 6.

ing $\mathcal{H}(t_e)$. In this case, the robot repeatedly traversed the inner loop (left image) until its uncertainty was reduced below a certain threshold. After three and a half rounds it decided to again explore unknown terrain, but the diversity of hypotheses had decreased too much (middle image). Accordingly the robot was unable to accurately close the outer loop (right image). We repeated this experiment several times and in none of the cases was the robot able to correctly map the environment. In contrast, our approach using the additional constraint always generated an accurate map. One example is shown in Fig. 11. Here, the robot stopped the loop-closing after traversing half of the inner loop. In both cases we used 80 particles.

As this experiment illustrates, the termination of the loop-closing is important for the convergence of the filter and to obtain accurate maps in environments with several (nested) loops. Note that similar results in principle can also be obtained without this termination constraint if the number of particles is dramatically increased. Since exploration is an online problem and each particle carries its own map, it is of utmost importance to keep the number of particles as small as possible. Therefore, our approach can also be regarded as a contribution to limit the number of particles during FastSLAM.

5.5. Evolution of N_{eff}

In this experiment, we demonstrate the behavior of the optional termination criterion that triggers the active loop-closing behavior. Additionally to the constraint that the uncertainty $\mathcal{H}(t)$ must be bigger than the uncertainty at the entry point $\mathcal{H}(t_e)$ of the loop, the process is stopped whenever the effective number of particles N_{eff} stays constant for a certain period of time. This criterion, which can be applied if the underlying mapping system uses an adaptive resampling technique [23], was introduced to avoid that the robot moves through the loop even if no new information can be obtained from the sensor data. The robot re-traverses the loop only as long as the sensor data is useful to identify unlikely hypotheses about maps and poses.

One typical evolution of N_{eff} is depicted in the left-hand image of Fig. 12. To achieve a good visualization of the evolution of N_{eff} , we processed a recorded data file using 150 particles. Due to the adaptive resampling strategy, only very few resampling operations were needed. The robot started at position A and in the first part of the experiment moved through unknown terrain (between the positions A and B). As can be seen, N_{eff} decreases over time. After the loop has been closed correctly and unlikely hypotheses had partly been removed by the resampling action (position B), the robot re-traversed the inner loop and N_{eff} stayed more or less constant. This indicates that acquiring further data in this area has only a very small effect on the relative likelihood of the particles and the system could not determine which hypotheses represented unlikely configurations. Therefore, in such a situation it makes more sense to focus on new terrain acquisition and to not continue the loop-closing process.

Furthermore, we analyzed the length of the trajectory traveled by the robot. Due to the active loop-closing, our technique generates longer trajectories compared to a purely frontier-based exploration strategy. We performed several experiments in different environments in which the robot had the opportunity to close loops and measured the average overhead. During our experiments, we observed an overhead

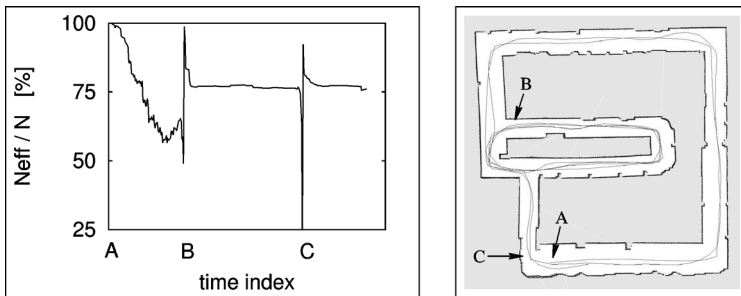


Figure 12. The graph plots the evolution of the N_{eff} function over time during an experiment carried out in the environment shown in the right image. The robot started at position A. Position B corresponds to the closure of the inner loop and position C corresponds to closure of the outer loop.

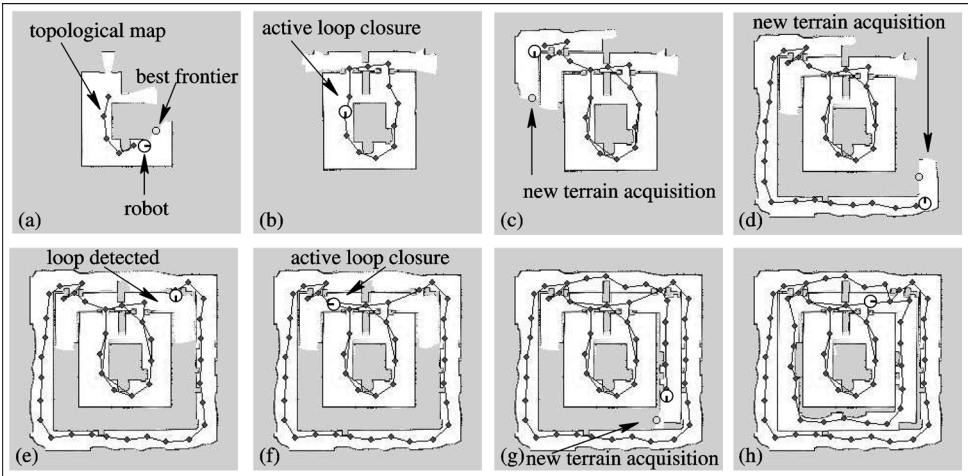


Figure 13. Snapshots during the exploration of a simulated environment with several nested loops. The red circles represent nodes of the topological map plotted on top of the most likely grid map. The yellow circle corresponds to the frontier cell the robot currently seeks to reach.

varying from 3 to 10%, but it obviously depends on the number of loops in the environment.

5.6. Multiple nested loops

To illustrate that our approach is able to deal with several nested loops, we performed a simulated experiment as shown in Fig. 13. The individual images in this figure depict eight snapshots recorded during exploration. Image (a) depicts the robot while exploring new terrain and image (b) while actively closing the most inner loop. After that, the robot focused on acquiring so far unknown terrain and maps the most outer loop as shown in (c) and (d). Then the robot detects a possibility to close a loop (e) and follows its previously taken trajectory (f). After aborting the loop-closing behavior, the robot again explores the loop in the middle (g), again closes the loop accurately and finishes the exploration task (h).

5.7. Computational resources

Note that our loop-closing approach needs only a few additional resources. To detect loops, we maintain an additional topological map for each particle. These topological maps are stored as a graph structure and for typical environments only a few kilobytes of extra memory are needed. To determine the distances based on the grid map in (3) and (4), our approach directly re-uses the result of a value iteration (alternatively Dijkstra’s algorithm) computed on the most likely grid map, which has already been computed in order to evaluate the frontier cells. Only the distance computation using the topological map needs to be done from scratch. However, since the number of nodes in the topological map is much smaller than

the number of grid cells, the computational overhead is comparably small. In our experiments, the time to perform all computations in order to decide where to move next increased by around 10 ms on a standard PC when activating the active loop-closing technique.

6. CONCLUSIONS

In this paper, we presented a novel approach for active loop-closing during autonomous exploration. We combined a Rao-Blackwellized particle filter for localization and mapping with a frontier-based exploration technique extended by the ability to actively close loops. Our algorithm forces the robot to re-traverse previously visited loops and in this way reduces the uncertainty in the pose estimate. As a result, we obtain more accurate maps compared to standard combinations of SLAM algorithms with exploration techniques. As fewer particles need to be maintained to build accurate maps, our approach can also be regarded as a contribution to limit the number of particles needed during FastSLAM.

Acknowledgements

This work has partly been supported by the German Science Foundation (DFG) under contract number SFB/TR-8 (project A3) and by the EC under contract FP6-004250-CoSy.

REFERENCES

1. A. A. Makarenko, S. B. Williams, F. Bourgoult and F. Durrant-Whyte, An experiment in integrated exploration, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne (2002).
2. W. Burgard, M. Moors, C. Stachniss and F. Schneider. Coordinated multi-robot exploration, *IEEE Trans. Robotics* (2005) (to appear).
3. S. Koenig and C. Tovey, Improved analysis of greedy mapping, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas, NV (2003).
4. G. Weiß, C. Wetzler and E. von Puttkamer, Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Munich, pp. 595–601 (1994).
5. B. Yamauchi, Frontier-based exploration using multiple robots, in: *Proc. 2nd Int. Conf. on Autonomous Agents*, Minneapolis, MN, pp. 47–53 (1998).
6. R. Grabowski, P. Khosla and H. Choset, Autonomous exploration via regions of interest, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas, NV (2003).
7. C. Stachniss and W. Burgard, Exploring unknown environments with mobile robots using coverage maps, in: *Proc. Int. Conf. on Artificial Intelligence*, Acapulco, pp. 1127–1132 (2003).
8. G. Dissanayake, H. Durrant-Whyte and T. Bailey, A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem, in: *Proc. ICRA '2000 Workshop on Mobile Robot Navigation and Mapping*, San Francisco, CA (2000).

9. A. Doucet, J. F. G. de Freitas, K. Murphy and S. Russel, Rao-blackwellized particle filtering for dynamic bayesian networks, in: *Proc. Conf. on Uncertainty in Artificial Intelligence*, Stanford, CA (2000).
10. A. Eliazar and R. Parr, DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, in: *Proc. Int. Conf. on Artificial Intelligence*, Acapulco (2003).
11. J.-S. Gutmann and K. Konolige, Incremental mapping of large cyclic environments, in: *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, Monterey, CA, pp. 318–325 (1999).
12. 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. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas, NV (2003).
13. M. Montemerlo, S. Thrun, D. Koller and B. Wegbreit, FastSLAM: a factored solution to simultaneous localization and mapping, in: *Proc. National Conf. on Artificial Intelligence*, Edmonton (2002).
14. K. Murphy, Bayesian map learning in dynamic environments, in: *Neural Info. Proc. Systems (NIPS)*, Denver, CO (1999).
15. S. Thrun, An online mapping algorithm for teams of mobile robots, *Int. J. Robotics Res.* (2001).
16. F. Bourgoult, A. A. Makarenko, S. B. Williams, B. Grocholsky and F. Durrant-Whyte, Information based adaptive robotic exploration, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne (2002).
17. H. Feder, J. Leonard and C. Smith, Adaptive mobile robot navigation and mapping, *Int. J. Robotics Res.* **18** (1999).
18. R. Sim, G. Dudek and N. Roy, Online control policy optimization for minimizing map uncertainty during exploration, in: *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, New Orleans, LA (2004).
19. F. Dellaert, D. Fox, W. Burgard and S. Thrun, Monte carlo localization for mobile robots, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Leuven (1998).
20. H. P. Moravec and A. E. Elfes, High resolution maps from wide angle sonar, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, St. Louis, MO, pp. 116–121 (1985).
21. R. van der Merwe, N. de Freitas, A. Doucet and E. Wan, The unscented particle filter, *Technical Report CUED/F-INFENG/TR380*, Cambridge University (2000).
22. J. S. Liu, Metropolized independent sampling with comparisons to rejection sampling and importance sampling, *Statist. Comput.* **6**, 113–119 (1996).
23. G. Grisetti, C. Stachniss and W. Burgard, Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling, in: *Proc. IEEE Int. Conf. on Robotics and Automation*, Barcelona, pp. 2443–2448 (2005).
24. N. Roy, M. Montemerlo and S. Thrun, Perspectives on standardization in mobile robot programming, in: *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas, NV (2003).

ABOUT THE AUTHORS



Cyrill Stachniss studied computer science at the University of Marburg and at the University of Freiburg. He received his MS degree in Computer Science in 2002 and is currently a PhD student in the research laboratory for Autonomous Intelligent Systems at the University of Freiburg. His research interests lie in the areas of mobile robot exploration, SLAM, as well as collision avoidance.



Dirk Hähnel studied Computer Science at the University of Bonn. He did his PhD at the Department of Computer Science at the University of Freiburg and is currently with Intel Research, Seattle. His areas of interest lie in mobile robot navigation, and especially in the learning of two-dimensional and three-dimensional maps.



Wolfram Burgard studied Computer Science at the University of Dortmund, Germany, and received his PhD degree from the Department of Computer Science of the University of Bonn in 1991. Since 1999, he has been Associate Professor at the Department of Computer Science of the University of Freiburg where he heads the Research Laboratory for Autonomous Intelligent Systems. His areas of interest lie in artificial intelligence and robotics. They cover mobile robot navigation, multi-robot systems, state estimation, human–robot interaction, activity monitoring and networked robots.



Giorgio Grisetti studied Computer Science at ‘La Sapienza’ University, Rome. He got his MS from the Engineering Faculty, in 2001 with summa cum laude. Currently, he is a PhD Student at ‘La Sapienza’ University of Rome in the Multi Robot Systems Laboratory. His research topics includes SLAM and multi-robot state estimation.