Best Conference Paper Award
IEEE International Conference on Robotics
and Automation, San Francisco, April 2000

A Real-Time Algorithm for Mobile Robot Mapping With Applications to Multi-Robot and 3D Mapping

Sebastian Thrun[1] Wolfram Burgard[2] Dieter Fox[1]

[1]Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

[2]Computer Science Department
University of Freiburg
Freiburg, Germany

Abstract

We present an incremental method for concurrent mapping and localization for mobile robots equipped with 2D laser range finders. The approach uses a fast implementation of scan-matching for mapping, paired with a sample-based probabilistic method for localization. Compact 3D maps are generated using a multi-resolution approach adopted from the computer graphics literature, fed by data from a dual laser system. Our approach builds 3D maps of large, cyclic environments in real-time. It is remarkably robust. Experimental results illustrate that accurate maps of large, cyclic environments can be generated even in the absence of any odometric data.

1 Introduction

Building maps of indoor environments is a pivotal problem in mobile robotics. The problem of mapping is often referred to as the *concurrent mapping and localization problem*, to indicate its chicken-and-egg nature: Building maps when a robot's locations are known is relatively straightforward, as early work by Moravec and Elfes has demonstrated more than a decade ago [10]. Conversely, localizing a robot when a map is readily available is also relatively well-understood, as a flurry of algorithms has successfully demonstrated [1]. In combination, however, the problem is hard.

Recent progress has led to a range of new methods. Most of these approaches build maps incrementally, by iterating localization and incremental mapping for each new sensor scan the robot receives [8, 13, 19, 20]. While these methods are fast and can well be applied in real-time, they typically fail when mapping large cyclic environments. This is because in environments with cycles, the robot's cumulative error can grow without bounds, and when closing the cycle error has to be corrected backwards in time (which most existing methods are incapable of doing). A recent family of probabilistic methods based on EM overcome this problem [15, 18]. EM searches the most likely map by simultaneously considering the locations of all past scans, using a probabilistic argument for iterative refinement during map construction. While these approaches have successfully mapped large cyclic environments, they are batch algorithms that cannot be run in real-time. Thus, a natural goal is to devise methods that combine the advantages of both methodologies, without giving up too much of the full power of EM so that large cycles can be mapped in real-time.

Most previous approaches also construct 2D maps only, and they only address single-robot mapping. Here our focus is also on multi-robot mapping and 3D maps.

This paper presents a novel algorithm that combines ideas from the EM approach, but nevertheless is strictly incremental. The basic idea is to combine the idea of posterior estimation—a key element of the EM-based approach—with that of incremental map construction using maximum likelihood estimators—a key element of previous incremental approaches. The result is an algorithm that can build large maps in environments with cycles, in real-time on a low-end computer. The posterior estimation approach makes it possible to integrate data collected my more than one robot, since it enables robots to globally localize themselves in maps built by other robots. We also extend our approach to the generation of 3D maps, where a multi-resolution algorithm is used to generate low-complexity 3D models of indoor environments.

In evaluations using a range of mobile robots, we found that our approach is extremely robust. Figure 1 shows some of the robots used in our experiments; not shown there are RWI B21 and Nomad Scout robots which we also used in evaluating our approach. Some of our robots have extremely poor odometry, such as the robot shown in Figure 1c. We show a collection of results obtained under a vast range of conditions, including cases where *no* odometry was available for mapping at all.

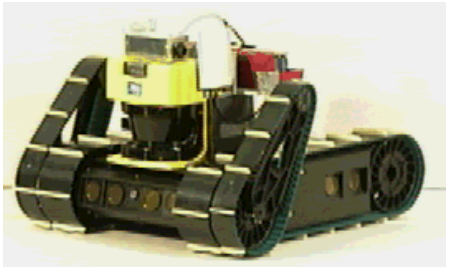
(a)  (b)  (c) 

Figure 1: Robots: (a) Pioneer robots used for multi-robot mapping. (b) Pioneer robot with 2 laser range finders used for 3D mapping. (c) Urban robot for indoor and outdoor exploration. The urban robot's odometry is extremely poor. All robots have been manufactured by RWI/ISR.

2 Mapping

At the most fundamental level, the problem of concurrent mapping and localization can be treated as a maximum likelihood estimation problem, in which one seeks to find the most likely map given the data. In this section, we will briefly restate the well-known map likelihood function, and then discuss a family of incremental on-line algorithms that approximate the maximum likelihood solution. Our approach is somewhat specific to robots equipped with 2D laser range finders or similar proximity sensors, which have become very popular in recent years. As we will show, our approach works well even in the complete absence of odometry data, and it also extends to the generation of 3D with laser range finders.

2.1 Likelihood Function

Let m be a map. Following the literature on laser scan matching [6, 9], we assume that a map is a collection of scans and their poses; the term pose refers to the x-y location relative to some hypothetical coordinate system and a scan's orientation θ. At time t, the map is written

$$m_t \quad = \quad \{\langle o_\tau, \hat{s}_\tau \rangle\}_{\tau=0,\ldots,t} \tag{1}$$

where o_τ denotes a laser scan and $\hat{s}_\tau$ its pose, and τ is a time index. Pictures of maps can be found pervasively throughout this paper.

The goal of mapping is to find the most likely map given the data, that is,

$$\operatorname*{argmax}_m P(m \mid d_t) \tag{2}$$

The data d_t is a sequence of laser range measurements and odometry readings:

$$d_t \quad = \quad \{s_0, a_0, s_1, a_1, \ldots, s_t\} \tag{3}$$

where s_τ denotes an observation (laser range scan), a_τ denotes an odometry reading, and t and τ are time indexes. Without loss of generality, we assume here that observations and odometry readings are alternated.

As shown in [18], the map likelihood function $P(m \mid d_t)$ can be transformed into the following product:

$$P(m \mid d_t) \quad = \quad \eta\, P(m) \int \cdots \int \prod_{\tau=0}^{t} P(o_\tau \mid m, s_\tau) \cdot \prod_{\tau=0}^{t-1} P(s_{\tau+1} \mid a_\tau, s_\tau)\, ds_1 \ldots ds_t \tag{4}$$

where η is a normalizer (which is irrelevant when computing (3)) and $P(m)$ is prior over maps which, if assumed to be uniform, can safely be omitted. Thus, the map likelihood is a function of two terms, the motion model, $P(s_{\tau+1} \mid a_\tau, s_\tau)$, and the perceptual model, $P(o_\tau \mid m_t, s_\tau)$. Since we can safely assume stationarity (i.e., neither model depends on the time index τ), we omit the time index and instead write $P(s \mid a, s')$ for the motion model and $P(o \mid m, s)$ for the perceptual model.

Throughout this paper, we adopt the probabilistic motion model shown in Figure 2. This figure depicts (projected into 2D) the probability of being at pose s, if the robot previously was at s' and executed action a. As can be seen, the shape of this conditional density resembles that of a banana. This distribution is obtained by the (obvious) kinematic equations, assuming that robot motion is noisy along its rotational and translational component.

The perceptual model is inherited from the rich literature on scan matching and projection filtering [6, 9]. Here the assumption is that when a robot receives a sensor scan, it is unlikely that future measurements perceive an obstacle within space previously perceived as free. The larger the distance between current and previous measurements, the lower the likelihood. This is illustrated in Figure 3. This figure shows a sensor scan (dots at the outside), along with the likelihood function (grayly shaded area): the darker a region, the smaller the likelihood of observing an obstacle. Notice that the likelihood function only applies a "penalty" to regions in the visual range of the scan; it is usually computed using ray-tracing.

A key feature of both models, the motion model and the perceptual model, is the fact that they are differentiable.

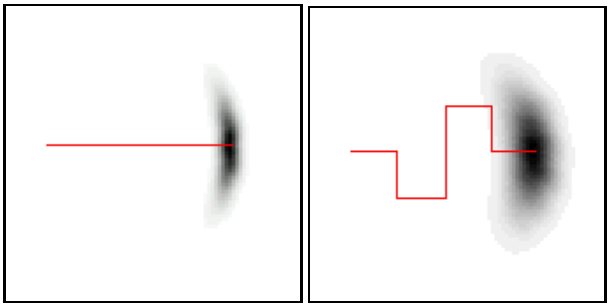


Figure 2: The motion model: shown here is the probability distribution $P(s \mid a, s')$ for the robot's posterior pose s after moving distance a beginning at location s', for two example trajectories.

More specifically, our approach uses the gradients

$$\frac{\partial P(s \mid a, s')}{\partial s} \qquad \frac{\partial P(o \mid m, s)}{\partial s} \tag{5}$$

for efficiently searching the most likely pose of a robot given its sensor measurements. The derivation of the equations is relatively straightforward (though tedious) and will not be given for brevity. However, in our implementation the gradient computation is carried out highly efficiently, enabling us to compute 1,000 or more gradients per second on a low-end PC. Further down, we will exploit the fact that the gradient can be computed so quickly, and use hill climbing for determining the most likely solution of mapping-related subproblems.

Let us briefly get back to the general likelihood function (4). Obviously, maximization of (4) yields the most likely map. Unfortunately, it is infeasible to maximize (4) in real-time, while the robot moves. This is because the likelihood cannot be maximized incrementally; instead, sensor measurements often carry information about past robot locations, which makes it necessary to revise past estimates as new information arrives. As noticed on [6, 9], this is most obvious when a robot "closes a loop," that is, when it traverses a cyclic environment. When closing a loop, the robot's error might be arbitrarily large, and generating a consistent map requires the correction of the map backwards in time. Therefore, past approaches, such as the EM algorithm presented in [15, 18], are off-line algorithms that may sometimes take multiple hours for computing the most likely map.

2.2 Conventional Incremental Mapping

Before describing our approach for incremental likelihood maximization, let us first consider a baseline approach, which is extremely popular in the literature. This approach is incremental, is attacks the concurrent localization and mapping problem, but it is unable to revise the map backwards the time and therefore is unable to handle cyclic environments (and close a loop). Nevertheless, it is used as a subcomponent in our algorithm which follows.

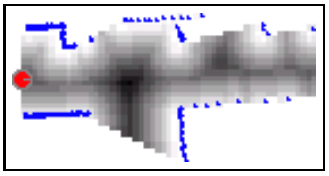


Figure 3: Likelihood function generated from a single sensor scan. The robot is on the left (circle), and the scan is depicted by 180 dots in front of the robot. The likelihood function is shown by the grey shading: the darker a region, the smaller the likelihood for sensing an object there. Notice that occluded regions are white (and hence incur no penalty).

The idea is simple, and probably because of its simplicity it is popular: *Given a scan and an odometry reading, determine the most likely pose. Then append the pose and the scan to the map, and freeze it once and forever.*

Mathematically, the most likely pose at time t is given by

$$\hat{s}_t \quad = \quad \operatorname*{argmax}_{s_t} P(s_t \mid o_t, a_{t-1}, \hat{s}_{t-1}) \tag{6}$$

which is usually determined using hill climbing (gradient ascent). The result of the search, $\hat{s}_t$ is then appended to the map along with the corresponding scan o_t:

$$m_{t+1} \quad = \quad m_t \cup \{\langle o_t, , \hat{s}_t \rangle\} \tag{7}$$

As noticed above, this approach typically works well in non-cyclic environments. When closing cycle, however, this approach suffers form two crucial shortcomings:

1. Pose errors can grow arbitrarily large. When closing the loop in a cyclic environment, search algorithms like gradient descent might fail to find the optimal solution.

2. When closing a loop in a cyclic environment, Past poses may have to be revised to generate a consistent map, which this approach is incapable of doing.

The brittleness of the approach, thus, is due to two factors: Past estimates are never revised, and only a single guess is maintained as to where the robot is, instead of a full distribution. Notice that neither of these restrictions applies to the EM family of mapping algorithms [15, 18].

2.3 Incremental Mapping Using Posteriors

Following the literature on probabilistic mapping with EM [15, 18] and the literature on Markov localization [16, 3], our approach computes the full *posterior* over robot poses, instead of the maximum likelihood pose only (as given in (6)). The posterior is a probability distribution over poses conditioned on past sensor data:

$$Bel(s_t) \quad = \quad P(s_t \mid d_t, m_{t-1}) \tag{8}$$

The short notation Bel indicates that the posterior is the robot's subjective belief as to where it might be. In past

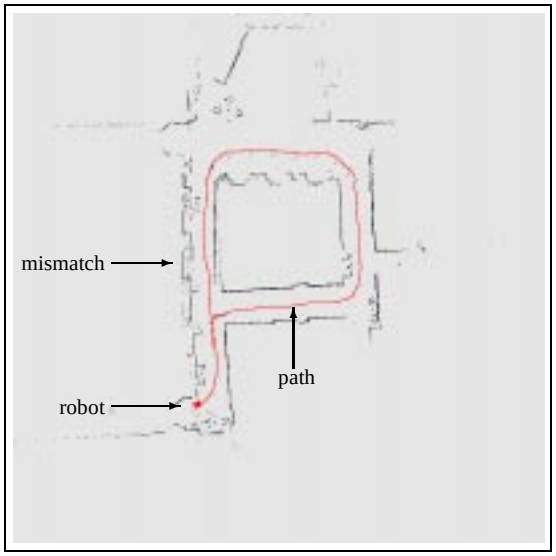


Figure 4: Result obtained using the most simple incremental approach, which is common in the literature. This approach fails to close the cycle and leads to inconsistent maps.

work, various researchers have found that algorithms that maintain a posterior estimation—instead of a single maximum likelihood guess—are typically more robust and hence lead to more scalable solutions.

The algorithm for computing the posterior is identical to the Markov localization algorithm [16]. It is incremental. Initially, at time $t = 0$, the belief $Bel(s_0)$ is centered on a (fictitious) origin of the coordinate system $\langle x = 0, y = 0, \theta = 0 \rangle$. Thus, $Bel(s_0)$ is initialized by a Dirac distribution (point distribution). Suppose at time t we know the previous belief $Bel(s_{t-1})$, which is distribution over poses s_{t-1} at time $t-1$, and we just executed action a_{t-1} and observed o_t. Then the new belief is obtained through:

$$\begin{aligned} & Bel(s_t) \\ &= \eta\, P(o_t \mid s_t, m_{t-1}) \int P(s_t \mid a_{t-1}, s_{t-1})\, Bel(s_{t-1})\, ds_{t-1} \end{aligned} \tag{9}$$

where η is (different) normalizer, and m_{t-1} is the best available map.

Update equation (9) is derived using the common Markov localization approach (see also [16, 3]), assuming a static map:

$$\begin{aligned} Bel(s_t) &= P(s_t \mid d_t, m_{t-1}) \\ &= P(s_t \mid d_{t-1}, a_{t-1}, o_t, m_{t-1}) \\ &= \eta\, P(o_t \mid s_t, d_{t-1}, a_{t-1}, m_{t-1}) \\ &\qquad P(s_t \mid d_{t-1}, a_{t-1}, m_{t-1}) \\ &= \eta\, P(o_t \mid s_t, m_{t-1}) \\ &\qquad \int P(s_t \mid d_{t-1}, a_{t-1}, s_{t-1}, m_{t-1}) \\ &\qquad P(s_{t-1} \mid d_{t-1}, a_{t-1}, m_{t-1})\, ds_{t-1} \end{aligned} \tag{10}$$

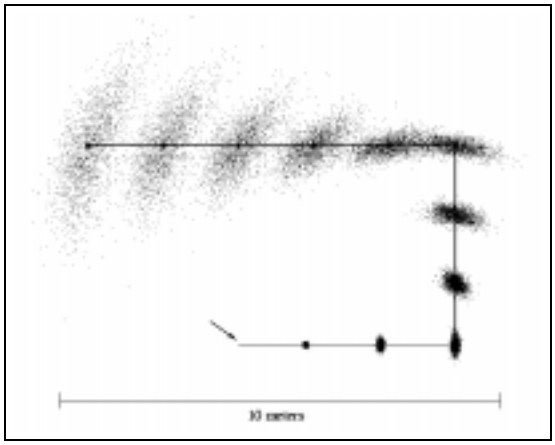


Figure 5: Sample-based approximation for the posterior $Bel(s)$. Here each density is represented by a set of samples, weighted by numerical importance factors. Particle filters are used to generate the sample sets.

$$\begin{aligned} &= \eta\, P(o_t \mid s_t, m_{t-1}) \\ &\qquad \int P(s_t \mid a_{t-1}, s_{t-1})\, Bel(s_{t-1})\, ds_{t-1} \end{aligned}$$

After computing the posterior, the new map is generated by maximizing a slightly different expression for pose estimation (c.f., $\hat{s}_t$ in Equation (6)):

$$\bar{s}_t = \operatorname*{argmax}_{s_t} Bel(s_t) \tag{11}$$

which leads to the new map

$$m_{t+1} = m_t \cup \{\langle o_t, , \bar{s}_t \rangle\} \tag{12}$$

Just as in (6) and (7), the map is grown by adding a scan at location $\bar{s}_t$. However, here we use the entire posterior $Bel(s_t)$ for determining the most likely pose, not just the most recent sensor scan and its pose (as in (6)). As a result, the increasing diameter of uncertainty is modeled so that, when closing the loop, the correct uncertainty can be taken into account: the larger the loop, the wider the margin of uncertainty. This difference is important when mapping large cyclic environments—where a robot needs to know where to search when closing a loop.

Our approach uses *samples* to approximate the posterior. Figure 5 shows an example, for a sequence of robot poses along a U-shaped trajectory. Here each of the sample sets is an approximation of densities (of the type shown in Figure 2). The idea of using samples goes back to Rubin's importance sampler [14] and, in the context of temporal posterior estimation is known as *particle filters* [12]. It has, with great success, been applied for tracking in computer vision [7] and mobile robot localization [2, 3]. As argued in the statistical literature, this representation can approximate almost arbitrary posteriors at a convergence rate of $\frac{1}{\sqrt{N}}$ [17]. It is convenient for robotics, since it is easy to

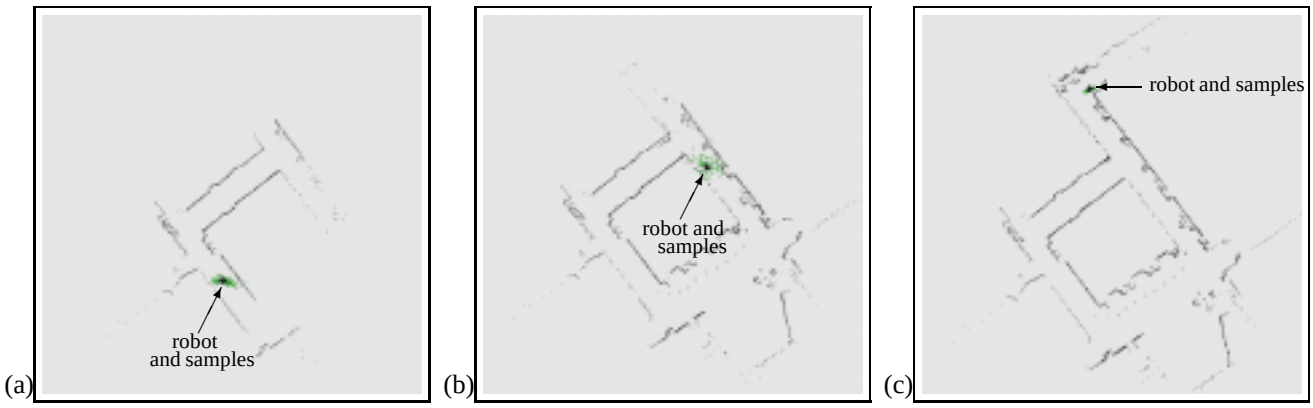


Figure 6: Incremental algorithm for concurrent mapping and localization. The dots, centered around the robot, indicate the posterior belief which grows over time (a and b). When the cycle is closed as in (c), the posterior becomes small again.

implement, and both more efficient and more general than most alternatives [3].

The sample-based representation directly facilitates the optimization of (11) using gradient descent. Our approach performs gradient descent using each sample as a starting point, then computes the goodness of the result using the obvious likelihood function. If the samples are spaced reasonably densely (which is easily done with only a few dozen samples), one can guarantee that the global maximum of the likelihood function can be found. This differs from the simple-minded approach above, where only a single starting pose is used for hill-climbing search, and which hence might fail to produce the global maximum (and hence the best map).

2.4 Backwards Correction

As argued above, when closing cycles it is imperative that maps are adjusted backwards in time. The amount of backwards correction is given by the difference (denoted Δ_{s_t}):

$$\Delta_{s_t} \quad = \quad \bar{s}_t - \hat{s}_t \tag{13}$$

where $\bar{s}_t$ is computed according to Equation (11) and $\hat{s}_t$ is obtained through Equation (6). This expression is the difference between the *incremental* best guess (c.f., our baseline approach) and the best guess using the full posterior.

If $\Delta_{s_t} = 0$, which is typically the case when *not* closing a loop, no backwards correction has to take place. When $\Delta_{s_t} \neq 0$, however, a shift occurred due to reconnection with a previously mapped area, and poses have to be revised backwards in time.

Our approach does this in three steps:

1. First, the size of the loop is determined by determining the scan in the map which led to the adjustment (this is a trivial side-result in the posterior computation).

2. Second, the error Δ_{s_t} is distributed *proportionally* among all poses in the loop. This computation does *not* yield a maximum likelihood match; however, it places the intermediate poses in a good starting position for subsequent gradient descent search.

3. Finally, gradient descent search is applied iteratively for all poses inside the loop, until the map is maximally consistent (maximizes likelihood) under this new constraint arising from the cycle.

These three steps implement an efficient approximation to the maximum likelihood estimator for the entire loop. Our approach has been found to be extremely robust in practice (we never obtained a single wrong result) and also extremely fast (the entire maximization can be carried out between two sensor measurements for all experiments reported in this paper).

2.5 Multi-Robot Mapping

The posterior estimation component of our approach makes it straightforward to generate maps with multiple robots. Here we assume that the initial pose of the robots relative to each other is unknown; however, we make the important restrictive assumption that each robot *starts within the map of a specific robot*, called the team leader. To generate a single, unified map, each robot must localize itself in the map of the team leader.

The insight for multi-robot mapping is closely tight to the notion of posterior estimation. As the reader might have noticed, our posterior computation is equivalent to *Monte Carlo localization* [2, 3], a version of Markov localization capable of performing *global localization*. Thus, to localize a robot in the map of the team leader, its initial samples are distributed uniformly across the team leader's map, as shown in Figure 7a. The posterior estimation then quickly localizes the robot (see [2, 3]), which then enables both robots to build a single, uniform map.

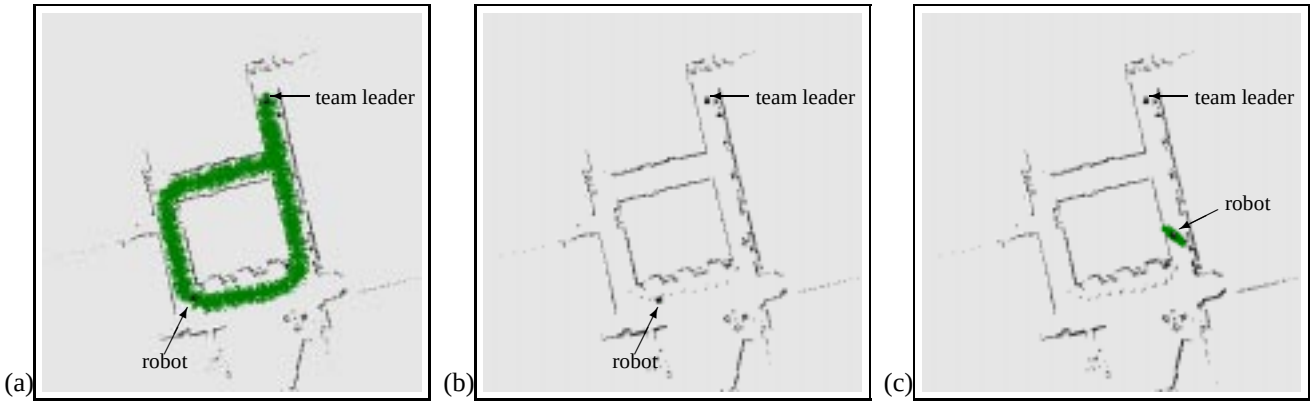


Figure 7: A second robot localizes itself in the map of the first, then contributes to building a single unified map. In (a), the initial uncertainty of the relative pose is expressed by a uniform sample in the existing map. The robot on the left found its pose in (b), and then maintains a sense of location in (c).

2.6 3D Mapping

Finally, a key goal of this research is to generate accurate 3D maps. With accurate localization in place, this extension is obtained using a robot equipped with two laser range finder, such as the one shown in Figure 1b. Here the forward-looking laser is used for concurrent mapping and localization in 2D, and the upward-pointed laser is used to build a 3D map of the environment.

At first glance, one might simply generate 3D maps by connecting nearby laser measurements into small polygons. However, this approach has two disadvantages: First, it is prone to noise, and second, the number of measurements are excessively large, and the resulting maps are therefore overly complex.

Our approach filters our outliers based on distance; if two measurements are further than a factor 2 apart from the expected measurement under the robot motion (and assuming the robot faces a straight wall), this measurement does not become part of a polygone. We also apply a simplification algorithm [4, 5], previously developed to simplify polygonal models for real-time rendering in computer graphics.[1] In a nutshell, this approach iteratively simplifies multi-polygon surface models by fusing polygons that look similar when rendered. The result is a model with much reduced complexity, which nevertheless is similarly accurate and looks about as good as the original when rendered. The simplification uses only a small fraction of the available time, hence is easily applied in real-time.

3 Results

3.1 Mapping A Cycle

In our experiments, scans were only appended to the map when the robot moved a presprecified distance (2 meters); all scans, however, were used in localization. This kept the complexity of maps manageable (a few hundred scans, instead of several thousand). Also, to make the mapping problem more challenging, we occasionally introduced random error into the odometry (30 degrees or 1 meter shifts).

Figure 6 shows results obtained using the same data set as in Figure 4. Here the robot traverses the cycle, but it also keeps track of its posterior belief $Bel(s)$ represented by samples, as shown by the dots centered around the maximum likelihood pose in Figure 6. When the cycle is closed (Figure 6b), the robot's error is significant; however, the "true" pose is well within its posterior at this time. Our approach quickly identifies the true pose, corrects past beliefs, and reduces the robot's uncertainty accordingly. The final map is shown in Figure 6c. As can be seen there, the map is highly accurate and the error in the cycle has been eliminated. All these results have been obtained in real-time on a low-end PC.

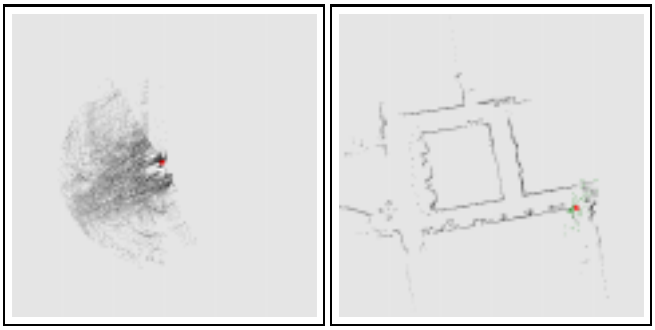


Figure 8: Mapping without odometry. Left: Raw data, right: map, generated on-line in real-time.

3.2 Mapping Without Odometry

To test the robustness of the approach, we attempted to build the same map but in the *absence of odometry data*. The raw data, stripped of the odometry information, is shown in Figure 8a. Obviously, these data are difficult to interpret (we are not aware of an algorithm for mapping

[1] We gratefully acknowledge Michael Garland's assistance in using the software.

Figure 9: Autonomous exploration and mapping using the urban robot: Raw data and final map, generated in teal-time during exploration.

that works well without odometry information). Figure 8b shows the resulting map. This map has been generated using identical software settings. When traversing the cycle, this map is less accurate than the one generated with odometry data. However, when closing the cycle, these errors are identified and eliminated, and the final result is optically equivalent to the one with odometry. The reader should notice, however, that the odometry-free results are only possible because the environment possess sufficient variation. In a long, featureless corridor, our approach would clearly fail in the absence of odometry data. Nevertheless, these results illustrate the robustness of our approach.

3.3 Autonomous Exploration

Figure 9 shows results of an autonomously exploring robot, obtained at a recent DARPA demonstration. These results were obtained using the Urban Robot shown in Figure 1c. This robot is able to traverse extremely rugged terrain. However, its skid-steering mechanism using tracks is extremely erroneous, and odometry errors are often as large as 100%, depending on the conditions of the floor.

Figure 9 shows the raw data (left diagram) and the map (right diagram) along with the robot's trajectory. This is the worst map ever generates using our approach: some of the walls are not aligned perfectly, but instead are rotated by approximately 2 degrees. However, given the extremely poor odometry of the robot, and the fact that our approach does not resort to assumptions like recto-orthogonal walls, these results are actually surprisingly accurate. The map is clearly sufficient for navigation. We suspect if the environment possessed a loop, the final result would be more accurate.

3.4 Multi-Robot Mapping

Figure 7 illustrates our approach to multi-robot mapping. Here a second robot localizes itself in the map built by the team leader (left diagram), as explained above. After a short motion segment, the posterior is focused on a single location (center diagram) and the incoming sensor data is now used to further the map. The right diagram shows the

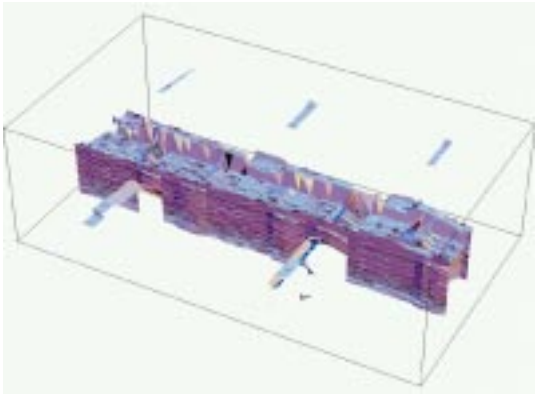


Figure 10: 3D Map.

situation after a few more seconds, Here the second robot has progressed through the map of the team leader. It still knows its position with high accuracy.

3.5 3D Mapping

Results for 3D mapping are shown in Figures 10 and 11. Figure 10 shows a short corridor segment. The free-flying surfaces are ceiling regions inside offices, which the robot's lasers sensed while moving through the corridor.

Figure 11 shows a sequence of rendered views of a larger (cyclic) map, which is approximately 60 meters long. The rendering algorithm is a standard virtual reality tool (VRweb), which enables the user to remotely inspect the building by "flying through the map." The top rowin Figure 11 has been generated from raw laser data; this model contains 82,899 polygons. The bottom row is the simplified polygonal model, which contains only 8,289 polygons. The appearance of both is similar; however, rendering the more compact one is an order of magnitude faster.

4 Discussion

This paper presented a new, online method for robust mapping and localization in indoor environments. The approach combines ideas from incremental mapping (maximum likelihood, incremental map construction) with ideas of more powerful, non-incremental approaches (posterior estimation, backwards correction). The result is a fast and robust algorithm for real-time mapping of indoor environments, which extends to multi-robot mapping and mapping in 3D. A fast algorithm was employed to generate compact 3D models of indoor environments.

Experimental results illustrated that large-scale environments can be mapped in real-time. The resulting maps were highly accurate. We also demonstrated that our approach can generate 3D maps, and it can fuse information collected though multiple robot platforms.

When compared to EM, the ability to generate maps in real

Figure 11: Views of the 3D map, for the high-res model (top row) and the lower resolution model (bottom row).

time comes at a price of increased brittleness. EM is a principled approach to finding the best map, which simultaneously revises beliefs arbitrarily far in the past—which makes it necessarily inapplicable in real-time. However, our approach inherits some of the power of EM by using posterior estimations and a fast mechanisms for backwards revision, specifically tuned for mapping cyclic environments. As a result, our approach can handle a large range of environments in real-time.

Nevertheless, our approach surpasses previous incremental approaches in robustness, specifically in environments with cycles. Our results make us confident that the approach is very robust to errors, in particular odometry errors.

Acknowledgments

We thank Michael Garland for his assistance with the polygon fusing software, Todd Pack and RWI/ISR for their superb support with the robot hardware, and the members of CMU's Robot Learning Lab for many fruitful discussions. The idea to build 3D maps was brought to our attention by our DARPA-PM LTC John Blitch, which we gratefully acknowledge.

This research is sponsored in part by DARPA via TACOM (contract number DAAE07-98-C-L032) and Rome Labs (contract number F30602-98-2-0137), and by the National Science Foundation (regular grant number IIS-9877033 and CAREER grant number IIS-9876136), which is gratefully acknowledged.

References

[1] J. Borenstein, B. Everett, and L. Feng. *Navigating Mobile Robots: Systems and Techniques.* A. K. Peters, 1996.

[2] F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *CVPR-99*.

[3] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *AAAI-99*.

[4] M. Garland and P. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH-97*.

[5] M. Garland and P. Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *IEEE Visualization-98*.

[6] J.-S. Gutmann and C. Schlegel. Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *EUROMICRO-96*.

[7] M. Isard and A. Blake. Condensation: conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1998.

[8] J.J. Leonard, H.F. Durrant-Whyte, and I.J. Cox. Dynamic map building for an autonomous mobile robot. *International Journal of Robotics Research*, 11(4), 1992.

[9] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotic Systems*, 1998.

[10] H. P. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *ICRA-85*..

[11] H.P. Moravec and M.C. Martin. Robot navigation by 3D spatial evidence grids. Internal Report, CMU, 1994.

[12] M. Pitt and N. Shephard. Filtering via simulation: auxiliary particle filter. *Journal of the American Statistical Association*, 1999.

[13] W.D. Rencken. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *IROS-93*.

[14] D.B. Rubin. Using the SIR algorithm to simulate posterior distributions. In *Bayesian Statistics 3*, 1988.

[15] H. Shatkay. *Learning Models for Robot Navigation*. PhD thesis, CSD, Brown University, 1998.

[16] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *IJCAI-95*.

[17] M.A. Tanner. *Tools for Statistical Inference*. Springer, 1993.

[18] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31, 1998.

[19] B. Yamauchi and R. Beer. Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 1996.

[20] B. Yamauchi, P. Langley, A.C. Schultz, J. Grefenstette, and W. Adams. Magellan: An integrated adaptive architecture for mobile robots. TR 98-2, ISLE, Palo Alto, 1998.