In: Proceedings of the Thirteenth National Conference on Artificial Intelligence
AAAI, Portland, Oregon, August 1996

Integrating Grid-Based and Topological Maps for Mobile Robot Navigation

Sebastian Thrun[†‡]
[†]Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

Arno Bücken[‡]
[‡]Institut für Informatik
Universität Bonn
D-53117 Bonn, Germany

Abstract

Research on mobile robot navigation has produced two major paradigms for mapping indoor environments: grid-based and topological. While grid-based methods produce accurate metric maps, their complexity often prohibits efficient planning and problem solving in large-scale indoor environments. Topological maps, on the other hand, can be used much more efficiently, yet accurate and consistent topological maps are considerably difficult to learn in large-scale environments.

This paper describes an approach that integrates both paradigms: grid-based and topological. Grid-based maps are learned using artificial neural networks and Bayesian integration. Topological maps are generated on top of the grid-based maps, by partitioning the latter into coherent regions. By combining both paradigms—grid-based and topological—, the approach presented here gains the best of both worlds: accuracy/consistency and efficiency. The paper gives results for autonomously operating a mobile robot equipped with sonar sensors in populated multi-room environments.

Introduction

To efficiently carry out complex missions in indoor environments, autonomous mobile robots must be able to acquire and maintain models of their environments. The task of acquiring models is difficult and far from being solved. The following factors impose practical limitations on a robot's ability to learn and use accurate models:

1. **Sensors.** Sensors often are not capable to directly measure the quantity of interest (such as the exact location of obstacles).
2. **Perceptual limitations.** The perceptual range of most sensors is limited to a small range close to the robot. To acquire global information, the robot has to actively explore its environment.
3. **Sensor noise.** Sensor measurements are typically corrupted by noise, the distribution of which is often unknown (it is rarely Gaussian).
4. **Drift/slippage.** Robot motion is inaccurate. Odometric errors accumulate over time.
5. **Complexity and dynamics.** Robot environments are complex and dynamic, making it principally impossible to maintain exact models.
6. **Real-time requirements.** Time requirements often demand that the internal model must be simple and easily accessible. For example, fine-grain CAD models are often disadvantageous if actions must be generated in real-time.

Recent research has produced two fundamental paradigms for modeling indoor robot environments: the *grid-based (metric) paradigm* and the *topological paradigm*. Grid-based approaches, such as those proposed by Moravec/Elfes (Moravec 1988) and many others, represent environments by evenly-spaced grids. Each grid cell may, for example, indicate the presence of an obstacle in the corresponding region of the environment. Topological approaches, such a those described in (Engelson & McDermott 1992; Kortenkamp & Weymouth 1994; Kuipers & Byun 1990; Matarić 1994; Pierce & Kuipers 1994), represent robot environments by graphs. Nodes in such graphs correspond to distinct situations, places, or landmarks (such as doorways). They are connected by arcs if there exists a direct path between them.

Both approaches to robot mapping exhibit orthogonal strengths and weaknesses. Occupancy grids are considerably easy to construct and to maintain even in large-scale environments (Buhmann *et al.* 1995; Thrun & Bücken 1996). Since the intrinsic geometry of a grid corresponds directly to the geometry of the environment, the robot's position within its model can be determined by its position and orientation in the real world—which, as shown below, can be determined sufficiently accurately using only sonar sensors, in environments of moderate size. As a pleasing consequence, different positions for which sensors measure the same values (*i.e.*, situations that look alike) are naturally disambiguated in grid-based approaches. This is not the case for topological approaches, which determine the position of the robot relative to the model based on landmarks or distinct sensory features. For example, if the robot traverses two places that look alike, topological approaches often have difficulty determining if these places are the same or not (particularly if these places have been reached via different paths). Also, since sensory input usually depends strongly on the view-point of the robot, topological approaches may fail to recognize geometrically nearby places.

On the other hand, grid-based approaches suffer from their enormous space and time complexity. This is because the resolution of a grid must be fine enough to capture every important detail of the world. Compactness in a key advantage of topological representations. Topological maps are usually more compact, since their resolution is determined by the complexity of the environment. Consequently, they permit fast planning, facilitate interfacing to symbolic planners and problem-solvers, and provide more natural interfaces for human instructions. Since topological approaches usually

Grid-based approaches	Topological approaches
+ easy to build, represent, and maintain + recognition of places (based on geometry) is non-ambiguous and view point-independent + facilitates computation of shortest paths	+ permits efficient planning, low space complexity (resolution depends on the complexity of the environment) + does not require accurate determination of the robot's position + convenient representation for symbolic planners, problem solvers, natural language interfaces
– planning inefficient, space-consuming (resolution does not depend on the complexity of the environment) – requires accurate determination of the robot's position – poor interface for most symbolic problem solvers	– difficult to construct and maintain in larger environments – recognition of places (based on landmarks) often ambiguous, sensitive to the point of view – may yield suboptimal paths

Table 1: Comparison of grid-based and topological approaches to map building.

do not require the exact determination of the geometric position of the robot, they often recover better from drift and slippage—phenomena that must constantly be monitored and compensated in grid-based approaches. To summarize, both paradigms have orthogonal strengths and weaknesses, which are summarized in Table 1.

This paper advocates to integrate both paradigms, to gain the best of both worlds. The approach presented here combines both grid-based (metric) and topological representations. To construct a grid-based model of the environment, sensor values are interpreted by an artificial neural network and mapped into probabilities for occupancy. Multiple interpretations are integrated over time using Bayes' rule. On top of the grid representation, more compact topological maps are generated by splitting the metric map into coherent regions, separated through *critical lines*. Critical lines correspond to narrow passages such as doorways. By partitioning the metric map into a small number of regions, the number of topological entities is several orders of magnitude smaller than the number of cells in the grid representation. Therefore, the integration of both representations has unique advantages that cannot be found for either approach in isolation: the grid-based representation, which is considerably easy to construct and maintain in environments of moderate complexity (e.g., 20 by 30 meters), models the world consistently and disambiguates different positions. The topological representation, which is grounded in the metric representation, facilitates fast planning and problem solving.

The robots used in our research are shown in Figure 1. All robots are equipped with an array of 24 sonar sensors. Throughout this paper, we will restrict ourselves to the interpretation of sonar sensors, although the methods described here have (in a prototype version) also been operated using cameras and infrared light sensors in addition to sonar sensors, using the image segmentation approach described in (Buhmann *et al.* 1995). The approach proposed here has extensively been tested in various indoor environments, and is now distributed commercially by a leading mobile robot manufacturer (Real World Interface, Inc.) as part of the regular navigation software.

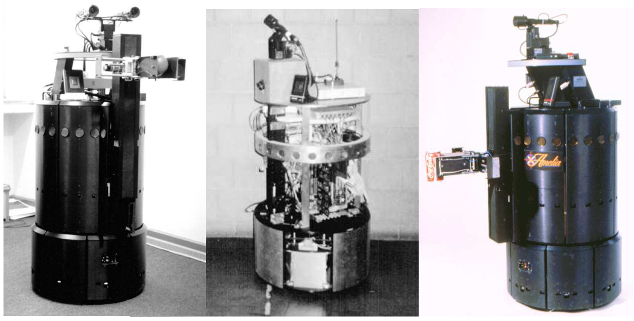


Figure 1: The robots used in our research: RHINO (University of Bonn), XAVIER, and AMELIA (both CMU).

Grid-Based Maps

The metric maps considered here are two-dimensional, discrete occupancy grids, as originally proposed in (Elfes 1987; Moravec 1988) and since implemented successfully in various systems. Each grid-cell $\langle x, y\rangle$ in the map has an *occupancy value* attached, which measures the subjective belief whether or not the center of the robot can be moved to the center of that cell (*i.e.*, the occupancy map models the *configuration space* of the robot, see *e.g.*, (Latombe 1991)). This section describes the four major components of our approach to building grid-based maps (see also (Thrun 1993)): (1) *sensor interpretation*, (2) *integration*, (3) *position estimation*, and (4) *exploration*. Examples of metric maps are shown in various places in this paper.

Sensor Interpretation

To build metric maps, sensor reading must be "translated" into occupancy values $occ_{x,y}$ for each grid cell $\langle x, y\rangle$. The idea here is to train an artificial neural network using Back-Propagation to map sonar measurements to occupancy values. The input to the network consists of the four sensor readings closest to $\langle x, y\rangle$, along with two values that encode $\langle x, y\rangle$ in polar coordinates relative to the robot (angle to the first of the four sensors, and distance). The output target for the network is 1, if $\langle x, y\rangle$ is occupied, and 0 otherwise. Training examples can be obtained by operating a robot in a known environment and recording its sensor readings; notice that each sonar scan can be used to construct many training examples for different x-y coordinates. In our implementation, training examples are generated with a mobile robot simulator.

Figure 2 shows three examples of sonar scans along with their neural network interpretation. The darker a value in the circular region around the robot, the larger the occupancy value computed by the network. Figures 2a&b depict situations in a corridor. Situations such as the one shown in Figure 2c—that defy simple interpretation—are typical for cluttered indoor environments.

Integration Over Time

Sonar interpretations must be integrated over time, to yield a single, consistent map. To do so, it is convenient to interpret

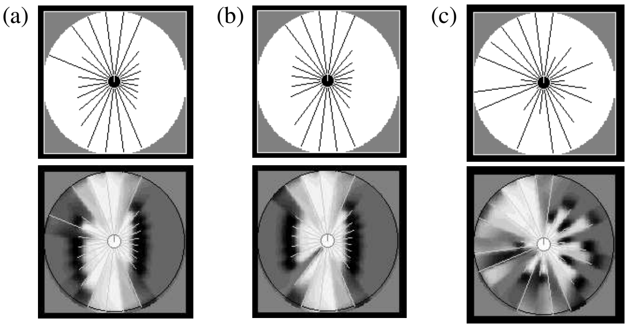


Figure 2: Sensor interpretation: Three example sonar scans (top row) and local occupancy maps (bottom row), generated by the neural network.

the network's output for the t-th sensor reading (denoted by $s^{(t)}$) as the *probability* that a grid cell $\langle x, y \rangle$ is occupied, conditioned on the sensor reading $s^{(t)}$:

$$Pr(occ_{x,y} | s^{(t)})$$

A map is obtained by integrating these probabilities for all available sensor readings, denoted by $s^{(1)}, s^{(2)}, \ldots, s^{(T)}$. In other words, the desired occupancy value for each grid call$\langle x, y \rangle$ can be expressed as the probability

$$Pr(occ_{x,y} | s^{(1)}, s^{(2)}, \ldots, s^{(T)}),$$

which is conditioned on *all* sensor reading. A straightforward approach to estimating this quantity is to apply Bayes' rule (Moravec 1988; Pearl 1988). To do so, one has to assume independence of the noise in different readings. More specifically, given the true occupancy of a grid cell $\langle x, y \rangle$, the conditional probability $Pr(s^{(t)} | occ_{x,y})$ must be assumed to be independent of $Pr(s^{(t')} | occ_{x,y})$ for $t \neq t'$. This assumption is not implausible—in fact, it is commonly made in approaches to building occupancy grids. The desired probability can now be computed as follows:

$$Pr(occ_{x,y} | s^{(1)}, s^{(2)}, \ldots, s^{(T)}) =$$
$$1 - \left(1 + \frac{Pr(x)}{1 - Pr(x)} \prod_{\tau=1}^{T} \frac{Pr(occ_{x,y} | s^{(\tau)})}{1 - Pr(occ_{x,y} | s^{(\tau)})} \frac{1 - Pr(x)}{Pr(x)} \right)^{-1}$$

Here $Pr(x)$ denotes the prior probability for occupancy (which, if set to 0.5, can be omitted in this equation). Notice that this formula can be used to update occupancy values *incrementally*. An example map of a competition ring constructed at the 1994 AAAI autonomous robot competition is shown in Figure 3.

Position Estimation

The accuracy of the metric map depends crucially on the alignment of the robot with its map. Unfortunately, slippage and drift can have devastating effects on the estimation of the robot position. Identifying and correcting for slippage and drift is therefore imperative for grid-based approaches to robot navigation (Feng, Borenstein, & Everett 1994; Rencken 1993).

Figure 4 gives an example that illustrates the importance of position estimation in grid-based robot mapping. In Figure 4a, the position is determined solely based on dead-

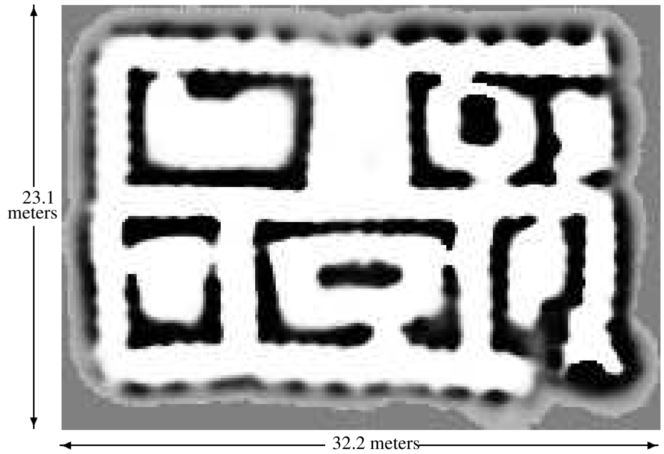
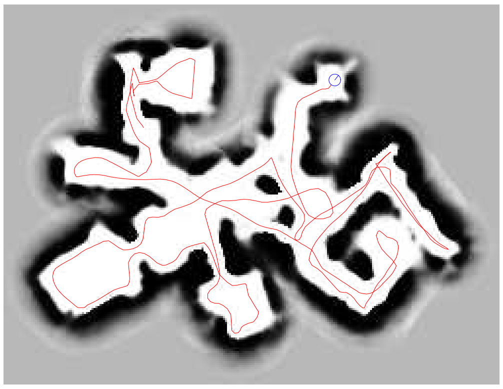


Figure 3: Grid-based map, constructed at the 1994 AAAI autonomous mobile robot competition.

reckoning. After approximately 15 minutes of robot operation, the position error is approximately 11.5 meters. Obviously, the resulting map is too erroneous to be of practical use. Figure 4b is the result of exploiting and integrating three sources of information:

1. **Wheel encoders.** Wheel encoders measure the revolution of the robot's wheels. Based on their measurements, odometry yields an estimate of the robot's position at any point in time. Odometry is very accurate over short time intervals.
2. **Map correlation.** Whenever the robot interprets an actual sensor reading, it constructs a "local" map (such as the ones shown in Figure 2). The *correlation* of the local and the corresponding section of the global map is a measure of their correspondence (Schiele & Crowley 1994). Thus, the correlation—which is a function of the robot position—gives a second source of information for estimating the robot's position.
3. **Wall orientation.** The third source of information estimates and memorizes the *global wall orientation* (Crowley 1989; Hinkel & Knieriemen 1988). This approach rests on the restrictive assumption that walls are either parallel or orthogonal to each other, or differ by more than 15 degrees from these canonical wall directions. In the beginning of robot operation, the global orientation of walls is estimated by searching straight line segments in consecutive sonar measurements. Once the global wall orientation has been estimated, it is used to readjust the robot's orientation based on future sonar measurements.

All three mechanisms basically provide a probability density for the robot's position (Thrun & Bücken 1996). Gradient descent is then iterated to determine the most likely robot position (in an any-time fashion). Notice that position control based on odometry and map correlation alone (items 1 and 2 above) works well if the robot travels through mapped terrain, but seizes to function if the robot explores and maps unknown terrain. The third mechanism, which arguably relies on a restrictive assumption concerning the nature of indoor environments, has proven extremely valuable when autonomously exploring and mapping large-scale indoor environments.



(a)

(b)

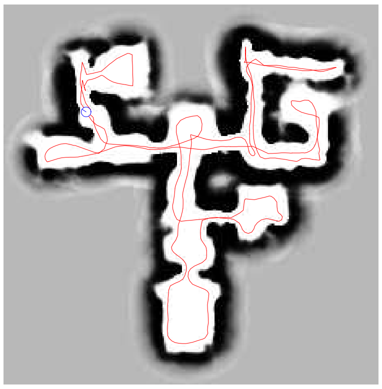


Figure 4: Map constructed without (a) and with (b) the position estimation mechanism described in this paper.

Exploration

To autonomously acquire maps, the robot has to explore. The idea for (greedy) exploration is to let the robot always move on a minimum-cost path to the nearest unexplored grid cell; The cost for traversing a grid cell is determined by its occupancy value. The minimum-cost path is computed using a modified version of *value iteration*, a popular dynamic programming algorithm (Howard 1960) (which bears similarities to A* (Nilsson 1982)).

In a nutshell, starting at each unexplored grid-cell, value iteration propagates values through the map. After convergence, each value measures the cumulative costs for moving to the cost-nearest unexplored grid cell. Figure 5a shows a value function after convergence. All white regions are unexplored, and the grey-level indicates the cumulative costs for moving towards the nearest unexplored point. Notice that the all minima of the value function correspond to unexplored regions—there are no local minima. Once value iteration converges, greedy exploration simply amounts to steepest descent in the value function, which can be done very efficiently. Figure 5b, sketches the path taken during approximately 15 minutes of autonomous exploration. The value function can, however, be used to generate motion control at any time (Dean & Boddy 1988), long before dynamic programming converges. Value iteration has the nice property that values are generated for *all* cells in the grid, not just the current robot position.

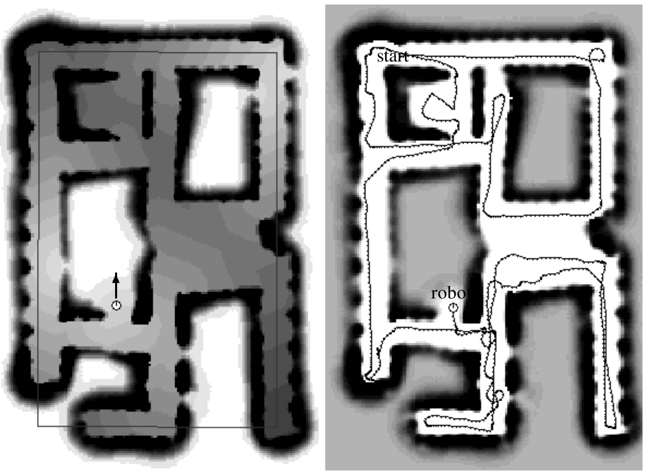


Figure 5: **Autonomous exploration.** (a) Exploration values, computed by value iteration. White regions are completely unexplored. By following the grey-scale gradient, the robot moves to the next unexplored area on a minimum-cost path. (b) Actual path traveled during autonomous exploration, along with the resulting metric map. The large black rectangle in (a) indicates the global wall orientation θ_{wall}.

Thus, if the robot has to change its path to avoid a collision with an unexpected obstacle, it can directly continue exploration without further planning. During exploration, the robot moves constantly, and frequently reaches a velocity of 80 to 90 cm/sec (see also (Buhmann *et al.* 1995; Fox, Burgard, & Thrun 1995)).

In grid maps of size 30 by 30 meters, optimized value iteration, done from scratch, requires approximately 2 to 10 seconds on a SUN Sparc station. For example, the planning time in the map shown in Fig. 3 is typically under 2 seconds, and re-planning (which becomes necessary when the map is updated) is performed usually in a tenth of a second. In the light of these results, one might be inclined to think that grid-based maps are sufficient for autonomous robot navigation. However, value iteration (and similar planning approaches) require time quadratic in the number of grid cells, imposing intrinsic scaling limitations that prohibit efficient planning in large-scale domains. Due to their compactness, topological maps scale much better to large environments. In what follows we will describe our approach for deriving topological graphs from grid maps.

Topological Maps

Topological maps are built on top of the grid-based maps. The key idea is simple but very effective: The free-space of a grid-based map is partitioned into a small number of regions, separated by *critical lines*. Critical lines correspond to narrow passages such as doorways. The partitioned map is then mapped into a isomorphic graph. The precise algorithm is illustrated in Figure 6, and works as follows:

1. **Thresholding.** Initially, each occupancy value in the occupancy grid is thresholded. Cells whose occupancy value is below the threshold are considered free-space (denoted by C). All other points are considered occupied (denoted by $\bar{C}$).

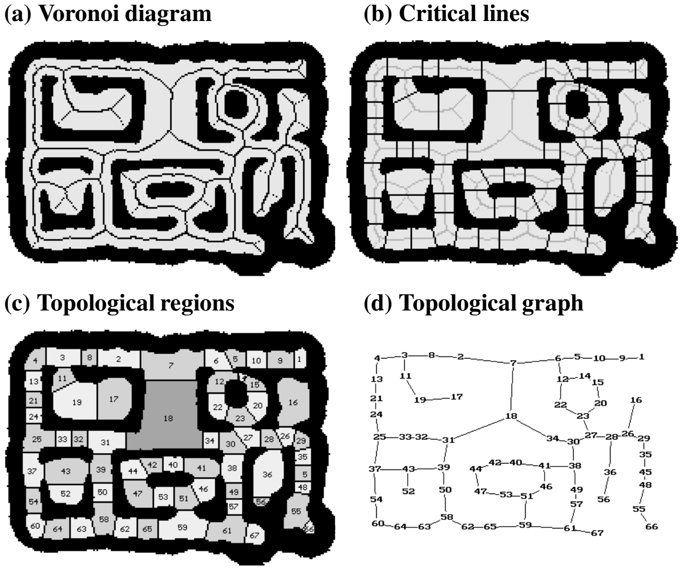


Figure 7: Extracting the topological graph from the map depicted in Figure 3: (a) Voronoi diagram, (b) Critical points and lines, (c) regions, and (d) the final graph.

Figure 6: **Extracting topological maps.** (a) Metric map, (b) Voronoi diagram, (c) critical points, (d) critical lines, (e) topological regions, and (f) the topological graph.

2. **Voronoi diagram.** For each point in free-space $\langle x, y \rangle \in C$, there is one or more *nearest point(s)* in the occupied space $\bar{C}$. We will call these points the *basis points of* $\langle x, y \rangle$, and the distance between $\langle x, y \rangle$ and its basis points the *clearance of* $\langle x, y \rangle$. The Voronoi diagram (Latombe 1991) is the set of points in free-space that have at least two different (equidistant) basis-points (see Figure 6b).
3. **Critical points.** The key idea for partitioning the free-space is to find *"critical points."* Critical points $\langle x, y \rangle$ are points on the Voronoi diagram that minimize clearance locally. In other words, each critical point $\langle x, y \rangle$ has the following two properties: (a) it is part of the Voronoi diagram, and (b) the clearance of all points in an ε-neighborhood of $\langle x, y \rangle$ is *not* smaller. Figure 6c illustrates critical points.
4. **Critical lines.** Critical lines are obtained by connecting each critical point with its basis points (*cf.* Figure 6d). Critical points have exactly two basis points (otherwise they would not be local minima of the clearance function). Critical lines partition the free-space into disjoint regions (see Figure 6e).
5. **Topological graph**. The partitioning is mapped into an isomorphic graph. Each region corresponds to a node in the topological graph, and each critical line to an arc. Figure 6f shows an example of a topological graph.

Critical lines are motivated by two observations. Firstly, when passing through a critical line, the robot is forced to move in a considerably small region. Hence, the loss in performance inferred by planning using the topological map (as opposed to the grid-based map) is considerably small. Secondly, narrow regions are more likely blocked by obstacles (such as doors, which can be open or closed).

Figure 7 illustrates the process of extracting a topological map from the grid-based map depicted in Figure 3. Figure 7a shows the Voronoi diagram of the thresholded map, and Figure 7b depicts the critical lines (the critical points are on the intersections of critical lines and the Voronoi diagram). The resulting partitioning and the topological graph are shown in Figure 7c&d. As can be seen, the map has been partitioned into 67 regions.

Performance Results

Topological maps are abstract representations of metric maps. As is generally the case for abstract representations and abstract problem solving, there are three criteria for assessing the appropriateness of the abstraction: *consistency*, *loss*, and *efficiency*. Two maps are *consistent* with each other if every solution (plan) in one of the maps can be represented as a solution in the other map. The *loss* measures the loss in performance (path length), if paths are planned in the more abstract, topological map as opposed to the grid-based map. *Efficiency* measures the relative time complexity of problem solving (planning). Typically, when using abstract models, efficiency is traded off with consistency and performance loss.

Consistency

The topological map is always consistent with the grid-based map. For every abstract plan generated using the topological map, there exists a corresponding plan in the grid-based map (in other words, the abstraction has the *downward solution property* (Russell & Norvig 1995)). Conversely, every path that can be found in the grid-based map has an abstract representation which is a admissible plan in the topological map (*upward solution property*). Notice that although consistency appears to be a trivial property of the topological maps, not every topological approach proposed in the literature generates maps that would be consistent with their corresponding metric representation.

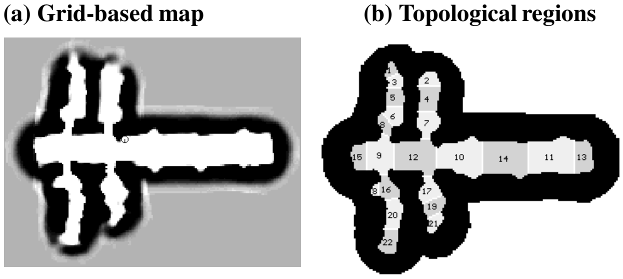


Figure 8: Another example of a map.

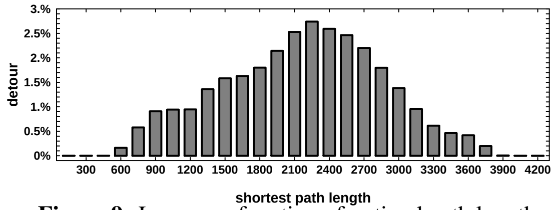


Figure 9: Loss, as a function of optimal path length.

Loss

Abstract representations—such as topological maps—lack detail. Consequently, paths found in the topological map may not be as short as paths found using the metric representation. To measure performance loss, we empirically compared paths generated using the metric map shown in Figure 3 with those generated using the corresponding topological map, shown in Figures 7d. Value iteration can be applied using both representations. In grid-based maps, value iteration is applied just as described above. However, instead of planning paths to unexplored regions, paths were planned from a particular start point to a particular goal point. To compare the results to those obtained using topological representations, first the corresponding shortest path in the topological graph was determined. Subsequently, the shortest path was determined that followed exactly this topological plan. As a result, the quality of topological plans can directly be compared to those derived using the metric map.

We conducted a total of 23,881,062 experiments, each using a different starting and goal position that were generated systematically with an evenly-spaced grid. The results are intriguing. The average length of the shortest path is 15.88 meters. If robot motion is planned using the topological map, this path length increases on average only by 0.29 meters, which is only 1.82% of the total path length. It is remarkable that in 83.4% of all experiments, the topological planner returns a loss-free plan. The largest loss that we found in our experiments was 11.98 meters, which occurred in 6 of the 23,881,062 experiments. Figure 9a shows the average loss as a function of the length of the shortest path. Figure 8 depicts a different map. Here the loss is zero, since both maps are free of cycles.

Efficiency

The most important advantage of topological planning lies in its efficiency. Dynamic programming is quadratic in the number of grid cells. The map shown in Figure 3 happens to possess 27,280 explored cells. In the average case, the number of iterations of value iteration is roughly equivalent to the length of the shortest path, which in our example map is 94.2 cells. Thus, in this example map, value iteration requires on average $2.6{\cdot}10^6$ backups. Planning using the topological representation is several orders of magnitudes more efficient. The average topological path length is 7.84. Since the topological graph shown in Figure 7d has 67 nodes, topological planning requires on average 525 backups. Notice the enormous gain in efficiency! Planning using the metric map is $4.9{\cdot}10^3$ more expensive than planning with the topological map. In other words, planning on the topological level increases the efficiency by more than three orders of magnitude, while inducing a performance loss of only 1.82%.

The map shown in Figure 8, which is smaller but was recoded with a higher resolution, consists of 20,535 explored grid cells and 22 topological regions. On average, paths in the grid-based map lead through 84.8 cells, while the average length of a topological plan is 4.82 (averaged over 1,928,540 systematically generated pairs of points). Here the complexity reduction is even larger. Planning using the metric map is $1.6{\cdot}10^4$ more expensive than planning with the topological map. While these numbers are empirical and only correct for the particular maps investigated here, we conjecture that the relative quotient is roughly correct for other maps as well.

It should be noted that the compactness topological maps allows us to exhaustively pre-compute and memorize all plans connecting two nodes. Our example maps contain 67 (22) nodes, hence there are only 2,211 (231) different plans that are easily generated and memorized. If a new path planning problem arrives, topological planning amounts to looking up the correct plan.

The reader may also notice that topological plans often do not directly translate into motion commands. In (Thrun & Bücken 1996), a local "triplet planner" is described, which generates cost-optimal plans for triplets of adjacent topological regions. As shown there, triplet plans can also be pre-computed exhaustively, but they are not necessarily optimal, hence cause some small additional performance loss (1.42% and 1.19% for the maps investigated here).

Discussion

This paper proposes an integrated approach to mapping indoor robot environments. It combines the two major existing paradigms: grid-based and topological. Grid-based maps are learned using artificial neural networks and Bayes' rule. Topological maps are generated by partitioning the grid-based map into critical regions.

Building occupancy maps is a fairly standard procedure, which has proven to yield robust maps at various research sites. To the best of our knowledge, the maps exhibited in this paper are significantly larger than maps constructed from sonar sensors by other researchers. The most important aspect of this research, however, is the way topological graphs are constructed. Previous approaches have constructed topological maps from scratch, memorizing only partial metric information along the way. This often led to problems of disambiguation (*e.g.*, different places that look alike), and problems of establishing correspondence (*e.g.*, different views of the same place). This paper advocates to integrate both, grid-based and topological maps. As a direct consequence, different places are naturally disambiguated, and nearby locations are detected as such. In the integrated approach, landmarks play only an indirect role, through the grid-based position estimation mechanisms. Integration of landmark information over multiple measurements at multiple locations is automatically done in a consistent way. Visual landmarks, which often come to bear in topological approaches, can certainly be incorporated into the current approach, to further improve the accuracy of position estimation. In fact, sonar sensors can be understood as landmark detectors that indirectly—through the grid-based map—help determine the actual position in the topological map (*cf.* (Simmons & Koenig 1995)).

One of the key empirical results of this research concerns the cost-benefit analysis of topological representations. While grid-based maps yield more accurate control, planning with more abstract topological maps is several orders of magnitude more efficient. A large series of experiments showed that in a map of moderate size, the efficiency of planning can be increased by three to four orders of magnitude, while the loss in performance is negligible (*e.g.*, 1.82%). We believe that the topological maps described here will enable us to control an autonomous robot in multiple floors in our university building—complex mission planning in environments of that size was completely intractable with our previous methods.

A key disadvantage of grid-based methods, which is inherited by the approach presented here, is the need for accurately determining the robot's position. Since the difficulty of position control increases with the size of the environment, one might be inclined to think that grid-based approaches generally scale poorly to large-scale environments (unless they are provided with an accurate map). Although this argument is convincing, we are optimistic concerning the scaling properties of the approach taken here. The largest cycle-free map that was generated with this approach was approximately 100 meters long; the largest single cycle measured approximately 58 by 20 meters. We are not aware of any purely topological approach to robot mapping that would have been demonstrated to be capable of producing consistent maps of comparable size. Moreover, by using more accurate sensors (such as laser range finders), and by re-estimating robot positions backwards in time (which would be mathematically straightforward, but is currently not implemented because of its enormous computational complexity), we believe that maps can be learned and maintained for environments that are an order of magnitude larger than those investigated here.

Acknowledgment

The authors wish to thank the RHINO mobile robot group at the University of Bonn, in particular W. Burgard, A. Cremers, D. Fox, M. Giesenschlag, T. Hofmann, and W. Steiner, and the XAVIER mobile robot group at CMU. We also thank T. Ihle for pointing out an error in a previous version of this paper.

This research is sponsored in part by the National Science Foundation under award IRI-9313367, and by the Wright Laboratory, Aeronautical Systems Center, Air Force Materiel Command, USAF, and the Advanced Research Projects Agency (ARPA) under grant number F33615-93-1-1330. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of NSF, Wright Laboratory or the United States Government.

References

Buhmann, J.; Burgard, W.; Cremers, A. B.; Fox, D.; Hofmann, T.; Schneider, F.; Strikos, J.; and Thrun, S. 1995. The mobile robot Rhino. *AI Magazine* 16(1).

Crowley, J. 1989. World modeling and position estimation for a mobile robot using ultrasonic ranging. In *Proceedings 1989 IEEE International Conference on Robotics and Automation*.

Dean, T. L., and Boddy, M. 1988. An analysis of time-dependent planning. In *Proceeding Seventh NCAI*, AAAI.

Elfes, A. 1987. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation* 3(3):249–265.

Engelson, S., and McDermott, D. 1992. Error correction in mobile robot map learning. In *Proceedings 1992 IEEE International Conference on Robotics and Automation*.

Feng, L.; Borenstein, J.; and Everett, H. 1994. "where am I?" sensors and methods for autonomous mobile robot positioning. TR UM-MEAM-94-12, University of Michigan at Ann Arbor.

Fox, D.; Burgard, W.; and Thrun, S. 1995. The dynamic window approach to collision avoidance. TR IAI-TR-95-13, University of Bonn.

Hinkel, R., and Knieriemen, T. 1988. Environment perception with a laser radar in a fast moving robot. In *Proceedings Symposium on Robot Control*.

Howard, R. A. 1960. *Dynamic Programming and Markov Processes*. MIT Press.

Kortenkamp, D., and Weymouth, T. 1994. Topological mapping for mobile robots using a combination of sonar and vision sensing. In *Proceedings Twelfth NCAI*, AAAI.

Kuipers, B., and Byun, Y.-T. 1990. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. TR, University of Texas at Austin.

Latombe, J.-C. 1991. *Robot Motion Planning*. Kluwer Academic Publishers.

Matarić, M. J. 1994. Interaction and intelligent behavior. Technical Report AI-TR-1495, MIT, AI-Lab.

Moravec, H. P. 1988. Sensor fusion in certainty grids for mobile robots. *AI Magazine* 61–74.

Nilsson, N. J. 1982. *Principles of Artificial Intelligence*. Springer Publisher.

Pearl, J. 1988. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers.

Pierce, D., and Kuipers, B. 1994. Learning to explore and build maps. In *Proceedings Twelfth NCAI*, AAAI.

Rencken, W. 1993. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Russell, S., and Norvig, P. 1995. *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Schiele, B., and Crowley, J. 1994. A comparison of position estimation techniques using occupancy grids. In *Proceedings IEEE International Conference on Robotics and Automation*.

Simmons, R., and Koenig, S. 1995. Probabilistic robot navigation in partially observable environments. In *Proceedings IJCAI-95*.

Thrun, S., and Bücken, A. 1996. Learning maps for indoor mobile robot navigation. TR CMU-CS-96-121, Carnegie Mellon University.

Thrun, S. 1993. Exploration and model building in mobile robot domains. In *Proceedings ICNN-93*, IEEE NN Council.