Augmenting Topology-Based Maps with Geometric Information

Elisabetta Fabrizi [a] Alessandro Saffiotti [b,1]

[a] *Dip. di Informatica e Sistemistica*
Università di Roma "La Sapienza"
Via Eudossiana 18, I-00184 Roma, Italy

[b] *Center for Applied Autonomous Sensor Systems*
Dept. of Technology, Örebro University
Fakultetsgatan 1, S-70182 Örebro, Sweden

Abstract

Topology-based maps are a new representation of the workspace of a mobile robot, which capture the structure of the free space in the environment in terms of the basic topological notions of connectivity and adjacency. A topology-based map can represent the environment in terms of open spaces (rooms and corridors) connected by narrow passages (doors and junctions). In this paper, we show how to enrich a topology-based map with geometric information useful for the generation and execution of navigation plans. Both the topology-based map and its geometric information are automatically extracted from sensor data. We illustrate the use of topology-based maps for planned behavior-based navigation on a real robot.

Key words: Environment modeling, topological maps, occupancy grids, behavior-based navigation, mathematical morphology.

1 Introduction

The most common representations of the space in mobile robotics can be grouped in two classes: *metric maps*, and *topological maps*. A metric map represents the environment according to the absolute position of the objects. A topological map is a more abstract representation that describes relationships among features of the environment, without any absolute reference system

[1] Contact author. Email: `asaffio@aass.oru.se`, `http://www.aass.oru.se/`.

Preprint submitted to Elsevier Science *12 September 2001*

Appers in: *Robotics and Autonomous Systems* **40**(2):91-97, 2002

(7; 15; 6; 3). Topological maps are usually represented in graph form. Unfortunately, there is no uniform semantics associated with these graphs, and the meanings attributed by different authors to the nodes and links are widely different. For example, in the maps defined by Kuipers and Byun (7) nodes represent *places*, characterized by sensor data, and arcs represent *paths* between places, characterized by control strategies. By contrast, the maps defined by Thrun (15) are obtained by partitioning a probabilistic occupancy grid into *regions* (nodes) separated by *narrow passages* (arcs) according to a measure of local clearance.

In (4) we have proposed *topology-based maps* as a high-level representation of the working space. Topology-based maps are intended to capture the topological structure of spatial information, and have a uniform semantics based on concepts from general topology. Thanks to their abstract nature, topology-based maps are compact and fairly stable with respect to sensor noise and to small changes in the environment.

Topology-based maps contain information about the connectivity of the space, which is needed to plan a navigation strategy. Autonomous robots navigating in real-world, dynamic environments typically do not need a detailed global metric model, since the generation of the actual trajectory is usually done reactively during execution. Some geometric information, however, is often necessary in order to plan a correct navigation strategy: e.g., the planner should distinguish between a corridor-like space that can be traversed by wall following, and an open space that should be traversed by dead reckoning. For this reason, many authors annotate their topological maps with some metric information (12; 11; 14). Often, these maps are given *a priori*.

In this paper, we show how topology-based maps can be automatically built and annotated with geometric information using sensor data. We start from a fuzzy occupancy grid built from sonar ranges, and extract a topology-based graph in which nodes represent wide free spaces and arcs represent passages connecting these spaces. We then use techniques borrowed from the field of image processing to compute the shape parameters of each space: these parameters allow us to classify the nodes into corridors and rooms, and to associate them with geometric information like length, width, and orientation.

2 Topology-based maps

In this section, we show how to partition the environment into a set of open spaces (rooms and corridors) connected by narrow passages. An illustrative example is given in Section 4 below.

Let I be a *fuzzy gridmap*, defined as a two-dimensional array whose elements, called cells, have real values in the interval $[0, 1]$. In our case, the gridmap is a representation of the empty space built from range data, where the value in each cell represents the degree of belief in the emptiness of the corresponding portion of the environment (8; 5). Note that the procedure below could be applied to maps representing other local features of the environment, like temperature or odor.

We then proceed to extract topological information from this gridmap. The key step for this is to think of the gridmap as a *gray-level image*, and to use techniques from image processing to analyze it. In particular, we use fuzzy mathematical morphology (2) to gather information about the *shape* of the empty space represented in the gridmap; and fuzzy digital topology (10) to extract the *topological structure* of this information.

Mathematical morphology deals with the extraction of shape from a digital image. The shape to be extracted is specified in terms of a *structuring element*, a small pattern that is matched against the neighborhood of each pixel in the image. The matching rules are defined by a pair of operators, *dilation* and *erosion*. In our case, the shapes we want to extract are large open spaces. To achieve this, we use a conic structural element to expresses the fuzzy concept of a large space: intuitively, a larger open space is one in which we can fit a larger portion of the structuring element. This structuring element is used to filter the fuzzy map by an *opening* operator, that is, erosion followed by dilation. The result of applying this operator to our fuzzy gridmap is a new gridmap where the value of each cell represents how much that cell belongs to a large open space.

The morphological information represented in the transformed gridmap is still dispersed into small portions. To capture the topological structure of this information, we use a watershed algorithm (16) to partition, or *segment*, the gridmap into a set of connected components. The intuition here is to see the gridmap as a landscape with valleys and peaks, and to compute the watershed that separates the valleys. The watershed partitions the gridmap into a set of connected regions, which can be interpreted as "large open spaces" bounded by occupied space or narrow passages. These regions constitute the *nodes* of our topology-based map, where the *arcs* represent connections between adjacent regions.

Figures 1 to 3 in Section 4 below illustrate the above steps. More details on this procedure can be found in (4).

3 Extracting geometrical information

In order to capture all the information necessary to perform complex navigation tasks, we extract geometrical features from the partitioned gridmap. The image processing field offers several well defined techniques to gather this kind of information from a gray-level image. The so called *regional descriptors* can represent the geometrical properties of the regions in the image. Note that in our case regions correspond to nodes of the topology-based map, and represent parts of the space. A particularly useful class of descriptors for our goals are the *moments*.

Given a fuzzy gridmap, we indicate by $I(x, y)$ the value of the cell (x, y). The moment of order $p + q$ is defined by

$$m_{pq} = \sum_x \sum_y x^p y^q I(x, y),$$

Moments have several attractive properties. For instance, they fully characterize the information contained in the image: a uniqueness theorem (9) guarantees that, under general conditions, the infinite set of moments $m_{pq}, p, q = 0, 1, \ldots$ uniquely determines $I(x, y)$ and vice versa. Moreover, it is easy to associate a physical meaning to each moment. For example, in a binary map the moment m_{00} represents the area of the region of I with non-zero values. (The extension to a fuzzy map is natural.)

For our purposes, we use moments to extract, for each region in our partitioned gridmap, the following features:

- center
- orientation
- width
- length
- eccentricity

The *center* synthesizes the notion of *position* of the region, and it is defined by

$$\bar{x} = \frac{m_{10}}{m_{00}} \; \bar{y} = \frac{m_{01}}{m_{00}}.$$

Based on the center of a region, we can define *central moments* by

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q I(x, y).$$

These moments are invariant with respect to translation and scaling of the figure.

If the region is elongated, it is possible to define an orientation with respect to the coordinate axis. To do this, note that the second order central moments can be organized in the following matrix:

$$\begin{bmatrix} \mu_{11} & \mu_{12} \\ \mu_{21} & \mu_{22} \end{bmatrix}$$

The eigenvectors of the matrix point in the directions of maximal region spread, with the constraint that they are to be orthogonal. They are called *principal axes*. The *orientation* θ of the region can be described by the largest eigenvalue and corresponding eigenvector. θ can be interpreted as the rotation that makes the matrix of second order moments diagonal. The eigenvalues measure the degree of spread, and as a consequence are proportional to the *width* and *length* of the region.

The last parameter that we use to characterize a region of space is the *eccentricity*, defined by

$$e = \frac{\mu_{02} cos^2(\theta) + \mu_{20} sin^2(\theta) - \mu_{11} sin^2(\theta)}{\mu_{02} sin^2(\theta) + \mu_{20} cos^2(\theta) + \mu_{11} cos^2(\theta)},$$

that tells us how much the region is elongated or rounded. This parameter is useful to distinguish room-like regions from corridor-like regions, and therefore to make a decision about the motion strategy to perform. Note that with a small number of parameters, each region is enriched with a large amount of information useful to perform navigation tasks. Obviously, it is possible to compute more geometrical parameters depending on the task. The ones described above are those that we have found sufficient for our experiments in planning and execution of behavior-based robot navigation.

4 Example

In order to illustrate our technique, we show a simple example of extraction of a topology-based map, and of its corresponding geometric properties. This experiment has been run on a Nomad 200 robot equipped with a ring of 16 Polaroid ultrasonic sensors in an unmodified office environment.

Figure 1 shows the fuzzy gridmap representing the free space built using the technique described in (8; 5). Each cell represents a square of side 10 cm.

Fig. 1. The fuzzy gridmap of free space for our test environment.

Darker cells indicate more confidence in that cell's being empty. The environment consists in two corridors meeting at an entrance hall, and two rooms cluttered with random furniture. The longer corridor includes a larger area with some table and chairs, which is visible on the left of the grid. The overall size of the environment is about 30×7 meters.

Figure 2 (left) shows the fuzzy gridmap resulting from the fuzzy morphological opening of the original gridmap through a conic structural element. Darker cells indicate areas of space that belong to wider open spaces. Note that the information in each cell now represents a property of the spatial neighborhood of that cell rather than a property of the points inside the cell.

Figure 2 (right) shows the regions extracted by topological segmentation of the above gridmap, together with the axes giving the center, size, and orientation of each region, computed as described above. The corresponding annotated topology-based map is shown in Figure 3. Width and length are in meters; orientation is in degrees from the horizontal axis in the gridmap; center and eccentricity values are not shown. However, eccentricity is used to classify each node as either a room or a corridor: each node is given a name from its class plus a progressive number. The map also associates each arc (genericall called a "door") with its position relative to each one of the two regions that

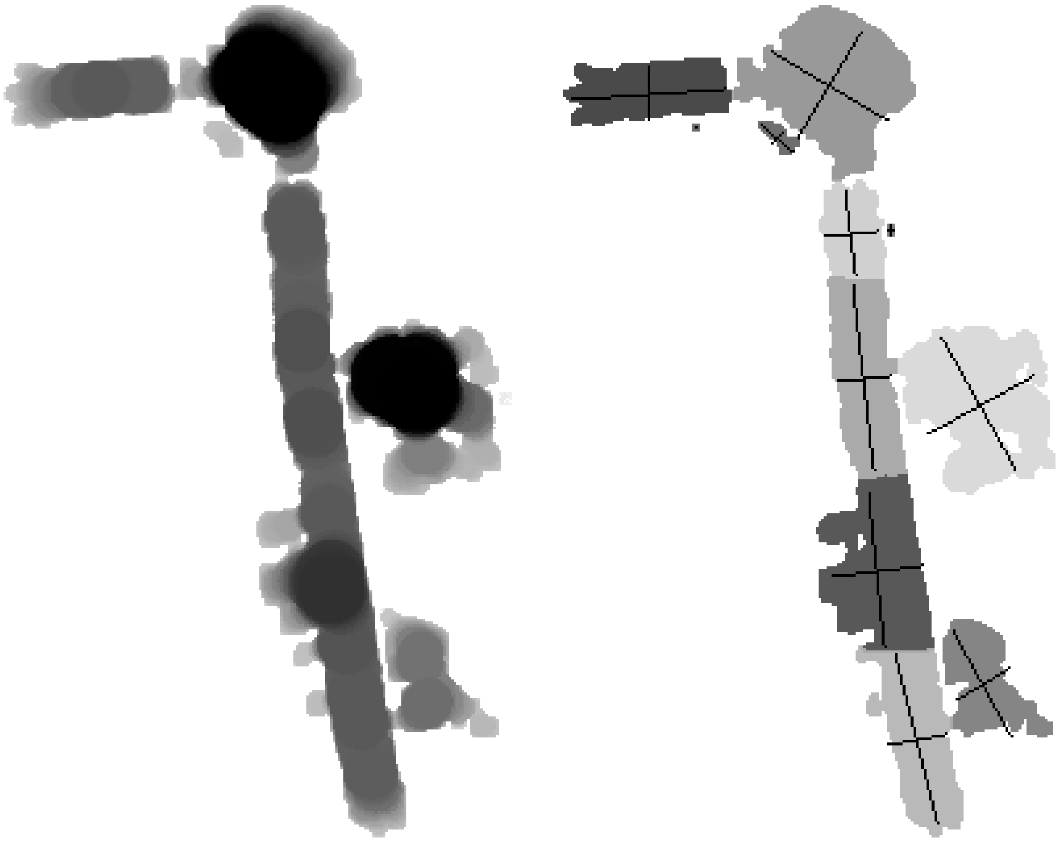

Fig. 2. The morphological opening of the previous gridmap (left), and the corresponding regions (right).

it connects.

The full extraction procedure, from the gridmap to the final graph, took less than a second on a Pentium-II processor at 200 MHz in this experiment. The size of the gridmap was 400×280 cells.

5 Behavior-based navigation on topology-based maps

In order to illustrate the possible uses of topology-based maps, we discuss in this section how we have integrated them in the Thinking Cap (13), a fuzzy behavior-based navigation system based on the approach proposed in (12).

Like most modern behavior-based architectures, the Thinking Cap goes beyond pure reactivity and uses environment models and temporal projection to plan a specific behavior activation strategy for a given goal in a given environment, or *behavioral plan.* A behavioral plan is a set of "*situation* $\Rightarrow$ *behavior*" rules, where *situation* is a formula of fuzzy logic. These plans are generated by simple goal regression from knowledge about the preconditions and effects

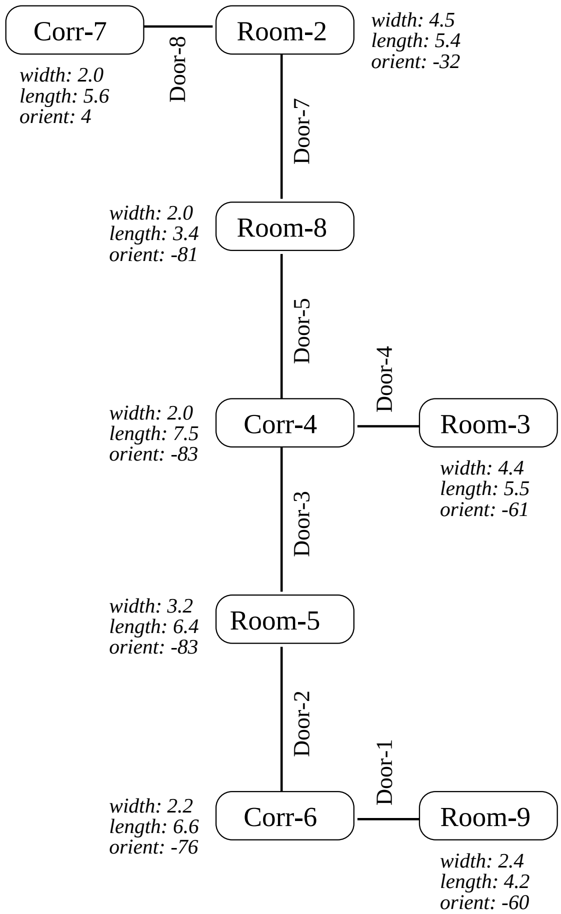

Fig. 3. The topology-based map extracted from our gridmap.

of each behavior. Table 1 shows the preconditions and effects for some of the behaviors implemented in the Thinking Cap.

Topology-based maps contain all the local geometric information needed for the execution of the navigation behaviors in the behavioral plan, like the expected position of rooms, doors and corridors. They also contain all the information needed by the planner to generate a behavioral plan. The static preconditions in Table 1 (i.e., all but the "at" predicate) are easily verified on a topology-based map: doors are arcs in the map; corridors and room are nodes, classified according to their eccentricity; and "partof(x, y)" is true of a door x and a space y if arc x is incident in y. The "at" predicate is used to do goal regression. For instance, if we are given the goal to be "at(Room3)" and we know that Door1 is part of Room3, we can generate the plan

at(Door1) $\Rightarrow$ Cross(Door1,Room3)

and post the new goal of being "at(Door1)". This goal in turn can be achieved by following Corr1 if we know that Door1 is part of it, and so on until we post a "at" goal which is true in the current state.

behavior	*precondition*	*effect*
Cross(d, x)	at(d) $\wedge$ door(d) $\wedge$ partof(d, x)	at(x)
Follow(c, x)	at(c) $\wedge$ corridor(c) $\wedge$ partof(x, c)	at(x)
GoTo(x)	at(r) $\wedge$ room(r) $\wedge$ partof(x, r)	at(x)

Table 1
Preconditions and effects for three simple behaviors.

Figure 4 shows a simple navigation experiment where we use the topology-based map above. The map is visualized as a set of rectangular areas (rooms and corridors) connected by gateways (doors). The robot has been given the goal to be "at(Room9)" while it was at Room3. The planner then has generated the following plan (shown in a simplified form):

$$\begin{aligned}
\text{at(Room3)} &\Rightarrow \text{GoTo(Door4)}\\
\text{at(Door4)} &\Rightarrow \text{Cross(Door4,Corr4)}\\
\text{at(Corr4)} &\Rightarrow \text{Follow(Corr4,Door3)}\\
\text{at(Door3)} &\Rightarrow \text{Cross(Door3,Room5)}\\
\text{at(Room5)} &\Rightarrow \text{GoTo(Door2)}\\
\text{at(Door2)} &\Rightarrow \text{Cross(Door2,Corr6)}\\
\text{at(Corr6)} &\Rightarrow \text{Follow(Corr6,Door1)}\\
\text{at(Door1)} &\Rightarrow \text{Cross(Door1,Room9)}
\end{aligned}$$

The dotted line in Figure 4 shows the trajectory generated by the robot while executing this plan in the actual environment. Note that the plan only provides *elastic contraints* on this trajectory: the actual trajectory is generated by the behaviors by reactively adapting to the actual execution contingencies, including the actual geometry of the environment and obstacles. Also note that in this experiment localization was based on dead reckoning alone.

More details on our approach to generate and execute behavioral plans can be found in (12).

6 Conclusions

Topology-based maps provide a new notion of map based on concepts from general topology. The ability to annotate topology-based maps with geometric information makes them more useful for autonomous navigation. A peculiar aspect of our approach is the reliance on techniques borrowed from the field of image processing to extract the maps from sensor data. The extraction procedure proved to be robust with respect to sensor noise, and fast enough to be performed in real time during navigation.

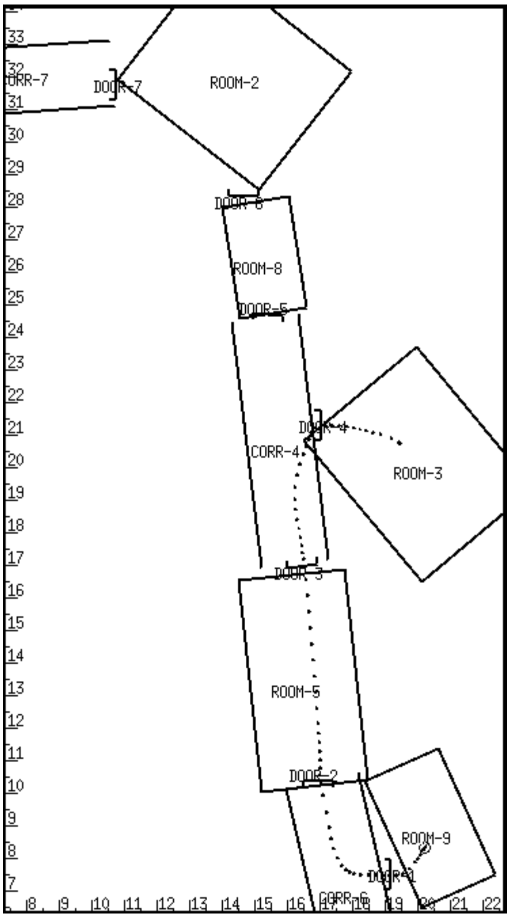

Fig. 4. A navigation experiment using the above topology-based map.

We have shown that annotated topological maps can be used to generate and execute navigation plans on one specific robot navigation system. We speculate that they can be used in a similar way on most other systems that use a planner to plan a behavior-based navigation strategy (e.g., (1; 11; 14)). In our future work, we intend to investigate the use of prior structural knowledge (e.g., about the shape and orientation of rooms) to improve the quality of the extracted topology-based maps. We also plan to study the use of these maps to address the important issue of self-localization.

Acknoledgements

This work was partly supported by the Swedish KK Foundation. Thanks to Isabelle Bloch, Holger Schellwat, and Nino Trainito for enlightening discussions on mathematical morphology, on general topology, and on model-based mapping, respectively.

References

[1] R. C. Arkin. Integrating behavioral, perceptual and world knowledge in reactive navigation. *Robotics and Autonomous Systems*, 6:105–122, 1990.

[2] I. Bloch and H. Maître. Fuzzy mathematical morphologies: a comparative study. *Pattern Recognition*, 28(9):1341–1387, 1995.

[3] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes. Robotic exploration as graph construction. *IEEE T. on Robotics and Automation*, 7(6):859–865, 1991.

[4] E. Fabrizi and A. Saffiotti. Extracting topology-based maps from gridmaps. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, 2000, pp. 2972–2978. http://www.aass.oru.se/˜asaffio/.

[5] E. Fabrizi, G. Oriolo and G. Ulivi. Accurate map building via fusion of laser and ultrasonic range measures. In D. Driankov and A. Saffiotti, editors, *Fuzzy logic techniques for autonomous vehicle navigation*, pp. 257–280. Physica/Springer-Verlag, 2001.

[6] D. Kortenkamp and T. Weymouth. Topological mapping for mobile robots using combination of sonar and vision sensing. In *Proc. of the AAAI Conf.*, pages 979–984, Menlo Park, CA, 1994.

[7] B. Kuipers and Y. T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8:47–63, 1991.

[8] G. Oriolo, G. Ulivi, and M. Vendittelli. Fuzzy maps: a new tool for mobile robot perception and planning. *J. of Robotic Systems*, 14:179–197, 1997.

[9] A. Papoulis. *Probability, random variables and stichastic processes.* McGraw-Hill, 1965.

[10] A. Rosenfeld. Fuzzy digital topology. *Information Control*, 40(1):76–87, 1979.

[11] B-S. Ryu and H.S. Yang. Integration of reactive behaviors and enhanced topological map for robust mobile robot navigation. *IEEE T. on Systems, Man, and Cybernetics*, A 29(5):474–485, 1999.

[12] A. Saffiotti, K. Konolige, and E. H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526, 1995.

[13] A. Saffiotti. Thinking Cap. www.aass.oru.se/˜asaffio/Software/TC/.

[14] H. Surmann and L. Peters. The uses of fuzzy control for the autonomous robot Moria. In D. Driankov and A. Saffiotti, editors, *Fuzzy logic techniques for autonomous vehicle navigation*, pp. 343–366. Physica/Springer-Verlag, 2001.

[15] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 1:21–71, 1999.

[16] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE T. on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.