A Virtual Sensor for Room Detection

P. Buschka A. Saffiotti

Center for Applied Autonomous Sensor Systems
Dept. of Technology, Örebro University
S-70182 Örebro, Sweden
{par.buschka,alessandro.saffiotti}@aass.oru.se

Abstract

Indoor environments typically consist of sets of connected room-like spaces. We present a local technique that uses range data to detect these spaces during navigation. Our technique includes two parts: segmentation, which isolates room-like spaces and detects when the robot has entered a new one; and feature extraction, which associates each space with a set of geometric features useful for navigation or recognition. Many such features can be considered: here we propose a new method to compute width and length of a rectangular room in a way which is largely invariant with respect to the configuration of the furniture. We report experimental results that show the performance of our technique, and hint at a possible use of this technique for coarse localization on a topological map.

1 Introduction

Most indoor environments are structured in terms of enclosed spaces, like rooms, halls and corridors, and passages between them, like doorways and junctions. For a mobile robot to operate in these environments, it is useful to be able to recognize these architectural elements, to detect the transition between one element and the next, and to distinguish different elements. This is necessary, for instance, in order to perform coarse self-localization, thus knowing at each moment in which room or corridor the robot is located. This ability can also be useful to select an appropriate navigation strategy, for instance using wall following inside corridors and point-to-point navigation inside rooms. Finally, this ability can facilitate human-robot communication, by allowing robots and human to use the same terms (e.g., "room-31") to refer to locations in the environment.

Common sensors used in mobile robots, like range-finders and vision sensors, cannot directly detect structural elements like rooms and corridors. In this paper, we propose a technique to build a *virtual sensor* which detects room-like spaces during navigation, and associates them with a number of characteristic features. Our virtual sensor uses range (sonar) data and odometric information as its main inputs.

A few approaches to detect room-like spaces have already been proposed in the literature. Some of these approaches focus on the computation of a unique *signature* of a room, so that the same room can later be re-identified. For instance, Sarachik [12] finds the dimensions of a room using vision, and Janét [7] extracts room signatures using a neural network. These approaches, however, do not deal with the problem of partitioning, or *segment*, the space into a set of rooms: they simply assume that the robot knows itself to be inside a room [12], or that the part of an occupancy grid relative to the room has already been extracted by hand [7]. Other approaches deal with the segmentation problem, by partitioning the space into a set of distinct, topologically connected sub-spaces. For instance, Thrun [13] partitions an occupancy grid into regions separated by local narrowings. The identified regions, however, do not necessarily correspond to rooms and corridors. Fabrizi and Saffiotti [5] use techniques borrowed from the field of image processing to partition an occupancy grid into "large open spaces separated by narrow passages." Their approach correctly partitions the grid into rooms and corridors. However, this is a global approach, that relies on the availability of a full, metrically consistent map of the environment, and it cannot directly be used to build the type of on-line virtual sensor that we envisage in this paper.

The algorithm proposed in this paper addresses all of the above problems: it automatically segments the space into room and corridor regions; and it computes a set of characteristic parameters for each region. Moreover, the algorithm is incremental: it only maintains a local (in a topological sense) map of the space recently explored by the robot, and it generates information about each detected room while the

Procs. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)
Lausanne, CH, 2002. Pages 637-642.

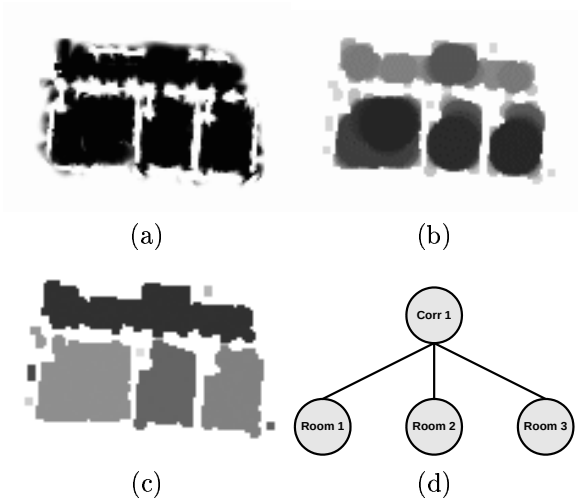


Figure 1: *(a) original gridmap; (b) fuzzy morphological opening; (c) watershed segmentation; (d) corresponding topological map.*

rooms are being visited.

The rest of this paper is organized as follows. We first outline the global segmentation algorithm used to detect room-like regions, and show how to modify it to perform incremental detection. We then describe a new model-based approach to compute parameters (width and length) of these regions that is insensitive to the furniture configuration. The resulting algorithm is illustrated by a series of experiments performed on a real robot. These experiments are also used to evaluate the robustness of the algorithm under different conditions. We conclude the paper by hinting at the use of our virtual sensor to perform coarse topological localization.

2 Node Extraction

In this section, we show how to partition the environment into a set of open spaces (rooms and corridors) connected by narrow passages. Full details of this technique can be found in [5].

We start by building a *fuzzy gridmap*, defined as a two-dimensional array of cells with associated real values in the interval $[0, 1]$. In our case, the gridmap is a representation of the workspace of the robot, and each cell value represents the degree of belief in the emptiness of a portion of the environment (Fig. 1a). Details on fuzzy gridmaps and on how to build them from range data can be found, e.g., in [10, 4].

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 [1] to gather information about the *shape* of the empty space represented in the gridmap; and fuzzy digital topology [11] 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 — see Fig. 1 (b).

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 [14] 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. Fig. 1 (c) shows the regions obtained by applying this algorithm to our sample gridmap, and Fig. 1 (d) shows the corresponding topology, where links indicate adjacency relations.

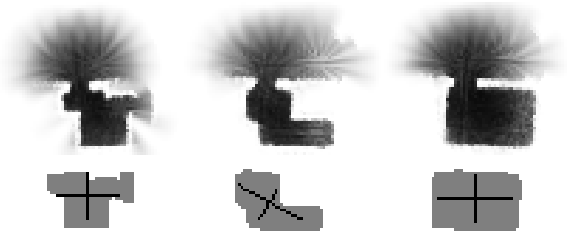
Some precaution must be taken to reduce the impact of noise in the original gridmap. Noisy data may produce small undulations in the landscape, which may result in spurious watersheds. Also, specular reflection phenomena may result in small "phantom" regions of empty spaces outside the space boundaries. In order to filter these sources of noise, we fuse regions that are only separated by a low watershed [9], and we remove regions whose size is below a given threshold.

3 Incremental Node Extraction

The above node extraction procedure is global, and it cannot be directly used to build the type of online virtual sensor that we have in mind because of two reasons. First, we want our sensor to have a limited computational cost. But the global gridmap continuously grows as the robot visits new areas in

(a) (b) (c)

Figure 2: *Incremental extraction. (a) Initial state. (b) A new node is detected. (c) Oldest data are erased and the local gridmap is re-processed.*



(a) (b) (c)

Figure 3: *2nd-order moments of three room regions.*

the environment, thus requiring increasing computational resources for node extraction. Second, we want to use only range and odometric information in our sensor. But the error in the odometric position estimate of the robot accumulates over time, and then a gridmap built using only odometric estimates will inevitably become distorted after some time.

In order to address these problems, we modify the node extraction algorithms to operate on a *local gridmap*, and to only use the most recent range measurements. The process is illustrated in Fig. 2, where the robot path is also shown.

The local gridmap is built incrementally, by adding at each step all the new measurements received since the previous update. After the update, we apply the node extraction process to the local gridmap, as shown in (a) in the figure. The morphological operations used in that process can be done locally over an area limited to the cells that were affected by the last update, increased by twice the size of the structuring element [1]. Then the process is iterated. In (b), a new, earlier unknown node is generated in the segmentation process. We take this to mean that the robot has entered a new node, and then update the topological boundaries of the local gridmap: we discard the oldest measurements, rebuild the gridmap using the measurements left, and repeat the segmentation (c).

The selection of which data to keep is topological. We want the measurements from the nearest topological neighborhood to be kept, and discard the rest. This means we keep the measurements associated with the nodes that the robot has traversed and are closest to the node the robot currently occupies.

More precisely, we define the entry point $e_{i,t}$ for node i at node extraction time t. Let $N_{i,t},\ i = 1, \ldots, n_t$ denote all space occupied by node i of n_t nodes at time t and let $p_{j,t},\ j = 1, \ldots, m_t$ denote the point from which measurement j of m_t measurements was taken at time t, ordered so that $p_{j,t}$ was measured before $p_{j+1,t}$. Then $e_{i,t} = p_{l,t}$ where $l = \min\{j | p_{j,t} \in N_{i,t}\}$. Intuitively, $e_{i,t}$ is the point where the robot first entered the node i.

We also order the entry points and their associated nodes so that $e_{i,t}$ was measured before $e_{i+1,t}$, and as a consequence node $N_{i,t}$ was traversed before node $N_{i+1,t}$.

Every time we do a node extraction we find these entry points for each node traversed. We then compare the number of nodes extracted, and if $n_t > n_{t-1}$ then we have a new node. This condition provides the main output of the incremental extraction: a signal that the robot has moved from one node to the next. This also means that we need to change the notion of "locality" of our local gridmap. We delete all the measurements $p_{j,t} \in N_{r,t},\ r < (n_t - s)$ where s denotes the number of nodes for which we want to save measurements, that is, our topological neighborhood. By doing so, we keep a local gridmap that only corresponds to the latest s nodes extracted.

4 Computing the Node Parameters

The above algorithm gives us a way to detect when the robot has entered a new room-like node. For many applications, it is useful to complement this information by a set of parameters associated to each node. These can be used to distinguish different rooms, to recognize an already visited room, or to help in deciding the navigation strategy.

A few approaches have been proposed in the literature to compute parameters of a room-like space [12, 7, 3]. These approaches typically extract parameters meant to uniquely identify a given room. A common choice to do this, inspired by the field of image processing, is to compute the n-th order moments of the room region in the gridmap [3]. For instance, in our previous work we have used central 2nd order moments to compute the width and length of the nodes [6].

Moment-based approaches depend on the geometric shape of free-space in the room. Because of this, these approaches are intrinsically not invariant with respect to the configuration of the objects (furniture

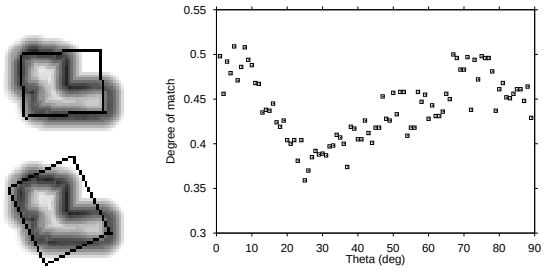


Figure 4: *Left: bounding boxes for two different angles. Right: degree of match for each angle.*

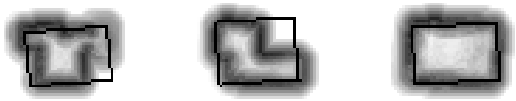


Figure 5: *Best-match boxes for three room regions.*

or people) inside the room. Fig. 3 illustrates this problem. The top row shows three gridmaps built from (simulated) sonar data by visiting the same room under three different layouts of the furniture — in the last case the room was empty. The bottom row shows the central 2nd order moments representing the major and minor axes of the shape: both the value and the orientation of these axes differ in the three cases.

In the work presented here, we explore the use of a model-based technique to extract parameters that describe the room shape and dimensions in a more robust way. We make two key assumptions: (i) the nodes that we try to identify (rooms and corridors) have a rectangular shape; and (ii) the observed empty space in a room fully lies inside the rectangular contour of that room. Based on these assumptions, we extract the room contour by searching the bounding rectangle of the empty space in the room that has the greatest overlap with the frontier of that empty space. See [8] for a related approach.

We proceed in four steps:

First, we extract the fuzzy frontier of a given room region R by computing its fuzzy morphological gradient, called grad(R). The morphological gradient is given by the difference between the grid obtained by dilating R and by eroding R with the same fuzzy structuring element. In our case, we use a cone with a support of n pixels. The result is a grid where the value at each cell tells us how close that cell is to the frontier of R. Fig. 4 (left) shows the fuzzy morphological gradient computed for the room region shown in Fig. 3 (b).

Second, we then consider all possible orientations between 0° and 90°, discretized by a fixed step d. For each such orientation θ, we compute the bounding box of region R oriented as θ, and we call it $\text{box}_R(\theta)$. Fig. 4 (left) shows the boxes obtained for $\theta = 5°$ and $\theta = 25°$, respectively.

Third, for each θ, we compute the degree by which $\text{box}_R(\theta)$ overlaps the fuzzy gradient grad(R) of R. This is given by

$$\text{overlap}_R(\theta) = \frac{1}{N_R} \sum_{c \in \text{box}_R(\theta)} \text{grad}(R)(c), \qquad (1)$$

where N_R is the number of pixels in $\text{box}_R(\theta)$. Intuitively, this degree tells us how much the pixels in the bounding box $\text{box}_R(\theta)$ lay on, or close to, the contour of region R. Fig. 4 (right) shows the value of $\text{overlap}_R(\theta)$ for $\theta \in [0, 90]$ in our example. The two boxes drawn on the left of the picture are the best and worst matching boxes, respectively.

Finally, we chose the θ that maximizes the value of $\text{overlap}_R(\theta)$, and we compute the parameters of the bounding box (width and height) for this θ.

Fig. 5 shows the results obtained by applying the above procedure to the regions in Fig. 3 above. In the three cases, the procedure computes a similar bounding box, which corresponds to the actual boundaries (wall) of the room. Correspondingly, the values of the width and height parameters are similar in all the three cases. This suggests that these parameters describe the shape and dimensions of the room irrespective of the internal configuration of objects, provided that the two assumptions above hold. The experimental results reported below support this conjecture.

Other parameters can be extracted and associated to the nodes if we wish so. For instance, the width and height values can be used to compute the eccentricity of the region, which can be used to classify spaces into rooms (low eccentricity) and corridors (high eccentricity) [6]. Also, we can use parameters that describe the shape of the empty space in the room, like the n-th order moments mentioned above, in order to improve the ability to discriminate different rooms with similar dimensions. This would be done, however, at the expense of being less robust with respect to changes inside the room.

5 Experiments

The above method has been implemented in a software module which takes as input a stream of range measurements plus odometric information, and produces as output a signal indicating that the robot

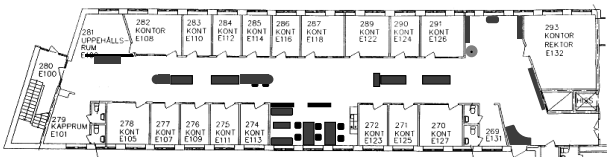


Figure 6: *The office environment used for the experiments, with an approximate size of* 46×12 *meters.*

has entered a new node, together with a set of parameters for the node just exited. This module acts as a virtual sensor for on-line room detection. In our current implementation, ranges are provided by sonar sensors. The extracted parameters include the width and height of the best-fit rectangle, computed as described above.

We have run three series of experiments on a Magellan Pro robot manufactured by IRobot and equipped with a ring of 16 sonars equidistantly placed around the robot. The robot was tele-operated in an office building with several rooms and a large corridor (Fig. 6). Several of these rooms have the same size but different configurations of furniture. The rooms were visited so that a gridmap of the entire room could be built.

The first set of experiments was intended to test the accuracy of node extraction, that is, to make sure that exactly one node was generated for each room traversed. The robot was guided into 25 rooms, and the result of the segmentation was inspected visually. Only rooms that were segmented in exactly one node were counted as correct extractions. (We ignored

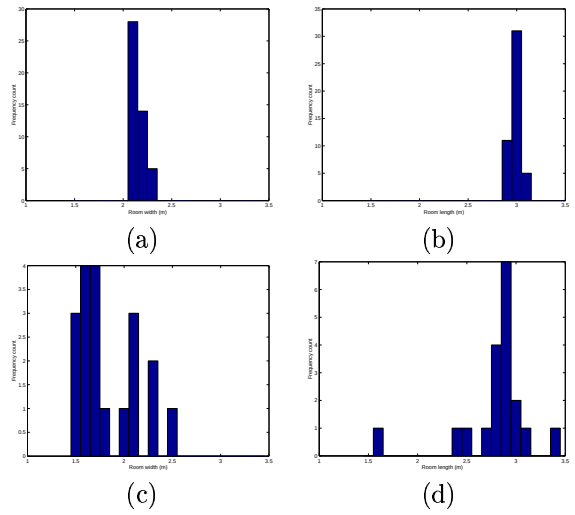


Figure 7: *Top: Histogram of width and length from same room, different set of measurements. Bottom: Histogram of width and length extracted from similar rooms.*

corridors.) 21 rooms out of 25 were extracted correctly while 4 were incorrectly split into two nodes. There are two main reasons for these errors: first, the furniture in these rooms actually divided the space into two loosely separated areas; second, although our fuzzy segmentation technique produces, for each pair of adjacent regions, a *degree of connectivity* between them, we then use a fixed threshold to decide which regions are distinct and which ones are not. A better solution would be to use this degree to associate each edge between regions to a value that indicates its strength: multiple topological hypotheses could then be implicitly represented and maintained, each one weighted by a degree.

The second set of experiments was meant to test the stability of the parameters extracted when the same room was visited in different ways, or when different sets of measurements were used. To do this, we manually selected entry points from extracted nodes, remove the measurements that preceded those points, and then applied our node and parameter extraction algorithm to the remaining measurements. The histograms in Fig. 7 (top) show the distribution of the measured width (a) and length (b) for 40 different experiments in the same room. These results show that our method is fairly insensitive to the size of the local map used.

The last set of experiments was designed to evaluate the robustness of the extracted features (width and length) with respect to changes in the internal shape of the room. We have extracted these features for 19 rooms that all have the same width and length, but have different configurations of furniture and were visited in different ways. The histograms in Fig. 7 (bottom) show the distribution of the measured width (a) and length (b) for these 19 rooms. Although the variance is obviously greater than when repeatedly visiting the same room, the results show that the extracted parameters are relatively insensitive to the room configuration, as it was desired.

6 Discussion and Conclusions

We have presented a method to incrementally extract room-like nodes from range data in a mobile robot. This method has been implemented as a "virtual sensor" that produces two types of information: (i) an indication that the robot has moved from one node to the next, and (ii) a set of parameters that characterize each node. Our algorithm has a low computational cost, and can be run in real time during robot navigation: processing of a typical local gridmap (80×100 cells and three regions) is done in about 190 msec on a Pentium-II 400 MHz processor, including both segmentation and parameter extraction.

Experimental evidence indicates that the information provided by our virtual sensor is relatively stable with respect to transient changes in the geometry of the rooms, e.g., due to a different configuration of the furniture. Robustness could be further improved by associating each node transition with a value that indicates the strength of that transition. Another extension could be to increase the set of parameters computed for each node, in order to better discriminate between similar rooms.

The proposed virtual room sensor has a number of potential applications, some of which have been mentioned in the Introduction. In our work, we will use this sensor for two tasks: (i) to incrementally build a topological map of an indoor environment as a graph of rooms and corridors; and (ii) to perform coarse self-localization on this map. For point (ii) we are considering the use of Markov techniques [2].

More specifically, assume we have a topological map M where each node N represents a room and has some attributes associated with it. We denote by $Bel_t(N)$ our belief of being in node N at time t. This belief can be updated, whenever we detect a node transition a or we measure the parameters r, by Bayes rule:

$$Bel_{t+1}(N) = K\, p(r|N) \sum_{N' \in M} p(N|a, N') Bel_t(N')$$

where K is a normalizer, $p(r|N)$ is the probability of measuring parameters r when in node N, $p(N|a, N')$ is the probability of being at node N given that we were at node N' and detected a transition a. The a and r measures needed in order to perform the above update are precisely the information provided by the virtual sensor proposed here.

7 Acknowledgement

This work was partly supported by the Swedish KK Foundation. Thanks to Tom Ducket for insightful discussions and valuable advices.

References

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

[2] A.R. Cassandra, L.P. Kaelbling, and J.A. Kurien. Acting under uncertainty: Discrete bayesian models for mobile robot navigation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1996.

[3] J. D. Courtney and A. K. Jain. Mobile robot localization via classification of multisensor maps. In *Proc. of the IEEE Inte. Conf. on Robotics and Automation*, pages 1672–1678, 1994.

[4] 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*, pages 257–280. Physica/Springer-Verlag, 2001.

[5] E. Fabrizi and A. Saffiotti. Extracting topology-based maps from gridmaps. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2972–2978, San Francisco, CA, 2000. Online at http://www.aass.oru.se/~asaffio/.

[6] E. Fabrizi and A. Saffiotti. Augmenting topology-based maps with geometric information. *Robotics and Autonomous Systems*, 2002. Online at http://www.aass.oru.se/~asaffio/.

[7] J. A. Janet, D. Schudel, and R. C. Luo. Global self localization for actual robots: Generating and sharing topological knowledge using the region feature neural network, 1996.

[8] Patric Jensfelt and Henrik I. Christensen. Pose tracking using laser scanning and minimalistic environmental models. *IEEE Transactions on Robotics and Automation*, 17(2):138–147, April 2001.

[9] L. Najman and M. Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE T. on Pattern Analysis and Machine Intelligence*, 18(12):1163–1173, 1996.

[10] G. Oriolo, G. Ulivi, and M. Vendittelli. Fuzzy maps: a new tool for mobile robot perception and planning. *J. of Robotic Systems*, 14(3):179–197, 1997.

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

[12] K. Sarachik. Characterising an indoor environment with a mobile robot and uncalibrated stereo. In *Proc. IEEE Conf on Robotics and Automation*, pages 984–989, 1989.

[13] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

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