Extracting Structure of Buildings using Layout Reconstruction

Matteo Luperto and Francesco Amigoni

Abstract— **Metric maps, like occupancy grids, are the most common way to represent indoor environments in mobile robotics. Although accurate for navigation and localization, metric maps contain little knowledge about the structure of the buildings they represent. However, if explicitly identified and represented, this knowledge can be exploited in several tasks, such as semantic mapping, place categorization, path planning, human robot communication, and task allocation. The *layout* of a building is an abstract geometrical representation that models walls as line segments and rooms as polygons. In this paper, we propose a method to reconstruct two-dimensional layouts of buildings starting from the corresponding metric maps. The method identifies the representative lines along which the walls are aligned and uses these lines to segment the area in smaller parts, which are finally clustered in rooms. In this way, our method is able to find regularities within a building, abstracting from the noisy information of the metric map. Experimental results show that our approach performs robustly on different types of input metric maps, with noise, clutter, and partial data.**

I. Introduction

Understanding the environments in which they operate is an important capability for autonomous mobile robots. Buildings are strongly structured environments that are often organized in regular patterns. Metric maps, like grid maps [1], which are the usual environment representation employed in mobile robotics, do not explicitly contain any knowledge about the building structure, but represent only the space occupation for navigation and localization purposes. However, the identification of the building structure is useful for several tasks, such as semantic mapping, place categorization, path planning, human robot communication, and task allocation [2]. For instance, knowledge about the building structure can be useful to efficiently spread the robots to incrementally build the map of an initially unknown environment, in the context of multirobot exploration [3].

The *layout* of a building is a geometrical representation of the building's walls and rooms. Each room is represented either by a polygon (in 2D) or by a box model or a set of planes (in 3D). Walls are accordingly represented as line segments or planes. A layout of a building thus represents rooms that compose the building disregarding clutter relative to furniture, noise, and missing data, which are often affecting metric maps. *Layout reconstruction* is the task of retrieving the layout from a metric representation of the environment and can be performed at room or at floor level, starting from 2D grid maps [4] and from perfectly-aligned 3D point clouds [5], [6].

Authors are with the Artificial Intelligence and Robotics Laboratory, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy {matteo.luperto, francesco.amigoni}@polimi.it.

In this paper, we propose a method that reconstructs the layout of buildings starting from 2D metric maps. The method derives a more abstract representation of the structure of buildings in order to reason on a "clean" and stable knowledge. The proposed method identifies the *representative lines* along which the walls of a building are aligned and uses these lines to segment the area in smaller parts, called *faces*, which are finally clustered in rooms. The representative lines allow to find regularities between parts of the same building; for instance, two rooms placed at the opposite sides of the building can have aligned walls or a long corridor can connect rooms with the same shape and sharing the same wall. Our method is executed offline and can be applied to different input metric maps, either obtained by a robot or received from an external source: complete grid maps, partial grid maps, evacuation maps, and blueprints.

To show a possible use of our method, we consider a setting where a robot is exploring an unknown building. Our method is able to process the partial grid map built so far, to distinguish between fully and partially explored rooms, and to provide a rough estimate of the geometrical shape and size of the latter ones.

II. Related Work

Automatic analysis of representations of floors of buildings in order to extract structural information is an interdisciplinary topic addressed in different research fields, such as robotics, architecture, computer vision, and image analysis. While an exhaustive survey is out of the scope of this paper, in this section we cover a significant sample of methods, with a particular focus on those developed for mobile robots.

Room segmentation separates a metric map into parts, each one corresponding to a different room. Recently, the authors of [2] presented a survey of room segmentation techniques for mobile robots. They identify four main families of room segmentation approaches, namely Voronoi-based partitioning, graph partitioning, feature-based segmentation, and morphological segmentation. According to the survey, the most popular approach for room segmentation is the *Voronoi-based partitioning*. Given a metric map, the Voronoi diagram is composed of points with maximal distance from at least two points belonging to the closest obstacles. As an example, in [7], authors segment the map using *critical points*. Critical points are the points of the Voronoi diagram that are closer to obstacles than their neighbouring points, and can be used to identify narrow passages such as doorways. Critical points are used to segment the environment into a number of regions, which are later merged with their adjacent regions according to different heuristics.

Segmentation using *graph partitioning* techniques is used in [8], where a topological map is built incrementally using spectral clustering techniques. A similar approach, using Normalized Cut for partitioning, is used in [9], [10].

Feature-based segmentation techniques perform segmentation by identifying local features of the metric map directly from sensor data, by labeling their locations, and by propagating the labels to other locations in a smoothing process. An example is [11], where an AdaBoost classifier is applied to a feature vector that describes the shape of a single laser scan. The labeled scans are used to determine the labels of the cells of a grid map using an associative Markov network. Adjacent cells with the same label are considered as a single room. This framework can be combined with Voronoi segmentation [12]. In [13], a 2D map of an environment is divided into a set of small units called places. Segmentation and labeling is performed using an energy maximization framework which finds clusters of places and their labels such that each label appropriately describes the functional features of the associated place cluster.

An example of *morphological segmentation* techniques is that of [14], where a fuzzy-morphological operator is applied to divide areas that are connected by narrow passages in a fuzzy grid map.

Methods representing the four above families of approaches for room segmentation are evaluated and compared in [2]. No method clearly outperforms the others, although Voronoi-based segmentation techniques appear to be the most accurate.

An approach for room segmentation that shares some similarities with our layout reconstruction approach can be found in [15]. Similarly to our method, Canny edge transform and Hough line transform are used to extract lines from a grid map. The main difference from our method is that, while we extract a small set of representative lines that are used for identifying walls, in [15] a larger set of lines are used for obtaining a scalable grid representation of the environment, similar to an octo-map and called *A-grid*, which is eventually used to perform segmentation. Moreover, differently from our method, that of [15] does not perform layout reconstruction.

Authors of [5] present a method that performs room segmentation from a 3D point cloud perfectly aligned to a reference system. Points are projected on the floor plan, in order to find walls as sets of points whose projections are close to each other. The entire floor is segmented in areas using these projections of walls. Areas that are adjacent but not separated by a "peak-gap-peak" pattern in the projected points distribution are merged in the final segmentation. A similar solution, where the area is segmented in triangular regions that are later merged together can be found in [16].

Our approach differs from room segmentation methods, which typically partition the metric maps in rooms, because it extracts from the metric map a more abstract representation in which rooms are modeled using geometrical primitives. In our layout representation, a room is a polygon, while, in room segmentation, a room is a set of cells of the grid map.

Similarly to our work, [4] proposes a method that segments a metric map of an indoor environment while reconstructing its layout. It uses a framework based on a Markov Logic Network and data-driven Markov Chain Monte Carlo (MCMC) sampling. Using MCMC, the system samples many possible semantic worlds (layouts of the environment) and selects the one that best fits the sensor data. Differently from our work, which identifies the building structure by analyzing the entire metric map, each transition from a state to another state in the MCMC is based on local edit operations on the layout of a single room, ignoring other parts of the building.

Most of the above methods , in particular those developed outside the field of robotics, use a perfectly-aligned collection of 2D (or 3D) point data, as for example acquired by laser range scanners. There are also methods for *scene understanding* that use visual data to reconstruct the layout of a single room. In [17], vertical planes representing walls in a candidate room layout are fitted to a 3D point cloud obtained from SLAM (using a monocular camera). A particle filter updates the candidate room layouts until the final one is extracted as that with maximum likelihood.

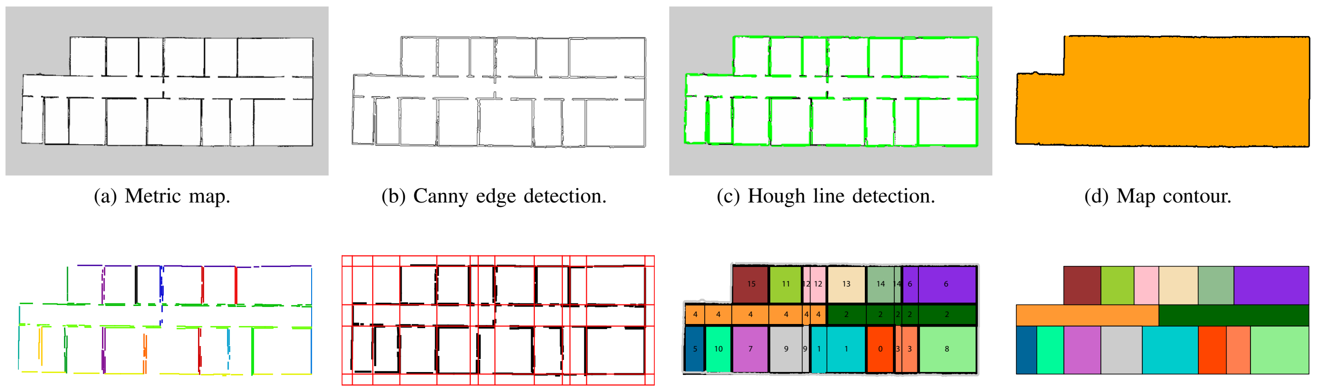
In [18], the layout of a cluttered room is recovered by extracting straight lines from images and grouping them according to three mutually orthogonal vanishing points. Vanishing points are used to generate candidates for the box layout. The more robust box layout estimation is then selected according to edge-based image features.

III. Our Method

The method we propose in this paper starts from a metric map, identifies the walls in it, and uses them for finding rooms and for reconstructing the layout of the environment. The method is composed of a number of steps executed sequentially, sketched in Algorithm 1 and detailed in the following with the help of a running example (Fig. 1). Some of the steps are inspired by the approach of [6]. More precisely, we use the method developed in [6] for dividing the metric map in smaller parts (which are used to identify rooms) using a set of lines. However, differently from our work, [6] reconstructs the 3D layout of a building from point clouds aligned and registered in the same coordinate system and knowing the exact number and location of rooms.

The starting point is a metric map M representing an indoor environment, typically a floor of a building. Without loss of generality, we assume that M is a 2D grid map, as the one in Fig. 1a, namely a two-dimensional matrix of cells (pixels), each one representing the probability that the corresponding area is occupied by an obstacle. The metric map M can be obtained from different sources, such as map building (SLAM) algorithms, blueprints, and evacuation maps, as described in Section IV.

The first operation on M is the detection of significative edges using the Canny edge detection algorithm [19], which partitions the cells of M into free cells and obstacle cells. The binary metric map resulting from the application of the Canny edge detection algorithm is called M' and an example is shown in Fig. 1b.



(a) Metric map. (b) Canny edge detection. (c) Hough line detection. (d) Map contour.

(e) Clustering of line segments. (f) Representative lines and faces. (g) Clustering of faces (h) Reconstructed layout.

Fig. 1: An example run of our method.

```
Input: a grid map $M$
Output: the reconstructed layout $\mathcal{L}$
/* Compute features from the map */
$M' \leftarrow CannyEdgeDetection(M)$
$S \leftarrow pHoughLineTransform(M')$

/* Obtain contour of the map */
$M'' \leftarrow thresholdMap(M, q)$
$Inner \leftarrow computeMapContour(M'')$

/* Create clusters of collinear segments */
$\mathcal{C} \leftarrow meanShiftClustering(S)$
$\mathcal{C}' \leftarrow spatialClustering(\mathcal{C}, line\_distance\_threshold)$

/* Find faces from representative lines */
$lines \leftarrow getsRepresentativeLines(\mathcal{C}')$
$F \leftarrow findFaces(lines)$

/* Compute spatial affinity between faces */
$L \leftarrow computeAffinityMatrix(F)$
$D \leftarrow diag(\sum_{j=1}^{n} L_{i,j})$
$A \leftarrow D^{-1}L$

/* Remove external faces outside border */
for $f \in F$ do
    if $area(f \cap Inner) < \delta$ then
        $F \leftarrow F \setminus f$
    end
end

/* Cluster together faces into rooms */
$\mathcal{L} \leftarrow DBSCAN(F, A, \epsilon, minPoints)$
return $\mathcal{L}$
```

Algorithm 1: Our method for layout reconstruction.

The set of edges (obstacle cells) is then processed by a probabilistic Hough line transform algorithm [20] to detect the line segments S that approximate the edges in M'. Fig. 1c shows such line segments in green. Then, the contour of the map is obtained using the contour detection algorithm of [21]. The contour is obtained from the original map M after the application a threshold q that divides cells in free or occupied. Fig. 1d shows in yellow the area inside the map border.

The line segments S are then clustered together according to the angular coefficients of their supporting lines using the mean shift clustering algorithm [22]. At the end of the angular clustering, we obtain a set of clusters $\mathcal{C} = \{C_1, C_2, \ldots\}$ such that each $C_j \subseteq S$ and $C_1 \cup C_2 \cup \ldots = S$. Each cluster C_j represents the set of line segments with similar angular coefficient α_j.

Next, the line segments belonging to the same angular cluster C_j are further clustered according to their spatial separation. Consider two line segments s and s' belonging to an angular C_j with angular coefficient α_j and call l and l' the lines passing through the middle points of s and s' and with angular coefficient α_j. If the distance between the parallel lines l and l' is less than a threshold (set, after some initial tests, to 90 cm in our experiments), then s and s' are put in the same spatial cluster. At the end of this step, we have a set of clusters $\mathcal{C}' = \{C_{1,1}, C_{1,2}, \ldots, C_{2,1}, C_{2,2}, \ldots\}$, such that $C_1 = C_{1,1} \cup C_{1,2} \cup \ldots$ and $C_2 = C_{2,1} \cup C_{2,2} \cup \ldots$, and so on. Fig. 1e shows the results of the spatial clustering where the line segments belonging to the same cluster in $\mathcal{C}'$ are displayed with the same color.

At this point, for each cluster $C_{j,k}$, a *representative line* $l_{j,k}$ that represents all the line segments in $C_{j,k}$ is determined. This representative line is computed as the line with angular coefficient α_j associated to C_j and that passes through the median of the set of middle points of the line segments in $C_{j,k}$. Each representative line, in red in Fig. 1f, indicates the direction of a wall within the building. The intersections between all lines divide the area of map M' into different areas, called *faces*, as shown in Fig. 1f. We call F the set of faces.

Rooms are determined by grouping faces together. Adjacent faces that are separated by an edge corresponding to a wall should belong to different rooms, while adjacent faces that are separated by an edge not corresponding to any wall should be grouped together in the same room. More precisely, for each pair of faces f and f' that share a common edge, we compute a weight $w(f, f')$ as follows. Given an edge $e_{f,f'}$ shared by two faces f and f' (and belonging to the representative line $l_{j,k}$ of a spatial cluster $C_{j,k}$), its weight is calculated (as described in [6]) as $w_{f,f'} = cov(e_{f,f'})/len(e_{f,f'})$, where $len(e_{f,f'})$ is the length of $e_{f,f'}$ and $cov(e_{f,f'})$ is the length of the projections, on $e_{f,f'}$,

A_{BC}	$87.8\% \pm 4.5\%$	A_{FC}	$87.6\% \pm 5.7\%$

TABLE I: Layout reconstruction results on 20 grid maps obtained from gmapping.

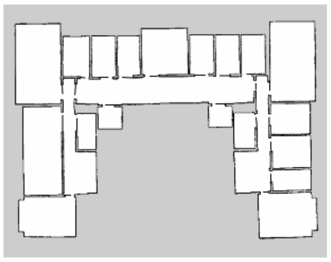
of the line segments in $C_{j,k}$. The larger the weight $w_{f,f'}$, the stronger the hypothesis that there is a wall (obstacle) along $e_{f,f'}$ (namely, between faces f and f'). If an edge is completely covered by projections of line segments in $C_{j,k}$, then its weight is 1. Following the definition of [6], weighted edges are used to compute an affinity measure L between all pairs of faces. L is similar to a Laplacian, and its entries $L_{f,f'}$ are defined as:

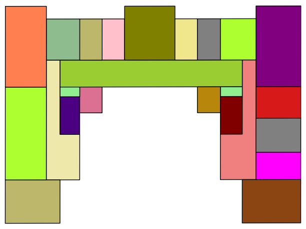
$$L_{f,f'} = \begin{cases} e^{-w(f,f')/\sigma} & \text{if } f \neq f' \text{ and } f \text{ and } f' \text{ are adjacent} \\ 1 & \text{if } f = f' \\ 0 & \text{otherwise} \end{cases}$$

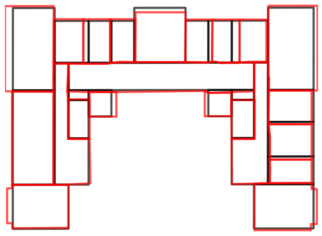
where σ is a regularization factor. From matrix L, a local affinity matrix A is defined as $A = D^{-1}L$, with $D = diag(\sum_{j=1}^{n} L_{i,j})$, where n is the number of faces in F. Each element $A_{f,f'}$ indicates an affinity value considering the local connectivity between faces f and f'. The matrix A is used as input for DBSCAN [23], which clusters faces. DBSCAN groups together faces that are close to each other in a dense portion of the feature space represented by matrix A. (In our experiments, we set the two DBSCAN parameters to $\epsilon = 0.85$ and *minPoints*$= 1$.)

Before applying DBSCAN, some faces are discarded. Specifically, discarded faces are those called *external* and such that the area of their intersection with the inner area of M (obtained from the contour) is smaller than a threshold δ. Remaining faces are called *partial* if they are adjacent to an external face via an edge whose weight is less than a threshold (0.2 in our experiments) and *internal* otherwise. Partial faces cover the area of rooms that are not fully known according to the data collected in M, as for example during the exploration of a building. Partial faces contain *frontiers*, namely boundaries between explored and unexplored parts of the environment. Frontiers are present due to the limited range of sensors [24]. Examples of partial faces can be seen, highlighted in gray, in Fig. 5.

Then, DBSCAN is applied to internal faces and a set of clusters $R = \{F_1, F_2, \ldots\}$ of F is obtained. Each cluster F_i corresponds to a room r_i, which is represented as a polygon obtained by merging together all the faces belonging to F_i. Fig. 1g shows the results of DBSCAN applied to our example map. The number i in each face indicates its cluster F_i. Different clusters are displayed with different colors. The final reconstructed layout $\mathcal{L} = \{r_1, r_2, \ldots\}$ is finally displayed in Fig. 1h. Note that it is a "clean" and abstract representation of the original grid map of Fig. 1a, which nevertheless retains the main structural features.


(a) Grid map.


(b) Reconstructed layout.


(c) Reconstructed layout (black) and ground truth (red).

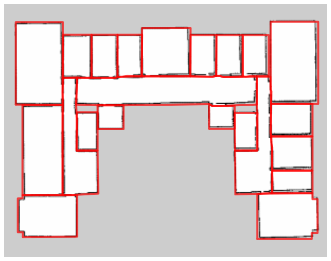

(d) Grid map and ground truth (red).

Fig. 2: An example of a layout reconstruction.

no furniture	recall	$90.0\% \pm 4.1\%$
	precision	$90.1\% \pm 6.3\%$
furniture	recall	$89.5\% \pm 6.2\%$
	precision	$94.0\% \pm 2.2\%$

TABLE II: Layout reconstruction results on [2] datasets.

IV. Experimental Results

In this section, we evaluate our method for reconstructing the layout of a building from a metric map. Our method is implemented in Python[1] and experiments are run on a single core of an Intel Core i7-3610QM@2.30 GHz CPU with 8 GB RAM. For a single metric map, layout reconstruction takes approximatively 12 seconds on average.

A. Results with Grid Maps from Stage

A reconstructed layout is expected to present two main characteristics: (1) all the rooms of the real building (and only them) should be in the layout; (2) the shape of each reconstructed room should match that of its real counterpart. Evaluation is performed both visually and quantitatively, comparing the reconstructed layout $\mathcal{L}$ and a ground truth layout Gt. Following the approach of [2], we introduce two matching functions between rooms in $\mathcal{L}$ and in Gt, namely *forward coverage* FC and *backward coverage* BC. FC represents how well the reconstructed layout $\mathcal{L}$ is described by the ground truth layout Gt, while BC represents how well the ground truth layout Gt is described by the reconstructed layout $\mathcal{L}$. Specifically:

$$FC : r \in \mathcal{L} \mapsto r' \in Gt$$

$$BC : r' \in Gt \mapsto r \in \mathcal{L}$$

[1]Code and datasets are available upon request.

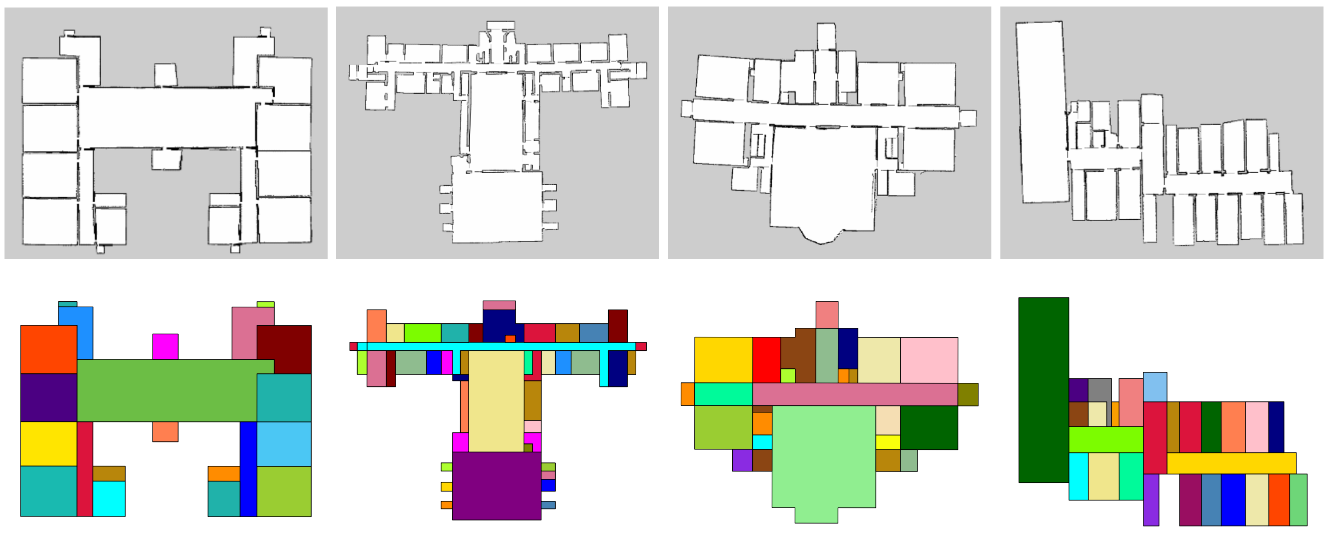


Fig. 3: Examples of layout reconstructions.

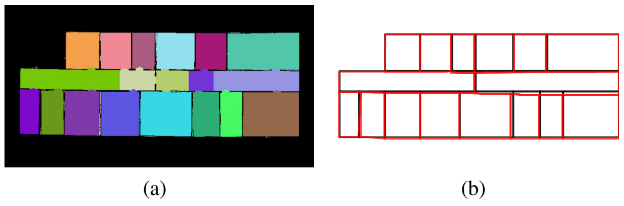


Fig. 4: Segmentation by a Voronoi-based approach, from [2] (a) and comparison between our reconstructed layout (black) and the ground truth layout (red) (b) for the metric map of Fig. 1a.

For each room $r \in \mathcal{L}$, FC finds the room $r' \in Gt$ that maximally overlaps r; conversely, for each room $r' \in Gt$, BC finds the room $r \in \mathcal{L}$ with the maximum overlap with r'. Calling *area*() a function that computes a polygon area, the overlap between a room $r \in \mathcal{L}$ and a room $r' \in Gt$ is defined as $area(r \cap r')$. Matching functions FC and BC are used for computing two measures of accuracy called *forward accuracy* A_{FC} and *backward accuracy* A_{BC}:

$$A_{FC} = \frac{\sum_{r \in \mathcal{L}} area(r \cap FC(r))}{\sum_{r \in \mathcal{L}} area(r)}$$

$$A_{BC} = \frac{\sum_{r' \in Gt} area(BC(r') \cap r')}{\sum_{r' \in Gt} area(r')}$$

The numbers of rooms in $\mathcal{L}$ and Gt can be different, due to over- or under-segmentation. Over-segmentation results in high A_{FC} and low A_{BC}, while under-segmentation results in high A_{BC} and low A_{FC}.

We consider 20 grid maps obtained running the ROS implementation[2] of the gmapping algorithm [25] on data collected by a robot equipped with a laser range scanner

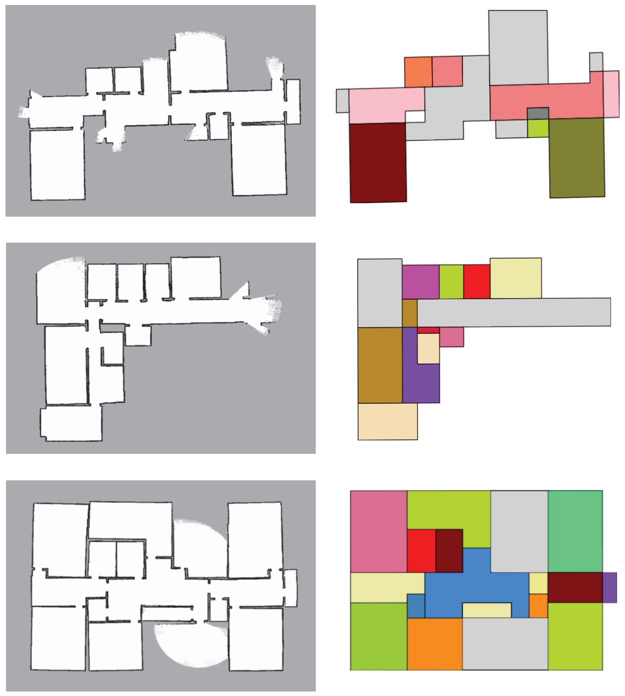


Fig. 5: Examples of layout reconstructions from partial grid maps (partially explored rooms are in gray).

moving in 20 school buildings simulated in Stage[3]. It is important to point out that the evaluation is performed by comparing the layouts reconstructed from the grid maps built by the robot and the *actual* layouts of the simulated buildings fed to Stage. This allows us to evaluate if our layout reconstruction approach is able to cope with noise and errors introduced in the mapping process. Layout reconstruction accuracy is reported in Table I. On the set of 20 worlds

[2]`http://wiki.ros.org/gmapping`

[3]`http://wiki.ros.org/stage`

mapped by the simulated robot, our method is able to reconstruct successfully and with good accuracy the layout of the original buildings. For example, Fig. 2 presents a grid map obtained by the robot (Fig. 2a) and the layout reconstructed using our method (Fig. 2b). For this particular example, we have $A_{FC} = 90.1\%$ and $A_{BC} = 93.3\%$. In Fig. 2c, the reconstructed layout (in black) and the ground truth building layout (in red) are superimposed. Although the layout is generally accurate, there all small differences in the geometry of rooms, due to approximations introduced by our method, which result in a slight performance degradation. In Fig. 2d the grid map and the ground truth building layout (in red) are superimposed. While the grid map provides a good representation of the environment, some inaccuracies, such as irregular gaps between walls, are present. Our method tries to filter those inaccuracies when reconstructing the layout.

Fig. 3 shows four other examples of reconstructed layouts from grid maps. In all the four cases, our method is able to correctly reconstruct the layout of the environment, coping well with some alignment and rotation errors and gaps between rooms in the metric maps (e.g., see the third grid map). Few inaccuracies are introduced, like long corridors split into smaller units, thus producing over-segmentation. Another error is made sometimes when a there is a small gap within the building (e.g., due to large wall or a pillar), which is misclassified as occupied face and subsequently added to an adjacent room or considered as a small independent room. Note that our method can successfully retrive the layout of large-scale buildings, as can be seen from the second example of Fig. 3, where the rooms connected to the main corridor (light blue) are the classrooms of a high school.

Since our method does not assume a Manhattan world it can be potentially used in non-Manhattan environments, such as those with round or diagonal walls. However, since in our method walls are approximated by straight lines, round walls are approximated by a polyline (see, again, the third example of Fig. 3). Other examples are shown in the video attachment.

Maps that present (relatively) small misalignments are adjusted by our method, as shown by the last two examples of Fig. 3. A trade-off exists between the accuracy of alignment that can be reached by our method and its ability to approximate round walls with polylines. This trade-off can be set by modifying the parameters of the line segment clustering step, where collinear walls are clustered together and representative lines are identified. (Being more tolerant in clustering line segments with different angular coefficients results in more alignment.) If alignment is not required, round walls can be effectively approximated by fine-grained polylines, with the side effect that line segments that are affected by alignment problems are not straightened up (and are wrongly recognized as diagonal walls). Similarly, strong regularization of unaligned maps results into a coarse approximation of round walls. In all the examples presented in this paper we preferred strong alignment over good approximation of round walls.

B. Results with Publicly Available Datasets

Here, we present the results of the evaluation of our method on the two datasets (each composed of 20 metric maps) used in [2]. The datasets contain metric maps with and without furniture, respectively, and, according to [2], are evaluated employing precision and recall metrics, which are defined similarly to A_{FC} and A_{BC}:

$$\mathit{Precision} = \frac{\sum_{r \in \mathcal{L}} \mathit{area}(r \cap FC(r))/\mathit{area}(r)}{|\mathcal{L}|}$$

$$\mathit{Recall} = \frac{\sum_{r' \in Gt} \mathit{area}(BC(r') \cap r')/\mathit{area}(r')}{|Gt|}$$

When the dataset with furniture is used, our method filters out some of the clutter introduced by furniture by automatically clustering and filtering out isolated obstacle cells using DBSCAN directly on the original map M (this is a very simple way to handle furniture, which can be definitely improved). Results, reported in Tab. II, are good and confirm those obtained with our dataset.

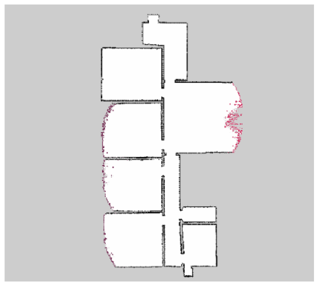
A comparison with the results obtained by room segmentation methods of [2] is unfair, because these methods directly partition the cells of the metric maps, while our method uses a more abstract representation based on representative lines and faces, thus introducing some approximations. An example of the approximations introduced by our layout reconstruction with respect to room segmentation is displayed in Fig. 4. Fig. 4a shows the Voronoi-based segmentation of the metric map of Fig. 1a, as reported in [2]. It can be observed that Voronoi-based methods tend to produce over-segmentation. In Fig. 4b, we compare our reconstructed layout (in black) with the real structure of the building (in red). Despite being able to correctly capture the building layout, for this map we have $A_{FC} = 93.6$ and $A_{BC} = 94.6$, due to some visible approximations. That said, comparing Table II with Table II of [2], we can say that our method produces results comparable (within 1 sigma) with those of the methods surveyed in [2], further confirming that the reconstructed layout actually captures the structure of the building. Moreover, the availability of the layout enables a number of possible tasks, as suggested in the following.

Additional results obtained on our dataset and on those used in [2] are reported in the video attachment.

C. Results with Partial Grid Maps

Fig. 5 shows three examples of layout reconstruction starting from partial grid maps. Partially explored rooms are identified correctly and marked in gray in the layout. The faces composing a partial explored room are added to the reconstructed layout, providing a guess of the unknown room appearance. If a partially explored room is not “closed” by any line segment found in other rooms, then the room is arbitrarily completed with a wall parallel to the closest line segment.

These results suggest that our method can be used for reconstructing the shape of partially seen rooms, on the basis of the layout reconstructed from of the rest of the building.


(a) Frontiers.

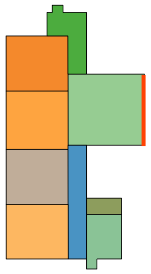

(b) Reconstructed layout.

Fig. 6: Frontiers (highlighted in Fig. 6a) in almost fully-explored rooms and frontiers that lead to unexplored parts of the building (in red in Fig. 6b) can be identified.

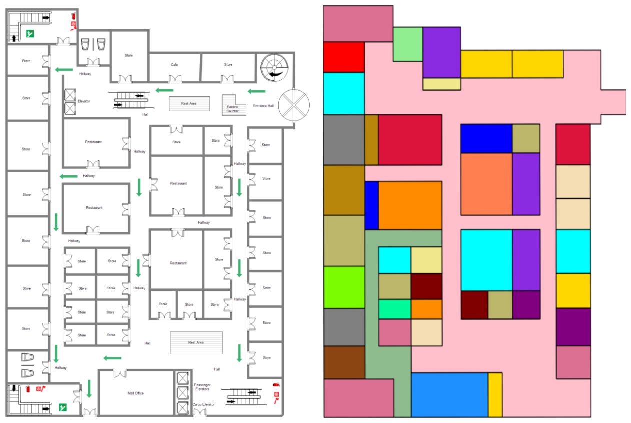


Fig. 7: An example of a reconstructed layout from a blueprint of a highly structured building.

Such knowledge can be potentially used for speeding-up exploration, as it can be seen from the example of Fig. 6. By reconstructing the shape of partial rooms, the three frontiers on the left are correctly identified as being inside almost fully explored rooms, while the frontier that appears to be more promising to continue the exploration is that on the right, which is in a room that is not closed by any representative line (highlighted in red). The entire layout of the building can be seen for reference in the first example of Fig. 3. This example shows a possible use of the outcomes of our method. A complete exploration method that exploits layout reconstruction is currently being developed and its description is out of the scope of this paper.

D. *Results with Blueprints and Evacuation Maps*

Our method can be used also for reconstructing the 2D layout of a building from its blueprint or evacuation map. Blueprints and evacuation maps are usually images in which the walls of buildings are represented together with symbols and words that explain the meaning and the locations of some features (e.g., the functions of the rooms and the presence of fire extinguishers). We pre-process these maps as follows. Words (like the name of the building or the functions of rooms) are recognized and filtered out using a standard OCR method (Tesseract). Doors and other symbols (like fire extinguishers and doors) are easily retrieved, since they are represented with standard symbols, by using template-matching algorithms.

The reconstructed layout of a building obtained from a blueprint can be a useful source of knowledge, especially when the environment in which robots operate is initially unknown, as in a situation where a team of robots perform search and rescue operations and an evacuation map of the environment can be seen on a wall. One of the possible uses of such knowledge is localization, as done in [26], where a robot can localize itself using a floor plan instead of a metric map. A reconstructed layout may introduce some approximations compared to the real shape of the environment, as explained in the previous section. However, [27], [28] show how partially inaccurate hand-drawn sketch maps can be effectively used by robots for localization. In principle, using similar approaches, a reconstructed layout of a building can be used for localization even if it is slightly inaccurate.

An example of a layout reconstructed from a blueprint is shown in Fig. 7. Other examples are reported in the video attachment. Quantitative results of layout reconstruction starting from partial grid maps and from blueprints (and evacuation maps) are similar to those of Table I.

E. *Results with a Grid Map Acquired by a Real Robot*

Finally, Fig. 8 shows the application of our method to a grid map obtained from the Radish repository [29]. We used the same parameter values set for the maps of the previous datasets (demonstrating the robustness of the approach). Clutter is initially and partially removed by filtering out isolated obstacle cells retrieved and clustered using DB-SCAN (as in Section IV-B). The figure clearly shows that our method is able to reliably reconstruct the layout of the environment in presence of noise and of cluttered maps. Note that we reconstruct the layout of partial rooms, like the one at the bottom left of the map. Moreover, we highlight with bold lines the reconstructed walls, namely the edges between faces whose weight $w(f, f')$ is close to 1.

By visual comparison of our result to those of [4] obtained on the same map of Fig. 8, it can be noticed that, while the method of [4] effectively reconstructs the layout of all fully-explored rooms (see Fig. 7 of [4] for reference), our method is able to estimate the layout of both fully-explored and partially-explored rooms, thus reconstructing the layout of the entire floor and not only of the explored map, and that representative lines are used to align together rooms that share one or more walls. Open doors in the map, which results in diagonal line segments close to the doorway, are aligned by our method to the closest representative lines. While the identification of walls and representative lines can be performed effectively in both real and simulated environments (without changing any parameter), the clustering of faces into rooms may results in under-segmentation when used on real-world maps, because partially-explored faces

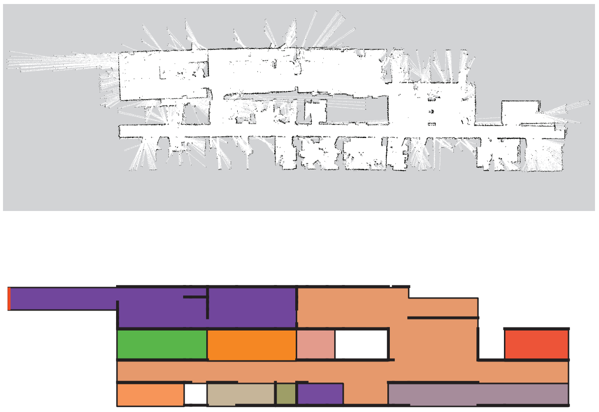


Fig. 8: An example of a reconstructed layout for a map from [29]. (Walls are in bold.)

are added to the closest room (as it happens to the corridor at the bottom of Fig. 8).

V. Conclusions

In this paper, we have presented a method for reconstructing the layout of a building given its metric map. Our method identifies lines that represent walls and use them to approximate the shape of the rooms. Experimental results show that our method performs well and is able to cope with complete and partial metric maps of large-scale buildings and also with blueprints and evacuation maps provided as input.

Future work includes the combination of our method with room segmentation approaches to investigate if better performance can be obtained by looking both at the global structural features and at the local features of the buildings. Moreover, we would like to more precisely assess the performance of our method in presence of clutter and noise in metric maps and to extend it to 3D point clouds, possibly integrating our representative lines in 2D with wall candidates identified as planes in 3D. Finally, we plan to use the reconstructed layouts for semantic classification of rooms and for predicting the layout of partially explored rooms.

Acknowledgements

The authors are glad to thank Matteo Calabrese and Valerio Arcerito for their contributions to the implementation of the system described in this paper.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[2] R. Bormann, F. Jordan, W. Li, J. Hampp, and M. Hägele, "Room segmentation: Survey, implementation, and analysis," in *Proc. ICRA*, 2016, pp. 1019–1026.

[3] A. Quattrini Li, R. Cipolleschi, M. Giusto, and F. Amigoni, "A semantically-informed multirobot system for exploration of relevant areas in search and rescue settings," *Auton Robot*, vol. 40, no. 4, pp. 581–597, 2016.

[4] Z. Liu and G. von Wichert, "A generalizable knowledge framework for semantic indoor mapping based on Markov logic networks and data driven MCMC," *Future Gener Comp Sy*, vol. 36, pp. 42–56, 2014.

[5] I. Armeni, O. Sener, A. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3D semantic parsing of large-scale indoor spaces," in *Proc. CVPR*, 2016, pp. 1534–1543.

[6] C. Mura, O. Mattausch, A. J. Villanueva, E. Gobbetti, and R. Pajarola, "Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts," *Comput Graph*, vol. 44, pp. 20–32, 2014.

[7] S. Thrun, "Learning metric-topological maps for indoor mobile robot navigation," *Artif Intell*, vol. 99, no. 1, pp. 21–71, 1998.

[8] E. Brunskill, T. Kollar, and N. Roy, "Topological mapping using spectral clustering and classification," in *Proc. IROS*, 2007, pp. 3491–3496.

[9] J.-L. Blanco, J. Gonzalez, and J.-A. Fernandez-Madrigal, "Consistent observation grouping for generating metric-topological maps that improves robot localization," in *Proc. ICRA*, 2006, pp. 818–823.

[10] S. Hemachandra, M. Walter, S. Tellex, and S. Teller, "Learning spatial-semantic representations from natural language descriptions and scene classifications," in *Proc. ICRA*, 2014, pp. 2623–2630.

[11] O. Mozos, *Semantic labeling of places with mobile robots*, ser. Springer Tracts in Advanced Robotics. Springer, 2010, vol. 61.

[12] S. Friedman, H. Pasula, and D. Fox, "Voronoi random fields: Extracting the topological structure of indoor environments via place labeling," in *Proc. IJCAI*, 2007, pp. 2109–2114.

[13] K. Sjoo, "Semantic map segmentation using function-based energy maximization," in *Proc. ICRA*, 2012, pp. 4066–4073.

[14] P. Buschka and A. Saffiotti, "A virtual sensor for room detection," in *Proc. IROS*, 2002, pp. 637–642.

[15] R. Capobianco, G. Gemignani, D. Bloisi, D. Nardi, and L. Iocchi, "Automatic extraction of structural representations of environments," in *Proc. IAS-13*, 2014, pp. 721–733.

[16] E. Turner, P. Cheng, and A. Zakhor, "Fast, automated, scalable generation of textured 3d models of indoor environments," *IIEEE J Sel Top Signa*, vol. 9, no. 3, pp. 409–421, 2015.

[17] A. Furlan, S. Miller, D. Sorrenti, F.-F. Li, and S. Savarese, "Free your camera: 3D indoor scene understanding from arbitrary camera motion," in *Proc. BMVC*, 2013.

[18] V. Hedau, D. Hoiem, and D. Forsyth, "Recovering the spatial layout of cluttered rooms," in *Proc. ICCV*, 2009, pp. 1849–1856.

[19] J. Canny, "A computational approach to edge detection," *IEEE T Pattern Anal*, no. 6, pp. 679–698, 1986.

[20] N. Kiryati, Y. Eldar, and A. M. Bruckstein, "A probabilistic hough transform," *Pattern Recogn*, vol. 24, no. 4, pp. 303–316, 1991.

[21] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Comput Vision Graph*, vol. 30, no. 1, pp. 32–46, 1985.

[22] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE T Pattern Anal*, vol. 24, no. 5, pp. 603–619, 2002.

[23] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Proc. KDD*, 1996, pp. 226–231.

[24] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proc. CIRA*, 1997, pp. 146–151.

[25] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with Rao-Blackwellized particle filters," *IEEE T Robot*, vol. 23, pp. 34–46, 2007.

[26] W. Winterhalter, F. Fleckenstein, B. Steder, L. Spinello, and W. Burgard, "Accurate indoor localization for rgb-d smartphones and tablets given 2D floor plans," in *Proc. IROS*, 2015, pp. 3138–3143.

[27] B. Behzadian, P. Agarwal, W. Burgard, and G. D. Tipaldi, "Monte Carlo localization in hand-drawn maps," in *Proc. IROS*, 2015, pp. 4291–4296.

[28] F. Boniardi, B. Behzadian, W. Burgard, and G. D. Tipaldi, "Robot navigation in hand-drawn sketched maps," in *Proc. ECMR*, 2015, pp. 1–6.

[29] A. Howard and N. Roy, "The robotics data set repository (Radish)," 2003. [Online]. Available: http://radish.sourceforge.net/