

Automatic modeling of cluttered multi-room floor plans from panoramic images

Giovanni Pintore¹, Fabio Ganovelli², Alberto Jaspe Villanueva¹, and Enrico Gobbetti¹

¹ Visual Computing, CRS4, Italy
² Visual Computing Group, ISTI CNR, Italy

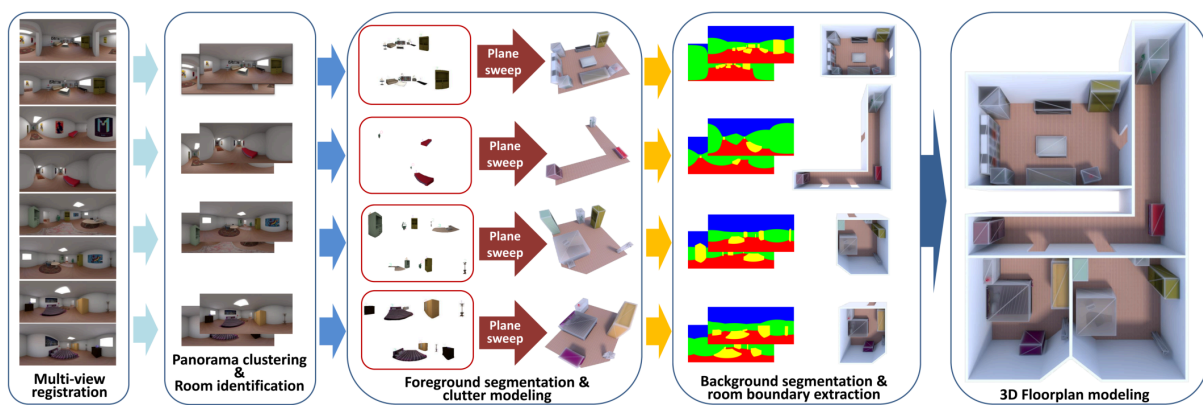


Figure 1: Overview. Starting from a small set of overlapping panoramic images of a multi-room environment, we perform multi-view registration. In parallel, we extract the masks of interior objects visible in each image. Using mutual visibility and photoconsistency information, we identify rooms. The pose and size of the interior objects of each room is determined using a plane-sweeping approach which jointly analyzes the set of panoramas and associated associated to the room. Finally, all the recovered information is exploited for the extraction of room boundaries and interconnections, leading to a structured indoor model in terms of rooms bounded by walls, ceiling, and floors and containing a set of objects described in terms of their bounding volumes.

Abstract

We present a novel and light-weight approach to capture and reconstruct structured 3D models of multi-room floor plans. Starting from a small set of registered panoramic images, we automatically generate a 3D layout of the rooms and of all the main objects inside. Such a 3D layout is directly suitable for use in a number of real-world applications, such as guidance, location, routing, or content creation for security and energy management. Our novel pipeline introduces several contributions to indoor reconstruction from purely visual data. In particular, we automatically partition panoramic images in a connectivity graph, according to the visual layout of the rooms, and exploit this graph to support object recovery and rooms boundaries extraction. Moreover, we introduce a plane-sweeping approach to jointly reason about the content of multiple images and solve the problem of object inference in a top-down 2D domain. Finally, we combine these methods in a fully automated pipeline for creating a structured 3D model of a multi-room floor plan and of the location and extent of clutter objects. These contribution make our pipeline able to handle cluttered scenes with complex geometry that are challenging to existing techniques. The effectiveness and performance of our approach is evaluated on both real-world and synthetic models.

CCS Concepts

• **Computing methodologies** → **Computer graphics; Shape inference; Reconstruction;**

1 Introduction

Creating high-level structured 3D models of indoor scenes from captured data is a fundamental task in many fields [BTS*17]. The

structures of interest are of different kinds. Several applications, such as the generation or update of building information models (BIM) mostly focus on determining the bare architectural structure in terms of room walls, floors, and ceilings [MMJV*14, TCZ15].

By contrast, information on the interior clutter, in terms of 3D footprint of major indoor objects is also required for many other use cases, such as guidance, energy management, security, evacuation planning, location awareness and routing [IYF15].

A wide variety of solutions exists for capturing 3D information on indoor environments, ranging from mobile laser scanners to active depth sensors. However, purely image-based methods became very appealing since the inception of inexpensive mobile cameras, like those on modern smartphones. The visual capturing process is particularly fast, simple and cost-effective when exploiting emerging 360° cameras, since a good coverage of a complex environment generally requires very few shots, and such a panoramic coverage provides a visual representation readily usable in navigation applications [PGGS16].

Inferring indoor structure just from visual data is, however, not an easy task, due to the many ambiguities resulting from sparse coverage, occlusions, and lack of visual detail. The topic has thus been the focus of much research in the past decade. Current solutions, however, still have important limitations. In particular, methods based on dense multi-view approaches need a texture-rich environment and generally require many images, while methods working with sparse sampling typically require manual intervention or impose severe constraints on the shape and structure the environment and of the clutter (see Sec. 2).

Our approach In this work, we propose a novel light-weight approach to compute, from a small set of registered panoramic images, a multi-room 3D layout in terms of room boundaries and 3D bounding volumes of all major objects (see Sec. 3). We use mutual visibility information and photoconsistency to create an interconnection graph between poses in order to split the image set in different room groups. We exploit this graph to support interior object identification, room identification and room boundaries extraction. The pose and size of the clutter objects in each room are recovered by starting from a per-image segmentation that identify the *masks* of indoor objects, and then using a *virtual plane sweeping* approach to jointly perform object inference in a top-down 2D domain using all the images associated to a room. The resulting 3D clutter model of all rooms, in terms of image mask, position, orientation, and dimensions of each object, is then exploited to enhance image segmentation and geometric context reasoning for the room identification and room geometry extraction phases. As a result, the final model is partitioned into interconnected rooms bounded by walls, ceiling, and floors and containing a set of objects described in terms of their bounding volumes.

Contribution At the system level, we contribute a novel approach extending and combining in a non-trivial way several state-of-the-art solutions for indoor reconstruction from sparse panoramic images. We also introduce the following novel specific techniques:

- We introduce a photo-consistency approach to order and group a sparse set of panoramic images in a connectivity graph. The core idea of our method is to detect rooms by clustering nodes in a fully connected graph, whose edges are weighted by the similarity among images under a specially crafted warping transformation. This grouping improves both object recognition and

room structure identification, filtering undesired contributions from other images, such as, for example, images too far from the object or parts of other environments visible through open doors. Compared to previous approaches [PGP*18, CF14], which try to roughly infer walls position from the sparse input 3D points to estimate space partitioning, our approach provides more flexibility and robustness (see Sec. 7).

- We introduce a plane sweeping approach to solve the problem of object inference in a top-down 2D domain starting from single-image cues. To do this, we define a specific *parameterization* and a novel *loss function*, which are used to transform in the same model space the contribution of different images and to evaluate object hypotheses. This approach allows us to exploit cues from different images, improving inference performance with respect to single-image methods based on cuboids [ZSTX14]. Moreover, the approach is designed to work with an extremely limited number of images per object (e.g. 2-3), without involving the dense scene coverage required by other methods [IYF15, BFFFS14].
- We exploit the model of foreground clutter and the image grouping to enhance image segmentation and to complete room geometry extraction in terms of walls, floor and ceilings. As demonstrated by our results (Sec. 7), this extra information leads to the generation of a more accurate and complete model with respect to current methods for indoor reconstruction from panoramic imagery (e.g., [CF14, YZ16, YJL*18, PGP*18]), which are mostly based on background segmentation via super-pixels.

2 Related Work

3D reconstruction and modeling of indoor scenes has attracted a lot of research in recent years. Reconstructed model characteristics are application-dependent, ranging from purely geometric to fully semantic reconstructions, as well as scale-dependent, ranging from single rooms to large-scale scenes [IYF15]. In this work, we focus on pipelines for analyzing small sets of images to generate structured geometric abstractions of multi-room environments with clutter [HDGN17, ZCC16].

From the capture point of view, many works require a fairly dense 3D point cloud of the environment. While in the past this was only possible with costly laser scanners, this approach is becoming more widespread due to the emergence of new sensors, including mobile RGB-D sensors. The methods, however still require a lot of post-processing to extract structured models from raw data [MMP16]. When it comes to the construction of a real 3D model (e.g., a mesh), existing methods typically produce a set of planar patches at a room scale [XAAH13], simple primitives for a part of a scene [GPMAL09], a dense mesh from a voxel grid [TCZ15] or a polygon soup without any structure or semantics [XF14]. For large scenes, current state-of-the-art methods solve room segmentation and reconstruction in a top-down 2D domain [TCZ15, MMP16]. A prominent example is the work of Ikehata et al. [IYF15], which propose a 3D modeling framework that reconstructs an indoor scene as a structured model exploiting panoramic RGB-D images.

In this work, we focus on purely image-based techniques, which are gaining popularity in several domains, since they are based on widely available and low-cost sensors. Even though, at least in

certain situations, the accuracy of dense image-based methods has shown to be competitive with laser sensor systems at a fraction of the cost [SCD*06], the lack of explicit 3D information requires aiding reconstruction by imposing domain-specific constraints. For example, several authors exploit the heavily constraining *Manhattan World* assumption to reconstruct the 3D structure of moderately cluttered interiors e.g., [FCSS09, FMR11, TXLK11]) or the 3D footprint of interior objects (e.g., [LGHK10, HHF12]).

A number of authors have focused on joint estimation of room shape and object location in the single-view case, typically to infer room layouts from a single image. A classic approach, achieving good success with interiors containing large pieces of furniture, is to analyze the scene by fitting 3D cuboid models [LGHK10, HHF10, HHF12]. These methods have been extended through Markov chain Monte Carlo (MCMC) sampling of part-based 3D object models [DBK*13] in order to achieve more accurate recovery of fine structures. Schwing et al. [SFPU13] used a branch-and-bound method to jointly infer 3D room layout and objects that are aligned with the dominant orientations, while Satkin et al. [SRLH14] proposed a top-down matching approach to align 3D models from a database with an image. The latter method employs multiple features to match 3D models to images, including pixel-wise object probability, estimated surface normals, and image edges. CNNs have also been used for the same purpose, as in the work by Su et al. [SQLG15], in which a CNN was trained for pose estimation using rendered models of 12 object categories from the PASCAL 3D dataset [XMS14], or in the work of Tulsiani et al. [TKCM16], which combine object localization and reconstruction from a single image using CNNs for detection, segmentation, and view estimation. These single-image methods are promising, but strictly limited to very small scenes, visible from a single point-of-view, and containing a limited number of object categories.

Recent state-of-the-art have extended single-image analysis to omnidirectional images. Some approaches exploit a super-pixel segmentation and an analysis of edges to recover room layout [YZ16] and depth estimation of the whole panoramic image [YJL*18]. These methods are limited to Manhattan-world environments and do not return a structured model. Recent data-driven approaches [ZCSH18, YWP*19] have also demonstrated success in recovering the 3D boundary of a single uncluttered room meeting the Manhattan World constraint, or to infer the whole context of a cluttered room containing a limited set of object categories [XSKT17, ZSTX14].

Multi-room environments typically require the joint analysis of images taken from multiple points of view. Bao et al. [BFFFS14] apply both single-view and multi-view reasoning to extend the number of recognized categories, but, in contrast to our work, focuses on small scenes (i.e., room corners) and requires using a large number of pin-hole images (at least 10 images).

Cabral et al. [CF14] adopted stitched equirectangular images to improve indoor reconstruction provided by a dense multi-view pipeline [FCSS09]. As clutter and homogeneous zones in indoor scenes tend to leave large reconstruction holes for image-based methods, their method exploits the labeling of the panoramas to complete the multi-view reconstruction obtained from pin-hole images. However, such an approach required a considerable number of images

and a dense point cloud, in addition to considerable efforts in terms of user interaction and processing time. Sharing the same simplified segmentation of Cabral et al. [CF14] (i.e., wall, ceiling and floor), Pintore et al. [PGP*18] recover the 3D layout of multi-room floorplans from a set of spherical images without involving externally calculated 3D data, by combining sparse multi-view features from images registration and single image analysis. In the same way, most panoramic imagery methods [YZ16, CF14, PGP*18, PPG*18] base image segmentation on color homogeneity of indoor structures, a reasonable assumption for boundary structure but not for foreground objects.

In this work, we improve over these approaches, and in particular over the approach of Pintore et al. [PGP*18] in several ways. First, we automatically group sets images per room through photoconsistency analysis rather than occlusion analysis, thus reducing the risk of creating a single footprint for connected but distinct rooms in case of sparse sampling or non-textured environments which do not generate enough 3D features for visibility computation. Secondly, we segment clutter on the images, and therefore implicitly improve the background segmentation in addition to generating a richer model. Finally, we significantly improve the accuracy of room shape recovering, by effectively merging multiple information sources coming from the room segmentation and the clutter analysis.

3 Overview

Our pipeline, illustrated in Fig. 1, starts from a small set of omnidirectional images in the equirectangular projection. As prerequisites we assume that (a) input images are aligned to the gravity vector; (b) multi-view registration is possible; (c) target objects are visible from at least two points-of-view; (d) the bases of objects are below the camera horizon.

Constraint (a) is easily obtained on all modern mobile devices that have an IMU on board. Otherwise, vertical alignment can be obtained by rotating the global up vector so that the vertical edges are aligned with the vertical direction in the images. We thus consider vertical alignment to be a separate problem to be solved prior to the application of our pipeline, and we work only with oriented images. Constraints (b) and (c) require that images have at least some overlap, to ensure multi-view registration and detection of 3D features. In practice, this is obtained by 2-4 images per room. The last constraint, (d), is met by the vast majority of indoor objects, which are lying on the floor or attached to wall at low heights (e.g., furniture, sinks). The only objects that do not meet this constraint are objects hanging from the ceiling (such as lamps) or at the top of the walls (such as, for instance, some air conditioners). Removing this constraint to reconstruct objects at the top of the walls would require an initial estimation of the ceiling level, which is not always obvious, e.g., in the presence of sloped ceilings. As such top objects are not typically necessary for many applications, for example where it is more important to determine room shape and walkable floor space, we have focused our reconstruction on objects lying on the floor or attached to walls at low height, as the vast majority of indoor objects.

Given just a set of images of an environment meeting the above constraints, we start by performing a multi-view registration to recover camera poses, multi-view 3D features and a bounding volume

of the entire scene. In parallel, for each image, we classify image pixels into foreground (clutter) and background (wall, ceiling, and floor layout) exploiting a state-of-the-art approach for single-panorama analysis [YJL*18]. As a result, we determine a *mask* for each panoramic image containing pixels from foreground. We then create an interconnection graph between poses, according to their mutual visibility and photoconsistency, in order to split the image set in different room groups (Sec. 4). We exploit this graph, together with poses, 3D features, and masks, to simplify object recovery (Sec. 5) and to support room boundary extraction (Sec. 6). Pose and size of the clutter objects (Sec. 5) are determined from the clutter segmentation and the room grouping using a *virtual plane sweeping* approach (Sec. 5.3), based on a specific parameterization (Sec. 5.1) and cost function (Sec. 5.2). Clutter models are then used to enhance segmentation of the input images into floor, ceiling, and floor superpixels and to guide the extraction of room boundaries and interconnections (Sec. 6). As a result, we recover a structured 3D model of the floorplan comprehensive of clutter objects.

4 Partitioning of the panorama set

The room grouping phase aims to partition the image set into rooms, without manual intervention or prior layout knowledge. This grouping guides all subsequent analysis and geometric reasoning operations.

Following this idea, our algorithm works in two steps. In the first step, we build a graph connecting each pair of panoramic images that shares 3D points computed in the MVS registration phase, and weight each arc connecting two images with the likelihood that they were taken in the same room. In the second step, we run a graph clustering algorithm based on *random walks* to obtain a connected component for each room.

4.1 Arc weighting approaches

It could be tempting to weight arcs based on the count of shared 3D features between panoramas. However, in many cases the indoor scenario may contain very few 3D features. Even when there are many of them, it easily happens that panoramas taken nearby a door share a high number of features, no matter on which side of the door they were taken from. Because of this, we exploit 3D features just to build the initial graph and not for arc weighting. We have experimentally found that initial graph construction is very robust to the number of shared features used as threshold for arc creation. All results presented here use a threshold of 10 shared features, which is low enough to avoid rejecting good candidates.

Arc weighting is, therefore, determined by comparing images using an image similarity metric. Standard full-image methods, however, are not directly applicable for this task. In particular, in this setting is very likely to identify as similar images taken in different rooms with the same type of furniture and the same type of wall, floor, and ceiling colors. Moreover, full-image similarity measures are also hampered by the strong occlusions and distortions of indoor panoramas, leading to possibly strong differences among images taken from nearby viewpoints.

The core idea of our method is, instead, that if we take two panorama images and manage to warp a reasonably large unoccluded

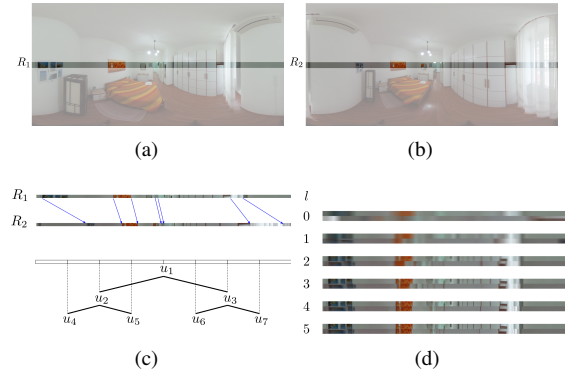


Figure 2: Fig. 2(a) and Fig. 2(b): two images of the same room with the horizon stripes R_1 and R_2 highlighted. Fig. 2(c): correspondences between the two stripes R_1 and R_2 and hierarchical scheme for top-down computation of optimal warping. Fig. 2(d): the first 5 levels of warping computation. Please note that the actual number of pixels for each level is $B * 2^l$. Stripes have been resized to the same length for the sake of comparison.

portion of one image onto a matching portion of the other, then it's likely that they are images of the same room. In our case, we can, in particular, leverage the fact that all our panoramic images are acquired approximately from the same height. Thus, any warping will map pixels between the horizon rows of the two images, that is, the central horizontal rows (or close to them in case of approximate equal elevation, see Sec. 4.3). In other words, we are considering just an horizontal slice of each panorama taken at eye level, that typically means above chairs, tables and most other clutter.

As it can be seen in the examples of Fig. 2(a) and Fig. 2(b), discontinuities on furniture, corners, doors and windows are captured along with their topological relationship (i.e., in one dimension, the order of the objects in the image corresponds to the topological order).

4.2 One-dimensional image warping

Let R_1 and R_2 be two rows of pixels. We obtain the warped version of R_1 , $W(R_1)$, by defining the function $W : \mathbb{R} \rightarrow \mathbb{R}$ as the piece-wise linear interpolation of a series of k values $W(\frac{i}{k+1}) = u_i, 0 \leq u_i \leq 1, i = 1 \dots k$. This warping is easily interpreted and implemented as the rendering of a texture mapped sequence of $k+1$ equally sized rectangles covering a row of pixels with the same width as R_1 and having $u_i, i = 1 \dots k$ as texture coordinates. We proceed by iterating an optimization algorithm in a top-down fashion for $k = 2^l, l = 0 \dots l = \log_2(\frac{n}{B})$, where n is the length of $R_{1,2}$ and B is the number of pixels at the minimal resolution. At each level l , the down-scaled version of $R_{1,2}$ are used, more precisely those with width equal to $B * 2^l$, and the error of a warping is computed as the average distance between the color of corresponding pixels, that is $E[|W(R_1) - R_2|]$. For $l = 0$ the warping is defined by a single variable/texture coordinate u_0 , for $l = 1$ by 3 variables u_0, u_1, u_2 , and so on. When the error minimization at level l is completed, the output values of the 2^l variables are passed to the next level $l+1$, and the remaining 2^{l+1} are initialized to random values.

4.3 Robust implementation

Although our approach is very straightforward, a robust implementation requires some more insights. First of all, any meaningful

warping should have a bijective W , which means that variable u_i should be increasing, i.e., $u_i < u_{i+1}$. This is easily achieved by minimizing over a set of variables t_i that defines the u_i in a hierarchical way. In other terms $u_1 = t_1$, $u_2 = 0 * (1 - t_2) + u_1 * t_1$, $u_3 = u_1 * (1 - t_2) + 1 * t_2$ etc. Second, we use one more variable Δ as an offset to all the others, that is $W_{\Delta}(\frac{i}{k+1}) = \Delta + u_i$. This is done in order to more easily represent the rotational component between the two panoramas. Finally, we do not actually use a one-pixel-thick row of pixels. In order to account for little height differences between the shooting point of the panoramas and for small inaccuracies in vertical registration, we use a thicker row. On the other hand, please note that only pixels at the horizon are mapped to pixels at the horizon, that is, a linear warping is inaccurate for pixels off the central row. We found a working compromise by using 4 pixels thick rows. Please note that, for the sake of illustration, the stripes $R_{1|2}$ in Fig. 2 are much thicker (160 pixels).

4.4 Graph partitioning

We associate to each arc (i, j) the weight $w(i, j) = 1 - \text{Err}(R_1, R_2)$ where Err is the error corresponding to the optimal warping for a given pair. Given the weighted graph, we exploit a method based on random walks [HK01] to compute a partition of the images in groups, one group per room. The idea of the random walk methods is to interpret $w(i, j)$ as the probability that a traveling agent in i will move to node j . In this setting, letting agents walk in the graph will make the natural clusters emerge as the arcs internal to a cluster are traversed more often than the arcs connecting nodes of different clusters.

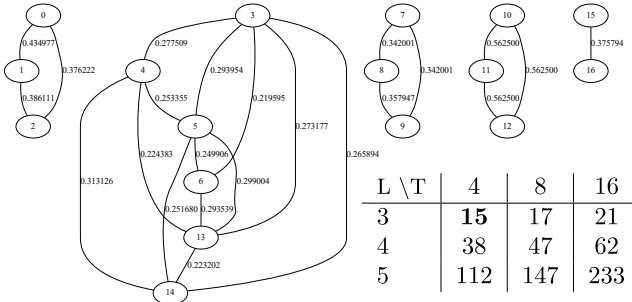


Figure 3: Panorama partitioning for dataset R2. The dataset is in Fig. 7, second column. The cells in the table report the overall running times (in seconds) to complete the partitioning at several combination of stripe thickness T in 4, 8, 16 (columns) and optimization level L in 3, 4, 5 (rows). The partitioning of the graph computed by the algorithm returns 5 rooms, independently of the parameters.

4.5 Parameter tuning and results

In order to verify that our method efficiently works without any manual intervention or per-dataset parameter tuning, we ran a series of tests on all datasets. In the table included in Fig. 3, the cells report the overall running times (in seconds) to complete the partitioning at several combination of stripe thickness T in 4, 8, 16 (columns) and the optimization level L in 3, 4, 5 (rows). The figure also shows the corresponding partitioned graph obtained by our algorithm, which is the same for all cases. This shows that changing the parameters has only effect on the running time but not on the final outcome, and that we can safely use the fastest configuration $T = 4, L = 3$ (top left cell of the table).

5 Recovery and modeling of clutter

We model each clutter object $O(z^*)$ as a cuboid whose 2D footprint is an oriented rectangle $F(z^*)$. The bottom face of the cuboid lies on the floor plane (i.e., $z = z_{min}$) and the top face on the plane with $z = z^*$ (Fig. 4(a)). Assuming the floor plane is known, this cuboid is fully defined by 6 parameters, which are, respectively, 2D position, 2D size, orientation around z axis of $F(z^*)$, and height z^* .

It should be noted that the object masks in individual images (e.g., Fig. 4(b)) do not necessarily describe a complete object shape but, as a result of automatic segmentation, only salient parts of it. Object identification is, therefore, done per room through a geometric reasoning process that takes as input both per-image information and global information.

At the image level, we segment the panorama into layout (background) and objects (foreground) with the method of Yang et al. [YJL*18], which fuses the results of saliency and object detection algorithms to recover candidate object positions also when objects have unusual shapes or are partially visible. As a result, each panorama image is enriched with the pixel mask of candidate foreground objects.

At the image group level, we exploit the output of our graph partitioning phase to apply all the reasoning phases only to images that are very likely taken in the same room, and therefore seeing (a subset of) the same objects. Multi-view registration is used to know the relative pose among images, as well as the position of a set of triangulated 3D features, which are used to refine object size and position as detailed below.

Object parameters are obtained by solving an optimization problem using a *virtual plane sweeping* approach. We set the *virtual camera* (i.e., scene center) at the position of one the input cameras (i.e., the first one), looking the floorplan from the ceiling to the floor along z direction, also setting this pose as the center of the floorplan model (Fig. 4(a)). In this reference frame, we define a projective function (Sec. 5.1), to parameterize on z each image contribution, and a cost function $E(z)$ (Sec. 5.2), to jointly evaluate the contribution of multiple images (Sec. 5.3).

5.1 Parameterization

Let $P(x, y, z)$ be a point in object space and $R_k T_k$ the reference frame associated to the equirectangular image k (Fig. 4(b)). The correspondence between points and image coordinates (u, v) is established as follows:

$$u = \frac{\arctan(\frac{p'_x y}{p'_x z})}{2\pi} * w$$

$$v = \frac{\arctan(\frac{\sqrt{p'^2_x + p'^2_y}}{p'_z})}{\pi} * h + \frac{h}{2}$$
(1)

where $P' = [R_k T_k] P$ are the local coordinates of the 3D point with respect to the k reference frame, w and h are respectively width and height of the image. With this relationship we can map the pixels of each mask onto the XY plane $Z = z$ (Fig. 4(c)) for any given z . By matching these projections among several images, we aim to identify objects and determine their cuboid representation. Using this approach, object recovery can be cast as an optimization process

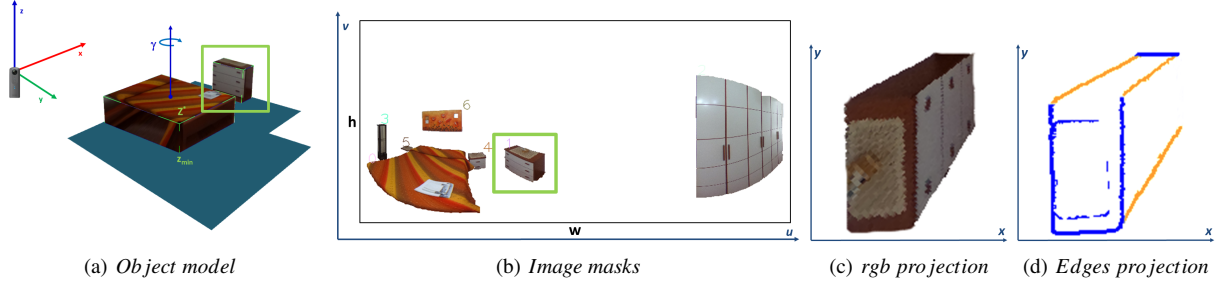


Figure 4: Foreground segmentation and projection. We model each clutter object as a cuboid lying on the floor plan 4(a). Given objects contours/masks of an equirectangular image 4(b), we generate, for each z , a representative projection of each mask/object (i.e., highlighted in green), both for rgb values 4(c) and edges 4(d).

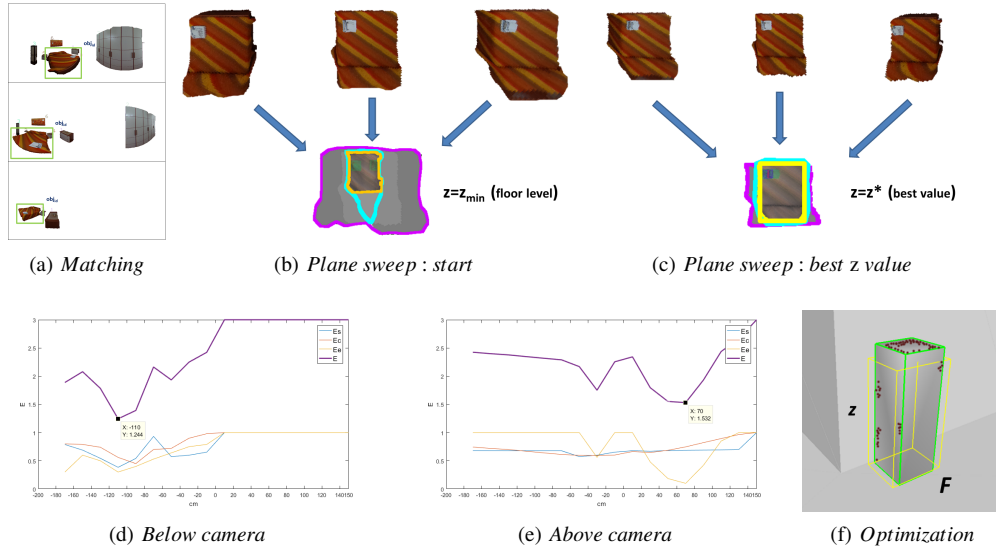


Figure 5: Object modeling steps. Fig. 5(a) shows the masks associated to object obj_{id} . Fig 5(b) shows the masks projection on the floor plane. Violet contour represents the union of the projections, Cyan contour their intersection, orange contour encloses points with best color consistency. Fig. 5(c) shows the projections of the masks when varying z , and in particular for the value z^* that minimizes the cost function. Yellow rectangle represents the recovered shape $F(z^*)$. Fig. 5(d) shows the cost function trend during plane sweeping for objects lower than the height of the virtual camera (e.g., same object-Bed of previous illustrations). Fig. 5(e) shows components trend for objects higher than the virtual camera (i.e., Cabinet from dataset R2). Fig. 5(f) illustrates optimization step, where yellow boundary represents the initial $O(z^*)$ approximation and the green boundary the final fitting on the 3D sparse points.

of a matching cost function (Sec. 5.2), which is minimized through an efficient plane sweeping approach (Sec. 5.3).

5.2 Cost function

We assume that at least $1 < m \leq n$ images contain a contour/mask of the same targeted object $O(F^*, z^*)$, where n is the number of images inside a group/room (Sec. 4). We want to define a cost function that is minimum when z is the actual height of the object's cuboid. The cost function $E(z)$ is:

$$E(z) = E_s(z) + E_c(z) + E_e(z) \quad (2)$$

where E_s, E_c, E_e denote different cost components. E_s evaluates how much the projected shapes of the same object coincide and fall on the same portion of space, E_c evaluates the consistency of the color, while E_e the consistency of the estimated shape edges with the color gradient.

$E_s = 1 - IoU$ (shape component) measures the similarity among masks, and is a value that depends on the intersection over union

ratio of the m projected masks (Fig. 5, cyan contour (intersection) violet contour (union)).

E_c , measures color consistency, on the assumption that the same object viewed from different positions has the same color. To cope with lighting and shading variations, we compute this measure from the hue, computing standard deviation σ_i of k points lying inside the intersection, as $E_c = \frac{\sum_{i=0}^k \sigma_i}{k \sigma_{max}}$, normalized on a maximum std value (i.e., $\sigma_{max} = 4$).

E_e , the edges component, is meant to measure, instead, the consistency in shape, under the assumption that significant edges are consistent among views. In order to compute this component, we first determine the minimum area rectangle enclosing intersection points with $\sigma_{hue} \leq \sigma_{max}$ (enclosing orange contour of Fig. 5(b) and Fig. 5(c)), which is assumed to be the current 2D footprint $F(z_c)$ for the given z_c , so that the resulting 3D cuboid is $O(z_c)$. From the current 2D footprint $F(z_c)$ of $O(z_c)$, we compute E_e (edges component), as the 2D mean squared distance of the projected edges (Fig. 4(d)) from the footprint $F(z_c)$, normalized to a max range (i.e.,

50cm). In the case of edges, note that we remove vertical lines of the image (i.e., vertical structures in the world space), because, on the XY projection, they would only provide noisy information in terms of radial lines not consistent with the shape of the object (Fig. 4(d), orange edges).

We exploit such cost function $E(z)$ in the optimization described in Sec. 5.3.

5.3 Object recovery through multi-view optimization

The first step of multi-view object recovery consists in determining the set of objects contained in a room by fusing the information coming from single-view analysis. To do that we partition the set of masks in disjoint groups, each group being associated to a different object (Fig. 5(a)). Each group contains masks from different images that have an intersection on the floor plane $z = z_{min}$. In case of ambiguities (i.e., a mask that has overlaps with multiple groups), we chose the solution with the minimum value of $E(z_{min})$ (Fig. 5(b)). At the end of this initialization, we have, thus, a list of objects obj_{id} , each associated to a list of m image masks related to that object.

Once the matching is established we perform, for each object, a plane sweeping along z , searching for the $z^* \in [z_{min} \dots z_{max}]$ which minimize $E(z)$, where $z = 0$ is the height of the virtual camera. Fig. 5(b) shows the projection of 3 masks for the initial z value (e.g., $z = z_{min}$), while Fig. 5(c) shows the projection in the proximity of the minimum of $E(z)$ (i.e., *Bed* from dataset R2, see results at Sec. 7). Figures 5(d) and 5(e) illustrate the trend of E components during plane sweeping. Fig. 5(d) shows the typical trend for objects below the virtual camera, i.e. objects whose upper part is visible from the camera. In this case the main contribution is given almost exclusively by *shape* and *color* components. Since the presence of objects under the camera can be identified a priori by the position of their masks in the images (e.g., all masks below the horizon), plane sweep is stopped in advance for $z = 0$. Fig. 5(e) shows the components trend for objects higher than the model center (i.e., *Cabinet* from dataset R2). Unlike the previous case, the components of shape and color are not very influential for the estimation of the height of the object, as the upper surface of it is not visible. Instead, the identification of the edges becomes discriminating, as evidenced by the trend in the graph (Fig. 5(e)).

As a result of plane sweeping, we obtain an axis-aligned approximation of the object cuboid $O(z^*)$. Starting from this approximation, we find all the 6 parameters (\bar{F}, z) by solving the following optimization problem with Levenberg-Marquardt iterations:

$$O^*(\bar{F}, z) \equiv \arg \min_{\bar{F}, z} [E_c(\bar{F}, z) + E_e(\bar{F}, z) + E_f(\bar{F}, z)] \quad (3)$$

Differently from the cost function in eq.2, in this case we do not compute E_s since the footprint is already defined by \bar{F} . Moreover, E_c and E_e are computed directly on the candidate footprint \bar{F} . Additionally, we introduce the E_f component, which is the mean squared distance from the cuboid faces of the 3D features close to the cuboid center (Fig. 5(f), green shape).

The above process is iterated for each object and for each room, until we populate the scene with all the 3D clutter models. We then exploit 3D clutter data to complete and enhance the reconstruction of the whole floor plan with room walls and ceilings (Sec. 6).

6 Recovery and modeling of the structural 3D floor plan

To recover walls, ceiling, and floor, we extend state-of-the-art approaches for large and complex indoor scenes by exploiting the results from graph partitioning and clutter modeling.

As discussed in Sec. 2, current methods generally exploit a simplified image segmentation into ceiling, floor and wall super-pixels (e.g., [CF14, PGP*18]), where, basically, the room shape is defined in 2D by the ceiling/floor super-pixels footprint, using *wall* super-pixels as 2D anchor points. This simplified classification, in absence of other information, is prone to substantial errors, both in terms of labeling accuracy (Fig. 6(a)), and, above all, in the positioning of the anchor points (Fig. 6(b)).

In our work, instead, we exploit image grouping (Sec. 4) and recovered clutter models (Sec. 5) to enhance images segmentation and labeling. The goal is to use the extra information to more accurately estimate anchor points and to provide additional constraints for the optimization and recovery of room geometry. In this work, this is done by extending the approach of Pintore et al. [PGP*18], which works with sparse panoramic imagery.

Our novel boundaries extraction pipeline is summarized in the pseudo-code 1, highlighting (in **bold**) our specific contributions with respect to the original approach [PGP*18]. Starting from the

Algorithm 1 Room boundary extraction

```

1:  $I_{eq}$  equirectangular registered images
2:  $F_{3D}$  multi-view features
3:  $C_{3D}$  clutter models
4: for all  $img \in I_{eq}$  do
5:    $imgSP \leftarrow createSP(img)$ 
6:    $maskedSP \leftarrow labelClutterSP(imgSP, C_{3D})$ 
7:    $labeledSP \leftarrow clutterDepth(maskedSP, C_{3D})$ 
8:    $labelBackgroundSP(labeledSP)$ 
9:    $labeledSP \leftarrow inFillBackgroundSP(labeledSP, F_{3D})$ 
10:  $S_{3D}$  3D room shapes (empty, to be computed)
11:  $Rooms \leftarrow groupImages(I_{eq})$ 
12: for all  $r \in Rooms$  do
13:    $F_{floor}, F_{ceiling}$  floor and ceiling 3D facets
14:    $A_{wall}$  wall 2D anchor points
15:    $A_{clutter}$  clutter 2D anchor points
16:    $F_{floor}, F_{ceiling} \leftarrow transformToFacets(labeledSP)$ 
17:    $A_{wall}, A_{clutter} \leftarrow transformToPoints(labeledSP)$ 
18:   filterWallPoints( $A_{wall}, Rooms$ )
19:    $shape2D \leftarrow computeShape(F_{floor}, F_{ceiling})$ 
20:   refineShape( $shape2D, A_{wall}, A_{clutter}$ )
21:    $shape3D \leftarrow make3D(F_{floor}, F_{ceiling})$ 
22:    $S_{3D} \leftarrow shape3D$ 
23: buildFloorplan( $S_{3D}, C_{3D}$ )

```

super-pixels segmentation of the original images (*createSP*), we project-back the recovered 3D clutter models on the segmented images (Fig. 6(a)), labeling as *clutter* super-pixels where the projection fall (*labelClutterSP*, and assigning to them the depth of the projected model (*clutterDepth*). Only for the remaining super-pixels, we perform background labeling (e.g., *ceiling*, *floor*, *wall*) and 3D features infilling, as described by [PGP*18], (*labelBackgroundSP* and *inFillBackgroundSP*).

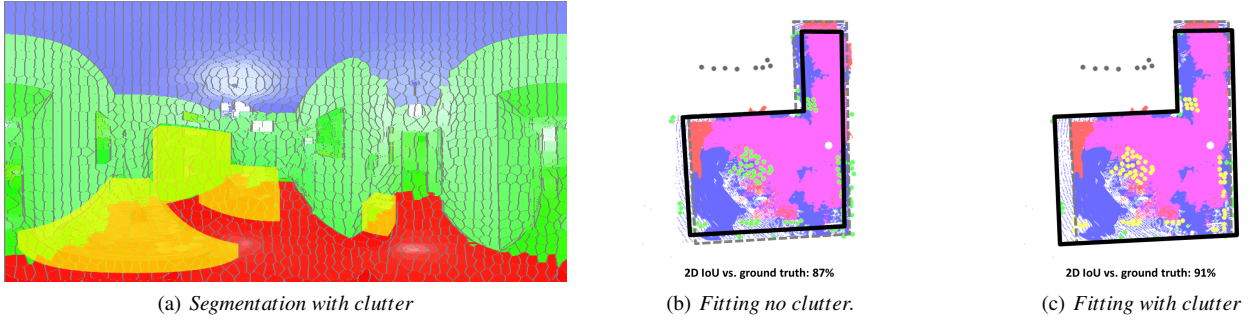


Figure 6: Floorplan modeling. Fig. 6(a) shows an alpha comparison between background labeling without considering clutter 3D models (blue (ceiling), red (floor) and green (wall)) and the labeling with projected clutter (yellow). Under the yellow labelled part is visible a glimpse of the segmentation without clutter [CF14]. Fig. 6(b) shows the room shape fitting without considering clutter. The initial shape estimated on ceiling-floor facets (dashed line) is shaped indistinctly on both anchor points of the wall and of the clutter (black contour). Grey dots represent points seen through a door but actually belonging to another room. Such points are filtered through the grouping information. Fig. 6(c) shows the fitting considering clutter information, which results in a more accurate reconstruction. We also show a quantitative comparison of the 2D intersection-over-union between the recovered shape and the ground truth (fitting w/ clutter vs fitting w/o clutter).

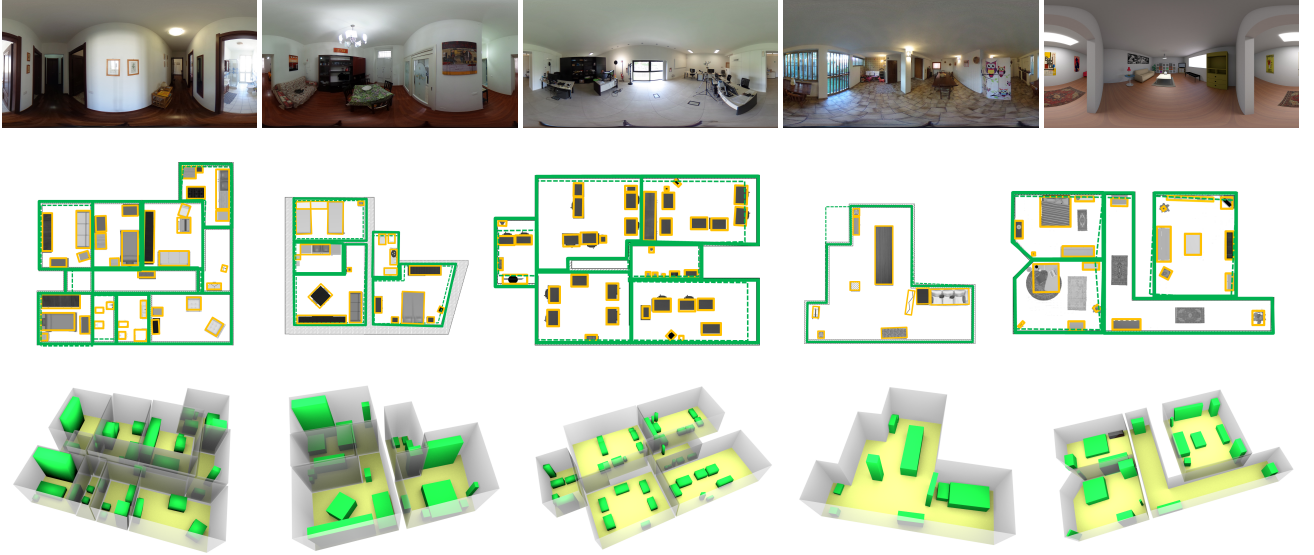


Figure 7: Recovered models vs. ground truth. We illustrate, for each dataset, the recovered floorplan models compared vs. ground truth and other approach [PGP*18]. The footprint of recovered objects is illustrated with orange rectangles over the 2D ground truth (e.g., gray background), rooms boundaries reconstructed by our method are showed with green line and rooms boundaries recovered by the other method by dotted green lines. Below the comparison we show 3D views of the recovered models.

Using the panorama-set partitioning algorithm introduced in Sec. 4, we assign each cluster of images to a different room. Then, for each room, we transform labeled super-pixels to 3D facets (ceiling and floor) and 2D anchor points (wall and clutter) (Fig. 6(c)). See Pintore et al. [PGP*18] for details). Using this projection, we estimate for each room, according to the original pipeline, a first 2D shape (*shape2D*) from ceiling and floor facets (*estimateShape*). Since the clutter has been explicitly removed prior to the application of the pipeline, the estimation is much improved in this phase with respect to Pintore et al. [PGP*18], as demonstrated in Sec. 7. We further improve over the original approach (*refineShape*), by pruning wall anchor points using the visibility graph information 4 and by integrating the *clutter* anchor points (Fig. 6(c)) in shape optimization.

Specifically, we filter-out wall anchor points A_{wall} (*filterWall-*

Points) that are seen at the same time by cameras of different visibility groups/rooms (e.g., a typical example of this situation are the points seen through an open door). It should be noted that in our method wall points no longer contain parts of the clutter. Then, we optimize the 2D polygon (*refineShape*) representing the room footprint (*shape2D*) to not only minimize its distance from wall points A_{wall} , but also imposing that *all* the anchor points $A_{clutter}$ are contained inside the polygon:

$$R_{2k} \equiv \begin{cases} \underset{\bar{R}}{\operatorname{argmin}} [\operatorname{dist}(A_{wall}, \operatorname{shape2D})] \\ \operatorname{shape2D} \ni A_{clutter} \end{cases} \quad (4)$$

Each *shape2D* hypothesis is generated by varying a vector of $2k$ corners $\bar{R}(x_0, y_0, \dots, x_k, y_k)$ [PGP*18], and imposing for each step that $A_{clutter}$ points must be contained inside the candidate shape.

Once the 2D walls arrangement is optimized, we extrude the 3D room using floor and ceiling 3D information (*make3D*).

Finally, we join in the same 3D representation the 3D clutter C_{3D} and all the 3D boundaries S_{3D} (*buildFloorplan*). The partitioning data is then used to complete the model with the passages between rooms (i.e., weak arcs on the camera trajectory computed at Sec. 4), determining whether they are doors or open passages.

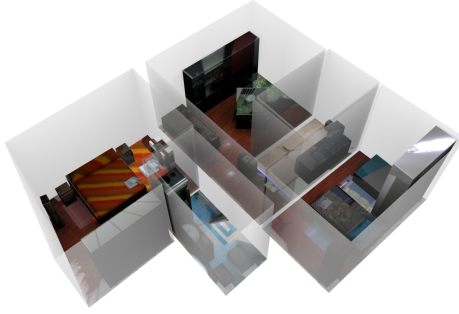


Figure 8: Model rendering example. 3D rendering of reconstructed R2 dataset with colors from textures.

7 Results

Our reconstruction pipeline is implemented in C++ on top of *OpenCV*. For multiview alignment we use *Photoscan* (<http://www.agisoft.com>). The system, starting from a collection of spherical images and their multi-view alignment, automatically produces a structured 3D floor plan in terms of interconnected rooms bounded by floor, walls and ceilings, and including the bounding volumes of clutter objects. Fig. 8 shows a recovered textured model of dataset R2.

7.1 Real and synthetic datasets

We tested our system on a variety of large and complex both real-world and synthetic multi-room scenes, containing over 130 clutter objects of different types (bed, cabinet, desk, chair, plant, lamp, lavatory, etc.).

Real-world datasets are employed to show the performance of the system on practical test cases. To simplify comparisons, we exploited publicly available multi-view data (<http://vic.crs4.it/download/datasets/>), which contains the measures of rooms and the height of the first camera (170 cm from the floor for almost all datasets), thus allowing the scaling of the models to metric real-world size (i.e., *cm*). Datasets have already a white compensation, provided by the capture hardware, and our pipeline was applied without any further pre-processing. Ground truth objects from real data have been manually approximated, from real laser measures, through representative CAD models having the same size, orientation and position of real clutter, so that they have the same bounding volume of real objects.

In addition, we exploited synthetic datasets to evaluate the system with respect to precise ground-truth data. Specifically, we modeled synthetic scenes by rendering photorealistic equirectangular images of 3D models from another public repository of large indoor scenes (<https://www.ifi.uzh.ch/en/vmml/>

research/datasets.html). We have enriched those models with additional clutter and photorealistic details.

7.2 Reconstruction performance

We ran our tests on a PC with Intel Core i7-4770 (3.40GHz) processor and 32GB RAM. On this PC, foreground segmentation [YJL*18] takes about 4 *seconds* per image, while multi-view registration takes about 2 *minutes* for a dataset of 24 images using *Photoscan*. Object partitioning takes about 10 *seconds* for the same dataset of 24 images. Object inference takes about 8 *seconds* per object, while boundaries estimation 12 *seconds* per room. The average overall time to reconstruct a furnished room is about 1 *minute*.

For comparison, single panorama approaches [XSKT17, YJL*18] take about 8 *minutes* to reconstruct a cluttered room. The multi-view approach of Bao et al. [BFFFS14] also takes about 1 *minute* but to reconstruct only a small portion of the room.

7.3 Quantitative evaluation

In Tab. 1, we present the quantitative performance for the boundary extraction phase. We compare results, in terms of 3D layout recovered, with respect to ground truth and the methods of Pintore et al. [PGP*18] and Yang et al. [YZ16].

We also show, for each floor plan, the amount of reconstructed objects over the amount of target objects, assuming as target objects the segmented clutter that is visible from at least two images. The unreconstructed objects are therefore *failure cases* of our method, independent from the segmentation performance, which is external to our methods [YJL*18]. The experiments show that our method can fail if object masks do not contain enough geometric cues, as showed, for example, in Fig. 10.

Clutter error shows the average and maximum error on recovered objects with respect to ground truth. We measure positional error (*2D Pos. cm*) as distance between object centroids on the 2D ground plane, orientation error (*Orient. deg*) as the angle between ground truth cuboid and its estimated pose, area error (*Area percentage %*), with respect to ground truth object footprint, and the object height error (*Height cm*). We intentionally separated the various components (position, area, height), instead of using, for example, intersection-over-union, to facilitate comparison with other clutter modeling methods (Tab. 3).

Our clutter modeling method produces a very low average position, orientation and area error in all tests, where the major deviations, especially in terms of dimensions, are due mostly to specific cases of not well defined shapes (e.g., plants, pillows on a bed, TV on a cabinet, etc.). Further details and comparisons on clutter recovery performance are exposed in Tab. 3, Fig. 7 and Fig. 9.

Imgs per room and *Imgs assignment* show a numerical comparison of our clustering method with the geometric occlusion approach of Pintore et al. [PGP*18], which is the method most similar to ours, in terms of input data and constraints imposed on models (see Sec. 2). We show how our visual approach allows the entire pipeline to work with a smaller number of images per room, and how, instead, [PGP*18] needs some more images because it discards the images that it is not able to geometrically assign. Moreover,

Scene			Clutter error				Imgs per room		Imgs assignment		Room 3D IoU		
Name	Objects	m^2	2D Pos.	Orient.	Area	Height[cm]	Our	[PGP*18]	Our	[PGP*18]	Our	[PGP*18]	[YZ16]
Real-data R1	35/36	96	2 ± 5 cm	0.2 ± 1.8 deg	2 ± 26 %	2 ± 15 cm	2.5	2.6	97 %	76 %	89 %	83 %	75 %
Real-data R2	19/20	78	3 ± 21 cm	0.7 ± 2.3 deg	2 ± 18 %	1 ± 8 cm	3	4	100 %	99 %	90 %	82 %	74 %
Real-data R4	43/44	196	2 ± 8 cm	0.4 ± 2.1 deg	3 ± 1 %	3 ± 2 cm	3	6	91 %	72 %	88 %	74 %	70 %
Real-data R5	10/10	55	4 ± 11 cm	0.5 ± 1.0 deg	2 ± 8 %	2 ± 3 cm	2.5	5	100 %	70 %	91 %	84 %	61 %
Synthetic data S1	20/21	188	4 ± 16 cm	0.1 ± 1.0 deg	3 ± 32 %	1 ± 5 cm	2.5	4	100 %	86 %	90 %	72 %	49 %

Table 1: Floorplan performance. We present a summary of performance on large, representative floorplans, detailing clutter and rooms structure reconstruction errors. For each dataset we show the ratio of objects recovered, the scene area, the average and maximum error on recovered objects with respect to ground truth, the average number of images needed per room, the percentage of correct image assignments and the resulting 3D intersection-over-union ratio (average of all rooms) with respect to ground truth, compared to the pipeline of Pintore et al. [PGP*18]. For completeness, we also include the average results of Yang et al. [YZ16] for rooms where such single-view approach works.

	Coverage	Bedroom1	Living	Bedroom2	Restroom	Kitchen
Number of poses	Min	3	5	3	2	2
	Med	5	9	5	3	3
	Max	7	13	7	4	4
3D IoU no clutter info	Min	85%	87%	80%	88%	87%
	Med	87%	90%	84%	90%	88%
	Max	89%	91%	86%	92%	89%
3D IoU with clutter info	Min	90%	91%	89%	91%	89%
	Med	90%	92%	89%	92%	90%
	Max	91%	92%	90%	93%	90%

Table 2: Performance vs. sparsity. We show the 3D layout recovery performance using the smallest subset of images necessary for our approach to work (i.e., Min coverage), a denser subset (i.e., Med coverage) and using all the captured images (i.e., Max coverage). Results demonstrate how incorporating clutter information makes it possible to obtain significantly better results with a reduced number of images.

our method achieves a better accuracy in the assignment of images (i.e., average compared to all rooms), in particular in the case of semi-open spaces without doors (e.g., *Real-data R4* and *R5*).

Room 3D IoU shows instead the 3D intersection-over-union ratio (average of all room) with ground truth, compared to the pipeline of Pintore et al. [PGP*18], and, for completeness, with the pipeline of Yang et al. [YZ16] (<https://github.com/YANG-H/Panoramix>). Yang et al. approach [YZ16], although single-view and limited to simple rooms visible from a single point of view, still remains a good benchmark, since even more recent single-view geometric approaches [YJL*18] are based on the same methodology of segmentation and background reconstruction.

In all cases, our approach outperforms the other methods, mostly due to the exploitation of image grouping and of clutter information in the recovery of room structure and shape. For the sake of clarity, recent data-driven single-view methods for room extraction [YWP*19], even if not directly comparable, achieve average 3D IoU performance of about 77% (imposing *Manhattan World* constraint). Fig. 7 illustrates floorplan results for presented cases, showing the footprint of recovered objects (orange rectangles) over 2D floorplan (gray background), rooms boundaries reconstructed by our method (green line) and rooms boundaries recovered by the other method [PGP*18] (dotted green).

It should be noted that our system is designed to work with a very small set of images. Since most of the datasets used in our experiments provided a denser coverage than that necessary for our method, we tested different image subsets having different sparsity, in order to analyze the behavior of the method with different sets of images. We show in Tab. 2 a performance comparison for dataset *R2*

(e.g., performances are similar for the other datasets), in term of 3D layout recovery, using the smallest subset of images (i.e., *Min* coverage), a denser subset (i.e., *Med* coverage) and using all the available images (i.e., *Max* coverage). The smaller subset corresponds to the minimal set of images where the assumptions listed in Sec. 3 are verified.

With our experiments, we found that the number of images does not affect object recovery, and has not a significant effect on layout estimation when exploiting clutter information. Performance drops, instead, when clutter information is not used for refining room shape.

	Our		[XSKT17]	
	Pos. Err.	Orient. Err.	Pos. Err.	Orient. Err.
Bed	2 ± 4 cm	0.0 ± 1.5 deg	25 ± 17 cm	1.0 ± 1.4 deg
Chair	1 ± 2 cm	0.5 ± 1.5 deg	52 ± 66 cm	10.7 ± 15 deg
Plant	2 ± 6 cm	—	9 ± 12 cm	—
Overall	3 ± 21 cm	1.0 ± 3.0 deg	28 ± 32 cm	4.3 ± 5.7 deg

Table 3: Object reconstruction comparison. We summarize clutter results for all datasets, detailing performances for some categories, also exposed by the method of Xu et al. [XSKT17]. Our method presents better results in all categories, also taking into account that our results are calculated on a wider variety of rooms and in our approach we also estimate orientation error for plants, which are hard to fully recover due to foliage (i.e., Fig. 9).

Tab. 3 summarizes clutter recovery results for all datasets, detailing performance for several main categories, comparing them with the performance of Xu et al. [XSKT17]. Xu et al. [XSKT17] method for single indoor panoramic images, extends Zhang et al. approach [ZSTX14] to model the clutter and to recover room shapes which are not necessarily cubic, exposing numerical results comparable with ours. Our method exhibits better performance in all categories, also taking into account that our results are calculated on a wider variety of rooms. The increase in performance is due to the fact that we use more than one view of the same object. In addition, with our approach we can also estimate orientation error for plants (i.e., Fig. 9).

Tab. 4 summarized our performance in terms of the percentage of image pixels whose 3D information can be estimated (*Completeness*), and in terms of percentage of correctly labeled pixels (*Accuracy*). These values are calculated, by means of synthetic datasets ground truth, using the labeling of individual pixels and not just of representative cuboids (Fig. 9(a)). We compare our results with two different multi-view approaches, Bao et al. [BFFFS14] and Cabral and Furukawa [CF14]. As discussed in Sec. 2, Bao et al. [BFFFS14] reconstruct small cluttered indoor scenes (e.g, a room portion), from a set of about 10 pin-hole images. Code and data are not available, but the original publication provides numerical results. It should be

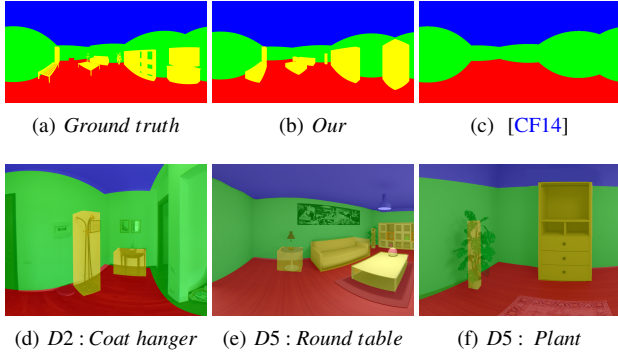


Figure 9: Re-projection of recovered models and image labeling. On the first row, we present a comparison, on the same example image of the S1 dataset, between ground truth 9(a), our method 9(b) and the semi-dense approach of Cabral et al. 9(c). On the second row, we show some examples of objects reconstructed by our method. While the hanger (Fig. 9(d)) and the rounded table (Fig. 9(e)) are correctly represented by their boundaries, approximation of the plant leads to a 2D size error on foliage (Fig. 9(f)), although the real footprint size (i.e., plant pot) is correct.

	Our	[BFFFS14]	[CF14]
Completeness	100%	91%	100%
Accuracy	91%	80%	68%

Table 4: Completeness and accuracy comparison. Comparison with other multi-view approaches [BFFFS14, CF14] in terms of the percentage of image pixels whose 3D information can be estimated (Completeness), and in terms of percentage of correctly labeled pixels (Accuracy).

noted that such method [BFFFS14] recovers only 86% of the objects inside the room, since they cannot recover surfaces that do not contain SfM points. Cabral et al. [CF14], instead, adopted equirectangular images to improve the indoor reconstruction provided by a dense multi-view pipeline [FCSS09]. Although their full approach is not directly applicable to our sparse data, their super-pixels structure classification can be used as a benchmark, as it provides a complete classification of each individual equirectangular image (Fig. 9(c)). As source code and data were not available, we reimplemented the approach by following the paper description.

Results show that our method guarantees a reconstruction of the scene even in the areas without 3D points, and with an accuracy higher than these other approaches, even if we are using a much more limited number of images (Fig. 9(b)).

Fig. 10 shows examples of failure cases. As described in the previous sections, our method needs at least two segmented views of the same object to reconstruct it, which must somehow contain consistent shape and color cues.

Fig. 10(a) shows two image masks coming from different objects. Since, geometrically, they have a common surface that projects into the same portion of 2D plane (see Sec. 5.3), these masks are wrongly matched. Fig. 10(b) shows, instead, a case of occlusion affecting the object footprint. In this case, objects have been recognized and matched, but the first view (left image) does not contain enough information to estimate the object footprint. We have experienced that these cases depend mainly on the position of the camera and not on the scene type. In particular, images taken too far away from the object or from a not optimal pose are more subject to errors, mainly during the masks matching step (Sec. 5).

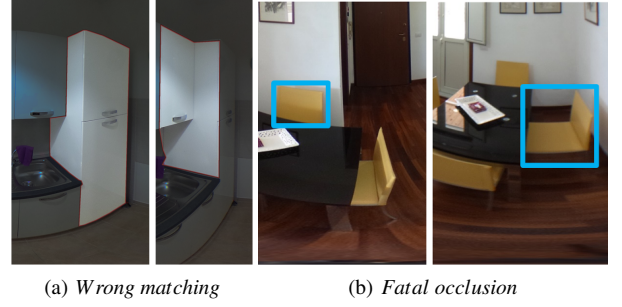


Figure 10: Example of failure cases. In Fig. 10(a) two masks have been merged even if they come from different objects. Fig. 10(b) shows instead two masks correctly recognized but with a fatal occlusion in the left one.

8 Conclusion

We have presented a light-weight approach to capture and automatically reconstruct structured 3D models of cluttered multi-room indoors. Starting from a small set of overlapping panoramic images, our method automatically generates a 3D layout of the rooms and of all major objects inside. Such a 3D layout is directly suitable for use in a number of real-world applications, such as guidance, energy management, location, routing, or content creation for security.

Our main advantages are in providing a full pipeline that exploits an automatic partitioning into rooms and seamlessly merges clutter detection and room shape reconstruction. As demonstrated by our results, our approach, in addition to providing shape and location of 3D interior objects, also increases the precision in wall structure recovery with respect to competing methods working on sparse panoramic images.

In our future work, we plan to exploit the reconstructed semantic models for simulation and visualization in the security and energy management areas, especially in applications related to the monitoring of public and private buildings. We also plan to further enrich the models with semantic information attached to objects, also going towards automatic 3D modeling by replacing the cuboid approximation with fully 3D models using a data-driven approach.

Acknowledgments. This work has received funding from Sardinian Regional Authorities under projects VIGELAB, AMAC, and TDM (POR FESR 2014-2020 Action 1.2.2). We also acknowledge the contribution of the European Union's H2020 research and innovation programme under grant agreement No 820434 (ENCORE).

References

- [BFFFS14] BAO S. Y., FURLAN A., FEI-FEI L., SAVARESE S.: Understanding the 3D layout of a cluttered room from multiple images. In *Proc. IEEE WACV* (2014), pp. 690–697. 2, 3, 9, 10, 11
- [BTS*17] BERGER M., TAGLIASACCHI A., SEVERSKY L. M., ALLIEZ P., GUENNEBAUD G., LEVINE J. A., SHARF A., SILVA C. T.: A survey of surface reconstruction from point clouds. *Comput. Graph. Forum* 36, 1 (2017), 301–329. 1
- [CF14] CABRAL R., FURUKAWA Y.: Piecewise planar and compact floorplan reconstruction from images. In *Proc. CVPR* (2014), pp. 628–635. 2, 3, 7, 8, 10, 11
- [DBK*13] DEL PERO L., BOWDISH J., KERMGARD B., HARTLEY E., BARNARD K.: Understanding bayesian rooms using composite 3D object models. In *Proc. CVPR* (2013), pp. 153–160. 3

- [FCSS09] FURUKAWA Y., CURLESS B., SEITZ S. M., SZELISKI R.: Reconstructing building interiors from images. In *Proc. ICCV* (2009), pp. 80–87. [3](#), [11](#)
- [FMR11] FLINT A., MURRAY D., REID I.: Manhattan scene understanding using monocular, stereo, and 3D features. In *Proc. ICCV* (2011), pp. 2228–2235. [3](#)
- [GPMAL09] GOLPARVAR FARD M., PEA-MORA F., ARBOLEDA C., LEE S.: Visualization of construction progress monitoring with 4d simulation model overlaid on time-lapsed photographs. *Journal of Computing in Civil Engineering* 23, 6 (2009), 391–404. [2](#)
- [HDGN17] HUANG J., DAI A., GUIBAS L., NIESSNER M.: 3Dlite: Towards commodity 3D scanning for content creation. *ACM TOG* 36, 6 (2017), 203:1–203:14. [2](#)
- [HHF10] HEDAU V., HOIEM D., FORSYTH D.: Thinking inside the box: Using appearance models and context based on room geometry. In *Proc. ECCV* (2010), pp. 224–237. [3](#)
- [HHF12] HEDAU V., HOIEM D., FORSYTH D.: Recovering free space of indoor scenes from a single image. In *Proc. CVPR* (2012), pp. 2807–2814. [3](#)
- [HK01] HAREL D., KOREN Y.: On clustering using random walks. In *Proc. FST TCS* (2001), pp. 18–41. [5](#)
- [IYF15] IKEHATA S., YANG H., FURUKAWA Y.: Structured indoor modeling. In *Proc. ICCV* (2015), pp. 1323–1331. [2](#)
- [LGHK10] LEE D. C., GUPTA A., HEBERT M., KANADE T.: Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *Proc. NIPS* (2010), pp. 1288–1296. [3](#)
- [MMJV*14] MURA C., MATTAUSCH O., JASPE VILLANUEVA A., GOBBETTI E., PAJAROLA R.: Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts. *Computers & Graphics* 44 (2014), 20–32. [1](#)
- [MMP16] MURA C., MATTAUSCH O., PAJAROLA R.: Piecewise-planar reconstruction of multi-room interiors with arbitrary wall arrangements. *Computer Graphics Forum* 35, 7 (2016), 179–188. [2](#)
- [PGGS16] PINTORE G., GANOVELLI F., GOBBETTI E., SCOPIGNO R.: Mobile mapping and visualization of indoor structures to simplify scene understanding and location awareness. In *ECCV Workshops* (2016), pp. 130–145. [2](#)
- [PGP*18] PINTORE G., GANOVELLI F., PINTUS R., SCOPIGNO R., GOBBETTI E.: 3D floor plan recovery from overlapping spherical images. *Computational Visual Media* 4, 4 (2018), 367–383. [2](#), [3](#), [7](#), [8](#), [9](#), [10](#)
- [PPG*18] PINTORE G., PINTUS R., GANOVELLI F., SCOPIGNO R., GOBBETTI E.: Recovering 3D existing-conditions of indoor structures from spherical images. *Computers & Graphics* 77 (2018), 16–29. [3](#)
- [SCD*06] SEITZ S. M., CURLESS B., DIEBEL J., SCHARSTEIN D., SZELISKI R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proc. CVPR* (2006), vol. 1, pp. 519–528. [3](#)
- [SFP13] SCHWING A. G., FIDLER S., POLLEFEYS M., URTASUN R.: Box in the box: Joint 3D layout and object reasoning from single images. In *Proc. ICCV* (2013), pp. 353–360. [3](#)
- [SQLG15] SU H., QI C. R., LI Y., GUIBAS L. J.: Render for CNN: Viewpoint estimation in images using cnns trained with rendered 3D model views. In *Proc. ICCV* (2015), pp. 2686–2694. [3](#)
- [SRLH14] SATKIN S., RASHID M., LIN J., HEBERT M.: 3DNN: 3D nearest neighbor. data-driven geometric scene understanding using 3d models. *International Journal of Computer Vision* 111 (2014), 69–97. [3](#)
- [TCZ15] TURNER E., CHENG P., ZAKHOR A.: Fast, automated, scalable generation of textured 3d models of indoor environments. *IEEE Journal of Selected Topics in Signal Processing* 9, 3 (2015), 409–421. [1](#), [2](#)
- [TKCM16] TULSIANI S., KAR A., CARREIRA J., MALIK J.: Learning category-specific deformable 3d models for object reconstruction. *IEEE TPAMI* 39, 4 (2016), 719–731. [3](#)
- [TXLK11] TSAI G., XU C., LIU J., KUIPERS B.: Real-time indoor scene understanding using bayesian filtering with motion cues. In *Proc. ICCV* (2011), pp. 121–128. [3](#)
- [XAAH13] XIONG X., ADAN A., AKINCI B., HUBER D.: Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction* 31 (2013), 325–337. [2](#)
- [XF14] XIAO J., FURUKAWA Y.: Reconstructing the world’s museums. *International Journal of Computer Vision* 110, 3 (Dec 2014), 243–258. [2](#)
- [XMS14] XIANG Y., MOTTAGHI R., SAVARESE S.: Beyond PASCAL: A benchmark for 3D object detection in the wild. In *Proc. WACV* (2014), pp. 75–82. [3](#)
- [XSKT17] XU J., STENGER B., KEROLA T., TUNG T.: Pano2CAD: Room layout from a single panorama image. In *Proc. WACV* (2017), pp. 354–362. [3](#), [9](#), [10](#)
- [YJL*18] YANG Y., JIN S., LIU R., , YU J.: Automatic 3D indoor scene modeling from single panorama. In *Proc. CVPR* (2018), pp. 3926–3934. [2](#), [3](#), [4](#), [5](#), [9](#), [10](#)
- [YWP*19] YANG S.-T., WANG F.-E., PENG C.-H., WONKA P., SUN M., CHU H.-K.: DuLa-Net: A dual-projection network for estimating room layouts from a single RGB panorama. In *Proc. IEEE CVPR* (2019). [3](#), [10](#)
- [YZ16] YANG H., ZHANG H.: Efficient 3D room shape recovery from a single panorama. In *Proc. CVPR* (2016), pp. 5422–5430. [2](#), [3](#), [9](#), [10](#)
- [ZCC16] ZHANG E., COHEN M. F., CURLESS B.: Emptying, refurnishing, and relighting indoor spaces. *ACM TOG* 35, 6 (2016), 174:1–174:14. [2](#)
- [ZCSH18] ZOU C., COLBURN A., SHAN Q., HOIEM D.: LayoutNet: Reconstructing the 3d room layout from a single rgb image. In *Proc. CVPR* (2018), pp. 2051–2059. [3](#)
- [ZSTX14] ZHANG Y., SONG S., TAN P., XIAO J.: Panocontext: A whole-room 3d context model for panoramic scene understanding. In *Proc. ECCV* (2014), pp. 668–686. [2](#), [3](#), [10](#)