General 3D Room Layout from a Single View by Render-and-Compare

Sinisa Stekovic[1], Shreyas Hampali[1], Mahdi Rad[1], Sayan Deb Sarkar[1], Friedrich Fraundorfer[1], and Vincent Lepetit[1,2]

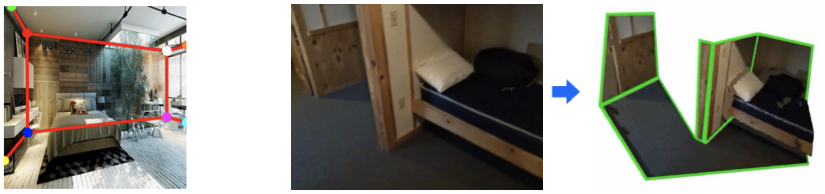
[1] Institute for Computer Graphics and Vision, Graz University of Technology, Graz, Austria
[2] Université Paris-Est, École des Ponts ParisTech, Paris, France
{sinisa.stekovic, hampali, rad, sayan.sarkar, fraundorfer, lepetit}@icg.tugraz.at
Project page: https://www.tugraz.at/index.php?id=40222

Abstract. We present a novel method to reconstruct the 3D layout of a room—walls, floors, ceilings—from a single perspective view in challenging conditions, by contrast with previous single-view methods restricted to cuboid-shaped layouts. This input view can consist of a color image only, but considering a depth map results in a more accurate reconstruction. Our approach is formalized as solving a constrained discrete optimization problem to find the set of 3D polygons that constitute the layout. In order to deal with occlusions between components of the layout, which is a problem ignored by previous works, we introduce an analysis-by-synthesis method to iteratively refine the 3D layout estimate. As no dataset was available to evaluate our method quantitatively, we created one together with several appropriate metrics. Our dataset consists of 293 images from ScanNet, which we annotated with precise 3D layouts. It offers three times more samples than the popular NYUv2 303 benchmark, and a much larger variety of layouts.

Keywords: Room Layout, 3D Geometry, Analysis-by-Synthesis

1 Introduction

The goal of layout estimation is to identify the layout components—floors, ceilings, walls— and their 3D geometry from one or multiple views, despite the presence of clutter such as furniture, as illustrated in Fig. 1. This is a fundamental problem in scene understanding from images, with potential applications in many domains, including robotics and augmented reality. When enough images from different views are available, it is possible to recover complex 3D layouts by first building a dense point cloud [1,19]. Single view scenarios are far more challenging even when depth information is available, since layout components may occlude each other, entirely or partially, and large chunks of the layout are then missing from the point cloud. Moreover, typical scenes contain furniture and the walls, the floors, and the ceilings might be obstructed. Important features such as corners or edges might be only partially observable or even not visible at all.



(a) A cuboid layout (b) This paper

Fig. 1. (a) Most current methods for single view layout estimation make the assumption that the view contains a single room with a cuboid shape. This makes the problem significantly simpler as the structure and number of corners remain fixed, but can only handle a fraction of indoor scenes. (b) By contrast, our method is able to estimate general 3D layouts from a single view, even in case of self-occlusions. Its input is either an RGBD image, or an RGB image from which a depth map is predicted.

As shown in Fig. 1(a), many recent methods for single view scenarios avoid these challenges by assuming that the room is a simple 3D cuboid [4, 9, 11, 16, 20, 27, 32] or that the image contains at most 3 walls, a floor, and a ceiling [37]. This is a very strong assumption, which is not valid for many rooms or scenes, such as the ones depicted in Fig. 1(b) and Fig. 4. In addition, most of these methods only provide the *2D projection* of the layout [9, 16, 27, 32], which is not sufficient for many applications. Other methods rely on panoramic images from viewpoints that do not create occlusions [29, 35, 36], which is not always feasible.

The very recent method by Howard-Jenkins *et al.* [10] is probably the only method to be able to recover general layouts from a single perspective view. However, it does not provide quantitative evaluation for this task, but only for cuboid layouts in the case of single views. In fact, it often does not estimate well the extents of the layout components, and how they are connected together.

In this paper we introduce a formalization of the problem and an algorithm to solve it. Our algorithm takes as input a single view which can be an RGBD image, or even only a color image: When a depth map is not directly available, it is robust enough to rely on a predicted one from the color image [18, 25]. As shown on the right of Fig 1(b), its output is a 3D model that is "structured", in the sense that the layout components connected in the scene are also connected in the 3D model in the same way, similarly to what a human designer would do. Moreover, we introduce a novel dataset to quantitatively evaluate our method.

More exactly, we formalize the problem of recovering a 3D polygonal model of the layout as a constrained discrete optimization problem. This optimization selects the polygons constituting the layout from a large set of potential polygons. To generate these polygons, like [10] and earlier work [22] for point clouds, we rely on 3D planes rather than edges and/or corners to keep the approach simple in terms of perception and model creation. However, as mentioned above, not all 3D planes required in the construction of the layout are visible in the image. Hence, we rely on an analysis-by-synthesis approach, sometimes referred to as 'render-and-compare'. Such approaches do not always require a realistic rendering, in

Dataset	Layout	Mode	Cam. Param.	Eval. Metrics	#TestSamples
Hedau *et al.* [8]	Cuboid	RGB	Varying	2D	105
LSUN [33]	Cuboid	RGB	Varying	2D	1000
NYUv2 303 [31]	Cuboid	RGBD	Constant	2D	100
ScanNet-Layout (ours)	General	RGBD	Constant	3D	293

Table 1. Comparison between test sets of different datasets for single-view layout estimation on real images. ScanNet-Layout does not provide a training set.

terms of texture or lighting, as in [15, 30] for example: We render a depth map for our current layout estimate, and compare it to the measured or predicted depth map. From the differences, we introduce some of the missing polygons to improve our layout estimate. We iterate this process until convergence.

Our approach therefore combines machine learning and geometric reasoning. Applying "pure" machine learning to these types of problems is an appealing direction but it is challenging to rely only on machine learning to obtain structured 3D models as we do. Under the assumption that the room is box-shaped [4, 11], this is possible because of the strong prior on feasible 2D layouts [9, 16]. In the case of general layouts, this is difficult, as the variability of the layouts are almost infinite (see Fig. 4 for examples). Moreover, only very limited annotated data is available for the general problem. Thus, we use machine learning only to extract image cues on the 3D layout from the perspective view, and geometric reasoning to adapt to general configurations based on these image cues.

To evaluate our method, we manually annotated 293 perspective views from the ScanNet test dataset [5] together with 5 novel 2D and 3D metrics, as there was no existing benchmark for the general problem. This is three times more images than NYUv2 303 [28, 31], a popular benchmark for evaluating cuboid layouts. Other single-view layout estimation benchmarks are Hedau *et al.* [8] and LSUN [33], that are cuboid datasets with only 2D annotations, and Structured3D [34], a dataset containing a large number of synthetic scenes generated under the Manhattan world assumption. Our ScanNet-Layout dataset is therefore more general, and is publicly available. Table 1 summarizes the difference between benchmarks. We also compare our method to cuboid-specific methods on NYUv2 303, which contains only cuboids rooms, to show that our method performs comparably to these specialized methods while being more general.

Main contributions. First, we introduce a formalization of the general layout estimation from single views into a constrained discrete optimization problem. Second, We propose an algorithm based on this formalization and are able to generate a simplistic 3D model for general layouts from single perspective views (RGB or RGBD). Finally, we provide a novel benchmark with a dataset and new metrics for the evaluation of methods for general layout estimations from single perspective views.

2 Related work

We divide here previous works on layout estimation into two categories. Methods from the first category start by identifying features such as room corners or edges from the image. Like our own approach, methods from the second category rely on 3D planes and their intersections to build the layout. We discuss them below.

2.1 Layout Generation from Image Features

Some approaches to layout estimation, mostly for single-view scenarios, attempt to identify features in the image such as room corners or edges, before connecting them into a 2D layout or lifting them in 3D to generate a 3D room layout. Extracting such features, and lifting them in 3D are, however, very challenging. A common assumption is the Manhattan constraint that enforces orthogonality and parallelism between the layout components of the scene, often done by estimating vanishing points [8, 20, 24, 27], a process that can be very sensitive to noise.

Another assumption used by most of the current methods is that only one box-shaped room is visible in the image [9,16,20,27,31,32]. This is a strong prior that achieves good results, but only if the assumption is correct. For example, in [20], 3D cuboid models are fitted to an edge map extracted from the image, starting from an initial hypothesis obtained from vanishing points. From this 3D cuboid assumption, RoomNet [16] defines a limited number of 11 possible 2D room layouts, and trains a CNN to detect the 2D corners of the box-shaped room. [32] relies on segmentation to identify these corners more robustly. These last approaches [16, 32] are limited to the recovery of a 2D cuboid layout.

Yet another approach is to directly predict the 3D layout from the image: [4, 11] not only predict the layout but also the objects and humans present in the image, using them as additional constraints. Such constraints are very interesting, however, this approach also requires the '3D cuboid assumption', as it predicts the camera pose with respect to a 3D cuboid.

[29,35,36] relax the cuboid assumption and can recover more general layouts. However, in addition to the Manhattan assumption, this line of work requires panoramic images captured so that they do not exhibit occlusions from walls. This requirement can be difficult to fulfill or even impossible for some scenes. By contrast, our method does not require the cuboid or the Manhattan assumptions, and handles occlusions in the input view to handle variety of general scenes.

2.2 Layout Generation from 3D Planes

An alternative to inferring room layouts from image features like room corners is to identify planes and infer the room layout from these plane structures. If complete point clouds are available, for example, from multiple RGB or RGBD images, identifying such planes is straightforward, and has been successfully applied for this task [2, 12, 21, 26]. Recently, successes in single RGB image based depth estimation [6, 17, 25] and 3D plane detection [18] opened up the possibility of layout generation from single RGB images. For example, Zou *et al.* [37] finds

the layout planes in dominant room directions and then reasons on the depth map to estimate the extents of the layout components. However, even though this method does not assume strict cuboid layouts, it assumes the presence of only five layout components—floor, ceiling, left wall, front wall and right wall.

Like our method, the work of Howard-Jenkins *et al.* [10] uses plane detected in images by a CNN to infer the non-cuboid 3D room layouts. The main contribution of their work is in the design of a network architecture to detect planar regions of the layout from a single image and to infer the 3D plane parameters from it. For this task, we use PlaneRCNN [18], which has very similar functionalities. By intersecting these planes, they can be first delineated, and thanks to a clustering and voting scheme, with help of predicted bounding boxes for the planes, the parts of the planes relevant to the layout can be identified.

However, [10] heavily relies on predicted proposal regions to estimate the extents of layout components. As their qualitative results show, it sometimes struggles to find the correct extents as the proposal regions can be very noisy, and the layout components can be disconnected. It also does not provide any quantitative evaluation for the general room layout estimation problem for single views and it is limited to cuboid rooms from the NYUv2 303 dataset [31].

In contrast, we formalize the problem as a discrete optimization problem, allowing us to reason about occlusions between layout components in the camera view, which often happens in practice, and retrieve structured 3D models.

3 Approach

We describe our approach in this section. We formalize the general layout estimation problem as a constrained discrete optimization problem (Section 3.1), explain how we generate a first set of candidate polygons from plane intersections (Section 3.2), detail our cost function involved in our formalization (Section 3.3), and how we optimize it (Section 3.4). When one or more walls are hidden in the image, this results in an imperfect layout, and we show how to augment the set of candidate polygons to include these hidden walls, and iterate until we obtain the final layout (Section 3.5). Finally, we describe how we can output a structured 3D model for the layout (Section 3.6).

3.1 Formalization

We formalize the problem of estimating a 3D polygonal layout $\hat{\mathcal{R}}$ for a given input image I as solving the following constrained discrete optimization problem:

$$\hat{\mathcal{R}} = \arg \min_{\mathcal{X} \subset \mathcal{R}_0(I)} K(\mathcal{X}, I) \text{ such that } p(\mathcal{X}) \text{ is a partition of } I \,, \tag{1}$$

where $K(\mathcal{X}, I)$ is a cost function defined below, $\mathcal{R}_0(I)$ is a set of 3D polygons for image I, and $p(\mathcal{X})$ is the set of projections in the input view of the polygons in $\mathcal{X}$. In words, we look for the subset of polygons in $\mathcal{R}_0(I)$, whose projections partition the input image I, and that minimizes $K(\cdot)$.

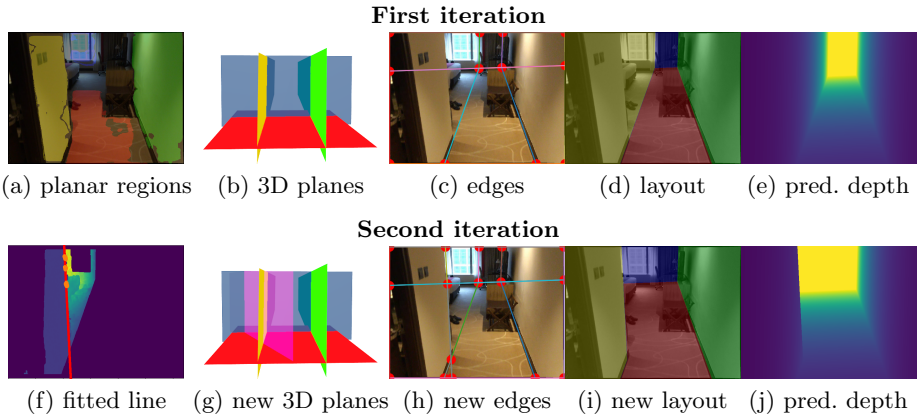


Fig. 2. Approach overview. We detect planar regions (a) for the layout components using PlaneRCNN and a semantic segmentation, and obtain equations of the corresponding 3D planes (b). The planes intersections give a set of candidate edges for the layout (c). From these edges, we find a first layout estimate in 2D (d) and 3D (e) as a set of polygons that minimizes the cost. From the depth discrepancy (f) for the layout estimate and the input view, we find missing planes (g), and extend the set of candidate edges (h). We iterate until we find a layout consistent with the color image (i) and the depth map (j).

There are two options when it comes to defining precisely $K(\mathcal{X}, I)$ and $\mathcal{R}_0(I)$: Either $\mathcal{R}_0(I)$ is defined as the set of all possible 3D polygons, and $K(\mathcal{X}, I)$ includes constraints to ensure that the polygons in $\mathcal{X}$ reproject on image cues for the edges and corners of the rooms, or $\mathcal{R}_0(I)$ contains only polygons with edges that correspond to edges of the room. As discussed in the introduction, extracting wall edges and corners from images is difficult in general, mostly because of lack of training data. We therefore chose the second option. We describe below first how we create the set $\mathcal{R}_0(I)$ of candidate 3D polygons, which includes the polygons constituting the 3D layout, and then the cost function $K(\mathcal{X}, I)$.

3.2 Set of Candidate 3D Polygons $\mathcal{R}_0(I)$

As discussed in the introduction, we rely on the intersection of planes to identify good edge candidates to constitute the polygons of the layout. We then group these edges into polygons to create $\mathcal{R}_0(I)$.

Set of 3D planes $\mathcal{P}_0$. First, we run on the RGB image a) PlaneRCNN [18] to detect planar regions and b) DeepLabv3+ [3] to obtain a semantic segmentation. We keep only the planar regions that correspond to wall, ceiling, or floor segments (the supplementary material provides more details). We denote by $\mathcal{S}(I)$ the set of such regions. An example is shown in Fig. 2(a). PlaneRCNN provides the equations of the 3D planes it detects, or, if a depth map of the image is available, we fit a 3D plane to each detected region to obtain more accurate parameters. The depth map can be measured or predicted from the input image I [18, 25].

As can be seen in Fig. 2(a), the regions provided by PlaneRCNN typically do not extend to the full polygonal regions that constitute the layout. To find these polygons, we rely on the intersections of the planes in $\mathcal{P}_0$ as detailed below. In order to limit the extent of the polygons to the borders of the input image, we also include in $\mathcal{P}_0$ the four 3D planes of the camera frustum, which pass through two neighbouring image corners and the camera center.

Some planes required to create some edges of the layout may not be in this first set $\mathcal{P}_0$. This is the case for example for the plane of the hidden wall on the left of the scene in Fig. 2. Through an analysis-by-synthesis approach, we can detect the absence of such planes, and add plausible planes to recover the missing edges and obtain the correct layout. This will be detailed in Section 3.5.

Set of 3D corners $\mathcal{C}_0$. By computing the intersections of each triplet of planes in $\mathcal{P}_0$, we get a set $\mathcal{C}_0$ of candidate 3D corners for the layout. To build a structured layout, it is important to keep track of the planes that generated the corners and, thus, we define each corner $C_j \in \mathcal{C}_0$ as a set of 3 planes:

$$C_j = \{P_j^1, P_j^2, P_j^3\}\,, \tag{2}$$

where $P_j^1 \in \mathcal{P}_0$, $P_j^2 \in \mathcal{P}_0$, $P_j^3 \in \mathcal{P}_0$, and $P_j^1 \neq P_j^2$, $P_j^1 \neq P_j^3$, and $P_j^2 \neq P_j^3$. For numerical stability, we do not consider the cases where at least two planes are almost parallel, or when the 3 planes almost intersect on a line. Furthermore, we discard the corners that reproject outside the image. We also discard those corners that have negative depth values.

Set of 3D edges $\mathcal{E}_0$. We then obtain a set $\mathcal{E}_0$ of candidate 3D edges by pairing the corners in $\mathcal{C}_0$ that share exactly 2 planes:

$$E_k = \{C_{\sigma(k)}, C_{\sigma'(k)}\}\,, \tag{3}$$

where $\sigma(k)$ and $\sigma'(k)$ are 2 functions giving the indices of the corners that are the extremities of edge E_k. Fig. 2(c) gives an example of set $\mathcal{E}_0$.

Set of 3D polygons $\mathcal{R}_0(I)$. We finally create the set $\mathcal{R}_0(I)$ of candidate polygons as the set of all closed loops of edges in $\mathcal{E}_0$ that lie on the same plane and do not intersect each other.

3.3 Cost Function $K(\mathcal{X}, I)$

Our cost function is split into a 3D and a 2D part:

$$K(\mathcal{X}, I) = K_{3\text{D}}(\mathcal{X}, I) + \lambda K_{2\text{D}}(\mathcal{X}, I)\,. \tag{4}$$

For all our experiments, we used $\lambda = 1$.

Cost function $K_{3\text{D}}(\cdot)$ measures the dissimilarity with the depth map $D(I)$ for the input view, and the depth map $D'(\mathcal{X})$ created from the polygons in $\mathcal{X}$, as illustrated in Fig. 2(e). It is based on the observation that the layout should always be located behind the objects of the scene:

$$K_{3\text{D}}(\mathcal{X}, I) = \frac{1}{|I|}\sum_{\mathbf{x}} \max(D(I)[\mathbf{x}] - D'(X)[\mathbf{x}], 0)\,, \tag{5}$$

where the sum is over all the image locations $\mathbf{x}$ and $|I|$ denotes the total number of image locations. Since the projections of the polygons in $\mathcal{X}$ are constrained to form a partition of I, $K_{3D}(\cdot)$ can be rewritten as

$$K_{3D}(\mathcal{X}, I) = \frac{1}{|I|} \sum_{R \in \mathcal{X}} \sum_{\mathbf{x} \in p(R)} \max(D(I)[\mathbf{x}] - D'(\mathcal{X})[\mathbf{x}], 0) = \frac{1}{|I|} \sum_{R \in \mathcal{X}} k_{3D}(R, I) \,, \tag{6}$$

where $p(R)$ is the projection of polygon R in the image. $K_{3D}(\cdot)$ is computed as a sum of terms, each term depending on a single polygon in $\mathcal{X}$. These terms are precomputed for each polygon in $\mathcal{R}_0(I)$, to speed up the computation of $K_{3D}(\cdot)$.

Cost function $K_{2D}(\cdot)$ measures the dissimilarity between the polygons in $\mathcal{X}$ and the image segmentation into planar regions $\mathcal{S}(I)$:

$$\begin{aligned} K_{2D}(\mathcal{X}, I) &= \sum_{R \in \mathcal{X}} \Big(\big(1 - \text{IoU}(p(R), S(I, R))\big) + \text{IoU}(p(R), \mathcal{S}(I) \setminus S(I, R)) \Big) \\ &= \sum_{R \in \mathcal{X}} k_{2D}(R, I) \,, \end{aligned} \tag{7}$$

where IoU is the Intersection over Union score, $S(I, R)$ is the planar region detected by Plane-RCNN and corresponding to the plane of polygon R. Like $K_{3D}(\cdot)$, $K_{2D}(\cdot)$ can be computed as a sum of terms that can be precomputed before optimization. Computing cost function $K(\cdot)$ is therefore very fast.

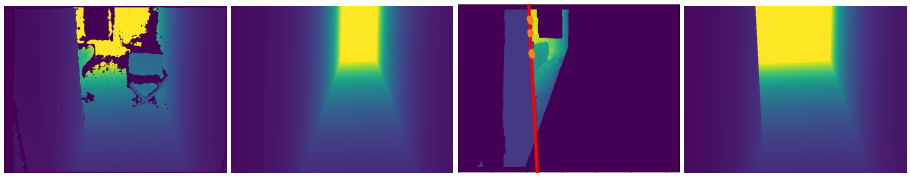
3.4 Optimization

To find the solution to our constrained discrete optimization problem introduced in Eq. (1), we simply consider all the possible subsets $\mathcal{X}$ in $\mathcal{R}_0(I)$ that pass the partition constraint, and keep the one that minimizes $K(\mathcal{X}, I)$.

The number N of polygons in $\mathcal{R}_0(I)$ varies with the scene, but is typically of a few tens. For example, we obtain 12 candidate polygons in total for the example of Fig. 2. The number of non-empty subsets to evaluate is theoretically $2^N - 1$, which is slightly higher than 2000 for the same example. However, most of these subsets can be trivially discarded: Associating polygons with corresponding planes and considering that only one polygon per plane is possible significantly reduces the number of possibilities, to 36 in this example. The number can be further reduced by removing the polygons that do not have a plausible shape to be part of a room layout. Such shapes can be easily recognized by considering the distance between the non-touching edges of the polygon. Finally, this reduces the number to merely 20 plausible subsets of polygons in the case of the example. Precomputing the k_{3D} and k_{2D} terms takes about 1s, and the optimization itself takes about 400ms—most of this time is spent to guarantee the partition constraint in our implementation, which we believe can be significantly improved. The supplementary material details the computation time more.

3.5 Iterative Layout Refinement

As mentioned above in Section 3.2, we often encounter cases where some of the planes required to create the layout are not in $\mathcal{P}_0$ because they are hidden by



(a) (b) (c) (d)

Fig. 3. Layout Refinement. We identify planes which are occluded by other layout planes but necessary for the computation of the layout. First, we compare the depth map for the input view (a) to the rendered layout depth (b). (c) If the discrepancy is large, we fit a line (shown in red) through the points with the largest discrepancy change (orange). By computing the plane passing through the line and the camera center, we obtain a layout (d) consistent with the depth map for the input view.

another layout plane. Fortunately, we can detect such mistakes, and fix them by adding a plane to $\mathcal{P}_0$ before running the layout creation described above again.

To detect missing planes, we render the depth map $D'(\hat{\mathcal{R}})$ for the current layout estimate $\hat{\mathcal{R}}$ and measure the discrepancy with the depth map $D(I)$ for the image as illustrated in Fig. 3. As the depth maps $D(I)$ acquired by RGBD cameras typically contain holes around edges, we use the depth completion method by [13] before measuring the discrepancy. If discrepancy is large, *i.e.* there are many pixel locations where the rendered map has smaller values than the original depth map, this indicates a mistake in the layout estimate that can be fixed by adding a plane. This is because the layout cannot be in front of objects.

There is a range of planes that can improve the layout estimate. We chose the conservative option that does not introduce parts not visible in the input image. For a polygon R in $\hat{\mathcal{R}}$ with a large difference between $D'(\hat{\mathcal{R}})$ and $D(I)$, we first identify the image locations with the largest discrepancy changes, and fit a line to these points using RANSAC, as shown in Fig. 2(f). We then add the plane P that passes through this line and the camera center to $\mathcal{P}_0$ to obtain a new set of planes $\mathcal{P}_1$. This is illustrated in Fig. 2(g): the intersection between P and R will create the edge missing from the layout, which is visible in Fig. 2(h). From $\mathcal{P}_1$, we obtain successively the new sets $\mathcal{C}_1$ (corners), $\mathcal{E}_1$ (edges), and $\mathcal{R}_1$ (polygons), and solve again the problem of Eq. (1) after replacing $\mathcal{R}_0$ by $\mathcal{R}_1$. We repeat this process until we do not improve the differences between $D'(\hat{\mathcal{R}})$ and $D(I)$, for the image locations segmented as layout components.

For about 5% of the test samples, the floor plane is not visible because of occlusions by some furniture. When none of the detected planes belongs to the floor class, we create an additional plane by assuming that the camera is $1.5m$ above the floor. For the plane normal, we take the average of the outer products between the normals of the walls and the $[0, 0, 1]^\top$ vector.

3.6 Structured Output

Once the solution $\hat{\mathcal{R}}$ to Eq. (1) is found, it is straightforward to create a structured 3D model for the layout. Each 3D polygon in $\hat{\mathcal{R}}$ is defined as a set of coplanar 3D edges, each edge is defined as a pair of corners, and each corner is defined from 3 planes. We therefore know which corners and edges the polygons share and how they are connected to each other. For example, the 3D layout of Fig. 1 is made of 14 corners, 18 edges, and 5 polygons.

4 Evaluation

We evaluate our approach in this section. First, we present our new benchmark for evaluating 3D room layouts from single perspective views, and our proposed metrics. Second, we evaluate our approach on our benchmark and include both quantitative and qualitative results on general room layouts. For reference, we show that our approach performs similarly to methods assuming cuboid layouts on the NYUv2 303 benchmark, which only includes cuboid layouts, without making such strong assumptions. More qualitative results, detailed computation times, and implementation details are given in the supplementary material.

We have considered additionally evaluating our approach on the LSUN [33] and the Hedau [8] room layout benchmarks. However, because these datasets do not provide the camera intrinsic parameters, these datasets were unpractical for the evaluation of our approach. We have also considered evaluating our approach on 3D layout annotations of the NYUv2 dataset [28] from [7]. However, as the annotations are not publicly available anymore, we were not able to produce any quantitative results for this dataset. Furthermore, the improved annotations from [37] are not publicly available anymore, and the authors were unfortunately not able to provide these annotations in time for this submission.

4.1 ScanNet-Layout Benchmark

Dataset creation. For our ScanNet-Layout dataset, we manually labelled 293 views sampled from the 100 ScanNet test scenes [5], for testing purposes only. As shown in Fig. 4 and in the supplementary material, these views span different layout settings, are equally distributed to represent both cuboid and general room layouts, challenging views that are neglected in previous room layout datasets, and in some cases we include similar viewpoints to evaluate effects of noise (*e.g.* motion blur). The ScanNet-Layout dataset is available on our project page.

To manually annotate the 3D layouts, we first drew the layout components as 2D polygons. For each polygon, we then annotated the image region where it is directly visible without any occlusions from objects or other planes. From these regions, we could compute the 3D plane equations for the polygons. Since we could not recover the plane parameters for completely occluded layout components, we only provide 2D polygons without 3D annotations for them.

Evaluation metrics. To quantitatively evaluate the fidelity of the recovered layout structures and their 2D and 3D accuracy, we introduce 2D and 3D metrics.

Fig. 4. Results of our method on the ScanNet-Layout. First row: Manual annotations; Second row: Predictions using an RGBD input; Third row: Predictions using an RGB input. Fourth row: 3D models created using the RGBD mode of our approach. Furniture is shown only to demonstrate consistency of our predictions with the geometry of the scene. Our approach performs well in both RGBD and RGB modes. Our approach in RGB mode fails to detect one component in the third example, due to noisy predictions from PlaneRCNN. The rest of the examples show that, when depth information is not available, predictions from CNN can still be utilized in many different scenarios. More qualitative results, including a video, can be found in the supplementary material.

For the 2D metrics, we first establish one-to-one correspondences $\mathcal{C}$ between the N predicted polygons $\hat{\mathcal{R}}$ and the M ground truth polygons $\mathcal{R}_{\text{gt}}$. Starting with the largest ground truth polygon, we iteratively find the matching predicted polygon with highest intersection over union. At each iteration, we remove the ground truth polygon and its match from further consideration. The metrics are:

- Intersection over Union (IoU): $\frac{2}{M+N} \sum_{(R_{\text{gt}},R)\in\mathcal{C}} \text{IoU}(R_{\text{gt}}, R)$, where IoU is the Intersection-over-Union measure between the projections of the 2 polygons. This metric is very demanding on the global structure and 2D accuracy;
- Pixel Error (PE): $\frac{1}{|I|} \sum_{\mathbf{x}\in I} \text{PE}(\mathbf{x})$, with $\text{PE}(\mathbf{x}) = 0$ if the ground truth polygon and the predicted polygon projected at image location $\mathbf{x}$ were matched together, and 1 otherwise. This metric also evaluates the global structure;
- Edge Error (EE): This is the symmetrical Chamfer distance [23] between the polygons in $\hat{\mathcal{R}}$ and $\mathcal{R}_{\text{gt}}$, and evaluates the accuracy of the layout in 2D;
- Root Mean Square Error (RMSE) between the predicted layout depth $D(\mathcal{R})$ and the ground truth layout depth $D(\mathcal{R}_{\text{gt}})$, excluding the pixels that lie on

	Mode	IoU ↑ (%)	PE ↓ (%)	EE ↓	RMSE ↓	RMSE_{uts} ↓
Hirzer [9]	RGB	48.6 ± 22.2	24.1 ± 15.1	29.6 ± 19.4	-	-
Ours	RGB	*63.5 ± 25.2*	*16.3 ± 14.7*	*22.3 ± 14.9*	*0.5 ± 0.5*	*0.4 ± 0.5*
Ours	RGBD	$\mathbf{75.9 \pm 23.4}$	$\mathbf{9.9 \pm 12.9}$	$\mathbf{11.9 \pm 13.2}$	$\mathbf{0.2 \pm 0.4}$	$\mathbf{0.2 \pm 0.3}$

Table 2. Quantitative results on our ScanNet-Layout benchmark. (↑: higher values are better, ↓: lower values are better) The numbers for Hirzer *et al.* [9] demonstrate that the approaches assuming cuboid layouts under-perform on our ScanNet-Layout benchmark. Our approach in RGB performs much better as it is not restricted by these assumptions. Our approach in RGBD mode shows even more improvement.

completely occluded layout components, as we could not recover 3D data for these components. This metric evaluates the accuracy of the 3D layout.

– RMSE_{uts} that computes the RMSE after scaling the predicted layout depth to the range of ground truth layout depth by factor $s = \text{median}(D(\mathcal{R}_{\text{gt}}))$ / $\text{median}(D(\mathcal{R}))$. This metric is used when the depth map is predicted from the image, as the scale of depth prediction methods is not reliable.

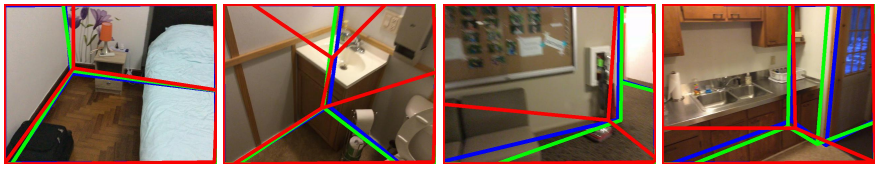
We note that the PE and EE metrics are extensions of existing metrics in cuboid layout benchmarks. As the PE metric is forgiving when missing out small components, we introduce the IoU metric that drastically penalizes such errors.

4.2 Evaluation on ScanNet-Layout

We evaluate our method on ScanNet-Layout under two different experimental settings: When depth information is directly measured by a depth camera, and when only the color image is available as input. In this case, we use PlaneR-CNN [18] to estimate both the planes parameters and the depth map.

Table 2 reports the quantitative results. The authors of [9], one of the state-of-the-art methods for cuboid layout estimation, kindly provided us with the results of their method. As this method is specifically designed for cuboid layouts, it fails on more general cases, but also on many cuboid examples for viewpoints not well represented in the training sets of layout benchmarks [8, 31, 33] (Fig. 5(b)).

The good performance of our method for all metrics shows that the layouts are recovered accurately in 2D and 3D. When measured depth is not used, performance decreases due to noisy estimates of the plane parameters, but in many cases, the predictions are still accurate. This can be best observed in qualitative comparisons with RGBD and RGB views in Fig. 4. In many cases, RGB information is enough to estimate 3D layouts and are comparable to results with RGBD information. However, the third example clearly demonstrates that small errors in planes parameters can lead to visible errors.



(a) (b) (c) (d)

Fig. 5. Visual comparison to Hirzer *et al.* [9], which assumes only cuboid layouts. The layouts estimated by the Hirzer method are shown in red, the layouts recovered by our approach using RGB information only are shown in green, and ground truth is shown in blue. (a) Both approaches perform similarly. (b) The Hirzer method makes a mistake even for this cuboid layout. (c) and (d) The Hirzer method fails as the cuboid assumption does not hold. Our approach performs well in all of the examples.

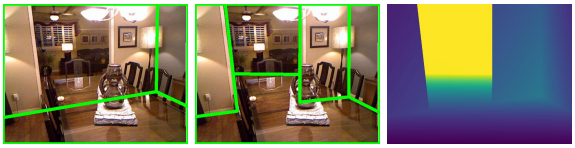
	Mode	PE ↓	Median PE ↓
Zhang *et al.* [31]	RGBD	8.04	-
Ours	RGBD	8.9	4.6
Schwing *et al.* [27]	RGB	13.66	-
Zhang *et al.* [31]	RGB	13.94	-
RoomNet [16] (from [9])	RGB	12.31	-
Hirzer *et al.* [9]	RGB	8.49	-
Howard-Jenkins *et al.* [10]	RGB	12.19	-
Ours	RGB	13.0	10.1

Table 3. Quantitative results on NYUv2 303, a standard benchmark for cuboid room layout estimation. Our method performs similarly to the other methods designed for cuboid rooms without using this assumption. While Hirzer *et al.* [9] performs best on this benchmark, it fails on ScanNet-Layout, even for some of the cuboid rooms (Fig. 5).

4.3 Evaluation on NYUv2 303

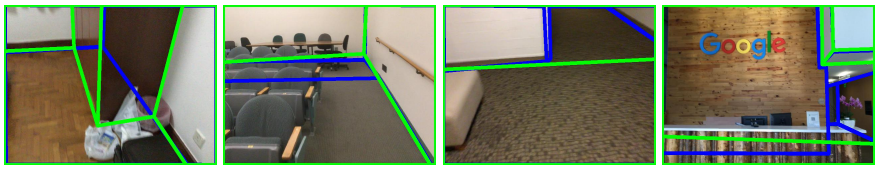
For reference, we evaluate our approach on the NYUv2 303 dataset [28, 31]. It is designed to evaluate methods predicting 2D room layouts under the cuboid assumption. We show that our method also performs well on it without exploiting this assumption. This dataset only provides annotations for the room corners under the cuboid assumption. Since the output of our method is more general, we transform it into the format expected by the NYUv2 303 benchmark. For each of the possible cuboid layout components—1 floor, 1 ceiling, 3 walls—we find the planes for which its normal vector best fits the layout component: When fewer than 3 walls are visible, the annotations of walls in the dataset are ambiguous and we apply the Hungarian algorithm [14] to find good correspondences.

Table 3 gives the quantitative results. When depth is available, our method is slightly worse than the Zhang *et al.* [31] method, designed for cuboid rooms. When using only color images, our method performs similarly to the other approaches, specialized for cuboid rooms, even if the recent Hirzer *et al.* [9] method



(a) (b) (c)

Fig. 6. Qualitative result on NYUv2 303. (a) Layout obtained by enforcing the cuboid assumption, (b) the original layout retrieved by our method, which corresponds better to the scenes. (c) Depth map computed from our estimated layout.



(a) (b) (c) (d)

Fig. 7. Failure cases on ScanNet-Layout, our estimations in green, ground truth in blue. (a): Some furniture were segmented as walls. (b): PlaneRCNN did not detect the second floor plane. Even human observers may fail to see this plane. (c): Large areas in the measured depth map were missing along edges. Filling these areas with [13] is not always sufficient to detect discrepancy. (d): As the floor is not visible in the image, it is unclear whether the manual annotation or the estimated floor polygon is correct.

performs best on this dataset. Here, we used the depth maps predicted by [25] to estimate the plane parameters. Fig. 6 shows that some layouts we retrieved fit the scene better than the manually annotated layout. To conclude, our method performs closely but is more general than the cuboid-based methods.

4.4 Failure Cases

Fig. 7 shows the frequent causes of failures. Most of the failures are due to noisy outputs from PlaneRCNN and DeepLabv3+ that lead to both false-positive and missing layout planes. Our render-and-compare approach is not robust enough to large noise in depth and this should be addressed in future work.

5 Conclusion

We presented a formalization of the general room layout estimation into a constrained discrete optimization problem, and an algorithm to solve this problem. The occasional errors made by our method come from the detection of the planar regions, the semantic segmentation, and the predicted depth maps, pointing to the fact that future progress in these fields will improve our layout estimates.

Acknowledgment. This work was supported by the Christian Doppler Laboratory for Semantic 3D Computer Vision, funded in part by Qualcomm Inc.

References

1. Budroni, A., Boehm, J.: Automated 3D Reconstruction of Interiors from Point Clouds. International Journal of Architectural Computing (2010)
2. Cabral, R., Furukawa, Y.: Piecewise Planar and Compact Floorplan Reconstruction from Images. In: Conference on Computer Vision and Pattern Recognition (2014)
3. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In: European Conference on Computer Vision (2018)
4. Chen, Y., Huang, S., Yuan, T., Qi, S., Zhu, Y., Zhu, S.C.: Holistic++ Scene Understanding: Single-View 3D Holistic Scene Parsing and Human Pose Estimation with Human-Object Interaction and Physical Commonsense. In: International Conference on Computer Vision (2019)
5. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Niessner, M.: Scannet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In: Conference on Computer Vision and Pattern Recognition (2017)
6. Godard, C., Aodha, O.M., Brostow, G.J.: Unsupervised Monocular Depth Estimation with Left-Right Consistency. In: Conference on Computer Vision and Pattern Recognition (2017)
7. Guo, R., Hoiem, D.: Support Surface Prediction in Indoor Scenes. In: International Conference on Computer Vision (2013)
8. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the Spatial Layout of Cluttered Rooms. In: International Conference on Computer Vision (2009)
9. Hirzer, M., Roth, P.M., Lepetit, V.: Smart Hypothesis Generation for Efficient and Robust Room Layout Estimation. IEEE Winter Conference on Applications of Computer Vision (2020)
10. Howard-Jenkins, H., Li, S., Prisacariu, V.: Thinking Outside the Box: Generation of Unconstrained 3D Room Layouts. In: Asian Conference on Computer Vision (2019)
11. Huang, S., Qi, S., Xiao, Y., Zhu, Y., Wu, Y.N., Zhu, S.C.: Cooperative Holistic Scene Understanding: Unifying 3D Object,layout, and Camera Pose Estimation. In: Advances in Neural Information Processing Systems (2018)
12. Ikehata, S., Yang, H., Furukawa, Y.: Structured Indoor Modeling. In: International Conference on Computer Vision (2015)
13. Ku, J., Harakeh, A., Waslander, S.L.: In Defense of Classical Image Processing: Fast Depth Completion on the CPU. In: CRV (2018)
14. Kuhn, H.W., Yaw, B.: The Hungarian Method for the Assignment Problem. Naval Res. Logist. Quart (1955)
15. Kundu, A., Li, Y., Rehg, J.M.: 3D-RCNN: Instance-Level 3D Object Reconstruction Via Render-And-Compare. In: Conference on Computer Vision and Pattern Recognition (2018)
16. Lee, C.Y., Badrinarayanan, V., Malisiewicz, T., Rabinovich, A.: Roomnet: End-To-End Room Layout Estimation. In: International Conference on Computer Vision (2017)
17. Lee, J.H., Han, M.K., Ko, D.W., Suh, I.H.: From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation. In: arXiv Preprint (2019)
18. Liu, C., Kim, K., Gu, J., Furukawa, Y., Kautz, J.: Planercnn: 3D Plane Detection and Reconstruction from a Single Image. In: Conference on Computer Vision and Pattern Recognition (2019)

19. Liu, C., Wu, J., Furukawa, Y.: Floornet: A Unified Framework for Floorplan Reconstruction from 3D Scans. In: European Conference on Computer Vision (2018)
20. Mallya, A., Lazebnik, S.: Learning Informative Edge Maps for Indoor Scene Layout Prediction. In: International Conference on Computer Vision (2015)
21. Murali, S., Speciale, P., Oswald, M.R., Pollefeys, M.: Indoor Scan2BIM: Building information models of house interiors. In: International Conference on Intelligent Robots and Systems (2017)
22. Nan, L., Wonka, P.: Polyfit: Polygonal Surface Reconstruction from Point Clouds. In: International Conference on Computer Vision (2017)
23. Olson, C.F., Huttenlocher, D.P.: Automatic Target Recognition by Matching Oriented Edge Pixels. Journal of Machine Learning Research (1997)
24. Ramalingam, S., Pillai, J.K., Jain, A., Taguchi, Y.: Manhattan Junction Catalogue for Spatial Reasoning of Indoor Scenes. In: Conference on Computer Vision and Pattern Recognition (2013)
25. Ramamonjisoa, M., Lepetit, V.: SharpNet: Fast and Accurate Recovery of Occluding Contours in Monocular Depth Estimation. In: International Conference on Computer Vision Workshops (2019)
26. Sanchez, V., Zakhor, A.: Planar 3D modeling of building interiors from point cloud data. In: International Conference on Computer Vision (2012)
27. Schwing, A.G., Hazan, T., Pollefeys, M., Urtasun, R.: Efficient Structured Prediction for 3D Indoor Scene Understanding. In: Conference on Computer Vision and Pattern Recognition (2012)
28. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor Segmentation and Support Inference from RGBD Images. In: European Conference on Computer Vision (2012)
29. Sun, C., Hsiao, C.W., Sun, M., Chen, H.T.: Horizonnet: Learning Room Layout with 1D Representation and Pano Stretch Data Augmentation. In: Conference on Computer Vision and Pattern Recognition (2019)
30. Xu, Y., Zhu, S.C., Tung, T.: DenseRaC: Joint 3D Pose and Shape Estimation by Dense Render-and-Compare. In: International Conference on Computer Vision (2019)
31. Zhang, J., Kan, C., Schwing, A.G., Urtasun, R.: Estimating the 3D Layout of Indoor Scenes and Its Clutter from Depth Sensors. In: International Conference on Computer Vision (2013)
32. Zhang, W., Zhang, W., Gu, J.: Edge-Semantic Learning Strategy for Layout Estimation in Indoor Environment. IEEE Transactions on Cybernetics (2019)
33. Zhang, Y., Yu, F., Song, S., Xu, P., Seff, A., Xiao, J.: Large-Scale Scene Understanding Challenge: Room Layout Estimation. In: Conference on Computer Vision and Pattern Recognition Workshops (2015)
34. Zheng, J., Zhang, J., Li, J., Tang, R., Gao, S., Zhou, Z.: Structured3D: A Large Photo-realistic Dataset for Structured 3D Modeling. In: European Conference on Computer Vision (2020)
35. Zou, C., Colburn, A., Shan, Q., Hoiem, D.: LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image. In: Conference on Computer Vision and Pattern Recognition (2018)
36. Zou, C., Su, J.W., Peng, C.H., Colburn, A., Shan, Q., Wonka, P., Chu, H.K., Hoiem, D.: 3D Manhattan Room Layout Reconstruction from a Single 360 Image. In: arXiv Preprint (2019)
37. Zou, C., Guo, R., Li, Z., Hoiem, D.: Complete 3D Scene Parsing From an RGBD Image. International Journal of Computer Vision (2019)