PixCuboid: Room Layout Estimation from Multi-view Featuremetric Alignment

Gustav Hanning Kalle Åström Viktor Larsson
Lund University

Abstract

Coarse room layout estimation provides important geometric cues for many downstream tasks. Current state-of-the-art methods are predominantly based on single views and often assume panoramic images. We introduce PixCuboid, an optimization-based approach for cuboid-shaped room layout estimation, which is based on multi-view alignment of dense deep features. By training with the optimization end-to-end, we learn feature maps that yield large convergence basins and smooth loss landscapes in the alignment. This allows us to initialize the room layout using simple heuristics. For the evaluation we propose two new benchmarks based on ScanNet++ and 2D-3D-Semantics, with manually verified ground truth 3D cuboids. In thorough experiments we validate our approach and significantly outperform the competition. Finally, while our network is trained with single cuboids, the flexibility of the optimization-based approach allow us to easily extend to multi-room estimation, e.g. larger apartments or offices. Code and model weights are available at https://github.com/ghanning/PixCuboid.

1. Introduction

For indoor scenes, knowing the room layout (i.e. position of the walls, ceiling, and floor) can provide important geometric cues for downstream applications. For example, for anchoring virtual content to walls in AR applications. Room layouts can also serve as a map for localization problems with low-fidelity sensors (e.g. radar, ultrasonics or 1D lidars). In such cases, the layout provides an abstract representation of the scene, excluding clutter and retaining only key structural elements. Recent methods have focused on the monocular case, predicting the full room layout from a single image or panorama. In this paper, we argue that only considering single images makes the task unnecessarily hard, since in many application contexts multi-view imagery is readily available. Instead, we consider solving the room layout estimation problem from a **collection of posed images**, e.g. obtained via SLAM or Structure-from-Motion. Knowing the camera geometry greatly simplifies the problem as it resolves the global scale and the parallax between views allows for proper geometric reasoning about the extent of the room. Most methods currently estimate the room layout in a feed-forward manner via regression directly from the images. In contrast, we propose an optimization-based approach which can naturally integrate image and pose information from an arbitrary number of images. Inspired by PixLoc [30], the optimization is based on direct alignment of learned dense feature maps. This is performed in a coarse-to-fine manner, allowing our method to obtain high accuracy results even from poor initial estimates. The feature extraction network is trained end-to-end with unrolled optimization to ensure a smooth loss landscape for the alignment, leading to large convergence basins. As our method operates directly on the RGB images, it does not require performing costly dense 3D reconstruction, instead fitting directly to the pixels. In the paper, we focus on room shapes consisting of a single cuboid. However, neither the learned features or optimization framework is specific for cuboids, but can in principle be applied to any parametric room representation consisting of flat surfaces that allow warping between images, e.g. 3D polygons or compositions of multiple cuboids.

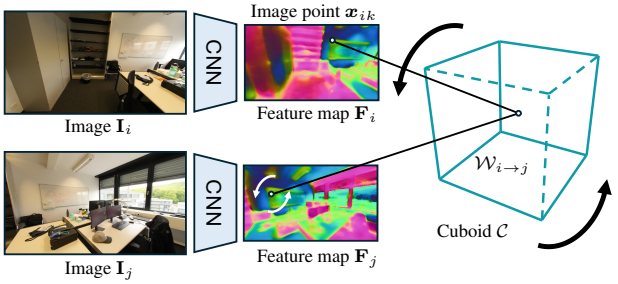

Figure 1. **Featuremetric alignment with PixCuboid.** From the posed input images $\{\mathbf{I}_i\}$ (two or more) we extract feature maps $\{\mathbf{F}_i\}$. Points $\{\boldsymbol{x}_{ik}\}$ are sampled in each feature map and warped ($\mathcal{W}_{i\to j}$) via the cuboid $\mathcal{C}$ to the other views. We find the optimal cuboid by minimizing the featuremetric error (Eq. (2)).

In the paper, we make the following contributions:

- We propose a featuremetric approach for room layout estimation based on multi-view alignment of deep features.
- We propose a simple cuboid initialization heuristic which only relies on the orientation and position of the cameras.
- We provide new benchmarks based on ScanNet++ v2 [46] and 2D-3D-Semantics [2] where we provide manually verified ground truth 3D cuboids and code for evaluation.

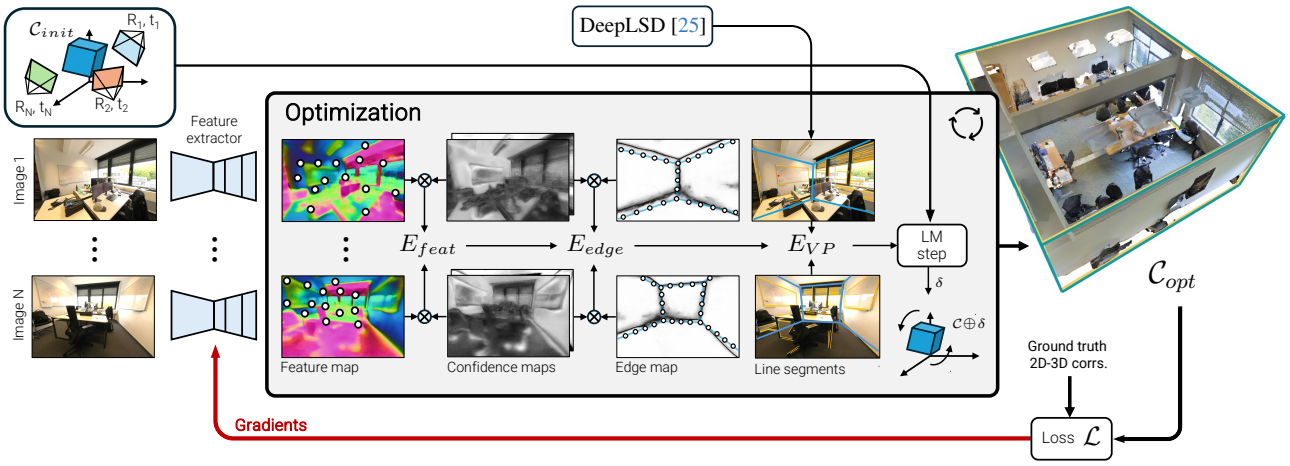

Figure 2. **Room Geometry from PixCuboid**. Taking only posed images as input, PixCuboid estimates the room cuboid $\mathcal{C}_{opt}$ by optimizing from a coarse initial estimate $\mathcal{C}_{init}$ (see Figure 6 for examples). First, a deep network predicts dense feature-, confidence- and edge maps for each image. The optimization then minimizes a combination of three terms: **a)** a multi-view featuremetric alignment which warps features between images using the cuboid faces, **b)** a monocular edge cost which tries to align the projected cuboid with the learned edge map, and **c)** a vanishing point-based cost that aligns the orientation of the cuboid with lines detected in the image. The network is trained end-to-end by supervising on the result of the optimization, propagating gradients back to the network through the optimization steps.

2. Related Work

Featuremetric alignment, minimizing the distance between extracted features, has been used to tackle a wide variety of problems, for example monocular depth and egomotion estimation [32], camera tracking [44], structure-from-motion [18] and visual localization [30]. These methods typically extract features with a convolutional neural network to improve accuracy and robustness as compared to photometric alignment. Similar to our approach, [44] and [30] apply featuremetric alignment at multiple scales, sequentially refining the camera pose and their networks are trained end-to-end via differentiable optimization steps. In contrast, we have fixed poses and optimize the scene representation (cuboid).

Room layout estimation from a single view, inferring the location of the floor, ceiling and walls from an image captured indoors, is a well-studied problem. Early algorithms [10, 16] predicted the layout from one perspective image and relied on geometric reasoning with e.g. vanishing points and line segments. With the emergence of deep learning they were superseded by methods [15, 22, 24, 49] trained on annotated datasets [33]. Due to the limited field-of-view of the perspective image a lot of focus has recently been given to room layout estimation from a 360° equirectangular panorama, which contains a more complete view of the surrounding environment. A wide range of architectures have been suggested, for example recurrent [36, 39], graph convolutional [28] and transformer-based [8, 35, 40] neural networks. A common approach [35, 36, 39, 50] is to identify boundaries between floor, wall and ceiling in the image, from which the layout is derived. Monocular methods, however, suffer from scale ambiguity and often assume knowledge about the camera height relative to the floor to fix the scale. Another approach is to use multiple views, which only a few prior works have considered. PSMNet [40] predicts the layout from a pair of panoramic images but requires an approximate relative pose as input, a limitation that the later GPR-Net [35] gets rid of by direct regression with a pose transformer. MVLayoutNet [11] and the multi-view method of Pintore et al. [26] can leverage more than two panoramas, but as no public implementations exist we do not compare against these networks. In this work we formulate layout estimation as an optimization problem that combines geometric cues with learned deep features, extracted from a simple U-Net style network architecture. Our proposed method PixCuboid uses multiple posed perspective views for a larger context and to resolve the global scale.

Many layout estimation methods make assumptions about the room shape. Due to its simplicity a box or cuboid assumption is commonly employed [10, 24, 49]. A less restrictive model is the ”Manhattan world” where walls meet at right angles [36, 39, 50]. Lee et al. [16] proposed the ”indoor world” model which extends the Manhattan world with a single-floor, single-ceiling constraint. In this work we focus on single cuboid-shaped rooms, but show that our optimization-based approach is flexible and can be extended to other setups, e.g. apartments consisting of multiple cuboids.

Layout from 3D point clouds and RGB-D. Several prior works have used laser scans or RGB-D sensors to reconstruct indoor scenes. Xiao and Furukawa [42] present a system to produce 3D models from laser points, textured with ground-level photographs to create interactive maps of museums.

Later, DNN-based methods [5, 19, 48] extract floor plans from point clouds captured with RGB-D sensors. In contrast, PixCuboid operates on posed RGB images and does not rely on additional information from lasers or depth sensors.

Plane reconstruction is the task of detecting planar surfaces in images and differs from room layout estimation in that not only the floor, ceiling and walls should be reconstructed. Both single view [20, 47] and multi-view [1, 13, 41] methods exist. PlanarRecon [43] and UniPlane [12] are two recent methods that focus on plane reconstruction from posed monocular videos by the use of 3D feature volumes.

For an in-depth review of reconstruction methods for indoor environments we refer the interested reader to the survey paper of Pintore et al. [27].

3. Method

We assume that the room shape can be represented by a cuboid $\mathcal{C}$, where the six faces correspond to the walls, floor and ceiling. As input, our method takes a collection of images $\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_n$ captured inside the room, together with their camera poses $(\boldsymbol{R}_i, \boldsymbol{t}_i)$ and intrinsics $\boldsymbol{K}_i$.

To estimate the room cuboid $\mathcal{C}$ we propose an optimization-based approach where an initial cuboid is refined using both multi-view consistency and monocular cues. Each image is passed independently through a CNN that produce dense feature maps, which are used to define the cost function in the optimization. Similar to [30], the network is trained end-to-end by only supervising on the result of the cuboid optimization, ensuring that the features we learn provide useful cues for the cuboid fitting. See Fig. 2 for an overview of our method which we call PixCuboid.

In the next section we detail the cuboid parameterization, followed by definition of the cost functions (Sec. 3.2). Sec. 3.3 describes how the feature extractor is trained and finally in Sec. 3.4 we propose a simple heuristic for finding initial cuboid estimates.

3.1. Cuboid Parameterization

Let $\boldsymbol{R} \in SO(3)$ be the rotation that rotates the coordinate system such that the cuboid is axis aligned. We then parameterize the offsets $\boldsymbol{d} \in \mathbb{R}^6$ for each of the six faces of the cuboid along either the x-, y- or z-axis. So the first face is defined by the plane $(1,\ 0,\ 0)\boldsymbol{X} = d_1$, and so on. Together, the rotation $\boldsymbol{R}$ and vector $\boldsymbol{d}$ then minimally parameterizes the 9 degrees of freedom of the cuboid $\mathcal{C}$. The translation of the cuboid is encoded in the offsets $\boldsymbol{d}$. Note that the parameterization is not unique, as the axes can be switched by changing the rotation and switching corresponding elements of $\boldsymbol{d}$. An important aspect of this parameterization (compared to e.g. explicitly parameterizing the translation) is that it decouples the parameters for each face. This allow us to easily perform optimization over a subset of faces in cases where the cuboid is only partially observable.

3.2. Geometric Optimization of Cuboids

To refine cuboids we define a cost function depending on three terms: **a)** a featuremetric cost E_{feat}, measuring multi-view consistency by performing alignment between feature maps, **b)** a monocular edge cost E_{edge}, forcing the projected edges of the cuboid to align with a predicted edge map, and **c)** a VP-based cost E_{VP}, where the vanishing points defined by the cuboid is compared with extracted line segments. The full cost function is then given by

$$E(\mathcal{C}) = E_{feat}(\mathcal{C}) + \alpha E_{edge}(\mathcal{C}) + \beta E_{VP}(\mathcal{C}). \quad (1)$$

If 2D line segments are not available we set $\beta = 0$. The three terms are further detailed in the following paragraphs.

Featuremetric cost: Inspired by the success of PixLoc [30], we leverage featuremetric alignment. From each image $\mathbf{I}_i \in \mathbb{R}^{W\times H\times 3}$ a dense feature map $\mathbf{F}_i \in \mathbb{R}^{W\times H\times D}$ is extracted with a CNN. The featuremetric cost function E_{feat} is then defined by measuring the consistency of the warped features using the faces of the cuboid, i.e. $E_{feat}(\mathcal{C}) =$

$$\sum_{i,j}\sum_{k} w_{ijk}\rho\left(\left\|\mathbf{F}_i[\boldsymbol{x}_{ik}] - \mathbf{F}_j\left[\mathcal{W}_{i\to j}(\boldsymbol{x}_{ik}, \mathcal{C})\right]\right\|^2\right). \quad (2)$$

Here $\{\boldsymbol{x}_{ik}\}$ is a set of sampled image points in image $\mathbf{I}_i$ and $\mathcal{W}_{i\to j}$ represents the warping of these points to image $\mathbf{I}_j$ via the cuboid, i.e. first projecting them onto the planes of $\mathcal{C}$ and then into $\mathbf{I}_j$. This warp is a differentiable function of the cuboid parameters allowing us to optimize over $\mathcal{C}$. $[\cdot]$ denotes lookup with sub-pixel interpolation and ρ is a robust loss function. The residual weights $w_{ijk} = \mathbf{C}_{\mathbf{F}i}[\boldsymbol{x}_{ik}]\mathbf{C}_{\mathbf{F}j}[\mathcal{W}_{i\to j}(\boldsymbol{x}_{ik}, \mathcal{C})]$ are interpolated from the confidence images $\mathbf{C}_{\mathbf{F}i}, \mathbf{C}_{\mathbf{F}j} \in \mathbb{R}^{W\times H}$ which are predicted by the network. We also utilize $\mathbf{C}_{\mathbf{F}i}$ for point sampling: the image points $\{\boldsymbol{x}_{ik}\}$ are drawn, without replacement, from the probability map $\mathbf{C}_{\mathbf{F}i}^{\gamma}$, where $\gamma \in \mathbb{R}$.

Cuboid edge cost: In addition to the feature map $\mathbf{F}_i$ we also let the CNN predict a dense edge map $\mathbf{E}_i \in \mathbb{R}^{W\times H}$ which aims to delineate the room edges. We sample 3D points $\{\boldsymbol{X}_j\}$ uniformly on the twelve edges of the cuboid $\mathcal{C}$. The edge cost E_{edge} is then defined by evaluating the edge map $\mathbf{E}_i$ at the projections of these points, i.e.

$$E_{edge}(\mathcal{C}) = \sum_i \sum_j w_{ij}\mathbf{E}_i[\Pi_i(\boldsymbol{R}_i\boldsymbol{X}_j + \boldsymbol{t}_i)]^2, \quad (3)$$

where $\Pi_i : \mathbb{R}^3 \to \mathbb{R}^2$ is the projection into the image using the known intrinsics $\boldsymbol{K}_i$. As with E_{feat}, we predict a confidence map $\mathbf{C}_{\mathbf{E}i}$ from which the weights w_{ij} are interpolated.

Vanishing point cost: Finally, the VP cost E_{VP} measures consistency between line segments in the images and the orientation of the cuboid. Projected into image $\mathbf{I}_i$, the cuboid defines three vanishing points, given by the columns of $\boldsymbol{R}_i\boldsymbol{R}^T = \begin{bmatrix}\boldsymbol{v}_{i,1} & \boldsymbol{v}_{i,2} & \boldsymbol{v}_{i,3}\end{bmatrix}$. We extract line segments

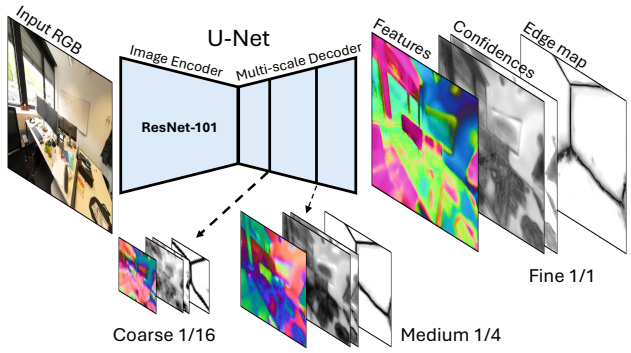

Figure 3. **Network Architecture**. Our feature extractor uses a U-Net [29] architecture consisting of a ResNet-101 image encoder followed by a multi-scale decoder extracting dense feature-, confidence- and edge maps.

$\{\boldsymbol{l}_{ij}\}$ from the image using DeepLSD [25] and use the consistency measure of [38] between line segments and vanishing points to form the vanishing point cost function

$$E_{VP}(\mathcal{C}) = \sum_i \sum_j \min \left(\min_{k \in \{1,2,3\}} D_{VP}(\boldsymbol{l}_{ij}, \boldsymbol{v}_{ik}), \tau \right)^2, \tag{4}$$

where each line is softly assigned to one of the three VPs by only considering the minimum residual. Here

$$D_{VP}(\boldsymbol{l}, \boldsymbol{v}) = |\hat{\boldsymbol{l}}^T \boldsymbol{l}_1| / \sqrt{\hat{l}_1^2 + \hat{l}_2^2} \tag{5}$$

denotes the distance between the line segment endpoint $\boldsymbol{l}_1$ and a line $\hat{\boldsymbol{l}} = \bar{\boldsymbol{l}} \times \boldsymbol{v}$ passing through its midpoint $\bar{\boldsymbol{l}}$ and the vanishing point $\boldsymbol{v}$. We cap the distance at τ to account for line segments that are not aligned with any of the cuboid sides. Note that the lines do not need to coincide with the outline of the room to be consistent with the vanishing points.

Coarse-to-fine optimization: The feature network follows a U-Net style [29] architecture with a ResNet-101 [9] encoder and our multi-scale decoder predicts the feature-, confidence- and edge maps at three different resolutions (see Fig. 3). Similar to [30], the optimization is then performed in a coarse-to-fine manner. For each scale level $s \in \{\text{coarse}, \text{medium}, \text{fine}\}$ the cost is minimized using the corresponding network outputs,

$$\mathcal{C}^s = \arg\min_{\mathcal{C}} E(\mathcal{C}, \mathbf{F}^s, \mathbf{C_F}^s, \mathbf{E}^s, \mathbf{C_E}^s), \tag{6}$$

each optimization initialized from the output of the previous scale, i.e. medium scale is starting from $\mathcal{C}^{\text{coarse}}$ and so on. The minimization is performed using standard Levenberg-Marquardt [17, 23] optimization, iterated until convergence.

3.3. Learning to Optimize Room Layouts

For training we assume that we have posed images together with a ground truth 3D mesh that has semantic labels for walls, ceiling and floor. We train the network end-to-end, propagating gradients through the optimization process. During training, the same coarse-to-fine approach is used, but for each scale s we compute $\mathcal{C}^s$ using a fixed number of unrolled Levenberg-Marquardt iterations minimizing $E(\mathcal{C})$. Supervision for the network is only applied on resulting optimized cuboids $\mathcal{C}^s$. This forces the network to learn features and confidence maps that are useful for optimization.

As rooms are not perfectly cuboid shaped, we propose to supervise using points sampled from the ground truth mesh. We sample points from the mesh corresponding to the wall, ceiling and floor semantic labels and project into the images, filtering for occlusion using the ground truth depth maps. This yields a set of 2D-3D correspondences $\{(\boldsymbol{x}_{ik}^{GT}, \boldsymbol{X}_{ik}^{GT})\}$ for each image. The loss for a cuboid is then computed as

$$\mathcal{L}(\mathcal{C}) = \frac{1}{N} \sum_{i,j} \sum_k \| \mathcal{W}_{i \to j}(\boldsymbol{x}_{ik}^{GT}, \mathcal{C}) - \Pi_j(\boldsymbol{R}_j \boldsymbol{X}_{ik}^{GT} + \boldsymbol{t}_j) \|^2, \tag{7}$$

where N is the total number of points successfully warped between images. The loss is applied after each scale level and our network is trained to minimize

$$\mathcal{L}_{total} = \mathcal{L}(\mathcal{C}^{\text{coarse}}) + \gamma_m \mathcal{L}(\mathcal{C}^{\text{medium}}) + \gamma_f \mathcal{L}(\mathcal{C}^{\text{fine}}), \tag{8}$$

where γ_m and γ_f are the relative weightings of the scales. After each scale we check whether the optimization failed (loss above some threshold), and if so stop gradients to the next scale (setting γ_m or γ_f to zero for this instance). This prevents oversmoothing the fine and medium feature maps.

For the coarse optimization the cuboid parameters are initialized from the ground truth layout (see Sec. 4.1 for details), perturbed by random rotation, translation and scaling, so that the network is exposed to examples of varying difficulty throughout the training. In addition to the network parameters we learn a separate LM dampening factor for each of the nine cuboid parameters similar to [30].

Pre-training: As the training procedure requires the network to at least be somewhat successful to get meaningful gradients from the optimization, we perform some simple pre-training. While featuremetric alignment can work reasonably with off-the-shelf ImageNet pre-trained features [18] , this is not the case for the edge loss. Therefore we pre-train only the edge map, supervising with a weighted MSE loss against a line-drawing of the projected ground truth cuboid.

3.4. Initializing the Optimization

At inference time the cuboid is initialized based on the camera poses. We set the cuboid z axis to the mean of the camera negative y axes (assumed to be pointing down on average). The x and y axes of the cuboid are selected by randomly sampling two basis vectors in the plane orthogonal to the z axis. We then run a few iterations of optimization with E_{VP} to further enhance the orientation of the initial cuboid.

Plane offsets $\boldsymbol{d}$ are set so that the cuboid contains all cameras with some margin. In Sec. 5.2 we compare this initialization strategy to using uniformly sampled rotation matrices $\boldsymbol{R}$ and also validate the gain of vanishing point optimization.

4. Experimental Setup

For the experiments we use the following settings: $\alpha = 0.05$, $\beta = 40$ and $\tau = 0.05$. The number of LM iterations on each scale level is 15. Cuboids are initialized as described in Sec. 3.4, using a margin of 2.5 m between camera centers and cuboid faces. We additionally expand the cuboid after each optimization step if needed so that the cameras are at least 0.1 m from the faces.

4.1. Datasets

ScanNet++: From ScanNet++ v2 [46], an indoor dataset comprised of 1006 scenes of varying types, we create a new benchmark specifically to evaluate multi-view cuboid room layout estimation. For each scene we use the ground truth semantic mesh to fit a cuboid to the vertices classified as either "floor", "wall" or "ceiling" by minimizing the distance between the cuboid and the vertices using L-BFGS [21]. Since not all scenes are cuboid-shaped, and because the L-BFGS optimization occasionally fails, we manually inspect the results to determine which scenes should be included. This way 391 out of 856 scenes in the existing ScanNet++ v2 training set were selected as a new training set. From the existing validation set we pick 28 out of 50 scenes. As the test scenes of ScanNet++ do not include any semantic mesh we split the 28 validation scenes into new validation and test sets with 10 and 18 scenes, respectively.

For every scene in this new dataset we sample image tuples, consisting of ten random DSLR images, to be used as the input to multi-view room layout estimation methods. For each training scene, 250 tuples are sampled and 20 tuples for each of the validation and test scenes. In total there are hence $391 \times 250 = 97750$ image tuples for training, $10 \times 20 = 200$ for validation and $18 \times 20 = 360$ for testing. For experiments with $k < 10$ images, we select the first k from each tuple.

2D-3D-Semantics: To enable comparison with panorama-based room layout estimation methods, and to assess the generalization capabilities of PixCuboid, we use 2D-3D-Semantics [2]. This is an indoor dataset divided into six areas in three different buildings. Each area is further split into spaces, for example offices or hallways. Similar to ScanNet++ v2 we fit a cuboid to every space by minimizing its distance to points labeled "floor", "wall" or "ceiling" in the supplied point cloud. Again we inspect the results to find cuboid-shaped spaces. In addition, only spaces with at least two panorama images captured within the fitted cuboid are considered. For spaces with more than two such panoramas we randomly select two of them. The result is a set of 160 spaces that can be used for evaluation. As we do not train or tune our model on 2D-3D-Semantics no training/validation/test split is performed. The panorama images for all spaces are divided into four perspective views with 90° horizontal field-of-view to allow comparison between methods that work on the two different types of images.

We will make the ground truth cuboids, image tuples and evaluation code publicly available for both datasets.

4.2. Metrics

We evaluate the estimated layouts with the commonly used 3D Intersection over Union (IoU) and the Chamfer distance between the predicted layout and the ground truth cuboid. For methods that predict a cuboid-shaped layout we compute the rotation error between the two cuboids, taking rotational symmetries into account. We also report the area under the recall curve for the rotation error, using one coarse (20°) and one fine (1°) threshold. These metrics are averaged over image tuples (ScanNet++) or spaces (2D-3D-Semantics). For single-view methods we only consider the prediction with the highest IoU for each tuple/space.

The predicted and ground truth room layouts are rendered into the perspective views and we compute the depth RMSE and the ratio of pixels for which the normal angle error is less than 10°, averaged over all images. For single-view panorama methods the prediction corresponding to the perspective view is used. Finally we measure the mean time in seconds to predict the room layout for one tuple or space.

4.3. Training Details

We train our model on the ScanNet++ training set and tune its hyperparameters on the validation set (using our data split as explained in Sec. 4.1). The five first images in each tuple, which are undistorted beforehand, are used as input. We utilize a two-stage training process as described in Sec. 3.3. First the edge maps are pre-trained with a weighted MSE loss, followed by training of the full network with the loss in Eq. (7). See our supplementary material for a more detailed description of the training process.

5. Results

5.1. Room Layout Estimation

Baselines: On ScanNet++ PixCuboid is compared against the single-view layout estimation methods Total3DUnderstanding [24] and Implicit3DUnderstanding [49]. They both take a single perspective image as input and simultaneously predict the room layout, camera pose and 3D object bounding boxes and meshes. On 2D-3D-Semantics we also compare with three recent panorama-based methods: Deep3DLayout [28], LED^2-Net [39] and PSMNet [40]. Deep3DLayout and LED^2-Net predict the room layout from a single panoramic view while PSMNet uses two panorama images. To the networks LED^2-Net and PSMNet we input

Method	Input	Output	Trained on
Total3D [24]	1×🖼	Cuboid	SUN RGB-D [33], Pix3D [37]
Implicit3D [49]	1×🖼	Cuboid	SUN RGB-D [33], Pix3D [37]
Deep3DLayout [28]	1×🖼	Mesh	MatterportL. [51], Pano3DL. [28]
LED^2-Net [39]	1×🖼	Polygon	Realtor360 [45]
PSMNet [40]	2×🖼	Polygon	ZInD [6]
PixCuboid (ours)	N×🖼	Cuboid	ScanNet++ v2 [46]

Table 1. An overview of the room layout estimation methods included in our experimental evaluation. The methods take a varying number of perspective (🖼) or panoramic (🖼) images as input.

the average camera height over the dataset from the ground truth. PSMNet does not estimate the room height so we give it the height of the corresponding ground truth cuboid as well, along with the relative pose between the two panoramas (without added noise). For all methods we use the authors' official implementations and their provided pre-trained network weights. For Deep3DLayout we utilize the weights fine-tuned on Pano3DLayout [28]. For a summary of all methods, see Tab. 1.

The two-view GPR-Net [35], the multi-view MVLayoutNet [11] and the work of Pintore et al. [26] are also relevant competing methods, but as no public implementations are available we do not include them in our benchmark.

Results: For ScanNet++ we run Total3DUnderstanding, Implicit3DUnderstanding and PixCuboid on the five first images of each image tuple in the test set. As described in Sec. 4.2 we use the best prediction (highest IoU) out of five for the two single-view methods. The results are presented in Tab. 2 (top section). PixCuboid outperforms the two competing methods by a large margin on all metrics (except runtime), however it should be noted that our method is the only one trained on ScanNet++ v2.

On 2D-3D-Semantics the perspective methods (Total3DUnderstanding, Implicit3DUnderstanding) make eight predictions (two panoramas, split into four perspective views) per space. We also run Deep3DLayout and LED^2-Net which predict two room layouts for each space (one per panorama image). Again we emphasize that for all these single-view methods only the top prediction is included in the metrics. PSMNet and PixCuboid outputs a single room layout prediction from the two panoramas and the eight perspective images, respectively (thus taking the same image data as input). We report the results in Tab. 2 (bottom section). The panorama methods generally predict better layouts than the single-view perspective-based ones, but are outperformed by PixCuboid on all metrics.

In Fig. 4 we show qualitative examples of predicted room layouts in 2D-3D-Semantics. For single-view methods the room layout with the highest IoU metric is visualized. Some failure cases of PixCuboid are visualized in Fig. 5. Due to the vanishing point cost E_{VP} our method is often able to estimate the orientation of the cuboid accurately, even when the plane offsets $\boldsymbol{d}$ cannot be properly determined.

5.2. Ablation Experiments

We validate the design of PixCuboid in a series of ablation experiments on our ScanNet++ v2 test set. In addition to previous metrics we also report the success rate on the finest scale (proportion of image tuples with average warp error (7) smaller than 3 pixels).

First, the learned feature maps are compared to those of PixLoc (trained on Extended CMU Seasons [3, 31]) and to using the images directly (i.e. photometric alignment). Here only the featuremetric cost E_{feat} is considered in the LM optimization ($\alpha = \beta = 0$). Results are shown in the top section of Tab. 3, from which it is clear that learning features specifically for room layout estimation is beneficial.

		3D		Rotation			Pixel-wise		
	Method	IoU ↑	Chamfer ↓	Mean ↓	AUC@{1,20}°↑		Depth ↓	Normal ↑	Time ↓
ScanNet	Total3D [24] CVPR20	34.5	1.44 m	13.7°	0.0	37.6	0.86 m	37.2	**0.16 s**
	Implicit3D [49] CVPR21	30.8	1.60 m	16.4°	0.0	31.0	1.02 m	33.1	**0.16 s**
	PixCuboid (ours)	**87.2**	**0.22 m**	**1.3°**	**45.7**	**95.3**	**0.09 m**	**96.1**	0.31 s
2D-3D-S	Total3D [24] CVPR20	31.8	1.57 m	8.9°	0.0	56.9	1.41 m	44.1	0.32 s
	Implicit3D [49] CVPR21	31.6	1.59 m	10.4°	0.0	51.5	1.51 m	37.6	**0.15 s**
	Deep3DLayout [28] TOG21	58.8	0.68 m	N/A	N/A	N/A	0.44 m	52.6	1.30 s
	LED^2-Net† [39] CVPR21	68.7	0.45 m	N/A	N/A	N/A	0.40 m*	34.6*	0.71 s
	PSMNet†‡ [40] CVPR22	43.5	1.04 m	N/A	N/A	N/A	0.63 m*	51.4*	2.32 s
	PixCuboid (ours)	**89.0**	**0.18 m**	**0.7°**	**48.8**	**97.0**	**0.10 m**	**96.1**	0.42 s

Table 2. Room layout estimation results on our ScanNet++ v2 test set and the cuboid-shaped spaces of 2D-3D-Semantics. For single-view methods the metrics are computed using the best prediction for each image tuple or space.

† Uses the ground truth camera height.

‡ Uses the ground truth room height and *vanishing angle* (see code of [40] for details).

* Excludes views (10 for LED^2-Net and 104 for PSMNet, out of 1280) where the layout does not contain the camera.

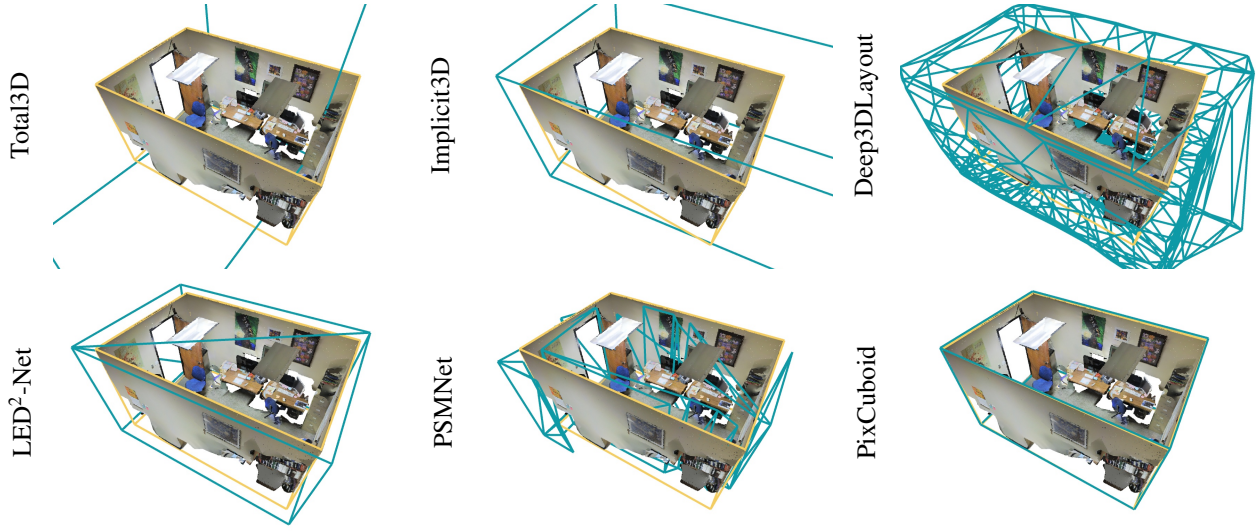

Figure 4. Qualitative comparisons of predicted room layouts for one space in 2D-3D-Semantics. The ground truth point cloud is shown for visualization purposes, but **the methods only have the RGB images as input**. Predictions are shown in **blue** and the ground truth cuboids in yellow. None of the methods are trained on this dataset. For more examples see our supplementary material.

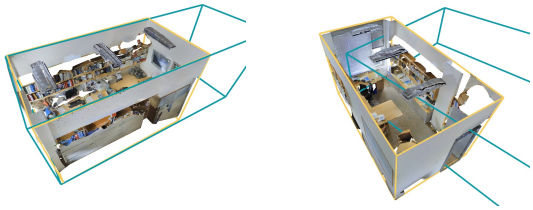

Figure 5. Failure cases of PixCuboid on 2D-3D-Semantics. Predicted room layouts are shown in **blue** and ground truth cuboids in yellow. See our supplementary material for more examples.

Next, we try different combinations of the cost functions E_{feat}, E_{edge} and E_{VP} (Tab. 3, second section). Using E_{edge} on its own (second row) gives better layout predictions than with only featuremetric cost E_{feat} (first row), showcasing the significance of the learned edge map. The two costs complements each other as seen on row 3, where the success rate increases by almost 18 percentage points compared to optimization with just E_{edge}. Adding the vanishing point cost E_{VP} results in better metrics across the board (rows 4-6). This cost is particularly helpful in determining the orientation of the cuboid, with large reductions in the rotation error and better estimation of the surface normals.

To study the impact of point sampling we compare the guided sampling outlined in Sec. 3.2 to random sampling and to sampling from the ground truth points $\boldsymbol{x}_{ik}^{GT}$ that lie on the floor, wall or ceiling. We train models with these different point sampling approaches and give the results in the third section of Tab. 3. Guided sampling is used during evaluation. The results show that training with guided sampling performs the best, although the differences are minor.

		IoU ↑	Rot. ↓	Success ↑
Feat.	RGB	24.2	36.4°	1.4%
	PixLoc (CMU)	25.3	33.4°	1.1%
	PixCuboid (E_{feat} only)	**35.2**	**23.0°**	**5.8%**
Cost	E_{feat}	35.2	23.0°	5.8%
	E_{edge}	76.1	4.7°	43.6%
	E_{feat} + E_{edge}	81.9	3.8°	61.4%
	E_{feat} + E_{VP}	44.6	1.5°	21.7%
	E_{edge} + E_{VP}	83.1	**1.3°**	51.9%
	E_{feat} + E_{edge} + E_{VP}	**87.2**	**1.3°**	**67.2%**
Sampl.	Random	86.9	1.4°	65.8%
	Floor/wall/ceiling	86.6	1.4°	66.4%
	Guided	**87.2**	**1.3°**	**67.2%**
Res.	Low (256 px)	84.2	**1.3°**	63.3%
	Medium (512 px)	**87.2**	**1.3°**	**67.2%**
	High (768 px)	85.7	**1.3°**	66.1%
Init.	Random	60.8	18.0°	40.0%
	Random + VP	72.2	10.9°	51.9%
	Y down	84.9	2.1°	64.7%
	Y down + VP	**87.2**	**1.3°**	**67.2%**
Scales	Coarse	81.0	1.4°	36.7%
	Coarse + medium	86.5	1.4°	65.6%
	Coarse + medium + fine	**87.2**	**1.3°**	**67.2%**

Table 3. Ablation experiments on our ScanNet++ v2 test set. The full set of metrics is available in the supplementary material.

We run PixCuboid on both lower and higher resolution input images (resized to 256 and 768 pixels in height, respectively) and compare to our standard size 512 px, which is what the network sees during training (Tab. 3, section

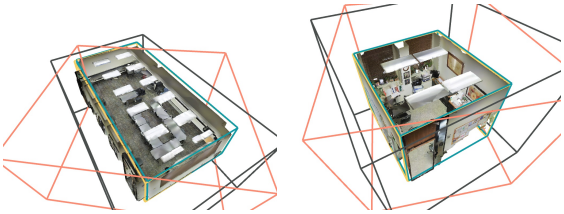

Figure 6. Cuboid initialization examples on 2D-3D-Semantics. Even from poor initial estimates (**red**), PixCuboid can align the cuboids with the help of vanishing points (**gray**) and is able to converge to accurate layouts (**blue**). The ground truth is shown in **yellow**. See our supplementary material for more examples.

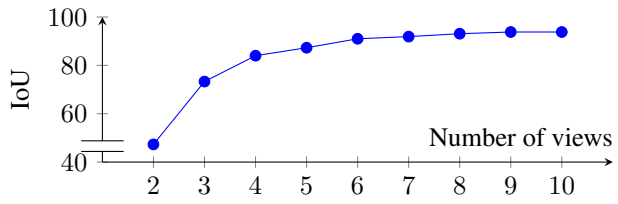

Figure 7. IoU as a function of the number of input views. Results on our ScanNet++ v2 test set.

four). The success threshold is adjusted to account for the difference in image dimensions. Our model performs only slightly worse with the low resolution images. There is no gain in using a higher resolution - the best performance is reached with the medium sized 512 px input.

The importance of cuboid initialization is examined by employing four different strategies. As a baseline we sample the orientation $\boldsymbol{R}$ uniformly and then select plane offsets $\boldsymbol{d}$ so that all cameras are inside the cuboid, with a margin of 2.5 m on each side (Tab. 3, section five, first row). The proposed initialization scheme in Sec. 3.4, where $\boldsymbol{R}$ is set based on the mean camera y axis (assumed to be pointing down), performs significantly better (third row). For both of these initialization procedures we also try running five iterations of cuboid optimization with E_{VP}, after which $\boldsymbol{d}$ is re-computed so that all cameras are again contained by the cuboid with the same 2.5 m margin (rows two & four). We see that this extra step results in improved layouts, especially when using random initialization. Fig. 6 shows examples of cuboids initialized with our proposed method.

Lastly the coarse-to-fine optimization is ablated in the last section of Tab. 3. Here we stop the optimization after the first ("coarse", first row) and second ("medium", second row) scale level and compare to using all three scales (third row). A reasonable room layout is generally found already after the coarse optimization, but it is not so accurate. At the second scale the cuboid is refined, resulting in an increase in the success rate by almost 30 pp. At the last scale level we see only a marginal improvement to the metrics.

In Fig. 7 we also look at how the quality of PixCuboid's predictions vary with the number of input views. As expected the IoU increases with the number of views, with the largest gain seen when going from two to three views.

Figure 8. Multi-room layout estimation on Replica.

5.3. Multi-room Layout Estimation

As a final experiment we apply PixCuboid to a multi-room scenario: a scene from the Replica [34] dataset with four interconnected rooms. We use the trajectory and pre-rendered frames from [4]. A simple algorithm is devised to handle multiple rooms. We start by subsampling the list of frames by a factor of 60, and then loop over the remaining frames in order. When 8 frames have been accumulated, the cuboid optimization is run. In this experiment we optimize the new cuboid together with previously added ones, with shared orientation, floor/ceiling height and wall locations. Then, we skip over subsequent frames until the camera has moved outside the last cuboid at which point we start accumulating new frames and repeat the process. A new cuboid is accepted only if it does not overlap (IoU > 0.01) with existing ones. Results are presented in Fig. 8.

6. Conclusion

We have presented PixCuboid which formulates the room layout estimation task as an optimization problem by aligning deep features across images. As the network learns relatively low-level features (useful for pixel-wise direct alignment), our method is able to generalize to new datasets without retraining, in contrast to competing methods. The flexibility of the optimization-based approach allow us to include an arbitrary number of images. Further, other geometric constraints can easily be integrated, e.g. optimizing multiple cuboids with the same orientation or shared room height. We have focused on indoor room estimation, but we believe the method can be generalized to outdoor images, i.e. fitting cuboids or other geometric primitives to buildings.

Acknowledgments The work was supported by ELLIIT, the Swedish Research Council (Grant No. 2023-05424), and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. Compute was provided by the supercomputing resource Berzelius provided by National Supercomputer Centre at Linköping University and the Knut and Alice Wallenberg foundation.

References

[1] Samir Agarwala, Linyi Jin, Chris Rockwell, and David F Fouhey. PlaneFormers: From Sparse View Planes to 3D Reconstruction. In *European Conference on Computer Vision (ECCV)*, 2022. 3

[2] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2D-3D-Semantic Data for Indoor Scene Understanding. *arXiv preprint arXiv:1702.01105*, 2017. 1, 5

[3] Hernan Badino, Daniel Huber, and Takeo Kanade. The CMU Visual Localization Data Set. http://3dvis.ri.cmu.edu/data-sets/localization, 2011. 6

[4] Leonard Bruns, Jun Zhang, and Patric Jensfelt. Neural Graph Mapping for Dense SLAM with Efficient Loop Closure. *arXiv preprint arXiv:2405.03633*, 2024. 8

[5] Jiacheng Chen, Chen Liu, Jiaye Wu, and Yasutaka Furukawa. Floor-SP: Inverse CAD for Floorplans by Sequential Room-wise Shortest Path. In *International Conference on Computer Vision (ICCV)*, 2019. 3

[6] Steve Cruz, Will Hutchcroft, Yuguang Li, Naji Khosravan, Ivaylo Boyadzhiev, and Sing Bing Kang. Zillow Indoor Dataset: Annotated Floor Plans With 360° Panoramas and 3D Room Layouts. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 6

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009. 11

[8] Yuan Dong, Chuan Fang, Liefeng Bo, Zilong Dong, and Ping Tan. PanoContext-Former: Panoramic Total Scene Understanding with a Transformer. In *Computer Vision and Pattern Recognition (CVPR)*, 2024. 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 4, 11

[10] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the Spatial Layout of Cluttered Rooms. In *International Conference on Computer Vision (ICCV)*, 2009. 2

[11] Zhihua Hu, Bo Duan, Yanfeng Zhang, Mingwei Sun, and Jingwei Huang. MVLayoutNet: 3D Layout Reconstruction with Multi-view Panoramas. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 1289–1298, 2022. 2, 6

[12] Yuzhong Huang, Chen Liu, Ji Hou, Ke Huo, Shiyu Dong, and Fred Morstatter. UniPlane: Unified Plane Detection and Reconstruction from Posed Monocular Videos. *arXiv preprint arXiv:2407.03594*, 2024. 3

[13] Linyi Jin, Shengyi Qian, Andrew Owens, and David F Fouhey. Planar Surface Reconstruction from Sparse Views. In *International Conference on Computer Vision (ICCV)*, 2021. 3

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 11

[15] Chen-Yu Lee, Vijay Badrinarayanan, Tomasz Malisiewicz, and Andrew Rabinovich. RoomNet: End-to-End Room Layout Estimation. In *International Conference on Computer Vision (ICCV)*, 2017. 2

[16] David C Lee, Martial Hebert, and Takeo Kanade. Geometric Reasoning for Single Image Structure Recovery. In *Computer Vision and Pattern Recognition (CVPR)*, 2009. 2

[17] Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168, 1944. 4

[18] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In *International Conference on Computer Vision (ICCV)*, 2021. 2, 4

[19] Chen Liu, Jiaye Wu, and Yasutaka Furukawa. FloorNet: A Unified Framework for Floorplan Reconstruction from 3D Scans. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[20] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. PlaneRCNN: 3D Plane Detection and Reconstruction from a Single Image. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[21] Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989. 5

[22] Arun Mallya and Svetlana Lazebnik. Learning Informative Edge Maps for Indoor Scene Layout Prediction. In *International Conference on Computer Vision (ICCV)*, 2015. 2

[23] Donald W Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441, 1963. 4

[24] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. Total3DUnderstanding: Joint Layout, Object Pose and Mesh Reconstruction for Indoor Scenes from a Single Image. In *Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 5, 6

[25] Rémi Pautrat, Daniel Barath, Viktor Larsson, Martin R Oswald, and Marc Pollefeys. DeepLSD: Line Segment Detection and Refinement with Deep Image Gradients. In *Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 4

[26] Giovanni Pintore, Fabio Ganovelli, Ruggero Pintus, Roberto Scopigno, and Enrico Gobbetti. 3d floor plan recovery from overlapping spherical images. *Computational visual media*, 4:367–383, 2018. 2, 6

[27] Giovanni Pintore, Claudio Mura, Fabio Ganovelli, Lizeth Fuentes-Perez, Renato Pajarola, and Enrico Gobbetti. State-of-the-art in Automatic 3D Reconstruction of Structured Indoor Environments. In *Computer Graphics Forum*, pages 667–699. Wiley Online Library, 2020. 3

[28] Giovanni Pintore, Eva Almansa, Marco Agus, and Enrico Gobbetti. Deep3DLayout: 3D Reconstruction of an Indoor Layout from a Spherical Panoramic Image. *ACM Transactions on Graphics (TOG)*, 40(6):1–12, 2021. 2, 5, 6

[29] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-assisted Intervention (MICCAI)*, 2015. 4

[30] Paul-Edouard Sarlin, Ajaykumar Unagar, Mans Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, et al. Back to the Feature: Learning Robust Camera Localization from

Pixels to Pose. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2, 3, 4, 11

[31] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[32] Chang Shu, Kun Yu, Zhixiang Duan, and Kuiyuan Yang. Feature-metric Loss for Self-supervised Learning of Depth and Egomotion. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[33] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. SUN RGB-D: A RGB-D Scene Understanding Benchmark Suite. In *Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 6

[34] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 8

[35] Jheng-Wei Su, Chi-Han Peng, Peter Wonka, and Hung-Kuo Chu. GPR-Net: Multi-view Layout Estimation via a Geometry-aware Panorama Registration Network. In *Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 6

[36] Cheng Sun, Chi-Wei Hsiao, Min Sun, and Hwann-Tzong Chen. HorizonNet: Learning Room Layout with 1D Representation and Pano Stretch Data Augmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[37] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3D: Dataset and Methods for Single-Image 3D Shape Modeling. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[38] Jean-Philippe Tardif. Non-Iterative Approach for Fast and Accurate Vanishing Point Detection. In *International Conference on Computer Vision (ICCV)*, 2009. 4

[39] Fu-En Wang, Yu-Hsuan Yeh, Min Sun, Wei-Chen Chiu, and Yi-Hsuan Tsai. LED²-Net: Monocular 360° Layout Estimation via Differentiable Depth Rendering. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 5, 6

[40] Haiyan Wang, Will Hutchcroft, Yuguang Li, Zhiqiang Wan, Ivaylo Boyadzhiev, Yingli Tian, and Sing Bing Kang. PSMNet: Position-aware Stereo Merging Network for Room Layout Estimation. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 5, 6

[41] Jamie Watson, Filippo Aleotti, Mohamed Sayed, Zawar Qureshi, Oisin Mac Aodha, Gabriel Brostow, Michael Firman, and Sara Vicente. AirPlanes: Accurate Plane Estimation via 3D-Consistent Embeddings. In *Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[42] Jianxiong Xiao and Yasutaka Furukawa. Reconstructing the World's Museums. *International Journal of Computer Vision (IJCV)*, 110:243–258, 2014. 2

[43] Yiming Xie, Matheus Gadelha, Fengting Yang, Xiaowei Zhou, and Huaizu Jiang. PlanarRecon: Real-time 3D Plane Detection and Reconstruction from Posed Monocular Videos. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[44] Binbin Xu, Andrew J Davison, and Stefan Leutenegger. Deep Probabilistic Feature-metric Tracking. *IEEE Robotics and Automation Letters (RA-L)*, 6(1):223–230, 2020. 2

[45] Shang-Ta Yang, Fu-En Wang, Chi-Han Peng, Peter Wonka, Min Sun, and Hung-Kuo Chu. DuLa-Net: A Dual-Projection Network for Estimating Room Layouts from a Single RGB Panorama. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

[46] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A High-Fidelity Dataset of 3D Indoor Scenes. In *International Conference on Computer Vision (ICCV)*, 2023. 1, 5, 6

[47] Zehao Yu, Jia Zheng, Dongze Lian, Zihan Zhou, and Shenghua Gao. Single-Image Piece-wise Planar 3D Reconstruction via Associative Embedding. In *Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[48] Yuanwen Yue, Theodora Kontogianni, Konrad Schindler, and Francis Engelmann. Connecting the Dots: Floorplan Reconstruction Using Two-Level Queries. In *Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

[49] Cheng Zhang, Zhaopeng Cui, Yinda Zhang, Bing Zeng, Marc Pollefeys, and Shuaicheng Liu. Holistic 3D Scene Understanding from a Single Image with Implicit Representation. In *Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 5, 6

[50] Chuhang Zou, Alex Colburn, Qi Shan, and Derek Hoiem. LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[51] Chuhang Zou, Jheng-Wei Su, Chi-Han Peng, Alex Colburn, Qi Shan, Peter Wonka, Hung-Kuo Chu, and Derek Hoiem. Manhattan Room Layout Reconstruction from a Single 360° Image: A Comparative Study of State-of-the-Art Methods. *International Journal of Computer Vision (IJCV)*, 129:1410–1431, 2021. 6

A. Training Details

In this section we provide a more thorough description of how PixCuboid's neural network is trained. We employ a two-stage training procedure where first the edge maps $\mathbf{E}_i$ are pre-trained with a weighted MSE loss (summed over all scale levels), using line renderings of the ground truth cuboids as target images. Pixels that lie on the cuboid edges are up-weighted by a factor of five. The input images are resized to 512 pixels in height while maintaining the aspect ratio. The ResNet-101 [9] encoder is initialized with weights trained on ImageNet-1K [7] and is not frozen during this first training stage. In the first training stage a batch size of 10 is used and 150 examples are randomly sampled from each training scene at every epoch.

Next, the full network is trained with the loss in Eq. (7), applied at each scale and summed. We define a success threshold for the loss at each scale level (48, 12 and 3 pixels, respectively) and if the optimization is not successful the loss on the subsequent level is zeroed out (by setting $\gamma_m = 0$ or $\gamma_f = 0$, otherwise $\gamma_m = \gamma_f = 1$), to prevent learning from examples that are too difficult. Random horizontal cropping is performed after resizing, resulting in images of size 512x512. The 2D-3D correspondences $\{(\boldsymbol{x}_{ik}^{GT}, \boldsymbol{X}_{ik}^{GT})\}$ are found by taking the vertices of the semantic mesh labeled "floor", "wall" or "ceiling" that are visible in each particular view. From these we select 256 points randomly during training to compute the loss. As described in Sec. 3.3 we use guided point sampling for the image points $\{\boldsymbol{x}_{ik}\}$. 256 points are sampled, individually on each scale level with $\gamma = 4$. We set $\beta = 0$ during training since the vanishing point cost does not include any learned component, and let $\alpha = 0.1$. 40 points are sampled on each cuboid edge to compute the edge cost. We run three iterations of LM optimization on each scale level. Cuboids are initialized from the ground truth by applying a random rotation in the $[0°, 15°]$ range, followed by translation of up to 0.5 m in each direction and a resizing of the sides between $[-1, 1.5]$ m. The learned damping parameters of the LM optimization are handled like in [30]. Feature maps $\mathbf{F}_i$ have dimension 128, 128 and 32 on the coarse, medium and fine scale levels, respectively. We use a batch size of 4 and sample 50 examples per training scene at each epoch in the second training stage. The network weights (30 million parameters) are saved at each epoch and we pick the ones that minimize the warp loss Eq. (7) on the validation set.

In both stages the Adam [14] optimizer is used to train the network, with a learning rate of 5×10^{-6}. Gradients are clipped to the $[-1, 1]$ range. The training is run for 10 epochs. The pre-training takes 13 h and the training of the full network finishes in 27 h, using a NVIDIA TITAN V GPU.

B. Results

In Fig. 9 we display additional examples of predicted room layouts in 2D-3D-Semantics. Figs. 10 and 11 present extra failure cases and cuboid initializations, respectively. Table Tab. 4 contains the full set of results for the ablation study in Sec. 5.2.

We also visualize the feature-, edge- and confidence maps (on the finest scale level) for the first five images of an image tuple in our ScanNet++ v2 test in Fig. 12. PixCuboid learns features that are consistent between views (second column) and ignores clutter by assigning high confidence to image points that lie on the floor, walls or ceiling (third column). It can often predict the cuboid edges accurately despite the presence of occluding objects (fourth column).

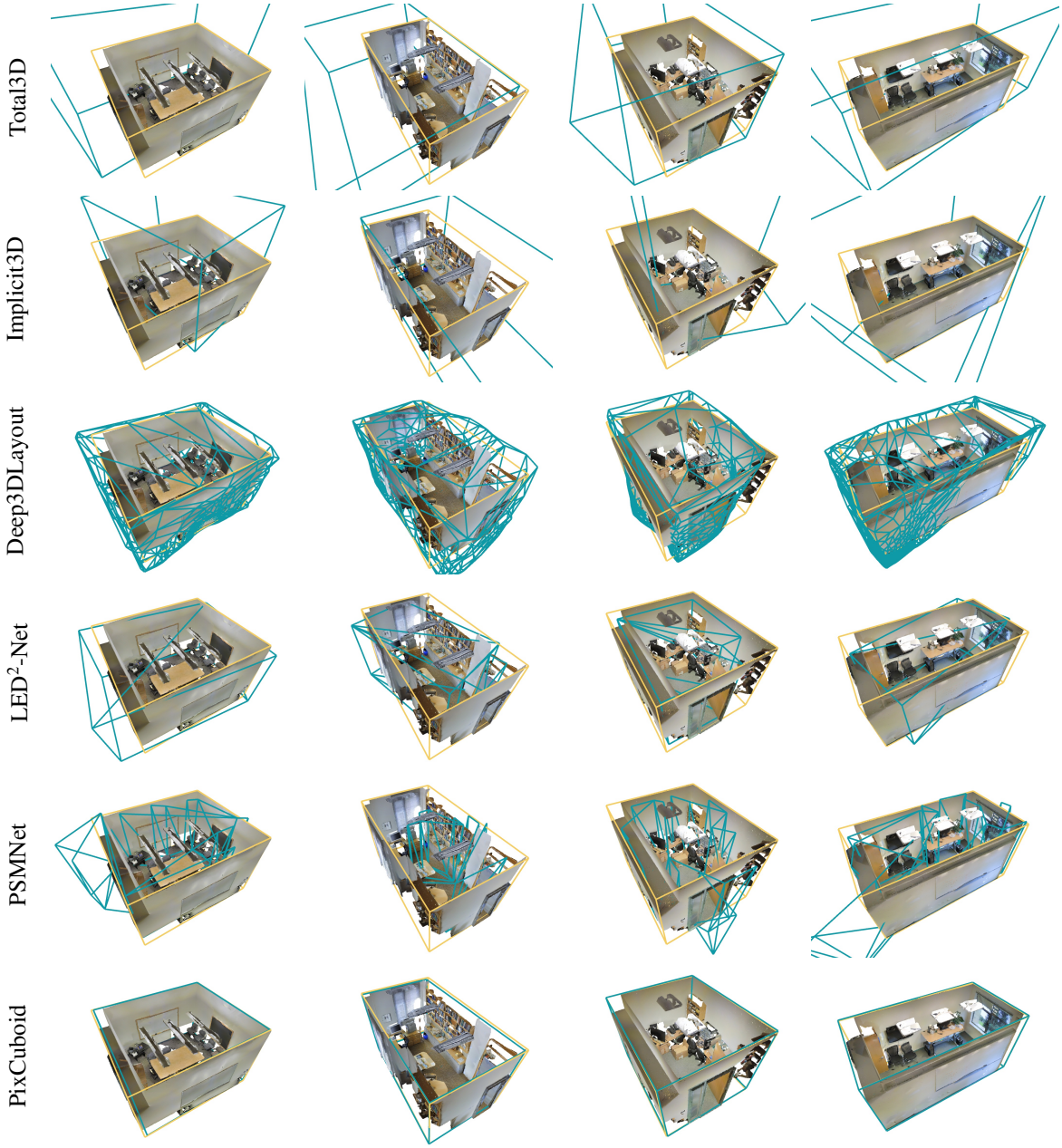

Figure 9. Qualitative comparisons of predicted room layouts for spaces in 2D-3D-Semantics. Predictions are shown in blue and the ground truth cuboids in yellow. None of the methods are trained on this dataset.

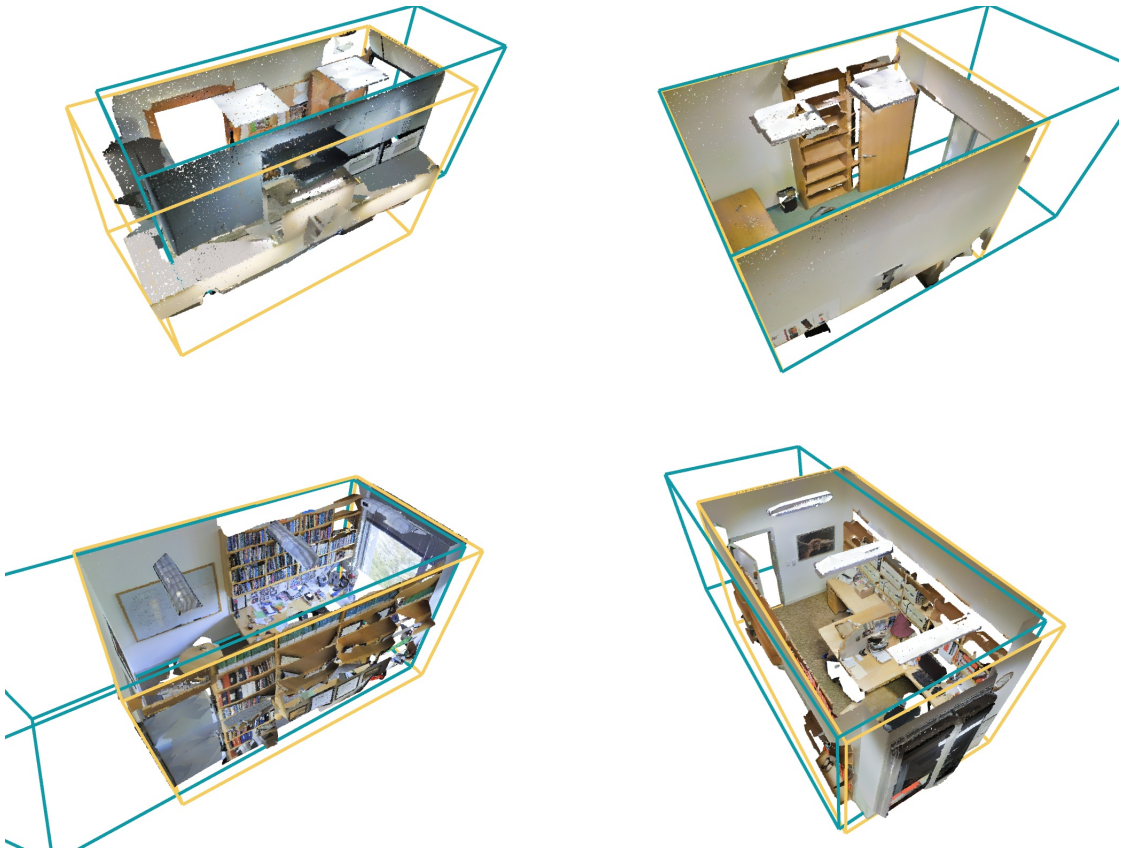

Figure 10. Failure cases of PixCuboid on 2D-3D-Semantics. Predicted room layouts are shown in **blue** and ground truth cuboids in **yellow**.

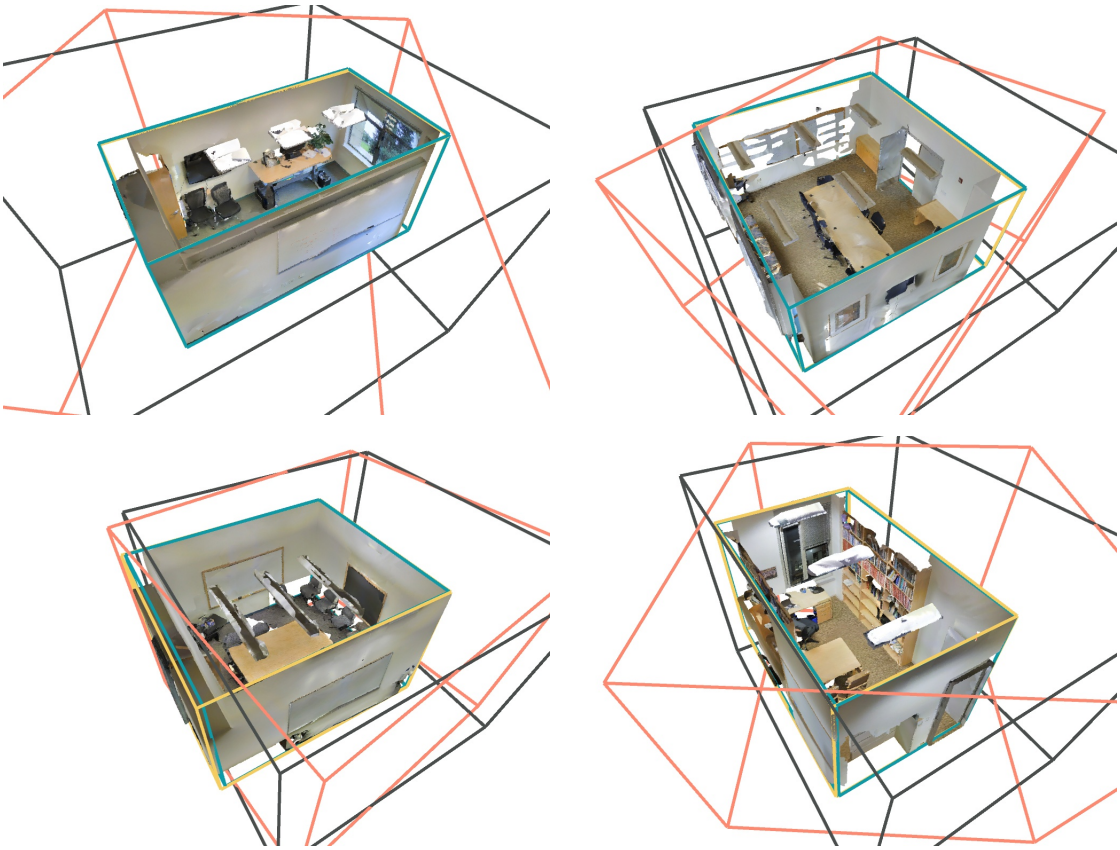

Figure 11. Cuboid initialization examples on 2D-3D-Semantics. Even from poor initial estimates (**red**), PixCuboid can align the cuboids with the help of vanishing points (**gray**) and is able to converge to accurate layouts (**blue**). The ground truth is shown in **yellow**.

		IoU ↑	Chamfer ↓	Rot. ↓	Depth ↓	Normal ↑	Success ↑
Feat.	RGB	24.2	2.51 m	36.4°	1.02 m	7.6	1.4%
	PixLoc (CMU)	25.3	2.30 m	33.4°	1.02 m	10.9	1.1%
	PixCuboid (E_{feat} only)	**35.2**	**1.77 m**	**23.0°**	**0.75 m**	**30.8**	**5.8%**
Cost	E_{feat}	35.2	1.77 m	23.0°	0.75 m	30.8	5.8%
	E_{edge}	76.1	0.51 m	4.7°	0.23 m	86.5	43.6%
	E_{feat} + E_{edge}	81.9	0.39 m	3.8°	0.17 m	88.9	61.4%
	E_{feat} + E_{VP}	44.6	1.24 m	1.5°	0.57 m	79.4	21.7%
	E_{edge} + E_{VP}	83.1	0.29 m	**1.3°**	0.13 m	95.5	51.9%
	E_{feat} + E_{edge} + E_{VP}	**87.2**	**0.22 m**	**1.3°**	**0.09 m**	**96.1**	**67.2%**
Sampl.	Random	86.9	0.23 m	1.4°	0.10 m	95.8	65.8%
	Floor/wall/ceiling	86.6	0.25 m	1.4°	0.10 m	95.7	66.4%
	Guided	**87.2**	**0.22 m**	**1.3°**	**0.09 m**	**96.1**	**67.2%**
Res.	Low (256 px)	84.2	0.28 m	**1.3°**	0.12 m	95.6	63.3%
	Medium (512 px)	**87.2**	**0.22 m**	**1.3°**	**0.09 m**	**96.1**	**67.2%**
	High (768 px)	85.7	0.26 m	**1.3°**	0.12 m	95.6	66.1%
Init.	Random	60.8	1.28 m	18.0°	0.52 m	60.4	40.0%
	Random + VP	72.2	0.79 m	10.9°	0.32 m	74.6	51.9%
	Y down	84.9	0.27 m	2.1°	0.12 m	93.6	64.7%
	Y down + VP	**87.2**	**0.22 m**	**1.3°**	**0.09 m**	**96.1**	**67.2%**
Scales	Coarse	81.0	0.32 m	1.4°	0.15 m	94.6	36.7%
	Coarse + medium	86.5	0.23 m	1.4°	0.10 m	95.8	65.6%
	Coarse + medium + fine	**87.2**	**0.22 m**	**1.3°**	**0.09 m**	**96.1**	**67.2%**

Table 4. Ablation experiments on our ScanNet++ v2 test set.

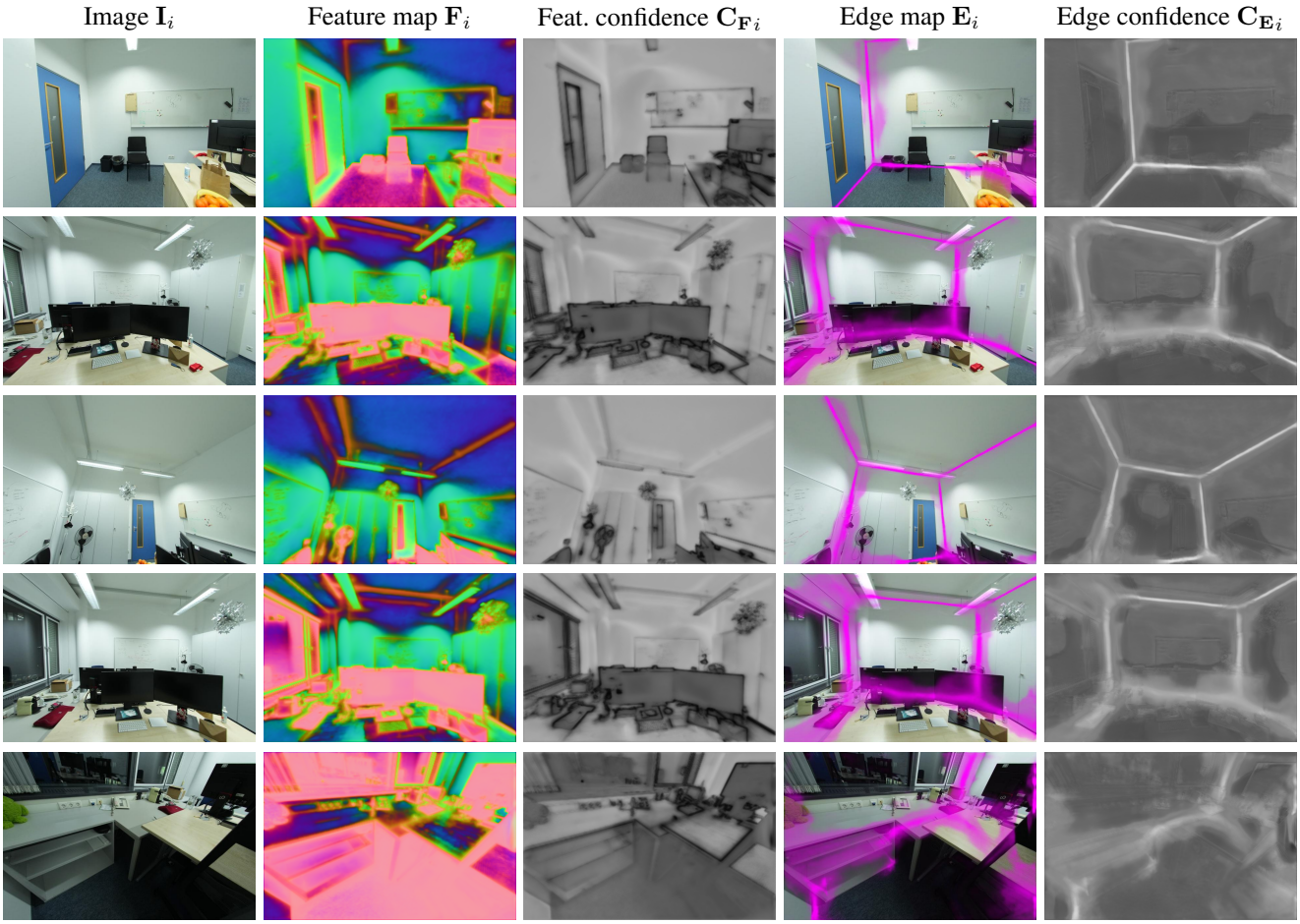

Figure 12. Example feature- and edge maps with corresponding confidence maps for one image tuple in our ScanNet++ v2 test set. Results are shown only on the finest level. Feature maps have been mapped to RGB using PCA. Confidence ranges from low (black) to high (white).