PolyLayout: Multi-room Manhattan Layout Estimation

Gustav Hanning[1], Shaohui Liu[2], Rémi Pautrat[3], Marc Pollefeys[2,3], Kalle Åström[1], and Viktor Larsson[1]

[1]Lund University [2]ETH Zurich [3]Microsoft Spatial AI Lab

Abstract. Estimating room layouts from multi-view imagery is a core task for indoor scene understanding. Existing methods are typically limited either by poor generalization to new datasets or restrictive geometric assumptions of the room shape or camera configuration. Most also estimate rooms independently, failing to exploit shared building structure such as dominant directions, ground plane or ceiling height.
We propose PolyLayout, a multi-room layout estimation method that parameterizes room layouts as Manhattan 3D polygons and optimizes them jointly across multiple rooms. The optimization objective is predicted by a neural network on top of robust pre-trained visual features and trained end-to-end with supervision only on output room layouts. At the same time, camera projection and polygon updates remain explicit and model-based. This separation between learned scoring and geometry improves generalization to new datasets and camera parameters. During optimization, PolyLayout adaptively refines the polygon topology through iterative wall split and merge operations while jointly utilizing structural cues across rooms. We introduce two new multi-view multi-room layout benchmarks by providing layout annotations to existing datasets, and experiments show that PolyLayout outperforms prior approaches, both in terms of accuracy and robustness.
Project page: `https://ghanning.github.io/PolyLayout`

1 Introduction

Estimating the 3D layout of indoor environments is a core problem in computer vision, with applications in robotics, augmented reality and holistic scene understanding. In particular, the goal of *room layout estimation* is to predict the location of the walls, floor and ceiling. Many existing methods aim to estimate the layout from a single image, which makes reconstruction difficult and ambiguous. Recently, Plane-DUSt3R [17] and PixCuboid [12] leverage multiple perspective views for the task. Plane-DUSt3R retrains the foundation model DUSt3R [43] to predict the structural planes of the scene, but the method generalizes poorly. PixCuboid is an optimization-based approach to layout estimation centered around featuremetric alignment. However, it is restricted to single cuboid-shaped rooms which significantly limits the practicality of the method.

In this paper we present a novel multi-view room layout estimation method, dubbed PolyLayout, which shows strong performance across multiple datasets

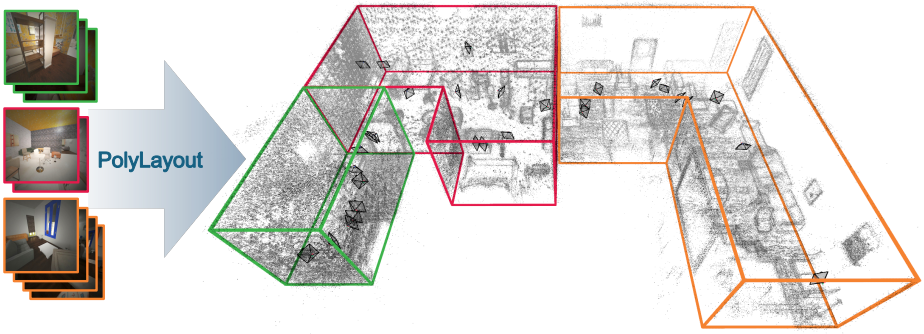

Fig. 1: Our room layout estimation method PolyLayout predicts the layout of multiple rooms simultaneously from posed images. By sharing the orientation and floor/ceiling height it achieves higher accuracy than comparable single room methods.

and uses a more general Manhattan world [5] assumption. Like PixCuboid, our method takes a set of posed perspective images as input and progressively refines an initial room layout by minimizing a learned optimization objective. Instead of cuboids, we represent layouts as 3D polygons where planes meet at right angles, which makes our method applicable to a larger class of rooms. For scenes consisting of multiple rooms, we further propose to optimize these together, sharing the same orientation and optionally the floor and ceiling height. By sharing parameters in the optimization, our method benefits from enhanced convergence and higher accuracy. We also suggest a new network architecture compared to PixCuboid, with a DINOv2 [25] encoder, and train it end-to-end through the optimization. PolyLayout is evaluated on both synthetic and real data and we validate the design in a number of ablation experiments. The main contributions of this work are:

- We present a multi-view room layout estimation method with strong generalization capabilities, where rooms are represented by Manhattan polygons.
- We propose to jointly optimize the orientation and floor/ceiling height in multi-room scenes, improving the accuracy of the predicted layouts.
- We introduce an adaptive procedure for updating the layout topology during optimization and quantify its effect in ablations.
- We create two new datasets, based on Aria Synthetic Environments [3] and ScanNet++ v2 [45] respectively, to benchmark multi-view multi-room layout estimation and show that PolyLayout outperforms competing methods.

2 Related Work

Image-based room layout estimation is the task of reconstructing the floor, ceiling and walls given one or more views of the indoor environment. Many methods [14, 18, 24, 34, 35, 47] have been proposed to estimate the layout from

a single perspective image. However, the unknown scale and limited field-of-view make this a very challenging problem. Other works [28, 38, 41, 49] utilize a 360° panorama which gives a more complete coverage of the surroundings but still suffer from scale ambiguity.

Layout estimation from multiple views is a less studied area of research. Flint *et al*. [9] reconstruct indoor scenes from video sequences by combining geometric and photometric cues. PSMNet [42] and GPR-Net [37] use a pair of panoramas to predict the room layout with transformer-based networks. MVLayoutNet [15] and Pintore *et al*. [29] take as input multiple panoramic views and can reconstruct multi-room scenes. Recently, two methods have been suggested for estimating room layouts from multiple perspective images. Huang *et al*. [17] present Plane-DUSt3R which finds the wall, ceiling and floor planes of an indoor scene from unposed views by leveraging the foundation model DUSt3R [43]. PixCuboid [12] is an optimization-based method that also uses multiple perspective images but assumes known camera poses and is restricted to cuboid-shaped rooms. It is the most similar method to ours. In contrast to PixCuboid we support more complex room layouts where the number of walls is determined automatically at inference time. Our method can also share parameters between rooms for increased accuracy.

Layout estimation from point clouds. Monte Carlo Scene Search [11] and SceneCAD [2] are examples of methods that reconstruct the layout of a scene from RGB-D scans. SceneScript [3] takes a point cloud obtained from a visual-inertial SLAM system as input and models the layout as a sequence of structured language commands. The closely related SpatialLM [22] fine-tunes an open-source LLM for the same task. Our method PolyLayout does not require point clouds, instead fitting 3D layout polygons to the images directly.

Floor plan reconstruction is a closely related problem which MonteFloor [36] and RoomFormer [46] solve by extracting 2D polygons from a density map created from a point cloud. BADGR [20] reconstructs the floor plan and simultaneously refines the camera poses of input panorama images using diffusion. Our method is richer as it outputs a set of 3D polygons - but they can easily be converted into a 2D floor plan.

Room layout assumptions. Most room layout estimation methods make prior assumptions about the room geometry. Several works [12, 14, 24, 47] assume cuboid rooms to simplify the estimation task but are thus unable to reconstruct many real-world building environments. The Manhattan world [5], where planes are aligned with three principal axes, is a more general model employed by numerous layout estimation methods [38, 41, 49]. By adding a single-floor, single-ceiling restriction we get the indoor world model [18]. The Atlanta world [31] differs from the Manhattan world in that the walls are vertical but not necessarily orthogonal and is used for example by AtlantaNet [27]. In this paper we utilize the Manhattan world model: room layouts are represented as polygons having edges aligned with the local x and y axes. The Manhattan frame is estimated during optimization. Additionally, our flexible approach allows for sharing the floor and ceiling height in multi-room scenes, resulting in the indoor world model.

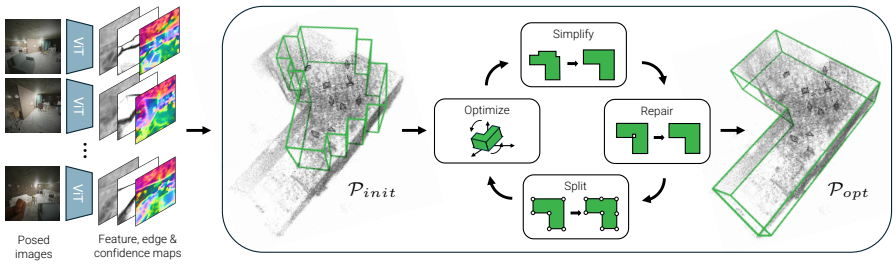

Fig. 2: Room layout optimization with PolyLayout. Given posed images, our room layout estimation method PolyLayout refines an initial room layout polygon $\mathcal{P}_{init}$ through successive Levenberg-Marquardt optimization steps, resulting in the final layout $\mathcal{P}_{opt}$. First, a neural network extracts deep features, confidence and edge maps for each input image. The layout is initialized from the known camera poses, and is then optimized by iteratively minimizing the cost function $E(\mathcal{P})$ (Sec. 3.2). After each LM step the polygon is simplified and any self-intersections removed, followed by splitting of the polygon edges (walls). The point cloud is included only for visualization purposes and is not an input to our method.

Featuremetric alignment. Direct alignment of deep features has been used in a number of areas such as point cloud registration [16], camera localization [30] and SfM [21]. When trained end-to-end, the learned features offer improved robustness and accuracy. More recently, Hanning *et al.* [12] incorporate a featuremetric cost in the refinement of cuboid room layouts. We take inspiration from their work but utilize a vision transformer [6] (ViT) model for better features.

3 Method

Our method takes as input a set of images $\{\mathbf{I}_i\}_{i=1}^n$ with known poses $(\boldsymbol{R}_i, \boldsymbol{t}_i)$ and intrinsics $\boldsymbol{K}_i$. The output is a Manhattan polygon $\mathcal{P}$ where walls meet at right angles (Sec. 3.1), representing the room geometry. It is initialized from the camera poses (Sec. 3.4) and then refined through a coarse-to-fine optimization (Sec. 3.2). During optimization walls can dynamically be removed by simplifying the polygon or added by splitting existing walls (Sec. 3.5). The method naturally extends to the multi-room setting where parameters are shared across rooms to improve accuracy (Sec. 3.6). We use a network that consists of a ViT encoder and two convolutional decoders to predict a dense feature map $\mathbf{F} \in \mathbb{R}^{W \times H \times D}$, an edge map $\mathbf{E} \in \mathbb{R}^{W \times H}$, as well as two confidence maps $\mathbf{C_F}, \mathbf{C_E} \in \mathbb{R}^{W \times H}$ for each image $\mathbf{I} \in \mathbb{R}^{W \times H \times 3}$ (Sec. 3.3). An overview is displayed in Fig. 2.

3.1 Room Layout Parameterization

The room layout is parameterized by the orientation $\boldsymbol{R} \in SO(3)$ and a plane offset vector $\boldsymbol{d} = \begin{bmatrix} d_1 & d_2 & \cdots & d_p \end{bmatrix} \in \mathbb{R}^p$, where $p \geq 6$. The matrix $\boldsymbol{R}$ is the global-to-local rotation and $\boldsymbol{d}$ defines the floor, ceiling and wall planes. In the local frame,

the floor and ceiling are given by $z = d_1$ and $z = d_2$, respectively. The remaining plane offsets specify the walls of the room in order, alternating between x and y: the first wall is thus $x = d_3$, the second $y = d_4$, and so on. We show an example of this representation in Fig. 3 (left). There is an even number of wall planes, at least four. In the case of four planes, the shape is simply a cuboid.

3.2 Optimization of Room Layouts

Our method is based on optimizing an initial layout through successive Levenberg-Marquardt [19, 23] steps in a coarse-to-fine manner, with the cost function

$$E(\mathcal{P}) = E_{feat}(\mathcal{P}) + \alpha E_{edge}(\mathcal{P}) + \beta E_{VP}(\mathcal{P}) + \gamma E_{per}(\mathcal{P}). \tag{1}$$

We adapt the three cost functions used in [12] to the polygon case: E_{feat} which measures multi-view consistency of warped deep image features, E_{edge} that aligns the layout edges with predicted edge maps, and E_{VP} which compares the three vanishing points defined by the layout to detected line segments. More specifically, the featuremetric cost

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

where $\{\boldsymbol{x}_{ik}\}$ are image points sampled in image $\mathbf{I}_i \in \mathbb{R}^{W\times H\times 3}$ and $\mathcal{W}_{i\to j}$ denotes the warping to image $\mathbf{I}_j$ via the polygon $\mathcal{P}$. $\mathbf{F}_i, \mathbf{F}_j \in \mathbb{R}^{W\times H\times D}$ are dense feature maps extracted by a neural network and $[\cdot]$ represents lookup with sub-pixel interpolation. ρ is a robust loss function and

$$w_{ijk} = \mathbf{C}_{\mathbf{F}i}[\boldsymbol{x}_{ik}] \mathbf{C}_{\mathbf{F}j}[\mathcal{W}_{i\to j}(\boldsymbol{x}_{ik}, \mathcal{P})] \tag{3}$$

are per-point weights interpolated from confidence maps $\mathbf{C}_{\mathbf{F}i}, \mathbf{C}_{\mathbf{F}j} \in \mathbb{R}^{W\times H}$. The points $\{\boldsymbol{x}_{ik}\}$ are sampled from the probability map $\mathbf{C}_{\mathbf{F}i}^{\kappa}$, $\kappa \in \mathbb{R}$, without replacement. In contrast to the cuboid case it is possible to have self-occlusions with polygons, which need to be taken care of when warping between images.

The edge cost E_{edge} is computed by sampling 3D points $\{\boldsymbol{X}_k\}$ along the edges of the polygon and projecting into each view:

$$E_{edge}(\mathcal{P}) = \sum_{i} \sum_{k} w_{ik} \mathbf{E}_i[\Pi_i(\boldsymbol{R}_i \boldsymbol{X}_k + \boldsymbol{t}_i)]^2. \tag{4}$$

Here, $\Pi_i : \mathbb{R}^3 \to \mathbb{R}^2$ denotes the projection using the known intrinsics $\boldsymbol{K}_i$ and $\mathbf{E}_i \in \mathbb{R}^{W\times H}$ is the predicted edge map. As with the featuremetric cost we use a confidence image $\mathbf{C}_{\mathbf{E}i}$ from which the weights w_{ik} are interpolated.

Thanks to our Manhattan assumption the polygon $\mathcal{P}$ has only three vanishing points (VPs), given for each view by the columns of $\boldsymbol{R}_i \boldsymbol{R}^T = \left[\boldsymbol{v_{i,1}}\ \boldsymbol{v_{i,2}}\ \boldsymbol{v_{i,3}}\right]$. The vanishing point cost E_{VP} measures the consistency [39] of these VPs and line segments $\{\boldsymbol{l}_k^i\}$ detected in the images $\mathbf{I}_i$:

$$E_{VP}(\mathcal{P}) = \sum_{i} \sum_{k} \min \left(\min_{m\in\{1,2,3\}} D_{VP}(\boldsymbol{l}_k^i, \boldsymbol{v}_{i,m}), \tau \right)^2. \tag{5}$$

Line segments are softly assigned to only one VP by taking the minimum distance

$$D_{VP}(\boldsymbol{l}, \boldsymbol{v}) = |\hat{\boldsymbol{l}}^T \boldsymbol{l}_1| / \sqrt{\hat{l}_1^2 + \hat{l}_2^2} \tag{6}$$

between one of the segment end points $\boldsymbol{l}_1$ and the line $\hat{\boldsymbol{l}} = \bar{\boldsymbol{l}} \times \boldsymbol{v} = \begin{bmatrix} \hat{l}_1 & \hat{l}_2 & \hat{l}_3 \end{bmatrix}$ passing through its midpoint $\bar{\boldsymbol{l}}$ and the vanishing point $\boldsymbol{v}$. The distance is capped at τ to handle line segments that do not align with any VP.

In this work we additionally introduce a perimeter cost E_{per} to penalize complex polygon layouts $\mathcal{P}$:

$$E_{per}(\mathcal{P}) = |d_3 - d_{p-1}| + |d_4 - d_p| + \sum_{i=4}^{p-1} |d_{i+1} - d_{i-1}|. \tag{7}$$

In cases where only a subset of walls are visible in the images, this cost induces a slight shrinking bias that prevents the unobserved regions from diverging.

Coarse-to-fine optimization: Our network outputs feature, confidence and edge maps on three different scale levels, with gradually increasing resolution (1/16, 1/4 and 1/1 of the input image size). We start the optimization at the coarsest level and initialize subsequent scales with the output from the previous.

3.3 Learning to Optimize Room Layouts

The network is trained end-to-end, supervised on the optimized room layouts at each scale. We use a set of ground truth 2D-3D correspondences $\{(\boldsymbol{x}_{ik}^{GT}, \boldsymbol{X}_{ik}^{GT})\}$ with points on the walls, floor and ceiling to compute the loss

$$\mathcal{L}(\mathcal{P}) = \frac{1}{N} \sum_{i,j} \sum_{k} \rho\left(\| \mathcal{W}_{i \to j}(\boldsymbol{x}_{ik}^{GT}, \mathcal{P}) - \Pi_j(\boldsymbol{R}_j \boldsymbol{X}_{ik}^{GT} + \boldsymbol{t}_j) \|^2 \right) \tag{8}$$

and propagate the gradients back through the unrolled optimization process. We apply the loss to the optimized layout at each scale, but only if the optimization succeeded at the previous level ($\mathcal{L}(\mathcal{P})$ below a threshold) to prevent oversmoothing of the fine feature maps. During training, we disable the VP and perimeter costs ($\beta = \gamma = 0$) as they contains no learned components. The polygons are initialized by randomly rotating and shifting the planes of the ground truth layouts for each room. As in previous works [12, 30], we learn per-parameter LM dampening factors along with the weights of the feature extractor network. Walls share a single dampening factor. Similar to [12], we also pre-train the edge maps with a weighted MSE loss, where the target images are line renderings of the ground truth polygon edges. Our network has a DINOv2 ViT encoder and two convolutional heads for the feature and edge maps. The architecture is detailed in the supplementary material.

3.4 Room Layout Initialization

During inference we initialize $\mathcal{P}$ from the known camera poses. The mean of the camera y axes gives an up vector (z), which together with two randomly

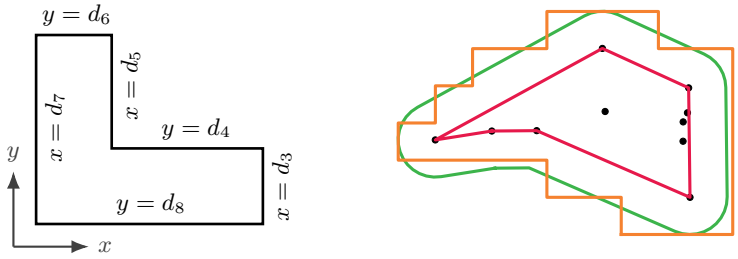

Fig. 3: Left: The walls of the room layout polygon $\mathcal{P}$ are defined by axis-aligned planes, alternating between x and y. **Right**: To initialize $\mathcal{P}$ from the positions of the cameras (black points) we first compute the concave hull (**red**), then buffer it (**green**) and convert it to a Manhattan polygon through rasterization (**orange**).

sampled basis vectors orthogonal to z form the rotation matrix $\boldsymbol{R}$. As in [12] we then take a few LM optimization steps minimizing $E_{VP}(\mathcal{P})$ to get a more accurate initial orientation. In the local xy plane defined by $\boldsymbol{R}$ an α-shape [7] is computed around the camera positions and expanded by a fixed distance δ. The shape is converted into a Manhattan polygon by rasterizing it and then tracing the outline of the resulting binary image. Floor and ceiling heights are set so that the minimum distance to the camera centers is δ. The proposed initialization strategy is visualized in Fig. 3 (right image).

3.5 Updating the Polygon Layout

After each optimization step at inference time, we first check if any camera center is outside the room layout, in which case the closest walls (or floor/ceiling) are moved outwards as needed. We then simplify the layout polygon using a variant of the Visvalingam-Whyatt algorithm [40]. In the original formulation the importance of a polygon vertex is given by the area spanned by itself and its two neighbors, and the vertex with minimum importance is iteratively removed. To maintain the Manhattan property of our polygon we adjust the algorithm to remove two adjacent walls at a time, with importance equal to the area of the rectangle spanned by the two walls. Only walls that have converged (small $|\Delta d_i|$) are eligible for removal. Any self-intersections that might have been introduced by the LM step or simplification are removed using the Shapely [10] library. After optimization on the coarse and medium scale levels, walls longer than a threshold are iteratively split, up to a specified maximum number of planes p_{max}.

3.6 Multi-room Optimization

The parameterization described in Sec. 3.1 allows for joint optimization of multiple rooms with shared orientation and floor/ceiling height. Alternative parameterizations, *e.g.* via rotation/translation/size $(\boldsymbol{R}, \boldsymbol{t}, s_x, s_y, s_z)$, entangle the parameters for each plane with the translation, making them harder to share

between different rooms. Sharing parameters across rooms greatly reduces the degrees of freedom and improves convergence, resulting in more accurate layout predictions. We argue that rooms within a building having the same orientation is extremely common, and show in ablation experiments (Sec. 5.2) how accuracy improves when our method exploits this fact. Note that wall locations can also be shared, but it is not explored in this work. We further assume prior knowledge of the image assignment for multi-room scenes, *i.e.* we know which images belong to which room.

4 Experimental Setup

We use the following parameters when evaluating PolyLayout: $\alpha = 0.05$, $\beta = 40$ and $\gamma = 5 \times 10^{-4}$ (on the finest scale, $\gamma = 0$). 256 points are sampled in each view to compute the featuremetric cost E_{feat} and ρ is the generalized loss function from [4]. For the edge cost E_{edge} we sample 40 points uniformly along every polygon edge. The VP distance threshold τ is set to 0.05 and we detect line segments with DeepLSD [26], using up to 100 segments per image for E_{VP}. We initialize the room layout polygon as described in Sec. 3.4, and run five iterations of VP optimization to align it with detected line segments. The number of LM steps per scale level is 15. After each optimization step the polygon is simplified using an importance threshold of 0.5. Unless otherwise stated the orientation $\boldsymbol{R}$ and floor/ceiling height (d_1, d_2) are shared between the rooms in multi-room scenes. The setup is more extensively described in our supplementary material.

4.1 Training

To train PolyLayout's neural network we follow the procedure given in [12], but swap their ResNet-101 [13] based CNN for a DINOv2 backbone (ViT-S/14) and two convolutional heads. We take their training set, 391 cuboid rooms from ScanNet++ v2, and extend it with 107 manually annotated Manhattan room layouts from the same dataset. More details and a comparison between the two models can be found in the supplementary material.

4.2 Datasets

Aria Synthetic Environments [3] (ASE) is a synthetic dataset containing 100,000 multi-room scenes with simulated camera trajectories and ground truth 3D floor plans. It has a mix of cuboid-shaped rooms and more general Manhattan layouts. All rooms within a scene have the same floor and ceiling height. We create new validation and test sets for multi-view room layout estimation by randomly selecting 100 scenes for each set. Our method is tuned on the validation set. From the ground truth, which is given as a list of walls, doors and windows, we derive the geometry of every room by connecting adjacent walls. Five sets of images, consisting of 10 images per room, are sampled for all scenes. We

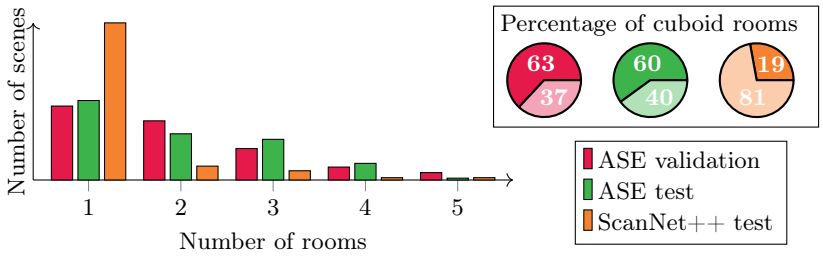

Fig. 4: Distribution of the number of rooms in our ASE and ScanNet++ datasets. The pie charts show the ratio of cuboid (dark color) versus non-cuboid rooms.

make sure that the images have sufficient coverage of the scene by employing a visibility-based sampling. Details are given in the supplementary material.

ScanNet++: For evaluation on real data we manually annotate the room layouts of 80 scenes from ScanNet++ v2 [45]. Both single- and multi-room scenes are included, with cuboid-shaped rooms as well as more general layouts that are not necessarily Manhattan or have shared ceiling height. The annotation is performed by fitting planes to mesh vertices selected by the user in a 3D graphical user interface. When planes have been fitted to the floor, ceiling and walls the annotation program automatically finds the intersections between the planes to create the ground truth room layout. For the annotation we only consider rooms that have a single horizontal ceiling and flat walls. A small number of additional rooms are excluded due to incomplete lidar scans. Overall, only a minor portion of the dataset is discarded. For each scene three sets of images are sampled, with 10 DSLR images per room. The image sampling process is described in our supplementary material. Note that these 80 scenes annotated by us are not part of the ScanNet++ dataset presented in [12], and do not overlap with our training data.

2D-3D-Semantics: We additionally make use of the 2D-3D-Semantics [1] dataset from [12], with 160 cuboid rooms. For every room there are two panoramas, split into four perspective images each. To support our multi-room setup the rooms are grouped together within each building.

The distribution of the number of rooms per scene and their type for ASE and ScanNet++ is shown in Fig. 4. We will make the image sets and ground truth layouts for these two datasets available together with code for evaluation.

4.3 Metrics

We adopt the 3D (IoU, Chamfer distance), pixel-wise (depth RMSE, normal angle recall at 10°) and mean prediction time metrics proposed in [12], but compute them per scene instead of per room for ASE and ScanNet++. In addition, we define wall and room recall as follows. For each wall in the ground truth layouts a grid of 3D points is sampled with 0.25 m spacing. The minimum distance between each point and the predicted room layout is then computed. If more

than 90% of the points are within 0.25 m the wall is considered to be successfully captured by the prediction. This "wall recall" metric is averaged over all ground truth walls. The "room recall" is the proportion of rooms with 100% wall recall.

5 Results

5.1 Room Layout Estimation

Baselines on ASE and ScanNet++: We compare PolyLayout with the two recent multi-view room layout estimation methods Plane-DUSt3R [17] and PixCuboid [12]. We further include SceneScript [3] as a representative point-based approach, as well as the floor plan reconstruction method RoomFormer [46]. Plane-DUSt3R and PixCuboid are run individually for each room while SceneScript, RoomFormer and PolyLayout are run on a per-scene basis. We use the authors' official implementations and pre-trained network weights.

SceneScript has three variants: using a point cloud, posed images, or both. Since only the point cloud version is publicly available, we use that. We generate point clouds via dense COLMAP [32,33] reconstruction with fixed camera poses. As SceneScript is sensitive to outliers, we filter the cloud to keep only points inside the ground truth room layouts or within 1 m of them. For reference, we also report SceneScript on the semi-dense ASE point clouds produced by SLAM using all scene images. Note also that the method was trained on ASE with these point clouds, and that our test set is a subset of the ASE training scenes.

For Plane-DUSt3R the metric-scale version is used. We tried fixing the camera poses but got very poor results, so we instead align the predicted poses to the ground truth using COLMAP's model aligner tool.

Both SceneScript and Plane-DUSt3R output walls which do not form a closed volume, and so we do not report the IoU for these methods. Additionally, the floor and ceiling are removed from the ground truth layout when computing the Chamfer distance for SceneScript and Plane-DUSt3R.

RoomFormer takes a density map as input, which we create by projecting the point cloud from the dense COLMAP reconstruction onto the ground plane. The output is a set of 2D polygons that are lifted into 3D using the ground truth floor and ceiling height. We use the checkpoint trained on Structured3D [48].

Baselines on 2D-3D-Semantics: The availability of panoramic images makes comparison with panorama-based methods Deep3DLayout [28], LED^2-Net [41] and PSMNet [42] possible. We further include results for Total3D-Understanding [24] and Implicit3DUnderstanding [47], which estimate the layout from a single perspective image. We do not compare with MVLayoutNet [15] or Pintore *et al*. [29] as there is no publicly available code. For 2D-3D-S, all methods except PolyLayout are run per room. When computing 3D and recall metrics for single-view methods we only consider the prediction with maximum IoU against the ground truth layout. All metrics are calculated per room for this dataset.

Certain methods (SceneScript, Plane-DUSt3R, RoomFormer) may predict empty layouts. In this case we consider the IoU and recall to be zero, but skip

Table 1: Room layout estimation on our Aria Synthetic Environments and ScanNet++ v2 test sets and the cuboid-shaped spaces of 2D-3D-Semantics.

	Method	3D		Recall		Pixel-wise		
		IoU ↑	Chamfer ↓	Wall ↑	Room ↑	Depth ↓	Normal ↑	Time ↓
	SceneScript† [3] ECCV24	N/A	0.12 m‡	96.1	91.0	0.07 m	98.6	5.31 s
ASE [3]	SceneScript [3] ECCV24	N/A	2.44 m‡	16.7	9.1	1.18 m	64.5	7.45 s
ASE [3]	Plane-DUSt3R [17] ICLR25	N/A	23.16 m‡	0.1	0.0	1.90 m	19.7	105.60 s
ASE [3]	PixCuboid [12] ICCVW25	68.1	0.93 m	54.9	45.5	0.57 m	90.0	1.01 s
ASE [3]	RoomFormer [46] CVPR23	13.8	2.37 m	5.0	0.5	1.75 m	67.8	**0.06 s**
ASE [3]	PolyLayout (ours)	**94.3**	**0.12 m**	**89.0**	**79.0**	**0.09 m**	**98.2**	5.48 s
ScanNet [45]	SceneScript [3] ECCV24	N/A	1.80 m‡	11.9	0.3	0.80 m	55.8	6.16 s
ScanNet [45]	Plane-DUSt3R [17] ICLR25	N/A	16.82 m‡	0.2	0.0	1.06 m	18.2	57.72 s
ScanNet [45]	PixCuboid [12] ICCVW25	78.8	0.36 m	58.9	26.1	0.26 m	87.8	0.87 s
ScanNet [45]	RoomFormer [46] CVPR23	10.3	1.61 m	3.4	0.0	0.98 m	48.6	**0.11 s**
ScanNet [45]	PolyLayout (ours)	**87.4**	**0.20 m**	**69.3**	**36.3**	**0.16 m**	**91.6**	3.54 s
2D-3D-Semantics [1]	Total3D [24] CVPR20	31.8	1.57 m	6.9	0.0	1.41 m	44.1	0.32 s
2D-3D-Semantics [1]	Implicit3D [47] CVPR21	31.6	1.59 m	5.6	0.0	1.51 m	37.6	**0.15 s**
2D-3D-Semantics [1]	Deep3DLayout [28] TOG21	58.8	0.68 m	16.7	0.0	0.44 m	52.6	1.30 s
2D-3D-Semantics [1]	LED^2-Net [41] CVPR21	68.7	0.45 m	20.6	5.0	0.40 m	34.6	0.71 s
2D-3D-Semantics [1]	PSMNet [42] CVPR22	43.5	1.04 m	7.3	0.0	0.63 m	51.4	2.32 s
2D-3D-Semantics [1]	PixCuboid [12] ICCVW25	89.0	0.18 m	**93.8**	**85.0**	**0.10 m**	96.1	0.42 s
2D-3D-Semantics [1]	PolyLayout (ours)	**90.0**	**0.15 m**	92.2	80.6	**0.10 m**	**96.4**	1.11 s

† Uses the semi-dense point cloud of the ASE dataset, computed from all images in the scene. SceneScript is trained on ASE, from which our test set is extracted.
‡ Calculated after first removing the floor & ceiling from the ground truth layout.

computing the Chamfer distance as it is not well-defined. Similarly, we skip views in which the predicted layout is not visible for the pixel-wise metrics. The prediction time metric measures only the inference time and does not include for example the dense COLMAP reconstruction (SceneScript, RoomFormer) or DeepLSD line detection (PixCuboid, PolyLayout).

Results: We first evaluate on our ASE test set (Tab. 1, top). Using the semi-dense SLAM point cloud, SceneScript can reconstruct the room geometry very accurately (first row). With point clouds created just from the 10 images per room there is a significant drop in performance (second row). Plane-DUSt3R, trained on Structured3D [48], does not generalize and achieves near-zero recall and worse pixel-wise metrics than SceneScript (third row). While only capable of estimating cuboid rooms, PixCuboid outperforms the two previously mentioned methods (fourth row). RoomFormer is by far the fastest but struggles to reconstruct floor plans from the relatively low density point cloud (fifth row). PolyLayout (last row) is considerably more accurate than the competing methods, even rivaling SceneScript with the semi-dense point cloud on some metrics.

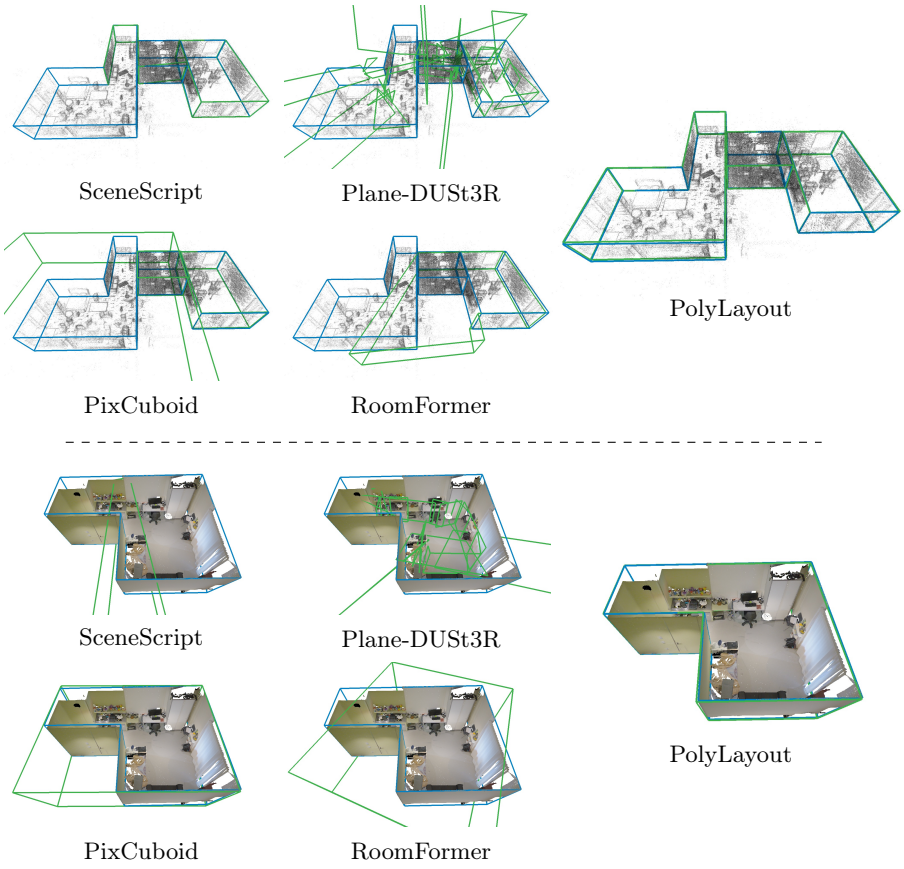

Fig. 5: Qualitative comparisons of predicted room layouts for one scene in each of our ASE and ScanNet++ test sets. Predictions are shown in **green** and the ground truth layouts in **blue**. The point cloud is displayed only for visualization purposes.

Next, the same methods are applied to the new ScanNet++ v2 test set with manually annotated room layouts (Tab. 1, center). This dataset contains rooms with more detailed geometry and shorter wall segments, making the recall problem more challenging. Results follow the same trend as for ASE, with PixCuboid and PolyLayout being the top two methods. The difference between them is smaller and can be explained by the fact that many of the rooms are not cuboids, but nearly cuboid-shaped.

Finally, we evaluate on the cuboid-shaped spaces of 2D-3D-Semantics (Tab. 1, bottom). As the rooms do not have the same floor and ceiling height we run PolyLayout with only shared orientation. Our method scores better on the 3D and pixel-wise metrics than PixCuboid, despite the latter being developed specifically for cuboid room layout estimation. No other method show competitive results on this dataset.

In Fig. 5 we visualize the predicted room layouts for one scene in each of the ASE and ScanNet++ test sets. Additional examples and failure cases can be found in the supplementary material.

5.2 Ablation Experiments

We perform several ablation experiments on the ASE test set to validate our design decisions. Parameter sharing across rooms is first considered, comparing single and multi-room layout estimation. In Tab. 2 (top section) it can be seen that when the orientation $\boldsymbol{R}$ is shared (second row) PolyLayout can more precisely estimate the layouts, with improvements on all metrics relative to the single room case (first row). With a common floor and ceiling height (d_1, d_2) we get even better 3D and pixel-wise metrics (third row).

The choice of encoder is justified in the second section of Tab. 2. Here we compare the ResNet-101 CNN from [12] (third row) with our proposed DINOv2 network (last row). Despite the networks having roughly the same number of parameters (30M vs 28.5M) the DINOv2-based one is superior, with large gains across every metric. To isolate the impact of the learned features we run PolyLayout with just the featuremetric cost E_{feat} (first two rows). DINOv2 is the best also in this case.

Table 2: Ablation experiments on our Aria Synthetic Environments test set.

		3D		Recall		Pixel-wise	
		IoU ↑	Chamfer ↓	Wall ↑	Room ↑	Depth ↓	Normal ↑
Sharing	None	93.5	0.13 m	88.5	78.5	0.11 m	97.9
	Orientation	93.8	0.13 m	88.9	**79.6**	0.10 m	98.0
	Orientation, floor, ceiling	**94.3**	**0.12 m**	**89.0**	79.0	**0.09 m**	**98.2**
Network	ResNet-101 (E_{feat})	24.6	3.55 m	8.5	0.5	1.70 m	28.5
	DINOv2 ViT-S/14 (E_{feat})	45.1	2.05 m	24.1	7.3	1.05 m	58.1
	ResNet-101	82.7	0.43 m	66.8	43.2	0.29 m	93.9
	DINOv2 ViT-S/14	**94.3**	**0.12 m**	**89.0**	**79.0**	**0.09 m**	**98.2**
Cost	E_{feat}, E_{edge}, E_{VP}	93.5	0.19 m	**89.3**	**79.1**	0.11 m	98.1
	E_{feat}, E_{edge}, E_{VP}, E_{per}	**94.3**	**0.12 m**	89.0	79.0	**0.09 m**	**98.2**
Init.	Cuboid	90.5	0.22 m	83.4	73.7	0.16 m	96.9
	Circle	93.2	0.15 m	87.3	77.7	0.11 m	97.8
	Concave hull	**94.3**	**0.12 m**	**89.0**	**79.0**	**0.09 m**	**98.2**
Shape	No split	93.6	0.14 m	**89.2**	**82.2**	0.10 m	98.0
	No simplification	92.9	0.16 m	86.9	72.5	0.11 m	97.5
	Simplify & split	**94.3**	**0.12 m**	89.0	79.0	**0.09 m**	**98.2**

Next, the third section of Tab. 2 shows that penalizing the perimeter of the layout polygon $\mathcal{P}$ with the perimeter cost E_{per} results in better 3D and pixel-wise metrics but slightly lower recall.

Three different ways to initialize the layout are compared (Tab. 2, section four). In addition to the method shown in Fig. 3 we initialize the polygon as a cuboid (first row) and an approximate circle (second row). The cuboid is initialized following [12] but the walls are split after the coarse and medium scales (same as for the polygons). To create the circle we compute the centroid of the camera centers in the local xy plane and set the radius to be maximum distance between it and the cameras, plus a margin of 1 m. Points (4 on each quadrant) are then sampled equiangularly on the circle defined by the centroid and radius, and pairs of walls added to connect adjacent points. Both of these methods result in inferior layout predictions compared to the concave hull initialization.

Lastly, we try disabling the wall splitting and simplification (Tab. 2, bottom section). With the proposed concave hull initialization the layout polygon typically starts with a fairly large number of walls, so PolyLayout is nearly as good without the splitting (first row compared to last). Turning off the simplification (second row) results in overly complex layouts and a bigger performance hit.

5.3 Layout Estimation without Known Poses

Our method assumes known camera poses but we believe this is a minor limitation in practice as they can easily be recovered with off-the-shelf methods such as Structure-from-Motion or recent regression-based alternatives.

As an experiment, we run π^3 [44] for each room in our ASE test set, then align the predicted poses with the ground truth using COLMAP's model aligner (to allow for evaluation) and finally estimate the layouts with PolyLayout. Results in Tab. 3 show that accuracy is only slightly worse with π^3's poses.

Table 3: Room layout estimation with predicted camera poses.

	3D		Recall		Pixel-wise	
Poses	IoU ↑	Chamfer ↓	Wall ↑	Room ↑	Depth ↓	Normal ↑
π^3 [44]	89.4	0.21 m	84.4	73.2	0.15 m	97.0
Ground truth	**94.3**	**0.12 m**	**89.0**	**79.0**	**0.09 m**	**98.2**

5.4 Comparison to Robust Model Fitting on Point Clouds

Finally we compare against traditional baselines which try to directly fit parametric models to a 3D point cloud. As before, we use COLMAP to produce dense point clouds from fixed poses. To simplify the model fitting we only consider the cuboid rooms from the ScanNet++ v2 dataset.

The first baseline method fits a cuboid directly to the dense point cloud using RANSAC [8], followed by non-linear refinement on the inlier set. As a second baseline we run our LM optimization as usual, but replace the cost function with $E_{dist}(\mathcal{C}) = \sum_i \rho\left(D(\mathcal{C}, \boldsymbol{P_i})\right)$ measuring the distance D between the cuboid $\mathcal{C}$ and the point cloud $\{\boldsymbol{P}_i\}$, in an ICP-style alignment. We apply a robust loss to handle points that are not on the walls, floor or ceiling. For both baselines we tune parameters on the validation set. We compare with PolyLayout using the featuremetric cost E_{feat}, with and without the edge cost E_{edge}. The second baseline and our method initialize the cuboids as in [12], but do not refine the orientation with E_{VP}. More details can be found in our supplementary material.

We present the results in Tab. 4. Both baselines struggle with outlier points and cannot match our featuremetric alignment. With RANSAC the probability to sample an inlier set is low, leading to poorly fitting cuboids. The distance-based optimization performs better but is still inferior to the learned costs.

Table 4: Comparison to point-based cuboid fitting on ScanNet++ v2.

	3D		Recall		Pixel-wise	
	IoU ↑	Chamfer ↓	Wall ↑	Room ↑	Depth ↓	Normal ↑
RANSAC	26.1	1.34 m	2.5	0.3	0.59 m	36.9
E_{dist}	45.9	1.17 m	19.1	5.0	0.57 m	49.6
E_{feat}	63.5	0.74 m	54.4	30.3	0.40 m	64.2
$E_{feat} + E_{edge}$	**92.4**	**0.15 m**	**95.0**	**88.1**	**0.06 m**	**97.2**

6 Conclusion

We have introduced PolyLayout, an optimization-based method for estimating Manhattan room layouts from perspective images. Our method supports an arbitrary number of input views and can jointly optimize the layouts of multiple rooms within a building. Results on both synthetic and real datasets show that it outperforms competing methods by a large margin.

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. The authors would like to thank Almer Hanning for annotating the ScanNet++ room layouts.

References

1. Armeni, I., Sax, S., Zamir, A.R., Savarese, S.: Joint 2D-3D-Semantic Data for Indoor Scene Understanding. arXiv preprint arXiv:1702.01105 (2017)
2. Avetisyan, A., Khanova, T., Choy, C., Dash, D., Dai, A., Nießner, M.: SceneCAD: Predicting Object Alignments and Layouts in RGB-D Scans. In: European Conference on Computer Vision (ECCV) (2020)
3. Avetisyan, A., Xie, C., Howard-Jenkins, H., Yang, T.Y., Aroudj, S., Patra, S., Zhang, F., Frost, D., Holland, L., Orme, C., Engel, J., Miller, E., Newcombe, R., Balntas, V.: SceneScript: Reconstructing Scenes With An Autoregressive Structured Language Model. In: European Conference on Computer Vision (ECCV) (2024)
4. Barron, J.T.: A General and Adaptive Robust Loss Function. In: Computer Vision and Pattern Recognition (CVPR) (2019)
5. Coughlan, J.M., Yuille, A.L.: Manhattan World: Compass Direction from a Single Image by Bayesian Inference. In: International Conference on Computer Vision (ICCV) (1999)
6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. International Conference on Learning Representations (ICLR) (2021)
7. Edelsbrunner, H., Kirkpatrick, D., Seidel, R.: On the Shape of a Set of Points in the Plane. IEEE Transactions on information theory **29**(4), 551–559 (2003)
8. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24**(6), 381–395 (1981)
9. Flint, A., Murray, D., Reid, I.: Manhattan Scene Understanding Using Monocular, Stereo, and 3D Features. In: International Conference on Computer Vision (ICCV) (2011)
10. Gillies, S., van der Wel, C., Van den Bossche, J., Taves, M.W., Arnott, J., Ward, B.C., others: Shapely (May 2025). `https://doi.org/10.5281/zenodo.5597138`
11. Hampali, S., Stekovic, S., Sarkar, S.D., Kumar, C.S., Fraundorfer, F., Lepetit, V.: Monte Carlo Scene Search for 3D Scene Understanding. In: Computer Vision and Pattern Recognition (CVPR) (2021)
12. Hanning, G., Åström, K., Larsson, V.: PixCuboid: Room Layout Estimation from Multi-view Featuremetric Alignment. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops (2025)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Computer Vision and Pattern Recognition (CVPR) (2016)
14. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the Spatial Layout of Cluttered Rooms. In: International Conference on Computer Vision (ICCV) (2009)
15. Hu, Z., Duan, B., Zhang, Y., Sun, M., Huang, J.: MVLayoutNet: 3D Layout Reconstruction with Multi-view Panoramas. In: Proceedings of the 30th ACM International Conference on Multimedia. pp. 1289–1298 (2022)
16. Huang, X., Mei, G., Zhang, J.: Feature-metric Registration: A Fast Semi-supervised Approach for Robust Point Cloud Registration without Correspondences. In: Computer Vision and Pattern Recognition (CVPR) (2020)
17. Huang, Y., Dai, X., Wang, J., Qi, X., Yuan, Y., Yue, X.: Unposed Sparse Views Room Layout Reconstruction in the Age of Pretrain Model. In: International Conference on Learning Representations (ICLR) (2025)

18. Lee, D.C., Hebert, M., Kanade, T.: Geometric Reasoning for Single Image Structure Recovery. In: Computer Vision and Pattern Recognition (CVPR) (2009)
19. Levenberg, K.: A method for the solution of certain non-linear problems in least squares. Quarterly of applied mathematics **2**(2), 164–168 (1944)
20. Li, Y., Boyadzhiev, I., Liu, Z., Shapiro, L., Colburn, A.: BADGR: Bundle Adjustment Diffusion Conditioned by GRadients for Wide-Baseline Floor Plan Reconstruction. In: Computer Vision and Pattern Recognition (CVPR) (2025)
21. Lindenberger, P., Sarlin, P.E., Larsson, V., Pollefeys, M.: Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In: International Conference on Computer Vision (ICCV) (2021)
22. Mao, Y., Zhong, J., Fang, C., Zheng, J., Tang, R., Zhu, H., Tan, P., Zhou, Z.: SpatialLM: Training Large Language Models for Structured Indoor Modeling. In: Neural Information Processing Systems (NeurIPS) (2025)
23. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear parameters. Journal of the society for Industrial and Applied Mathematics **11**(2), 431–441 (1963)
24. Nie, Y., Han, X., Guo, S., Zheng, Y., Chang, J., Zhang, J.J.: Total3DUnderstanding: Joint Layout, Object Pose and Mesh Reconstruction for Indoor Scenes from a Single Image. In: Computer Vision and Pattern Recognition (CVPR) (2020)
25. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Howes, R., Huang, P.Y., Xu, H., Sharma, V., Li, S.W., Galuba, W., Rabbat, M., Assran, M., Ballas, N., Synnaeve, G., Misra, I., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning Robust Visual Features without Supervision. In: International Conference on Learning Representations (ICLR) (2025)
26. Pautrat, R., Barath, D., Larsson, V., Oswald, M.R., Pollefeys, M.: DeepLSD: Line Segment Detection and Refinement with Deep Image Gradients. In: Computer Vision and Pattern Recognition (CVPR) (2023)
27. Pintore, G., Agus, M., Gobbetti, E.: AtlantaNet: Inferring the 3D Indoor Layout from a Single 360° Image beyond the Manhattan World Assumption. In: European Conference on Computer Vision (ECCV) (2020)
28. Pintore, G., Almansa, E., Agus, M., Gobbetti, E.: Deep3DLayout: 3D Reconstruction of an Indoor Layout from a Spherical Panoramic Image. ACM Transactions on Graphics (TOG) **40**(6), 1–12 (2021)
29. Pintore, G., Ganovelli, F., Pintus, R., Scopigno, R., Gobbetti, E.: 3d floor plan recovery from overlapping spherical images. Computational visual media **4**, 367–383 (2018)
30. Sarlin, P.E., Unagar, A., Larsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Hammarstrand, L., Kahl, F., et al.: Back to the Feature: Learning Robust Camera Localization from Pixels to Pose. In: Computer Vision and Pattern Recognition (CVPR) (2021)
31. Schindler, G., Dellaert, F.: Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In: Computer Vision and Pattern Recognition (CVPR) (2004)
32. Schönberger, J.L., Frahm, J.M.: Structure-from-Motion Revisited. In: Computer Vision and Pattern Recognition (CVPR) (2016)
33. Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise View Selection for Unstructured Multi-View Stereo. In: European Conference on Computer Vision (ECCV) (2016)

34. Schwing, A.G., Hazan, T., Pollefeys, M., Urtasun, R.: Efficient Structured Prediction for 3D Indoor Scene Understanding. In: Computer Vision and Pattern Recognition (CVPR) (2012)
35. Stekovic, S., Hampali, S., Rad, M., Sarkar, S.D., Fraundorfer, F., Lepetit, V.: General 3D Room Layout from a Single View by Render-and-Compare. In: European Conference on Computer Vision (ECCV) (2020)
36. Stekovic, S., Rad, M., Fraundorfer, F., Lepetit, V.: MonteFloor: Extending MCTS for Reconstructing Accurate Large-Scale Floor Plans. In: International Conference on Computer Vision (ICCV) (2021)
37. Su, J.W., Peng, C.H., Wonka, P., Chu, H.K.: GPR-Net: Multi-view Layout Estimation via a Geometry-aware Panorama Registration Network. In: Computer Vision and Pattern Recognition (CVPR) (2023)
38. Sun, C., Hsiao, C.W., Sun, M., Chen, H.T.: HorizonNet: Learning Room Layout with 1D Representation and Pano Stretch Data Augmentation. In: Computer Vision and Pattern Recognition (CVPR) (2019)
39. Tardif, J.P.: Non-Iterative Approach for Fast and Accurate Vanishing Point Detection. In: International Conference on Computer Vision (ICCV) (2009)
40. Visvalingam, M., Whyatt, J.: Line generalisation by repeated elimination of points. Cartographic Journal **30**(1), 46–51 (1993)
41. Wang, F.E., Yeh, Y.H., Sun, M., Chiu, W.C., Tsai, Y.H.: LED^2-Net: Monocular 360° Layout Estimation via Differentiable Depth Rendering. In: Computer Vision and Pattern Recognition (CVPR) (2021)
42. Wang, H., Hutchcroft, W., Li, Y., Wan, Z., Boyadzhiev, I., Tian, Y., Kang, S.B.: PSMNet: Position-aware Stereo Merging Network for Room Layout Estimation. In: Computer Vision and Pattern Recognition (CVPR) (2022)
43. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: DUSt3R: Geometric 3D Vision Made Easy. In: Computer Vision and Pattern Recognition (CVPR) (2024)
44. Wang, Y., Zhou, J., Zhu, H., Chang, W., Zhou, Y., Li, Z., Chen, J., Pang, J., Shen, C., He, T.: π^3: Permutation-Equivariant Visual Geometry Learning. International Conference on Learning Representations (ICLR) (2025)
45. Yeshwanth, C., Liu, Y.C., Nießner, M., Dai, A.: ScanNet++: A High-Fidelity Dataset of 3D Indoor Scenes. In: International Conference on Computer Vision (ICCV) (2023)
46. Yue, Y., Kontogianni, T., Schindler, K., Engelmann, F.: Connecting the Dots: Floorplan Reconstruction Using Two-Level Queries. In: Computer Vision and Pattern Recognition (CVPR) (2023)
47. Zhang, C., Cui, Z., Zhang, Y., Zeng, B., Pollefeys, M., Liu, S.: Holistic 3D Scene Understanding from a Single Image with Implicit Representation. In: Computer Vision and Pattern Recognition (CVPR) (2021)
48. 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 (ECCV) (2020)
49. Zou, C., Colburn, A., Shan, Q., Hoiem, D.: LayoutNet: Reconstructing the 3D Room Layout from a Single RGB Image. In: Computer Vision and Pattern Recognition (CVPR) (2018)