

# PlanarSplatting: Accurate Planar Surface Reconstruction in 3 Minutes

Bin Tan<sup>1</sup> Rui Yu<sup>2</sup> Yujun Shen<sup>1</sup> Nan Xue<sup>†,1</sup>

<sup>1</sup>Ant Group <sup>2</sup>University of Louisville

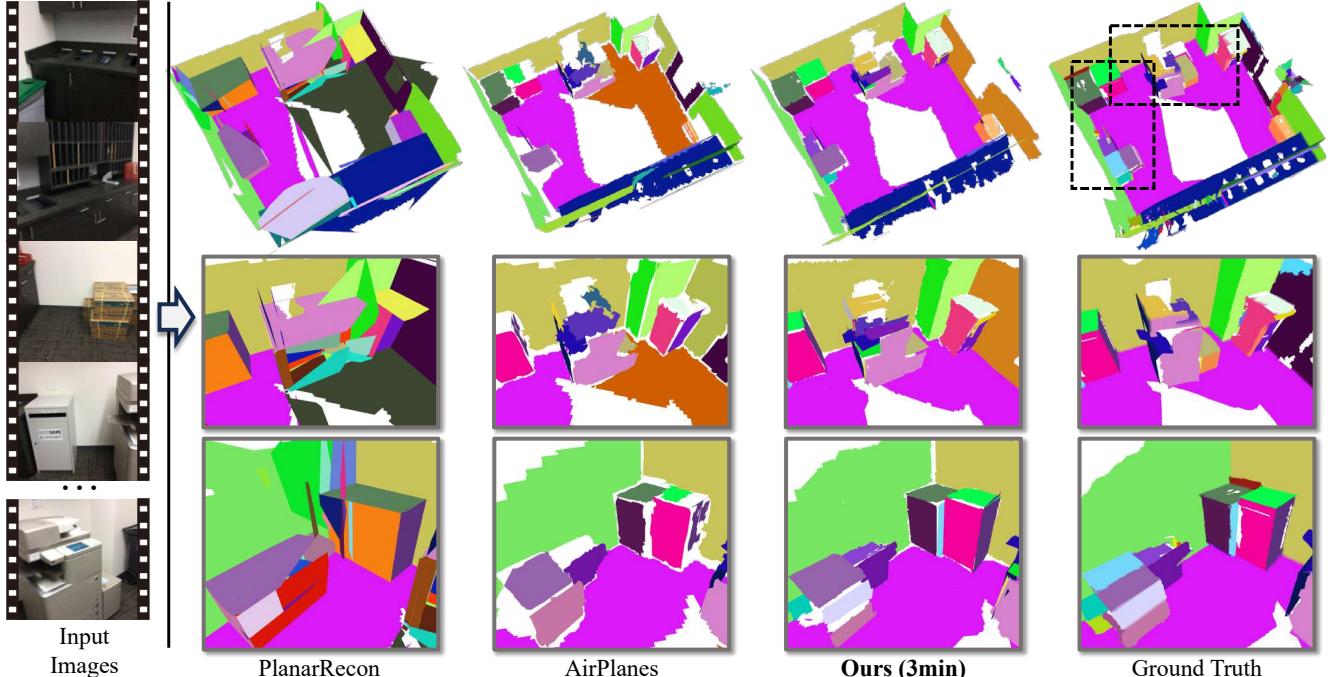


Figure 1. We introduce *PlanarSplatting*, a fast and accurate optimization-based planar reconstruction method for indoor scenes. We show our planar reconstruction results on the ScanNetV2 [2] dataset achieved in 3 minutes. Compared to prior art PlanarRecon [37] and AirPlanes [35], our *PlanarSplatting* reconstructs more complete and detailed 3D planes.

## Abstract

This paper presents *PlanarSplatting*, an ultra-fast and accurate surface reconstruction approach for multi-view indoor images. We take the 3D planes as the main objective due to their compactness and structural expressiveness in indoor scenes, and develop an explicit optimization framework that learns to fit the expected surface of indoor scenes by splatting the 3D planes into 2.5D depth and normal maps. As our *PlanarSplatting* operates directly on the 3D plane primitives, it eliminates the dependencies on 2D/3D plane detection and plane matching/tracking for planar surface reconstruction. Furthermore, with the essential

merits of plane-based representation coupled with CUDA-based implementation of planar splatting functions, *PlanarSplatting* reconstructs an indoor scene **in 3 minutes** while having significantly better geometric accuracy. Thanks to our ultra-fast reconstruction speed, the largest quantitative evaluation on the ScanNet and ScanNet++ datasets over hundreds of scenes clearly demonstrated the advantages of our method. We believe that our accurate and ultrafast planar surface reconstruction method will be applied in the structured data curation for surface reconstruction in the future. The code of our CUDA implementation will be publicly available.

<sup>†</sup>Corresponding author.



Figure 2. **Novel View Synthesis:** Seamless integration of our *PlanarSplatting* with recent Gaussian Splatting methods (e.g., 3DGS [13] and 2DGS [11]) achieves indoor-scene rendering with improved performance and significant optimization time reduction simultaneously.

## 1. Introduction

We humans are long-term immersed in structured indoor scenes, ranging from bedrooms to offices. This fact has ignited a wealth of studies for reconstructing the indoor environment with various 3D structures such as lines [9, 20, 39], planes [12, 16, 18, 19, 33, 37, 42] and blocks [7, 23]. Among them, the 3D plane is the most common representation because of its simplicity and completeness in describing physical surfaces, thus motivating us to study the problem of 3D reconstruction for indoor scenes using 3D planes, *i.e.*, planar 3D reconstruction.

Planar 3D reconstruction has been extensively studied for years as a model fitting problem, in which a 3D scene geometry (*e.g.*, point clouds, or meshes) was assumed to be known and the main goal is fitting the scene in a set of 3D planes [25, 30, 38]. Recently, the paradigm has been gradually simplified in image-based solutions in single-view and multi-view 3D reconstruction, eliminating the acquisition of known 3D scene geometry. To make the problem trackable, existing methods were extensively based on the image-level characterization of 2D/3D planes. That is to say, those methods have to detect 3D planes for each input image, match or track planes in across viewpoints, and finally reconstruct and merge 3D planes as the compact 3D representation of indoor scenes. Recent PlanarRecon [37] attempted to address these problems end-to-end by learning a 3D volume from monocular videos. Then, 3D planes can be detected, tracked, and fused in a consistent 3D space.

Image-based planar 3D reconstruction mostly followed keypoint-based 3D reconstruction pipelines and treated planes as a special kind of visual features. However, because of the essential difference between the local point features and regional plane masks in image space, we argue, the existing approaches did not fully leverage the advantages of planar representations. Some evidences could be observed from the results of PlanarRecon [37], which are

usually very coarse and lose many details of the scene.

In this paper, we are going to address the issues remained in image-based planar 3D reconstruction, aiming at obtaining a **complete**, **structural**, and **compact** indoor scene reconstruction from multi-view images. Our main idea is approximating the indoor scenes with a collection of solid 3D planar primitives from multi-view input images, directly optimizing them to have consistent 3D planes, but eliminating any suboptimal precomputing of 2D plane primitives (*e.g.*, plane masks). We introduce *PlanarSplatting* that explicitly optimizes rectangular plane primitives in 3D space by differentiably splatting them into 2.5D depth and normal maps. Thanks to our well-designed plane splatting function, *PlanarSplatting* effectively leverage monocular geometry cues from modern foundational models [3, 10, 41] for accurate plane optimization. As shown in Fig. 1, the final high-quality planar surface can be reconstructed by simply merging adjacent 3D plane primitives.

Our *PlanarSplatting* is directly designed on the 3D plane primitives and efficiently implemented with CUDA, which accordingly induces a seamlessly integration with recent Gaussian Splatting (GS) methods for high-quality indoor novel view synthesis (NVS). As shown in Fig. 2, benefiting from our fast and accurate scene reconstruction (within 3 minutes), GS-based methods can be well-initialized and optimized without densification, leading to better rendering results and significantly less training time. It demonstrates the strong potential to promote the unity of surface reconstruction and novel view synthesis for indoor scenes.

In the experiments, our *PlanarSplatting* shows its powerful ability for accurate indoor planar surface reconstruction for hundreds of scenes from two real-world indoor datasets, *i.e.* ScanNetV2 [2] and ScanNet++[40]. Furthermore, we show that with the combination of our *PlanarSplatting* and GS-based methods (*e.g.*, 3DGS [13] and 2DGS [11]), we can effectively improve the rendering quality with less training time and fewer points.

## 2. Related Work

**Indoor Planar Reconstruction.** Reconstructing indoor scenes with 3D plane primitives has been studied for a long time [4–6, 29, 36]. Traditionally, it is usually achieved by deducing and fitting the plane geometry directly from 3D data (*e.g.*, point clouds [24, 25, 30, 31] and line clouds [15]), or from 2D single-view images with strict scene assumptions (*e.g.*, the Manhattan World constrain) [16, 22, 26] which seriously limited their application. In recent years, some learnable-based methods are proposed to formulate this problem as single-view 3D plane segmentation [17, 18, 28, 32, 42, 43] and cross-view plane instance matching [1, 12, 33]. Although impressive results have been achieved, these works are hard to extend to multiple views. PlanarRecon [37] was the first end-to-end work proposed to deal with holistic indoor plane recovery by learning a plane-related 3D volume from posed RGB videos. Then, 3D planes can be extracted, tracked, and fused from the learned 3D volume incrementally. Most recently, AirPlanes [35] developed a two-step method that first reconstructed a dense scene mesh and then learned consistent 3D plane embeddings from 2D plane embeddings for plane extraction from the dense mesh. Note that the aforementioned learning-based methods require 2D/3D plane annotations as their supervision, leading to performance bottlenecks due to the difficulty of obtaining a large scale of plane annotations. In contrast, we proposed *PlanarSplatting* to reconstruct accurate and complete indoor planar surfaces by directly optimizing a set of solid 3D plane primitives from posed multi-view images without any extra plane detection or matching operations. Benefiting from our differentiable planar primitive rendering, *PlanarSplatting* directly leverages monocular depth/normal cues from modern foundation models for optimization without plane annotations. Concurrent work, AlphaTablets [8], explores planar representations, which involves learning planar tablets equipped with alpha maps for shape modeling and pseudo-mesh construction for rendering. Our *PlanarSplatting* distinguishes itself from AlphaTablets by learning plane radii, providing a more explicit parameterization of 3D planes. As a result, we achieve a much more efficient rasterization scheme at the level of minutes while AlphaTablets takes about 2 hours per scene.

**Primitive-based Scene Representation.** Optimizing explicit primitives such as points [11, 13, 34, 44], volumes [21], and superquadric [23] to represent the 3D scene has been studied for a long time. The core of these methods is to design a differentiable rendering process to optimize the attributes of primitives by gradient descent. Among them, a typical paradigm is to render images from primitives with splatting techniques which is realized with radial basis functions defined on the primitive (*e.g.*, the Gaussian function). Inspired by these works, we proposed

*PlanarSplatting* to optimize solid 3D plane primitives in a differentiable rendering manner with a novel shape-aware plane splatting function to better fit the scene geometry. Benefiting from our efficient CUDA implementation, *PlanarSplatting* can reconstruct the accurate indoor planar scene within 3 minutes and empower recent Gaussian-based works (*e.g.*, 3DGS [13]) to further improve the rendering quality of novel views as shown in Fig. 2.

## 3. The Proposed PlanarSplatting

As shown in Fig. 3, given a set of posed multi-view images, our *PlanarSplatting* can reconstruct the indoor planar scene from them by optimizing a set of learnable 3D planar primitives (Sec. 3.1). With the proposed Differentiable Planar Primitive Rendering (Sec. 3.2), we can explicitly learn these planar primitives from a coarse initialization to accurately recover the scene geometry. These optimized 3D planar primitives are then merged to achieve the 3D plane instances for the final planar reconstruction.

### 3.1. Learnable Planar Scene Representation

Since our key idea is directly optimizing explicit 3D primitives for planar scene reconstruction, we first present the representation of our learnable 3D planar primitives and then introduce how we initialize the scene with these planar primitives.

**Learnable Planar Primitive.** As shown in Fig. 4, we formulate a planar primitive  $\pi$  as a 3D rectangle which is equipped with several learnable parameters including the plane center  $\mathbf{p}_\pi \in \mathbb{R}^3$ , the plane rotation  $\mathbf{q}_\pi \in \mathbb{R}^4$  (in quaternion representation) and the plane radii  $\mathbf{r}_\pi$ . Specifically, to improve the optimization flexibility of plane shape, we use the design of double direction plane radii (Double Radii) for  $\mathbf{r}_\pi$  as:

$$\mathbf{r}_\pi = \{r_\pi^{x+}, r_\pi^{x-}, r_\pi^{y+}, r_\pi^{y-}\} \in \mathbb{R}_+^4, \quad (1)$$

where  $r_\pi^{x+}, r_\pi^{x-}, r_\pi^{y+}, r_\pi^{y-}$  are the radii defined on the positive/negative direction of the X-axis/Y-axis of the rectangle as shown in Fig. 4. For the sake of description, we further define the positive direction of the X-axis and Y-axis of the 3D planar primitive as two orthogonal unit vectors  $\mathbf{v}_\pi^x, \mathbf{v}_\pi^y \in \mathbb{R}^3$  which can be calculated as:

$$\mathbf{v}_\pi^x = \mathbf{R}(\mathbf{q}_\pi)[1, 0, 0]^\top, \quad \mathbf{v}_\pi^y = \mathbf{R}(\mathbf{q}_\pi)[0, 1, 0]^\top, \quad (2)$$

where  $\mathbf{R}(\mathbf{q}_\pi) \in \mathbb{R}^{3 \times 3}$  means the rotation matrix of the quaternion  $\mathbf{q}_\pi$ . Similarly, the normal of the planar primitive  $\mathbf{n}_\pi \in \mathbb{R}^3$  can be calculated as:

$$\mathbf{n}_\pi = \mathbf{R}(\mathbf{q}_\pi)[0, 0, 1]^\top. \quad (3)$$

With these learnable parameters, the 3D planar primitive can be moved to align with the potential scene surface and deformed to fit the surface shape during optimization.

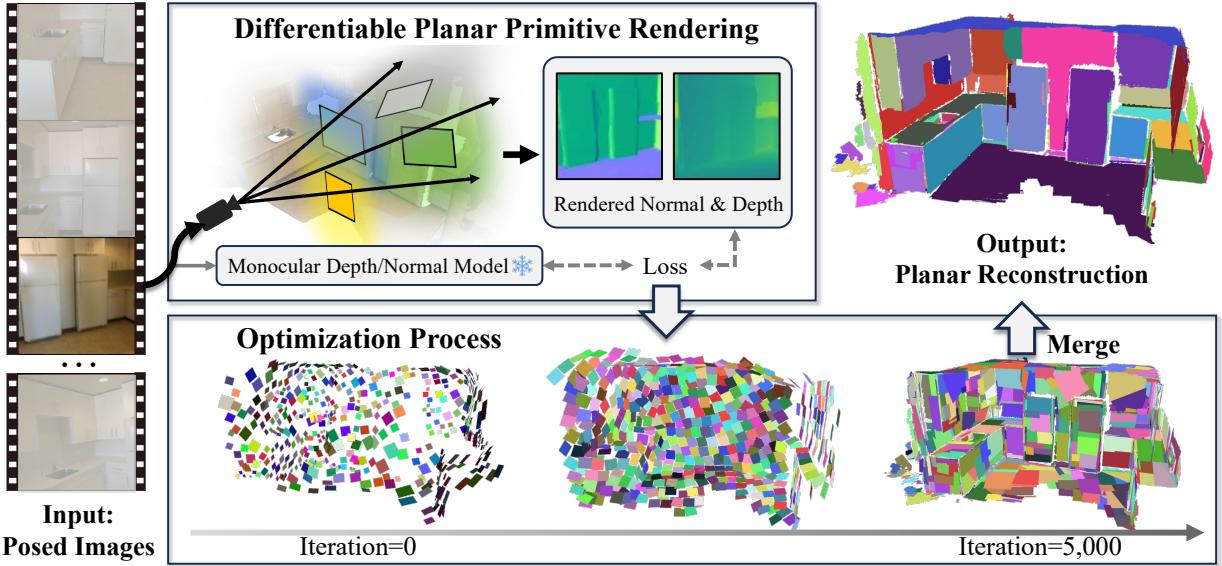


Figure 3. **Illustration of our proposed *PlanarSplatting*.** Given a set of posed multi-view images of indoor scenes, our method renders depth and normal maps from 3D plane primitives. Then, with the supervision of monocular cues, these 3D plane primitives are gradually optimized to recover the scene geometry and finally merged to get the planar reconstruction result.

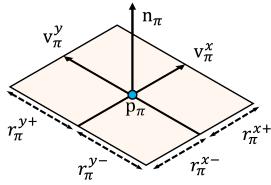


Figure 4. Representation of 3D plane primitive with learnable shape parameters including plane center, radii, and rotation.

**Scene Initialization.** We use monocular depth from recent foundation models [10] to fast initialize our 3D planar primitives at the beginning of optimization. Specifically, we use depths from Metric3Dv2 [10] to get a very coarse scene geometry. Then we randomly sample 2,000 points on the coarse mesh to achieve the plane centers of our 3D planar primitives. The initial radii of each primitive  $\pi$  is set to  $0.5Dist(\pi)$ . Here,  $Dist(\pi)$  means the distance closest to  $\pi$  to its neighbors. We use the normal direction on the coarse mesh to initialize the plane rotation. At the bottom of Fig. 3, we show an example of our initial planar primitives. With such a coarse initialization, our *PlanarSplatting* can finally reconstruct the accurate and complete scene surface.

### 3.2. Differentiable Planar Primitive Rendering

To optimize the learnable planar primitives  $\Pi = \{\pi_i\}_{i=1}^K$ , we introduce the differentiable planar primitive rendering with a carefully designed plane splatting function which enables the planar primitives to accurately fit the scene geometry with the supervision only from 2D multi-view images.

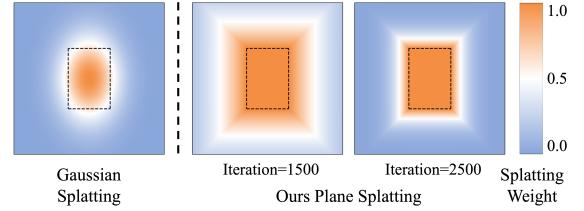


Figure 5. **Illustration of the proposed plane splatting function.** Naive Gaussian Splatting can not effectively approximate the boundary of our rectangular plane primitive (shown in black dashed border). In contrast, our proposed plane splatting function can approximate the boundary of the rectangle as the number of iterations increases, allowing our 3D planar primitives to better fit the surface of the scene.

**Ray-to-Plane Intersection.** To project 3D planar primitives to the 2D image space, we first calculate the intersections between planar primitives and the rays emitted from image pixels. Specifically, given a ray  $\mathbf{r} = \{\mathbf{o}, \mathbf{d}\}$  starting from the camera center  $\mathbf{o} \in \mathbb{R}^3$  with direction  $\mathbf{d} \in \mathbb{R}^3$ , its intersection  $\mathbf{x}_\pi^r \in \mathbb{R}^3$  to one planar primitive  $\pi$  is:

$$\mathbf{x}_\pi^r = \mathbf{o} + \frac{(\mathbf{p}_\pi - \mathbf{o}) \cdot \mathbf{n}_\pi}{\mathbf{d} \cdot \mathbf{n}_\pi} \mathbf{d}, \quad (4)$$

where  $\mathbf{p}_\pi$  and  $\mathbf{n}_\pi$  are the center and the normal of the planar primitive  $\pi$ .

**Plane Splatting Function.** After achieving the ray-to-plane intersection  $\mathbf{x}_\pi^r$ , we calculate its splatting weight with our plane splatting function which will be used for rendering.

A vanilla selection of the splatting function is the

anisotropic Gaussian function:

$$w(\mathbf{x}_\pi^r, \pi) = \exp \left( -\frac{1}{2} (\mathbf{x}_\pi^r - \mathbf{p}_\pi)^\top \Sigma^{-1} (\mathbf{x}_\pi^r - \mathbf{p}_\pi) \right), \quad (5)$$

where  $\mathbf{p}_\pi$  is the center of the planar primitive  $\pi$ . The covariance matrix  $\Sigma$  can be calculated like [11, 13]. However, as shown in Fig. 5 and Fig. 6, the Gaussian-based splatting function will make ambiguous boundaries of our rectangular planar primitive leading to the degeneration of reconstruction quality.

Thus, we propose to calculate the splatting weight with a novel rectangle-based plane splatting function. To a given intersection  $\mathbf{x}_\pi^r$  between ray  $\mathbf{r}$  and planar primitive  $\pi$ , we first calculate its projection distance  $\mathcal{P}_X, \mathcal{P}_Y \in \mathbb{R}$  to the X-axis and Y-axis of the planar primitive as:

$$\mathcal{P}_X = (\mathbf{x}_\pi^r - \mathbf{p}_\pi) \cdot \mathbf{v}_\pi^x, \quad \mathcal{P}_Y = (\mathbf{x}_\pi^r - \mathbf{p}_\pi) \cdot \mathbf{v}_\pi^y. \quad (6)$$

Then, we calculate the splatting weight along the X-axis of the plane  $\pi$  as:

$$w_X(\mathbf{x}_\pi^r) = \begin{cases} 2\sigma(5\lambda(r_\pi^{x+} - |\mathcal{P}_X|)), & \text{if } \mathcal{P}_X > 0, \\ 2\sigma(5\lambda(r_\pi^{x-} - |\mathcal{P}_X|)), & \text{otherwise,} \end{cases} \quad (7)$$

where  $\sigma(\cdot)$  is the Sigmoid function and  $\lambda$  is the hyperparameter to control the splatting weight. Similarly, we continue to calculate the splatting weight along the Y-axis of the plane  $\pi$  as:

$$w_Y(\mathbf{x}_\pi^r) = \begin{cases} 2\sigma(5\lambda(r_\pi^{y+} - |\mathcal{P}_Y|)), & \text{if } \mathcal{P}_Y > 0, \\ 2\sigma(5\lambda(r_\pi^{y-} - |\mathcal{P}_Y|)), & \text{otherwise,} \end{cases} \quad (8)$$

where  $r_\pi^{x+}, r_\pi^{x-}, r_\pi^{y+}, r_\pi^{y-}$  are the radii parameters of the plane  $\pi$ . At last, the final splatting weight can be calculated as:

$$w(\mathbf{x}_\pi^r) = \begin{cases} w_X, & \text{if } w_X < w_Y, \\ w_Y, & \text{otherwise.} \end{cases} \quad (9)$$

In Fig. 5, we show that with the increment of hyperparameter  $\lambda$ , our plane splatting function gradually approximates the shape of the rectangular plane primitive. In practice, we increase the value of  $\lambda$  with an exponential function during optimization up to the maximum value of 300 as:

$$\lambda = \min(20e^{-(1-0.001*ite)}, 300), \quad (10)$$

where  $ite$  means the iteration number during optimization.

**Blending Composition.** For all ray-to-plane intersections, we filter them with splatting weight lower than 0.0001 and then sort the remaining intersections according to their depth from near to far. Then,  $M$  nearest intersections of each ray are selected for rendering ( $M = 30$  in this paper). Denote the selected intersections of a ray  $\mathbf{r}$  as  $P^r = \{\mathbf{x}_{\pi_{\tau(j)}}^r\}_{j=1}^M$ . Here,  $\tau(j)$  indicates the index of plane

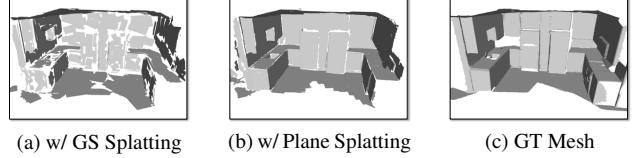


Figure 6. Reconstruction comparison with different splatting functions. ‘w’ means ‘with’.

among all planar primitives. At last, we render the depth and normal map of a certain image  $\mathbf{I}$  as:

$$\mathbf{D}_{\text{render}}^{\Pi}(\mathbf{r}) = \sum_{j=1}^M T_j w(\mathbf{x}_{\pi_{\tau(j)}}^r) t_j, \quad (11)$$

$$\mathbf{N}_{\text{render}}^{\Pi}(\mathbf{r}) = \sum_{j=1}^M T_j w(\mathbf{x}_{\pi_{\tau(j)}}^r) \mathbf{n}_{\pi_{\tau(j)}}, \quad (12)$$

where

$$T_j = \prod_{i=1}^{j-1} (1 - w(\mathbf{x}_{\pi_{\tau(i)}}^r)). \quad (13)$$

Here  $t_j$  is the depth of the intersection and  $\mathbf{n}_{\pi_{\tau(j)}}$  is the normal of planar primitive  $\pi_{\tau(j)}$ . To supervise the rendered depth and normal map, we use the pretrained model of Metric3Dv2 [10] to predict the depth map  $\mathbf{D}_{\text{pre}}$  and normal map  $\mathbf{N}_{\text{pre}}$  to serve as pseudo labels. Besides, we also test the normal map predicted by Omnidata [3]. Finally, the render loss can be calculated as:

$$\begin{aligned} \mathcal{L}_{\text{render}}^{\Pi} = & \alpha_1 \sum_{\mathbf{r} \in \mathbf{I}} \|1 - \mathbf{N}_{\text{render}}^{\Pi}(\mathbf{r})^\top \mathbf{N}_{\text{pre}}(\mathbf{r})\|_1 + \\ & \alpha_1 \sum_{\mathbf{r} \in \mathbf{I}} \|\mathbf{N}_{\text{render}}^{\Pi}(\mathbf{r}) - \mathbf{N}_{\text{pre}}(\mathbf{r})\|_1 + \\ & \alpha_2 \sum_{\mathbf{r} \in \mathbf{I}} \|(\mathbf{D}_{\text{render}}^{\Pi}(\mathbf{r})) - \mathbf{D}_{\text{pre}}(\mathbf{r})\|_1, \end{aligned} \quad (14)$$

where  $\alpha_1 = 2.0, \alpha_2 = 2.0$ ,  $\mathbf{r}$  is the ray/pixel emitted from image  $\mathbf{I}$ .

### 3.3. Optimization

**Loss Function.** To each scene, we optimize our *Planar-Splatting* with the Adam optimizer [14] for 5,000 iterations on the ScanNetV2 [2] dataset and 10,000 iterations on the ScanNet++ [40] dataset using the loss as shown in Eq. (14).

**Plane Splitting.** During optimization, we introduce a splitting operation on planes according to the gradients of their radii to better fit the scene geometry. If the average radii gradients on the X-axis ( $r^{x+}$  and  $r^{x-}$ ) of the plane are greater than 0.2, we split the plane along the Y-axis. Similarly, we split the planes along the X-axis, if their radii gradients on the Y-axis ( $r^{y+}$  and  $r^{y-}$ ) are larger than 0.2. We conduct the splitting operation every 1,000 iterations.

Table 1. Quantitative comparison of planar reconstruction results on the ScanNetV2 [2] dataset. ‘P. Ann.’ means using 2D/3D plane annotations in training stages.

Method	P. Ann.	Geometry		Segmentation			Planar		
		Chamfer ↓	F-score ↑	VOI ↓	RI ↑	SC ↑	Fidelity ↓	Acc ↓	Chamfer ↓
PlanarRecon [37]	✓	9.89	43.47	3.201	0.919	0.405	18.86	16.21	17.53
AirPlanes [35]	✓	5.30	64.92	<b>2.268</b>	<u>0.957</u>	<b>0.568</b>	8.76	<b>7.98</b>	<u>8.37</u>
2DGS [11] + RANSAC	✗	14.15	31.33	4.030	0.924	0.257	40.02	14.77	27.40
Metric3Dv2 [10] + RANSAC	✗	5.93	65.45	2.744	0.927	0.488	9.29	11.36	10.33
SR [27] + RANSAC	✗	5.40	65.45	2.507	0.946	0.515	9.42	<u>10.13</u>	9.78
Ours@Omnidata	✗	<u>4.83</u>	<u>68.85</u>	2.502	0.948	0.532	<u>6.64</u>	11.76	9.20
Ours@Metric3Dv2	✗	<b>4.23</b>	<b>73.76</b>	<u>2.310</u>	<b>0.958</b>	<u>0.560</u>	<b>5.72</b>	10.65	<b>8.19</b>

Table 2. Quantitative comparison of planar reconstruction results on the ScanNet++ [40] dataset. ‘P. Ann.’ means using 2D/3D plane annotations in training stages.

Method	P. Ann.	Geometry		Segmentation			Planar		
		Chamfer ↓	F-score ↑	VOI ↓	RI ↑	SC ↑	Fidelity ↓	Acc ↓	Chamfer ↓
PlanarRecon [37]	✓	17.85	31.10	3.542	0.919	0.367	34.44	19.35	26.90
AirPlanes [35]	✓	13.75	32.58	2.859	0.941	0.470	28.16	12.58	20.37
2DGS [11] + RANSAC	✗	20.39	26.46	4.456	0.927	0.241	55.90	16.88	36.39
Metric3Dv2 [10] + RANSAC	✗	11.14	41.54	3.021	0.930	0.441	27.50	<b>10.81</b>	19.16
SR [27] + RANSAC	✗	13.15	35.93	3.013	0.938	0.442	30.25	<u>11.62</u>	20.94
Ours@Omnidata	✗	<u>9.33</u>	<u>47.04</u>	<b>2.772</b>	<b>0.946</b>	<b>0.523</b>	<u>17.24</u>	12.26	<b>14.75</b>
Ours@Metric3Dv2	✗	<b>9.01</b>	<b>50.68</b>	<u>2.782</u>	<u>0.944</u>	<u>0.521</u>	<b>17.17</b>	12.36	<u>14.77</u>

**Plane Merge.** After optimization, we greedily merge the adjacent 3D plane primitives with normal angle error lower than  $25^\circ$  and offset distance error lower than  $0.1m$ . Here, offset means the projection distance from the scene center to the plane surface.

**CUDA Implementation.** For fast optimization of solid 3D planar primitives, we implement the forward and backward process of our Differentiable Planar Primitive Rendering with CUDA, which enables our *PlanarSplatting* to reconstruct one scene within 3 minutes. We will release the code of our CUDA implementation for solid 3D planar primitives optimization after publication.

## 4. Experiments

### 4.1. Dataset, Metrics and Baselines

**Datasets.** We evaluate our *PlanarSplatting* on two large indoor datasets including ScanNetV2 [2] and ScanNet++ [40] which provide posed images. On the ScanNetV2 dataset, we use the test split according to AirPlanes [35] which includes 100 scenes with ground truth 3D plane annotations provided by [18]. On the ScanNet++ dataset, we randomly select 30 scenes for evaluation and extract 3D plane annotations from the ground truth meshes like [18].

**Evaluation Metrics.** According to PlanarRecon [37], we evaluate the geometry quality of all reconstructed 3D planes with Chamfer Distance and F-score. Following AirPlanes [35], we also evaluate the reconstruction quality of Top-20 largest planes from the ground truth and use the metrics including Planar Fidelity, Planar Accuracy, and Planar Chamfer. To evaluate 3D plane segmentation, we use

the metrics including Variation of Information (VOI), Rand Index (RI), and Segmentation Covering (SC) like [18, 37].

**Baselines.** Since our method is purely built upon geometry cues, we mainly compare our *PlanarSplatting* with those geometry-based methods including SR+RANSAC [27], Metric3Dv2+RANSAC [10] and 2DGS+RANSAC [11]. These baselines build dense scene meshes from multi-view images at first and then extract 3D planes with the RANSAC algorithm from the reconstructed meshes. We use the RANSAC implementation provided by AirPlanes [35] for all these geometry-based baselines. Besides, we also compare our *PlanarSplatting* with some plane annotation based methods that use 2D/3D plane labels/priors in their training stage including PlanarRecon [37] and AirPlanes [35], and report their results for reference.

### 4.2. Comparisons with Baselines

**Quantitative Results.** We first evaluate our *PlanarSplatting* on the ScanNetV2 [2] dataset. Specifically, we evaluate two versions of our *PlanarSplatting*, utilizing normal maps from different pretrained models for supervision: Metric3Dv2 [10] and Omnidata [3], respectively. As shown in Tab. 1, our method achieves the best geometry performance in the metric of Chamfer and F-score compared to all baselines. To the results of Segmentation and Planar, our method outperforms geometry-based baselines in most metrics and is comparable to the state-of-the-art AirPlanes which uses plane embeddings trained on the same ScanNetV2 dataset. In Tab. 2, we show the results on the ScanNet++ [40] dataset. Benefiting from our novel design for differentiable plane rendering and

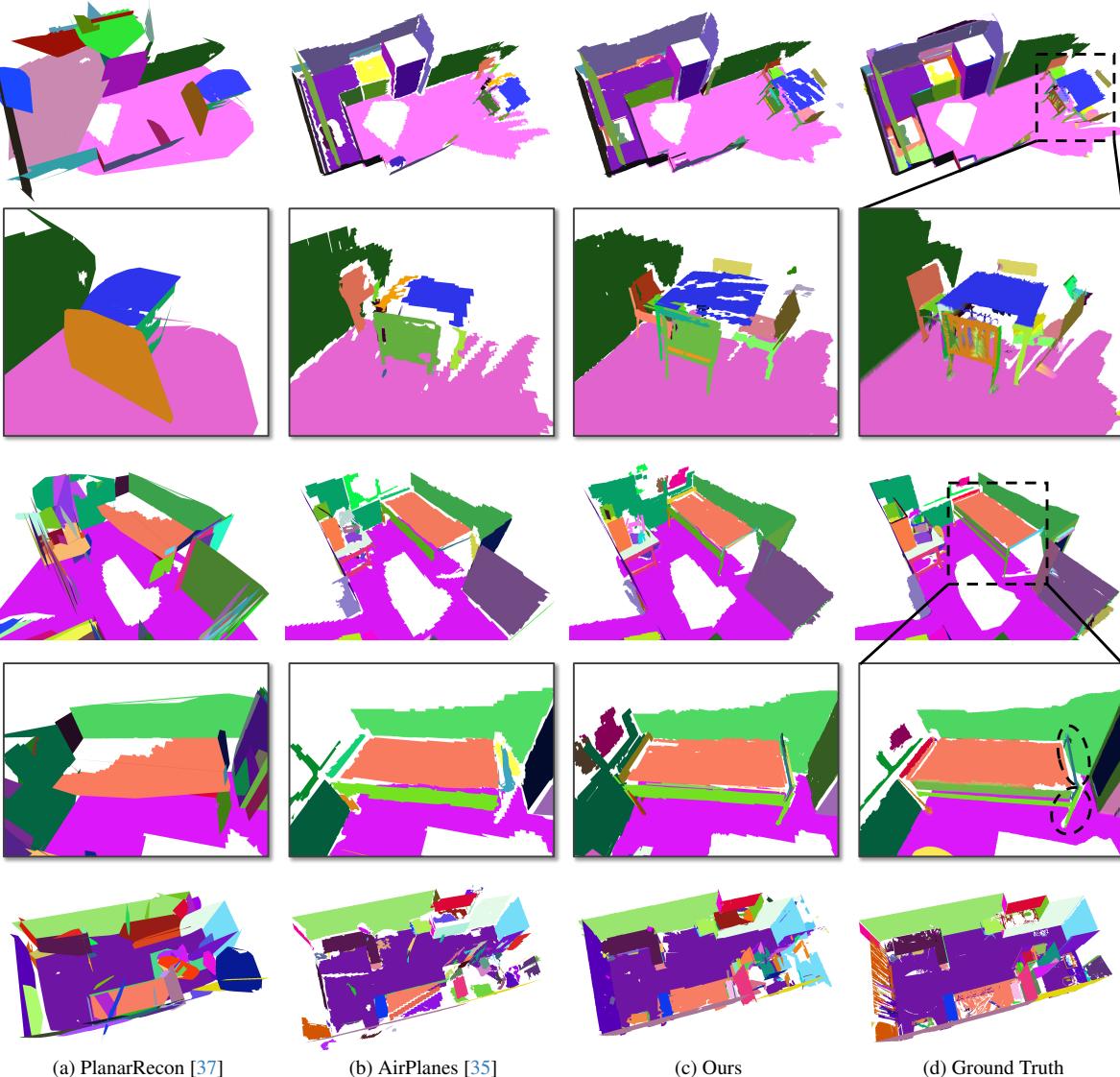


Figure 7. Qualitative comparison on the ScanNetV2 [2] (rows 1-4) and ScanNet++ [40] (last row) datasets.

optimization via monocular cues provided by modern foundation models, our *PlanarSplatting* still achieves the best Geometry performance among all baselines. Furthermore, our method outperforms both geometry-based and plane annotation based methods in most metrics of Segmentation and Planar, demonstrating the robustness and superiority of our proposed *PlanarSplatting*.

**Qualitative Results.** In Fig. 7, we show the comparisons of PlanarRecon [37], AirPlanes [35] and our *PlanarSplatting* with normal supervision of Omnidata [3]. PlanarRecon reconstructs coarse 3D planes from input multi-view images while our *PlanarSplatting* accurately reconstructs scene geometry via directly optimizing 3D plane primitives in the whole 3D space. With the help of a plane embedding model, AirPlanes achieves more semantic plane segmentation from

co-plane regions. However, as shown in the zoom-in results in Fig. 7, AirPlanes loses many geometry details in the scene. In contrast, our *PlanarSplatting* successfully reconstructs detailed structures such as the chairs and the legs of the bed, resulting in both accuracy and completeness in results. These comparisons effectively indicate the superiority of our method.

### 4.3. Ablation Studies

We verify the design of our *PlanarSplatting* on 10 scenes randomly selected from the ScanNetV2 [2] dataset. We optimize 3D plane primitives with 5,000 iterations with the normal supervision of Omnidata [3] in default.

**Plane Initialization.** We first assess the sensitivity of the plane initialization strategy of our *PlanarSplatting*. We ablate the used Metric3Dv2 [10] initialization to the sphere

Table 3. Ablation studies of the proposed *PlanarSplatting* on the ScanNetV2 dataset. ‘w’ means ‘with’ and ‘w/o’ means ‘without’.

	Chamfer ↓	F-score ↑	VOI ↓
w/ Sphere Init.	8.39	57.53	3.073
w/ Sphere Init. (30K)	<b>4.58</b>	69.91	2.522
w/o Double Radii	4.73	70.25	2.491
w/o Plane Splitting	4.69	70.30	2.477
Ours (Full)	<u>4.66</u>	<b>71.30</b>	<b>2.473</b>

Table 4. Quantitative comparison of novel view synthesis on the ScanNetV2 [2] dataset. ‘Plane’ means time for our plane optimization. ‘GS’ means time for 2D/3D Gaussian optimization. ‘#P’ means the average number of Gaussian points.

	PSNR ↑	SSIM ↑	LPIPS ↓	Avg. Time (min)			#P
				Plane	GS	Total	
3DGS [13]	24.417	0.781	0.321	-	12.2	12.2	1.27M
Ours+3DGS [13]	<b>25.471</b>	<b>0.816</b>	<b>0.296</b>	2.5	3.1	<b>5.6</b>	<b>0.37M</b>
2DGS [11]	24.766	0.796	0.323	-	14.2	14.2	0.76M
Ours+2DGS [11]	<b>25.380</b>	<b>0.813</b>	<b>0.296</b>	2.5	4.8	<b>7.3</b>	<b>0.37M</b>

initialization, in which 3D plane primitives are initialized by setting their radii as 0.05 and are placed on the bounding sphere of the scene with their normals pointing to the center of the scene. As reported in the top two rows of Tab. 3, initialization with Metric3Dv2 [10] mainly improves our method to be faster converged, while sphere initialization could also obtain promising results with more iterations.

**Plane Radii.** We then evaluate the design of double-direction plane radii (Double Radii) by replacing it with the vanilla single-direction plane radii. It means that we learn one radius at each axis of the plane primitive. As shown in Tab. 3, optimizing without Double Radii leads to a decrease in the performance of geometry and segmentation, demonstrating the effectiveness of our Double Radii.

**Plane Splitting.** As shown in the last two rows of Tab. 3, applying the Plane Splitting in optimization can further improve the performance in both geometry and segmentation.

#### 4.4. PlanarSplatting for Novel View Synthesis

Benefiting from our accurate and fast planar reconstruction, we show that our *PlanarSplatting* can seamlessly integrate with recent Gaussian Splatting methods for more efficient and better quality novel view synthesis in indoor scenes. We select two typical methods (3DGS [13] and 2DGS [11]) for evaluation and compare them with two variants including ‘Ours+3DGS’ and ‘Ours+2DGS’. Specifically, for our two variants, we directly sample points from our reconstructed 3D plane primitives to use as the initialization of these Gaussian Splatting methods. Different from the original 3DGS and 2DGS, we fix the position of 3D points and exclude the densification operation when optimizing our two variants. For the baselines of 3DGS and 2DGS, we initialize them with SfM and run with densification to achieve the best performance. We conduct the experiments

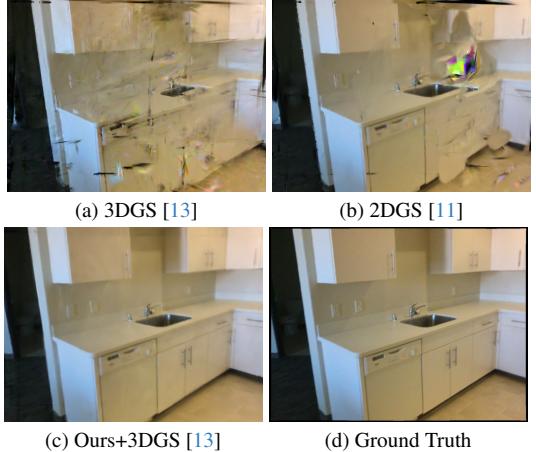


Figure 8. Comparison of novel view synthesis on the ScanNetV2 dataset [2]. With initialization from our planar reconstruction, we significantly improve the rendering result of 3DGS [13].

using the same scenes as those used in the ablation study. As shown in Tab. 4, our two variants ‘Ours+3DGS’ and ‘Ours+2DGS’ significantly outperform the original 3DGS and 2DGS in all metrics with even less optimization time and Gaussian points, demonstrating the superiority and potential of our *PlanarSplatting*. In Fig. 8, we further show that ‘Ours+3DGS’ effectively improves the rendering quality on the scene from the ScanNetV2 dataset.

#### 4.5. Limitations and Future Work

Although our *PlanarSplatting* can reconstruct the accurate indoor planar surface, it is not suitable for complex shapes such as curved surfaces. We leave this challenging problem in our future work for more flexible geometric modeling.

#### 5. Conclusion

In this paper, we present *PlanarSplatting*, a novel approach for multi-view 3D reconstruction of indoor scenes. By formulating the problem through differentiable rendering with plane splatting, we demonstrate the powerful capabilities of 3D planar representation for both accurate geometry reconstruction and compact structural scene modeling. Our efficient CUDA implementation enables fast 3D surface reconstruction, allowing comprehensive evaluation across over 100 scenes within hours using a single GPU. Furthermore, the seamless integration of *PlanarSplatting* with Gaussian Splatting significantly enhances both the quality and efficiency of indoor novel view synthesis, highlighting the broader potential of our approach for indoor scene understanding.

#### Acknowledgments

This work was supported by Ant Group Postdoctoral Programme. The authors would love to thank the anonymous reviewers and area chairs for their constructive comments.

## References

- [1] Samir Agarwala, Linyi Jin, Chris Rockwell, and David F. Fouhey. Planeformers: From sparse view planes to 3d reconstruction. In *Eur. Conf. Comput. Vis.*, pages 192–209, 2022. 3
- [2] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2432–2443, 2017. 1, 2, 5, 6, 7, 8
- [3] Ainaaz Eftekhari, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Int. Conf. Comput. Vis.*, pages 10766–10776, 2021. 2, 5, 6, 7
- [4] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Manhattan-world stereo. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1422–1429, 2009. 3
- [5] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Reconstructing building interiors from images. In *Int. Conf. Comput. Vis.*, pages 80–87, 2009.
- [6] David Gallup, Jan-Michael Frahm, and Marc Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1418–1425, 2010. 3
- [7] Abhinav Gupta, Alexei A. Efros, and Martial Hebert. Blocks world revisited: Image understanding using qualitative geometry and mechanics. In *Eur. Conf. Comput. Vis.*, pages 482–496, 2010. 2
- [8] Yuze He, Wang Zhao, Shaohui Liu, Yubin Hu, Yushi Bai, Yu-Hui Wen, and Yongjin Liu. Alphatables: A generic plane representation for 3d planar reconstruction from monocular videos. In *Adv. Neural Inform. Process. Syst.*, 2024. 3
- [9] Manuel Hofer, Michael Maurer, and Horst Bischof. Efficient 3d scene abstraction using line segments. *Comput. Vis. Image Underst.*, 157:167–178, 2017. 2
- [10] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(12):10579–10596, 2024. 2, 4, 5, 6, 7, 8
- [11] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH Conference Papers*, page 32, 2024. 2, 3, 5, 6, 8
- [12] Linyi Jin, Shengyi Qian, Andrew Owens, and David F. Fouhey. Planar surface reconstruction from sparse views. In *Int. Conf. Comput. Vis.*, pages 12971–12980, 2021. 2, 3
- [13] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139:1–139:14, 2023. 2, 3, 5, 8
- [14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Int. Conf. Learn. Represent.*, 2015. 5
- [15] Pierre-Alain Langlois, Alexandre Boulch, and Renaud Marlet. Surface reconstruction from 3d line segments. In *Int. Conf. 3D Vis.*, pages 553–563, 2019. 3
- [16] David C. Lee, Martial Hebert, and Takeo Kanade. Geometric reasoning for single image structure recovery. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2136–2143, 2009. 2, 3
- [17] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. Planenet: Piece-wise planar reconstruction from a single RGB image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2579–2588, 2018. 3
- [18] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4450–4459, 2019. 2, 3, 6
- [19] Jiachen Liu, Pan Ji, Nitin Bansal, Changjiang Cai, Qingan Yan, Xiaolei Huang, and Yi Xu. Planemvs: 3d plane reconstruction from multi-view stereo. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 8655–8665. IEEE, 2022. 2
- [20] Shaohui Liu, Yifan Yu, Rémi Pautrat, Marc Pollefeys, and Viktor Larsson. 3d line mapping revisited. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 21445–21455. IEEE, 2023. 2
- [21] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhöfer, Yaser Sheikh, and Jason M. Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4):59:1–59:13, 2021. 3
- [22] Branislav Micusík, Horst Wildenauer, and Markus Vincze. Towards detection of orthogonal planes in monocular images of indoor environments. In *Int. Conf. on Robot. and Auto.*, pages 999–1004, 2008. 3
- [23] Tom Monnier, Jake Austin, Angjoo Kanazawa, Alexei A. Efros, and Mathieu Aubry. Differentiable blocks world: Qualitative 3d decomposition by rendering primitives. In *Adv. Neural Inform. Process. Syst.*, 2023. 2, 3
- [24] Áron Monszpart, Nicolas Mellado, Gabriel J. Brostow, and Niloy J. Mitra. Rapter: rebuilding man-made scenes with regular arrangements of planes. *ACM Trans. Graph.*, 34(4):103:1–103:12, 2015. 3
- [25] Jann Poppinga, Narunas Vaskevicius, Andreas Birk, and Kaustubh Pathak. Fast plane detection and polygonalization in noisy 3d range images. In *Int. Conf. Intell. Robots and Syst.*, pages 3378–3383, 2008. 2, 3
- [26] Yiming Qian, Srikumar Ramalingam, and James H. Elder. LS3D: single-view gestalt 3d surface reconstruction from manhattan line segments. In *Asian Conf. Comput. Vis.*, pages 399–416, 2018. 3
- [27] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. Simplerecon: 3d reconstruction without 3d convolutions. In *Eur. Conf. Comput. Vis.*, pages 1–19, 2022. 6
- [28] Jingjia Shi, Shuaifeng Zhi, and Kai Xu. Planerecr: Unified query learning for 3d plane recovery from a single view. In *Int. Conf. Comput. Vis.*, pages 9343–9352, 2023. 3
- [29] Sudipta N. Sinha, Drew Steedly, and Richard Szeliski. Piecewise planar stereo for image-based rendering. In *Int. Conf. Comput. Vis.*, 2009. 3

- [30] Christiane Sommer, Yumin Sun, Leonidas J. Guibas, Daniel Cremers, and Tolga Birdal. From planes to corners: Multi-purpose primitive detection in unorganized 3d point clouds. *IEEE Robotics Autom. Lett.*, 5(2):1764–1771, 2020. [2](#), [3](#)
- [31] Bo Sun and Philippas Mordohai. Oriented point sampling for plane detection in unorganized point clouds. In *Int. Conf. on Robot. and Auto.*, pages 2917–2923, 2019. [3](#)
- [32] Bin Tan, Nan Xue, Song Bai, Tianfu Wu, and Gui-Song Xia. Planetr: Structure-guided transformers for 3d plane recovery. In *Int. Conf. Comput. Vis.*, pages 4166–4175, 2021. [3](#)
- [33] Bin Tan, Nan Xue, Tianfu Wu, and Gui-Song Xia. NOPE-SAC: neural one-plane RANSAC for sparse-view planar 3d reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 45(12):15233–15248, 2023. [2](#), [3](#)
- [34] Yifan Wang, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Trans. Graph.*, 38(6):230:1–230:14, 2019. [3](#)
- [35] Jamie Watson, Filippo Aleotti, Mohamed Sayed, Zawar Qureshi, Oisin Mac Aodha, Gabriel J. Brostow, Michael Firman, and Sara Vicente. Airplanes: Accurate plane estimation via 3d-consistent embeddings. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5270–5280, 2024. [1](#), [3](#), [6](#), [7](#)
- [36] Jianxiong Xiao and Yasutaka Furukawa. Reconstructing the world’s museums. *Int. J. Comput. Vis.*, 110(3):243–258, 2014. [3](#)
- [37] Yiming Xie, Matheus Gadelha, Fengting Yang, Xiaowei Zhou, and Huaizu Jiang. Planarrecon: Realtime 3d plane detection and reconstruction from posed monocular videos. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6209–6218, 2022. [1](#), [2](#), [3](#), [6](#), [7](#)
- [38] Sheng Xu, Ruisheng Wang, Hao Wang, and Ruigang Yang. Plane segmentation based on the optimal-vector-field in lidar point clouds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(11):3991–4007, 2021. [2](#)
- [39] Nan Xue, Bin Tan, Yuxi Xiao, Liang Dong, Gui-Song Xia, Tianfu Wu, and Yujun Shen. NEAT: distilling 3d wireframes from neural attraction fields. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 19968–19977. IEEE, 2024. [2](#)
- [40] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Int. Conf. Comput. Vis.*, pages 12–22, 2023. [2](#), [5](#), [6](#), [7](#)
- [41] Wei Yin, Chi Zhang, Hao Chen, Zhipeng Cai, Gang Yu, Kaixuan Wang, Xiaozhi Chen, and Chunhua Shen. Metric3d: Towards zero-shot metric 3d prediction from A single image. In *Int. Conf. Comput. Vis.*, 2023. [2](#)
- [42] Zehao Yu, Jia Zheng, Dongze Lian, Zihan Zhou, and Shenghua Gao. Single-image piece-wise planar 3d reconstruction via associative embedding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1029–1037, 2019. [2](#), [3](#)
- [43] Jiahui Zhang, Jinfu Yang, Fuji Fu, and Jiaqi Ma. Planeac: Line-guided planar 3d reconstruction based on self-attention and convolution hybrid model. *Pattern Recog.*, 153:110519, 2024. [3](#)
- [44] Yanshu Zhang, Shichong Peng, Alireza Moazeni, and Ke Li. Papr: Proximity attention point rendering. In *Adv. Neural Inform. Process. Syst.*, 2023. [3](#)