

PlanarNeRF: Online Learning of Planar Primitives with Neural Radiance Fields

Zheng Chen^{1*} Qingan Yan² Huangying Zhan² Changjiang Cai² Xiangyu Xu²
 Yuzhong Huang³ Weihan Wang⁴ Ziyue Feng⁵ Lantao Liu¹ Yi Xu²

¹Indiana University ²OPPO US Research Center ³University of Southern California
⁴Stevens Institute of Technology ⁵Clemson University

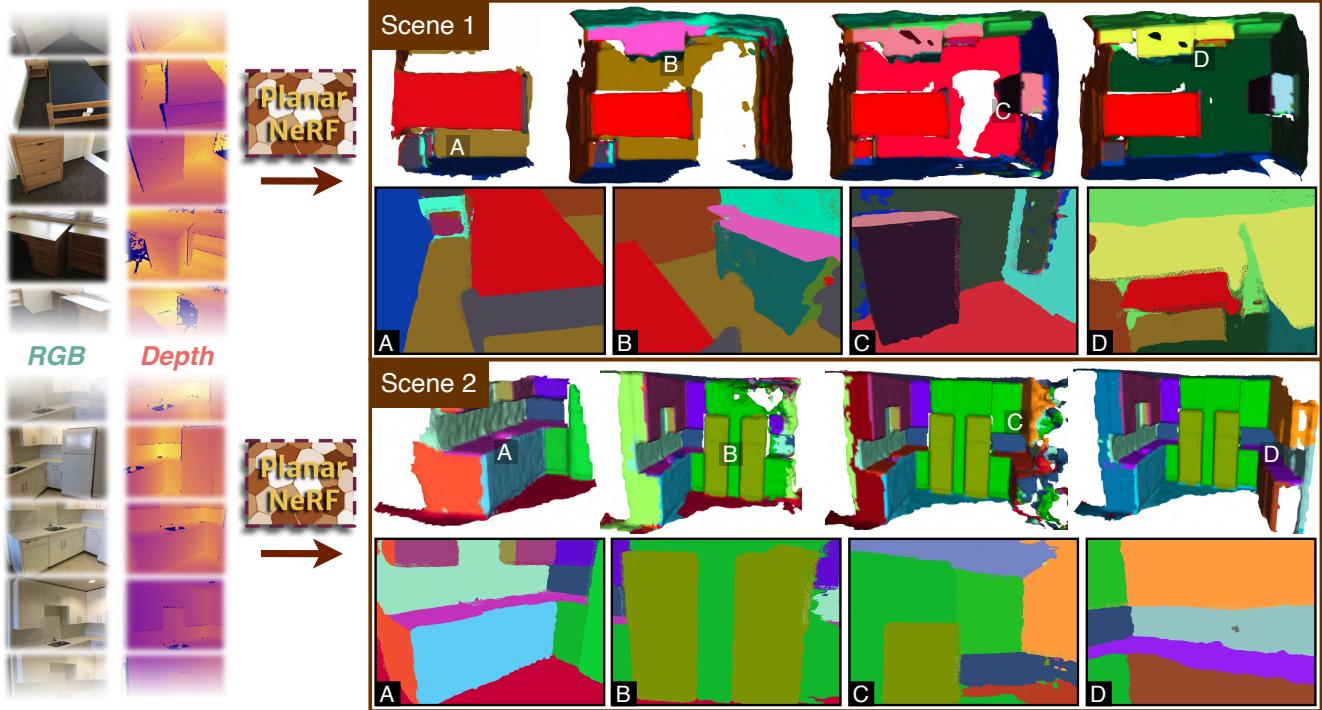


Figure 1. We introduce **PlanarNeRF**, a framework designed to detect dense 3D planar primitives from monocular RGB and depth sequences. The method learns plane primitives in an online fashion while drawing knowledge from both scene appearance and geometry. Displayed are outcomes from two distinct scenes (Best viewed in color). Each case exhibits two rows: the top row visualizes the reconstruction progress, while the bottom row showcases **rendered** 2D segmentation images at different time steps.

Abstract

Identifying spatially complete planar primitives from visual data is a crucial task in computer vision. Prior methods are largely restricted to either 2D segment recovery or simplifying 3D structures, even with extensive plane annotations. We present PlanarNeRF, a novel framework capable of detecting dense 3D planes through online learning. Drawing upon the neural field representation, PlanarNeRF brings three major contributions. First, it enhances 3D plane detection with concurrent appearance and geo-

try knowledge. Second, a lightweight plane fitting module is proposed to estimate plane parameters. Third, a novel global memory bank structure with an update mechanism is introduced, ensuring consistent cross-frame correspondence. The flexible architecture of PlanarNeRF allows it to function in both 2D-supervised and self-supervised solutions, in each of which it can effectively learn from sparse training signals, significantly improving training efficiency. Through extensive experiments, we demonstrate the effectiveness of PlanarNeRF in various scenarios and remarkable improvement over existing works.

*This work was done when Zheng Chen was an intern at OPPO US Research Center.

1. Introduction

Planar primitives stand out as critical elements in structured environments such as indoor rooms and urban buildings. Capturing these planes offers a concise and efficient representation, and holds great impact across a spectrum of applications, including Virtual Reality, Augmented Reality, and robotic manipulation, *etc.* Beyond serving as a fundamental modeling block, planes are widely used in many data processing tasks, including object detection [27], registration [17, 51], pose estimation [4], and SLAM [14, 15, 50].

Extensive efforts have been dedicated to exploring different plane detection methodologies. Nevertheless, notable limitations persist within current approaches. First, many of them produce only isolated per-view 2D plane segments [20, 36, 48]. Although certain methods [1, 13, 37] establish correspondences across sparse (typically two) views, they still lack spatial consistency, leading to incomplete scene representations. Recently, an end-to-end deep model [43] was introduced for 3D plane detection; however, its outcomes tend to oversimplify scene structures. Moreover, the aforementioned models heavily rely on extensive annotations — pose, 2D planes, and 3D planes — consequently limiting their generalization capabilities. While fitting-based methods like [26, 28] operate without annotations, they are typically restricted to offline detection, involving heavy iterations and posing computational challenges.

We propose PlanarNeRF, an online 3D plane detection framework (Fig. 1) that overcomes the above limitations. Specifically, we extend the neural field representation to regress plane primitives with both appearance and geometry for more complete and accurate results. The framework’s efficient network design allows for dual operational modes: **PlanarNeRF-S**, a supervised mode leveraging sparse 2D plane annotations; and **PlanarNeRF-SS**, a self-supervised mode that extracts planes directly from depth images. In PlanarNeRF-SS, we design a lightweight plane fitting module to estimate plane parameters from a highly sparse set of sampled points in each iteration. Then a global memory bank is maintained to ensure consistent tracking of plane instances across different views and to generate labels for the sparse points. The inherent multi-view consistency and smoothness of NeRF facilitate the propagation of sparse labels. Our key contributions are as follows:

- Introduction of PlanarNeRF, a novel approach for detecting dense 3D planar primitives using online learning, integrating insights from both the scene’s appearance and its geometric attributes.
- Development of a lightweight plane fitting module, estimating plane parameters for sparsely sampled points.
- Establishment of a novel global memory bank module with an update mechanism, ensuring consistent tracking of plane instances and facilitating plane label generation.
- Comprehensive evaluation of PlanarNeRF’s performance

through extensive experiments, demonstrating notable superiority over existing methodologies.

2. Related Work

Single View Plane Detection. Many studies focus on directly segmenting planes from individual 2D images. PlaneNet [19] was among the first to encapsulate the detection process within an end-to-end framework directly from a single image. Conversely, PlaneRecover [45] introduces an unsupervised approach for training plane detection networks using RGBD data. Meanwhile, PlaneRCNN [20] capitalizes on Mask-RCNN’s [12] generalization capability to identify planar segmentation in input images, simultaneously regressing 3D plane normals from fixed normal anchors. In contrast, PlaneAE [48] assigns each pixel to an embedding feature space, subsequently grouping similar features through mean-shift algorithms. Additionally, PlaneTR [36] harnesses line segment constraints and employs Transformer decoders [3] to enhance performance further. Despite these advancements, the detected plane instances lack consistency across different frames.

Multi-view Plane Detection. SparsePlanes [13] detects plane segments in two views and uses a deep neural network architecture with an energy function for correspondence optimization. PlaneFormers [1], eschewing handcrafted energy optimization, introduces a Transformer architecture to directly predict plane correspondences. NOPE-SAC [37] associates two-view camera pose estimation with plane correspondence in the RANSAC paradigm while enabling end-to-end learning. PlaneMVS [21] unifies plane detection and plane MVS with known poses and facilitates mutual benefits between these two branches. Although multi-view inputs enhance segmentation consistency, they still lack a global association, preventing the construction of complete scenarios. PlanarRecon [43] progressively fuses multi-view features and extracts 3D plane geometries from monocular videos in an end-to-end fashion, bypassing per-view segmentation. Nonetheless, it necessitates 3D ground truth prerequisites and tends to oversimplify the resulting output.

Dense 3D Fitting. For the 3D plane clustering, depth information typically serves as essential input [8, 31]. Efficient RANSAC [28] progressively estimates plane primitives within reconstructed dense point clouds, employing the consensus sampling paradigm [10]. [26] grows plane segments within designated seed regions. GoCoPP [47] explores optimal plane parameters and assigns discrete labels to individual 3D points, referring to an exploration mechanism. While these methods function effectively without annotations, their computational demands are substantial. Additionally, approaches like [16, 29, 44] adeptly fit well-shaped point clouds to geometric primitives under supervision but grapple with generalization challenges.

Neural Scene Reconstruction. The groundbreaking

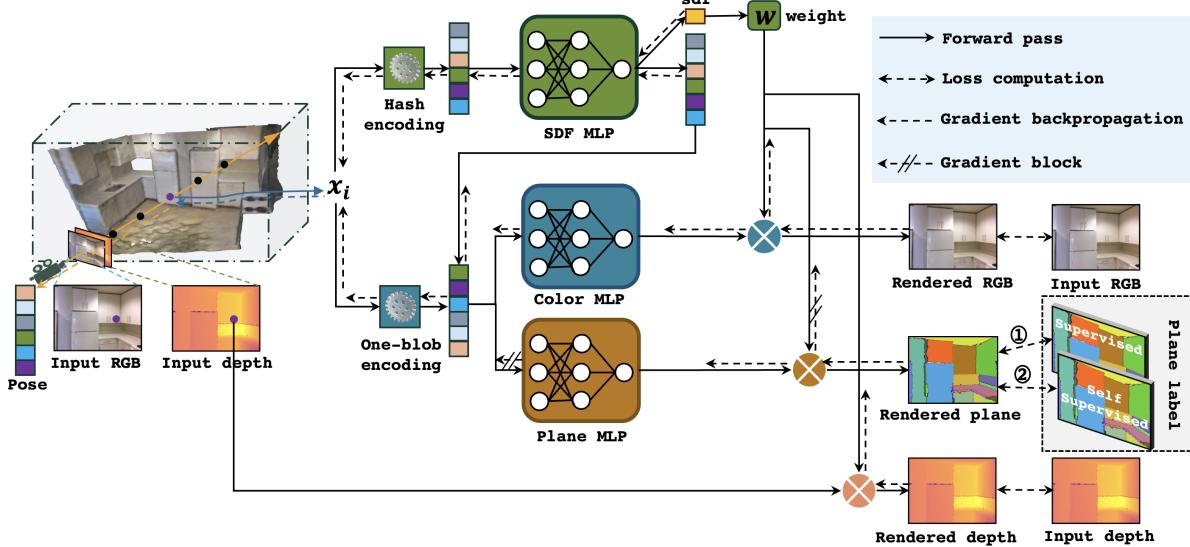


Figure 2. **Overview of PlanarNeRF.** PlanarNeRF processes monocular RGB and depth image sequences, enabling online pose estimation. It offers two modes: ① PlanarNeRF-S (supervised) with 2D plane annotations, and ② PlanarNeRF-SS (self-supervised) without annotations. The framework includes an efficient plane fitting module and a global memory bank for consistent plane labeling.

NeRF [23] introduced an innovative solution for 3D environment representation, upon which numerous studies have demonstrated outstanding performance in scene reconstruction [2, 11, 18, 32, 39–42, 46, 49]. In particular, Nice-SLAM [52] builds a series of learnable grid architectures serving as hierarchical feature encoders and conducts pose optimization and dense mapping. Nicer-SLAM [53] refines this approach by reducing the necessity for depth images and achieves comparable reconstruction results. Co-SLAM [38] adopts hash maps instead of grids as the feature container and introduces coordinate and parametric encoding for expedited convergence and querying.

3. Methodology

3.1. Preliminaries

NeRF (Neural Radiance Fields) [23] conceptualizes a scene as a continuous function, typically represented by a multi-layer perceptron (MLP). This function, defined as $F(\mathbf{x}, \mathbf{v}) \mapsto (\mathbf{c}, \sigma)$, maps a 3D point \mathbf{x} and a 2D viewing direction \mathbf{v} to the corresponding RGB color \mathbf{c} and volume density σ . Notably, both color and density predictions share the same network. For a ray $R(t) = \mathbf{o} + t\mathbf{v}$ with origin \mathbf{o} , the rendered color $\mathbf{C}(R)$ is obtained by integrating points along the ray via volume rendering:

$$\mathbf{C}(R) = \int_{t_n}^{t_f} T(t)\sigma(R(t))\mathbf{c}(R(t), \mathbf{v}) dt, \quad (1)$$

where $T(t) = \exp\left(-\int_{t_n}^t \sigma(R(s)) ds\right)$ is the accumulated transmittance from the near bound t_n to t , and t_f is the far bound. To enhance the learning performance on high-frequency details, NeRF incorporates positional encoding of 3D coordinates as input. The model is then trained to

minimize the discrepancy between observed and predicted color values.

Recent advancements in NeRF have enhanced rendering and reconstruction by modifying the original framework. Key improvements include using Signed Distance Fields (SDF) for predictions [39], employing separate neural networks for RGB and geometry with augmented inputs [34], recalculating weights in the rendering equation based on SDF [2], and adopting hash and one-blob encoding for positional data [24]. Additionally, depth rendering is used to improve geometry learning [49]. In PlanarNeRF, we incorporate these recent modifications, resulting in an updated and optimized color rendering equation:

$$\mathbf{C}(R) = \frac{1}{\sum_{i=1}^M w_i} \sum_{i=1}^M w_i \mathbf{c}_i(R(t), \mathbf{v}), \quad (2)$$

where M is the number of sampled points along the ray, and w_i is the weight computed based on SDF: $w_i = \sigma\left(\frac{s_i}{tr}\right) \sigma\left(-\frac{s_i}{tr}\right)$. Here, s_i is the predicted SDF values along the ray; tr is a predefined truncation threshold for SDF; and $\sigma(\cdot)$ is the sigmoid function. Similar to Eq. (2), the rendering equation for depth is:

$$D(R) = \frac{1}{\sum_{i=1}^M w_i} \sum_{i=1}^M w_i d_{p_i}(R(t), \mathbf{v}), \quad (3)$$

where d_{p_i} is the depth of sampled points along the ray.

3.2. Framework Overview

The overview of PlanarNeRF is depicted in Fig. 2. Alongside SDF and color rendering branches, an additional plane rendering branch (Sec. 3.3) is introduced to map 3D coordinates to 2D plane instances, utilizing appearance and geometry prior. The plane MLP and color MLP share the

same input, which combines a one-blob encoded 3D coordinate and a learned SDF feature vector. In PlanarNeRF-S, while consistent 2D plane annotations are requisite, they are often unavailable in real-world scenarios, where manual labeling for plane instance segmentation is costly. To tackle this challenge, we propose a lightweight plane fitting module (Sec. 3.4) to estimate plane parameters and a global memory bank (Sec. 3.5) to track consistent planes and produce plane labels. During the training phase, gradient back-propagation from the plane branch to the SDF is blocked to prevent potential negative impacts on geometry learning, with further qualitative analysis provided in Sec. 4.4.

3.3. Plane Rendering Learning

Similar to Eq. (2) and Eq. (3), we propose the rendering equation for planes as:

$$\mathbf{P}(R) = \frac{1}{\sum_{i=1}^M w_i} \sum_{i=1}^M w_i \mathbf{p}_i(R(t), \mathbf{v}), \quad (4)$$

where \mathbf{p}_i is the plane classification probability vector of sampled points along the ray.

Conventionally, instance segmentation learning has been approached using either anchor boxes [12, 20] or a bipartite matching [5, 6, 30, 36]. Anchor boxes-based methods often involve complex pipelines with heuristic designs. In contrast, bipartite matching-based methods establish an optimized correspondence between predictions and ground truths before computing the loss. The instance segmentation loss based on bipartite matching can be expressed as:

$$\mathcal{L}_{ins} = -\frac{1}{Q} \sum_{q=1}^Q \sum_{c=1}^C y_c \log \hat{y}_c, \quad (5)$$

where Q is the number of pixels; C is the number of classes. y_c is the c^{th} element in the ground truth label \mathbf{y} , and $y_c = \mathbb{1}_{\{c=m(\hat{\mathbf{y}}, \mathbf{y})\}}$, where $m(\cdot)$ is the matching function, and the assignment cost can be given by the intersection over union of each instance between the prediction and the ground truth. \hat{y}_c is the c^{th} element in the prediction probability vector $\hat{\mathbf{y}}$. Using bipartite matching stems from the inherent discrepancies in index values between instance segmentation predictions and the ground truth labels. We only need to match the segmented area and distinguish one instance from another.

In contrast to the instance segmentation methods previously discussed, PlanarNeRF employs a distinct approach for plane instance segmentation. We adopt a **fixed matching** technique, akin to that used in semantic segmentation, to compute the segmentation loss. This method is chosen because our primary objective is to learn consistent 3D plane instances. Consequently, it is imperative that the rendered 2D plane instance segmentation remains consistent across different frames. To uphold this consistency, we ensure that the indices in the predictions strictly match the val-

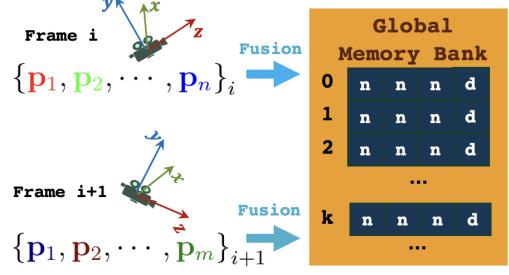


Figure 3. The global memory bank is updated with each new batch of estimated planes. Notations are explained in Sec. 3.4.

ues provided in the ground truth during loss computation.

3.4. Lightweight Plane Fitting

Before we produce the plane labels, we need to extract plane parameters based on depth images. To achieve this, we propose a lightweight plane fitting module, which can be treated as a function f that builds a map from input points to planes:

$$\{\{\mathbf{p}_i\}_{i=1}^N, \{PO_i\}_{i=1}^N\} = f_\Theta(PO), \quad (6)$$

where N is the total number of plane instances estimated in one batch; \mathbf{p} represents the plane parameters: $\mathbf{p}_i = [n_i^x, n_i^y, n_i^z, d_i]^T$ which form a plane equation $n_i^x \cdot x + n_i^y \cdot y + n_i^z \cdot z - d_i = 0$; PO represents the input points set from depth images and PO_i represent the set of points belonging to the i^{th} plane instance, and $PO_i = \{\mathbf{po}_j\}_{j=0}^{n_j}$, where $\mathbf{po}_j = [x_j, y_j, z_j]^T$; n_j is the number of points in the set; Θ is the set of hyperparameters based on Efficient RANSAC [28] and $\Theta = \{n_{min}, \epsilon, \epsilon_{cluster}, \tau_n, \hat{p}\}$, where:

- n_{min} : the minimum number of points belonging to one plane instance.
- ϵ : the absolute maximum tolerance Euclidean distance between a point and a plane.
- $\epsilon_{cluster}$: controls the connectivity of the points covered by an estimated plane. Large values usually lead to undersegmentation while small values usually lead to oversegmentation.
- τ_n : threshold for the normal deviation between the one point's normal and the estimated plane normal.
- \hat{p} : probability of missing the largest plane.

To achieve satisfying plane fitting performance for sparse points, hyperparameters Θ have to be carefully tuned. Details about tuning decisions are shown in Sec. 4.4.

3.5. Global Memory Bank

Plane estimations using Eq. (6) in different iterations are independent with each other. This lacks consistency as new data constantly comes in. We propose a novel global memory bank to maintain the plane parameters across different frames (Fig. 3). The key part of maintaining the bank is the similarity measure between two planes. According to Eq. (6), we are able to obtain the plane vector for each

Table 1. Comparisons for 3D geometry, memory and speed on ScanNet. **Red** for the best and **green** for the second best (same for the following).

Method	Val. Set	Acc. ↓	Comp. ↓	Recall ↑	Prec. ↑	F-score ↑	Mem. (GB) ↓	Time (ms) ↓
NeuralRecon [35] + Seq-RANSAC [10]	[25]	0.144	0.128	0.296	0.306	0.296	4.39	586
Atlas [25] + Seq-RANSAC [10]		0.102	0.190	0.316	0.348	0.331	25.91	848
ESTDepth [22] + PEAC [9]		0.174	0.135	0.289	0.335	0.304	5.44	101
PlanarRecon [43]		0.154	0.105	0.355	0.398	0.372	4.43	40
PlanarNeRF-SS (Ours)		0.059	0.073	0.661	0.651	0.654	4.09[†]	328 [†] / 131 ^{†*}
PlaneAE [48]	[48]	0.128	0.151	0.330	0.262	0.290	6.29	32
PlanarRecon [43]		0.143	0.098	0.372	0.412	0.389	4.43	40
PlanarNeRF-SS (Ours)		0.063	0.078	0.674	0.657	0.665	4.09[†]	328 [†] / 131 ^{†*}

[†] Since our method is an online learning method, we report the *memory* and *time* used during **training**. Others are offline-trained, hence **inference**.

* For PlanarNeRF-S. SS and S share the same geometry learning and GPU memory. The time gap is mainly caused by the self-supervised plane label generation (on CPU) in SS.

plane instance. An intuitive way to compare two vectors is to compute the Euclidean distance, $\|\mathbf{p}_1 - \mathbf{p}_2\|_2$. However, this way fails for planes because each element in the plane parameter vector has a physical meaning. A reasonable way to compare the distance between two plane parameters is:

$$dist'(\mathbf{p}_1, \mathbf{p}_2) = 1 - \left| \frac{\langle \mathbf{n}_1, \mathbf{n}_2 \rangle}{\|\mathbf{n}_1\| \|\mathbf{n}_2\|} \right| + |d_1 - d_2|, \quad d_1, d_2 \in \mathbb{R}_{\geq 0}, \quad (7)$$

where $\mathbf{p}_1 = [\mathbf{n}_1, d_1]^T$, $\mathbf{p}_2 = [\mathbf{n}_2, d_2]^T$. All offset values must be non-negative because a plane parameter vector and its negative version describe the same plane spatially, ignoring the normal orientations.

Unfortunately, Eq. (7) works well as a similarity measure but it is too sensitive to the estimation noises. Directly comparing two plane vectors lacks the robustness to the noises. To tackle this issue, we propose to use a simple yet robust way to compute the similarity measure. We observe in Eq. (6) that the results also inform us of the identity of each point, meaning there are two representations for one plane — the plane parameters (\mathbf{p}_i) or the points ($PO_i = \{\mathbf{p}_j\}_{j=0}^{n_j}$) belonging to the plane instance. The new similarity measure is based on the distance between **points to the plane**. Assume we use \mathbf{p}_1 to represent one plane and PO_2 for another, then we can have:

$$dist(\mathbf{p}_1, \mathbf{p}_2) = \frac{1}{n_j} \sum_{j=1}^{n_j} \frac{|n_1^x \cdot x_2 + n_1^y \cdot y_2 + n_1^z \cdot z_2 - d_1|}{((n_1^x)^2 + (n_1^y)^2 + (n_1^z)^2)^{\frac{1}{2}}}. \quad (8)$$

If a new plane is found highly similar to one of the plane vectors inside the bank, *i.e.* $dist(\mathbf{p}_{new}, \mathbf{p}_{bank}) < \tau_{dist}$, where τ_{dist} is the distance threshold for decisions, then we return the index of \mathbf{p}_{bank} (denoted by k , see Fig. 3) in the bank as the index annotation for the sampled points belonging to the plane instance \mathbf{p}_{new} . Otherwise, we add the \mathbf{p}_{new} into the bank. The plane label is given by $y_c = 1_{\{c=k\}}$ (see Eq. (5)). If PlanarNeRF-S is used, then y_c is assumed to be known.

To further increase the robustness of the global memory bank, we use the Exponential Moving Average (EMA) to update the plane parameters stored in the bank if the highly

Table 2. 3D plane instance segmentation comparison on ScanNet.

Method	VOI ↓	RI ↑	SC ↑
NeuralRecon [35] + Seq-RANSAC [10]	8.087	0.828	0.066
Atlas [25] + Seq-RANSAC [10]	8.485	0.838	0.057
ESTDepth [22] + PEAC [9]	4.470	0.877	0.163
PlanarRecon [43]	3.622	0.897	0.248
PlanarNeRF-SS (Ours)	2.940	0.922	0.237
PlanarNeRF-S (Ours)	2.737	0.937	0.251
PlaneAE [48]	4.103	0.908	0.188
PlanarRecon [43]	3.622	0.898	0.247
PlanarNeRF-SS (Ours)	2.952	0.928	0.235
PlanarNeRF-S (Ours)	2.731	0.940	0.252

similar plane in the bank is found:

$$\mathbf{p}_{bank} = \psi \mathbf{p}_{new} + (1 - \psi) \mathbf{p}_{bank}, \quad (9)$$

where ψ is the EMA coefficient. Note that before the update using EMA, the offset values must satisfy the constraint in Eq. (7). More details about producing plane labels using global memory bank can be seen in Algorithm 1 in the supplementary material.

4. Experiments

4.1. Baselines and Evaluation Metrics

PlanarNeRF has two working modes: **PlanarNeRF-S** where 2D plane annotations are used; and **PlanarNeRF-SS** where no annotations are used. We compare our method with four types of approaches: (1) Single view plane recovering [48]; (2) Multi-view depth estimation [22] with depth based plane detection [9]; (3) Volume-based 3D reconstruction [35] with Sequential RANSAC [10]; and (4) Learning-based 3D planar reconstruction [43].

Following the baseline work [43], we evaluate the performance of our method in terms of both geometry as well as plane instance segmentation. More specifically, for geometry evaluation, we use five metrics [25]: Completeness; Accuracy; Recall; Precision; and F-score. For plane instance segmentation, we use three metrics [20]: Rand Index (RI); Variation of Information (VOI); and Segmentation Covering (SC).

Table 3. Ablation studies for similarity measurement.

Method	VOI ↓	RI ↑	SC ↑
Raw plane param. [◦]	3.368	0.821	0.132
Corrected plane param. (Eq. (7))	3.017	0.829	0.200
Points-to-plane dist. (Eq. (8))	2.833	0.857	0.319

[◦] Directly applying Euclidean distance to raw plane parameters.

4.2. Datasets and Implementations

Our experiments are conducted using the ScanNetv2 dataset [7]. This dataset is comprised of RGB-D video sequences captured with a mobile device across 1,613 different indoor scenes. Due to the lack of ground truth data in the test set, following the previous work [43], we adopt the approach used by PlaneRCNN [20], creating 3D plane labels for both training and validation datasets. To be consistent with the previous work, we also assess our method’s performance on two distinct validation sets, which are differentiated by the scene splits previously employed in works [25, 48].

Besides ScanNetv2, we test our method on two additional datasets: Replica [33] and Synthetic scenes from NeuralRGBD [2]. As baselines lack reported results on these datasets, we present only our model’s outcomes. Detailed information about these datasets is available in the supplementary material. We employ Co-SLAM [38] as our backbone, with further implementation details provided in the supplementary material.

4.3. Qualitative Results

We show **qualitative** comparisons between our method and all the baselines in Fig. 4, where the results of two scenes in ScanNet are presented. Different color represents different plane instance. Note that the colors in predictions do not necessarily match the ones in the ground truths. PlaneAE is able to reconstruct the single-view planes but fails to organize them in 3D space consistently. ESTDepth + PEAC is better than PlaneAE but still suffers from a lack of consistency. NeuralRecon + Seq-RANSAC can produce good plane estimations but the geometry is poor and therefore diminishes the performance of instance segmentation. PlanarRecon can generate consistent and compact 3D planes but the results are oversimplified and many details of the rooms are missed. We can easily see that the results of our method are significantly superior to others in terms of both geometry and instance segmentation. PlanarNeRF-S can generate plane instance segmentation **highly close** to the ground truth when only 2D plane annotations are used. PlanarNeRF-SS also shows a high-standard segmentation quality even though no any annotations are used. If we consider a comparison in the space of plane parameters, *i.e.* planes sharing highly similar parameters are classified as one plane instance, our PlanarNeRF-SS gains more credits.

We also present **quantitative** comparisons for geometry quality (Table 1) and instance segmentation (Table 2). From Table 1, we can see that our method achieves systematic superiority to others in all geometry metrics with very

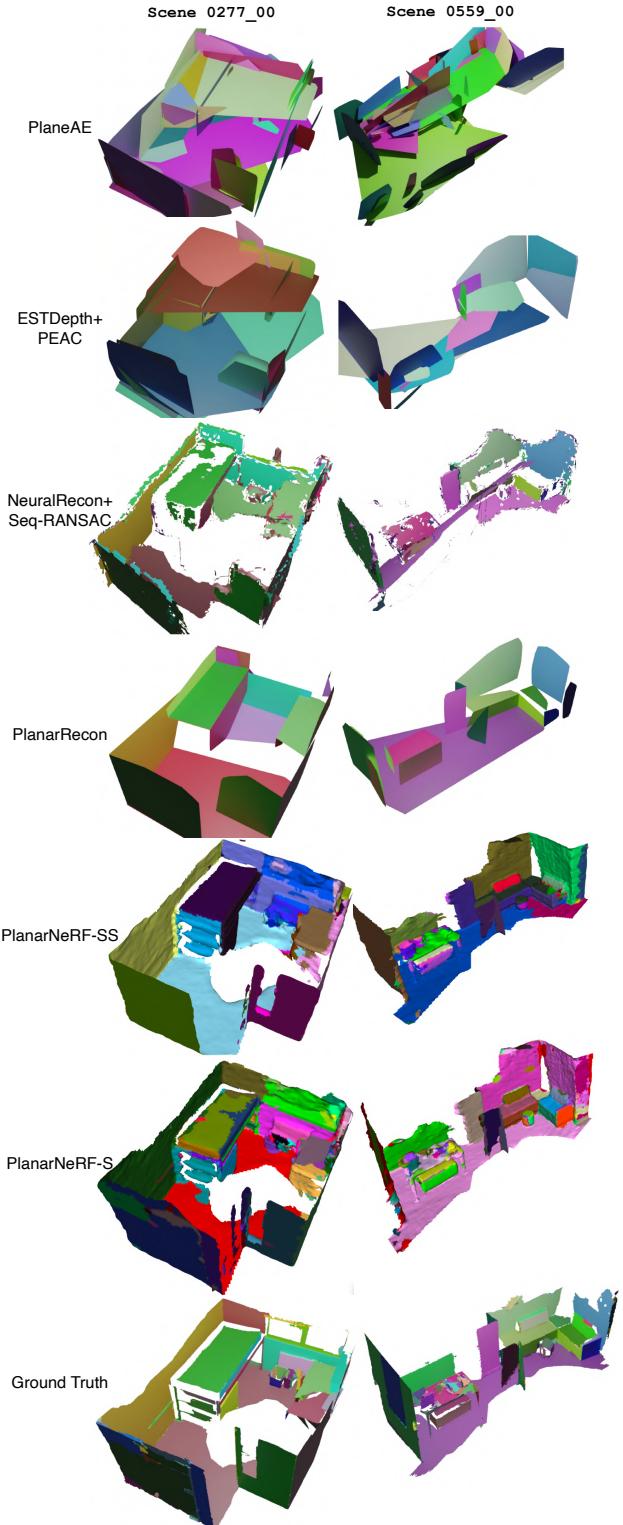


Figure 4. Qualitative comparison of PlanarNeRF and baselines on ScanNet.

low GPU memory consumption. PlanarNeRF is not as fast as PlanarRecon and ESTDepth+PEAC because our method

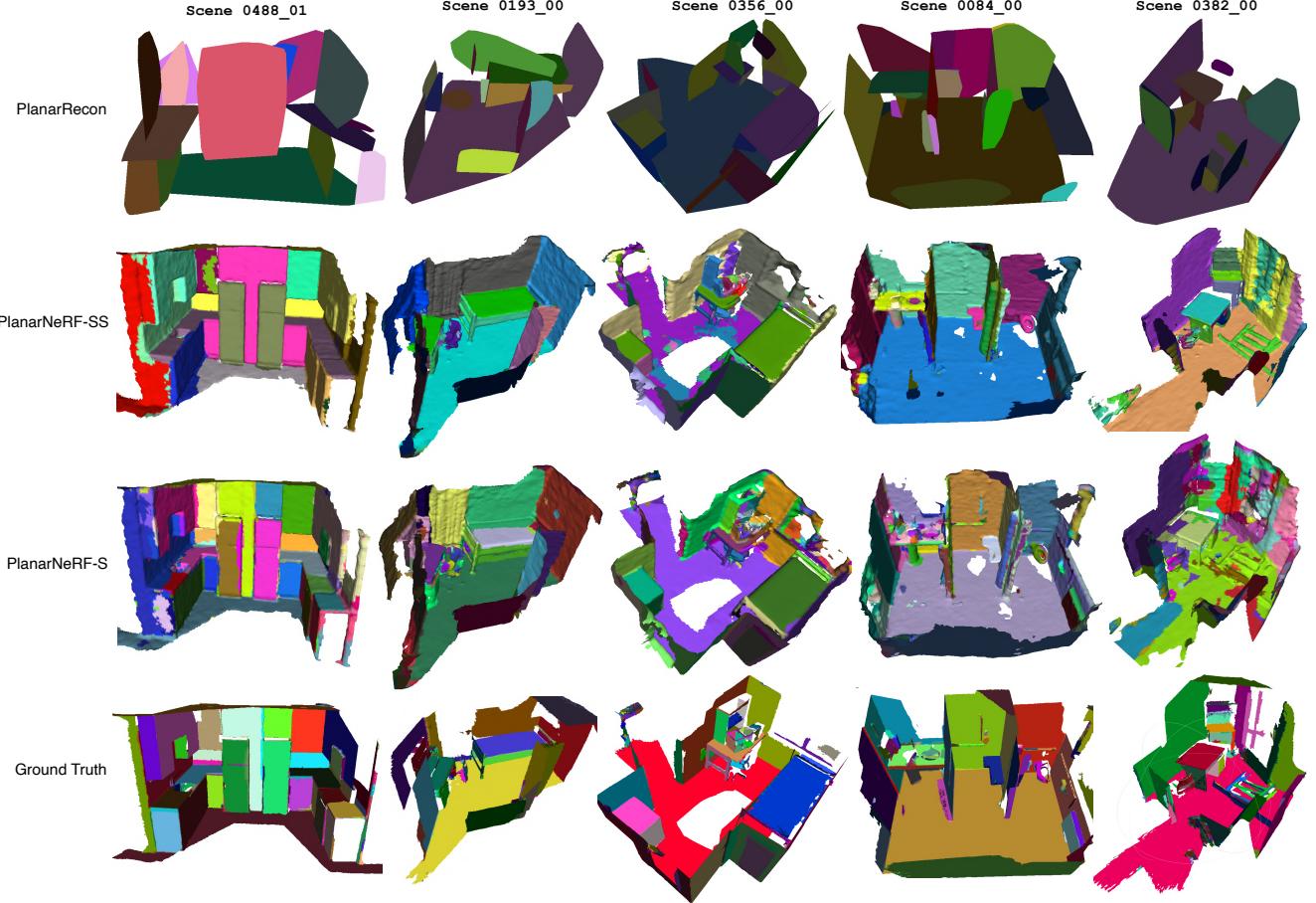


Figure 5. Qualitative comparison between the recent SOTA — PlanarRecon [43] and ours on ScanNet.

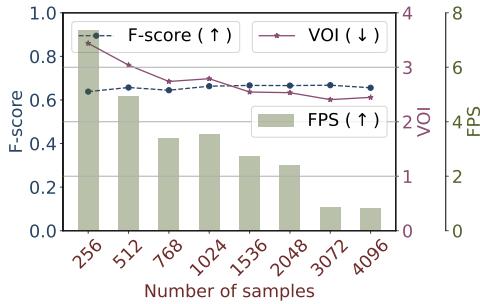


Figure 6. Ablation for the number of samples used in PlanarNeRF.

is an online-learning method; The training of SDF and color rendering takes around $180ms$ while self-supervised plane estimation and plane rendering learning takes around $148ms$. It is acceptable to be slower than the pure inference speed of the offline-trained models. From Table 2, we can still see the advantages of our method over other baselines in terms of the quality of plane instance segmentation.

From the above quantitative results, we can observe that PlanarRecon achieves the best performance among all the baselines. To further validate the advantages of our method,

Table 4. Ablation studies for similarity threshold.

τ_{dist}	0.01	0.1	0.2	0.3	0.5	0.7
VOI \downarrow	3.219	2.726	2.951	2.753	3.356	3.244
RI \uparrow	0.878	0.874	0.875	0.880	0.858	0.856
SC \uparrow	0.251	0.338	0.276	0.279	0.200	0.141

Table 5. Ablation studies for EMA coefficient.

ψ	0.6	0.7	0.8	0.9	0.99	0.999
VOI \downarrow	3.532	3.655	3.587	3.018	3.438	2.812
RI \uparrow	0.830	0.814	0.879	0.881	0.890	0.866
SC \uparrow	0.204	0.162	0.088	0.146	0.268	0.314

we show more **qualitative** comparisons between our PlanarNeRF with PlanarRecon in Fig. 5. Both of our methods (PlanarNeRF-SS and PlanarNeRF-S) maintain high-quality performance across very diverse indoor rooms.

4.4. Ablation Studies[†]

Replica and Synthetic. We show qualitative results of our model on Replica and Synthetic datasets in Fig. 7. Our

[†]For the purpose of ablation, we randomly select 10 scenes from the validation set. The results of all *quantitative* experiments through this section are based on the selected scenes.

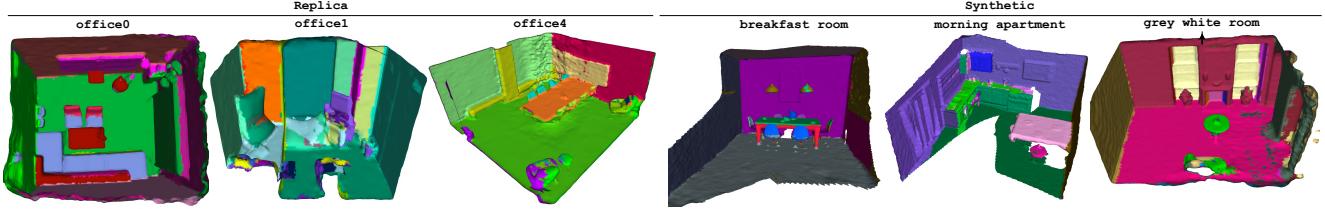


Figure 7. Results by PlanarNeRF for Replica and synthetic scenes from NeuralRGBD.

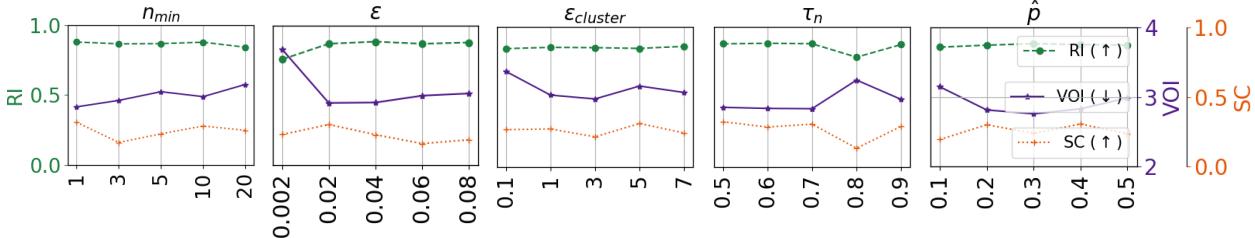


Figure 8. Tuning of different hyperparameters (see definitions of each in Sec. 3.4) in our lightweight plane fitting module.

model can generate excellent plane reconstructions without any annotations (pose/2D planes/3D planes) in an online manner. Note that there is no ground truth and none of the baselines reported results for those datasets. Therefore, we are only able to show the results from PlanarNeRF-SS. More results by our model on those datasets are listed in the supplementary material.

How many samples are used? The number of samples used in PlanarNeRF is very important because they are used for all learning modules (pose/SDF/color/plane) in the framework. It is also closely related to the computational speed. To achieve the best tradeoff, we have conducted thorough experiments. The detailed comparisons are presented in Fig. 6, where we report the geometry quality with F-score; segmentation quality with VOI; and the speed with Frames Per Second (FPS). In our work, we use the number of 768.

Plane similarity measure. To validate the usefulness of Eq. (8) and show the disadvantage of the Eq. (7), we quantitatively compare different plane similarity measures in Table 3, from where we can see that using Eq. (8) achieves the best performance.

Tuning the lightweight plane fitting. As we introduced in Sec. 3.4, our plane fitting module has several key hyperparameters. Satisfying plane fitting results requires a careful selection of the combination of those different hyperparameters. In PlanarNeRF, we use the values as $n_{min} = 1$; $\epsilon = 0.02$; $\epsilon_{cluster} = 1$; $\tau_n = 0.7$; $\hat{p} = 0.3$. Detailed quantitative comparisons in terms of plane instance segmentation quality with RI, VOI, and SC can be seen in Fig. 8.

Thersholt for similarity measure. After the computation of the similarity measure, we need a threshold to determine whether the two planes belong to one instance. If the thresholt is too small, there will be too much noise. If the thresholt is too large, parallel planes might be treated as one in-

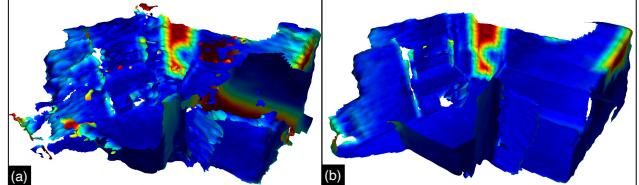


Figure 9. Error map with (a) allowing gradients backpropagation and (b) blocking gradients backpropagation. Red color means a high error and blue color means a low error. Note that the dark red region appears in (a) and (b) because the ground truth fails to capture the window area.

stance. We use the threshold of 0.1.

Coefficient for EMA. During the maintenance of the global memory bank, we use an EMA to update the plane parameters in the bank. The selection of the coefficient in EMA can also affect the final performance a lot. We take the value of ψ as 0.999. Please see a quantitative comparison in Table 5.

Gradient Backpropagation. In PlanarNeRF model architecture (Fig. 2), we stop backpropagating the gradients from the plane branch to the SDF branch during training. This is necessary because the gradients from plane rendering loss can disturb the training of the SDF MLP, weakening the reconstruction quality. We show the qualitative comparison using error maps in Fig. 9.

5. Conclusion

In this paper, we propose a novel plane detection model, PlanarNeRF. This framework introduces a unique methodology that combines plane segmentation rendering, an efficient plane fitting module, and an innovative memory bank for 3D planar detection and global tracking. These contributions enable PlanarNeRF to learn effectively from monocular RGB and depth sequences. Demonstrated through extensive testing, its ability to outperform existing methods marks a significant advancement in plane detection tech-

niques. PlanarNeRF not only challenges existing paradigms but also sets a new standard in the field, highlighting its potential for diverse real-world applications.

References

- [1] Samir Agarwala, Linyi Jin, Chris Rockwell, and David F Fouhey. Planeformers: From sparse view planes to 3d reconstruction. In *European Conference on Computer Vision*, pages 192–209. Springer, 2022. [2](#)
- [2] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgbd surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301, 2022. [3, 6](#)
- [3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. [2](#)
- [4] Chuchu Chen, Patrick Geneva, Yuxiang Peng, Woosik Lee, and Guoquan Huang. Monocular visual-inertial odometry with planar regularities. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6224–6231. IEEE, 2023. [2](#)
- [5] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems*, 34:17864–17875, 2021. [4](#)
- [6] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022. [4](#)
- [7] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. [6](#)
- [8] Zhuo Deng, Sinisa Todorovic, and Longin Jan Latecki. Unsupervised object region proposals for rgbd indoor scenes. *Computer Vision and Image Understanding*, 154:127–136, 2017. [2](#)
- [9] Chen Feng, Yuichi Taguchi, and Vineet R Kamat. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6218–6225. IEEE, 2014. [5](#)
- [10] Martin A Fischler and Robert C Bolles. 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. [2, 5](#)
- [11] Yiming Gao, Yan-Pei Cao, and Ying Shan. Surfelnerf: Neural surfel radiance fields for online photorealistic reconstruction of indoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 108–118, 2023. [3](#)
- [12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. [2, 4](#)
- [13] Linyi Jin, Shengyi Qian, Andrew Owens, and David F Fouhey. Planar surface reconstruction from sparse views. In

- Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12991–13000, 2021. 2
- [14] Michael Kaess. Simultaneous localization and mapping with infinite planes. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4605–4611. IEEE, 2015. 2
- [15] Pyojin Kim, Brian Coltin, and H Jin Kim. Linear rgbd slam for planar environments. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 333–348, 2018. 2
- [16] Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas J Guibas. Supervised fitting of geometric primitives to 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2652–2660, 2019. 2
- [17] Muxingzi Li and Florent Lafarge. Planar shape based registration for multi-modal geometry. In *BMVC 2021-The British Machine Vision Conference*, 2021. 2
- [18] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unterath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023. 3
- [19] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. Planenet: Piece-wise planar reconstruction from a single rgbd image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2588, 2018. 2
- [20] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4450–4459, 2019. 2, 4, 5, 6
- [21] Jiachen Liu, Pan Ji, Nitin Bansal, Changjiang Cai, Qingshan Yan, Xiaolei Huang, and Yi Xu. Planemvs: 3d plane reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8665–8675, 2022. 2
- [22] Xiaoxiao Long, Lingjie Liu, Wei Li, Christian Theobalt, and Wenping Wang. Multi-view depth estimation using epipolar spatio-temporal networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8258–8267, 2021. 5
- [23] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 3
- [24] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 3
- [25] Zak Murez, Tarrence Van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pages 414–431. Springer, 2020. 5, 6
- [26] Tahir Rabbani, Frank Van Den Heuvel, and George Vosselmann. Segmentation of point clouds using smoothness constraint. *International archives of photogrammetry, remote sensing and spatial information sciences*, 36(5):248–253, 2006. 2
- [27] Zhongzheng Ren, Ishan Misra, Alexander G Schwing, and Rohit Girdhar. 3d spatial recognition without spatially labeled 3d. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13204–13213, 2021. 2
- [28] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, pages 214–226. Wiley Online Library, 2007. 2, 4
- [29] Gopal Sharma, Difan Liu, Subhransu Maji, Evangelos Kalogerakis, Siddhartha Chaudhuri, and Radomír Měch. Parsenet: A parametric surface fitting network for 3d point clouds. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pages 261–276. Springer, 2020. 2
- [30] Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kortscheder. Panoptic lifting for 3d scene understanding with neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9043–9052, 2023. 4
- [31] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7–13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012. 2
- [32] Noah Stier, Anurag Ranjan, Alex Colburn, Yajie Yan, Liang Yang, Fangchang Ma, and Baptiste Angles. Finerecon: Depth-aware feed-forward network for detailed 3d reconstruction. *arXiv preprint arXiv:2304.01480*, 2023. 3
- [33] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 6
- [34] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022. 3
- [35] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607, 2021. 5
- [36] Bin Tan, Nan Xue, Song Bai, Tianfu Wu, and Gui-Song Xia. Planetr: Structure-guided transformers for 3d plane recovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4186–4195, 2021. 2, 4
- [37] Bin Tan, Nan Xue, Tianfu Wu, and Gui-Song Xia. Noepsac: Neural one-plane ransac for sparse-view planar 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 2

- [38] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13293–13302, 2023. [3](#), [6](#), [1](#)
- [39] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. [3](#)
- [40] Yusen Wang, Zongcheng Li, Yu Jiang, Kaixuan Zhou, Tuo Cao, Yanping Fu, and Chunxia Xiao. Neuralroom: Geometry-constrained neural implicit surfaces for indoor scene reconstruction. *arXiv preprint arXiv:2210.06853*, 2022.
- [41] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3295–3306, 2023.
- [42] Tong Wu, Jiaqi Wang, Xingang Pan, Xudong Xu, Christian Theobalt, Ziwei Liu, and Dahua Lin. Voxurf: Voxel-based efficient and accurate neural surface reconstruction. *arXiv preprint arXiv:2208.12697*, 2022. [3](#)
- [43] Yiming Xie, Matheus Gadelha, Fengting Yang, Xiaowei Zhou, and Huaizu Jiang. Planarrecon: Real-time 3d plane detection and reconstruction from posed monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6219–6228, 2022. [2](#), [5](#), [6](#), [7](#)
- [44] Siming Yan, Zhenpei Yang, Chongyang Ma, Haibin Huang, Etienne Vouga, and Qixing Huang. Hpnet: Deep primitive segmentation using hybrid representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2753–2762, 2021. [2](#)
- [45] Fengting Yang and Zihan Zhou. Recovering 3d planes from a single image via convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018. [2](#)
- [46] Botao Ye, Sifei Liu, Xuetong Li, and Ming-Hsuan Yang. Self-supervised super-plane for neural 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21415–21424, 2023. [3](#)
- [47] Mulin Yu and Florent Lafarge. Finding good configurations of planar primitives in unorganized point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6367–6376, 2022. [2](#)
- [48] Zehao Yu, Jia Zheng, Dongze Lian, Zihan Zhou, and Shenghua Gao. Single-image piece-wise planar 3d reconstruction via associative embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1029–1037, 2019. [2](#), [5](#), [6](#)
- [49] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in neural information processing systems*, 35:25018–25032, 2022. [3](#)
- [50] Lipu Zhou. Efficient second-order plane adjustment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13113–13121, 2023. [2](#)
- [51] Chen Zhu, Zihan Zhou, Ziran Xing, Yanbing Dong, Yi Ma, and Jingyi Yu. Robust plane-based calibration of multiple non-overlapping cameras. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 658–666. IEEE, 2016. [2](#)
- [52] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. [3](#)
- [53] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint arXiv:2302.03594*, 2023. [3](#)

PlanarNeRF: Online Learning of Planar Primitives with Neural Radiance Fields

Supplementary Material

6. Organization of Supplementary Material

In this supplementary material, we provide more details about our model design and evaluations in broader scenarios. We run our model in more scenes beyond the ones presented in the main paper. The organization of the supplementary material is as follows: In Sect. 7, we present the complete algorithm for maintaining the global memory bank. In Sect. 8, we provide more details about the implementation of PlanarNeRF. In Fig. 1 of the main paper, we show that our model is able to render 2D plane segmentation images. In Sect. 9, we show more 2D plane segmentation results by our model for different datasets. We present more qualitative results for ScanNet in Sect. 10; for Replica and Synthetic in Sect. 11 and Sect. 12.

7. Algorithm for Global Memory Bank

Algorithm 1 Global Memory Bank (Python-style)

```

1: Global input:  $f_\Theta, B \in \mathbb{R}^{Z \times 4}, T, \tau_{dis}, g$ 
2:  $g \leftarrow 0; B \leftarrow 0$                                  $\triangleright$  Initializations
3: for  $t$  in range( $T$ ) do           $\triangleright$  Loop over training
   iterations
4:   Local input:  $x_t, D_t$ 
5:   Local output:  $y \in \mathbb{R}^{N_t \times C}$             $\triangleright$  Plane label
6:    $PO \leftarrow unproj(D_t, x)$             $\triangleright$  Obtain 3D points
7:    $\{\mathbf{p}_i\}_{i=1}^{N_t}, \{PO_i\}_{i=1}^{N_t} = f_\Theta(PO)$      $\triangleright$  Lightweight
   plane fitting
8:   if  $t == 0$  then
9:      $B[:, :N_t, :] \leftarrow \{\mathbf{p}_i\}_{i=1}^{N_t}$ 
10:     $g \leftarrow g + N_t$ 
11:   else
12:     for  $j$  in range( $g$ ) do
13:       for  $k$  in range( $N_t$ ) do
14:         if  $dis(B[j, :], PO_k) < \tau_{dis}$  then
15:            $B[j, :] \leftarrow \psi B[j, :] + (1 - \psi) \mathbf{p}_k$ 
16:            $y[k, :] = \mathbb{1}_{\{c=j\}}$             $\triangleright c : Index$ 
17:         else
18:            $B[g, :] \leftarrow \mathbf{p}_k$ 
19:            $y[k, :] = \mathbb{1}_{\{c=g\}}$             $\triangleright c : Index$ 
20:            $g \leftarrow g + 1$ 
21:         end if
22:       end for
23:     end for
24:   end if
25:   Return  $y$                                  $\triangleright$  For training of current batch
26: end for

```

We provide the complete pipeline for maintaining the global memory bank and generating plane labels during the learning process of **PlanarNeRF-SS** in Algo. 1, where the global input variables are defined as follows: f_Θ is the lightweight plane fitting module; B is the global memory bank represented by a 2D matrix, storing global plane parameters. Z is a predefined integer that defines the number of maximum plane instances in the scene; T is the total number of image frames; τ_{dis} is the threshold for determining if given two planes belong to the same instance; g is a global index which always indicates the new position that new planes will be inserted in the global bank. In each iteration, t , the local input variables in each iteration are: x_t is the camera pose; D_t is the sampled depth values from depth images. The output for each iteration is a plane label matrix $y \in \mathbb{R}^{N_t \times C}$, where N_t is the number of plane instances estimated in the iteration t . The function *unproj* generates 3D points from camera pose and depth values.

8. Implementation Details of PlanarNeRF

PlanarNeRF adopts the Co-SLAM [38] as the backbone, and is able to predict pose, geometry, and plane instance segmentation simultaneously. Our model relies on samples in the image space to train the neural networks (MLPs). A hierarchical training scheme — a combination of local optimization and global optimization is used. Local optimization uses samples from each local frame and global optimization uses an accumulation of historical samples. We use 1024 and 768 samples in local optimization and global optimization, respectively. In PlanarNeRF, lightweight plane fitting, global memory bank update, and plane rendering training only occur in global optimization.

PlanarNeRF has three MLPs — SDF MLP, color MLP, and plane MLP, each of which is a 2-layer shallow MLP with 32 hidden units. During training, we use learning rate of $1e-3$ for optimizing pose vectors and $1e-2$ for optimizing all MLPs. The truncation distance tr in the rendering weights is set as $10cm$.

9. 2D Plane Segmentation

We show more examples of 2D plane segmentation rendering for ScanNet by our model in Fig. 10; for Synthetic in Fig. 11; and for Replica in Fig. 13. For ScanNet, we also list the ground truth in the last column. For other datasets, we are only able to present the results by our model since there is no plane label available.

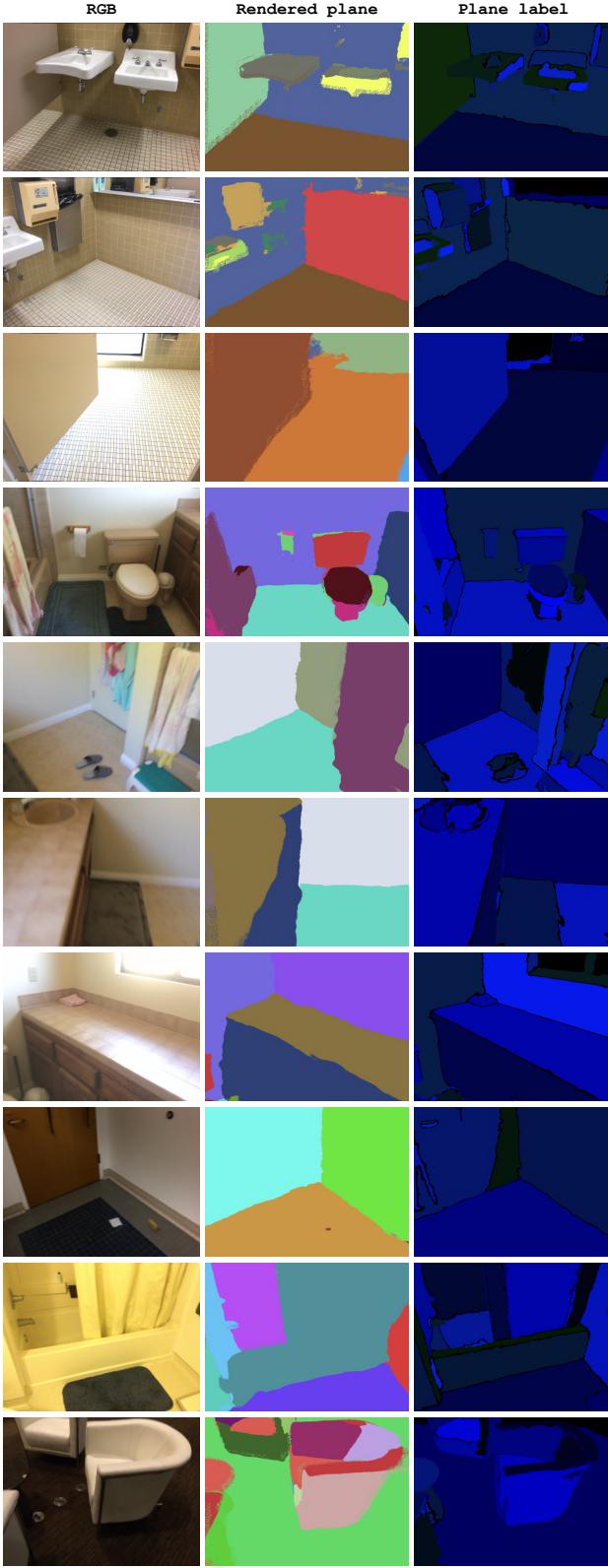


Figure 10. 2D plane segmentation rendering on ScanNet.

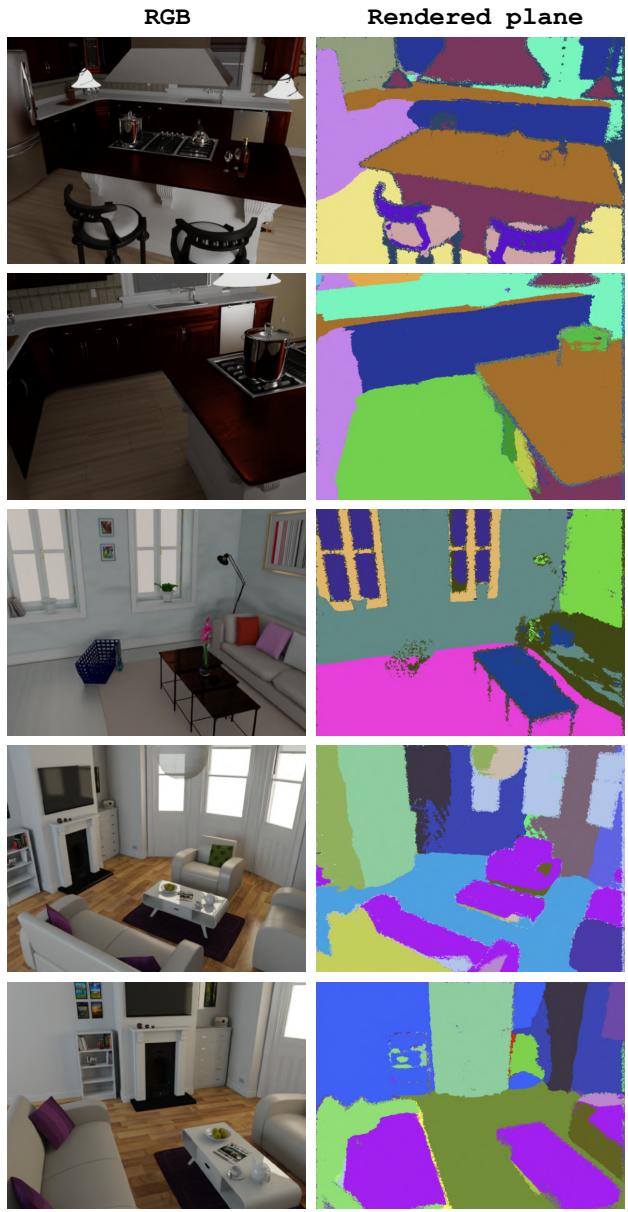


Figure 11. 2D plane segmentation rendering on Synthetic.

10. More Results of ScanNet

More comparisons on ScanNet between the recent SOTA work [43] and ours can be seen in Fig. 12, Fig. 14, Fig. 15, and Fig. 16. Meshes for each scene are arranged in a **row-wise** manner.

11. More Results of Replica

We show more qualitative results on Replica by our model in Fig. 17 and Fig. 18, in both of which we reveal the reconstruction process at five different time steps. Meshes for each scene are arranged in a **column-wise** manner. For

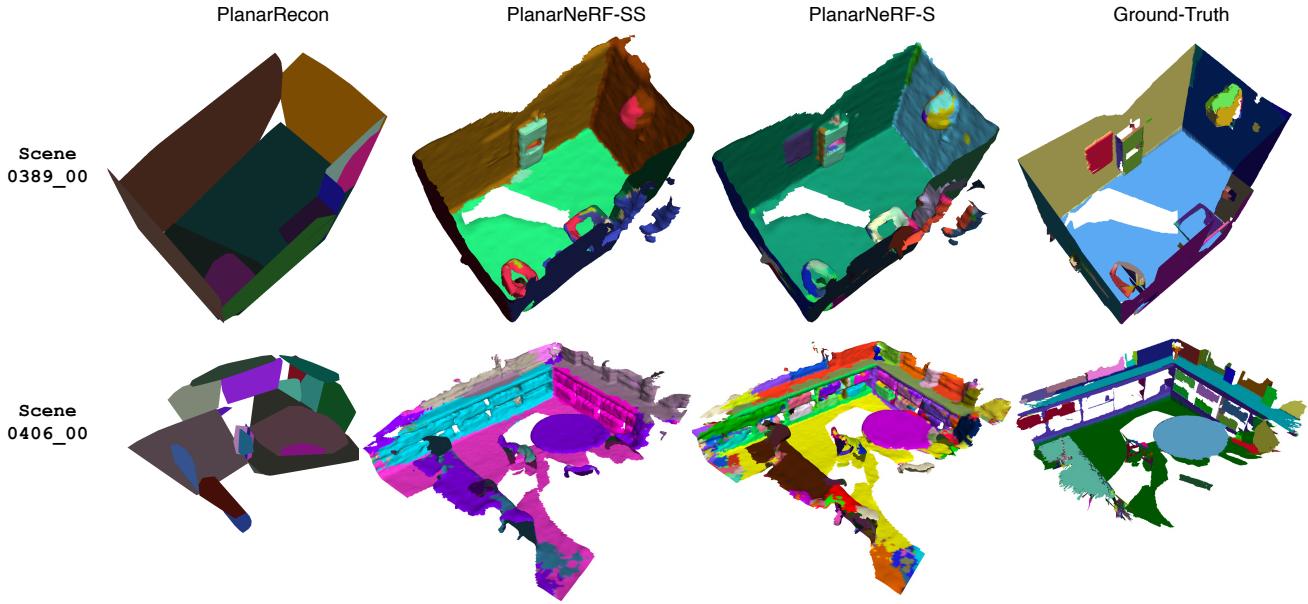


Figure 12. Qualitative comparison on ScanNet.

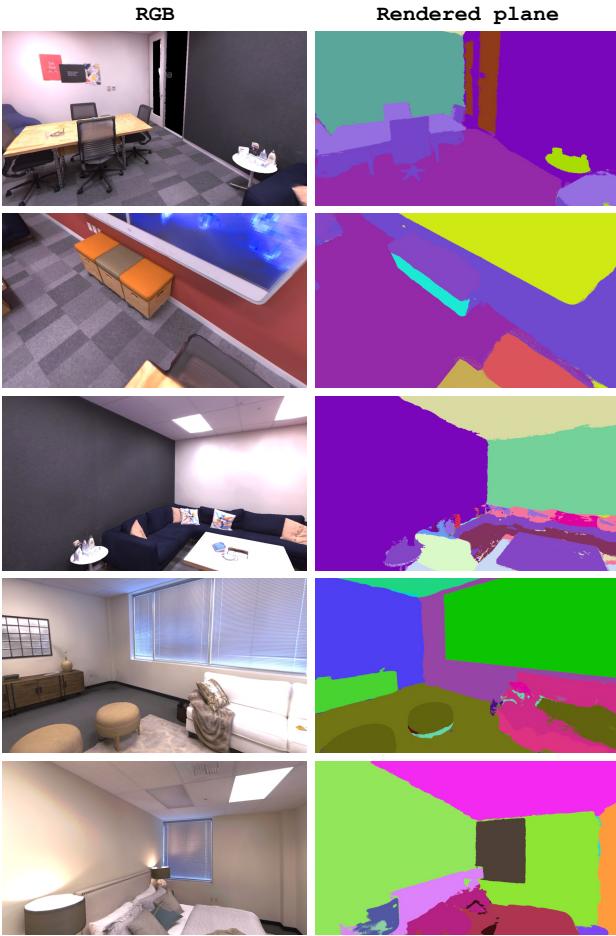


Figure 13. 2D plane segmentation rendering on Replica.

Replica and Synthetic (next section), we are not able to compare ours with other methods because no plane label is available.

12. More Results of Synthetic

We show more qualitative results on Synthetic by our model in Fig. 18. Meshes for each scene are arranged in a **column-wise** manner.

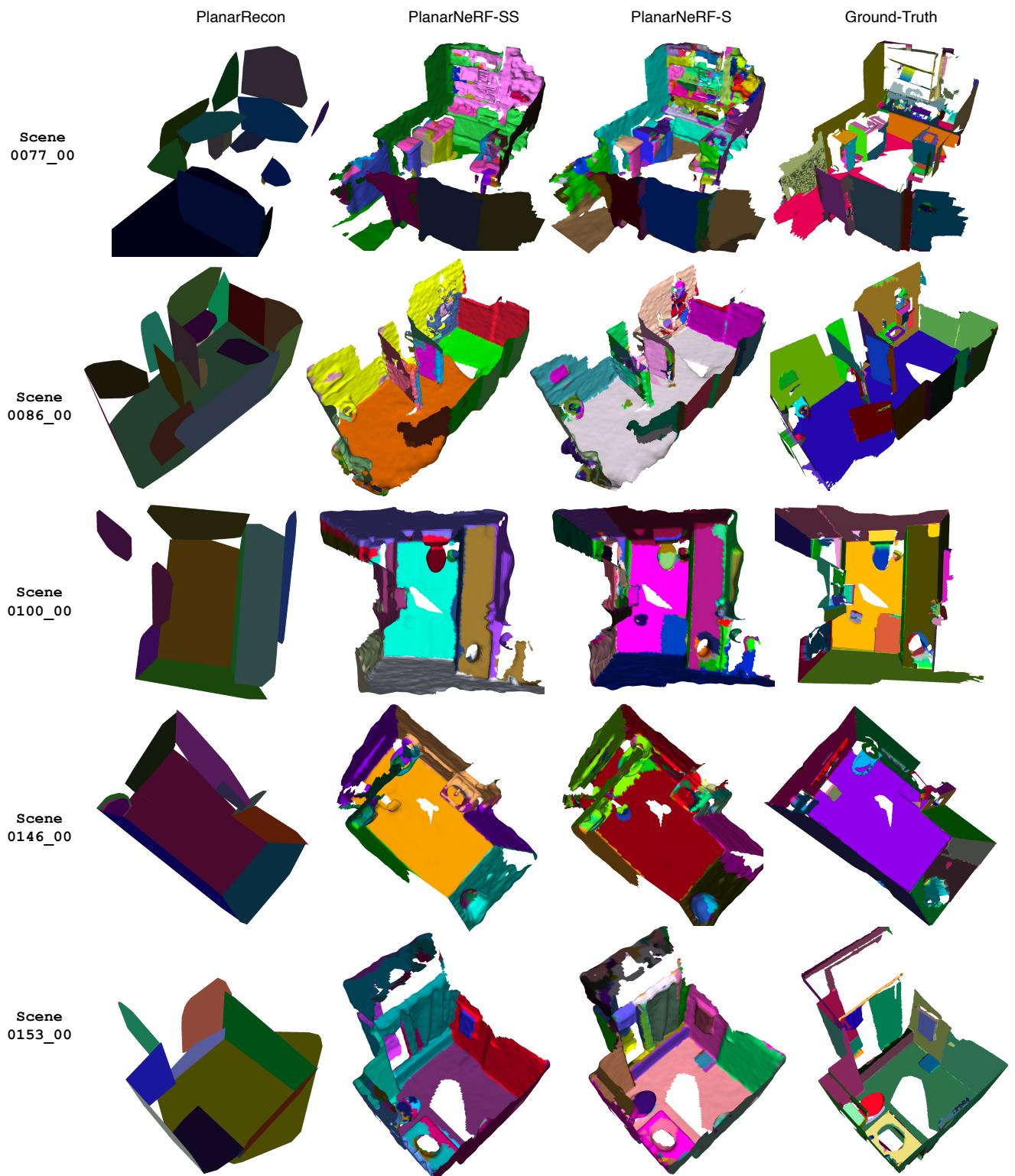


Figure 14. Qualitative comparison on ScanNet.

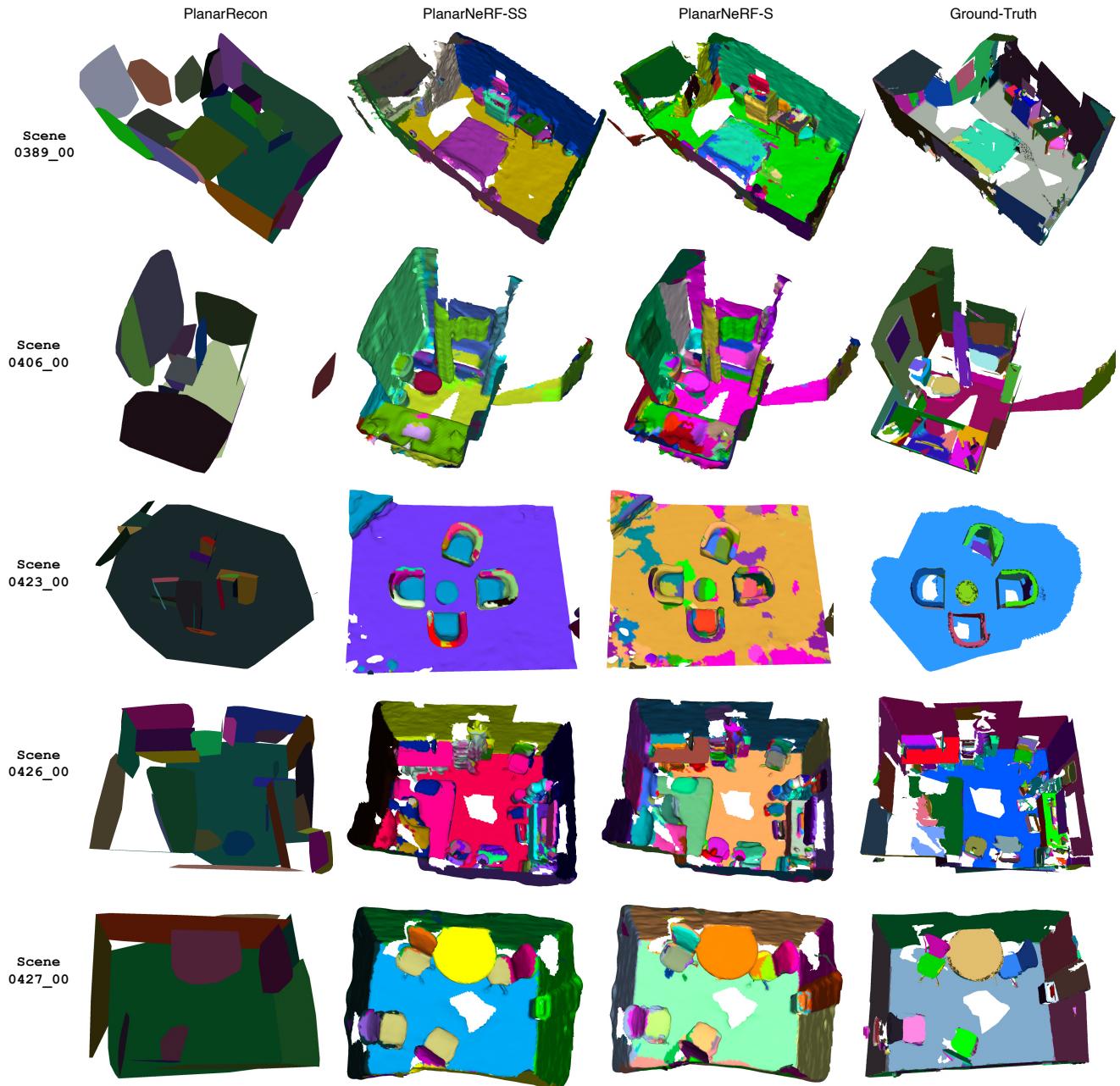


Figure 15. Qualitative comparison on ScanNet.

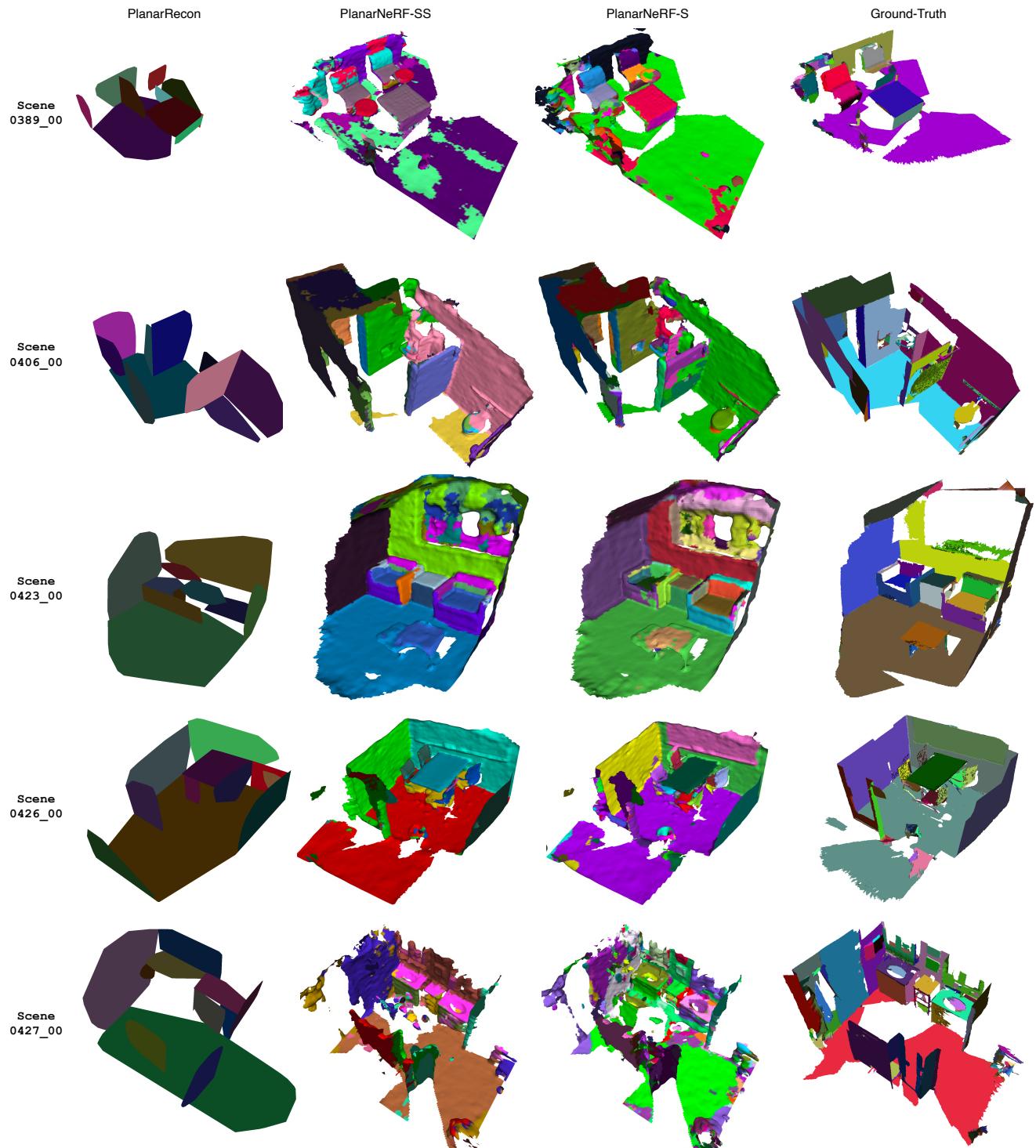


Figure 16. Qualitative comparison on ScanNet.

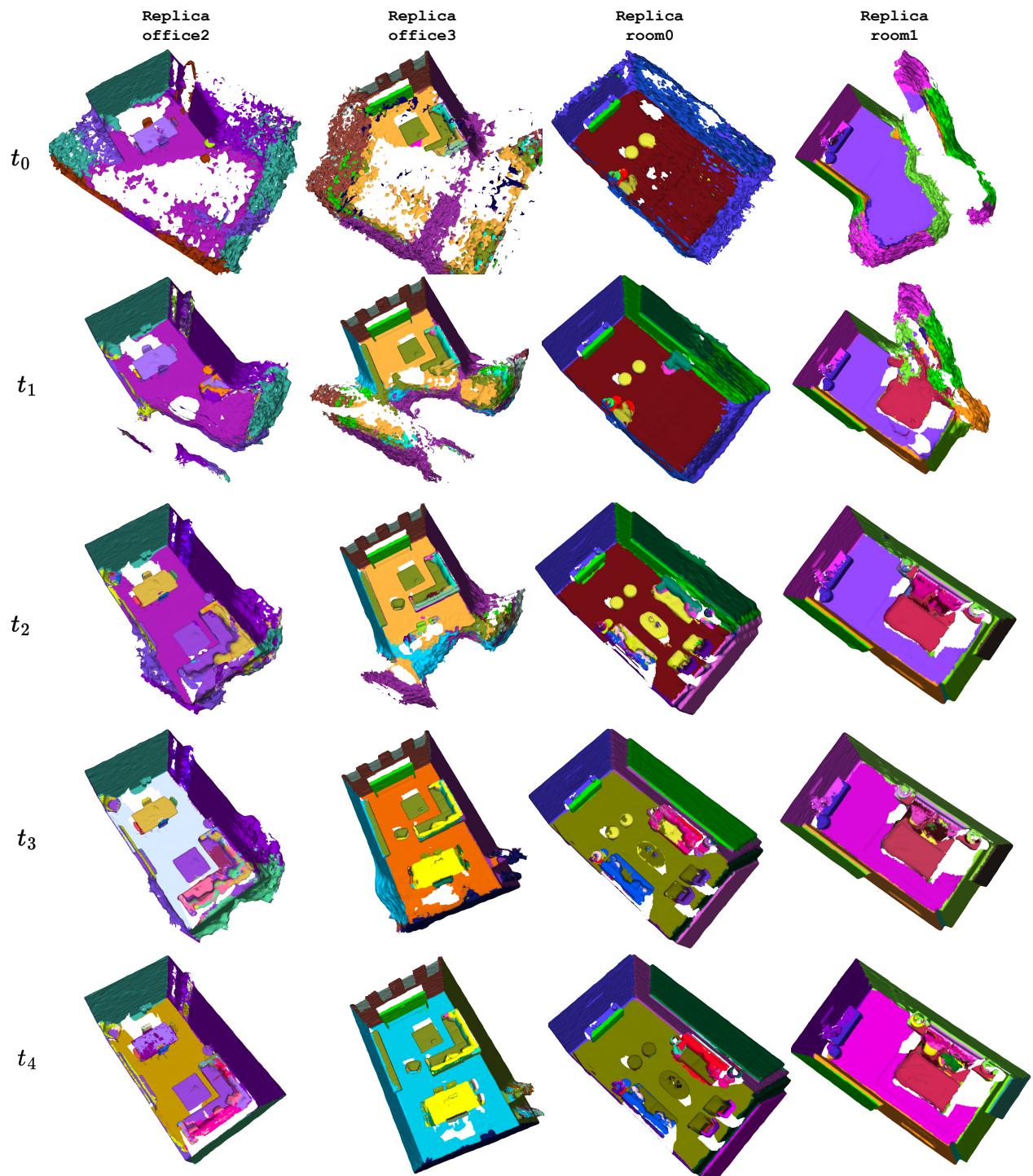


Figure 17. Reconstruction process of scenes in Replica.

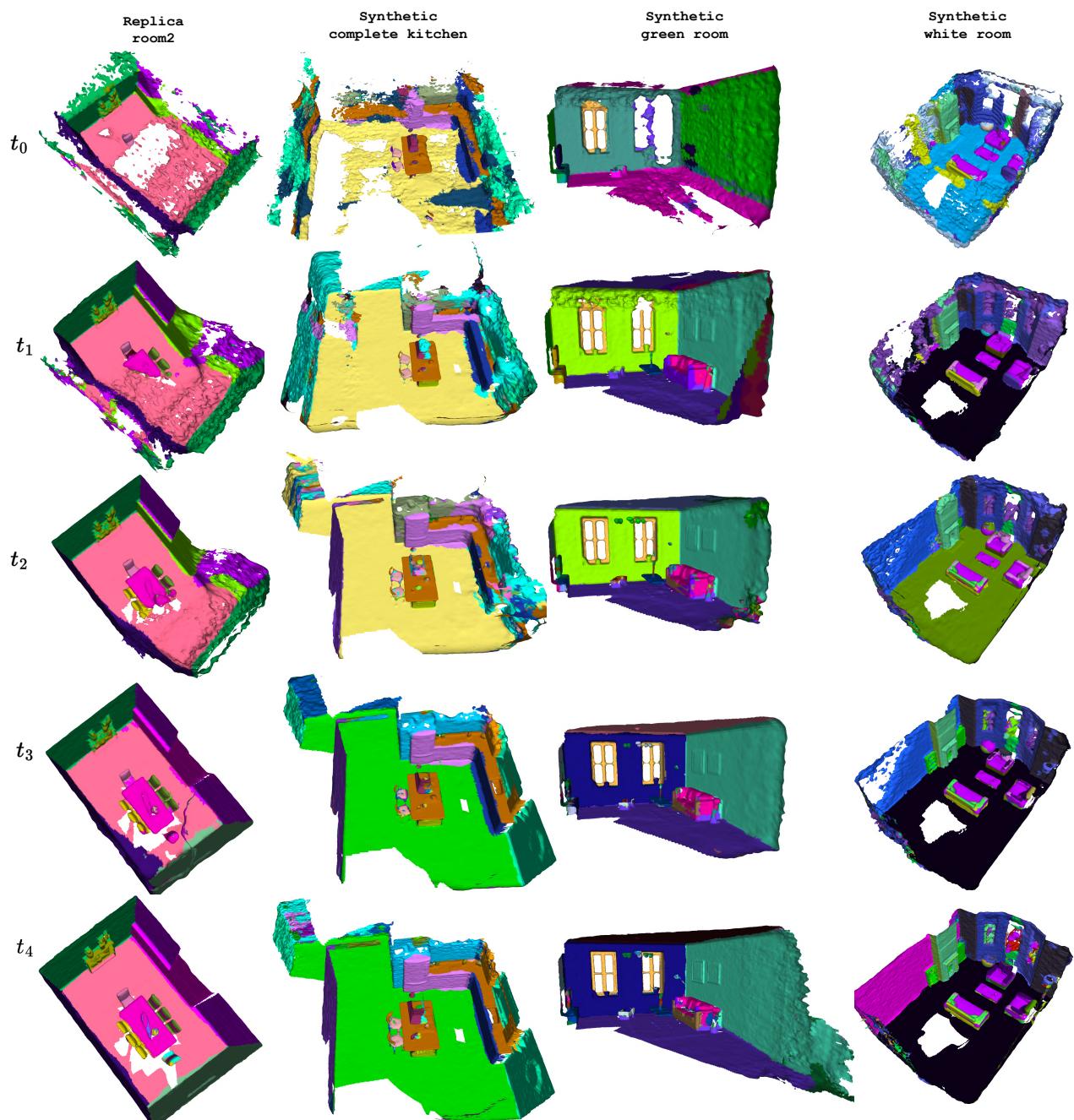


Figure 18. Reconstruction process of scenes in Replica (1st column) and Synthetic (Remaining columns).