PlanarRecon: Real-time 3D Plane Detection and Reconstruction from Posed Monocular Videos

Yiming Xie[1,2] Matheus Gadelha[3] Fengting Yang[4] Xiaowei Zhou[2†] Huaizu Jiang[1†]
[1]Northeastern University [2]Zhejiang University [3]Adobe Research [4]The Pennsylvania State University

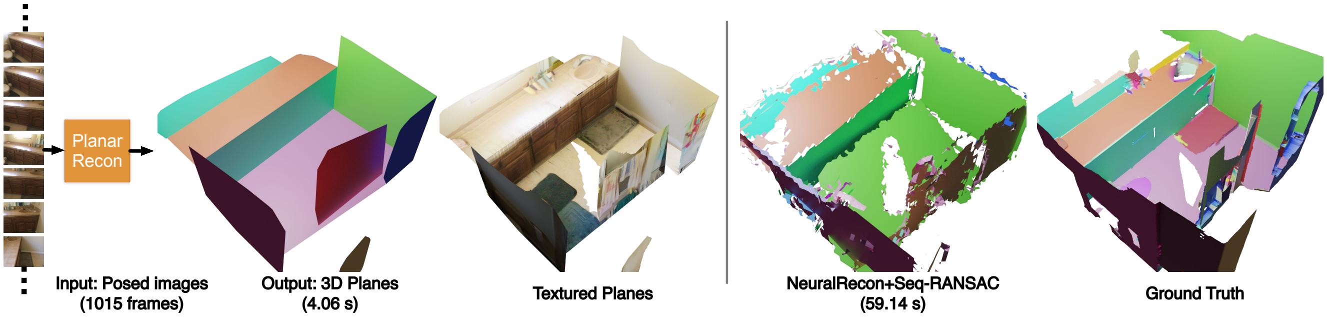
Planar Recon
Input: Posed images (1015 frames)
Output: 3D Planes (4.06 s)
Textured Planes
NeuralRecon+Seq-RANSAC (59.14 s)
Ground Truth

Figure 1. Comparison of the proposed approach, PlanarRecon, and our baseline. Colored planes mean different instances. We also show the textured mesh for our approach. For the multi-view method NeuralRecon [37] + Seq-RANSAC [14], the Sequential RANSAC is used to extract planes after the geometry reconstruction. Our model produces a much more accurate and coherent 3D plane detection in real-time. *Best viewed in color.*

Abstract

We present PlanarRecon – a novel framework for globally coherent detection and reconstruction of 3D planes from a posed monocular video. Unlike previous works that detect planes in 2D from a single image, PlanarRecon incrementally detects planes in 3D for each video fragment, which consists of a set of key frames, from a volumetric representation of the scene using neural networks. A learning-based tracking and fusion module is designed to merge planes from previous fragments to form a coherent global plane reconstruction. Such design allows PlanarRecon to integrate observations from multiple views within each fragment and temporal information across different ones, resulting in an accurate and coherent reconstruction of the scene abstraction with low-polygonal geometry. Experiments show that the proposed approach achieves state-of-the-art performances on the ScanNet dataset while being real-time. Code is available at the project page: https://neu-vi.github.io/planarrecon/.

1. Introduction

Recovering 3D planar surfaces from a posed monocular video is a critical task for many downstream applications in 3D vision, such as Augmented and Virtual Reality (AR and VR), interior modeling, and human-robot interaction. Planar surfaces provide a compact representation and important geometric cues of the 3D scene. AR, for example, to enable realistic and immersive interactions between AR effects and the surrounding physical scene, 3D plane detection needs to be accurate, consistent, and performed in real-time. While camera poses can be tracked accurately with state-of-the-art visual-inertial SLAM systems [1,5,30], real-time image-based 3D plane detection remains to be a challenging problem due to the low detection quality and high computation demands.

Most recent deep learning-based plane recovery works [22, 23, 38, 42, 48] focus on the single-view case. Their pipeline typically aims to jointly segment the plane instances and regress the plane parameters (*i.e.*, surface normals and offsets). Despite the significant progress made in this direction using deep neural networks, because of the single-view scale ambiguity, these methods cannot deliver absolute depth estimation in the unknown scenes. Moreover, fusing the plane detections from multiple views is not trivial. Jin *et al.* [19] extend the single-view approach [22] to sparse views (mostly 2 views), where time-consuming

This work was partially done when Yiming Xie was a research assistant at Zhejiang University. † Corresponding authors: Xiaowei Zhou and Huaizu Jiang.

energy minimization is used to fuse single-view detections. As their design does not consider the temporal consistency, however, it is still unclear how to extend this work to video inputs, which are more natural and common vision sources for applications like AR and VR.

In the traditional 3D vision, a few attempts [2, 3, 7, 15, 34, 44, 45] have been made to recover planes from multi-view images and videos. But they usually rely on hand-crafted features [2, 3, 34] and/or strong geometric priors of the scene [7, 15, 44, 45]. In real-world scenarios, these features may be unreliable and such priors may not always hold due to the scene complexity, such as lighting condition change, textureless regions, fixtures violating the Manhattan world assumption [8], etc. An efficient plane recovery method that is robust to the aforementioned challenges and makes no strong assumptions of the scene is desired.

To meet this demand, we propose a novel learning-based framework, called **PlanarRecon**, to perform 3D plane detection and reconstruction in real-time from a posed monocular video. The main idea of our proposed PlanarRecon is illustrated in Fig. 2. It consists of two major components. The first component is *fragment-based plane detection*. Given a video input, we sequentially split it into multiple non-overlapping fragments. For each fragment, PlanarRecon constructs 3D feature volumes by back-projection of the image features, which fuses information from multiple views. Based on the occupancy classification, for each occupied voxel, we estimate the plane parameters as well as its displacement to shift it to the geometric centroid of the plane it belongs to. Mean-shift clustering [48] is then performed to group voxels that have similar plane parameters and shifted positions to get plane detections in the fragment. The second component is *plane tracking and fusion*. PlaneRecon maintains a global reconstruction of planes using plane detections from all previous fragments. When a new fragment is processed, we resort to the attention mechanism [39] to compute the similarities between the global reconstructed planes and the current detections. A differentiable Hungarian matching algorithm is then used to obtain the correspondences of planes. The global reconstruction is updated accordingly to ensure temporal coherence.

Our model incrementally obtains 3D plane reconstructions from the input video. Thanks to its fast inference speed, the system can run in real-time, enabling more authentic interaction experiences with the scene for a downstream AR application, for instance. Compared with single-image based approaches [22, 23, 38, 42, 48], PlanarRecon directly regresses planes from 3D feature volumes. It not only fuses information from multiple views, but also offers a coherent reconstruction of the scene without the scale ambiguity. On the other hand, compared with traditional multi-view reconstruction approaches, our learning-based model is more robust to scene complexity [2, 3, 34] and does not rely on the existence of certain scene priors [7, 15, 44, 45]. Experimental results on the ScanNet benchmark [10] show that PlanarRecon achieves state-of-the-art accuracy.

To summarize, our main contributions are twofold: *i*) We propose PlanarRecon to detect and reconstruct 3D planes from a posed monocular video. To our best knowledge, PlanarRecon is the first learning-based approach in this direction. *ii*) We propose a novel volume-based plane reconstruction approach that can detect, track, and fuse plane instances, directly in 3D. Our model integrates observations from multiple frames and temporal information from the video, leading to globally coherent plane detection and reconstruction. Compared with existing approaches, our approach is more robust and runs significantly faster.

2. Related Work

Multi-view Plane Reconstruction. There is a long line of research on multi-view plane reconstruction from sequences of frames with known camera poses. Early works usually first perform sparse 3D reconstruction with point [3] or line features [2] and then group the sparse 3D representations with certain heuristics. However, the reconstruction accuracy is heavily dependent on the hand-crafted features, and not robust to other factors like lighting changes and textureless regions. Other works pose this problem as an image segmentation task. There are methods that assign each pixel to one of the plane hypotheses in an MRF formulation [15, 34]. Others extend this framework to handle non-planar surfaces [16] and introduce superpixel segmentation to better tackle the textureless regions [7]. Our approach, unlike the existing ones, employs convolutional neural networks (CNNs) to extract features and perform optimization in a data-driven manner. To the best of our knowledge, we are the first to perform multi-view plane reconstruction using a deep learning approach.

Another line of work takes monocular videos as input to simultaneously estimate the camera poses and reconstruct the planar surfaces in a SLAM fashion. However, these works commonly assume that the world consists of a horizontal ground and a few vertical planes (*e.g.*, facades or walls) [44, 45], and/or the planar structures only exist in low-gradient areas [7]. In our work, we do not make such assumptions, leading to a more applicable approach.

Learning-based Plane Reconstruction. While we are not aware of any existing deep learning-based work that reconstructs piece-wise planes from multi-view images or video sequences, there are a few studies aiming to recover the planar structures from one or two views. Several works treat the plane reconstruction from single views as an instance segmentation problem [22, 23, 38, 42, 48]. Deep networks are employed to jointly predict the plane instance segmentation and plane parameters. The segmenta-

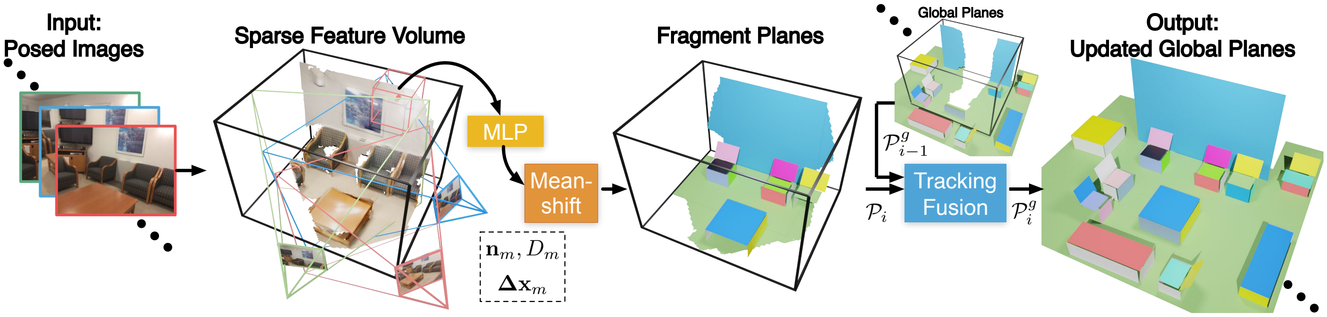

Figure 2. PlanarRecon architecture. PlanarRecon backprojects the image features into a fragment bound volume $\mathcal{F}_i$ and gradually sparsify the volume in a coarse-to-fine approach to form a sparse feature volume. A MLP network will be used to predict plane parameters $[\mathbf{n}_m, D_m]$ and votes $\Delta\mathbf{x}_m$ for each voxel. Then these hybrid geometric primitives $[\mathbf{n}_m, \mathbf{x}'_m = \mathbf{x}_m + \Delta\mathbf{x}_m]$ are passed through a Mean-shift clustering to form plane instance $\mathcal{P}_i$ in fragment bound volume $\mathcal{F}_i$. The tracking and fusion module will match 3D planes $\mathcal{P}_i$ in current fragment bounding volume and global planes $\mathcal{P}^g_{i-1}$ from previous fragments. The matched plane pairs will be refined, yielding the final 3D plane reconstruction.

tion masks are later projected into 3D using the predicted parameters for plane reconstruction. To further improve the reconstruction accuracy, other works detect and enforce the inter-plane relationships among their plane instances [29] or perform segmentation on horizontal and vertical planes separately with panorama inputs [36].

The problem becomes more challenging in the two-view case. To generate a unified 3D reconstruction, the model has to correctly associate the plane instances across frames. Previous work has proposed learning descriptors for each instance and matching them with certain optimization algorithms [19, 32]. However, it is still unclear how well this approach can be generalized to multi-view cases. Instead, we directly perform plane detection, tracking, and fusion in 3D, which reduces the ambiguity of the matching process and leads to higher reconstruction accuracy and more coherent results.

Learning-based Multi-view Stereo. Our work is also related to recent deep learning-based multi-view stereo (MVS) methods [6, 25, 43, 46, 47]. One representative work in this direction is MVSNet [46]. For each frame, MVSNet selects a few neighboring frames as sources and uses a plane-sweeping mechanism to stack per-frame features into a 4D cost volume which is later aggregated with a 3D CNN in order to predict the depth map of the reference frame. The final 3D model is reconstructed by fusing all the estimated depth maps. There are a few other works that do not rely on the plane-sweeping mechanism. Some approaches are inspired by the patch match algorithm [4], which randomly initialize a large number of depth proposals and propagate the ones with low photometric errors calculated with deep features [20, 40]. Sinha *et al.* first generate sparse depth maps with SuperPoint [11] and later perform depth completion [33].

All the works use depth maps as the intermediate 3D representation and only consider the information from neighboring frames. Several methods [12,17,21] integrate temporary information from a video sequence. Other works take the truncated signed distance function (TSDF) as 3D representation and project per-frame features into a pre-defined volumetric space for TSDF regression [27, 37]. In recent work [41], all frames' information is incorporated with neural radiance fields [26] for multi-view depth estimation. Our work adopts the framework from [37] for the initial 3D geometry estimation due to its high efficiency. But different from it, our method directly produces 3D plane reconstruction, which results in less complicated geometry and thus more friendly for downstream tasks that require high processing throughput (*e.g*., AR, physical simulation, interactive applications in general).

3. Method

Given a sequence of monocular video frames and their camera poses, we first find a set of suitable key frames sequentially from the incoming image stream. Following [17], a new incoming frame is selected as a key frame if its relative translation is greater than t_{max} or the relative rotation angle is greater than R_{max} when compared with the last selected key frame. When the number of key frames reaches N_k, the sequence of N_k consecutive key frames forms a local fragment. Each fragment is defined as $\mathcal{F}_i = \{I_{i,j}\}_{j=1}^{N_k}$, where i is the fragment index, j is the key frame index. Given these sequential fragments, our goal is to incrementally detect and reconstruct 3D planes that approximate the underlying 3D scene geometry. Here, we define a plane as a planar structure from a single object instance in the same way as proposed by Liu *et al.* [22].

Fig. 2 presents an overview of the proposed method. We divide our pipeline into two major components. In Section 3.1, we introduce fragment-based plane detection, where we first obtain plane detections for each fragment. Section 3.2 shows a plane tracking and fusion module, where we integrate plane detections from the current frag-

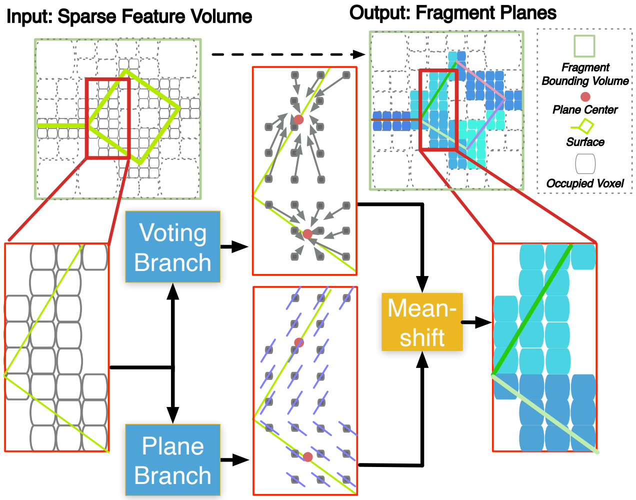

Figure 3. 2D illustration of fragment-based plane detection. Note that we ignore the sparse volume reconstruction, which is detailed in the supplementary material. The colored grids in the fragment bound volume mean different plane instances they belong to. *Best viewed in color.*

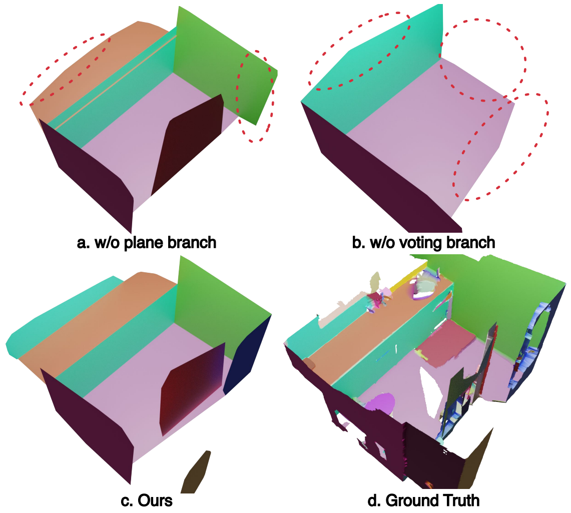

Figure 4. Effectiveness of the plane and voting branches. Colored planes mean different instances. The results in (a), (b) fail to separate planes and produce inaccurate reconstruction (as shown in the **red** circle). *Best viewed in color.*

ment and previous fragments together to form a coherent 3D plane reconstruction of the scene in an incremental manner.

3.1. Fragment-based Plane Detection

Overview. The task of extracting planes would be significantly easier if accurate geometry were available. While detailed geometric information is hard to obtain, removing the majority of the empty space is relatively simple. Thus, we adopt a coarse-to-fine approach to build 3D sparse feature volumes, where we classify each voxel as occupied or not (*i.e.*, if it belongs to the surface or not). For each occupied voxel, we add two sibling branches – a plane branch and a voting branch. Their goal is to estimate the plane parameters and displacement from the centroid of the plane it belongs to, respectively. Finally, clustering is performed on the occupied voxels to group the ones that have similar plane parameters and displacements together in order to get plane detections in 3D. An illustration of this procedure is shown in Fig. 3.

3D Sparse Feature Volume Construction. We define the 3D space for $\mathcal{F}_i$ as the region within a cubic-shaped bounding volume that encloses the view-frustums of all the key frames $\{I_{i,j}\}$. We obtain per-voxel features for occupancy classification by passing each of the key frames $I_{i,j}$ through a 2D CNN backbone. A 3D feature volume is obtained by back-projecting those image-based features into the fragment bounding volume $\mathcal{F}_i$. Average pooling is performed to aggregate features for the same voxel from multiple pixels across different views. Next, we use 3D sparse convolutions to efficiently process the feature volumes and estimate voxel-wise occupancy scores. We define that a voxel is occupied if it is within λ distance from the surface. A voxel whose occupancy score is lower than a threshold θ is considered as void space and therefore is removed from the following plane detection process. Similar to previous volumetric 3D reconstruction works [18,37], after this sparsification process, the fragment bounding volume $\mathcal{F}_i$ is upsampled twice in the next level, and this process is repeated in a coarse-to-fine manner. In our implementation, we use three levels of sparse volumes. We provide more details in the supplementary material.

Plane Branch. After the occupancy analysis, the model would obtain a set of features $\{[\mathbf{x}_m, \mathbf{f}_m]\}_{m=1}^{M}$ for the M occupied voxels[1], where $\mathbf{x}_m \in \mathbb{R}^3$ is the center position of a voxel. $\mathbf{f}_m \in \mathbb{R}^C$ is a C-dimensional feature obtained from the sparse volume constructed in the previous stage, which will be used to regress plane parameters, including the surface normal $\mathbf{n}_m \in \mathbb{R}^3$ and plane offset $d_m \in \mathbb{R}^1$, associated to the voxel.

For the surface normal, similar to [22], we maintain a set of *anchor normals*. For each occupied voxel, we first predict the anchor normal that is closest to the ground-truth value and then regress a 3D residual vector from it. We found it works better than directly regressing the normal values. As our model directly works in 3D, unlike [22], which obtains anchor normals by clustering the ground-truths in 2D, our anchor normals can directly be defined in the world coordinate system. In specific, we manually design six anchor normals, where two of them are parallel to the ground plane and the rest anchor normals is perpendicular to the ground.

[1] For simplicity, we omit the fragment index here.

For the plane offset regression, we estimate the distance $D_m \in \mathbb{R}^1$ from the center position of a voxel x_m to the plane it belongs to. Given an estimated plane normal n_m and distance D_m, the plane offset d_m can be obtained via

$$d_m = -\langle \tilde{\mathbf{x}}_m, \mathbf{n}_m \rangle, \;\; \tilde{\mathbf{x}}_m = \mathbf{x}_m + D_m \mathbf{n}_m, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product between two vectors.

The plane branch is instantiated by multi-layer perceptions (MLPs). We use the cross-entropy loss to supervise the learning of anchor normal selection, and the `smooth_L1` loss to supervise the anchor normal residual vector and plane offset distance regression.

Voting Branch. Given the plane parameter estimations, we can already group voxels together to get plane detections. However, such a strategy may fail to separate two planar surfaces that have similar plane parameters even they are far from each other, *e.g*. two tables surfaces that have the same height. To this end, inspired by [28], we also estimate the displacement $\Delta\mathbf{x}_m \in \mathbb{R}^3$ from the voxel center $\mathbf{x}_m$ to the centroid of the planar surface it belongs to. With the estimated displacement $\Delta\mathbf{x}_m$, we can shift the voxel $\mathbf{x}_m$ to the new position $\mathbf{x}'_m = \mathbf{x}_m + \Delta\mathbf{x}_m$. The predicted 3D displacement $\Delta\mathbf{x}'_m$ is supervised by L1 loss.

Voxel Clustering. Once we have geometric primitives for every single voxel (shifted voxels $\mathbf{x}'_m \in \mathbb{R}^3$ and surface normals $\mathbf{n}_m \in \mathbb{R}^3$), we group the voxels to form plane instances in the local fragment volume $\mathcal{F}_i$. We found that the plane offset d_m is not useful for the voxel clustering and introduces an extra computation burden. Following [48], we use an efficient mean-shift clustering algorithm. We use the centers of the clusters as the final plane parameters for each plane instance. We show clustering results with and without using the plane and voting branches in Fig. 4.

3.2. Plane Tracking and Fusion

Overview. To integrate plane detections from different fragments to form a globally coherent 3D plane reconstruction, we design a learning-based tracking and fusion module. We keep a global plane reconstruction $\mathcal{P}^g_i$ consisting of a set of plane instances $\mathcal{P}^g_i = \{P^g_{i,m}\}$, where $\mathcal{P}^g_0 = \emptyset$ and m is the plane instance index. As shown in Fig. 5, given the plane detections $\mathcal{P}_i = \{P_{i,n}\}$ from the fragment $\mathcal{F}_i$, we need to integrate $\mathcal{P}^g_{i-1}$ and $\mathcal{P}_i$ to get an updated global plane reconstruction $\mathcal{P}^g_i$. It involves two steps. First of all, we do *plane tracking* – we need to find matchings between planes of $\mathcal{P}^g_{i-1}$ and $\mathcal{P}_i$. Then, we *fuse* matched planes to get refined plane reconstructions. Each of the two steps is introduced in detail next.

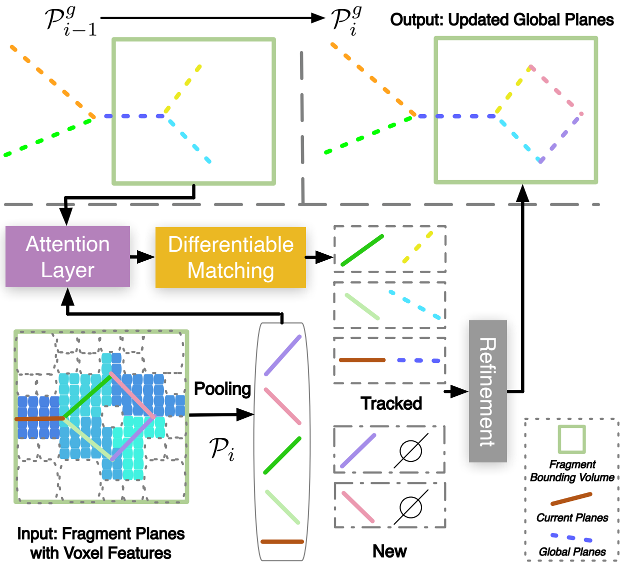

Figure 5. 2D illustration of 3D plane tracking and fusion. The colored line segments mean different plane instances with their descriptors. *Best viewed in color.*

Differentiable Matching for Plane Tracking. To find correspondences of two sets of plane instances, $\mathcal{P}^g_{i-1}$ and $\mathcal{P}_i$, we first need to compute their similarity scores $\mathbf{S}$. Inspired by [31], we found that integrating the contextual cues from all plane instances may boost matching accuracy. We therefore resort to the self-attention mechanism [39] to allow message passing between different plane instances. For the plane instance $P_{i,n}$, in addition to using its plane parameters $\mathbf{n}_{i,n}$ and $d_{i,n}$ (obtained from the cluster center), we also use the average pooled volume features $\bar{\mathbf{f}}_{i,n}$ from all the voxels in the cluster as an input to the attention module. We use self-attention and cross-attention graph networks, similar to [31], to get augmented feature vectors for the plane instances $P^g_{i-1,m}$ and $P_{i,n}$ as $\mathbf{h}^g_{i-1,m}$ and $\mathbf{h}_{i,n}$, respectively. The similarity score between them is then defined as

$$\mathbf{S}(m,n) = \langle \mathbf{h}^g_{i-1,m}, \mathbf{h}_{i,n} \rangle. \tag{2}$$

For plane tracking, we need to find an optimal matching matrix $\mathbf{M}^*$, where $\mathbf{M}(m,n) \in \{0,1\}$ such that

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \sum_{m,n} \mathbf{S}(m,n)\mathbf{M}(m,n). \tag{3}$$

The optimal matchings $\mathbf{M}^*$ can be efficiently solved using the Sinkhorn algorithm [9, 35], which is differentiable. We introduce its loss function in the supplementary material. As shown in Fig. 5, in practice, it is possible that some plane instances in $\mathcal{P}^g_{i-1}$ may not find their correspondences because a particular plane instance is not visible in the new fragment. We keep it in the updated global reconstruction $\mathcal{P}^g_i$. On the other hand, if a plane in $\mathcal{P}_i$ does not find its correspondence, it is likely that this is a new plane instance that is observed for the first time. Thus we initialize a new plane in $\mathcal{P}^g_i$.

3D Plane Refinement for Plane Fusion. To better leverage the temporal information, we refine plane parameters for the matched plane instances. Suppose $P^g_{i-1,m}$ and $P_{i,n}$ are two matched plane instances, the plane instance is updated as

$$P^g_{i,m} = \frac{\gamma P^g_{i-1,m} + P_{i,n}}{\gamma + 1}, \tag{4}$$

where γ is a parameter controlling the updating speed provided by a GRU. The GRU fuses plane features $\bar{\mathbf{f}}_{i,n}$ with $\bar{\mathbf{f}}^g_{i-1,m}$ and produces the updated plane features $\bar{\mathbf{f}}^g_{i,m}$, which will be passed through the MLP layers to predict γ. We slightly abuse notations here, $P^g_{i,m}$ can be either surface normal $\mathbf{n}^g_{i,m}$, plane offset $d^g_{i,m}$, or pooled volume feature vector $\bar{\mathbf{f}}^g_{i,m}$.

4. Experiments

In this section, we conduct a series of experiments to evaluate the 3D plane detection and reconstruction quality as well as different design considerations of PlanarRecon.

4.1. Setup

Datasets. We perform the experiments on ScanNetv2 [10]. The ScanNetv2 dataset contains RGB-D videos taken by a mobile device from 1613 indoor scenes. The camera pose is associated with each frame. As no ground truths are provided in test set, we follow PlaneRCNN [22] and generate 3D plane labels on the training and validation set. Our method is evaluated on two different validation sets with different scene splits used in previous works [23, 27].

Evaluation Metrics. We evaluate the performance of our method in terms of the 3D plane detection, which can be evaluated using instance segmentation metrics following previous works, and 3D reconstruction. For plane instance segmentation, due to the geometry difference between the ground truth and prediction meshes, we follow the semantic evaluation method proposed in [27]. More specifically, given a vertex in the ground truth mesh, we first locate its nearest neighbor in the predicted mesh and then transfer its prediction label. We employed three commonly used single-view plane segmentation metrics [22, 38, 42, 48] for our evaluation: rand index (RI), variation of information (VOI), and segmentation covering (SC). We also evaluate the geometry difference between predicted planes and ground truth planes. We densely sample points on the predicted planes and evaluate the 3D reconstruction quality using 3D geometry metrics presented by Murez *et al.* [27].

Baselines. Since there are no previous work that focus on learning-based multi-view 3D plane detection, we compare our method with three types of approaches: (1) single-view plane recovering [48]; (2) multi-view depth estimation [24] + depth-based plane detection [13]; and (3) volume-based 3D reconstruction [27, 37] + Sequential RANSAC [14].

Since baselines (1) and (2) predict planes for each view, we add a simple tracking module to merge planes predicted by the baseline in order to provide a fair comparison. The tracking and merging process we designed for our baselines is detailed in the supplementary material. We use the same key frames as in PlanarRecon for baselines (1) and (2). For (3), we first employ [27, 37] to estimate the 3D mesh of the scene, and perform sequential RANSAC to group the oriented vertices of the mesh into planes. Please refer to our supplementary material for the details of the sequential RANSAC algorithm. For [27, 37], we run sequential RANSAC every time when a new 3D reconstruction is completed to achieve incremental 3D plane detection.

4.2. Results

Geometric Accuracy. The 3D geometry evaluation results are shown in Tab. 1. Our method produces better performance than both single-view methods and multi-view methods. We believe the improvements come from volume-based 3D plane detection followed by a learning-based tracking and fusion module. Compared to single-view methods, PlanarRecon produces more accurate and globally coherent 3D planes as can be seen in Fig. 6. Our method has much higher performance than its single-view counterparts in terms of accuracy (Acc.), recall, precision (Prec.), and F-score. In comparison with the depth-based multi-view method, we outperform the baseline on all the geometry metrics. The volume representation allows our method to capture the local smoothness priors, leading to locally coherent results compared to estimating geometry per frame and then fuse them later in the depth-based method. Besides, the plane tracking module, which can be jointly learned with the 3D plane detector, is more robust than a hand-crafted tracking mechanism, resulting in globally coherent 3D plane detection results. Our method also surpasses the volumetric baselines in terms of accuracy, recall, precision, and F-score. The improvements potentially come from the fact that the model can be end-to-end trained, leading the network to learn features that better adapt to scene complexity.

Instance Segmentation Accuracy. The 3D plane instance segmentation evaluation results are presented in Tab. 2. Our method outperforms the baselines with sequential RANSAC across all the metrics. Sequential RANSAC tends to mis-group planar surfaces that are parallel and close to each other. For the single-view method, since it processes each frame independently, more plane instances are actually detected, which leads to a higher recall and so higher RI. However, its performance is worse than ours in terms of VOI and SC. Because these two metrics focus on informa-

Method	validation set	Comp ↓	Acc ↓	Recall ↑	Prec ↑	F-score ↑	Max Mem. (GB) ↓	Time (ms/keyframe) ↓
NeuralRecon [37] + Seq-RANSAC	Atlas [27]	0.144	0.128	0.296	0.306	0.296	**4.39**	586
Atlas [27] + Seq-RANSAC		**0.102**	0.190	0.316	0.348	0.331	25.91	848
ESTDepth [24] + PEAC [13]		0.174	0.135	0.289	0.335	0.304	5.44	101
Ours		0.154	**0.105**	**0.355**	**0.398**	**0.372**	4.43	**40**
PlaneAE [48]	PlaneAE [48]	**0.128**	0.151	0.330	0.262	0.290	6.29	**32**
Ours		0.143	**0.098**	**0.372**	**0.412**	**0.389**	**4.43**	40

Table 1. 3D geometry metrics on ScanNet. Our method outperforms the compared approaches by a significant margin in almost all metrics. ↑ indicates bigger values are better, ↓ the opposite. The best numbers are in bold. We use two different validation sets following Atlas [27] (top block) and PlaneAE [48] (bottom block).

Method	VOI ↓	RI ↑	SC ↑
NeuralRecon [37] + Seq-RANSAC	8.087	0.828	0.066
Atlas [27] + Seq-RANSAC	8.485	0.838	0.057
ESTDepth [24] + PEAC [13]	4.470	0.877	0.163
Ours	**3.622**	**0.897**	**0.248**
PlaneAE [48]	4.103	**0.908**	0.188
Ours	**3.622**	0.898	**0.247**

Table 2. 3D plane segmentation metrics on ScanNet. Our method also outperforms competing approaches in almost all metrics when evaluating plane segmentation metrics. ↑ indicates bigger values are better, ↓ the opposite. The best numbers are in bold. We use two different validation sets following Atlas [27] (top block) and PlaneAE [48] (bottom block).

Method	F-score ↑	VOI ↓	RI ↑	SC ↑
Ours, full approach	**0.372**	**3.622**	**0.897**	**0.248**
w/o plane branch	0.349	3.839	0.883	0.226
w/o voting branch	0.148	4.970	0.715	0.154
w/o learning-base fusion	0.358	3.798	0.897	0.222
w/o learning-base tracking	0.362	3.639	0.893	0.247

Table 3. Ablation studies. We report 3D geometry metrics F-score and 3D plane segmentation metrics on ScanNet.

tion variation and segmentation overlapping between prediction and ground truth, which focus more on the overall segmentation quality.

Efficiency. We also report the average running time of the baselines and our method in Tab. 1. Only the inference time on key frames is computed. Like in [37], for volumetric methods (Atlas, NeuralRecon, and ours), the running time is obtained by dividing the time of the number of key frames in the local fragment. The running time for PlanarRecon is measured on an NVIDIA V100 GPU. When comparing to NeuralRecon + sequential RANSAC, Atlas + sequential RANSAC and ESTDepth + PEAC, we use the running time reported in their paper [13, 24, 27, 37, 48]. As shown in Tab. 1, our runtime is 40ms per key frame, which corresponds to a frame rate of 25 key frames per second and outperforms most of our baselines. Specifically, our method runs ∼**2× faster** than ESTDepth [24] + PEAC [13] and ∼**15× faster** than NeuralRecon [37] + Seq-RANSAC. Similar to [37], we use volumetric representations, which can remove redundant computation in depth-based multi-view or single-view methods. Compared to other volume-based methods, our method can directly predict planes via a neural network, avoiding time-consuming RANSAC.

We report the maximum GPU memory usage in the inference stage in Tab. 1. Because our method uses the sparse volume as representation of scene, the GPU memory cost is a function of the surface area of the scene. Based on our experiments on ScanNet, PlanarRecon can reconstruct scenes using up to 4.43 GB GPU memory. In the training stage, PlanarRecon uses up to 22.87 GB GPU memory.

Qualitative Results. We provide the qualitative results in Fig. 6. To visualize NeuralRecon, we project the vertices to the planes they belong to and keep the edge of the mesh. For other methods, we can first get points from the depth map (baselines) or voxel center (ours). Then we project the point to the planes they belong to and generate the mesh using Delaunay triangulation.

Notice how our method produce a smaller amount of planes that better summarize the geometry of the scene. NeuralRecon + sequential RANSAC generates coherent results because the sequential RANSAC runs on the entire reconstructed mesh in each time step, which is *significantly* more time-consuming. Besides, this RANSAC-based approach is sensitive to the hyper-parameters choice. The ground in the third row is split into two planes because of the relatively small distance threshold. But if we set it with a larger value, some close planes tend to merge into one instance. PlaneAE and ESTDepth + PEAC suffer from obtaining coherent results. Due to the inconsistent predictions among different views, it is hard to match their predictions along with the whole video, leading to many false plane detections and inaccurate scene geometry.

4.3. Ablation Study

We also conduct several ablation experiments on the ScanNet dataset. The ablation study is shown in Tab. 3.

Plane branch. To validate the plane branch, we only use the shifted voxels $\mathbf{x}'_m$ to group voxels to form plane instances in local fragment volume $\mathcal{F}_i$. As shown in Tab. 3, the full approach can separate different plane instances accurately. Comparing visualization results (a) and (c)

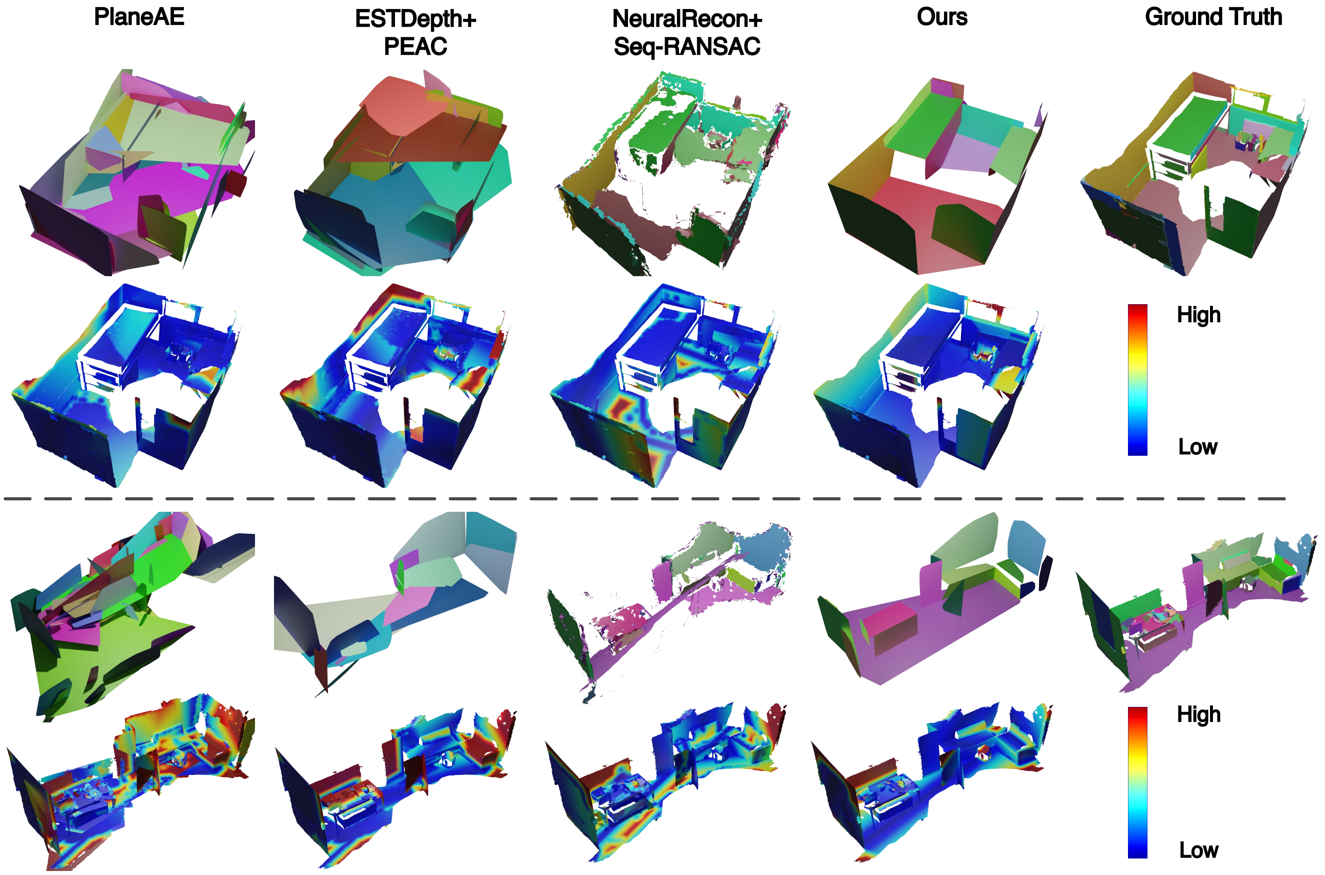

Figure 6. Qualitative results on ScanNet. Colored planes mean different instances. We present the error map for each method. When compared to other methods, PlanarRecon is capable of producing consistent sets of planes that better summarize and accurately represent the scene geometry. *Zoom in for details*.

in Fig. 4 also shows that the plane reconstruction result without the plane branch is prone to producing inaccurate planes.

Voting branch. We validate the voting branch by removing this module. As shown in Tab. 3 and Fig. 4(b), both geometry accuracy and segmentation accuracy drop rapidly. The results demonstrate that the voting branch is effective to get accurate 3D plane segments, especially when two planar surfaces are in the same plane.

Learning-based Tracking and Fusion. We track planes directly using the IoU of two planes instead of learning-based tracking. The IoU measures the intersection over a union of two sets of voxels which correspond to two planes. As shown in Tab. 3, the F-score has 1% improvement when the learning-based tracking is used. The 3D instance segmentation results are improved due to more robust matches. We also removed the learning-based fusion and instead directly use a fixed weight γ. As shown in Tab. 3, the 3D geometry metrics are improved with the fusion module.

4.4. Limitations and Failure Cases

Our approach assumes the existence of planar structures in the scene, which are very common in indoor scenes. But in scenarios where such planar surfaces do not exist, for instance, nature scenes like mountains, forests, etc., our approach may fail. We aim to build a planar simplification of the scene, which unavoidably introduces certain deviations if the surfaces are not perfectly planes. As shown in Fig. 6, the error mainly comes from the small planar areas and the areas on the edges.

5. Conclusion

In this work, we introduced a novel system, PlanarRecon, for real-time 3D plane detection and reconstruction with posed monocular video. The key idea is to use a volumetric representation of the scene and learning-based tracking and fusion module to detect, match and fuse planes in 3D for each video fragment incrementally. This design enables PlanarRecon to produce accurate and globally coherent 3D planes in real-time. Experiments show that PlanarRecon outperforms state-of-the-art methods and runs in real-time. The global 3D planes detected by PlanarRecon can be directly used in downstream applications like AR/VR.

Acknowledgement. YX and HJ would like to acknowledge the support from Google Cloud Program Credits.

References

[1] Augmented Reality with ARKit- Apple Developer. 1

[2] Caroline Baillard and Andrew Zisserman. Automatic reconstruction of piecewise planar models from multiple views. In *CVPR*, 1999. 2

[3] Adrien Bartoli. A random sampling strategy for piecewise planar scene segmentation. *CVIU*, 2007. 2

[4] Michael Bleyer, Christoph Rhemann, and Carsten Rother. Patchmatch stereo-stereo matching with slanted support windows. In *BMVC*, 2011. 3

[5] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. *ArXiv*, 2020. 1

[6] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *CVPR*, 2020. 3

[7] Alejo Concha and Javier Civera. DPPTAM: dense piecewise planar tracking and mapping from a monocular sequence. In *IROS*, 2015. 2

[8] James Coughlan and Alan L Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. *Advances in Neural Information Processing Systems*, 2000. 2

[9] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, 2013. 5

[10] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 2, 6

[11] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, 2018. 3

[12] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In *CVPR*, 2021. 3

[13] Chen Feng, Yuichi Taguchi, and Vineet R Kamat. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. In *ICRA*, 2014. 6, 7

[14] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 1981. 1, 6

[15] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Manhattan-world stereo. In *CVPR*, 2009. 2

[16] David Gallup, Jan-Michael Frahm, and Marc Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *CVPR*, 2010. 2

[17] Yuxin Hou, Juho Kannala, and Arno Solin. Multi-view stereo by temporal nonparametric fusion. In *ICCV*, 2019. 3

[18] Mengqi Ji, Jinzhi Zhang, Qionghai Dai, and Lu Fang. Surfacenet+: An end-to-end 3d neural network for very sparse multi-view stereopsis. *PAMI*, 2020. 4

[19] Linyi Jin, Shengyi Qian, Andrew Owens, and David F. Fouhey. Planar surface reconstruction from sparse views. In *ICCV*, 2021. 1, 3

[20] Jae Yong Lee, Joseph DeGol, Chuhang Zou, and Derek Hoiem. Patchmatch-rl: Deep mvs with pixelwise depth, normal, and visibility. In *ICCV*, 2021. 3

[21] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural rgb-> d sensing: Depth and uncertainty from a video camera. In *CVPR*, 2019. 3

[22] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *CVPR*, 2019. 1, 2, 3, 4, 6

[23] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. PlaneNet: Piece-Wise Planar Reconstruction from a Single RGB Image. In *CVPR*, 2018. 1, 2, 6

[24] Xiaoxiao Long, Lingjie Liu, Wei Li, Christian Theobalt, and Wenping Wang. Multi-view depth estimation using epipolar spatio-temporal networks. In *CVPR*, 2021. 6, 7

[25] Xinjun Ma, Yue Gong, Qirui Wang, Jingwei Huang, Lei Chen, and Fan Yu. Epp-mvsnet: Epipolar-assembling based depth prediction for multi-view stereo. In *ICCV*, 2021. 3

[26] 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. In *ECCV*, 2020. 3

[27] 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 *ECCV*, 2020. 3, 6, 7

[28] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *ICCV*, 2019. 5

[29] Yiming Qian and Yasutaka Furukawa. Learning pairwise inter-plane relations for piecewise planar reconstruction. In *ECCV*, 2020. 3

[30] Tong Qin, Jie Pan, Shaozu Cao, and Shaojie Shen. A General Optimization-based Framework for Local Odometry Estimation with Multiple Sensors. *ArXiv*, 2019. 1

[31] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020. 5

[32] Yifei Shi, Kai Xu, Matthias Nießner, Szymon Rusinkiewicz, and Thomas Funkhouser. Planematch: Patch coplanarity prediction for robust rgb-d reconstruction. In *ECCV*, 2018. 3

[33] Ayan Sinha, Zak Murez, James Bartolozzi, Vijay Badrinarayanan, and Andrew Rabinovich. Deltas: Depth estimation by learning triangulation and densification of sparse points. In *ECCV*, 2020. 3

[34] Sudipta N. Sinha, Drew Steedly, and Richard Szeliski. Piecewise planar stereo for image-based rendering. In *ICCV*, 2009. 2

[35] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 1967. 5

[36] Cheng Sun, Chi-Wei Hsiao, Ning-Hsu Wang, Min Sun, and Hwann-Tzong Chen. Indoor panorama planar 3d reconstruction via divide and conquer. In *CVPR*, 2021. 3

[37] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *CVPR*, 2021. 1, 3, 4, 6, 7

[38] Bin Tan, Nan Xue, Song Bai, Tianfu Wu, and Gui-Song Xia. Planetr: Structure-guided transformers for 3d plane recovery. In *ICCV*, 2021. 1, 2, 6

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 5

[40] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *CVPR*, 2021. 3

[41] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, 2021. 3

[42] Fengting Yang and Zihan Zhou. Recovering 3D Planes from a Single Image via Convolutional Neural Networks. In *ECCV*, 2018. 1, 2, 6

[43] Jiayu Yang, Wei Mao, Jose M Alvarez, and Miaomiao Liu. Cost volume pyramid based depth inference for multi-view stereo. In *CVPR*, 2020. 3

[44] Shichao Yang and Sebastian A. Scherer. Monocular object and plane SLAM in structured environments. *RA-L*, 2019. 2

[45] Shichao Yang, Yu Song, Michael Kaess, and Sebastian Scherer. Pop-up SLAM: semantic monocular plane SLAM for low-texture environments. In *IROS*, 2016. 2

[46] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, 2018. 3

[47] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multiview stereo depth inference. In *CVPR*, 2019. 3

[48] Zehao Yu, Jia Zheng, Dongze Lian, Zihan Zhou, and Shenghua Gao. Single-image piece-wise planar 3d reconstruction via associative embedding. In *CVPR*, 2019. 1, 2, 5, 6, 7