PLANING: A Loosely Coupled Triangle-Gaussian Framework for Streaming 3D Reconstruction

Changjian Jiang[1,2,3*] **Kerui Ren**[4,2*] **Xudong Li**[5,2] **Kaiwen Song**[6,2] **Guanghao Li**[7]
Linning Xu[8,2] **Tao Lu**[2] **Junting Dong**[2] **Yu Zhang**[1†] **Yu Feng**[9] **Bo Dai**[3,10] **Mulin Yu**[2†]

[1]Zhejiang University [2]Shanghai Artificial Intelligence Laboratory [3]The University of Hong Kong
[4]Shanghai Jiao Tong University [5]Northwestern Polytechnical University
[6]University of Science and Technology of China [7]Fudan University
[8]The Chinese University of Hong Kong [9]Zhejiang University of Technology
[10]HKU Shanghai Intelligent Computing Research Center
changjianjiang01@gmail.com, zhangyu80@zju.edu.cn, yumulin@pjlab.org.cn

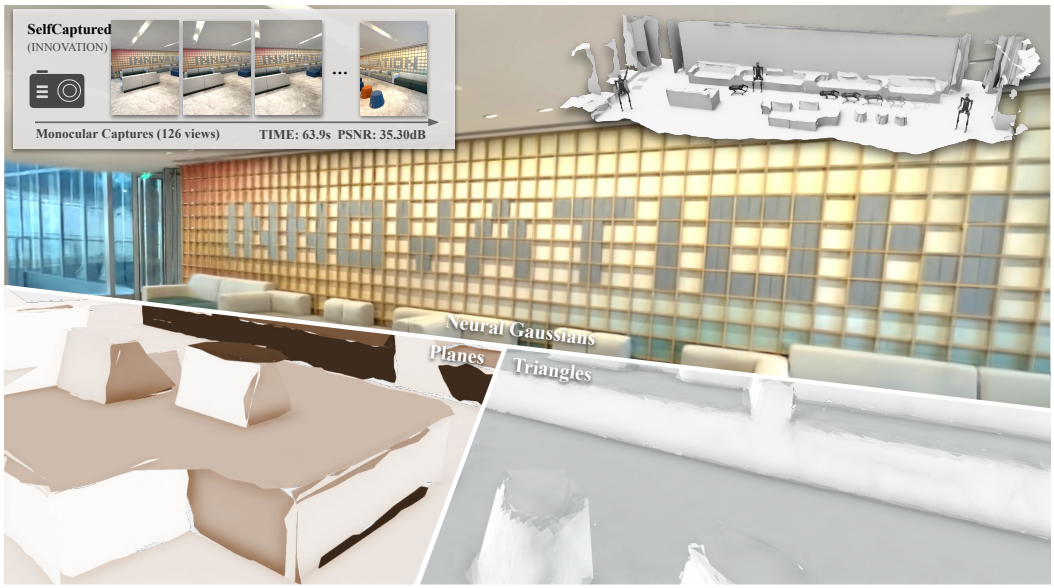

Figure 1: PLANING introduces a loosely coupled triangle-Gaussian representation for streaming 3D reconstruction, balancing geometric accuracy, high-fidelity rendering, and computational efficiency. Compared with 2DGS-based representations, explicit edge-preserving triangle primitives provide sharper structural boundaries and more stable geometric learning, leading to improved geometric reconstruction quality. The extracted compact planar structures further support high-performance simulation environments for locomotion training in embodied AI. Project page: **https://city-super.github.io/PLANING/**.

Abstract

Streaming reconstruction from monocular image sequences remains challenging, as existing methods typically favor either high-quality rendering or accurate geometry, but rarely both. We present PLANING, an efficient on-the-fly reconstruction framework built on a hybrid representation that loosely couples explicit geometric primitives with neural Gaussians, enabling geometry and appearance to be modeled

* Equal contribution. † Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

in a decoupled manner. This decoupling supports an online initialization and optimization strategy that separates geometry and appearance updates, yielding stable streaming reconstruction with substantially reduced structural redundancy. Despite operating online, PLANING improves dense mesh Chamfer-L2 by 18.52% over the offline baseline PGSR, surpasses the streaming baseline ARTDECO by 1.31 dB in PSNR on average across the six evaluated datasets, and reconstructs ScanNetV2 scenes in 2.1 minutes, over 5× faster than offline 2D Gaussian Splatting while matching per-scene optimization quality. Beyond reconstruction quality, the structural clarity and computational efficiency of PLANING make it well suited for a broad range of downstream applications, such as enabling large-scale scene modeling and simulation-ready environments for embodied AI.

1 Introduction

3D scene reconstruction is a core capability for embodied intelligence, autonomous driving, and AR/VR, providing the spatial understanding required for perception and interaction [39, 27]. While offline reconstruction methods following a capture-then-process paradigm have reached a high level of maturity, their reliance on time-intensive post-processing limits scalability and responsiveness in time-critical scenarios. This has driven a growing demand for on-the-fly reconstruction frameworks.

A central challenge in on-the-fly 3D reconstruction is a scene representation that jointly achieves high geometric accuracy and computational efficiency. Recently, 3D Gaussian Splatting (3DGS) [16] has emerged as a compelling explicit representation, offering high visual fidelity with efficient rendering, and has therefore been widely adopted in streaming reconstruction methods [23, 18, 4, 15, 22, 45]. Despite their success, existing streaming 3DGS-based methods share a fundamental limitation: *the absence of explicit, compact, and stable geometry.* While Gaussian primitives excel at appearance modeling, their ambiguous structural boundaries hinder stable surface reconstruction and geometric consistency. To compensate, these methods rely on a large number of primitives, leading to significant redundancy, increased computational cost, and limited scalability in streaming settings.

To address these challenges, (1) we develop a hybrid representation that decouples geometry from appearance for efficient reconstruction and high-fidelity rendering. For *geometry*, we introduce learnable triangle primitives with explicit edge definitions, enabling sharper structural boundaries and more stable geometric learning than 2DGS-based representations, which leads to improved geometric reconstruction quality. For *appearance modeling*, we adopt a neural-Gaussian formulation inspired by Scaffold-GS [20], where Gaussian attributes are decoded from fused triangle and Gaussian features to encourage local rendering consistency. This design establishes a complementary interaction: triangles serve as stable structural anchors, while rendering gradients propagated through neural Gaussians refine the underlying geometry in a controlled manner. Building upon this representation, (2) we introduce PLANING, an efficient streaming framework for monocular 3D reconstruction. Our framework leverages feed-forward geometric priors for robust pose estimation and stable scene initialization, while photometric and spatial filtering suppress redundant primitives. Furthermore, global map update enhances global geometric consistency during streaming reconstruction.

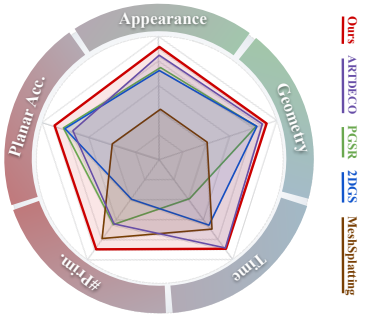

Figure 2: PLANING consistently outperforms existing streaming and per-scene reconstruction methods across geometry accuracy, rendering quality, computational efficiency, and primitive count, while maintaining clear and well-structured planar geometry.

Extensive experiments across diverse indoor and outdoor benchmarks demonstrate that our method outperforms state-of-the-art approaches in geometric accuracy, rendering quality, training efficiency, and primitive count, as illustrated in Fig. 1 and Fig. 2. By preserving salient structures and suppressing redundant geometry, our representation enables the extraction of compact and consistent 3D planes with significantly reduced triangle counts. This compact geometric representation is particularly beneficial for large-scale reconstruction and globally consistent pose estimation. Furthermore, the structural clarity and efficiency of our model make it well suited for simulation-ready scene modeling,

such as supporting local motion policy training in embodied AI. Our main contributions can be summarized as follows:

(1) We develop **a hybrid representation** that decouples geometry and appearance by combining explicit triangle primitives with neural Gaussians, enabling compact and stable geometry learning and high-fidelity rendering.

(2) We develop **an efficient on-the-fly monocular reconstruction framework** that leverages the proposed representation together with streaming-aware initialization and global map update.

(3) We demonstrate **state-of-the-art performance** in both geometric accuracy and rendering quality across diverse indoor and outdoor benchmarks, and showcase the versatility of our approach for downstream tasks including plane-guided pose refinement, large-scale scene reconstruction, and simulation-ready environments for embodied AI.

2 Related Work

3D Reconstruction and SLAM. Reconstructing 3D geometry from multi-view observations is a fundamental problem spanning offline reconstruction and online SLAM. Classical pipelines [30, 25, 2, 28] rely on calibrated images to recover camera poses and sparse geometry, but often lack the fidelity required for high-quality rendering. Neural rendering methods, notably NeRF [24], model scenes as continuous fields and enable photorealistic synthesis, yet their implicit formulation and costly volumetric sampling limit scalability and real-time applicability.

To improve efficiency, 3D Gaussian Splatting (3DGS) [16] introduces explicit primitives with rasterization-based rendering, enabling real-time reconstruction and making it attractive for streaming scenarios [22, 15, 9]. Recent works integrate 3DGS into SLAM pipelines by jointly optimizing camera poses and scene representations via rendering supervision, or by adopting incremental updates with lightweight bundle adjustment [45, 23]. Despite these advances, existing approaches struggle to simultaneously achieve robustness, geometric accuracy, and efficiency, particularly in monocular and long-sequence settings.

An alternative paradigm leverages feed-forward models pretrained on large-scale datasets [26, 35, 34, 19, 48] to directly infer scene structure with or without known poses. While these methods offer strong generalization and fast inference, they typically lag behind optimization-based approaches in reconstruction accuracy and global consistency. Overall, achieving a unified representation that balances geometric fidelity, efficiency, and robustness remains an open challenge.

3DGS Variants. Various extensions have explored alternative primitives to better align with scene geometry. 2DGS [13], GSS [6], and Quadratic Gaussian Splatting [46] replace anisotropic Gaussians with ellipsoidal or quadric forms for superior surface alignment. Other works incorporate explicit geometric elements, such as the 3D convexes [11] and triangles [14, 10, 1], to compactly model hard-edged scenes. Similarly, PlanarSplatting [33] utilizes rectangular primitives to achieve structured and efficient indoor planar reconstructions. Despite these advances, single-representation methods often struggle to balance geometric precision with rendering fidelity. To bridge this gap, recent dual-branch approaches such as GSDF [43] and 3DGSR [21] integrate neural signed distance fields (SDFs) with 3DGS. While this enables partial geometry–appearance decoupling, it introduces significant computational overhead and optimization complexity. Alternatively, 3D-GES [40] adopts a bi-scale formulation using 2D surfels for coarse structure and 3D Gaussians for fine detail. However, this design primarily targets appearance enhancement rather than achieving a principled, explicit decoupling of geometry and appearance.

3 Method

In this section, we first introduce our dual scene representation that combines learnable triangles with neural Gaussians (Sec. 3.1). We then describe how we adapt this representation into an on-the-fly reconstruction framework, achieving both efficiency and high-quality 3D reconstruction (Sec. 3.2).

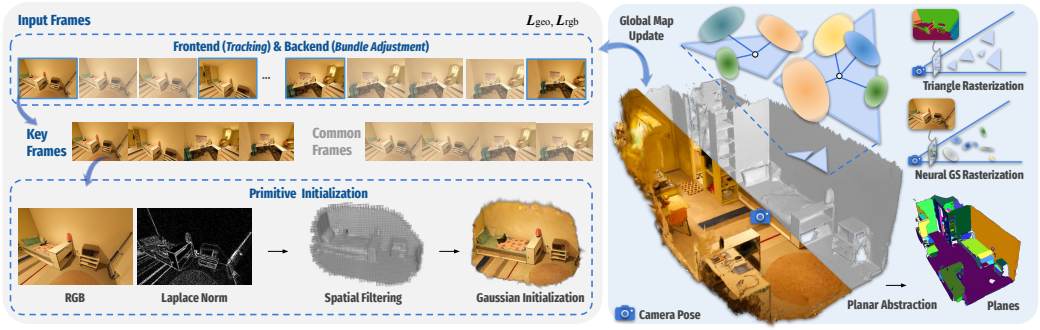

Figure 3: **Pipeline of PLANING.** PLANING adopts a hybrid representation in which triangles explicitly model scene geometry, while neural Gaussians decoded from these triangles render appearance. Built upon this representation, we develop a streaming reconstruction framework for unposed monocular image sequences, consisting of a frontend for camera tracking, a backend for global pose optimization, and a mapper for scene reconstruction. The mapper further incorporates an efficient primitive initialization strategy to suppress redundancy. The reconstructed triangle soup additionally enables compact planar abstraction for downstream applications.

3.1 Loosely-coupled Triangle-Gaussian Representation

We first detail the triangle primitives and our differentiable rasterizer. Subsequently, we explain the interaction between neural Gaussians and their corresponding triangles.

3.1.1 Learnable Triangles for Geometry

We propose triangle primitives based on a vertex-based formulation and a differentiable rasterizer.

Vertex-Based Primitive Definition. We parameterize each triangle primitive by its three learnable vertices $\{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2\}$. To facilitate efficient and differentiable rendering, we define a local coordinate frame for each triangle:

$$\begin{aligned} \mathbf{t}_u &= \frac{\mathbf{p}_0 - \boldsymbol{\mu}}{\|\mathbf{p}_0 - \boldsymbol{\mu}\|_2}, \mathbf{t}_v = \mathbf{n} \times \mathbf{t}_u, s_u = \|\mathbf{p}_0 - \boldsymbol{\mu}\|_2, \\ s_v &= |\mathbf{t}_v \cdot (\mathbf{p}_1 - \boldsymbol{\mu})|, \mathbf{n} = \frac{(\mathbf{p}_1 - \mathbf{p}_0) \times (\mathbf{p}_2 - \mathbf{p}_0)}{\|(\mathbf{p}_1 - \mathbf{p}_0) \times (\mathbf{p}_2 - \mathbf{p}_0)\|_2}, \end{aligned} \tag{1}$$

where the barycenter $\boldsymbol{\mu}$ is set as the origin of the local frame. Under this construction, the three vertices can be expressed in the local tangent plane as $\{\mathbf{p}'_0, \mathbf{p}'_1, \mathbf{p}'_2\} = \{(0,1)^T, (a,1)^T, (-1-a,-1)^T\}$, where $a = \mathbf{t}_u \cdot (\mathbf{p}_1 - \boldsymbol{\mu})$ is the only degree of freedom in the local frame.

Following 3D Convex Splatting (3DCS) [11], we further introduce two learnable triangle-wise parameters, $\delta > 0$ and $\sigma > 0$, to control edge sharpness and boundary smoothness. Each triangle is also associated with a learnable opacity parameter α, analogous to 3DGS [16].

Differentiable Triangle Rasterizer. We implement an efficient differentiable triangle rasterizer that enables direct supervision of triangles using prior normals and depths. To obtain unbiased depth rendering, we adopt an explicit ray-triangle intersection strategy, similar in spirit to 2DGS [13]. We further introduce the edge-preserving contribution function as:

$$w(\hat{\mathbf{x}}) = \text{Sigmoid}\left(-\sigma \log\left(\sum_{j=0}^{2} \exp\left(\delta \, \text{dist}(\hat{\mathbf{x}}, \mathbf{e}_j)\right)\right)\right) \alpha, \tag{2}$$

where $\text{dist}(\hat{\mathbf{x}}, \mathbf{e}_j)$ denotes the distance from the intersection point $\hat{\mathbf{x}}$ to the j-th triangle edge in the local tangent plane. Thanks to the local frame parameterization, these distances can be computed analytically as:

$$\begin{cases} \text{dist}(\hat{\mathbf{x}}, \mathbf{e}_0) = u + (1-a)v - 1, \\ \text{dist}(\hat{\mathbf{x}}, \mathbf{e}_1) = -2u + (2a+1)v - 1, \\ \text{dist}(\hat{\mathbf{x}}, \mathbf{e}_2) = u + (-2-a)v - 1, \end{cases} \tag{3}$$

where $\hat{\mathbf{x}} = (u, v)^T$. This closed-form formulation significantly simplifies both forward evaluation and gradient propagation. Notably, our contribution computation differs from 3DCS, where contributions are computed directly on the image plane rather than in the local surface domain.

Finally, triangles are rendered into depth and normal maps using front-to-back alpha compositing:

$$\mathbf{N}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{n}_i w\left(\hat{\mathbf{x}}_i\right) \prod_{j=1}^{i-1} \left(1 - w\left(\hat{\mathbf{x}}_j\right)\right), \quad \mathbf{D}(\mathbf{x}) = \sum_{i=1}^{N} d_i w\left(\hat{\mathbf{x}}_i\right) \prod_{j=1}^{i-1} \left(1 - w\left(\hat{\mathbf{x}}_j\right)\right), \tag{4}$$

where d_i denotes the distance from the i-th intersection point to the pixel. The N ordered intersection points $\{\hat{\mathbf{x}}_i\}$ between the triangles and pixel $\mathbf{x}$ are computed using our custom CUDA-based rasterizer. To enable accurate differentiable rendering of triangles, we define a new criterion for visibility determination and design a triangle-subdivision-based primitive depth sorting algorithm in the rasterizer to address rendering issues introduced by the edge-preserving contribution function . The detailed forward rendering pipeline is described in the Appendix A.1.

3.1.2 Neural Gaussians for Appearance Modeling

To achieve a decoupled yet consistent representation of geometry and appearance, we introduce neural Gaussians to flexibly encode view-dependent appearance. Inspired by Scaffold-GS [20], neural Gaussians are anchored to the triangles and used for appearance. Specifically, each learnable triangle is associated with a context feature $\mathbf{f}_\mathrm{t} \in \mathbb{R}^{24}$. Each Gaussian is parameterized by a learnable position offset $\mathbf{o}_\mathrm{g} \in \mathbb{R}^3$, spherical harmonics (SH) coefficients, an opacity $\alpha_\mathrm{g} \in \mathbb{R}$, a base scale $\mathbf{s}_\mathrm{g} \in \mathbb{R}^3$, a base quaternion $\mathbf{q}_\mathrm{g} \in \mathbb{R}^4$, and an individual feature $\mathbf{f}_\mathrm{g} \in \mathbb{R}^8$.

During rendering, the position of each Gaussian $\boldsymbol{\mu}_\mathrm{g} = \mathbf{o}_\mathrm{g} + \boldsymbol{\mu}_\mathrm{t}$, where $\boldsymbol{\mu}_\mathrm{t}$ denotes the barycenter of the associated triangle. Then we predict the final scale $\mathbf{s} = \mathbf{s}_\mathrm{g} \odot \mathrm{MLP}_s(\mathbf{f}_\mathrm{t} \oplus \mathbf{f}_\mathrm{g})$ and rotation $\mathbf{q} = \phi(\mathbf{q}_\mathrm{g} \odot \mathrm{MLP}_q(\mathbf{f}_\mathrm{t} \oplus \mathbf{f}_\mathrm{g}))$, where $\odot$ denotes element-wise multiplication, $\oplus$ denotes feature concatenation and $\phi(\cdot)$ denotes ℓ_2 normalization to ensure valid rotation quaternions. Through this design, geometry and appearance are represented in a consistent manner. Notably, each triangle hosts a flexible number of Gaussians, enabling the representation to adapt to local scene details.

3.2 Streaming Reconstruction Framework

3.2.1 Overview

As shown in Fig. 3, we build a streaming monocular reconstruction framework upon the proposed hybrid representation. Following [18], the system consists of a frontend for camera tracking, a backend for global pose optimization, and a mapper for scene reconstruction.

The frontend processes incoming frames in a streaming manner to estimate camera poses, select keyframes, and predict dense point maps using feed-forward models [17]. The backend performs loop closure detection [35] and global bundle adjustment [26] to improve global pose consistency. Given the optimized poses and geometric priors from the backend, the mapper incrementally reconstructs scene geometry and appearance using the proposed triangle-Gaussian representation. The resulting map is continuously available throughout the streaming process.

To improve efficiency and geometric consistency, we introduce redundancy-aware primitive initialization and global map update during streaming reconstruction. After all input frames have been processed, we apply a final global refinement to the existing map to improve global consistency and rendering quality. Following the final refinement, planar structures can be directly extracted from the triangle soup via a coarse-to-fine plane extraction algorithm. Furthermore, our framework supports dense mesh reconstruction through depth fusion. Details are provided in Appendices A.4–A.6.

3.2.2 Primitive Initialization

Upon receiving a keyframe from the backend, the framework initializes new primitives in regions with insufficient geometric coverage or high reconstruction error. To identify informative regions, we compute an insertion probability map based on the discrepancy between the rendered image and the observation using a Laplacian-of-Gaussian (LoG) operator [23]. New primitives are instantiated only when the insertion probability exceeds a stochastic threshold with upper bound τ_a, encouraging

Table 1: **Quantitative comparison of planar reconstruction.** We evaluate the geometric and planar metrics on the ScanNet++, ScanNetV2, and FAST-LIVO2 datasets. Ours achieves top-tier performance in most categories while significantly reducing primitive count and runtime (reported in minutes).

Method	ScanNet++							ScanNetV2							FAST-LIVO2					
	Geometry		Planar			Time	#Prim.	Geometry		Planar			Time	#Prim.	Geometry				Time	#Prim.
	Ch-L2↓	F-score↑	Fidelity↓	Acc↓	Ch-L2↓			Ch-L2↓	F-score↑	Fidelity↓	Acc↓	Ch-L2↓			Acc↓	Comp↓	Ch-L2↓	F-score↑		
2DGS†	3.89	81.64	8.16	7.19	7.67	16.1	415.3k	6.48	53.73	15.56	8.12	11.84	10.9	1196.8k	14.11	48.17	53.45	60.47	35.8	3197.0k
PGSR†	3.87	81.98	7.44	7.23	7.33	31.2	353.4k	6.59	54.28	15.88	8.46	12.17	21.3	629.1k	13.95	49.16	54.13	60.75	25.5	1065.8k
MeshSplatting†	9.13	47.19	37.87	10.71	24.29	38.5	1825k	11.15	30.73	40.16	11.45	25.81	9.7	291.3k	14.52	66.68	62.97	47.31	26.3	2505.1k
AirPlanes	25.19	19.21	47.10	25.97	36.53	3.7	/	6.34	55.33	9.68	8.90	9.29	3.5	/	-	-	-	-	-	-
PlanarSplatting	7.27	49.78	9.64	13.35	11.50	8.8	1.0k	6.54	51.67	9.72	10.77	10.24	3.1	1.76k	-	-	-	-	-	-
ARTDECO	3.82	83.08	15.84	7.92	11.88	5.6	478.3k	6.05	57.58	19.62	8.73	14.18	2.2	621.5k	14.17	63.63	57.19	54.23	6.5	501.6k
Ours	3.53	86.88	7.24	6.95	7.09	5.5	61.6k	5.68	62.15	10.55	7.58	9.07	2.1	56.1k	12.58	30.60	36.77	65.89	3.6	101.4k

/: w/o explicit geometric primitives, –: beyond the scope (indoor scenes) of the method, †: offline per-scene baselines are augmented with the same geometric-prior supervision as PLANING.

Table 2: **Quantitative comparison of appearance rendering.** We evaluate the rendering quality metrics across six diverse indoor and outdoor datasets. Our method achieves state-of-the-art performance in most categories while significantly reducing the runtime (reported in minutes).

Method	ScanNetV2			VR-NeRF			ScanNet++			Waymo			FAST-LIVO2			KITTI			Time
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	
2DGS†	27.74	0.873	0.234	29.61	0.905	0.200	32.00	0.937	0.129	26.97	0.854	0.324	29.24	0.867	0.287	22.29	0.751	0.338	31.9
PGSR†	27.73	0.880	0.233	29.37	0.903	0.201	31.38	0.937	0.133	27.42	0.865	0.306	29.22	0.870	0.280	22.88	0.785	0.284	39.9
MeshSplatting†	25.64	0.830	0.351	25.23	0.819	0.352	27.71	0.876	0.294	23.10	0.781	0.424	25.78	0.789	0.397	17.35	0.551	0.499	24.6
MonoGS	22.17	0.806	0.542	15.30	0.583	0.655	17.08	0.708	0.632	19.06	0.744	0.639	19.80	0.694	0.649	14.56	0.489	0.767	8.3
S3PO-GS	24.37	0.829	0.476	24.00	0.810	0.371	23.34	0.820	0.444	25.33	0.821	0.395	24.99	0.776	0.419	19.23	0.622	0.430	24.5
OnTheFly-NVS	23.33	0.823	0.376	29.10	0.895	0.237	21.54	0.794	0.357	27.22	0.848	0.300	21.92	0.735	0.443	17.17	0.584	0.427	1.3
ARTDECO	28.44	0.877	0.232	30.02	0.911	0.230	31.64	0.941	0.140	26.59	0.869	0.305	32.86	0.926	0.210	22.99	0.777	0.282	6.9
Ours	28.83	0.882	0.222	32.59	0.933	0.168	31.91	0.941	0.133	29.24	0.887	0.278	33.97	0.938	0.180	23.82	0.793	0.253	7.4

†: offline per-scene baselines are augmented with the same geometric-prior supervision as PLANING.

the model to focus on high-frequency and under-reconstructed regions. Implementation details are provided in Appendix A.2.

Spatial Filtering. To suppress structural redundancy and maintain map compactness, we further apply a depth-adaptive spatial filtering strategy. For each candidate primitive, we compute its back-projected 3D center and discard it if nearby primitives already exist within a local vicinity. The vicinity radius is adaptively scaled according to the observation depth:

$$V(d_i) = V_{\min} + (V_{\max} - V_{\min}) \left(\frac{d_i - d_{\min}}{d_{\max} - d_{\min}} \right)^p, \tag{5}$$

where d_i denotes the observation depth and $\{V_{\min}, V_{\max}, d_{\min}, d_{\max}, p\}$ are hyperparameters controlling the spatial extent. This strategy effectively prevents excessive primitive growth in already-reconstructed regions. Once a candidate location is selected, a triangle primitive is initialized using the local depth and normal priors. The triangle scale is determined according to the local geometric structure, while its orientation is aligned with the predicted surface normal.

Gaussian Initialization. To model appearance details, neural Gaussians are subsequently initialized around triangle barycenters. The number of Gaussians is adaptively determined based on local image complexity, enabling higher representational capacity in texture-rich regions. Gaussian attributes are initialized using local geometric and photometric observations.

3.2.3 Training

We supervise the triangles and Gaussians with separate geometric and appearance losses for decoupled optimization: $\mathcal{L} = \mathcal{L}_{\text{geo}} + \mathcal{L}_{\text{rgb}}$. For geometry, we leverage multi-view depth $\mathbf{D}_{\text{p}}$ and normal $\mathbf{N}_{\text{p}}$ priors from MASt3R [17] to supervise triangles, penalizing deviations from the rendered depth $\mathbf{D}_{\text{t}}$ and normals $\mathbf{N}_{\text{t}}$:

$$\mathcal{L}_{\text{geo}} = \lambda_{\text{d}} \|\mathbf{D}_{\text{t}} - \mathbf{D}_{\text{p}}\|_1 + \lambda_{\text{n}} \|\mathbf{N}_{\text{t}} - \mathbf{N}_{\text{p}}\|_1 + \lambda_{\text{o}} \mathcal{L}_{\text{o}}, \tag{6}$$

where λ_{d} and λ_{n} are user-prescribed weights. $\mathcal{L}_{\text{o}}$ is an entropy loss on triangle opacity α, following [8]. We regularly prune triangle primitives with $\alpha < 0.5$, which removes redundant geometry and

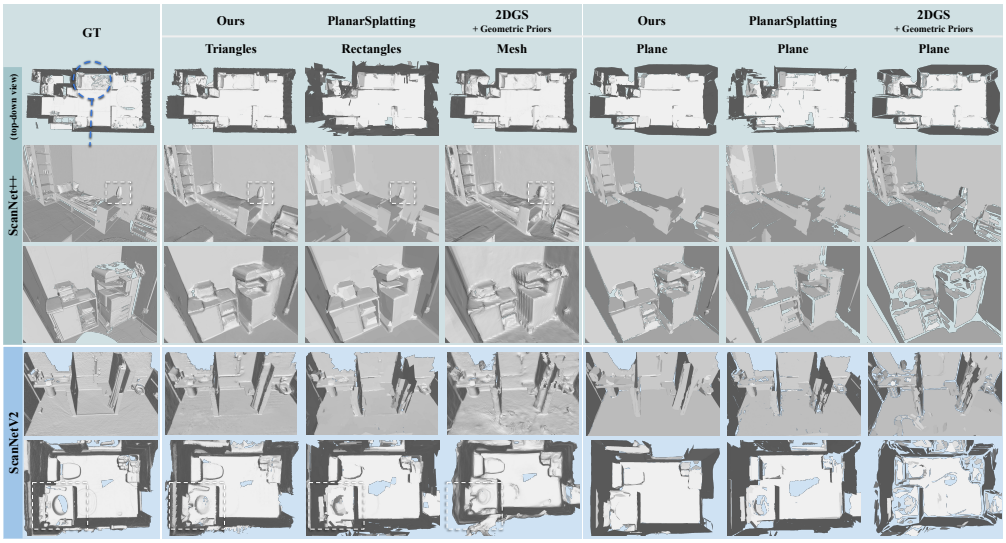

Figure 4: **Qualitative comparison of geometric reconstruction.** We visualize planar reconstruction and geometric modeling across different primitives, with 2DGS shown as dense mesh for comparison. Overall, our method preserves planar structures while capturing fine geometric details.

maintains a compact representation. The appearance loss supervises neural Gaussians via:

$$\mathcal{L}_{\text{rgb}} = (1-\lambda_{\text{c}})\|\mathbf{C}_{\text{gt}} - \mathbf{C}_{\text{gs}}\|_1 + \lambda_{\text{c}}\text{SSIM}\big(\mathbf{C}_{\text{gt}}, \mathbf{C}_{\text{gs}}\big) + \lambda_{\text{s}}\mathcal{L}_{\text{s}}, \tag{7}$$

where $\mathcal{L}_{\text{s}}$ is a volume regularization term adopted from Scaffold-GS [20]. Notably, appearance gradients from Gaussians are back-propagated to the triangles, enabling implicit refinement of the underlying geometry. More details are provided in Appendix A.3.

3.2.4 Global Map Update

In our streaming framework, camera poses are continuously refined within the backend, while primitives in the mapper are initialized and optimized using the poses available at that timestamp. This asynchronous update can lead to pose–model misalignment. To maintain consistency between the refined poses and the 3D model, we explicitly transform the primitives after the pose optimization. Specifically, we record the source keyframe for each primitive and apply a relative transformation $\Delta\mathbf{T} = \mathbf{T}_{\text{n}}\mathbf{T}_{\text{o}}^{-1}$ to its attributes when the corresponding keyframe pose changes from $\mathbf{T}_{\text{o}}$ to $\mathbf{T}_{\text{n}}$:

$$\mathbf{p}_{\text{t}}' = \Delta\mathbf{T}\mathbf{p}_{\text{t}}, \quad \mathbf{o}_{\text{g}}' = \Delta\mathbf{T}(\mathbf{o}_{\text{g}} + \boldsymbol{\mu}_{\text{t}}) - \boldsymbol{\mu}_{\text{t}}', \quad \mathbf{q}_{\text{g}}' = \mathcal{R}^{-1}(\Delta\mathbf{R}\mathcal{R}(\mathbf{q}_{\text{g}})), \tag{8}$$

where $\Delta\mathbf{R}$ is the rotation component of $\Delta\mathbf{T}$, and $\mathcal{R}(\cdot)$ maps quaternions to rotation matrices. Here, $\{\mathbf{p}_{\text{t}}', \boldsymbol{\mu}_{\text{t}}', \mathbf{o}_{\text{g}}', \mathbf{q}_{\text{g}}'\}$ denote the updated triangle and Gaussian parameters.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate PLANING on 56 real-world scenes from diverse benchmarks: 20 from ScanNet++ [41], 10 from ScanNetV2 [5], 6 from VR-NeRF [38], 4 from FAST-LIVO2 [47], 8 from KITTI [7], and 8 from Waymo [32], covering a wide range of indoor and outdoor environments.

Baselines. We compare PLANING with state-of-the-art methods across three categories. For per-scene reconstruction, we evaluate 2DGS [13], PGSR [3], and MeshSplatting [10]. For streaming reconstruction, we select ARTDECO [18], OnTheFly-NVS [23], S3PO-GS [4], and MonoGS [22]. For planar reconstruction, we include PlanarSplatting [33] and AirPlanes [37]. To ensure fair comparison, all per-scene reconstruction baselines are augmented with the same MASt3R geometric priors used in ours. For a controlled comparison, 2DGS, PGSR, MeshSplatting, AirPlanes, and PlanarSplatting are provided with the same camera poses estimated by PLANING, as they do not

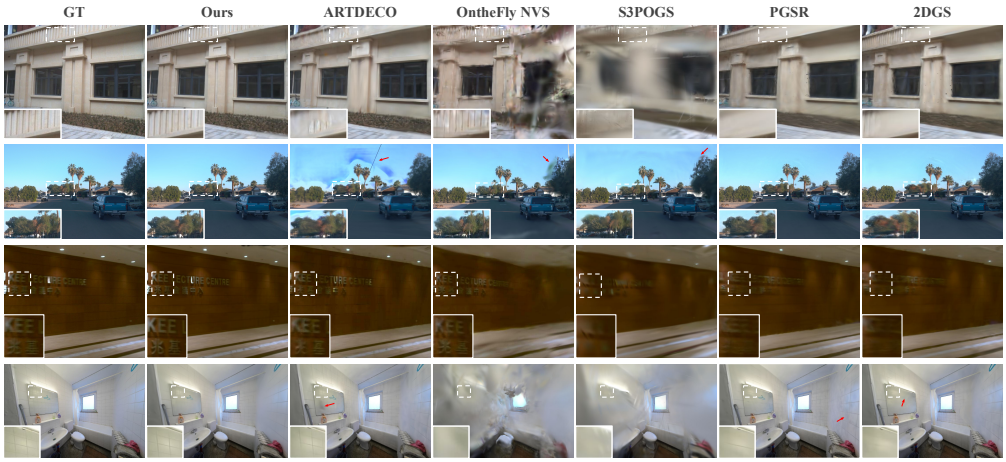

Figure 5: **Qualitative comparison of appearance rendering.** We evaluate our method against state-of-the-art approaches. White wireframes highlight regions where our method excels, faithfully reconstructing fine structures and more complete surfaces.

estimate camera poses within their reconstruction pipelines. Appendix C.4 shows that this shared pose input is sufficiently accurate for the reported controlled evaluation. In contrast, streaming baselines with native tracking are evaluated using their own estimated poses.

Metrics. We conduct a comprehensive evaluation of our framework across three tasks. For planar reconstruction, following PlanarSplatting [33], we evaluate plane geometry using Chamfer Distance and F-score. For datasets with ground-truth plane annotations, we further assess the top-20 largest planes using Planar Fidelity, Planar Accuracy, and Planar Chamfer metrics. For dense mesh reconstruction, we report Chamfer Distance and F-score, while for novel view synthesis (NVS), we use standard metrics including PSNR, SSIM [36], and LPIPS [44]. In addition, we report training time and the number of primitives to quantify computational efficiency.

Implementation Details. Following standard novel view synthesis practice, we hold out every eighth frame for evaluation. Test views are used only for pose estimation and final evaluation, but are excluded from Gaussian initialization and training. Following [18, 23], our method, ARTDECO [18], and OnTheFly-NVS [23] perform a 15k-iteration final global refinement over the maps constructed during streaming, whereas per-scene baselines are trained for 30k iterations. Additional implementation details are provided in Appendices A.6–A.8.

4.2 Results Analysis

Geometry Results. We first evaluate planar reconstruction against six baselines covering diverse scene representations, including triangles, 3D Gaussians, surfels, rectangles, and implicit planar embeddings. As shown in Tab. 1, our method consistently achieves the best accuracy, with the lowest Chamfer Distance and highest F-score, while maintaining a compact primitive count and fast training speed. As shown in Fig. 4, our hybrid representation preserves planar regularity and sharp geometric features by explicitly modeling planar structures with triangles. In contrast, rectangle-based representations, despite their compactness, lack the flexibility to capture fine-grained geometry, limiting their ability to model complex scene structures. Surfel-based methods, which tightly couple geometry and appearance, often suffer from appearance-induced distortions, resulting in uneven or erroneous surfaces even when geometric priors are applied. Detailed dense mesh reconstruction evaluations are provided in Appendix C.2, where our method achieves consistently improved accuracy.

Rendering Results. As shown in Tab. 2, our method achieves state-of-the-art rendering performance against both per-scene optimization and streaming reconstruction baselines. It shows clear advantages in texture-less and low-light regions (Fig. 5), where per-scene methods often suffer from overfitting or poor Gaussian initialization, and streaming methods are affected by pose drift. In contrast, our method leverages accurate geometry and robust feed-forward pose estimation, leading to more stable and consistent rendering quality.

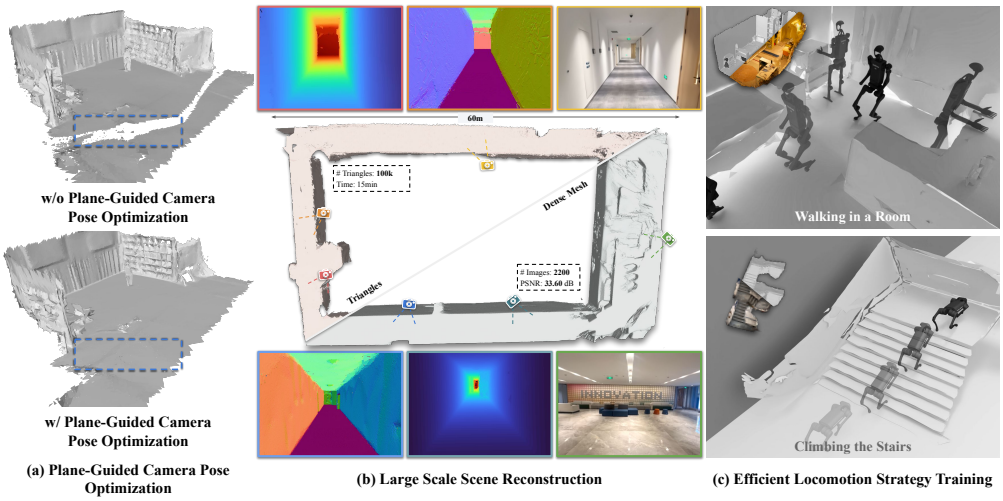

Figure 6: **Applications and system analysis.** From left to right: (a) plane-guided camera pose optimization, where planar constraints effectively reduce drift; (b) large-scale indoor reconstruction using over 2000 monocular images; (c) reinforcement learning in reconstructed environments, including humanoid locomotion and quadruped stair climbing.

Gain Attribution. Appendix A.8 (Tab. 4) isolates the effect of shared MASt3R supervision on the per-scene baselines. Although this augmentation improves the geometric accuracy of every baseline, PLANING retains a clear advantage under the same prior protocol. **PLANING's gains therefore cannot be attributed to geometric priors alone; instead, they arise from its hybrid representation and streaming-specific map management.** The hybrid representation decouples explicit structural geometry from appearance optimization, spatial filtering controls redundant map growth, and global map updates maintain geometric consistency under pose refinement. The ablations in Tab. 3 further confirm that each component contributes to overall reconstruction performance.

4.3 Applications

Plane-Guided Camera Pose Optimization. Most streaming reconstruction frameworks separate pose estimation from mapping, preventing effective use of global geometry and leading to drift. We feed the resulting planar map back to the frontend and refine camera poses with a point-to-plane alignment loss, improving global geometric consistency (Fig. 6(a)). Details are provided in Appendix B.1.

Large-Scale Scene Reconstruction. Large-scale reconstruction remains memory-intensive even with compact primitives. We adopt a dynamic loading strategy that swaps primitive parameters between GPU and CPU, allowing the framework to scale to large environments under limited GPU memory (Fig. 6(b)). Details are provided in Appendix B.2.

Downstream Locomotion Demonstration. To assess their downstream utility, we convert the compact planar assets reconstructed by our method into physics-based simulation environments for locomotion. In the evaluated Isaac Lab tasks, their structurally consistent contact geometry supports trained policies that achieve humanoid walking and quadruped stair climbing (Fig. 6(c)). These results demonstrate the utility of the reconstructed assets for physics-based locomotion simulation. Details are provided in Appendix B.3.

4.4 Ablation Studies

We conduct ablation studies to evaluate the contributions of our representation and framework design.

Representation Design. We replace triangles with surfels to ablate our triangle representation. Unlike surfels with soft Gaussian kernels, our sigmoid-based edge-preserving triangles provide explicit boundaries that confine gradients to structurally relevant pixels. This stabilizes optimization, leading to more accurate geometry and higher-quality rendering (Fig. 7). We further ablate the hybrid

Table 3: **Ablation studies on the ScanNetV2 dataset.** We conduct ablation studies on the hybrid representation and framework design, evaluating performance across both geometric and appearance metrics.

Setting	Geometry		Rendering			# Primitives
	Ch-L2↓	F-score↑	PSNR↑	SSIM↑	LPIPS↓	(#Geo/#GS)
Ours	**5.68**	**62.15**	**28.83**	**0.882**	0.222	56.1k/222.2k
w/o triangles	5.90	59.85	28.44	0.876	0.232	52.8k/157.3k
w/o hybrid	6.06	57.54	28.48	0.877	0.231	-/621.5k
w/o spatial filtering	6.01	58.86	28.66	0.880	**0.213**	211.5k/625.7k
w/o global map update	6.20	56.00	28.33	0.877	0.229	55.3k/166.6k

−: w/o geometric primitives.

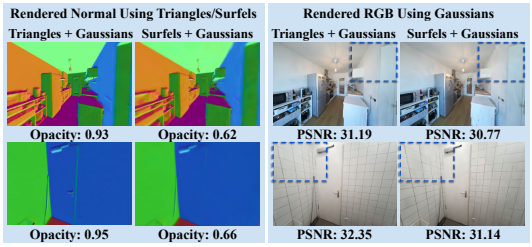

Figure 7: **Ablation on triangle representation.** Compared to surfels, our representation produces clearer opaque surfaces and finer rendering details.

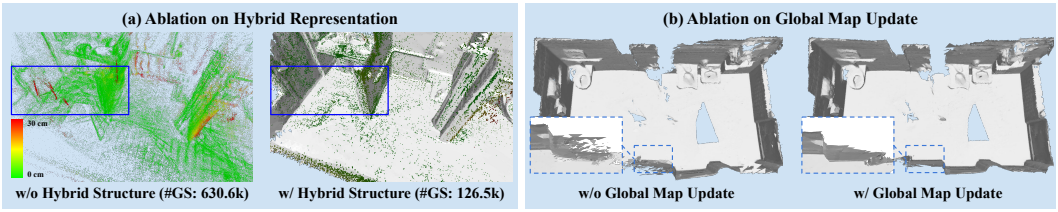

Figure 8: **Ablation on hybrid representation and global map update.** Our hybrid design reduces representation redundancy and mitigates geometric inconsistencies in feed-forward depth, while global map updates improve overall consistency. Point clouds visualize Gaussian centers.

representation by replacing it with unanchored neural Gaussians. As shown in Tab. 3, the hybrid representation improves both geometric accuracy and rendering quality. Moreover, the proposed representation reduces redundancy and encourages Gaussians to concentrate around the underlying surface, as shown in Fig. 8(a).

Framework Design. We conduct ablation studies on the mapping module of our on-the-fly reconstruction framework. Disabling **spatial filtering** substantially increases the number of primitives (+200% on ScanNetV2 and +245% on ScanNet++), confirming its effectiveness in reducing redundancy. As shown in Fig. 8(b), enabling the **global map update** improves geometric consistency and, consequently, rendering quality. More ablation results are provided in Appendix C.3.

5 Conclusion

PLANING addresses a key limitation of existing streaming Gaussian-based reconstruction methods: the lack of explicit, compact, and stable geometry for anchoring appearance optimization. By introducing a loosely coupled triangle-Gaussian representation together with a streaming reconstruction framework, PLANING decouples geometry from appearance while preserving high-fidelity rendering. This design mitigates geometric drift, redundancy, and optimization instability that arise when geometry and appearance are modeled by a single representation. Extensive results show that PLANING enables efficient and structurally consistent streaming reconstruction, while producing compact planar assets that support downstream applications such as pose refinement, large-scale reconstruction, and simulation-ready scene modeling.

Acknowledgements

This research was supported by the National Natural Science Foundation of China (Grant No. 62576311), in part by the Baima Lake Laboratory Joint Funds of the Zhejiang Provincial Natural Science Foundation of China under Grant No.LBMHD25F030001, and by the National Natural Science Foundation of China (Grant No. 62502247), the HKU Startup Fund, Institute of Data Science, and HKU-Shanghai Intelligent Computing Research Center. The authors declare no competing interests.

References

[1] Nathaniel Burgdorfer and Philippos Mordohai. Radiant triangle soup with soft connectivity forces for 3d reconstruction and novel view synthesis. *arXiv preprint arXiv:2505.23642*, 2025.

[2] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE transactions on robotics*, 37(6):1874–1890, 2021.

[3] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[4] Chong Cheng, Sicheng Yu, Zijian Wang, Yifan Zhou, and Hao Wang. Outdoor monocular slam with global scale-consistent 3d gaussian pointmaps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 26035–26044, 2025.

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[6] Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024.

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

[8] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024.

[9] Seongbo Ha, Jiung Yeon, and Hyeonwoo Yu. Rgbd gs-icp slam. In *European Conference on Computer Vision*, pages 180–197. Springer, 2024.

[10] Jan Held, Sanghyun Son, Renaud Vandeghen, Daniel Rebain, Matheus Gadelha, Yi Zhou, Anthony Cioppa, Ming C Lin, Marc Van Droogenbroeck, and Andrea Tagliasacchi. Meshsplatting: Differentiable rendering with opaque meshes. *arXiv preprint arXiv:2512.06818*, 2025.

[11] Jan Held, Renaud Vandeghen, Abdullah Hamdi, Adrien Deliege, Anthony Cioppa, Silvio Giancola, Andrea Vedaldi, Bernard Ghanem, and Marc Van Droogenbroeck. 3d convex splatting: Radiance field rendering with 3d smooth convexes. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21360–21369, 2025.

[12] SA Hojjatoleslami and Josef Kittler. Region growing: a new approach. *IEEE Transactions on Image processing*, 7(7):1079–1084, 1998.

[13] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024.

[14] Changjian Jiang, Kerui Ren, Linning Xu, Tao Lu, Jiong Chen, Jiangmiao Pang, Yu Zhang, Bo Dai, and Mulin Yu. Halogs: Loose coupling of compact geometry and gaussian splats for 3d scenes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 238:33–54, 2026.

[15] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21357–21366, 2024.

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[17] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, pages 71–91. Springer, 2024.

[18] Guanghao Li, Kerui Ren, Linning Xu, Zhewen Zheng, Changjian Jiang, Xin Gao, Bo Dai, Jian Pu, Mulin Yu, and Jiangmiao Pang. Artdeco: Towards efficient and high-fidelity on-the-fly 3d reconstruction with structured scene representation. *arXiv preprint arXiv:2510.08551*, 2025.

[19] Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

[20] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024.

[21] Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. *ACM Transactions on Graphics (TOG)*, 43(6):1–12, 2024.

[22] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18039–18048, 2024.

[23] Andreas Meuleman, Ishaan Shah, Alexandre Lanvin, Bernhard Kerbl, and George Drettakis. On-the-fly reconstruction for large-scale novel view synthesis from unposed images. *ACM Transactions on Graphics (TOG)*, 44(4):1–14, 2025.

[24] 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.

[25] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.

[26] Riku Murai, Eric Dexheimer, and Andrew J Davison. Mast3r-slam: Real-time dense slam with 3d reconstruction priors. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16695–16705, 2025.

[27] Zhenghao Qi, Shenghai Yuan, Fen Liu, Haozhi Cao, Tianchen Deng, Jianfei Yang, and Lihua Xie. Air-embodied: An efficient active 3dgs-based interaction and reconstruction framework with embodied large language model. *arXiv preprint arXiv:2409.16019*, 2024.

[28] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE transactions on robotics*, 34(4):1004–1020, 2018.

[29] Xuqian Ren, Matias Turkulainen, Jiepeng Wang, Otto Seiskari, Iaroslav Melekhov, Juho Kannala, and Esa Rahtu. Ags-mesh: Adaptive gaussian splatting and meshing with geometric priors for indoor room reconstruction using smartphones. In *International Conference on 3D Vision (3DV)*, 2025.

[30] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision*, pages 501–518. Springer, 2016.

[31] Christian Sigg, Tim Weyrich, Mario Botsch, and Markus H Gross. Gpu-based ray-casting of quadratic surfaces. In *PBG@ SIGGRAPH*, pages 59–65, 2006.

[32] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

[33] Bin Tan, Rui Yu, Yujun Shen, and Nan Xue. Planarsplatting: Accurate planar surface reconstruction in 3 minutes. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1190–1199, 2025.

[34] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025.

[35] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. *arXiv preprint arXiv:2507.13347*, 2025.

[36] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[37] Jamie Watson, Filippo Aleotti, Mohamed Sayed, Zawar Qureshi, Oisin Mac Aodha, Gabriel Brostow, Michael Firman, and Sara Vicente. Airplanes: Accurate plane estimation via 3d-consistent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5270–5280, 2024.

[38] Linning Xu, Vasu Agrawal, William Laney, Tony Garcia, Aayush Bansal, Changil Kim, Samuel Rota Bulò, Lorenzo Porzi, Peter Kontschieder, Aljaž Božič, et al. Vr-nerf: High-fidelity virtualized walkable spaces. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–12, 2023.

[39] Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16262–16272, 2024.

[40] Keyang Ye, Tianjia Shao, and Kun Zhou. When gaussian meets surfel: Ultra-fast high-fidelity radiance field rendering. *ACM Transactions on Graphics (TOG)*, 44(4):1–15, 2025.

[41] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023.

[42] 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.

[43] Mulin Yu, Tao Lu, Linning Xu, Lihan Jiang, Yuanbo Xiangli, and Bo Dai. Gsdf: 3dgs meets sdf for improved neural rendering and reconstruction. *Advances in Neural Information Processing Systems*, 37:129507–129530, 2024.

[44] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[45] Wei Zhang, Qing Cheng, David Skuddis, Niclas Zeller, Daniel Cremers, and Norbert Haala. Hi-slam2: Geometry-aware gaussian slam for fast monocular scene reconstruction. *IEEE Transactions on Robotics*, 41:6478–6493, 2025.

[46] Ziyu Zhang, Binbin Huang, Hanqing Jiang, Liyang Zhou, Xiaojun Xiang, and Shuhan Shen. Quadratic gaussian splatting: High quality surface reconstruction with second-order geometric primitives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 28260–28270, 2025.

[47] Chunran Zheng, Wei Xu, Zuhao Zou, Tong Hua, Chongjian Yuan, Dongjiao He, Bingyang Zhou, Zheng Liu, Jiarong Lin, Fangcheng Zhu, et al. Fast-livo2: Fast, direct lidar-inertial-visual odometry. *IEEE Transactions on Robotics*, 2024.

[48] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025.

The appendices provide additional implementation details, application analyses, supplementary experimental results, and limitations. Specifically, Sec. A details the technical implementation of our method and the baseline evaluation protocol, Sec. B describes application settings, Sec. C presents supplementary experimental results, and Sec. D discusses limitations.

A Technical Details

A.1 Differentiable Triangle Rasterizer

To enable unbiased depth and normal rendering with triangle primitives, we adopt an explicit ray–primitive intersection strategy [31], following 2DGS [13]. We define the transformation from a triangle's local coordinate system to world space as

$$\mathbf{H} = \begin{bmatrix} s_u \mathbf{t}_u & s_v \mathbf{t}_v & 0 & \boldsymbol{\mu} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{9}$$

where $\boldsymbol{\mu}$, s_u, $\mathbf{t}_u$, s_v, and $\mathbf{t}_v$ follow the definition of the local triangle frame in Eq. 1.

When combined with the edge-preserving contribution function (Eq. 2), this formulation leads to two practical challenges: (i) **inaccurate depth sorting** for large triangles whose barycenters deviate from true ray–triangle intersections; and (ii) **incorrect visibility estimation** when triangle barycenters are occluded while portions of the triangle remain visible.

To address these issues, we propose a subdivision-aware forward rendering pipeline that combines adaptive triangle subdivision for robust depth sorting with vertex-based visibility reasoning for accurate occlusion handling. The complete procedure is summarized in Algorithm 1, and its effectiveness is illustrated in Fig. 9(b).

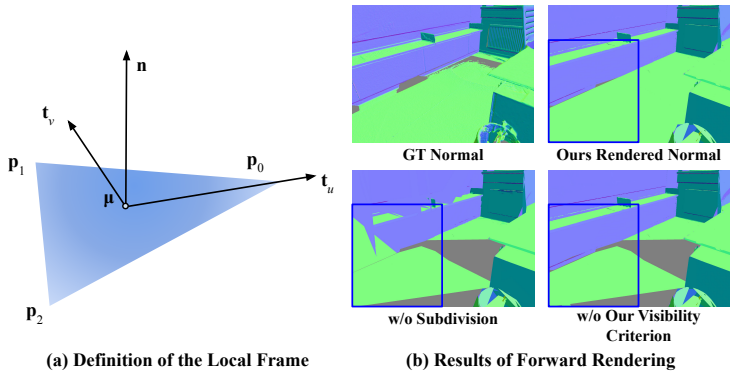

Figure 9: **Definition of the local frame** and **results of forward rendering**. Our triangle rasterizer enables correct and reliable forward rendering of triangles.

A.2 Primitive Initialization

Upon the arrival of a keyframe from the backend, the framework determines the optimal locations for instantiating new primitives. To maintain a compact global map and mitigate structural redundancy, triangle insertion is restricted to regions exhibiting insufficient geometric coverage or high reconstruction error, guided by image-level priors. Specifically, we first apply a photometric filter, which prioritizes high-frequency regions and poorly reconstructed areas by computing an insertion probability $P_a(u, v)$ at each pixel (u, v) using the Laplacian of Gaussian (LoG) operator $\Phi(\cdot)$ [23] to measure the discrepancy between the ground truth and rendered images as

$$P_a(u, v) = \max\left(\Phi(I) - \Phi(\tilde{I}), 0\right), \tag{10}$$

where $\Phi(I) = \min(\|\nabla^2(G_{\sigma_g} * I)(u, v)\|, 1)$, I and $\tilde{I}$ represent the ground-truth and rendered images, respectively, and G_{σ_g} denotes a Gaussian smoothing kernel. A new geometric primitive is considered only when $P_a(u, v)$ exceeds a stochastic insertion threshold. We denote the upper bound of the sampled threshold by τ_a.

ALGORITHM 1: Subdivision-aware Forward Rendering

Input: Triangle soup $\mathcal{T}$, camera pose $\mathbf{W}$, screen resolution
Output: Rendered depth and normal maps
Triangle Preprocessing:
Initialize visible triangle set $\mathcal{T}_v \leftarrow \emptyset$
foreach *triangle* $t \in \mathcal{T}$ **do**
  if *at least one vertex of* t *is visible* **then**
    Construct local triangle frame and transformation $\mathbf{H}$;
    Subdivide t recursively until all edges are shorter than threshold ϵ ;
    Assign parent triangle ID to all subdivision triangles ;
    Add subdivision triangles to $\mathcal{T}_v$;
  end
end
Subdivision Processing:
foreach *subdivision triangle* $t_s \in \mathcal{T}_v$ **do**
  if *at least one vertex of* t_s *is visible* **then**
    Project vertices to image plane ;
    Determine overlapped tiles ;
    Compute view-space depth using barycenter of t_s ;
    Generate sorting key (depth, tile ID) ;
  end
end
Depth Sorting:
Perform GPU-based radix sort on all subdivision triangles using sorting keys ;
Rendering:
foreach *pixel* $\mathbf{x} = (x, y)^T$ **do**
  Define the camera ray using two orthogonal homogeneous planes;
  Transform rays into local triangle coordinates using $(\mathbf{WH})^T$;
  Compute ray–triangle intersection $\hat{\mathbf{x}}$ on the original triangle ;
  Evaluate rendering contribution using Eq. 2 ;
end
Render depth and normal images following Eq. 4 in the main text ;

Spatial Filtering. To suppress structural redundancy, we apply a depth-adaptive spatial filter to candidate primitives passing the photometric filter. For each candidate pixel, we compute its back-projected 3D center $\mathbf{c}_i$ and discard it if nearby geometric primitives already exist within a local vicinity, where the vicinity radius is defined in Eq. 5. This strategy effectively prevents excessive primitive growth in already reconstructed regions.

Triangle Initialization. Once a candidate pixel (u, v) is selected, a triangle is initialized. Each triangle is parameterized by its vertices $\{\mathbf{p}_{\mathrm{t},k}\}_{k=0}^{2}$, opacity α_{t}, sharpness δ_{t}, smoothness σ_{t}, and a feature vector $\mathbf{f}_{\mathrm{t}}$. Following geometric scaling principles, the world-space scale is $s_{\mathrm{t}} = 3d_i/2f\sqrt{\Phi(I)}$, where f is the focal length. The triangle orientation is determined by the normal prior at (u, v). Specifically, three unit vectors $\{\mathbf{v}_{\mathrm{t},k}\}_{k=0}^{2}$ are sampled on the local tangent plane at the back-projected center $\mathbf{c}_i$, and the vertex positions are initialized as $\mathbf{p}_{\mathrm{t},k} = \mathbf{c}_i + s_{\mathrm{t}}\mathbf{v}_{\mathrm{t},k}$ for $k \in \{0, 1, 2\}$. The opacity is initialized as $\alpha_{\mathrm{t}} = 0.2\,C(u, v)$ to down-weight low-confidence regions, where $C(u, v)$ is the backend confidence score.

Gaussian Initialization. Then, neural Gaussians are initialized at triangle barycenters for appearance modeling. We adaptively set the number of Gaussians per triangle to $K_{\max}$ if $\Phi(I) > 0.4$, and $K_{\min}$ otherwise. Here, the hyperparameters $K_{\max}$ and $K_{\min}$ define the bounds of the representational capacity based on scene detail. For primitive attributes, we initialize offsets $\mathbf{o}_{\mathrm{g}}$, rotation $\mathbf{q}_{\mathrm{g}}$, and features $\mathbf{f}_{\mathrm{g}}$ to zero, while Gaussian opacity α_{g} is synchronized with α_{t}. The base scale is defined as $\mathbf{s}_{\mathrm{g}} = d_i/2f\sqrt{\Phi(I)} \cdot \mathbf{1}$ to align with local geometry. Crucially, the zero-order spherical harmonic coefficient $\mathbf{SH}_0$ is extracted from the pixel color at (u, v), with higher coefficients zero-initialized.

Table 4: **Effect of the shared MASt3R geometric-prior integration on ScanNetV2.** The three per-scene baselines use the same PLANING-estimated poses and precomputed MASt3R priors.

Method	MASt3R Prior	Ch-L2 ↓	F-score ↑
2DGS	✗	8.65	39.60
	✓	6.48	53.73
PGSR	✗	8.81	40.79
	✓	6.59	54.28
MeshSplatting	✗	14.63	23.12
	✓	11.15	30.73
PLANING	✓	**5.68**	**62.15**

A.3 Training Strategy

In our streaming reconstruction system, we adopt a staged training strategy to balance efficiency and reconstruction quality, following [18]. Specifically, when a keyframe is encountered, new primitives are initialized and the scene is optimized for M iterations (set to 20 in our implementation), while common frames are optimized for only $M/2$ iterations without adding new Gaussians. Training frames are sampled with a probability of 0.2 from the current frame and 0.8 from past frames to mitigate local overfitting. After processing the sequence in a streaming fashion, a global optimization is performed over all frames, prioritizing those with fewer prior updates.

A.4 Streaming Reconstruction with Global Refinement

PLANING performs incremental reconstruction as frames arrive. During this stage, the frontend estimates camera poses and selects keyframes, while the mapper continuously updates the hybrid map with newly observed scene content. The resulting map is therefore available throughout the streaming process.

After all input frames have been processed, we perform a final global refinement to improve global consistency and rendering quality. This stage refines the map constructed during the incremental reconstruction stage; it does not reconstruct the scene offline from scratch. Specifically, the refinement jointly optimizes the existing map using observations from the complete sequence, with priority given to frames that have received fewer previous updates.

Similar post-stream refinement stages are also adopted by existing online reconstruction frameworks, including OnTheFly-NVS [23], S3PO-GS [4], ARTDECO [18], SplaTAM [15], and HI-SLAM2 [45]. The reported runtime includes this final global refinement stage.

A.5 Planar Primitive Extraction

Planar primitives provide an efficient structural abstraction of the scene and can be directly leveraged in downstream tasks, such as robot local motion training. To extract these planes, we adopt a coarse-to-fine strategy based on GoCoPP [42], where the method is applied iteratively with progressively finer parameters to detect smaller planes from the residual points remaining after coarser planes are extracted.

A.6 More Implementation Details

For our method, we set $K_{\min} = 4$ and $K_{\max} = 8$, with loss weights $\lambda_{\mathrm{d}} = 10.0$, $\lambda_{\mathrm{n}} = 3.0$, $\lambda_{\mathrm{o}} = 0.2$, $\lambda_{\mathrm{c}} = 0.2$, and $\lambda_{\mathrm{s}} = 0.01$. For primitive insertion, we set the upper bound of the stochastic threshold to $\tau_a = 0.5$. For dense mesh extraction, our method fuses triangle-rendered depth maps into meshes using TSDF, following the procedure in 2DGS [13]. The pose protocol and geometric-prior integration are detailed in Appendices A.7 and A.8, respectively. For planar primitive extraction, since baseline methods, including 2DGS [13], PGSR [3], MeshSplatting [10], and ARTDECO [18], typically output dense meshes, we extract multi-level planar shapes from their results using the same strategy and parameters applied to our method to ensure a fair comparison. All experiments are performed on an Intel Core i9-14900K CPU and an NVIDIA RTX 4090 GPU.

A.7 Pose Protocol

To ensure a controlled and fair comparison, methods that do not estimate camera poses within their reconstruction pipelines, including 2DGS [13], PGSR [3], MeshSplatting [10], AirPlanes [37], and PlanarSplatting [33], are provided with the same camera poses estimated by PLANING. This protocol controls the pose input and isolates differences in reconstruction quality. Methods with native tracking are evaluated using their own estimated poses, such that their results reflect end-to-end performance under their respective tracking and mapping pipelines.

A.8 Geometric-Prior Integration

For the three per-scene baselines in Tab. 4, 2DGS [13], PGSR [3], and MeshSplatting [10], we augment each method's original objective $\mathcal{L}_{\mathrm{org}}$ with the same MASt3R [17] geometric supervision:

$$\mathcal{L}_{\mathrm{sum}} = \mathcal{L}_{\mathrm{org}} + \mathcal{L}_{\mathrm{geo}}, \quad \mathcal{L}_{\mathrm{geo}} = \lambda_{\mathrm{d}}^{\mathrm{base}} \|\mathbf{D} - \mathbf{D}_{\mathrm{p}}\|_1 + \lambda_{\mathrm{n}}^{\mathrm{base}} \|\mathbf{N} - \mathbf{N}_{\mathrm{p}}\|_1, \tag{11}$$

where $\mathbf{D}_{\mathrm{p}}$ and $\mathbf{N}_{\mathrm{p}}$ denote the MASt3R depth and normal priors, respectively. We use $\lambda_{\mathrm{d}}^{\mathrm{base}} = 0.2$ and $\lambda_{\mathrm{n}}^{\mathrm{base}} = 0.1$ for these three baselines, following the geometric-prior formulation of AGS-Mesh [29], without baseline-specific tuning. All original losses, optimization schedules, and hyperparameters are otherwise retained.

The MASt3R priors are precomputed once from the same input images and reused unchanged across these three baselines. We retain pixels with prior depth in the range $(0, 100)$ m and apply the same validity mask to both depth and normal supervision. Geometric supervision is applied throughout optimization, and all rendered depths and normals are expressed in the same camera-space convention before computing the losses.

The method-specific component is limited to obtaining the rendered depth $\mathbf{D}$ and normal $\mathbf{N}$ from each native representation. For 2DGS [13], $\mathbf{D}$ is rendered from its 2D Gaussian surfels and $\mathbf{N}$ is computed from the rendered depth following Eq. 15 of [13] and the released implementation. For PGSR [3], $\mathbf{D}$ is its unbiased Gaussian-rendered depth and $\mathbf{N}$ is derived from depth following Eq. 5 of [3]. For MeshSplatting [10], $\mathbf{D}$ is rasterized from its triangle representation and $\mathbf{N}$ is computed from depth following its public implementation.

Tab. 4 reports the effect of shared MASt3R supervision on all ScanNetV2 scenes. The same prior integration consistently improves all three baselines, reducing Chamfer-L2 by 23.8–25.2%. These consistent gains confirm that all three baselines benefit substantially from the same augmentation, supporting the fairness of the comparison.

B Application Details

B.1 Plane-Guided Camera Pose Optimization

In our streaming reconstruction system, we optionally feed back the reconstructed planar map to the frontend to refine camera poses via a point-to-plane alignment loss, improving global consistency. Specifically, in the mapper, we maintain a voxel map using a spatial hash to manage triangle primitives. During training, planar primitives are regularly extracted via region growing [12]. In our implementation, the voxel size is set to 3 cm, and plane extraction is performed every 10 frames. The extracted plane parameters and associated voxel keys are then shared with the frontend.

In the frontend, high-confidence points predicted by MASt3R [17] are associated with the planar map via the voxel grid. For each point $\mathbf{p}$ and its corresponding plane, we adopt a simple yet effective point-to-plane alignment loss:

$$\mathcal{L}_p = \|(\mathbf{p} - \mathbf{c}) \cdot \mathbf{n}\|_1, \tag{12}$$

where $\mathbf{n}$ and $\mathbf{c}$ denote the plane's normal and center, respectively.

B.2 Large Scale Scene Reconstruction

To enable large-scale scene reconstruction and alleviate GPU memory limitations, we introduce a dynamic loading strategy that swaps primitive parameters between the GPU and CPU. Specifically, we periodically evaluate the projected scale of each neural Gaussian on the most recent image plane.

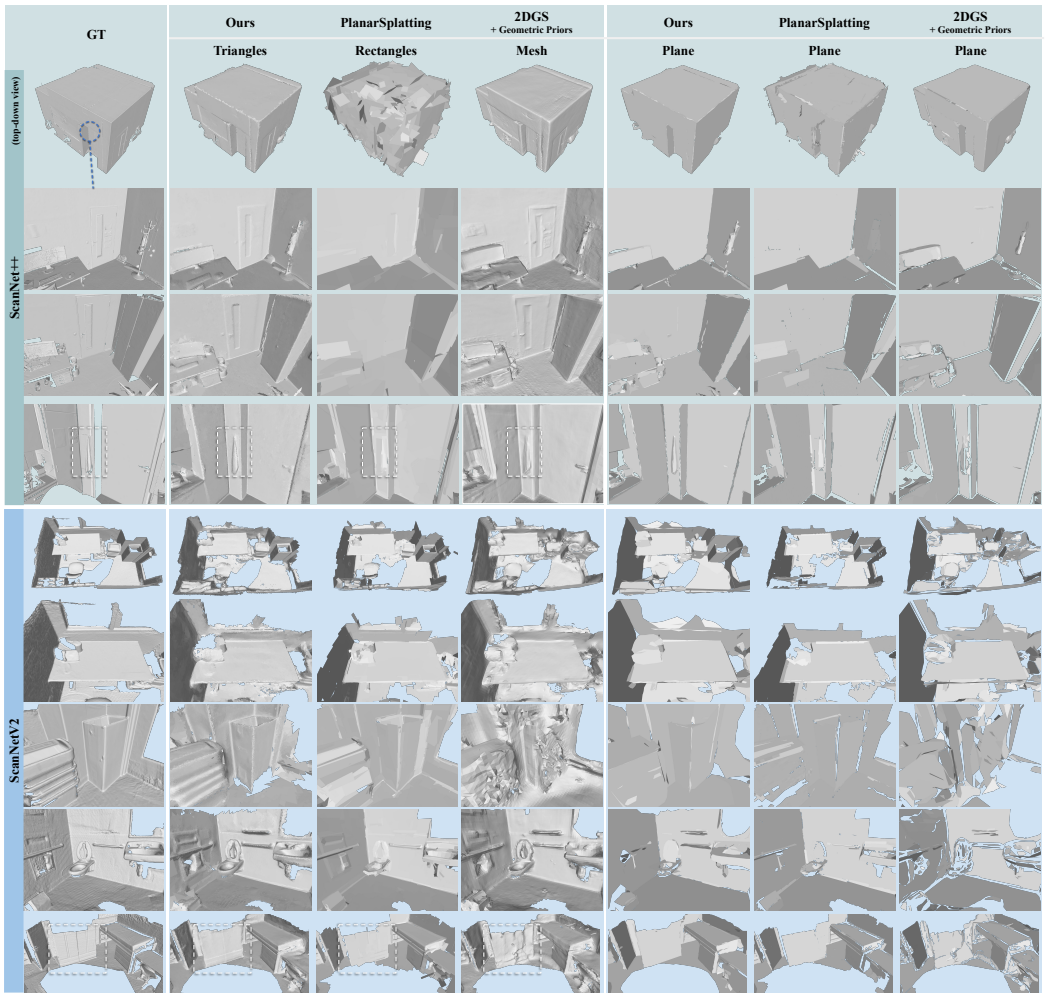

Figure 10: **More geometric comparison results.** We visualize planar reconstruction and geometric modeling across different primitives, with 2DGS shown as dense mesh for comparison. Overall, our method preserves planar structures while capturing fine geometric details.

Table 5: **Comparison of scene import and conversion time under non-headless (GUI-based) and headless Isaac Sim pipelines.** Both settings perform the same sequence of operations, including mesh import, collision geometry construction, and USD packaging, but differ in execution mode.

Setting	Non-headless (s)	Headless (s)	# Faces
Ours (Plane)	**89.73**	**5.27**	17k
2DGS	–	657	277k
2DGS*	120.00	37.21	17k

–: impractical runtime ($>$ 30 min), *: 16$\times$ mesh simplification.

A neural Gaussian is marked as *invisible* if its projected scale is smaller than a pixel, and a triangle is marked as *invisible* when all its associated neural Gaussians are invisible. Invisible triangles and their corresponding Gaussians are then offloaded from the GPU to the CPU. Upon detecting a loop closure in the *Backend* module, we reload the primitives initialized from the associated images from the CPU back to the GPU. This dynamic loading strategy allows our framework to efficiently scale to large environments.

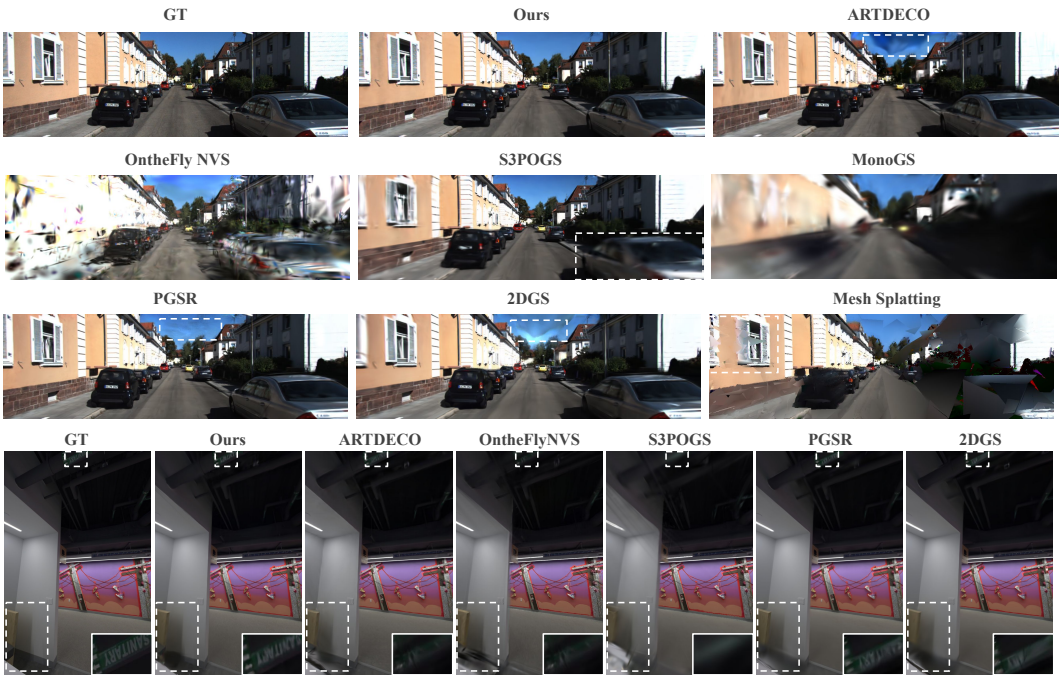

Figure 11: **More rendering comparison results.** White boxes highlight artifacts and fine-grained details from baseline methods. Our approach yields significantly sharper results on intricate structures, such as text, while achieving superior overall rendering quality.

B.3 Locomotion Policy Training

Beyond visual fidelity, our hybrid representation produces compact planar geometry for downstream physics simulation. By explicitly modeling large-scale structures, such as floors and walls, with triangles, PLANING provides structurally consistent scene assets for constructing the evaluated Isaac Lab locomotion environments.

Although 2DGS [13] is a strong reconstruction baseline, its dense meshes incur substantial preprocessing overhead in Isaac Sim. In our evaluated scene, the 2DGS mesh contains 277k faces and requires over 30 minutes for standard import and conversion. In contrast, the lower mesh complexity of PLANING reduces processing time in both the standard and headless (`convert_mesh`) pipelines. Quantitative comparisons are reported in Tab. 5.

We evaluate the downstream utility of the reconstructed geometry through locomotion experiments in Isaac Lab using the Unitree H1 humanoid and Unitree A1 quadruped. To require policies to rely on the reconstructed geometry, we omit the height scanner from the observations. Under this setting, policies trained with aggressively simplified 2DGS meshes fail to converge because of degraded planar geometry, whereas policies trained in our reconstructed scenes achieve stable locomotion under identical observation settings. Overall, the reconstructed assets provide compact geometry for the evaluated Isaac Lab environments, demonstrating their utility for downstream locomotion simulation.

C Supplementary Experiments

C.1 Supplementary Comparison Experiments

In Fig. 10, we provide additional reconstruction comparisons on the ScanNet++ [41] and ScanNetV2 [5] datasets, showing that our method more faithfully preserves the scene's geometric structures. Fig. 11 presents rendering quality comparisons on the KITTI [7] and VR-NeRF [38] datasets, demonstrating our method's robustness and applicability across diverse scenarios, including both indoor and outdoor environments.

Table 6: **Quantitative comparison of dense mesh reconstruction.**

Method	ScanNet++		ScanNetV2		FAST-LIVO2	
	Ch-L2↓	F-score↑	Ch-L2↓	F-score↑	Ch-L2↓	F-score↑
2DGS†	3.95	80.90	6.45	53.11	52.83	61.06
PGSR†	3.92	81.47	6.55	53.89	53.56	60.99
MeshSplatting†	9.24	46.30	11.05	31.22	61.03	51.61
ARTDECO	3.87	82.34	6.00	57.61	36.99	61.41
Ours	3.76	84.81	5.87	59.93	38.44	64.36

†: offline per-scene baselines are augmented with the same geometric-prior supervision as PLANING.

Table 7: **Ablation studies on the ScanNet++ dataset.** Our ablations are divided into two categories: representation design and framework design.

Setting	Geometry		Rendering			# Primitives
	Ch-L2↓	F-score↑	PSNR↑	SSIM↑	LPIPS↓	(#Geo/#GS)
Ours	**3.53**	**86.88**	31.91	0.941	0.133	61.6k/291.0k
w/o triangles	3.63	84.91	31.05	0.932	0.150	39.4k/201.9k
w/o hybrid	3.81	83.01	31.60	0.940	0.140	-/478.3k
w/o spatial filtering	3.71	84.94	**31.95**	**0.942**	**0.126**	233.6k/981.6k
w/o global map update	3.59	85.07	31.78	0.941	0.131	61.5k/291.4k

−: w/o geometric primitives.

Table 8: ATE RMSE (m) on the ScanNetV2 and ScanNet++ scenes used in our reconstruction experiments. Lower is better.

Dataset	MonoGS	S3PO-GS	OnTheFly-NVS	ARTDECO	PLANING
ScanNetV2	0.246	0.143	0.289	0.059	**0.056**
ScanNet++	0.212	0.088	0.188	**0.013**	0.016

C.2 Dense Mesh Reconstruction

We additionally evaluate the geometric quality of dense mesh reconstruction. To ensure a fair comparison, all meshes are extracted using a unified depth-fusion pipeline. Across all evaluated dense-mesh scenes, PLANING reduces mean per-scene Chamfer-L2 by 18.52% relative to PGSR, computed from unrounded results. As reported in Tab. 6, PLANING achieves the lowest Chamfer-L2 on ScanNet++ and ScanNetV2, and the highest F-score across all three datasets.

C.3 Supplementary Ablation Studies

We also conduct ablation studies on ScanNet++, using the same experimental setup as in the main text. As shown in Tab. 7, our representation design achieves the best performance in both rendering quality and geometric accuracy. Regarding system design, by enabling spatial filtering, our method can achieve higher geometric accuracy and comparable rendering quality with less than one-third of the primitives, highlighting the efficiency of our framework. Furthermore, we investigate the effect of the number of Gaussians on rendering quality. As shown in Fig. 12, PSNR and SSIM initially improve with increasing Gaussian count and then saturate.

C.4 Pose Accuracy

We assess camera-pose accuracy using absolute trajectory error (ATE) RMSE, measured in meters, on ScanNetV2 and ScanNet++. Tab. 8 compares PLANING with representative streaming reconstruction baselines. PLANING achieves 0.056 m ATE on ScanNetV2 and 0.016 m on ScanNet++. These results show that the PLANING-estimated poses are sufficiently accurate to serve as a common pose input in the controlled evaluation.

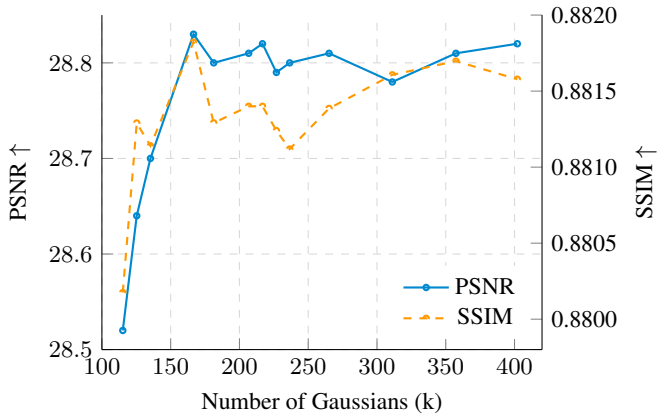

Figure 12: **Effect of the number of Gaussians on rendering quality.** PSNR and SSIM improve initially and then saturate as Gaussian count increases.

Table 9: Controlled comparison with HaloGS and GSDF on ScanNetV2 and ScanNet++. Runtime, reported in minutes, and primitive count are averaged over the two datasets.

Method	Dataset	Geometry Ch-L2 ↓	Planar Ch-L2 ↓	PSNR ↑	Time	# Prim.
HaloGS	ScanNetV2	6.25	11.49	28.21	31.5	137.6k
	ScanNet++	3.92	7.76	31.95		
GSDF†	ScanNetV2	7.32	12.39	28.07	85.0	369.7k
	ScanNet++	5.12	11.33	**32.04**		
PLANING	ScanNetV2	**5.68**	**9.07**	**28.83**	**3.8**	**58.9k**
	ScanNet++	**3.53**	**7.09**	31.91		

†: GSDF is augmented with the shared MASt3R depth and normal supervision described in Appendix A.8; all other losses, training stages, and optimization settings are retained.

C.5 Comparison with HaloGS and GSDF

We compare PLANING with two additional offline reconstruction methods, HaloGS [14] and GSDF [43], on the ScanNetV2 and ScanNet++ datasets. The comparison evaluates geometric reconstruction, planar reconstruction, rendering quality, runtime, and representation compactness across both datasets.

For this controlled comparison, HaloGS and GSDF are provided with the same camera poses estimated by PLANING. Runtime and primitive counts are averaged over the two datasets. We retain the original training stages and full optimization budgets of both baselines. For HaloGS, which already includes a complete geometric-prior supervision scheme, we retain its original losses, weights, and optimization strategy, replacing only its prior inputs with the same MASt3R depth and normal maps used by PLANING. For GSDF, we apply the geometric-prior integration protocol described in Appendix A.8 to its geometry-producing SDF branch, using ℓ_1 depth and normal supervision with weights $\lambda_d^{\text{base}} = 0.2$ and $\lambda_n^{\text{base}} = 0.1$, respectively. All other loss terms, training stages, and optimization settings are retained.

As shown in Tab. 9, PLANING reduces the ScanNetV2 geometric Chamfer-L2 from 6.25 cm for HaloGS to 5.68 cm, while requiring 58.9k rather than 137.6k primitives. It is also 8.3× faster than HaloGS and 22.4× faster than GSDF, while maintaining comparable rendering quality. These results demonstrate a favorable trade-off between geometric accuracy, representation compactness, and reconstruction efficiency.

Compared with HaloGS, PLANING performs 3D reconstruction in a streaming manner and maintains a persistent hybrid map as new observations arrive. Whereas HaloGS relies on frequent triangle splitting for structural adaptation, PLANING uses selective primitive initialization and spatial filtering to control map growth before redundant primitives accumulate, substantially improving reconstruction efficiency and representation compactness. Moreover, global map updates propagate

backend pose corrections to previously constructed triangles and anchored Gaussians, maintaining map consistency throughout streaming reconstruction.

Compared with GSDF, PLANING uses explicit triangle primitives instead of an implicit SDF geometry branch. This provides a direct surface representation that supports efficient incremental reconstruction while retaining high-fidelity rendering through the loosely coupled Gaussian appearance model. Together, these comparisons demonstrate the practical benefit of PLANING's streaming-oriented representation and map design.

D Limitations

PLANING is a modular framework whose components can benefit from future advances in scene representation and rendering. Our current formulation inherits limitations from the chosen primitives and static-scene assumption. In particular, neural Gaussian primitives are not well suited for semi-transparent or transparent objects, where unreliable appearance gradients may adversely affect geometry optimization. Moreover, PLANING does not explicitly model dynamic objects; inconsistent observations from moving regions may lead to erroneous primitive initialization and unstable geometry optimization. Addressing these limitations is left as future work.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We described the method and its quality.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed the limitation in the appendices.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: This work does not present theoretical results requiring formal proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We described clearly in the method and exp section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will release the code after acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We described clearly in the Exp section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Error bars are not reported because our evaluation follows standard 3D reconstruction practice by reporting averaged results over multiple benchmark scenes, while repeated runs are computationally expensive.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide them in the implementation section.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The proposed method is about 3D reconstruction.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Not applicable, as the released artifacts are intended for 3D reconstruction research and do not include high-risk models or sensitive datasets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We properly cited them.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: Not applicable, as this paper does not introduce new datasets, benchmarks, or other standalone assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve any crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve any crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Not applicable, as LLMs were used only for language polishing, editing, and formatting, and did not contribute to the core methodology, experiments, analysis, or scientific claims of this work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.