MG-SLAM: Structure Gaussian SLAM with Manhattan World Hypothesis

Shuhong Liu
The University of Tokyo

Heng Zhou
Columbia University

Liuzhuozheng Li
The University of Tokyo

Yun Liu
National Institute of Informatics

Tianchen Deng
Shanghai Jiao Tong University

Yiming Zhou
Saarland University of Applied Sciences

Mingrui Li [*]
Dalian University of Technology

Abstract

Gaussian SLAM systems have made significant advancements in improving the efficiency and fidelity of real-time reconstructions. However, these systems often encounter incomplete reconstructions in complex indoor environments, characterized by substantial holes due to unobserved geometry caused by obstacles or limited view angles. To address this challenge, we present Manhattan Gaussian SLAM (MG-SLAM), an RGB-D system that leverages the Manhattan World hypothesis to enhance geometric accuracy and completeness. By seamlessly integrating fused line segments derived from structured scenes, MG-SLAM ensures robust tracking in textureless indoor areas. Moreover, The extracted lines and planar surface assumption allow strategic interpolation of new Gaussians in regions of missing geometry, enabling efficient scene completion. Extensive experiments conducted on both synthetic and real-world scenes demonstrate that these advancements enable our method to achieve state-of-the-art performance, marking a substantial improvement in the capabilities of Gaussian SLAM systems.

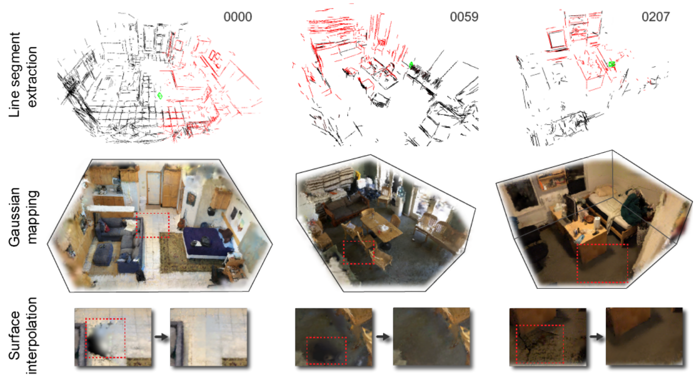

Figure 1: MG-SLAM leverages line segments to achieve SOTA results in camera pose estimation and scene reconstruction. Additionally, by applying structural surface constraints, we enhance and complete the scene through the interpolation of new Gaussians for absent geometry.

[*]Corresponding Author: 2905450254@mail.dlut.edu.cn

Preprint. Under review.

1 Introduction

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in computer vision that aims to map an environment while simultaneously tracking the camera pose. Learning-based dense SLAM methods, particularly neural radiance field (NeRF) approaches [51, 22, 41, 18, 52, 36, 8], have demonstrated remarkable improvements in capturing dense photometric information and providing accurate global reconstruction over traditional systems based on sparse point clouds or voxels [33, 35, 32, 14, 9, 17]. However, NeRF methods still face drawbacks such as over-smoothing, bounded scene representation, and computational inefficiencies. Recently, Gaussian-based SLAM [19, 28, 46, 26, 10, 25] has emerged as a promising approach utilizing volumetric radiance fields. Leveraging explicit 3D Gaussian representation [20], Gaussian SLAMs deliver high-fidelity rendering and fine-grained scene reconstruction, overcoming the limitations of NeRF-based methods.

Despite their strengths, Gaussian SLAM faces notable challenges in indoor scenes, which are often characterized by textureless surfaces and complex spatial layouts. These environments hinder robust tracking due to a lack of sufficient texture details critical for camera pose optimization. Moreover, the complex geometry of indoor scenes often leads to substantial unobserved areas due to occlusions or limited view coverage. These unseen regions pose a critical yet largely unexplored challenge for Gaussian SLAM, as the Gaussian representation can hardly interpolate the unobserved geometry without multi-view optimization. Consequently, substantial holes and gaps are left in the unseen areas of the map, an issue that has been largely overlooked in previous Gaussian SLAM studies.

To overcome these challenges, we leverage the renowned Manhattan World hypothesis [5] as a foundational strategy for refining and completing scene geometries. This assumption posits that the built environment predominantly adheres to a grid-like structure, with surfaces and lines aligning with three orthogonal directions. These lines and planar surfaces impose meaningful constraints on the tracking and mapping processes in the Gaussian SLAM system.

Specifically, we encompass enhancements in tracking, mapping, and scene completion. In tracking, we utilize line features derived from the structured scenes as robust feature foundations in textureless areas, backprojecting and reprojecting these line segments for pose optimization and full bundle adjustment. In mapping, we apply a photometric loss for the reprojected line features to refine the map. This approach ensures that the reconstructed scene adheres closely to the true structure of the environment, thereby improving both its geometry accuracy and rendering quality.

Furthermore, the Manhattan World hypothesis [5] facilitates the identification and interpolation of structured surfaces, such as floors and ceilings. These planar surfaces are critical to defining the overall geometry of space but are often partially obscured or missing from the captured views. By segmenting these incomplete surfaces—refined by the extracted lines as boundaries—we can predict their continuation beyond the directly observed portions by generating new Gaussians. This strategy enables us to optimize the representation of large surfaces within the scene, enhancing the completeness of the rendered map. Finally, we compress the Gaussian representation into mesh surfaces by incorporating regularization terms through Poisson reconstruction [15]. This approach enables the extraction of high-quality mesh, previously unavailable in the Gaussian SLAM systems, making it readily available for downstream tasks.

Overall, our work presents several key contributions, summarized as follows:

- We propose MG-SLAM, a novel RGB-D Gaussian SLAM system that capitalizes on the Manhattan World hypothesis [5]. This assumption introduces lines and planar surfaces for robust tracking, map refinement, and surface completion for neural-dense SLAM systems.
- We establish hypothesis surfaces using extracted line segments that represent planar boundaries. These surfaces guide our efficient interpolation of new Gaussians to fill gaps and holes in the reconstructed map, seamlessly addressing areas where current Gaussian SLAM systems face limitations due to unobserved geometry.
- Extensive experiments conducted on both large-scale synthetic and real-world datasets demonstrate that our system offers state-of-the-art (SOTA) tracking and comprehensive map reconstruction, achieving 50% lower ATE and 5dB enhancement in PSNR on real-world scenes, meanwhile operating at an exceptionally high frame rate. These advancements remarkably outperform previous Gaussian SLAM systems.

2 Related Work

2.1 Neural dense SLAM

Neural Implicit SLAM systems leveraging NeRF [30] are adept at handling complex topological structures by employing differentiable scene representation approaches [51, 4, 24, 18, 36, 11, 49, 12]. Despite these advancements, NeRF methods often struggle with the over-smoothing issue, where fine-grained object-level geometry and features are difficult to capture during reconstruction [26]. Moreover, these methods suffer from catastrophic loss as the scenes are implicitly represented by MLPs.

In contrast, the high-fidelity and fast rasterization capabilities of 3D Gaussian Splatting [20] enable higher quality and efficiency on scene reconstruction [44, 26, 39]. MonoGS [28] utilizes a map-centric SLAM approach that employs 3D Gaussian representation for dynamic and high-fidelity scene capture. SplaTAM [19] adopts an explicit volumetric approach using isotropic Gaussians, enabling precise camera tracking and map densification. However, existing Gaussian SLAM systems lack effective camera pose optimization, limiting tracking accuracy in complex scenes. Moreover, they struggle to effectively map unobserved areas, often resulting in incomplete reconstructions with gaps and holes. This limitation becomes more problematic in settings where the camera's movement is restricted, leading to significant unmodeled areas in structured indoor scenes. Additionally, current Gaussian-based SLAM methods are unable to generate meshes due to the discrete nature of 3D Gaussians which poses difficulties in surface extraction.

2.2 Reconstruction with structure optimization

Line features are known to remarkably enhance camera pose optimization [50, 43]. Traditional SLAM systems [42, 14, 48, 3] utilize the point feature and line segment for capturing high-level geometric elements and structural properties, which accurately refine the reconstructed map. However, these systems rely on sparse map expressions and struggle to interpret dense surfaces. The integration of line features in neural dense SLAM systems still remains an unexplored topic.

As for surface interpolation, several offline reconstruction approaches have been proposed. [16] introduced a NeRF-based method that optimizes reconstructed scenes by aligning the signed distance field (SDF) of planar surfaces. [40] and [27] proposed methods for geometric-aware Gaussian initialization from point clouds, which effectively preserve structural correlations, especially in textureless areas. However, these approaches are fundamentally based on optimization from observed views, which limits their ability to interpolate unseen structured geometry in reconstructed scenes.

3 Method

Figure 2 illustrates the pipeline of our proposed method. Under the constraints of the Manhattan World hypothesis, MG-SLAM introduces line segments and structured surfaces to enhance camera pose estimation and map reconstruction. Section 3.2 details the tracking mechanism that utilizes both point and line features. We utilize a specific strategy for fusing line segments to ensure reliable identification of line features. Section 3.3 discusses the Gaussian representation, including a specialized loss term dedicated to the reconstruction of line segments. Section 3.4 and appendix B.2 describe the completion and refinement of the scene, grounded in the assumption of structured surfaces. Appendix B.3 describes the mesh generation utilizing regularization losses.

3.1 Notation

We define the 2D image domain as $\mathcal{P} \in \mathbb{R}^2$, which encompasses appearance color information $\mathcal{C} \in \mathbb{N}^3$, semantic color $\mathcal{S} \in \mathbb{N}^3$, and depth data $\mathcal{D} \in \mathbb{R}^+$. Transitioning to the 3D world domain, denoted as $\mathcal{X} \in \mathbb{R}^3$, we introduce a camera projection function $\pi : \mathcal{X} \rightarrow \mathcal{P}$, mapping a 3D point $\mathcal{X}_i$ to its 2D counterpart $\mathcal{P}_i$, and conversely, a backprojection function $\theta : \mathcal{P} \rightarrow \mathcal{X}$, for the reverse mapping. $(R, t) \in SE(3)$ defines the camera pose. $\mathbb{I}$ is the identity matrix.

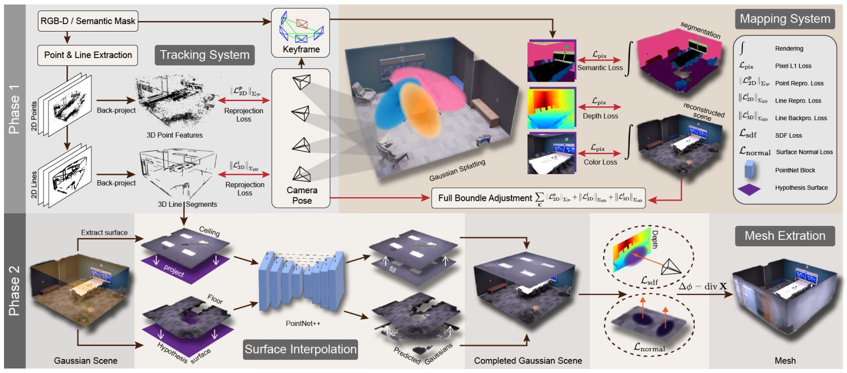
Phase 1
RGB-D / Semantic Mask
Point & Line Extraction
Tracking System
Keyframe
2D Points
Back-project
3D Point Features
$\|\mathcal{L}_{2D}^{p}\|_{\Sigma_{\mathcal{P}}}$
Reprojection Loss
2D Lines
Back-project
3D Line Segments
$\|\mathcal{L}_{2D}^{l}\|_{\Sigma_{2D}}$
Reprojection Loss
Camera Pose
Gaussian Splatting
$\mathcal{L}_{\text{pix}}$
Semantic Loss
$\mathcal{L}_{\text{pix}}$
Depth Loss
$\mathcal{L}_{\text{pix}}$
Color Loss
segmentation
reconstructed scene
Full Boundle Adjustment $\sum_{\mathcal{K}} \|\mathcal{L}_{2D}^{p}\|_{\Sigma_{\mathcal{P}}} + \|\mathcal{L}_{2D}^{l}\|_{\Sigma_{2D}} + \|\mathcal{L}_{3D}^{l}\|_{\Sigma_{3D}}$
Mapping System
$\int$ Rendering
$\mathcal{L}_{\text{pix}}$ Pixel L1 Loss
$\|\mathcal{L}_{2D}^{p}\|_{\Sigma_{\mathcal{P}}}$ Point Repro. Loss
$\|\mathcal{L}_{2D}^{l}\|_{\Sigma_{2D}}$ Line Repro. Loss
$\|\mathcal{L}_{3D}^{l}\|_{\Sigma_{3D}}$ Line Backpro. Loss
$\mathcal{L}_{\text{sdf}}$ SDF Loss
$\mathcal{L}_{\text{normal}}$ Surface Normal Loss
PointNet Block
Hypothesis Surface
Phase 2
Extract surface
Ceiling
project
Floor
Hypothesis surface
Gaussian Scene
PointNet++
Surface Interpolation
fill
Predicted Gaussians
Completed Gaussian Scene
Depth
$\mathcal{L}_{\text{sdf}}$
$\mathcal{L}_{\text{normal}}$
$\Delta\phi = \text{div}\,\mathbf{X}$
Mesh Extration
Mesh

Figure 2: The two-phase pipeline illustration of the proposed MG-SLAM. The upper section visualizes the parallel processes of the tracking and mapping systems. The lower section presents the post-optimization of scene completion and mesh extraction.

3.2 Tracking

We utilize the backprojection of point features and line segments extracted from 2D images into 3D space in parallel for optimizing the camera pose, based on the approach outlined in PLVS [14]. Moreover, we incorporate strategies for line segment fusion and suppression, designed to merge shorter segments into longer ones and eliminate unstable lines. This method is grounded in the understanding that longer line features tend to offer greater reliability and robustness throughout the tracking process.

3.2.1 Point reprojection error

Given a point observation p_i within a specific keyframe k, its reprojection error can be computed as:

$$\mathcal{L}_{2D}^{p} = \|\mathcal{P}_i - \pi(R_k \mathcal{X}_i^w + T_k)\|_{\Sigma_{\mathcal{P}_i}} \in \mathbb{R} \tag{1}$$

where $\mathcal{X}_i^w$ is the 3D point back-projected to the world domain and transformed to the camera frame $\mathcal{X}_i^c$. The term $\Sigma_{\mathcal{P}i} = \sigma^2 \mathcal{P}_i \mathbb{I}_2$ represents the covariance matrix, encapsulating the variance $\sigma^2 \mathcal{P}_i$ of point detection noise at different scales within a Gaussian image pyramid [14].

3.2.2 Line reprojection and backprojection error

For line segments, represented in the 3D world as a pair of endpoints $(\mathcal{X}_p, \mathcal{X}_q) \in \mathbb{R}^3$ and in the image plane as $(\mathcal{P}_p, \mathcal{P}_q) \in \mathbb{R}^2$, we identify the line segments on each level of the image pyramid using the EDlines method [1] for efficient extraction. Matching line segments leverage the Line Band Descriptor (LBD) method [47], and matches are scored according to the Hamming distances between descriptors.

Upon observing a line segment $l = (\mathcal{P}_p, \mathcal{P}_q)_k$, its back-projected 3D endpoints can be computed as $L = (\theta(\mathcal{P}_p), \theta(\mathcal{P}_q))_k$. Then the reprojection error of the line segment can be interpreted as:

$$\mathcal{L}_{2D}^{l} = \left[\left\| \begin{matrix} d_k \cdot \pi(R_k \mathcal{X}_p^w + t) \\ d_k \cdot \pi(R_k \mathcal{X}_q^w + t) \end{matrix} \right\| \right]_{\Sigma_{2D}} \in \mathbb{R}^2 \ , \quad d_k = \frac{\bar{\mathcal{P}}_p \times \bar{\mathcal{P}}_q}{\|\bar{\mathcal{P}}_p \times \bar{\mathcal{P}}_q\|} |_k \tag{2}$$

where $\bar{\mathcal{P}}_p = [\mathcal{P}_p^T, 1]^T$, $\bar{\mathcal{P}}_q = [\mathcal{P}_q^T, 1]^T$. The covariance matrix $\Sigma_{2D} = \sigma_{d_i}^2 \mathbb{I}_2$ is assumed to be diagonal for simplicity. Further, we calculate the perpendicular distance in 3D between a map point $\mathcal{X}_i^w \in \mathbb{R}^3$ and a 3D line segment L as follows:

$$e_{3D}^{l}(L, \mathcal{X}_i^c)_k = \frac{\|(\mathcal{X}_i^c - \theta(\mathcal{P}_p)) \times (\mathcal{X}_i^c - \theta(\mathcal{P}_q))\|_k}{\|\theta(\mathcal{P}_p) - \theta(\mathcal{P}_q)\|_k} \tag{3}$$

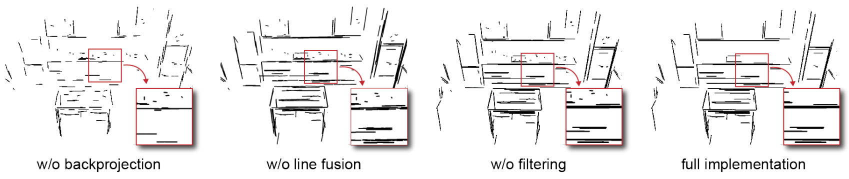

Figure 3: The ablation of the line segment extraction outcomes for our method on the scene frl_apartment_4 from the Replica Apartment dataset [38].

which considers the cross-product of the differences between the point's position and the back-projected positions of the line's endpoints, normalized by the distance between these back-projected endpoints. Additionally, the distance between the world domain point $\mathcal{X}_i^w$ and its corresponding back-projected point in the image plane $\mathcal{P}_i \in \mathbb{R}^2$ is determined by:

$$e^l_{\mathcal{P}}(\mathcal{P}_i, \mathcal{X}_i^c) = \|\mathcal{X}_i^c - \theta(\mathcal{P}_i)\|_k \ , \ \ \mathcal{X}_i^c = R_k \mathcal{X}_i^w + T_k \tag{4}$$

Consequently, the back projection error is formulated as:

$$\mathcal{L}^l_{\text{3D}} = \begin{bmatrix} e_{\text{3D}}(\theta(\mathcal{P}_i), \mathcal{X}_p^w) + \beta \cdot e_{\mathcal{P}}(\mathcal{P}_i, \mathcal{X}_p^w) \\ e_{\text{3D}}(\theta(\mathcal{P}_i), \mathcal{X}_q^w) + \beta \cdot e_{\mathcal{P}}(\mathcal{P}_i, \mathcal{X}_q^w) \end{bmatrix}_k \in \mathbb{R}^2 \tag{5}$$

In this context, $\beta \in [0, 1]$ acts as a weighting parameter, ensuring the endpoints of the 3D line segment remain stable throughout the optimization process, thereby preventing drift.

3.2.3 Line segment fusion

The stability of tracking can be notably enhanced by the presence of elongated line segments. However, the extraction process, particularly through EDLines, often yields fragmented segments as a consequence of various disturbances like image noise. Our method integrates the following key steps: (1) Fusing line segments that are directionally aligned within a one-degree angle difference, ensuring they follow the same path. (2) Ensuring these segments are close—within a 10-pixel distance of each other's nearest endpoints—yet do not overlap, preserving their distinctness while allowing for precise merging. (3) Verifying the vertical distance between the corresponding endpoints of one segment to the entirety of another is less than certain pixel threshold, maintaining geometric consistency. Only segments fulfilling all three criteria are merged, producing longer and more reliable lines. Furthermore, we filter out segments that fall below a predefined length threshold, relative to the image size, to improve the system's tracking robustness. Figure 3 displays the ablation results for our feature extraction approach for 3D line segments, highlighting the effectiveness of the backprojection, line fusion, and filtering strategies.

3.2.4 Full bundle adjustment error

The overall objective function minimized during the full bundle adjustment is given by:

$$\mathcal{L}_{\text{BA}} = \sum_{\mathcal{K}} \sum_{\mathcal{U}} \rho\left(\|\mathcal{L}^p_{\text{2D}}\|_{\Sigma_{\mathcal{P}}}\right) + \sum_{\mathcal{V}} \rho\left(\|\mathcal{L}^l_{\text{2D}}\|_{\Sigma_{\text{2D}}}\right) + \sum_{\mathcal{V}} \rho\left(\|\mathcal{L}^l_{\text{3D}}\|_{\Sigma_{\text{3D}}}\right) \tag{6}$$

Here, $\mathcal{K}$ represents the set of chosen keyframes, while $\mathcal{U}$ and $\mathcal{V}$ denote the sets of points and lines extracted within the current frame. The Huber cost function $\rho(\|e\|)_{\Sigma^{-1}}) = e^T \Sigma^{-1} e$ is applied to mitigate the influence of outliers. The optimization process utilizes the Levenberg-Marquardt method, which solves the augmented normal equations by iteratively updating the parameters Θ as:

$$(J^T \Sigma^{-1} J + \lambda \mathbb{I}) \Delta\Theta = -J^T \Sigma^{-1} \epsilon \tag{7}$$

where ϵ aggregates the reprojection and backprojection errors defined in Equation (6), and Σ combines the covariance matrices relevant to these errors.

3.3 Mapping

The map representation is based on Gaussians ellipsoids $\mathcal{G} = \alpha\mathcal{N}(\mu^w, \Sigma^w)$, where $\alpha \in [0, 1]$ is the opacity, μ^w and Σ^w are mean and covariance in world coordinate. Each $\mathcal{G}$ is associated with the color of appearance $\mathcal{C}$ and semantic color $\mathcal{S}$. Semantic segmentation is utilized to identify the surface such as floors for structure optimization and surface extraction, explained in Section 3.4.

3.3.1 Scene representation

We use the standard point rendering formula [45, 23] to splat $\mathcal{G}$ to render 2D image:

$$\mu_{\mathcal{P}} = \pi(R\mu_{\mathcal{G}} + T),\ \Sigma_{\mathcal{P}} = JR\Sigma_{\mathcal{G}}R^T J^T \tag{8}$$

where $\mu_{\mathcal{G}}$ and $\Sigma_{\mathcal{G}}$ are mean and covariance of the Gaussian. J is the Jacobian of the linear approximation of $\pi(\cdot)$. For each pixel pix $\in \mathcal{P}$, the influence of N Gaussians on this pixel can be combined by sorting the Gaussians in depth order and performing front-to-back volume rendering using the Max volume rendering formula [29]:

$$\mathcal{P}_{\text{pix}} = \sum_{i \in N} p_i \alpha_i \prod_{j=1}^{i-1}(1 - \alpha_j),\ \mathcal{D}_{\text{pix}} = \sum_{i \in N} z_i \alpha_i \prod_{j=1}^{i-1}(1 - \alpha_j) \tag{9}$$

where $\mathcal{P}_{\text{pix}}$ and $\mathcal{D}_{\text{pix}}$ represent the pixel-wise color ($\mathcal{C}_{\text{pix}}$ and $\mathcal{S}_{\text{pix}}$) and depth respectively. z is the distance to the mean of the Gaussian $\mathcal{G}$ along the camera ray.

3.3.2 Mapping loss

The scene representation is optimized based on the selected keyframes obtained from the tracking system. Specifically, we minimize the following photometric residual:

$$\mathcal{L}_{\text{pix}} = \left\|\mathcal{C}_{\text{pix}} - \mathcal{C}_{\text{pix}}^{\text{GT}}\right\| + \lambda_{\mathcal{S}} \left\|\mathcal{S}_{\text{pix}} - \mathcal{S}_{\text{pix}}^{\text{GT}}\right\| + \lambda_{\mathcal{D}} \left\|\mathcal{D}_{\text{pix}} - \mathcal{D}_{\text{pix}}^{\text{GT}}\right\| \tag{10}$$

In this context, Gaussians are optimized to adjust their optical and geometrical parameters to closely replicate the observed scene with fine-grained details. Furthermore, to impose constraints on the scene utilizing the identified line features that precisely delineate the lines and potential edges in the current view, an additional line residual is introduced for pixels associated with line features to enhance the map's accuracy. Consequently, the aggregate loss for mapping is determined as follows:

$$\mathcal{L}_{\text{Mapping}} = \sum_{\text{pix} \in \mathcal{P}} \mathcal{L}_{\text{pix}} + \lambda_{\ell} \sum_{\text{pix} \in \mathcal{V}} \mathcal{L}_{\text{pix}} \tag{11}$$

$\lambda_{\mathcal{S}}$, $\lambda_{\mathcal{D}}$ and λ_{ℓ} in Equation (10) and Equation (11) are weighting parameters.

3.4 Structure optimization

In our optimization process, we focus particularly on refining the representation of expansive surfaces that define the space, such as the floor and ceiling. By applying the Manhattan World assumption, we introduce the planar hypothesis surfaces that are informed by the structural regularity of the extracted 3D line features. These constraints are employed to identify surfaces that are not adequately represented and to interpolate new Gaussians for densifying or filling in gaps with textures that are reasonably predicted. Specifically, our method consists of the following steps:

Calibration Since the SLAM system relies on the initial camera pose as a reference frame, the reconstructed scenes usually do not satisfy our orthogonality assumption. As a result, we calibrate the reconstructed scene by applying a calibrating matrix K to the coordinates and rotations of the Gaussians. This matrix K is derived from clustering the directions of 3D line segments that are presumed to align with the scene axes.

Surface interpolation After aligning the structured surface boundaries to orthogonal directions, they extend across the $xy \to \mathbb{R}^2$ plane. We use the calibrated line segments to outline the rectangular

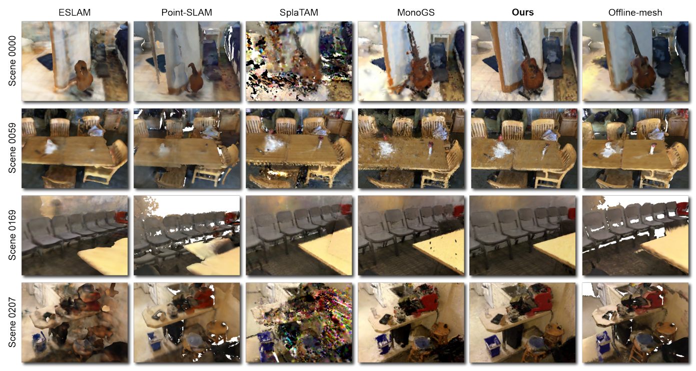

Figure 4: Qualitative comparison of our method and the baselines for novel-view synthesis on the ScanNet dataset [6]. The outcomes show that our method provides more robust and fine-grained reconstructions in real-world complex scenes.

boundary of the hypothesized plane, capitalizing on the dense line features commonly found at the corners of scenes. The chosen target Gaussians representing the surface are then projected from their 3D form onto the 2D hypothesis xy plane. Then we apply a density threshold to a predefined grid using the density function (Equation (13)) to detect any holes or gaps on the surface. New Gaussians are generated at the identified gaps that fall below a density threshold. We employ a PointNet++ model ([34]), fitting it to the presented surface to learn the color and texture patterns. This model then interpolates the texture color of the new Gaussians based on their spatial properties.

Through this method, we aim to seamlessly combine the detailed representation provided by Gaussians with the need for explicit scene geometry that accounts for unseen views, thereby achieving a more comprehensive and accurate reconstruction of indoor environments. The detailed explanation of the algorithms is outlined in Appendix B.2.

4 Experiment

4.1 Experimental setup

Datasets We evaluate our method using two datasets: Replica [38], a synthetic dataset, and ScanNet [6], a challenging real-world dataset. For Replica [38], the ground-truth camera pose and semantic maps are obtained through Habitat simulation [37]. In the case of ScanNet [6], the ground-truth camera poses are derived using BundleFusion [7].

Metrics To assess the quality of the reconstruction, we employ metrics such as PSNR, Depth-L1, SSIM, and LPIPS. For evaluating camera pose, we use the average absolute trajectory error (ATE RMSE). The real-time processing capability, essential for SLAM systems, is measured in frames per second (FPS).

Baselines We evaluate our tracking and mapping results against state-of-the-art methods, including NeRF-based approaches such as NICE-SLAM [51], ESLAM [18], and Point-SLAM [36], as well as recent Gaussian-based methods like SplaTAM [19] and MonoGS [28]. The results for MonoGS [28] are obtained using its RGB-D mode for fair comparison.

Ablation The ablation study demonstrating the effectiveness of each proposed component is detailed in Appendix C.

4.2 Evaluation of localization and reconstruction

In Figure 4, we visualize the novel-view synthesis results for MG-SLAM, comparing it against baseline methods in real-world scenes. Our method exhibits robust and high-fidelity reconstruction capabilities. In comparison to Point-SLAM [36], MG-SLAM offers complete scene reconstructions and finer texture details, benefiting from the use of Gaussians to handle complex geometries. Among Gaussian-based methods, SplaTAM [19] struggles in large multi-room scenes with complex camera

Table 1: Quantitative comparison of our method and the baselines in training view rendering on the ScanNet dataset [6]. Our method demonstrates SOTA performances among three metrics.

Methods	Metrics	Avg.	0000	0059	0106	0169	0207
ESLAM [18]	PSNR↑	15.48	15.67	14.45	15.42	14.55	17.32
	SSIM↑	0.650	0.686	0.632	0.627	0.655	0.650
	LPIPS↓	0.492	0.452	0.452	0.528	0.492	0.538
Point-SLAM [36]	PSNR↑	18.61	20.12	18.58	16.47	18.23	19.66
	SSIM↑	0.737	0.806	0.765	0.676	0.686	0.750
	LPIPS↓	0.523	0.485	0.499	0.544	0.542	0.544
SplaTAM [19]	PSNR↑	19.02	17.81	19.60	19.23	20.55	17.95
	SSIM↑	0.726	0.602	0.796	0.741	0.785	0.705
	LPIPS↓	0.337	0.467	0.290	0.322	0.260	0.346
MonoGS [28]	PSNR↑	20.08	19.76	19.25	20.18	20.57	20.62
	SSIM↑	0.782	0.772	0.767	0.785	0.790	0.798
	LPIPS↓	0.300	0.387	0.289	0.272	0.256	0.295
Ours	PSNR↑	**24.60**	**25.69**	**24.51**	**23.88**	**24.27**	**24.64**
	SSIM↑	**0.869**	**0.885**	**0.867**	**0.859**	**0.864**	**0.872**
	LPIPS↓	**0.244**	**0.265**	**0.232**	**0.240**	**0.227**	**0.256**

Table 2: Quantitative comparison of our method and the baselines in terms of ATE [cm] on the ScanNet dataset [6]. Our method demonstrates SOTA performances in most of presented scenes.

Methods	Avg.	0000	0059	0106	0169	0207
ESLAM [18]	7.68	8.47	8.70	**7.58**	7.45	6.20
Point-SLAM [36]	11.68	10.24	7.81	8.65	22.16	9.54
SplaTAM [19]	12.04	12.83	10.10	17.72	12.08	7.46
MonoGS [28]	13.86	15.94	13.03	19.44	10.44	10.46
Ours	**6.77**	**5.95**	**6.41**	8.07	**7.29**	**6.14**

loops due to its lack of bundle adjustment in the tracking system. Similarly, MonoGS [28] generally delivers reliable quality but struggles with object-level reconstruction drift. In contrast, MG-SLAM excels with robust tracking and effective bundle adjustment incorporating line segments, enabling superior detailed reconstructions even surpassing the ground-truth mesh derived from offline methods.

We provide quantitative assessments of reconstruction quality using the ScanNet dataset [6] in Table 1 and the Replica-V1 dataset [38] in Appendix D.1. Our approach delivers SOTA results, outperforming other Gaussian-based methods by a notable 4dB in PSNR on both datasets. The tracking evaluation results are shown in Table 2. Our method remarkably reduces the ATE RMSE (cm) error, achieving 50% improvements over Gaussian baselines. Moreover, benefiting from our line fusion strategy, MG-SLAM also shows superior tracking performance over traditional SLAM systems [31, 2, 14], as detailed in Appendix D.3.

Table 3: Quantitative comparison of our method and the Gaussian baselines in training view rendering on the Replica Apartment dataset [38]. Our method demonstrates SOTA performances.

Methods	Metrics	Avg.	apart.0	apart.1	apart.2	frl_apart.0	frl_apart.4
SplaTAM [19]	PSNR↑	25.16	13.12	24.57	25.52	30.72	31.86
	SSIM↑	0.790	0.415	0.821	0.858	0.924	0.930
	LPIPS↓	0.323	0.656	0.302	0.258	0.201	0.198
MonoGS [28]	PSNR↑	26.87	21.89	26.87	27.92	30.70	26.97
	SSIM↑	0.868	0.864	0.856	0.873	0.886	0.863
	LPIPS↓	0.285	0.397	0.285	0.273	0.225	0.247
Ours	PSNR↑	**30.63**	**28.76**	**30.84**	**29.07**	**31.85**	**32.65**
	SSIM↑	**0.925**	**0.905**	**0.924**	**0.917**	**0.932**	**0.945**
	LPIPS↓	**0.236**	**0.328**	**0.252**	**0.241**	**0.174**	**0.184**

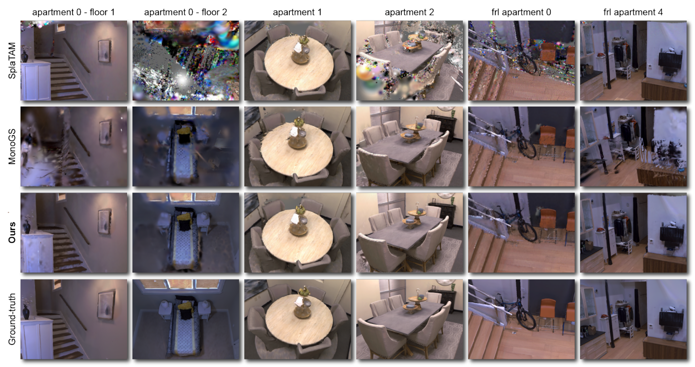

Figure 5: Qualitative comparison of our method and the Gaussian SLAM baselines for novel-view synthesis on the Replica Apartment dataset [38]. More detailed figures are available in Appendix D.2.

4.3 Evaluation on large-scale dataset

To evaluate the robustness of the system in large-scale indoor environments, we evaluate MG-SLAM on the Replica Apartment dataset [38]. This dataset contains extensive multi-room scenes, complex object geometries, and looping trajectories across rooms. Table 3 presents the rendering quality of our method compared to SplaTAM [19] and MonoGS [28] over five selected scenes. MG-SLAM shows a notable improvement over these baselines, particularly achieving a 7dB improvement in the extensive *apartment 0* scene, which features a two-floor, eight-room layout. This optimal performance is largely attributed to the inclusion of the fused line segments, which lays a solid foundation for loop closure and pose optimization. Figure 5 illustrates the results of novel-view rendering for apartment scenes. Our method shows remarkable enhancements over Gaussian baselines in geometry accuracy and the richness of fine details.

4.4 Evaluation of scene completion

The Gaussian SLAM faces limitations in inferring the geometry of unseen views. This issue is especially evident in indoor scenes that feature complex geometry, where basic surfaces like floors are often obscured and poorly represented. Figure 6 compared our method with SplaTAM [19] and MonoGS [28] in scenes with missing geometry. Utilizing the surface optimization strategy based on the Manhattan World hypothesis [5], our method can accurately detect and proactively generate new Gaussians efficiently to fill gaps with certain textures, whereas Gaussian baseline methods exhibit substantial defects on structured surfaces. Demonstrations for real-world scenes can be found in Appendix D.4.

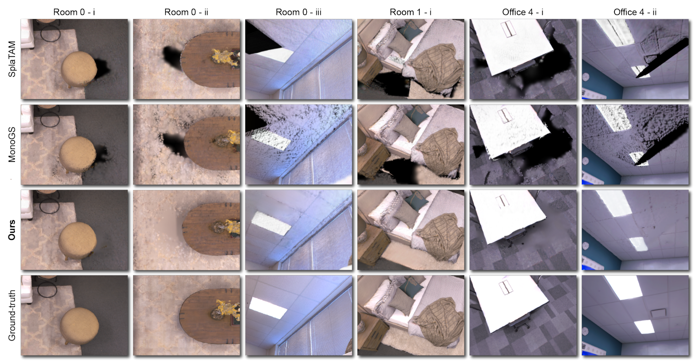

Figure 6: Qualitative comparison of our method and the Gaussian SLAM baselines for reconstruction across selected scenes from the Replica dataset [38].

5 Conclusion

In this study, we present MG-SLAM, a Gaussian-based SLAM method based on the Manhattan World hypothesis. MG-SLAM employs line segments for robust pose estimation and map refinement. Furthermore, by leveraging the line segments and planar assumption, we efficiently interpolate new Gaussians on gaps of missing geometry. Extensive experiments demonstrate that our method delivers state-of-the-art tracking and mapping performance, while also achieving remarkable high speed detailed in Appendix D.5. Limitations of our work are discussed in Appendix E.

References

[1] Cuneyt Akinlar and Cihan Topal. Edlines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642, 2011.

[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] Qi Chen, Yu Cao, Jiawei Hou, Guanghao Li, Shoumeng Qiu, Bo Chen, Xiangyang Xue, Hong Lu, and Jian Pu. Vpl-slam: A vertical line supported point line monocular slam system. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[4] Chi-Ming Chung, Yang-Che Tseng, Ya-Ching Hsu, Xiang-Qian Shi, Yun-Hung Hua, Jia-Fong Yeh, Wen-Chin Chen, Yi-Ting Chen, and Winston H Hsu. Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9400–9406. IEEE, 2023.

[5] James Coughlan and Alan L Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. *Advances in Neural Information Processing Systems*, 13, 2000.

[6] 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/CVF Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017.

[7] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4):1, 2017.

[8] Tianchen Deng, Guole Shen, Tong Qin, Jianyu Wang, Wentao Zhao, Jingchuan Wang, Danwei Wang, and Weidong Chen. Plgslam: Progressive neural scene represenation with local to global bundle adjustment. *arXiv preprint arXiv:2312.09866*, 2023.

[9] Tianchen Deng, Hongle Xie, Jingchuan Wang, and Weidong Chen. Long-term visual simultaneous localization and mapping: Using a bayesian persistence filter-based global map prediction. *IEEE Robotics & Automation Magazine*, 30(1):36–49, 2023.

[10] Tianchen Deng, Yaohui Chen, Leyan Zhang, Jianfei Yang, Shenghai Yuan, Danwei Wang, and Weidong Chen. Compact 3d gaussian splatting for dense visual slam. *arXiv preprint arXiv:2403.11247*, 2024.

[11] Tianchen Deng, Nailin Wang, Chongdi Wang, Shenghai Yuan, Jingchuan Wang, Danwei Wang, and Weidong Chen. Incremental joint learning of depth, pose and implicit scene representation on monocular camera in large-scale scenes. *arXiv preprint arXiv:2404.06050*, 2024.

[12] Tianchen Deng, Yanbo Wang, Hongle Xie, Hesheng Wang, Jingchuan Wang, Danwei Wang, and Weidong Chen. Neslam: Neural implicit mapping and self-supervised feature tracking with depth completion and denoising. *arXiv preprint arXiv:2403.20034*, 2024.

[13] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[14] Luigi Freda. Plvs: A slam system with points, lines, volumetric mapping, and 3d incremental segmentation. *arXiv preprint arXiv:2309.10896*, 2023.

[15] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *arXiv preprint arXiv:2311.12775*, 2023.

[16] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3d scene reconstruction with the manhattan-world assumption. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5511–5520, 2022.

[17] Jiaming He, Mingrui Li, Yangyang Wang, and Hongyu Wang. Ple-slam: A visual-inertial slam based on point-line features and efficient imu initialization. *arXiv preprint arXiv:2401.01081*, 2024.

[18] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17408–17419, 2023.

[19] 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. *arXiv preprint arXiv:2312.02126*, 2023.

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023.

[21] Lukas Koestler, Nan Yang, Niclas Zeller, and Daniel Cremers. Tandem: Tracking and dense mapping in real-time using deep multi-view stereo. In *Conference on Robot Learning*, pages 34–45. PMLR, 2022.

[22] Xin Kong, Shikun Liu, Marwan Taher, and Andrew J Davison. vmap: Vectorised object mapping for neural field slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 952–961, 2023.

[23] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, volume 40, pages 29–43. Wiley Online Library, 2021.

[24] Mingrui Li, Jiaming He, Yangyang Wang, and Hongyu Wang. End-to-end rgb-d slam with multi-mlps dense neural implicit representations. *IEEE Robotics and Automation Letters*, 8(11): 7138–7145, 2023.

[25] Mingrui Li, Jiaming He, Guangan Jiang, and Hongyu Wang. Ddn-slam: Real-time dense dynamic neural implicit slam with joint semantic encoding. *arXiv preprint arXiv:2401.01545*, 2024.

[26] Mingrui Li, Shuhong Liu, and Heng Zhou. Sgs-slam: Semantic gaussian splatting for neural dense slam. *arXiv preprint arXiv:2402.03246*, 2024.

[27] Yanyan Li, Chenyu Lyu, Yan Di, Guangyao Zhai, Gim Hee Lee, and Federico Tombari. Geogaussian: Geometry-aware gaussian splatting for scene rendering. *arXiv preprint arXiv:2403.11324*, 2024.

[28] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. *arXiv preprint arXiv:2312.06741*, 2023.

[29] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[30] John McCormac, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level slam. In *2018 international conference on 3D vision (3DV)*, pages 32–41. IEEE, 2018.

[31] 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.

[32] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[33] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011.

[34] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[35] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013.

[36] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18433–18444, 2023.

[37] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[38] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[39] Fabio Tosi, Youmin Zhang, Ziren Gong, Erik Sandström, Stefano Mattoccia, Martin R Oswald, and Matteo Poggi. How nerfs and 3d gaussian splatting are reshaping slam: a survey. *arXiv preprint arXiv:2402.13255*, 2024.

[40] Evangelos Ververas, Rolandos Alexandros Potamias, Jifei Song, Jiankang Deng, and Stefanos Zafeiriou. Sags: Structure-aware 3d gaussian splatting. *arXiv preprint arXiv:2404.19149*, 2024.

[41] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13293–13302, 2023.

[42] Lei Xu, Hesheng Yin, Tong Shi, Di Jiang, and Bo Huang. Eplf-vins: Real-time monocular visual-inertial slam with efficient point-line flow features. *IEEE Robotics and Automation Letters*, 8(2):752–759, 2022.

[43] Zewen Xu, Hao Wei, Fulin Tang, Yidi Zhang, Yihong Wu, Gang Ma, Shuzhe Wu, and Xin Jin. Plpl-vio: a novel probabilistic line measurement model for point-line-based visual-inertial odometry. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5211–5218. IEEE, 2023.

[44] Chi Yan, Delin Qu, Dong Wang, Dan Xu, Zhigang Wang, Bin Zhao, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. *arXiv preprint arXiv:2311.11700*, 2023.

[45] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019.

[46] Vladimir Yugay, Yue Li, Theo Gevers, and Martin R Oswald. Gaussian-slam: Photo-realistic dense slam with gaussian splatting. *arXiv preprint arXiv:2312.10070*, 2023.

[47] Lilian Zhang and Reinhard Koch. An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency. *Journal of visual communication and image representation*, 24(7):794–805, 2013.

[48] Weiqiang Zhao, Hang Sun, Xinyu Zhang, and Yijin Xiong. Visual slam combining lines and structural regularities: Towards robust localization. *IEEE Transactions on Intelligent Vehicles*, 2023.

[49] Heng Zhou, Zhetao Guo, Shuhong Liu, Lechen Zhang, Qihao Wang, Yuxiang Ren, and Mingrui Li. Mod-slam: Monocular dense mapping for unbounded 3d scene reconstruction. *arXiv preprint arXiv:2402.03762*, 2024.

[50] Lipu Zhou, Guoquan Huang, Yinian Mao, Shengze Wang, and Michael Kaess. Edplvo: Efficient direct point-line visual odometry. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7559–7565. IEEE, 2022.

[51] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022.

[52] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. *arXiv preprint arXiv:2302.03594*, 2023.

Structure Gaussian SLAM with Manhattan World Hypothesis

— Appendix —

A Implementation details

In this section, we describe the implementation details of our systems. Our tracking system is implemented in C++, while the mapping system uses Python3 and CUDA C. The experiments were conducted using a single RTX A100-80GB GPU and 24-core AMD EPYC 7402 processor. We use Adam optimizer for Gaussian representation optimization and network training. The hyperparameters for 3D Gaussian splitting [20] are the same as the original paper. Table 4 presents the values for hyperparameters in our system. For Gaussian insertion, in each keyframe, we sampled $\mu_{\mathcal{G}}$ in Equation (8) from the rendered depth map $\mathcal{D}_{\text{pix}}$ following the distribution of $\mathcal{N}(\mathcal{D}_{\text{pix}}, 0.2\sigma_{\mathcal{D}})$. For uninitialized areas, we initialize Gaussians by sampling from $\mathcal{N}(\bar{\mathcal{D}}_{\text{pix}}, 0.5\sigma_{\mathcal{D}})$ where $\bar{\mathcal{D}}_{\text{pix}}$ is the mean value. For pruning, we remove the Gaussians with opacity less than 0.6. Our system incorporates a segmentation loss introduced in [26], and we leverage this semantic information to effectively extract structured surfaces, such as floors and ceilings, from the reconstructed scenes. This segmentation sufficiently fulfills our objective to extract the current surfaces and train a PointNet++ [34] network to predict the textures of interpolated Gaussians, which are used to fill the missing geometry.

Table 4: MG-SLAM hyperparameters

Symbol	Explanation	Value
N_p	number of point features in tracking	4e3
N_ℓ	number of line segments in tracking	200
r_d	down-sample ratio for Gaussian sampling from point cloud	8
n_{map}	mapping iteration	150
SH_{degree}	degree of the spherical harmonics	0
l_{feat}	learning rate for SH features	2.5e-3
$l_{opacity}$	learning rate for opacity	5e-2
l_{scale}	learning rate for scaling	1e-3
l_{rotat}	learning rate for rotation	1e-3
$\lambda_{\mathcal{D}}$	weighting term for depth loss in mapping	0.1
$\lambda_{\mathcal{S}}$	weighting term for semantic loss in mapping	0.1
λ_ℓ	re-weighting term for line feature loss	0.25
$\mathcal{T}_\ell$	line segment filtering threshold (relative to image scale)	8e-2
$\mathcal{T}_{density}$	density threshold for hole detection	0.9
$b_{\mathcal{G}}$	batch size of Gaussian partition to train PointNet++ [34]	1e4
n_e	number of epoch to train PointNet++ [34]	25

B Additional methods

B.1 Gaussian density

In our Gaussian scenes, we define the density of the Gaussian $\nu : \mathbb{R}^3 \to \mathbb{R}_+$ as the sum of the Gaussian values weighted by their alpha-blending coefficients at any given grid points p as:

$$\nu(p) = \sum_{\mathcal{G}_i} \alpha_{\mathcal{G}_i} \exp(-\frac{1}{2}(p - \mu_{\mathcal{G}_i}^T)\Sigma_{\mathcal{G}_i}^{-1}(p - \mu_{\mathcal{G}_i}) \ , \tag{12}$$

where the $\mu_{\mathcal{G}_i}, \Sigma_{\mathcal{G}_i}, \alpha_{\mathcal{G}_i}$ are the means, covariances, and alpha-blending coefficients of the Gaussians. To simplify the calculation, $\nu(p)$ can be approximated by:

$$\nu^*(p) = \alpha_{\mathcal{G}^*} \exp(-\frac{1}{2}(p - \mu_{\mathcal{G}^*}^T)\Sigma_{\mathcal{G}^*}^{-1}(p - \mu_{\mathcal{G}^*}) \ , \tag{13}$$

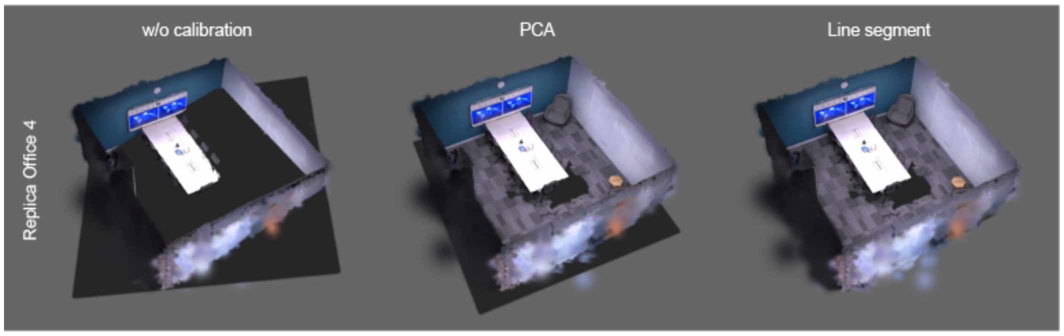

Figure 7: The ablation study compares surface identification approaches. The identified hypothesis surfaces using different methods are shaded in black. The left figure displays the scene without calibration, where the primary axes of the scene do not align with the world coordinates. The middle figure demonstrates the application of PCA to the centers of Gaussians $\mu_\mathcal{G}$ associated with structured surfaces, where the boundaries of $\mu_\mathcal{G}$ define the surface. The right figure shows the calibration results, achieved by clustering line segments and defining surfaces based on the boundaries of these line features.

where $\mathcal{G}^*$ is the nearest Gaussian that contributes most to p. This Gaussian density function $\nu^*(\cdot)$ facilitates the identification of holes on the reconstructed surface (Appendix B.2) and surface extraction in mesh generation (Appendix B.3).

B.2 Structured surface completion

In this section, we outline the detailed algorithms for our structured surface completion strategy. Algorithm 1 provides the pseudo-code for calibrating and identifying surface boundaries. Initially, we eliminate short line segments, which, despite their usefulness for tracking, are often considered noisy and less representative of the primary axes of the scene. By clustering the remaining longer segments, we identify the principal axes of the scene, aligning them with the three-dimensional coordinate system of the world through a calibration matrix. This calibration process ensures that the scene adheres to the Manhattan World assumption, where structured surfaces such as walls and floors are either parallel or perpendicular to the world coordinate axes.

Subsequently, we define the hypothesis surface, which acts as the ideal complete surface for interpolating holes. Specifically, we project 3D line features onto the plane $\bar{z}$, using the boundaries of these line features to define the hypothesis surface. Figure 7 illustrates the ablation of our calibration and surface identification method. Without calibration, we use the boundaries of $\mu_\mathcal{G}$, which are associated with structured surfaces, as the plane boundary and the mean value of $z \in \mu_\mathcal{G}$ for height. However, this approach results in misaligning the hypothesis surface with the actual floor. We also present a result using PCA on the centers of target Gaussians $\mu_\mathcal{G}$ to identify primary axes and calibrate the scene. Despite this effort, the predicted surface remains inaccurately positioned relative to the target surface, particularly showing disparities along the edges. And this disparity increases as the gaps on the surface become larger. By contrast, utilizing concrete 3D line features, which accurately represent scene boundaries and reflect primary axes, ensures precise identification of structured surfaces.

After calibration, we strategically interpolate new Gaussians to fill the holes and gaps in structured surfaces. The pseudo-code for our scene completion process is detailed in Algorithm 2. Holes on the hypothetical surface are specifically identified by the density function $\nu^*(p)$, as defined in Equation (13), where $\nu^*(p)$ falls below a threshold $\mathcal{T}_{density}$. To seamlessly fill these holes with appropriate texture, we train a PointNet++ model [34] that takes the Gaussian center $\mu\mathcal{G}_i$ as input and outputs texture colors. This approach is based on the understanding that surface textures in indoor scenes are regular and can be effectively predicted using spatial correlations. The training involves using Gaussians that are randomly sampled from the extracted incomplete surfaces, and we predict the texture color of interpolated Gaussians positioned at the holes. The scales of the generated Gaussians in the first two dimensions are determined based on the mean values of all existing Gaussians on the surface, while the third dimension is set to zero. This ensures the generated Gaussians are flat, facilitating the mesh extraction process.

We employed the PointNet++ model [34] for its proven efficiency in processing point clouds, capturing both local and global structural information. This model is ideally suited for the Manhattan World hypothesis, where the consistent flatness and regulation of surfaces simplify the task of texture interpolation. The straightforward yet effective architecture of the PointNet++ [34] allows it to discern detailed spatial patterns, seamlessly fits our framework to provide robust prediction without the computational overhead of more complex models.

Algorithm 1 The pseudo-code for calibration and identification of surface boundary

```
Require: Line segments $L \in \mathbb{R}^3$ and the reconstructed Gaussian scene $\mathcal{G}$
 1: $L_{\text{filtered}} \leftarrow$ select $|\ell| > \mathcal{T}_\ell$        ▷ Filter out small line segments
 2: for each line segment $\ell$ in $L_{\text{filtered}}$ do
 3:     $\{direction\} \leftarrow (\ell.end - \ell.start).normalized()$        ▷ Add direction vector
 4: end for
 5: $\vec{x}, \vec{y}, \vec{z} \leftarrow k\_means(\{direction\}, k = 3)$        ▷ Cluster directions to find three main axes
 6: $K \leftarrow \mathbb{I} \in \mathbb{R}^4$        ▷ Define calibration matrix
 7: $K[0:3, 0] \leftarrow \vec{x}$
 8: $K[0:3, 1] \leftarrow \vec{y}$
 9: $K[0:3, 2] \leftarrow \vec{z}$
10: for each $\mathcal{G}_i$ in $\mathcal{G}$ do
11:     $\mu_{\mathcal{G}_i} \leftarrow K \cdot \mu_{\mathcal{G}_i}$        ▷ Calibrate coordinates of Gaussian
12:     $\gamma_{\mathcal{G}_i} \leftarrow K \cdot \gamma_{\mathcal{G}_i}$        ▷ Calibrate rotation of Gaussian
13: end for
14: for each line segment $\ell$ in $L$ do
15:     $\ell \leftarrow K \cdot \ell$        ▷ Calibrate line segment
16: end for
17: $\mathcal{B} \leftarrow$ select maximum and minimum $x$ and $y$ for endpoints in $L$        ▷ Define surface boundary
18: Output: boundary of surface $\mathcal{B} \in \mathbb{R}^2$
```

Algorithm 2 The pseudo-code for scene completion

```
Require: The calibrated Gaussian scene $\mathcal{G}$, the mask $\mathcal{M}$ extracted from semantic segmentation, and
    the boundary of surface $\mathcal{B}$, threshold of density $\mathcal{T}_{density}$.
 1: $\mathcal{M}_{tg} \leftarrow$ semantic masks that represent target surfaces of the scene, e.g. floor
 2: for each surface $\mathcal{G}_i$ in $\mathcal{G}(\mathcal{M}_{tg})$ do
 3:     $\mathcal{G}_i \leftarrow DBSCAN.fit(\mathcal{G}_i)$        ▷ use DBSCAN [13] to eliminate outliers
 4:     $\bar{z} \leftarrow avg(z)$ for $z \in \mu_{\mathcal{G}_i}$
 5:     $\Pi \leftarrow grid(\mathcal{B})$        ▷ Create the 2D grid of hypothesis surface
 6:     $\mathcal{D}_i \leftarrow$ compute the Gaussian density for each point of the grid in $\Pi$.
 7:     if $\mathcal{D}_i < \mathcal{T}_{density}$ then
 8:         generates a new Gaussian onto the grid of low-density
 9:         $\mu_{pred} \leftarrow$ records the center coordinates of the generated new Gaussian.
10:     end if
11:     $B^*_{\mathcal{G}_i} \leftarrow$ ramdomly sample Gaussians from $\mathcal{G}$ to form training batch.
12:     for each batch $B_i$ in $B^*_{\mathcal{G}_i}$ do
13:         $\mathcal{F}(\mu_{\mathcal{G}_i} | \mathcal{G}_i \in B_i) \leftarrow$ train PointNet++ [34] model on $B_i$
14:     end for
15:     $\mathcal{G}_{\text{pred}} \leftarrow$ predict the color using $\mathcal{F}$ using the center of interpolated Gaussian in $\mu_{pred}$
16:     $\mathcal{G}_i \leftarrow add\ \mathcal{G}_{\text{pred}}$
17: end for
18: $\mathcal{G} \leftarrow$ update the overall scene
19: Output: structure optimized $\mathcal{G}$
```

B.3 Mesh generation

We employed the surface extraction method proposed by [15] with an additional normal regularization term during the scene refinement. To compress the Gaussians to align closely with the surface, [15] forces the Gaussians to be flat by setting one of their scaling factors nearly to zero. Consequently,

$\nu^*(p)$ can be further simplified to $\hat{\nu}^*(p) = \alpha_{\mathcal{G}^*} \exp\left(-2s_{\mathcal{G}^*}^{-2}\langle p - \mu_{\mathcal{G}^*}, n_{\mathcal{G}^*}\rangle^2\right)$ where $s_{\mathcal{G}}$ represents the smallest scaling factor and $n_{\mathcal{G}}$ its corresponding normal. Here, $\alpha_{\mathcal{G}^*} = 1$ is set to avoid semi-transparent Gaussians.

Subsequently, a regularization term is used to align the SDF function derived from the density function with its estimation from the scene as:

$$\mathcal{L}_{\text{sdf}} = \frac{1}{|\mathcal{X}_p|} \sum_{x \in \mathcal{X}_p} |sdf(x) - \hat{sdf}(x)| \tag{14}$$

Here, $\mathcal{X}_p$ is the sampled 3D points and $sdf(x) = \pm s_{\mathcal{G}^*}\sqrt{-2\log(\hat{\nu}^*(x))}$ represents the ideal SDF. The estimated SDF $\hat{sdf}(x)$ is determined by subtracting the depth of x from the corresponding rendered depth map. This regularization term encourages closer alignment of the estimated surface with the observed depth.

Moreover, to derive smooth structured surfaces, particularly in typically flat and textureless floor areas, we adopt the Manhattan World hypothesis, which assumes that planar floors are orthogonal to the vertical axis. We enforce this constraint by introducing a normal regularization term for the estimated SDF

$$\mathcal{L}_{\text{normal}} = \frac{1}{|\mathcal{X}_{\text{floor}}|} \sum_{x_{\text{2D}} \in \mathcal{P}_{\text{floor}}} \left| 1 - \hat{n} \cdot \frac{\nabla f(x)}{\|\nabla f(x)\|} \right| \tag{15}$$

In this context, the projected points x_{2D}, which pass through the camera ray and fall within the floor area, are encouraged to align their normal to the ideal vector $\hat{n} = \langle 0, 0, 1 \rangle$. The set $\mathcal{P}_{\text{floor}}$ consists of image pixels identified as the floor using the semantic segmentation described in Section 3.3.2.

Figure 8 visualizes the outcomes of surface extraction with and without the normal regularization loss. It can be observed that incorporating this term enables a smoother reconstruction of the floor. In Figure 9, we compare the extracted mesh from our system to the NeRF-based approaches. Although the mesh generated by our method does not outperform recent NeRF-based methods [18, 36] in terms of quality, it still achieves good geometric accuracy and effectively produces mesh for Gaussian SLAM systems, which was not available in previous Gaussian SLAM methods.

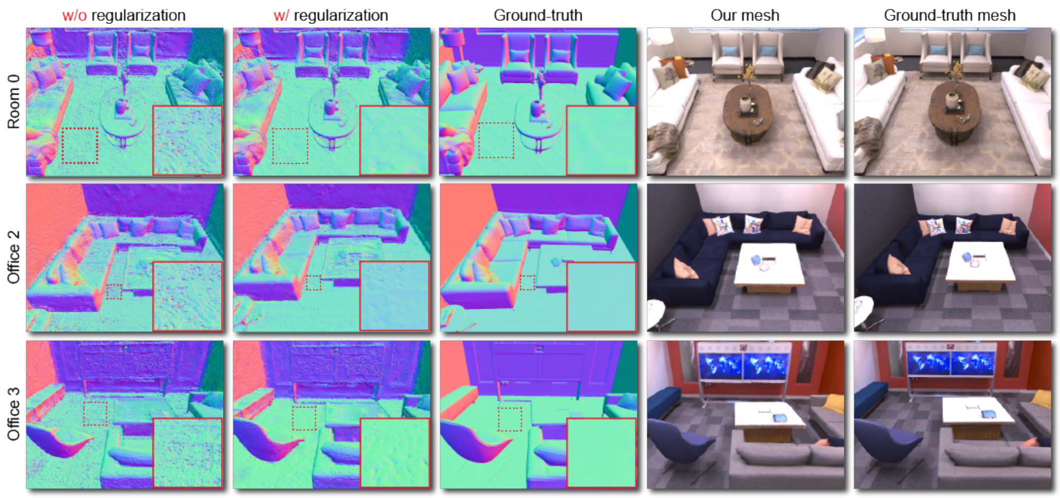

Figure 8: The ablation of normal regularization for surface extraction. We compare the normal map of the extracted mesh, both without and with normal regularization, to the ground-truth mesh. The regularization refines the planar surfaces of the scene, i.e. floors, effectively.

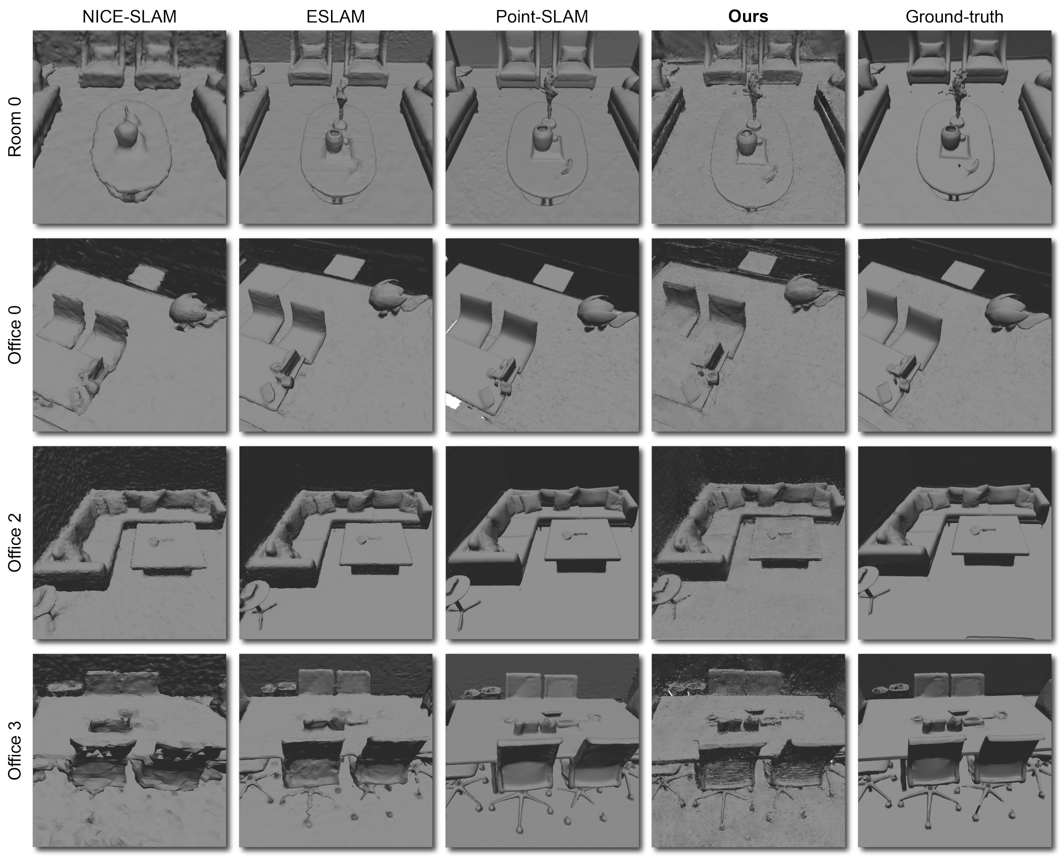

Figure 9: Comparison of the mesh quality of our method with NeRF-based approaches on the Replica dataset [38]. Previous Gaussian-based methods cannot effectively generate mesh, and therefore, cannot be compared here.

C Ablation study

In this section, we provide a comprehensive ablation analysis of each component's effect and the hyperparameters of our model.

C.1 Ablation of hyperparameters

Our tracking system, which advances the foundational PLVS system [14], utilizes feature points and fused line segments for optimizing camera poses and performing bundle adjustments. The upper row of Figure 10 illustrates the effects of varying the number of points and line segments on the ATE loss [cm] and the frame rate of the tracking system. We identified an optimal region where having too few points and lines results in a lack of sufficient anchor features, whereas too many points and lines can lead to notable feature mismatches that reduce tracking accuracy. The tracking FPS reduces as a tradeoff with the increase in the number of features. Compared to tracking solely by point features, including line segment features noticeably decreases the frame rate; however, it still maintains a high rate over 30 FPS and does not become the speed bottleneck of the overall systems. The bottom row of Figure 10 illustrates the effects of default keyframe intervals and downsample ratio on the Gaussian map. During the mapping process, we uniformly downsample the point cloud generated from RGB-D input by a specific ratio to accelerate the system and conserve memory. We observed a marginal PSNR disparity when maintaining a relatively small downsample ratio.

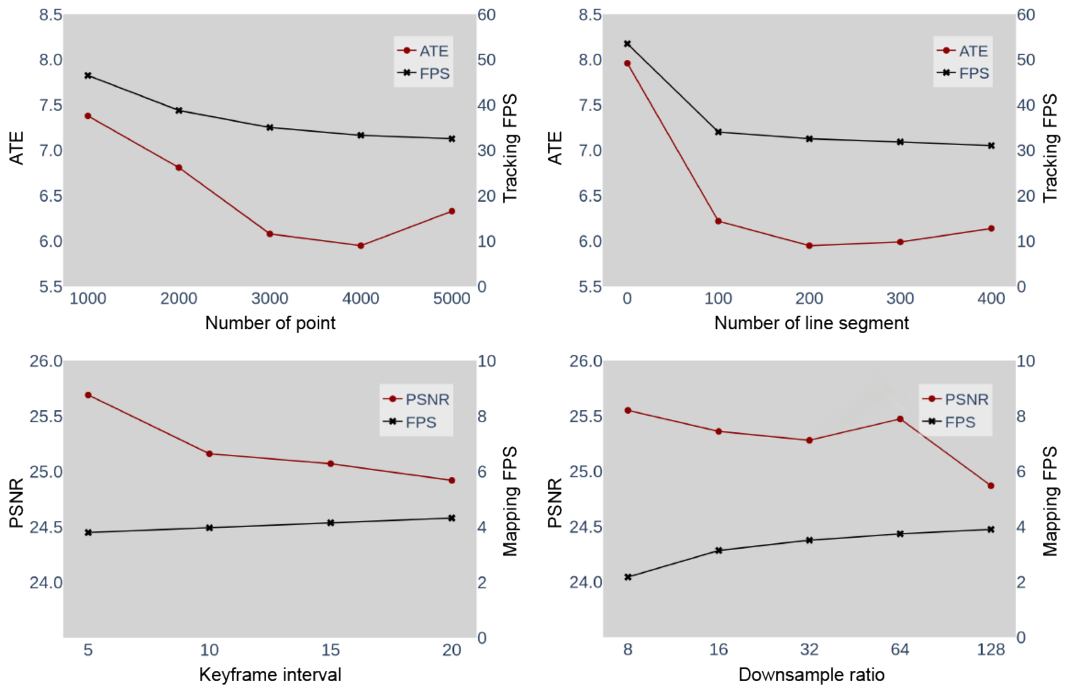

Figure 10: The ablation study examining the impact of the number of points, line segments, keyframe intervals, and downsample ratios in MG-SLAM on *scene 0000_00* from the ScanNet dataset [6].

Table 5: Quantitative comparison of the ablation study on line losses, measuring ATE [cm] and PSNR [dB], conducted on *scene 0000_00* of the ScanNet dataset [6]. The checkmark symbol indicates the employment of the method.

Methods			**Metric**	
line segment	segment fusion	line photo. loss	ATE [cm]	PSNR [dB]
			7.41	20.65
✓			6.58	22.76
✓	✓		**5.95**	24.32
✓	✓	✓	**5.95**	**25.69**

C.2 Ablation of tracking and mapping loss

MG-SLAM employs line segments for robust camera pose optimization and fine-grained map reconstruction. Specifically, fused line segments are utilized in the bundle adjustment of the tracking procedure, and a photometric loss related to line features is integrated into the map optimization. We use *scene 0000_00* from the ScanNet dataset [6] to conduct the ablation study that evaluates the impact of each loss. This scene was chosen because it contains rich line features, which allow for a clear assessment of each component's effectiveness. Table 5 presents the ATE [cm] and PSNR [dB] in relation to the use of each loss. We observed that integrating line features remarkably enhances tracking accuracy, and the fusion of line segments, which facilitates robust edge extraction, further improves the performance. For mapping, the photometric loss provides additional geometric constraints, thereby offering optimal reconstruction quality.

In Figure 11, we present a typical case in *scene 0000_00* to demonstrate the effectiveness of each loss component. This case features a bike that appears multiple times along the camera trajectory. Compared to relying solely on point features, adding fused line segments significantly improves the quality of scene reconstruction. This enhancement mitigates scene drift on the bike and floor by providing more accurate camera pose estimation. Moreover, incorporating the photometric loss of line features in mapping results in more detailed object-level geometry and more precise reconstruction of line-like textures.

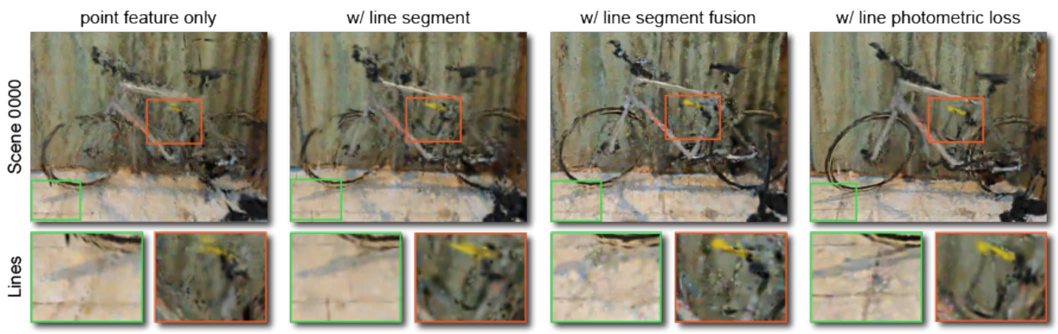

Figure 11: The visualization of ablation analysis evaluating the effect of line segment losses in MG-SLAM. From left to right, the systems are displayed starting completely without line features, then adding line segments, integrating segment fusion, and finally incorporating line photometric loss in mapping. Key differences are highlighted with colored boxes.

D More experimental results

D.1 Training view evaluation on Replica-V1 dataset

Table 6 provides a detailed quantitative analysis of our training view rendering on the Replica-V1 dataset [38]. Our approach achieves state-of-the-art results, outperforming existing Gaussian-based methods by a remarkable margin across three average metrics.

Table 6: Quantitative comparison of our method and the baselines in training view rendering on the Replica-V1 dataset [38]. Our method demonstrates SOTA performances in most cases among three metrics.

Methods	**Metrics**	**Avg.**	Room0	Room1	Room2	Office0	Office1	Office2	Office3	Office4
ESLAM [18]	PSNR↑	29.08	25.32	27.77	29.08	33.71	30.20	28.09	28.77	29.71
	SSIM↑	0.929	0.875	0.902	0.932	0.960	0.923	0.943	0.948	0.945
	LPIPS↓	0.336	0.313	0.298	0.248	0.184	0.228	0.241	0.196	0.204
Point-SLAM [36]	PSNR↑	35.17	32.40	34.08	35.50	38.26	39.16	33.99	33.48	33.49
	SSIM↑	0.975	0.974	0.975	0.980	0.983	**0.986**	0.960	0.960	**0.979**
	LPIPS↓	0.124	0.113	0.116	0.111	0.100	0.118	0.156	0.132	0.142
SplaTAM [19]	PSNR↑	33.98	32.48	33.72	34.96	38.34	39.04	31.90	29.70	31.68
	SSIM↑	0.969	0.975	0.970	**0.982**	0.982	0.982	0.965	0.950	0.946
	LPIPS↓	0.099	0.072	0.096	0.074	0.083	0.093	0.100	0.118	0.155
MonoGS [28]	PSNR↑	35.68	33.78	34.32	36.56	39.14	39.83	34.47	33.25	34.08
	SSIM↑	0.962	0.954	0.957	0.963	0.972	0.976	0.962	0.960	0.950
	LPIPS↓	0.087	0.071	0.086	0.075	**0.074**	0.087	**0.098**	0.098	**0.105**
Ours	PSNR↑	**36.90**	**34.67**	**35.52**	**37.10**	**40.04**	**41.38**	**35.91**	**34.85**	**35.75**
	SSIM↑	**0.973**	**0.976**	**0.978**	0.980	**0.985**	0.984	**0.968**	**0.964**	0.948
	LPIPS↓	**0.086**	**0.070**	**0.084**	**0.070**	0.076	**0.083**	0.101	**0.095**	0.112

D.2 Novel view synthesis on the large scale Replica Apartment dataset

We present additional experimental results on the Replica Apartment dataset [38], synthesized by [21]. This dataset poses an extremely challenging test case for visual SLAM systems due to its large scenes encompassing multiple rooms, each filled with numerous objects featuring complex geometries. Additionally, the camera trajectories include looping paths across multiple rooms, which introduces difficulties in maintaining consistent localization and mapping over extended periods.

We compare our method against SplaTAM [19] and MonoGS [28], which are Gaussian-based methods that have demonstrated impressive performance on the synthetic Replica-V1 dataset [38]. Figure 12, Figure 13, Figure 14, and Figure 15 visualize the mapping results, featuring the captured line segments and the final reconstruction outcomes on the left. On the right, comparisons with baseline methods and ground-truth are presented, with regions of interest highlighted with colored boxes.

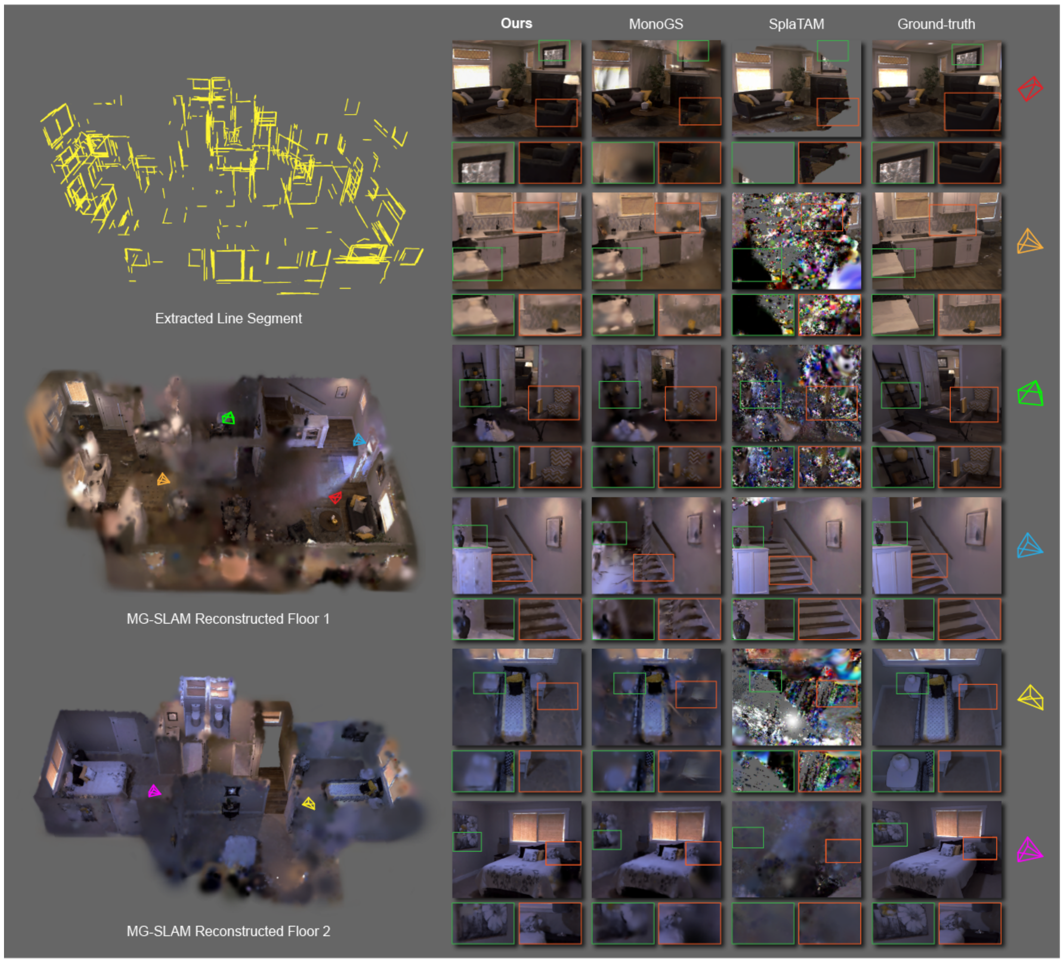

Figure 12: The novel view evaluation of the scene *apartment 0* from the Replica Apartment dataset [38].

Specifically, Figure 12 displays the reconstructed map of an extensive large scene containing two floors, eight rooms, and a staircase. SplaTAM [19] fails to track the camera trajectory, resulting in an entirely erroneously constructed second floor. MonoGS [28] establishes the whole map but struggles to capture the fine details of the environment and poses incorrect geometry of the objects and surfaces. In contrast, our method provides robust scene reconstruction with accurate geometry, which proves its effectiveness in extremely large indoor scenes.

Figure 13 shows that SplaTAM [19] struggles to accurately estimate the camera trajectory through two connected rooms, resulting in significant scene drift in the hallway. MonoGS [28] presents reliable pose estimation but faces challenges in reconstructing dense structural surfaces like walls, leaving gaps in these areas. In Figure 14, we observed that the map reconstructed by SplaTAM [19] includes many noisy Gaussians, likely because its abundant isotropic Gaussian representation cannot be efficiently optimized under limited camera angles. MonoGS [28] also encounters difficulties in reconstructing objects seen in the looping trajectory, such as the round table, due to inaccurate estimations of bundle adjustment across multiple rooms. Moreover, Figure 15 demonstrates a scene with a large camera rotation in the trajectory, under which MonoGS [28] produces incorrect geometry due to inaccurate camera tracking. Similarly, SplaTAM [19] displays noisy Gaussians in certain scenarios. Conversely, our method consistently delivers high-fidelity maps, accurately capturing scene geometry. This effectiveness can be attributed to the precise extraction and integration of line segments, which provides solid feature foundations for robust camera tracking and high-quality reconstruction in large scenes.

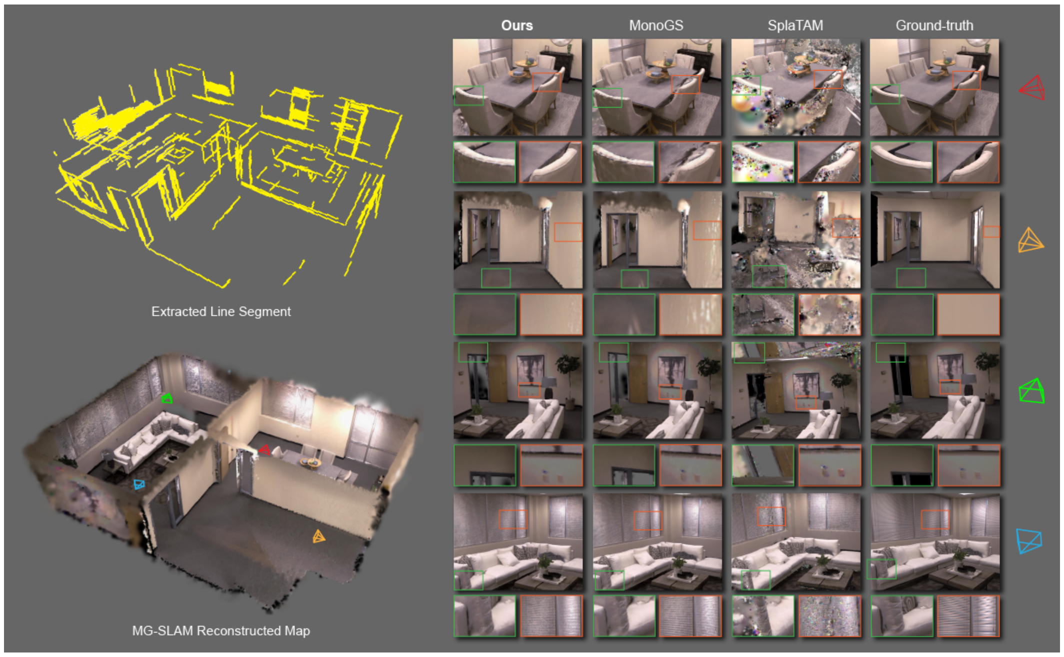

Figure 13: The novel view evaluation of the scene *apartment 1* from the Replica Apartment dataset [38]. The top left shows the line segments extracted in 3D space. The bottom left illustrates the scene as reconstructed by MG-SLAM.

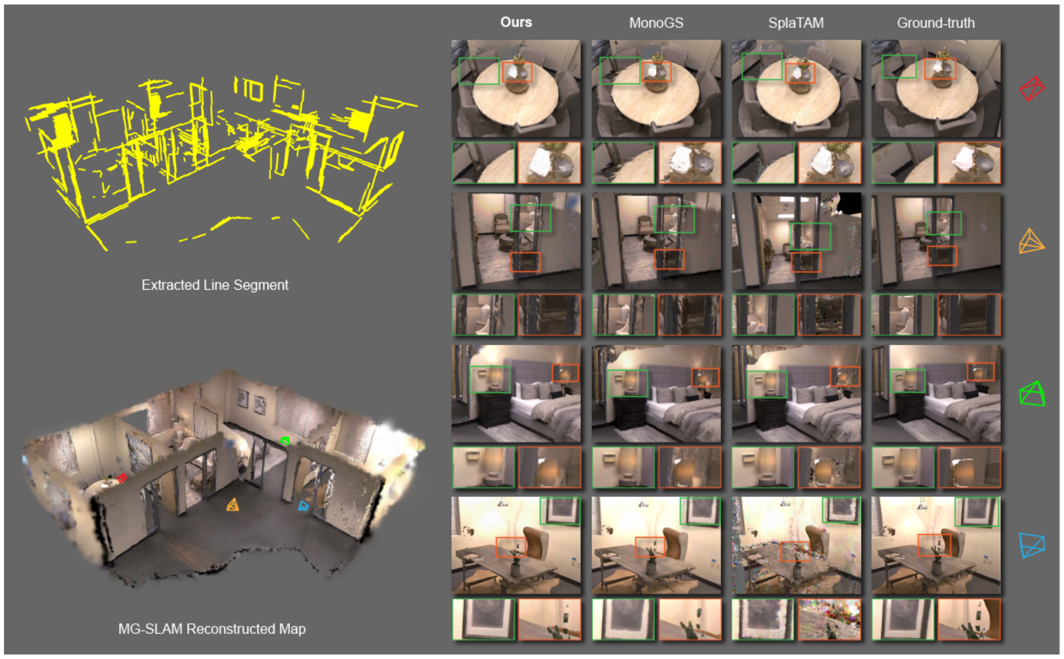

Figure 14: The novel view evaluation of the scene *apartment 2* from the Replica Apartment dataset [38].

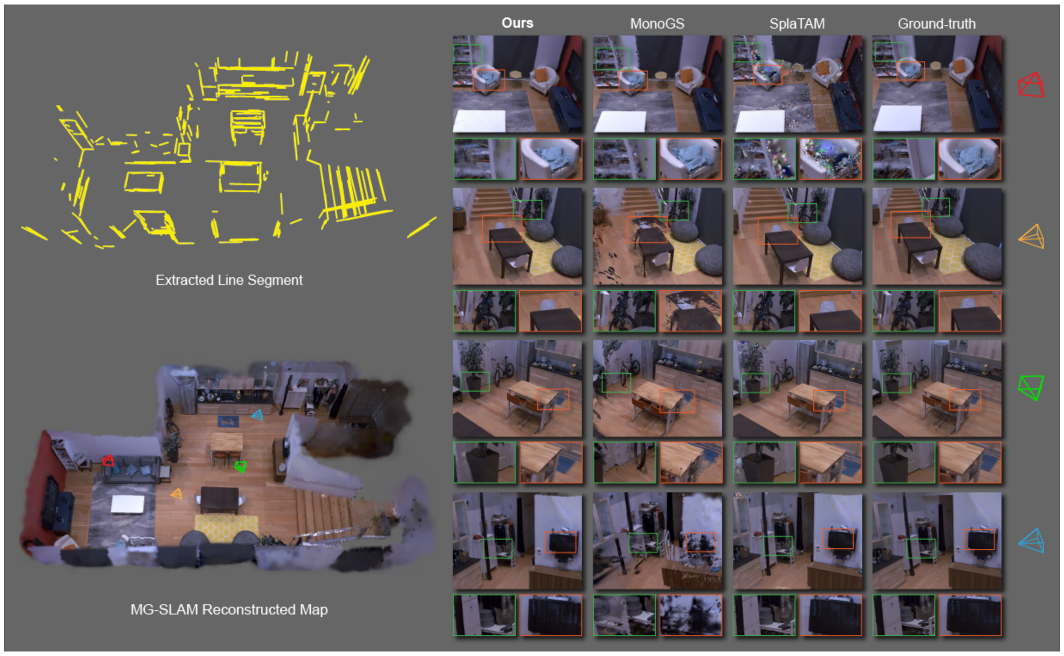

Figure 15: The novel view evaluation of the scene *frl apartment 4* from the Replica Apartment dataset [38].

D.3 Evaluation of localization

In this section, we provide additional tracking evaluation of traditional SLAM systems and our method using the ScanNet dataset [6] and the Replica Apartment dataset [38].

On the ScanNet dataset [6], as shown in Table 7, our system shows optimal tracking performance over the ORB-SLAM systems [31, 2], which are known for their robustness in feature-based tracking and mapping, and PLVS [14], which forms the basis of our tracking systems.

On the Replica Apartment dataset [38], we compare MG-SLAM with both traditional and neural dense SLAM systems. We only evaluate ESLAM [18] as a NeRF baseline here since Point-SLAM [36], discussed in our main text, operates extremely slowly and struggles in large scenes. For Gaussian-based methods, our comparisons include SplaTAM [19] and MonoGS [28]. Table 8 shows that MG-SLAM achieves minimal ATE loss compared to traditional and neural dense baselines. In extremely large scenes, such as *apartment 0*, ORB-SLAM systems [31, 2] lose the camera trajectory on typical textureless areas such as the staircase, and require relocalization. In contrast, MG-SLAM, which utilizes fused line segments as robust anchor features, exhibits SOTA tracking performance. In addition, it is worth recalling that our method offers dense and high-fidelity map reconstruction which is not available in traditional SLAM systems.

Table 7: Additional quantitative comparison of our method and the traditional RGB-D SLAM methods in terms of ATE [cm] on the ScanNet dataset [6]. Our method demonstrates SOTA performances in all presented scenes. Comparison with other neural dense SLAM systems is outlined in the Table 2 of the main text.

Methods	**Avg.**	0000	0059	0106	0169	0207
ORB-SLAM2 [31]	8.70	8.04	6.82	9.22	10.15	9.25
ORB-SLAM3 [2]	8.43	7.62	7.94	9.36	10.0	7.23
PLVS [14]	7.43	6.58	6.80	8.73	8.45	6.62
Ours	**6.77**	**5.95**	**6.41**	**8.07**	**7.29**	**6.14**

Table 8: Quantitative comparison of our method with both traditional and neural dense RGB-D SLAM systems in terms of ATE [cm] on the Replica Apartment dataset [38]. Our method shows SOTA performances, particularly notable improvements in the extensive two-floor *apartment 0* scene. The underlined values indicate that relocalization was necessary due to loss of tracking.

	Methods	**Avg.**	apart.0	apart.1	apart.2	frl_apart.0	frl_apart.4
Trad.	ORB-SLAM2 [31]	3.78	<u>12.83</u>	6.82	4.70	1.85	1.76
	ORB-SLAM3 [2]	3.45	<u>9.45</u>	6.54	4.17	1.63	1.48
	PLVS [14]	2.94	7.06	4.78	4.04	1.45	1.39
Dense	ESLAM [18]	7.08	17.33	13.80	7.65	3.21	3.67
	SplaTAM [19]	6.97	<u>25.14</u>	<u>15.67</u>	6.17	2.74	3.31
	MonoGS [2]	6.55	20.69	9.51	7.57	2.98	6.14
	Ours	2.08	5.03	3.78	2.75	0.92	0.88

D.4 Evaluation of surface completion on the ScanNet dataset

Figure 16 presents the comparison of MG-SLAM with Gaussian-based methods and our approach without surface optimization. The Gaussian-based methods exhibit noticeable gaps, especially in floor areas, unable to capture the full scene structure. In contrast, our method, when combined with a surface optimization strategy, effectively addresses these issues by seamlessly filling in gaps, thus delivering a more coherent and detailed scene reconstruction.

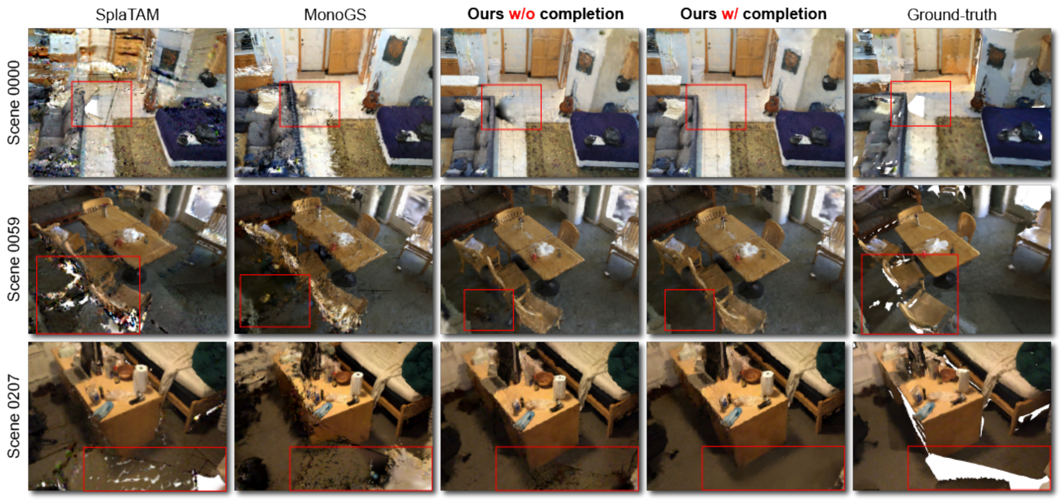

Figure 16: Comparison of novel-view synthesis of our methods and Gaussian-based approaches and without surface optimization. Gaps are accentuated using red boxes.

D.5 Frame rate and parameter size

Processing speed and memory efficiency are pivotal for SLAM systems to effectively operate in real-time and scale in real-world environments. Table 9 presents the comparative performance of our method against neural dense baseline approaches. Notably, NeRF-based Point-SLAM and Gaussian-based SplaTAM demonstrate limitations in achieving appropriate real-time processing capabilities, with both showing considerably low speeds in tracking and mapping alongside higher memory consumption. In contrast, MG-SLAM, implemented in a multi-processing system, exhibits exceptional superiority in tracking performance with the highest frame rate reaching 35 FPS. This improvement in tracking efficiency can be attributed to the incorporation of traditional feature-based optimization, which does not require iterative rendering and computing photometric loss for camera pose optimization, a common practice in neural dense systems, thereby significantly enhancing the tracking frame rate.

MG-SLAM also achieves high-speed and high-fidelity mapping, accelerated by a downsampling strategy that effectively reduces the number of Gaussians without compromising rendering performance.

The reconstructed scenes with fewer Gaussians also ensure moderate memory consumption, making it suitable for deployment in large-scale scenarios.

Table 9: System comparison in terms of tracking, mapping, and rendering FPS, and memory usage between our method and the baselines on the Replica dataset [38]. The values represent the average outcomes across 8 scenes. SOTA performances are highlighted.

Methods	Track. FPS [f/s]↑	Map. FPS [f/s]↑	Render. FPS [f/s]↑	SLAM FPS [f/s]↑	Param. [mb]↓
ESLAM	9.92	2.23	2.5	1.82	73.7
Point-SLAM	0.42	0.06	1.3	0.05	69.6
SplaTAM	0.35	0.22	122	0.14	277.1
MonoGS	2.57	3.85	**769**	1.80	**33.4**
Ours	**35.32**	**3.96**	324	**3.12**	78.5

E Limitation

MG-SLAM has certain limitations. The scene completion strategy, which relies on extracted line segments and the planar surface assumption, is only effective for large structured surfaces. It struggles to effectively interpolate unobserved geometry on complex objects within the scene. However, this problem has long been recognized as extremely challenging in the field of 3D reconstruction and SLAM systems. We believe that our efforts are making meaningful progress and pushing the boundaries further in this area.