CSIAM Trans. Appl. Math.
doi: 10.4208/csiam-am.SO-2025-0070

Vol. **x**, No. x, pp. 1-22
xx 2025

Structure-Aware Indoor RGB-D SLAM via Manhattan-Constrained 2D Gaussian Splatting

Wenwu Guo[1], Xia Yuan[1,2,*], Yanli Liu[2], Xiangyu Wu[1], Wenyi Ge[1], Guanyu Xing[2], Jing Hu[1] and Xi Wu[1]

[1] *School of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China.*
[2] *School of Computer Science, Sichuan University, Chengdu 610065, China.*

Received 1 July 2025; Accepted 8 August 2025

Abstract. Accurate and layout-consistent reconstruction remains a key challenge in indoor simultaneous localization and mapping (SLAM) due to the prevalence of planar and axis-aligned structures. Traditional visual and RGB-D SLAM methods often suffer from incomplete geometry and weak structural reasoning, while NeRF-based SLAM improves fidelity but is computationally expensive and unsuitable for real-time use. 3D Gaussian splatting offers improved efficiency but lacks structural priors, often resulting in distortions in structured scenes. To address these issues, we propose a structure-aware SLAM framework based on 2D Gaussian splatting, which provides efficient, view-consistent mapping. We introduce a lightweight regularization scheme under the Manhattan-world assumption to align Gaussian orientations and positions with dominant axes, improving layout consistency and geometric fidelity. Extensive experiments on Replica and TUM-RGBD datasets demonstrate that our method consistently outperforms existing SLAM baselines in terms of geometric accuracy and edge preservation across multiple indoor scenes.

AMS subject classifications: 68T45, 93C85

Key words: SLAM, 2DGS, Manhattan-regularized.

1 Introduction

Indoor simultaneous localization and mapping is a core technology for applications such as autonomous navigation, augmented reality (AR), and indoor robotics. It supports real-time 3D perception and spatial understanding, forming the foundation for tasks like object interaction and human-environment collaboration. Classic visual SLAM systems, such as ORB-SLAM3 [3], rely on sparse keypoint-based maps and perform well in

*Corresponding author. *Email address:* xyuan623@163.com (X. Yuan)

feature-rich outdoor scenarios. However, in indoor environments characterized by repetitive layouts and textureless surfaces, these methods often suffer from tracking failure, poor reconstruction completeness, and limited structural reasoning.

Depth-based SLAM methods attempt to mitigate the limitations of sparse visual systems by directly leveraging geometric data from RGB-D sensors. Representative approaches such as KinectFusion [20] and ElasticFusion [31] support real-time dense reconstruction through volumetric or surfed fusion and improve robustness in texture-sparse indoor environments. However, these methods often suffer from surface blending artifacts caused by viewpoint inconsistency, and their underlying representations struggle to preserve structural boundaries and geometric detail, particularly in large planar scenes. Moreover, the lack of global structural priors makes it difficult to model spatial regularity in structured or repetitive indoor environments.

Neural radiance fields (NeRF) have been introduced into SLAM systems to jointly optimize camera poses and continuous volumetric scene representations, enabling photorealistic and dense reconstruction from sparse inputs [16, 21]. While offering high-quality rendering, NeRF-based SLAM methods suffer from expensive volumetric sampling and MLP-based inference, making them computationally intensive and unsuitable for real-time applications. In addition, their implicit scene representation prevents efficient map updates and hinders integration with standard pose graph optimization. More fundamentally, these methods lack structural priors and struggle to capture the spatial regularities inherent to indoor environments, such as orthogonal walls, planar boundaries, and repetitive room layouts – leading to degraded geometric consistency and layout coherence.

Recently, 3D Gaussian splatting (3DGS) has emerged as an efficient and expressive alternative to NeRF-based scene representations for real-time SLAM. By modeling volumetric geometry with view-dependent anisotropic Gaussian primitives, 3DGS significantly reduces computational overhead while maintaining high rendering fidelity, and has been adopted in dense SLAM systems such as SplaTAM [13] and MonoGS [18]. Despite their efficiency, existing 3DGS-based methods remain limited in structured indoor environments. Unconstrained primitive placement often leads to geometric distortion, especially in scenes with dominant planar and orthogonal layouts. Moreover, the absence of structure-aware regularization and high-level spatial priors results in multi-view inconsistencies and degraded layout fidelity.

2D Gaussian splatting (2DGS) [10] replaces volumetric Gaussian spheres with 2D Gaussian disks, yielding improved rendering efficiency and view-consistent modeling. Its compact, differentiable formulation makes it particularly effective for representing thin and planar structures in indoor environments, while supporting real-time optimization – an essential property for SLAM. These characteristics make 2DGS a promising scene representation for efficient indoor mapping. Despite its advantages, combining 2DGS with Manhattan-world priors for SLAM remains unexplored. This integration presents several challenges. First, 2DGS lacks explicit structural primitives, making it difficult to directly impose axis-aligned constraints. Second, its flexibility in modeling

fine-grained geometry can conflict with rigid structural priors, potentially causing instability or oversmoothing. Third, ensuring layout consistency across frames in incremental SLAM requires real-time estimation and propagation of dominant directions under uncertainty.

To address these challenges, specifically the lack of structural embedding in 2DGS, the conflict between geometric flexibility and layout regularity, and the difficulty of enforcing global alignment during real-time updates, we propose a structure-aware SLAM framework that jointly leverages 2D Gaussian splatting and Manhattan-world regularization. Our main contributions are as follows:

- We propose the first SLAM framework that adopts two-dimensional Gaussian splatting as the core map representation. Compared to volumetric models, 2DGS improves rendering efficiency and aligns better with image-plane observations, enabling lightweight optimization and real-time performance while maintaining reconstruction quality.
- We incorporate structural regularization based on the Manhattan-world assumption into the 2DGS formulation. By designing a loss that encourages Gaussian orientations and placements to follow axis-aligned directions, the method effectively introduces layout priors to enhance structural consistency and geometric fidelity in indoor environments.
- Extensive experiments are conducted on public datasets including Replica and TUM-RGBD. The proposed method demonstrates superior performance in both geometric accuracy and edge detail preservation compared to existing SLAM baselines, validating the effectiveness of integrating 2D Gaussian splatting with Manhattan-world structural priors.

2 Related work

In this section, we briefly review traditional RGB-D SLAM methods commonly used for indoor 3D reconstruction, and introduce recent developments based on neural radiance fields, Gaussian splatting (GS), and learning-based SLAM approaches. These methods each demonstrate strengths in geometry modeling, pose estimation, and rendering quality.

2.1 Classic RGBD SLAM

Classic RGB-D SLAM methods leverage depth cameras to simultaneously capture color and depth data, which greatly simplifies 3D reconstruction and scale estimation. Early systems like KinectFusion enabled real-time dense mapping but were limited to small-scale, static environments. ElasticFusion improved adaptability through global consistency optimization and non-rigid model alignment. BundleFusion [6] further advanced

the field by enabling large-scale, real-time dense reconstruction with global pose optimization across entire scenes. Systems like RGB-D SLAM and ORB-SLAM3 [3] combined sparse feature tracking with graph-based optimization, striking a solid balance between accuracy and efficiency. These methods perform well in structured indoor settings and laid the groundwork for integrating deep learning and multimodal information in more recent research.

2.2 Learning-based SLAM

Learning-based SLAM systems have achieved remarkable progress by integrating deep neural networks into geometry representation, camera tracking, and scene reconstruction. Unlike classical approaches that rely on handcrafted features and geometric heuristics, these methods leverage learned priors to improve robustness and generalization. Code-SLAM [2] pioneered the idea of encoding dense geometry into a compact latent space, enabling joint optimization with camera poses. DROID-SLAM [29] further introduced a differentiable pose graph to refine dense frame correspondences using deep features, setting a new benchmark in learned SLAM. SLAM3R [17] built on this direction by integrating transformer-based sparse matching and differentiable bundle adjustment in an end-to-end pipeline, emphasizing scalability and efficiency. VGGT [30] leveraged voxel-based representations and vision transformers for robust joint tracking and mapping in large-scale environments. Most recently, FLARE [37] proposed a feed-forward framework that jointly estimates geometry, appearance, and camera poses from sparse, uncalibrated inputs, without iterative refinement. Despite their strong performance, these methods often require extensive training on large-scale datasets and may suffer from generalization issues when transferred to real-world environments. Moreover, their reliance on learned features and black-box models can limit interpretability and hinder explicit modeling of structural geometry.

2.3 NeRF-based SLAM

Classical NeRF-based SLAM systems rely exclusively on MLP, they encode the environment within the weights of the neural network. Predicting pixel values is done by decoding the MLP and integrating the sampled rays into the volume through body mapping. As the first NeRF-based SLAM system, iMAP [27] ensures near-frame-rate camera tracking by jointly optimizing photometric and geometric losses. Limited by MLP capacity, subsequent work combines implicit MLP with structural features to address catastrophic forgetting. For example, NICE-SLAM [39] employs a hierarchical strategy that integrates multiple levels of local data.Vox-Fusion uses octree to manage voxels. ESLAM [12], Co-SLAM [34], GO-SLAM [38], and Point-SLAM [22] introduce innovative approaches such as multiscale feature planes, combinatorial smoothing, detail encoding, real-time global optimization, and dynamic neural point cloud representation, and other innovative methods to enhance scene reconstruction. OrbeezSLAM [4], NeRF-SLAM [21], and

NeRF-VO [19] improve the tracking frequency and bit position estimation accuracy of NeRF-based SLAM ranging methods.

2.4 3DGS-based SLAM

Methods based on three-dimensional Gaussian splatting face challenges in indoor environments due to the abundance of planar structures and the need for detailed surface representation. To address these issues, several improvements have been proposed to enhance system performance. SplaTAM [13] uses a simplified 3D Gaussian model for accurate camera pose estimation and scene refinement. GS-SLAM [13] uses 3D Gaussian, opacity and spherical harmonics (SH) for scene modeling and introduces an adaptive Gaussian management mechanism with a robust camera tracking strategy. GS-SLAM uses 3D Gaussian, opacity, and spherical harmonics for scene modeling, and introduces an adaptive Gaussian management mechanism with a robust camera tracking strategy. MonoGS is the first monocular camera-based system to implement 3DGS-SLAM. GS-ICP [9] combines the algorithms of G-ICP [24] to achieve camera tracking using 3DGS explicit features. Photo-SLAM [11] is based on the ORB algorithm to achieve accurate camera position estimation and scene refinement. Photo-SLAM constructs a hybrid Gaussian map model based on the highly accurate camera poses provided by ORB-SLAM3 [3].

In addition, some approaches fuse LiDAR sensors with 3DGS to enhance system robustness.Gaussian-LIC [15] constructs a tightly coupled LiDAR-inertial-camera SLAM system to achieve real-time performance. MM-Gaussian [32] proposes a relocation module capable of trajectory repair after system tracking failure. While MM3DGS-SLAM [28] optimizes the camera attitude estimation in a visual inertia framework to further improve the system stability and accuracy.

3 Method

Our goal is to improve scene localization accuracy and edge reconstruction quality for indoor environments. To achieve this, we design a structured reconstruction pipeline whose overall architecture is illustrated in Fig. 1. Our method integrates two key components. First, we incorporate the 2D Gaussian splatting representation [10] to enhance geometric fidelity and surface sharpness, especially along object boundaries and edges (Section 3.2.2). Second, we introduce a Manhattan [5] constraint to guide the structural alignment of the reconstructed scene by enforcing dominant orthogonal directions, improving localization robustness and structural consistency (Section 3.3).

3.1 Preliminary

To effectively represent thin structures and improve surface reconstruction accuracy in complex scenes, our method adopts 2D Gaussian splatting [10] as the core scene representation. In contrast to conventional volumetric 3D Gaussians, 2DGS models each point

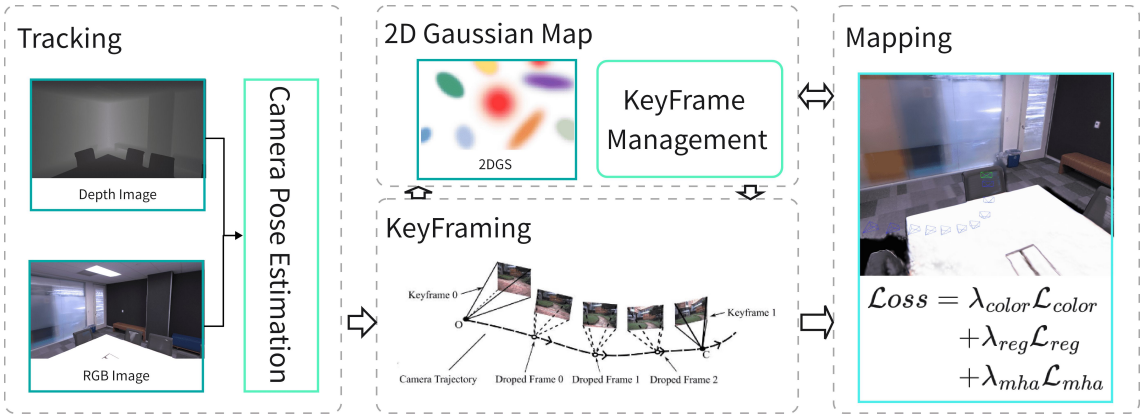


Figure 1: Our SLAM system uses a 2D Gaussian as the unique representation to unify all components of SLAM, including tracking, mapping, and keyframe management.

as a Gaussian distribution defined on a local 2D tangent plane embedded in 3D space. This formulation enables closed-form ray-plane intersections and rigid, view-consistent transformations under camera motion, thereby mitigating the view-dependent artifacts and covariance distortions observed in 3DGS. The improved geometric consistency and analytic tractability make 2DGS particularly suitable for SLAM systems, where accurate multi-view fusion, stable optimization, and surface-level regularization are essential for robust performance.

In 2DGS, each Gaussian element is parameterized by its center point $\mathbf{P}_k$, two orthogonal tangent vectors $\mathbf{T}_u$ and $\mathbf{T}_v$, and scale parameters (S_u, S_v). The plane normal vector $\mathbf{T}_w$ is given by the cross product $\mathbf{T}_w = \mathbf{T}_u \times \mathbf{T}_v$, forming the rotation matrix $\mathbf{R} = [\mathbf{T}_u, \mathbf{T}_v, \mathbf{T}_w]$. Combined with the scale matrix $\mathbf{S}$ (with the third scale set to zero), the local affine transformation matrix $\mathbf{H}$ is defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{RS} & \mathbf{P}_k \\ \mathbf{0} & 1 \end{bmatrix}. \tag{3.1}$$

Any point (u,v) in the local 2D Gaussian coordinate system maps to the 3D space as

$$\mathbf{P}(u,v) = \mathbf{H}(u,v,0,1)^\top, \tag{3.2}$$

and its corresponding 2D Gaussian density is expressed as

$$G(u,v) = \exp\left(-\frac{u^2+v^2}{2}\right). \tag{3.3}$$

Each Gaussian also encodes an opacity α and a viewpoint-dependent color $\mathbf{c}$ modeled via spherical harmonics. To avoid numerical instability during ray-to-Gaussian coordinate transformation, pixel ray planes h_x, h_y are transformed into the local coordinate system, with analytic solutions for (u,v) coordinates

$$u(\mathbf{x}) = \frac{h_{2u}h_{4v} - h_{4u}h_{2v}}{h_{1u}h_{2v} - h_{2u}h_{1v}}, \quad v(\mathbf{x}) = \frac{h_{4u}h_{1v} - h_{1u}h_{4v}}{h_{1u}h_{2v} - h_{2u}h_{1v}}. \tag{3.4}$$

The Gaussian density and depth information at pixel $\mathbf{x}$ are then computed accordingly. To improve evolution stability, especially when approximating Gaussian surfaces as line segments, a screen-space low-pass filtering is applied

$$\widehat{G}(\mathbf{x}) = \max\big(G\big(u(\mathbf{x}),v(\mathbf{x})\big), G(\mathbf{x}-\mathbf{c}\sigma)\big). \tag{3.5}$$

Finally, all Gaussian elements are composited for transparency using volume rendering

$$c(\mathbf{x}) = \sum_i c_i \alpha_i \widehat{G}_i\big(u(\mathbf{x})\big) \prod_{j=1}^{i-1} \big(1-\alpha_j \widehat{G}_j\big(u(\mathbf{x})\big)\big). \tag{3.6}$$

Beyond this geometric representation, 2DGS introduces two geometric regularization terms to improve reconstruction quality [1,7,36]:

- Depth distortion loss encourages depth compactness by minimizing pairwise depth distances between Gaussian intersections along each ray

$$L_d = \sum_{i,j} \omega_i \omega_j |z_i - z_j|, \tag{3.7}$$

  where ω_i is the volumetric blending weight and z_i is the depth of the i-th intersection.

- Normal consistency loss enforces alignment of each Gaussian's normal n_i with the surface normal N estimated from the depth map gradient

$$L_n = \sum_i \omega_i \big(1 - n_i^\top N\big), \tag{3.8}$$

  where N is computed by finite differences at the median ray point ($\alpha = 0.5$).

The total loss function combines photometric and geometric terms

$$L = L_c + \alpha L_d + \beta L_n, \tag{3.9}$$

where α and β are hyperparameters selected based on scene characteristics [14].

3.2 SLAM

Our SLAM system consists of two core components: tracking and mapping. Tracking focuses on estimating the camera pose for incoming RGB-D frames, while mapping optimizes the scene representation based on accumulated observations. Together, they enable real-time, robust localization and dense reconstruction.

3.2.1 Tracking

During tracking, only the current camera pose T_{CW} is optimized, while the 2D Gaussian map representation G remains fixed. Given an RGB-D frame, the photometric residual is

defined as

$$E_{\text{pho}} = I(G, T_{CW}) - \bar{I}, \tag{3.10}$$

where $I(G, T_{CW})$ is the RGB image rendered from the 2D Gaussian splatting representation under pose T_{CW}, and $\bar{I}$ is the observed RGB image. The geometric residual includes both depth and normal terms. The depth residual is

$$E_{\text{depth}} = D(G, T_{CW}) - \bar{D}, \tag{3.11}$$

where $D(G, T_{CW})$ denotes the depth map rasterized from the Gaussian representation, and $\bar{D}$ is the observed depth. Additionally, to enforce surface consistency, we include the normal residual

$$E_{\text{normal}} = N(G, T_{CW}) - \bar{N}, \tag{3.12}$$

where $N(G, T_{CW})$ and $\bar{N}$ are the rendered and observed surface normals, respectively. The combined residual minimized during pose optimization is

$$E = \lambda_{\text{pho}} E_{\text{pho}} + \lambda_{\text{depth}} E_{\text{depth}} + \lambda_{\text{normal}} E_{\text{normal}}, \tag{3.13}$$

where $\lambda_{\text{pho}}, \lambda_{\text{depth}}, \lambda_{\text{normal}}$ are weighting hyperparameters. Depth values are rasterized per pixel via alpha blending along the camera ray

$$D_p = \sum_{i \in \mathcal{N}} z_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{3.14}$$

where z_i is the depth of the i-th Gaussian along the ray, and α_i is its opacity. Analytical Jacobians for the camera pose parameters are derived to enable efficient gradient-based optimization of the residual E.

3.2.2 Gaussian-based map representation and update

Map representation via 2D Gaussian splatting. We adopt the 2D Gaussian splatting technique to represent and render radiance fields within a scene. Unlike conventional point-based or volumetric rendering approaches, 2DGS models each scene point as a 2D Gaussian splat, capturing its spatial location and appearance attributes. Each Gaussian is defined by a center (x, y), standard deviations (σ_x, σ_y), and a weight w_i representing its radiance or color intensity. By projecting and accumulating these Gaussians on the image plane, the geometry and photometric characteristics of the scene can be effectively reconstructed. During rendering, the contribution of each Gaussian is aggregated to form the final image $I(v)$ from a given viewpoint v, using the following formulation:

$$I(v) = \sum_i w_i \cdot G(x_i, y_i, \sigma_{x_i}, \sigma_{y_i}). \tag{3.15}$$

Here, $G(x_i, y_i, \sigma_{x_i}, \sigma_{y_i})$ denotes the i-th 2D Gaussian function, and w_i is its associated weight.

Mapping update. It is performed independently from tracking and focuses on optimizing the geometric and color representations of the scene. In each frame, we uniformly sample M pixels and compare the rendered outputs against the sensor observations to jointly optimize the geometric features f^g, color features f^c, and the parameters ξ and θ of the color decoder g and interpolation decoder F, respectively. The loss function combines multiple components to enforce consistency in color, depth, and surface normals. Additionally, we introduce a Manhattan constraint to encode geometric priors and further improve the accuracy and robustness of optimization. The total loss is defined as

$$L_{\text{loss}} = \lambda_c L_c + \lambda_d L_d + \lambda_n L_n + \lambda_{mha} L_{mha}. \tag{3.16}$$

Here, L_c denotes the L_1 color loss, which penalizes differences between rendered and observed RGB images. L_d is the L_1 depth loss that aligns rendered depth values with sensor depth. L_n represents the L_1 normal loss, encouraging consistent surface orientations. L_{mha} is the Manhattan constraint loss that enforces alignment of the reconstructed geometry with dominant scene axes. The hyperparameters λ_c, λ_d, λ_n, and λ_{mha} balance the contribution of each component in the total loss. Typically, we assign the color loss half the weight of the depth loss to balance photometric and geometric cues. As a structural regularization term, the Manhattan constraint significantly enhances the accuracy of geometric reconstruction. A detailed explanation of the formulation and implementation of the Manhattan constraint is provided in Section 3.3.

3.3 Map optimization based on Manhattan constraints

Existing methods [8,35] tend to optimize scene representations individually, without considering the inherent structural relationships across the entire environment. This can lead to reconstructions that are locally accurate but globally inconsistent, especially in indoor scenes with common regular architectural patterns. To overcome this limitation, we introduce a Manhattan constraint. By leveraging the orthogonality prior present in most indoor layouts, this constraint helps to regularize camera poses and guide the geometric alignment of reconstructed elements.

The overall procedure is illustrated in Fig. 2. Planar regions are first identified using a clustering-based method. For each planar region, we then estimate its normal vector and center point to determine whether a dominant axis can be reliably established. If a dominant axis cannot be identified, the process skips axis-based alignment and directly adjusts the nearby Gaussian elements to lie on their respective original planes. If a dominant axis is available, planes that are approximately parallel to it are realigned to be strictly parallel, enforcing global structural regularity. In contrast, planes that deviate significantly from the dominant direction are left unchanged to preserve local geometric details. Finally, all 2D Gaussian elements associated with planar regions are adjusted (rotated or projected) onto their respective aligned planes, ensuring geometric consistency and enabling better structural coherence across the reconstructed scene.

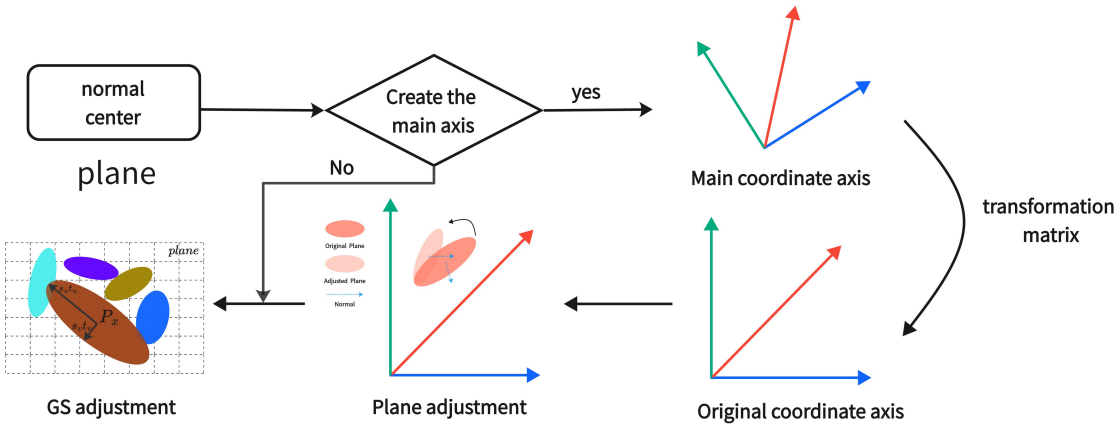


Figure 2: Manhattan constraint estimation pipeline (triggered on new keyframe insertion).

Plane recognition. In this work, we adopt a RANSAC-based plane segmentation method for dense point cloud data, following the efficient shape detection framework proposed by Schnabel *et al.* [23]. In each iteration, a minimal set of points is randomly sampled to fit a plane model, and inliers are identified based on their orthogonal distances to the estimated plane. The process is repeated until the number of remaining unexplained points falls below a predefined threshold.The resulting set of extracted planes is denoted as $\{\Pi_i\}_{i=1}^{N}$, where each plane Π_i is associated with a unit normal vector $\mathbf{n}_i \in \mathbb{R}^3$.

Establishing dominant axes. We collect all normals into the following set:

$$\mathcal{N} = \{\mathbf{n}_1, \mathbf{n}_2, \ldots, \mathbf{n}_N\}, \quad \|\mathbf{n}_i\| = 1. \tag{3.17}$$

Next, we apply mean shift clustering on the unit sphere S^2 to group the surface normals into dominant orientation modes. The goal is to find $K=3$ dominant directions $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ that minimize the following objective:

$$\min_{\{\mathbf{v}_k\}} \sum_{i=1}^{N} \min_{k \in \{1,2,3\}} \left(1 - \left|\mathbf{n}_i^\top \mathbf{v}_k\right|\right). \tag{3.18}$$

This objective is defined on the unit sphere, where the absolute value handles directional ambiguity of surface normals. After clustering, we obtain a set of approximate dominant directions

$$\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3] \in \mathbb{R}^{3\times 3}. \tag{3.19}$$

These vectors are typically close to orthogonal, but not strictly so. To enforce orthogonality (as required by the Manhattan world assumption), we apply singular value decomposition (SVD) to the matrix $\mathbf{V}$

$$\mathbf{V} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^\top \;\Rightarrow\; \mathbf{R} = \mathbf{U}\mathbf{W}^\top, \quad \mathbf{R} \in SO(3). \tag{3.20}$$

Here, **R** is a rotation matrix whose columns form an orthonormal basis representing the estimated Manhattan frame. For each input normal $\mathbf{n}_i$, we compute its smallest angular deviation from the estimated dominant directions (the columns of **R**)

$$\theta_i = \min_{j\in\{1,2,3\}} \arccos\left(\left|\mathbf{n}_i^\top \mathbf{r}_j\right|\right). \tag{3.21}$$

If $\theta_i < \theta_{\text{thresh}}$, then $\mathbf{n}_i$ is considered aligned with one of the dominant directions. Finally, the number of aligned normals is counted to produce a support score S

$$S = \sum_{i=1}^{N} \mathbf{1}(\theta_i < \theta_{\text{thresh}}). \tag{3.22}$$

If this score is low, it indicates that the scene is unlikely to follow a Manhattan structure. If the score is high, the estimated dominant axes can be considered valid and used as geometric priors in subsequent modeling or structural reasoning stages.

Plane adjustment toward dominant axes. To enhance structural consistency under the Manhattan-world assumption, we selectively adjust detected local planes whose normals are already close to one of the dominant scene directions. Let $\{\mathbf{r}_1,\mathbf{r}_2,\mathbf{r}_3\}$ denote the three orthogonal dominant axes. For a local plane with normal $\mathbf{n}_p$, we identify the nearest dominant axis

$$j^* = \arg\min_{j\in\{1,2,3\}} \arccos(|\mathbf{n}_p \cdot \mathbf{r}_j|). \tag{3.23}$$

If the angle is below a predefined threshold ϵ, we update the plane normal to align with the dominant axis

$$\mathbf{n}'_p = \mathbf{r}_{j^*}. \tag{3.24}$$

This adjustment preserves the plane's center $\mathbf{c}_p$, while enforcing axis alignment only for those planes already approximately Manhattan-aligned. Other planes with large angular deviations (e.g. slanted or free-form surfaces) remain unchanged to maintain geometric diversity.

Gaussian primitive projection and association. After adjusting selected planes, we associate nearby 2D Gaussian primitives based on spatial proximity. Each primitive is defined by its center **c** and normal $\mathbf{n}_g$. For each plane, we determine whether a primitive lies close enough using a distance threshold δ

$$\left|\mathbf{n}'_p \cdot (\mathbf{c}_p - \mathbf{c})\right| < \delta. \tag{3.25}$$

For associated primitives, we compute the displacement vector

$$\mathbf{d} = (\mathbf{c}_p - \mathbf{c}) \cdot \mathbf{n}'_p. \tag{3.26}$$

And update their parameters accordingly

$$\mathbf{c}' = \mathbf{c} + \mathbf{d} \cdot \mathbf{n}'_p, \quad \mathbf{n}'_g = \mathbf{n}'_p. \tag{3.27}$$

If a Gaussian lies near multiple planes, we compute its angle to each candidate plane normal

$$\theta_i = \arccos\left(\frac{\left|\mathbf{n}_g \cdot \mathbf{n}_p^{(i)}\right|}{\left\|\mathbf{n}_g\right\|\left\|\mathbf{n}_p^{(i)}\right\|}\right). \tag{3.28}$$

Among those satisfying the distance threshold, the Gaussian is associated with the plane yielding the smallest angle. Unassociated Gaussians – those too far from any plane – remain unchanged.This selective association mechanism ensures that only structured regions (i.e. those lying near detected planes) are adjusted, while primitives in unstructured or non-Manhattan regions are left untouched. Therefore, our method preserves generalization and does not degrade performance in complex or outdoor environments.

Manhattan world geometric constraint. To further enhance structural regularity, we apply the Manhattan world assumption by constraining the Gaussian primitives within each plane to follow orthogonal directional priors. Specifically, we compute the projected component δ_i of each Gaussian primitive along the sub-dominant axis within the local plane, and define the constraint loss as

$$L_{mha} = \sum_i \max(0, \delta_i - \epsilon)^2, \tag{3.29}$$

where ϵ sets the threshold that controls the allowable extension in the secondary direction. This loss penalizes deviations from the dominant orientation and encourages the point distributions to organize along axis-aligned directions.

Furthermore, when classifying primitives near plane intersections, we incorporate angular similarity under the Manhattan prior to assign each Gaussian primitive to the plane that best matches its geometric orientation. This approach reduces structural distortion and strengthens geometric coherence across the entire point cloud representation.

4 Experiments

4.1 Experimental setup

Datasets. We evaluated our method on two representative datasets – Replica [25] and TUM-RGBD [26] – to comprehensively validate its effectiveness in localization accuracy and geometric sharp edges. The Replica dataset consists of synthetic indoor scenes with clean and complete depth maps and minimal inter-frame motion, which is favorable for depth-based camera tracking. In contrast, TUM-RGBD is a real-world dataset with depth maps that suffer from severe sensor noise, motion blur, and missing regions. These challenges make it significantly more difficult to preserve sharp object boundaries and maintain geometric consistency, particularly in cluttered indoor environments.

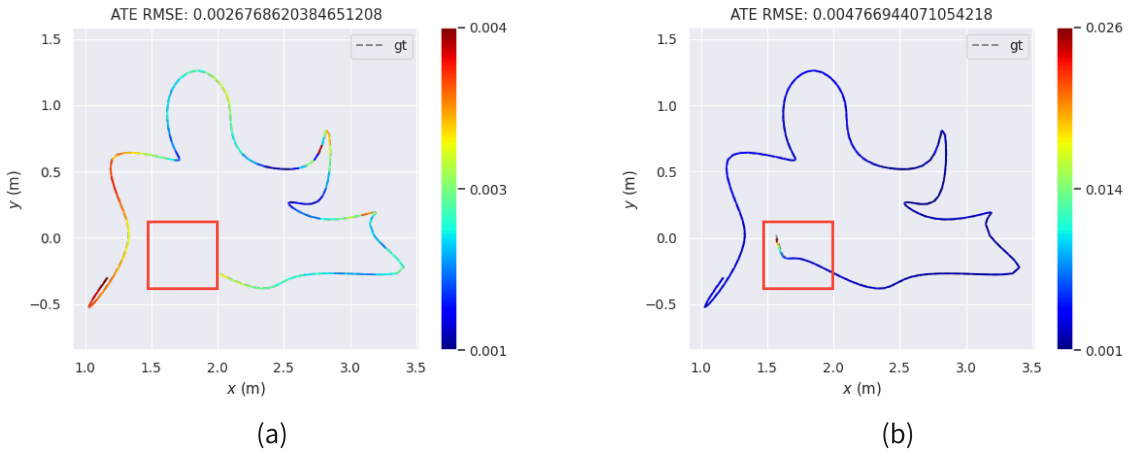
Evaluation metrics. We evaluate our method in terms of both tracking accuracy and reconstruction quality. For tracking, we report the root mean square error (RMSE) of absolute trajectory error (ATE). For reconstruction, we adopt standard photometric metrics including PSNR, SSIM, and LPIPS to assess image fidelity and perceptual similarity.

Baselines. We conducted comparative experiments between our method and representative RGB-D-based SLAM systems, covering MonoGS [18], Point SLAM [22], NICE-SLAM [39], and SplaTAM [13]. To ensure fairness in comparison, we evaluated this method under the same parameter settings and input conditions as each baseline method.

4.2 Quantitative and qualitative comparisons

Camera pose estimation comparisons. We evaluate the trajectory accuracy of the proposed system on the Replica and TUM RGB-D datasets. As shown in Table 1, part (a), our method exhibits slightly reduced accuracy in certain scenes of the Replica dataset. This is mainly due to short segments of rapid camera motion, which introduce motion blur and impair the reliability of plane detection. Consequently, the effectiveness of the Manhattan constraint model is diminished, resulting in accumulated pose errors in these regions. Supporting visualizations in Fig. 3 highlight this effect: before the fast motion segment, the trajectory remains accurate with strong planar consistency, while after the motion, tracking errors increase significantly (from 0.0027m to 0.0047m) due to degraded structural cues.

In contrast, as shown in Table 1, part (b), the TUM RGB-D dataset provides more favorable conditions for our method. The camera trajectories are relatively smooth, and the scenes contain abundant indoor structures that align well with the Manhattan-world



(a) (b)

Figure 3: ATE Evaluation on the Office4 dataset (ATE RMSE ↓ [m]): (a) Trajectory evaluated up to frame 763; (b) Same trajectory extended and evaluated up to frame 864.

Table 1: Online camera pose estimation results (ATE RMSE ↓ [cm]): (a) On the Replica dataset; (b) On the TUM RGB-D dataset. Best results are highlighted as **first,** *second*, and third.

Method	R0	R1	R2	Of0	Of1	Of2	Of3	Of4	Avg.
MonoGS	0.47	0.43	*0.31*	0.70	0.57	0.31	*0.31*	3.2	0.79
SplaTAM	**0.31**	*0.40*	**0.29**	0.47	**0.27**	*0.29*	0.32	**0.55**	**0.36**
Point-SLAM	0.61	0.41	0.37	**0.38**	*0.48*	0.54	0.69	*0.72*	*0.52*
NICE-SLAM	0.97	1.31	1.07	0.88	1.00	1.06	1.10	1.13	1.07
Ours	*0.38*	**0.39**	0.33	*0.41*	0.59	**0.25**	**0.16**	2.01	0.56

(a)

Method	fr1/desk	fr2/xyz	fr3/office	Avg.
MonoGS	*1.59*	1.54	*1.79*	*1.64*
SplaTAM	3.35	**1.24**	5.16	3.25
Point-SLAM	4.34	*1.31*	3.48	3.04
NICE-SLAM	2.7	1.8	3.0	2.5
Ours	**1.53**	1.51	**1.61**	**1.55**

(b)

assumption. These characteristics improve the robustness of plane detection and allow the structural regularization term to effectively take effect. Consequently, our method achieves higher trajectory accuracy on this dataset, demonstrating excellent performance and reliability in real-world indoor environments.

Quantitative rendering comparison. Compared with typical SLAM systems, our method demonstrates consistently superior reconstruction quality across various indoor scenes in both the Replica and TUM-RGBD datasets. As shown in Tables 3 and 2, although our method exhibits slightly lower trajectory estimation accuracy on the Replica dataset, it still delivers outstanding rendering quality. This is primarily attributed to the precise geometric consistency provided by two-dimensional Gaussian splatting, which can partially compensate for the impact of trajectory errors and thus enable high-quality view synthesis.

Qualitative rendering comparison. We qualitatively evaluate reconstruction fidelity on two representative RGB-D datasets. As shown in Fig. 4, our method better preserves fine geometric structures such as wall corners, window frames, and desk edges on the Replica dataset. Compared to MonoGS and SplaTAM, the reconstructed boundaries appear sharper and more consistent, while the baselines suffer from oversmoothing or geometric distortion. To further assess performance in cluttered real-world scenarios, Fig. 5 presents close-range reconstruction results on the TUM-RGBD dataset. Our method demonstrates notably improved geometric consistency and texture completeness, par-

Table 2: Quantitative reconstruction comparison on the TUM RGB-D dataset.

Method	Metric	fr1/desk	fr2/xyz	fr3/office	Avg.
MonoGS	PSNR↑	*21.88*	*24.16*	*20.48*	*22.17*
	SSIM↑	*0.87*	0.88	*0.83*	0.86
	LPIPS↓	0.28	0.22	*0.30*	0.27
SplaTAM	PSNR↑	21.16	23.11	19.92	21.40
	SSIM↑	0.87	*0.90*	0.82	*0.87*
	LPIPS↓	*0.25*	*0.21*	0.30	*0.25*
Point-SLAM	PSNR↑	17.08	19.02	16.99	17.70
	SSIM↑	0.79	0.83	0.77	0.80
	LPIPS↓	0.36	0.29	0.44	0.36
NICE-SLAM	PSNR↑	19.11	22.54	20.63	20.76
	SSIM↑	0.80	0.87	0.85	0.84
	LPIPS↓	0.31	0.23	0.29	0.28
Ours	PSNR↑	**23.65**	**25.14**	**22.31**	**23.70**
	SSIM↑	**0.91**	**0.92**	**0.87**	**0.90**
	LPIPS↓	**0.18**	**0.15**	**0.19**	**0.17**

Table 3: Quantitative reconstruction comparison on the Replica dataset.

Method	Metric	R0	R1	R2	Of0	Of1	Of2	Of3	Of4	Avg.
MonoGS	PSNR↑	*34.83*	*36.43*	*37.49*	*39.95*	*42.09*	*36.24*	*36.70*	*37.06*	*37.60*
	SSIM↑	0.95	0.95	0.96	0.97	0.97	0.96	0.96	0.95	0.96
	LPIPS↓	0.068	0.076	0.075	0.072	0.055	0.078	0.065	0.099	0.074
SplaTAM	PSNR↑	32.86	33.89	35.25	38.26	39.17	31.97	29.70	31.81	34.12
	SSIM↑	*0.98*	0.97	*0.98*	0.98	0.98	0.97	0.95	0.95	0.97
	LPIPS↓	*0.045*	*0.050*	*0.052*	*0.049*	*0.043*	*0.055*	0.060	*0.058*	*0.051*
Point-SLAM	PSNR↑	31.92	32.65	34.82	37.71	37.89	29.55	26.32	28.94	32.73
	SSIM↑	0.97	0.96	0.97	0.97	0.97	0.96	0.94	0.94	0.96
	LPIPS↓	0.056	0.065	0.068	0.057	0.049	0.072	0.076	0.080	0.066
NICE-SLAM	PSNR↑	33.61	35.40	36.84	39.12	40.98	35.23	34.21	35.46	36.36
	SSIM↑	0.97	*0.97*	**0.98**	*0.98*	*0.98*	*0.97*	*0.96*	*0.96*	*0.97*
	LPIPS↓	0.049	0.051	0.053	0.050	0.046	0.059	*0.058*	0.062	0.054
Ours	PSNR↑	**36.72**	**37.91**	**38.58**	**41.12**	**43.27**	**39.08**	**38.15**	**39.84**	**39.59**
	SSIM↑	**0.98**	**0.98**	0.97	**0.98**	**0.98**	**0.98**	**0.97**	**0.98**	**0.98**
	LPIPS↓	**0.032**	**0.034**	**0.035**	**0.030**	**0.028**	**0.033**	**0.036**	**0.034**	**0.033**

ticularly around high-frequency regions such as keyboards, screen edges, and cabinet surfaces. For instance, while SplaTAM shows texture smearing and misalignment on planar areas, our approach accurately preserves structure and fine details, highlighting the effectiveness of Manhattan-constrained 2DGS in complex indoor environments.

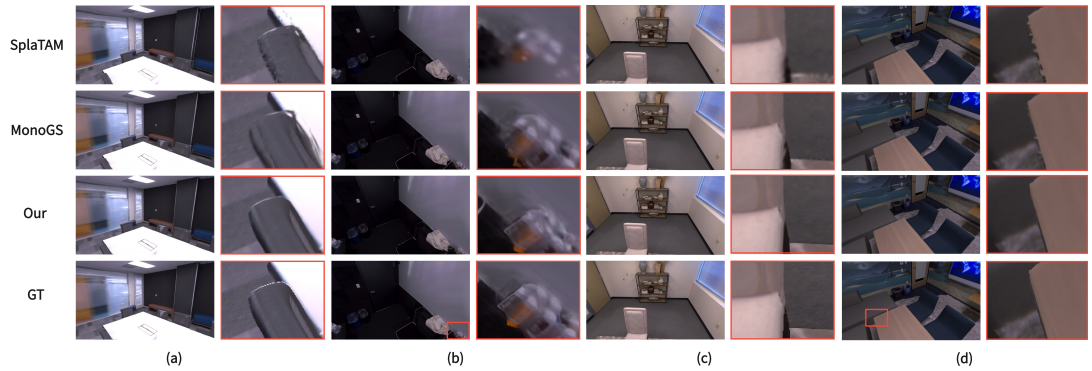


Figure 4: Qualitative reconstruction comparisons on the Replica dataset. Each scene shows a top-down view (left) and a close-up of key structural regions (right), highlighting the improvements in geometric detail and layout consistency achieved by our method compared to MonoGS and SplaTAM.

Figure 5: Qualitative reconstruction comparison on the TUM RGB-D dataset. Our method shows better edge sharpness and texture detail in near-field regions vompared to SplaTAM.

Novel view comparisons. To more effectively evaluate the model's generalization ability to unseen content, we split the TUM RGB-D dataset into two subsets: one is used exclusively for testing novel views and is excluded from training, while the other is used for standard model training. Tables 4 present the quantitative results on the training and novel views. Since these novel views are not part of the training data, the overall view distribution becomes sparser, leading to a drop in training-view PSNR compared

Table 4: Quantitative comparison across different indoor scenes: (a) Training tiews; (b) Novel views.

Methods	Metrics	fr1/desk	fr1/desk2	fr1/room	Avg.
SplaTAM	PSNR ↑	*20.74*	14.59	8.32	14.55
	SSIM ↑	*0.73*	0.60	0.31	0.55
	LPIPS ↓	*0.26*	0.41	0.58	0.42
MonoGS	PSNR ↑	18.88	*17.30*	*16.58*	*17.59*
	SSIM ↑	0.71	*0.64*	*0.58*	*0.64*
	LPIPS ↓	0.30	*0.39*	*0.47*	*0.39*
Ours	PSNR ↑	**22.59**	**20.91**	**19.17**	**21.00**
	SSIM ↑	**0.78**	**0.71**	**0.66**	**0.72**
	LPIPS ↓	**0.24**	**0.33**	**0.38**	**0.32**

(a)

Methods	Metrics	fr1/desk	fr1/desk2	fr1/room	Avg.
SplaTAM	PSNR ↑	*13.20*	8.57	6.46	9.41
	SSIM ↑	*0.46*	0.27	0.25	0.33
	LPIPS ↓	*0.44*	*0.57*	*0.63*	*0.55*
MonoGS	PSNR ↑	12.36	*10.23*	*10.97*	*11.19*
	SSIM ↑	0.45	*0.39*	*0.39*	*0.41*
	LPIPS ↓	0.48	0.60	0.65	0.58
Ours	PSNR ↑	**18.70**	**10.79**	**11.61**	**13.70**
	SSIM ↑	**0.76**	**0.41**	**0.42**	**0.53**
	LPIPS ↓	**0.29**	**0.58**	**0.60**	**0.49**

(b)

to previous experiments – an expected result under this protocol. Nevertheless, under this more challenging setting, our method consistently outperforms existing baselines, achieving higher PSNR and SSIM and lower LPIPS for novel view synthesis, demonstrating strong generalization and visual consistency even with sparse training input.

In addition to quantitative evaluation, we conducted qualitative analysis on the RGB-D TUM dataset. As shown in Fig. 6, our method demonstrates visually superior results across multiple scenes, for both training and novel views.For each sequence, we selected nearby training and novel viewpoints for comparison. The visual results show that SplaTAM and MonoGS suffer from noticeable degradation in novel views, such as structural distortions and color inconsistencies. In contrast, our method maintains stable and high-quality rendering in both view types, further confirming its strong generalization ability and robustness.

4.3 Manhattan constraint ablation study

Table 5 presents the quantitative impact of different Manhattan constraint coefficients on reconstruction accuracy. To comprehensively evaluate the effect of this regularization term, we conducted an ablation study on a complex indoor scene with rich geometric structures. As shown in Table 5 , setting $\lambda_{\text{mha}} = 0$ disables the constraint entirely, allow-

Table 5: Ablation study on the effect of the Manhattan regularization weight λ_{mha} on reconstruction quality.

Control groups			Metrics		
#	Manhattan constraint	λ_{mha} (Weight)	PSNR↑	SSIM↑	LPIPS↓
(1)	w/o	$\lambda_{\text{mha}}=0$	35.17	0.95	0.093
(2)	w/	$\lambda_{\text{mha}}=2$	**37.37**	**0.97**	**0.40**
(3)	w/	$\lambda_{\text{mha}}=4$	34.35	0.94	0.082

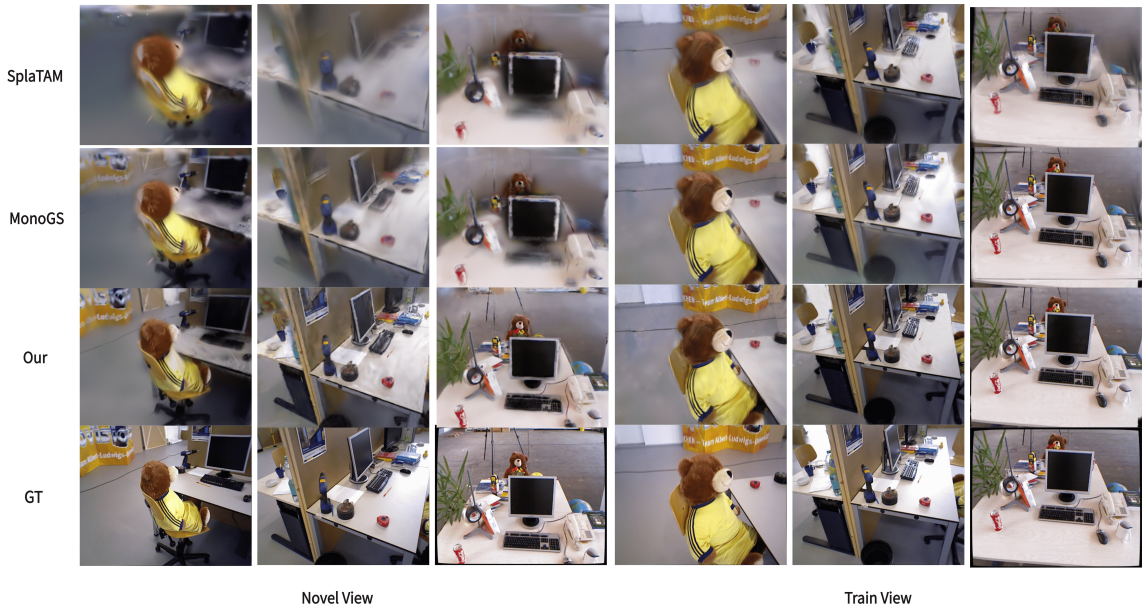


Figure 6: We show rendering results on the TUM RGB-D dataset for both novel and training views to qualitatively evaluate our method.

ing the system to operate as a pure 2DGS-based SLAM pipeline. Conversely, an excessively strong constraint ($\lambda_{\text{mha}}=4$) leads to degraded performance, likely due to over-regularization. In contrast, a moderate setting ($\lambda_{\text{mha}}=2$) yields the best performance metrics.

Along with the quantitative analysis, Fig. 7 illustrates the corresponding qualitative rendering results. Without the constraint, reconstructed surfaces exhibit slight warping, especially at vertical junctions between desk panels where the structure is less clear. A strong constraint strictly enforces vertical alignment but causes over-smoothing on critical boundary details such as the striped board and panel edges. The moderate weight ($\lambda_{\text{mha}}=2$) not only preserves sharp vertical edges but also maintains the natural variation of object shapes, achieving the best balance between structural regularity and detail preservation.

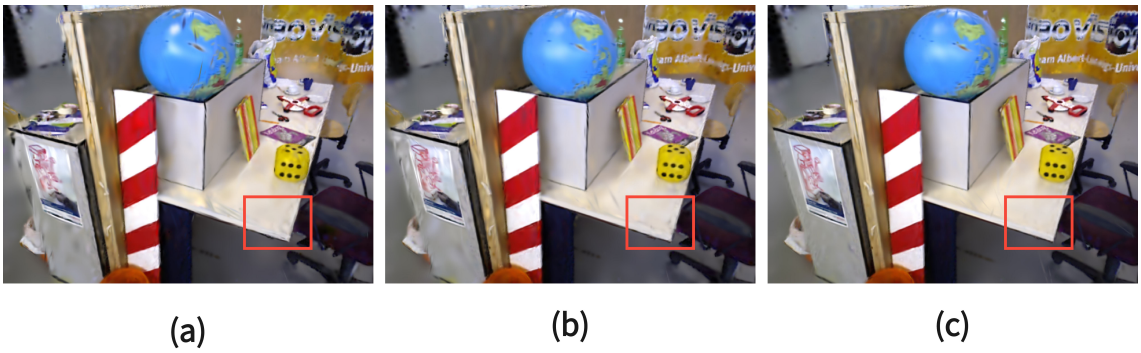


Figure 7: Rendering results under different Manhattan constraint settings: (a) No constraint, (b) $\lambda_{\text{mha}}=4$, and (c) $\lambda_{\text{mha}}=2$. Too weak or too strong constraints lead to suboptimal results, while a moderate weight ($\lambda_{\text{mha}}=2$) achieves the best balance between structural regularity and detail preservation.

4.4 Runtime comparison

We compare the runtime performance of our method against prior approaches on the Replica dataset (scene R0), using an NVIDIA RTX 3090 GPU. Unlike existing methods such as NICE-SLAM and Point-SLAM, which rely on sampling a small subset of pixels (200 for tracking and 1000 for mapping), our approach renders the entire. 1200×980 image (approximately 1.2 million pixels) per iteration to compute losses for both tracking and mapping. Despite this significantly higher pixel count – three orders of magnitude more – our method achieves comparable or superior efficiency, owing to the highly optimized rasterization of our 2D Gaussian representation. Concretely, our method requires only 6.52ms and 3.72ms per iteration for tracking and mapping, respectively. This results in a per-frame runtime of 0.65s for tracking and 0.55s for mapping, leading to a real-time performance of 0.91 FPS. In contrast, Point-SLAM and NICE-SLAM run at only 0.20 FPS and 0.12 FPS, respectively, while even MonoGS – despite processing far fewer pixels – achieves only 0.81 FPS. These results highlight the efficiency of our approach.

5 Conclusion

In this work, we presented a novel SLAM system based on 2D Gaussian splatting, specifically tailored for high-precision reconstruction in complex indoor environments. By leveraging the inherent stability of 2DGS in image-plane modeling and integrating structural constraints under the Manhattan world assumption, our approach effectively enhances both localization accuracy and geometric fidelity, particularly around planar surfaces and edges. Extensive experiments demonstrate that our method outperforms existing solutions in terms of geometric consistency and visual sharpness, establishing a new benchmark for indoor SLAM.

Limitations. While the system shows strong performance in structured environments, it also exhibits certain limitations in highly unstructured or non-Manhattan scenes. In

such cases, the predefined structural priors may no longer hold, leading to reduced pose accuracy or oversimplified geometry. Additionally, as our framework currently lacks dynamic scene modeling and fine-grained failure detection, unexpected sensor noise or ambiguous textures may affect stability. Future work will focus on addressing these limitations by incorporating adaptive structural reasoning, semantic priors, and robust real-time optimization strategies, with the aim of improving scalability, generalization, and fault tolerance across a wider range of real-world scenarios.

6 Acknowledgments

This work was supported by the Key Technology Research and Development Project of Sichuan Province (Grant Nos. 2025YFNH0008, 2024YFG0009, 2024ZHCG0176), and by the National Natural Science Foundation of China (Grant No. 62172290).

References

[1] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, *Mip-NeRF 360: Unbounded anti-aliased neural radiance fields*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 5460–5469, 2022.

[2] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, *CodeSLAM – Learning a compact, optimisable representation for dense visual SLAM*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2560–2568, 2018.

[3] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, *ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM*, IEEE Trans. Robot., 37(6):1874–1890, 2021.

[4] C.-M. Chung et al., *Orbeez-SLAM: A Real-time monocular visual SLAM with ORB features and NeRF-realized mapping*, in: Proceedings of the IEEE International Conference on Robotics and Automation, Curran Associates, Inc., 9400–9406, 2023.

[5] J. M. Coughlan and A. L. Yuille, *Manhattan world: Compass direction from a single image by Bayesian inference*, in: Proceedings of the Seventh IEEE International Conference on Computer Vision, Vol. 2, IEEE, 941–947, 1999.

[6] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, *BundleFusion: Real-time globally consistent 3D reconstruction using on-the-fly surface reintegration*, ACM Trans. Graph., 36(4):76a, 2017.

[7] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, *Plenoxels: Radiance fields without neural networks*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 5501–5510, 2022.

[8] A. Guédon and V. 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, IEEE, 5354–5363, 2024.

[9] S. Ha, J. Yeon, and H. Yu, *RGBD GS-ICP SLAM*, in: Computer Vision – ECCV 2024. Lecture Notes in Computer Science, Vol. 15094, Springer, 180–197, 2024.

[10] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, *2D Gaussian splatting for geometrically accurate radiance fields*, ACM SIGGRAPH 2024 Conference Papers, ACM, 32, 2024.

[11] H. Huang, L. Li, H. Cheng, and S.-K. Yeung, *Photo-SLAM: Real-time simultaneous localization and photorealistic mapping for monocular stereo and RGB-D cameras*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Curran Associates, Inc., 21584–21593, 2024.

[12] M. M. Johari, C. Carta, and F. Fleuret, *ESLAM: Efficient dense SLAM system based on hybrid representation of signed distance fields*, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 17408–17419, 2023.

[13] N. Keetha et al., *SplaTAM: Splat track & map 3D Gaussians for dense RGB-D SLAM*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Curran Associates, Inc., 21357–21366, 2024.

[14] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, *3D Gaussian splatting for real-time radiance field rendering*, ACM Trans. Graph., 42(4):139, 2023.

[15] X. Lang et al., *Gaussian-LIC: Real-time photo-realistic SLAM with Gaussian splatting and LiDAR-inertial-camera fusion*, arXiv:2404.06926, 2024.

[16] D. Liao and W. Ai, *VI-NeRF-SLAM: A real-time visual-inertial SLAM with NeRF mapping*, J. Real-Time Image Proc., 21(1):30, 2024.

[17] Y. Liu, S. Dong, S. Wang, Y. Yin, Y. Yang, Q. Fan, and B. Chen, *SLAM3R: Real-time dense scene reconstruction from monocular RGB videos*, in: 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 16651–16662, 2025.

[18] H. Matsuki, R. Murai, P. H. J. Kelly, and A. J. Davison, *Gaussian splatting SLAM*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 18039–18048, 2024.

[19] J. Naumann, B. Xu, S. Leutenegger, and X. Zuo, *NeRF-VO: Real-time sparse visual odometry with neural radiance fields*, IEEE Robot. Autom. Lett., 9(8):7278–7285, 2024.

[20] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, *KinectFusion: Real-time dense surface mapping and tracking*, in: Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality, IEEE, 127–136, 2011.

[21] A. Rosinol, J. J. Leonard, and L. Carlone, *NeRF-SLAM: Real-time dense monocular SLAM with neural radiance fields*, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 3437–3444, 2023.

[22] E. Sandström, Y. Li, L. Van Gool, and M. R. Oswald, *Point-SLAM: Dense neural point cloud-based SLAM*, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 18433–18444, 2023.

[23] R. Schnabel, R. Wahl, and R. Klein, *Efficient RANSAC for point-cloud shape detection*, Comput. Graph. Forum, 26(2):214–226, 2007.

[24] A. Segal, D. Haehnel, and S. Thrun, *Generalized-ICP*, Proc. Robot., Sci. Syst., MIT Press, 2009. `https://www.roboticsproceedings.org/rss05/p21.pdf`

[25] J. Straub et al., *The Replica dataset: A digital replica of indoor spaces*, arXiv:1906.05797, 2019.

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, *A benchmark for the evaluation of RGB-D SLAM systems*, in: Proceedings of the International Conference on Intelligent Robot Systems, IEEE, 573–580, 2012.

[27] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, *iMAP: Implicit mapping and positioning in real-time*, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, Curran Associates, Inc., 6209–6218, 2021.

[28] L. C. Sun et al., *MM3DGS SLAM: Multi-modal 3D Gaussian splatting for SLAM using vision, depth, and inertial measurements*, arXiv:2404.00923, 2024.

[29] Z. Teed and J. Deng, *DROID-SLAM: Deep visual SALM for monocular, stereo, and RGB-D cameras*, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., 34:16558–16569, 2021.

[30] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny, *VGGT: Visual geometry grounded transformer*, in: 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 5294–5306, 2025.

[31] T. Whelan, R. Salas-Moreno, B. Glocker, A. Davison, and S. Leutenegger, *ElasticFusion: Real-time dense SLAM and light source estimation*, Int. J. Robot. Res., 35:1697–1716, 2016.

[32] C. Wu, Y. Duan, X. Zhang, Y. Sheng, J. Ji, and Y. Zhang, *MM-Gaussian: 3D Gaussian-based multi-modal fusion for localization and reconstruction in unbounded scenes*, arXiv:2404.04026, 2024.

[33] C. Yan et al., *GS-SLAM: Dense visual SLAM with 3D Gaussian splatting*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Curran Associates, Inc., 19595–19604, 2024.

[34] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, *Voxfusion: Dense tracking and mapping with Voxel-based neural implicit representation*, in: Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, IEEE, 499–507, 2022.

[35] M. Yu, T. Lu, L. Xu, L. Jiang, Y. Xiangli, and B. Dai, *GSDF: 3DGS meets SDF for improved rendering and reconstruction*, arXiv:2403.16964, 2024.

[36] K. Zhang, G. Riegler, N. Snavely, and V. Koltun, *NeRF++: Analyzing and improving neural radiance fields*, arXiv:2010.07492, 2020.

[37] S. Zhang, J. Wang, Y. Xu, N. Xue, C. Rupprecht, X. Zhou, Y. Shen, and G. Wetzstein, *FLARE: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views*, in: Proceddings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 21936–21947, 2025.

[38] Y. Zhang, F. Tosi, S. Mattoccia, and M. Poggi, *GO-SLAM: Global optimization for consistent 3D instant reconstruction*, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 3727–3737, 2023.

[39] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, *NICE-SLAM: Neural implicit scalable encoding for SLAM*, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Curran Associates, Inc., 12776–12786, 2022.