GPGS: Geometric Priors for 3D Gaussian Splatting in Structural Environments

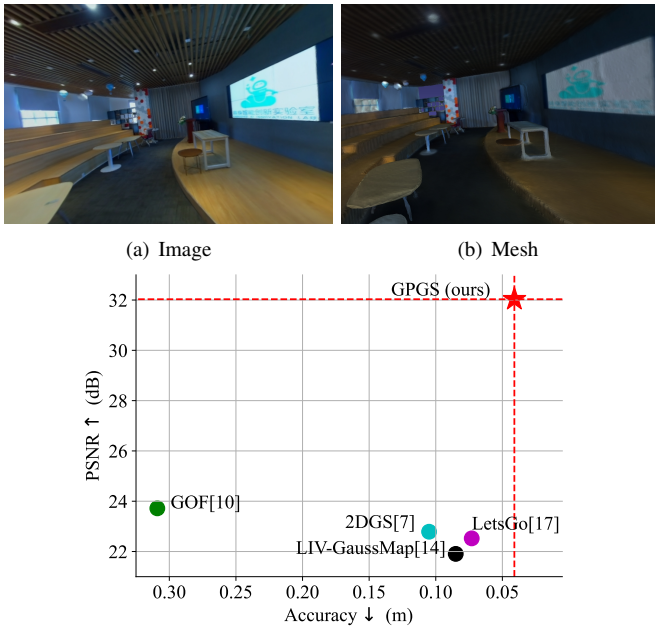
Ziwei Xu*, Wen Chen*,†, Shilong Wang, Zile Ouyang, Shengwei Bian and Shunbo Zhou†

***Abstract*— Recently, 3D Gaussian Splatting (3DGS) has garnered significant attention for its remarkable capacity to efficiently synthesize novel views with high fidelity. Nevertheless, 3DGS encounters challenges in accurately representing the geometry of real-world scenes. To address this issue, previous methods commonly utilize a depth-normal consistency term on 2D images to regulate the geometry of 3D Gaussians. However, these methods degrade in performance when dealing with low-texture surfaces or limited training views. In contrast, we present GPGS, a novel approach that directly regulates Gaussians in 3D space using Geometric Priors (GP). Given posed LiDAR scans and images, we organize the point clouds into a hierarchical voxel map. Each voxel contains occupancy information and explicitly reveals the internal planar or non-planar structure. We propose a novel divide-and-conquer strategy to separately regulate Gaussians in planar and non-planar voxels. For planar voxels, we design positional and rotational constraints to align Gaussians with the estimated plane. Considering the noisy ranging measurements of complex structures, we use depth-normal consistency to regularize Gaussians in non-planar voxels. Additionally, an occupancy-aware density control strategy is introduced to confine the densification process within occupied voxels, thus reducing artifacts. Extensive experiments on real-world datasets show that our proposed approach outperforms existing state-of-the-art methods in both geometric accuracy and visual quality.**

I. INTRODUCTION

3D reconstruction technology has broad application prospects in fields such as embodied intelligence [1], [2] and autonomous driving [3], [4]. Among them, differentiable reconstruction techniques based on Neural Radiance Fields (NeRF) [5] or 3D Gaussian Splatting (3DGS) [6] have received widespread attention and favor because they can visually achieve highly realistic novel-view synthesis. Although the appearance-first optimization strategy can ensure excellent visual quality, there is a notable scarcity of accurate geometric performance [7], [8]. This shortcoming substantially impedes the applications of these technologies in application scenarios with stringent requirements for images and geometry. In this paper, we focus on solving the problems faced by the 3DGS in simultaneously perusing high-quality image synthesis and accurate mesh.

To address this problem, several methods endeavor to recover detailed geometry through surface modeling. 2DGS [7] and Gaussian surfels [8] flatten the 3D ellipsoid of Gaussians into 2D oriented planar disks, thereby achieving a close alignment to the actual surfaces. These methods introduce depth-normal consistency terms to enhance the quality of surface reconstruction in a self-supervised manner. Building upon 2DGS, PGSR [9] further employs an unbiased depth rendering method and multi-view regularization terms to improve the overall performance. GOF [10] enables the direct extraction of meshes from 3D Gaussians by establishing a Gaussian opacity field and makes use of a ray-tracing-based volume rendering method. However, these image-only input methods encounter difficulties in accurately reconstructing surfaces with weak textures. Another group of methods to solve the problem depends on direct range measurements. These measurements are obtained from dense depth maps acquired by RGB-D cameras [11], [12], [13] or from sparse points captured by 3D LiDARs [14], [15], [16], [17], [18]. The incorporation of such measurements is advantageous for the initializations of Gaussians. Moreover, it offers direct depth supervision by comparing the depth rasterization results with the observed depth. This leads to faster convergences and a better spatial distribution of Gaussians. But the use of 2D depth maps captured from limited number of viewpoints to supervise the primitives within 3D space still suffers from a lack of constraints. In particular, these



(c) Comparison with SOTAs.

Fig. 1. A comparison of the visual and geometric performance between the proposed GPGS and state-of-the-art (SOTA) methods is presented. (a) The RGB image rendered by our method. (b) The mesh model extracted by our method. (c) All methods are evaluated on a real-world dataset with a ground-truth map. Our method demonstrates the best performance in both visual quality and geometric accuracy.

* Ziwei Xu and Wen Chen contributed equally to this work
All authors are with Huawei Cloud Computing Technologies Co.,Ltd., Shenzhen, China.
† Wen Chen and Shunbo Zhou are corresponding authors.

Manuscript 1391 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 1, 2025.

methods face significant challenges in addressing noisy range measurements and the misalignment between images and range data due to calibration or synchronization errors.

In this paper, we propose a novel approach capable of concurrently attaining high-fidelity visual appearance and high-accuracy geometrical structure. Unlike the conventional approach of solely supervising the geometry through the depth rasterization process, we utilize geometric priors derived from the point cloud map. This enables us to initialize and regulate the spatial distribution and shape of Gaussians directly in the 3D space. As shown in Fig. 1, this strategy not only leads to a substantial improvement in 3D geometry but also, as a by-product, reduces artifacts, thereby enhancing the visual quality. LI-GS [18] shares a similar underlying intuition with our approach in directly regulating Gaussian primitives within 3D space. It extracts plane-constrained Gaussian Mixture Models (GMMs) [19] from LiDAR point clouds and utilize them to optimize Gaussian primitives. However, the conversion from points to GMMs requires highly parallelized implementation on GPU. Moreover, the plane-constrained GMMs and the representation of Gaussian surfels may introduce unexpected bias in non-planar zones. In contrast, our approach employs a more lightweight method to extract geometric priors from LiDAR scans. Additionally, we introduce a divide-and-conquer strategy to regulate Gaussians in planar and non-planar zones respectively.

Specifically, the posed LiDAR scans are organized into a hierarchical voxel map [20]. This is achieved by adaptively partitioning the entire 3D space into voxels of varying sizes, with the division criterion being whether a voxel contains a plane. Each voxel provides occupancy information and explicitly indicates the inside planar or non-planar structure. Subsequently, a voxelized Gaussian map is constructed accordingly. In this map, each Gaussian is initialized based on the LiDAR points and their spatial distribution within the voxel. We propose a novel divide-and-conquer strategy to separately regulate the Gaussians in the planar or non-planar voxel. For Gaussians in planar voxel, we directly flatten 3D ellipsoids into 2D surfels, and design positional and rotational constrains to align surfels with the estimated plane in voxel. Considering the noisy and sparse measurements on complexity structure, we instead leverage the depth-normal consistency [7], [8] to regularize those Gaussians in non-planar voxel. Additionally, an occupancy-aware density control strategy is introduced. This strategy serves to restrict the densification process within occupied voxels, thereby reducing artifacts.

Our main contributions are summarized as follows.

- We present a novel method for environment reconstruction. This method fuses posed LiDAR scans and visual images using 3D Gaussian primitives organized by a hierarchical voxel structure.
- We design a divide-and-conquer strategy and leverage different geometric priors to regulate Gaussians located in the planar and non-planar voxels separately. It leads to improvements in geometric performance.
- We introduce an occupancy-aware density control method to restrict all Gaussians within occupied voxels. It reduce artifacts thus improve visual appearance.
- We extensively compare our method with state-of-the-art approaches on real-world datasets. Our method can concurrently achieve superior visual appearance and higher geometric accuracy.

II. Preliminaries

3D Gaussian Splatting [6] represents the geometry and appearance of a 3D space via a set of Gaussian primitives, which are explicitly parameterized by their position $\mathbf{p}_k$, covariance matrix Σ, opacity o_k and spherical harmonics coefficients $\mathbf{c}_k$ for view-dependent appearance. The Gaussian values can be calculated by:

$$\mathcal{G}(\mathbf{p}) = exp(-\frac{1}{2}(\mathbf{p}-\mathbf{p}_k)^T\Sigma^{-1}(\mathbf{p}-\mathbf{p}_k)) \tag{1}$$

For image rendering, each 3D Gaussian is transformed into camera coordinates and projected to image space, which obtains a 2D Gaussian $\mathcal{G}^{2D}$. These 2D Gaussians are sorted in front-to-back order and alpha-blended into a rendered image via volumetric rendering:

$$\mathbf{c}(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{c}_k o_k \mathcal{G}^{2D}(\mathbf{x}) \prod_{j=1}^{k-1}(1 - o_j \mathcal{G}^{2D}(\mathbf{x})) \tag{2}$$

Inspired by 3DGS, 2D Gaussian Splatting [7] represents the scene via planar Gaussian surfels defined in a local tangent space, which is parameterized by position $\mathbf{p}_k$, normal vector $\mathbf{n}_k$, two orthogonal vectors $\mathbf{t}_{u_k}$, $\mathbf{t}_{v_k}$ with corresponding radii in each direction $r_{u_k} \geq r_{v_k}$, alongside opacity and spherical harmonics which is identical to 3DGS. For a point $\mathbf{p} = [u, v]$ located in tangent space, the 2D Gaussian value can be calculated by:

$$\mathcal{G}^{2D}(\mathbf{p}) = exp(-\frac{u^2+v^2}{2}) \tag{3}$$

Image rendering using 2DGS shares a similar strategy with 3DGS by projecting Gaussian primitives from tangent space to image space and conduct the volumetric rendering described in Eq. 2.

III. Methodology

Fig. 2 presents an overview of ours proposed system, which starts from building a hierarchical voxel map from raw data for constructing geometric priors (Sec. III-A), follows the initialization of global voxelized Gaussian map converted from the voxel map (Sec. III-B), and is finalized by a novel divide-and-conquer geometric optimization strategy (Sec. III-C). The mathematical notations used in this paper are introduced in Tab. I.

A. Voxelization

Given a collection of RGB images, LiDAR scans, and 6-axis IMU measurements obtained from real-world scenes, it is assumed that these data sources are hardware-synchronized and these sensors are rigidly fixed relative to each other. Moreover, the extrinsic and intrinsic parameters of these

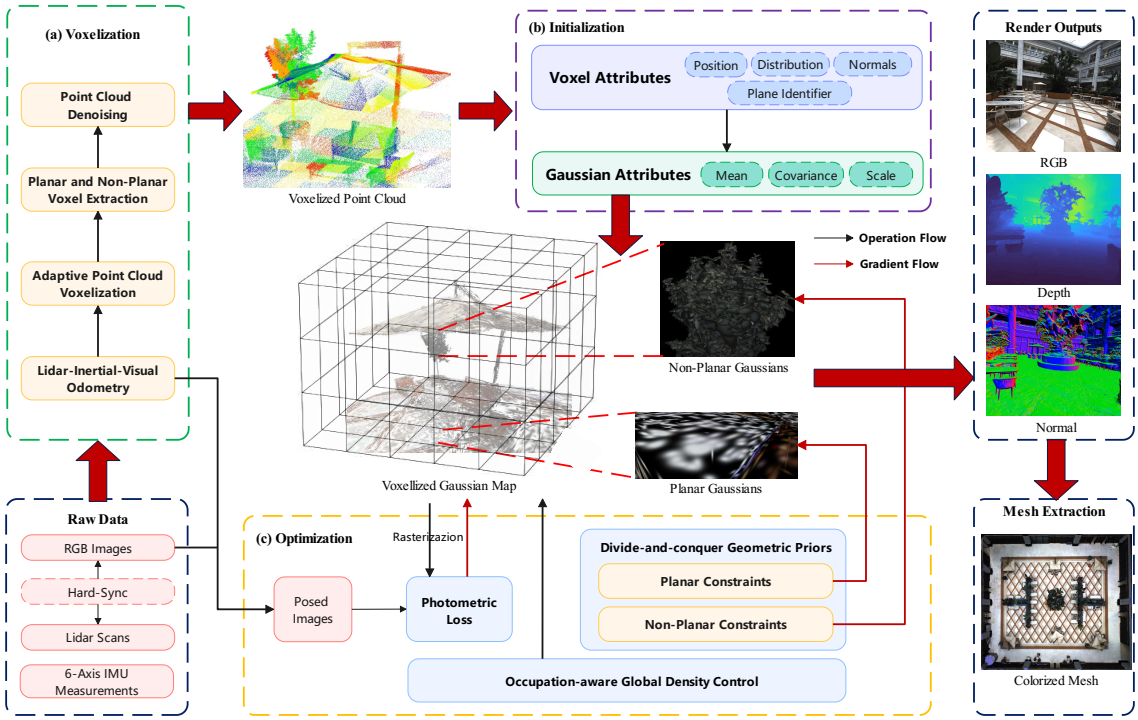


Fig. 2. Overview of our system.

TABLE I

EXPLANATIONS OF THE MATHEMATICAL NOTATIONS.

Notation	Explanation
	Voxel Map
i	The index of a specific voxel grid
$\mathcal{P}_i$	The coordinates of point set inside the i-th voxel
$\mathbf{\Sigma}_i = [\boldsymbol{\sigma}_{i0}, \boldsymbol{\sigma}_{i1}, \boldsymbol{\sigma}_{i2}]$	Covariance matrix of the i-th voxel
$\boldsymbol{\sigma}_{ik}$	Corresponding eigenvectors of $\mathbf{\Sigma}_i$
$\boldsymbol{\lambda}_i = [\lambda_{i0}, \lambda_{i1}, \lambda_{i2}]$	Eigenvalues of the covariance matrix in incremental order, $\lambda_{i0} \leq \lambda_{i1} \leq \lambda_{i2}$
d_i	Plane identifier of the i-th voxel
$\mathbf{n}_i = \boldsymbol{\sigma}_{i0}$	Normal vector of i-th voxel while $d_i = 1$
$\mathbf{c}_i$	Center position of the i-th voxel
	Gaussian Map
$\mathbf{p}_k$	Center of the k-th Gaussian
$\mathbf{R}_k = [\mathbf{r}_0, \mathbf{r}_1, \mathbf{r}_2]$	Rotation matrix of the k-th Gaussian
$\mathbf{S}_k = diag(s_0, s_1, s_2)$	Scaling matrix of the k-th Gaussian
o_k	Opacity of the k-th Gaussian

sensors have been pre-calibrated offline. We first employ the SOTA LiDAR-Inertial-Visual odometry method, FAST-LIVO2 [21], to fuse the captured multi-sensor measurements and estimate the poses of all LiDAR scans. Subsequently, a hierarchical voxel structure is utilized to organize the LiDAR scans with estimated poses. The adaptive point cloud voxelization method proposed in [20] is adopted for this purpose. The voxelization starts with a default voxel length[1]. It iteratively divides the current-layer 3D space into eight smaller sub-spaces of the next layer until reaching the default maximum layers. The voxelization of the next layer depends on whether all points in a voxel are coplanar. If so, the voxelization of this voxel stops; otherwise, it continues to the next layer with smaller voxels. The hierarchical voxel structure is crucial for extracting planar and non-planar voxels of different sizes, which serve as geometric priors for subsequent steps. Finally, to efficiently initialize Gaussian primitives, we perform point cloud denoising by downsampling points in planar voxels based on the fitted plane parameters.

[1]We set $0.5m$ for outdoor scenes and $0.25m$ for indoor scenes.

B. Initialization

After voxelization process, each voxel provides crucial geometric information, determined by the following parameter set $\mathcal{V} = \{i, \mathcal{P}_i, \mathbf{\Sigma}_i, \boldsymbol{\lambda}_i, d_i, \mathbf{n}_i, \mathbf{c}_i\}$, where the definitions are explained in Tab. I. Based on the plane identifier d_i of each voxel, which is determined when the smallest eigenvalue λ_{i0} is smaller than a certain threshold, the global map can be clearly distinguished into **planar regions** and **non-planar regions**. To simultaneously capture both smooth geometric structure of planar regions and the intricate details of non-planar regions, a divide-and-conquer scene representation strategy is introduced in this work, which represents the planar regions using Gaussian surfels and non-planar regions using the classic 3D Gaussian ellipsoids.

The geometry of Gaussian primitives, which includes both 2D surfels and 3D ellipsoids, can be defined by the following parameter set $\mathcal{G} = \{\mathbf{p}_k, \mathbf{R}_k, \mathbf{S}_k, o_k\}$ with the definition explained in Tab. I. The initial 3D Gaussian ellipsoids can be converted from non-planar voxels via following rules:

$$\mathbf{p}_k = \mathcal{P}_i(k), \mathbf{R}_k = \mathbf{\Sigma}_i, o_k = 0.5$$
$$s_0 = \frac{\sqrt[3]{\lambda_{i0}}}{|\mathcal{P}_i|}, s_1 = \frac{\sqrt[3]{\lambda_{i1}}}{|\mathcal{P}_i|}, s_2 = \frac{\sqrt[3]{\lambda_{i2}}}{|\mathcal{P}_i|} \tag{4}$$

when k-th Gaussian is bounded by the i-th voxel, and $|\mathcal{P}_i|$ denotes the total number of points inside the i-th voxel. For 2D Gaussian surfels, which are initialized from planar voxels

with the smallest eigenvalue $\lambda_{i0} \to 0$, we change the rule for initializing scaling matrix in order to obtain the correct size of Gaussian surfels:

$$s_0 = 0, s_1 = \frac{\sqrt{\lambda_{i1}}}{|\mathcal{P}_i|}, s_2 = \frac{\sqrt{\lambda_{i2}}}{|\mathcal{P}_i|} \tag{5}$$

C. Optimization

The design of our optimization method starts from considering the position and shape of Gaussian primitives as explicit indicators of actual surface. Fundamental works [6], [7] which solely optimizes scene representation with photometric supervision, tend to produce noisy distribution of Gaussian primitives due to the lack of geometric constraint. To tackle this problem, we propose a novel regularization method for our divide-and-conquer scene representation which fully exploits the prior knowledge embedded in hierarchical voxel map.

1) Divide-and-conquer Geometric Priors: Our geometric priors derived from the hierarchical voxel map employs domain-specific constraints to guide the distribution of Gaussian primitives separately in planar and non-planar regions. For each training step with one specific training view, we select N planar voxels and M non-planar voxels inside its view frustum and optimize the Gaussian primitives bounded by these voxels. The planar constraint can be computed via

$$\mathcal{L}_{planar} = \mathcal{L}_{pos} + \mathcal{L}_{rot} \tag{6}$$

where $\mathcal{L}_{pos}$ and $\mathcal{L}_{rot}$ denote positional and rotational constraint, respectively.

Positional constraint regularizes the center of 2D Gaussian surfels on the estimated voxel plane, which is calculated by minimizing the distance between the center of Gaussian surfels and the corresponding voxel plane:

$$\mathcal{L}_{pos} = \frac{1}{K}\sum_{k=1}^{K} \mathbf{n}_i^T(\mathbf{p}_k - \mathbf{c}_i) \tag{7}$$

when k-th Gaussian is bounded by the i-th voxel.

Rotational constraint ensures the alignment between the shape of Gaussian surfels and the estimated voxel plane. For rotation matrix of the k-th Gaussian surfel $\mathbf{R}_k = \mathbf{R}_{init} \cdot \mathbf{R}_{opt}$, where $\mathbf{R}_{init}$ denotes the initial orientation and $\mathbf{R}_{rot}$ indicates the optimizable part, we convert $\mathbf{R}_{rot}$ into Euler angles $[\phi, \theta, \psi]^T$ and compute the rotational constraint via

$$\mathcal{L}_{rot} = \frac{1}{K}\sum_{k=1}^{K}(|\theta_k| + |\psi_k|) \tag{8}$$

where ϕ denotes the rotation angle w.r.t the normal vector of Gaussian surfels and θ, ψ indicates the rotation w.r.t the orthogonal vectors of Gaussian surfels, respectively. The minimization of $\mathcal{L}_{rot}$ aligns the rotation axis of Gaussian surfels with the estimated normal vector of plane voxels, thus ensures the shape accuracy of Gaussian surfels.

For non-planar regions, due to high complexity in terms of geometry and limited scanning precision of LiDAR sensors, it is unreliable to directly reproduce geometrical priors from the distribution of points in non-planar voxels. To tackle this issue, we refer to the design of normal consistency loss introduced in [7], which encourages the distribution of 3D Gaussian ellipsoids to be aligned with the surface estimated by depth gradients:

$$\mathcal{L}_{normal} = \sum_{k\in M}(1 - \tilde{\mathbf{n}}_k^T \nabla \mathbf{D}) \tag{9}$$

where $\tilde{\mathbf{n}}_k$ denotes the predicted normal from 3D Gaussian ellipsoids and $\nabla\mathbf{D}$ defines the gradients of rendered depth maps in non-planar region, which is calculated by finite differences from adjacent depth points.

2) Occupation-aware Global Density Control: Refer to the fundamental work [6], the proposed system adaptively regulates the number, density and distribution of the Gaussian primitives throughout the optimization phase by cloning, splitting and pruning mechanisms. In order to maintain an accurate geometric structure of the Gaussian map while its density is dynamically adjusted, we introduce the occupation information provided by our hierarchical voxel map into the density control mechanism.

Specifically, while a Gaussian primitive is cloned or split, we record its corresponding voxel id i, voxel center $\mathbf{c}_i$ and voxel edge length l_i, and assign them to the densified Gaussian primitive. We propose the following occupation loss to penalize the distribution of Gaussian primitives while they move away from their corresponding voxels, or their scales grow larger than the voxel boundary:

$$\begin{aligned}\mathcal{L}_{occ} = \frac{1}{K}\sum_{k=1}^{K}[1 - exp(\sigma(||\mathbf{p}_k - \mathbf{c}_i|| - \frac{l_i}{2})) \\ +1 - exp(\lambda(max(s_{0k}, s_{1k}, s_{2k}) - \frac{l_i}{2}))]\end{aligned} \tag{10}$$

Furthermore, it is desired that Gaussian primitive which moves far away across the voxel boundary to be pruned. Based on the original strategy which prunes Gaussian primitives with low opacity value or extremely large scales [6], we introduce a distance-based criterion which prunes the Gaussian primitive when its distance to the corresponding voxel center exceeds the voxel edge length: $||\mathbf{p}_k - \mathbf{c}_i|| > l_i$.

The final loss is formulated by adding the geometric and occupation constraints above alongside the classic photometric loss $\mathcal{L}_c$ introduced in [6]:

$$\mathcal{L} = \mathcal{L}_c + \alpha\mathcal{L}_{planar} + \beta\mathcal{L}_{normal} + \gamma\mathcal{L}_{occ} \tag{11}$$

$$\mathcal{L}_c = 0.8\mathcal{L}_1(\tilde{\mathbf{I}}, \mathbf{I}) + 0.2\mathcal{L}_{D-SSIM}(\tilde{\mathbf{I}}, \mathbf{I}) \tag{12}$$

where $\tilde{\mathbf{I}}, \mathbf{I}$ denotes rendered and input images and α, β, γ represents corresponding loss weights.

3) Uncertainty in real-world datasets: Unlike benchmark datasets that are widely evaluated by the community with pixel-level accurate camera poses and multi-view consistent appearance [22], [23], self-captured datasets in real-world usually suffer from uncertainty in camera poses and varying lightning conditions, which lead to artifacts and floaters. To tackle these problems, we first integrate camera pose optimization strategy into our optimization phase, which automatically adjusts the camera view matrices $\mathcal{T} = [\mathbf{R}|\mathbf{t}]$

by calculating the camera gradients w.r.t reconstruction loss (Eq. 11). In this work, we refer to the formulation introduced in [24], which computes the gradients of reconstruction loss to translation and rotation components via

$$\frac{\delta \mathcal{L}}{\delta \mathbf{t}} = -\sum_k \frac{\delta L}{\delta \tilde{\mathbf{p}}_k}, \quad \frac{\delta \mathcal{L}}{\delta \mathbf{R}} = -\left[\sum_k \frac{\delta L}{\delta \tilde{\mathbf{p}}_k}(\mathbf{p}_k - \mathbf{t})^T\right]\mathbf{R} \quad (13)$$

where $\tilde{\mathbf{p}}_k$ denotes the position of Gaussian primitives in view space. An analytical format of camera gradient calculation can be found in [12]. Furthermore, we implement bilateral grid guided optimization [25] into our framework, which disentangles per-view exposure adjustment or color correction, thus improves multi-view consistency in terms of appearance.

D. Surface Extraction

Once the optimization is completed, we are capable of rendering high-quality RGB, depth and normal images from our Gaussian map. Refer to [7], we adopt TSDF algorithm to fuse our rendered RGB and depth maps and build the colorized mesh, which is structural aligned with the reconstructed Gaussian representation.

IV. Experiments

We report experimental results on real-world datasets. All collected scenes provide ground-truth point clouds for geometric accuracy evaluation. We present the details of these datasets in Section IV-A. Then we compare our method with state-of-the-art approaches based on metrics of visual appearance and geometric accuracy in Section IV-B. We also present ablation studies in Section IV-C. All the experiments are conducted on a desktop with a single NVIDIA Tesla V100 GPU and 32 GB memory.

A. Experimental Setup

We compare our method with state-of-the-art approaches on real-world datasets including 6 different scenes whose details are summarized in Tab. II. We choose 3 selected sequences with ground-truth maps in the Oxford spires dataset [26], which is a large-scale outdoor dataset with average trajectory length in each sequence exceeding 400 meters. The benchmark dataset is collected by a perception unit equipped with a Hesai QT64 LiDAR operating at 10 Hz, three synchronized RGB fisheye cameras facing forward, left and right with auto-exposure enabled operating at 20 Hz, and a cellphone-grade IMU operating at 400 Hz. The geometry of the ground truth is obtained from a Leica RTC360 TLS with average accuracy of $3-7mm$. To further evaluate the robustness of our framework in various environments, we construct a self-captured dataset with 3 indoor sequences from small to large scale.The dataset is collected by a consumer-grade scanner equipped with a Livox Mid360 LiDAR operating at 10 Hz, an integrated IMU operating at 200 Hz and two synchronized RGB fisheye cameras with auto-exposure enabled facing front-left and front-right operating at 10 Hz. We collect ground-truth data using a NavVis VLX 2 laser scanner and register the scans using NavVis SiteMaker software.

TABLE II

Details of the real-world datasets.

Dataset Information	Scene	# Images	# LiDAR Frames	Length(m)
Oxford Spires Dataset	palace	1545	3601	490.0
Sensors: Hesai QT64, cellphone-grade IMU, 3 × fisheye cameras.	college	1569	2611	290.0
Ground Truth: Leica RTC360 TLS.	quarter	1797	2396	390.0
Self-captured Dataset	pantry	1174	1174	48.8
Sensors: Livox Mid360 (IMU inside), 2 × fisheye cameras	roma	1398	2097	107.4
Ground Truth: NavVis VLX 2	office	2006	3007	217.5

For each sequence in both datasets, we calculate the trajectories of LiDAR and a selected camera by employing FAST-LIVO2 [21] to fuse images, LiDAR scans and IMU. The trajectories of other cameras are calculated by transforming the LiDAR trajectory with the pre-calibrated extrinsic parameters. The reconstruction of hierarchical voxel map refers to our voxelization module (Sec. III-A). Before the optimization phase, we constrain the number of training views for the Gaussian map to a maximum of 2000 by conducting an image filter which uniformly picks one image from every N adjacent observations.

We evaluate each method quantitatively considering both rendering and geometric quality. Following [6], [7], [8], we assess the rendering quality by comparing the renders of the Gaussian map with the input images, using metrics such as PSNR, SSIM, and LPIPS. For the evaluation of geometric quality, we follow [14], [18] to report accuracy, completeness, precision, recall and F1-score by comparing the reconstructed mesh and the ground-truth point cloud. The precision, recall and F1-score are computed using a threshold of 20 cm. The ground-truth point clouds are aligned with the datasets by conducting fine registration using Iterative Closest Points (ICP) algorithm [27], and the boundaries of ground-truth are cropped refer to the bounding box of denoised outputs from FAST-LIVO2 [21].

B. Results and Evaluation

We compare our method with state-of-the-art reconstruction methods based on Gaussian primitives using self-supervised depth-normal consistency terms (2DGS [7], GOF [10]) and depth supervision from ranging measurements (LIV-GaussMap [14], LetsGo [17]). For fair comparison, all methods are initialized using the points from our hierarchical voxel map, and trained by adopting the camera poses initialized from FAST-LIVO2 [21]. Following [7], meshes are extracted using the TSDF algorithm with voxel size set to $3cm$, SDF truncation value set to 0.12 and depth truncation set to $15m$ by integrating rendered RGB and depth maps from each method. Referring to the default configuration of Gaussian optimization in [6], we unify the hyperparameters for all methods and set the total iteration steps as 90000.

Figs. 3 and 4 visualize the results of our approach and state-of-the-art methods in terms of rendering and surface reconstruction, with the quantitative results reported in Tab. III. The results show that our approach outperforms all state-of-the-art methods in both visual and geometric quality, and also demonstrate the robustness of our GPGS across indoor and outdoor scenes with various scales.

To further assess the geometric quality, we present quantitative comparisons of all Gaussian-based methods and FAST-

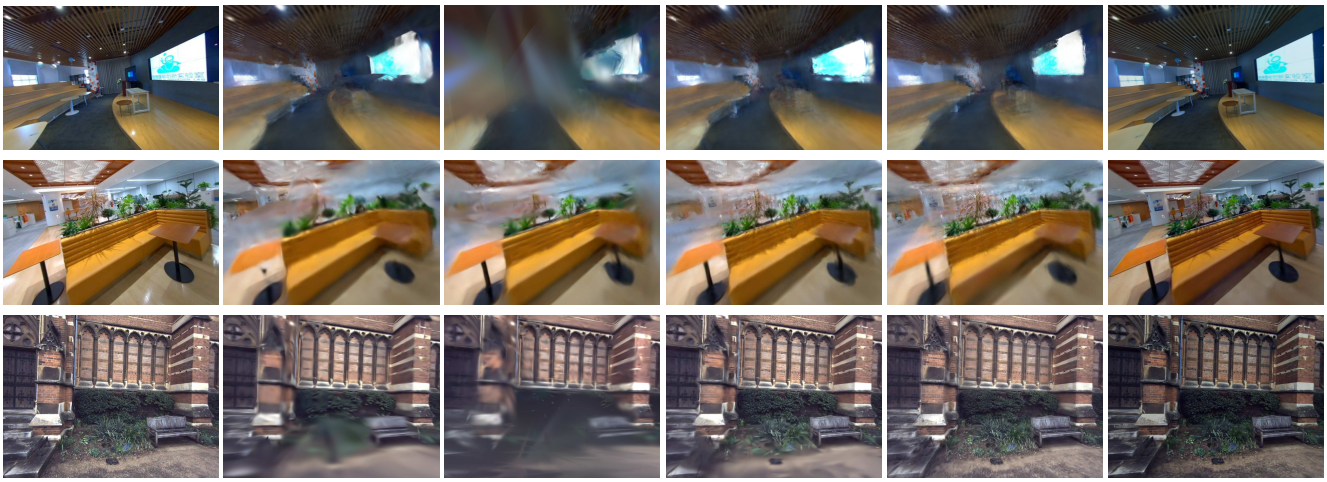


Fig. 3. Rendering quality comparison of various state-of-the-art methods on real-world datasets.

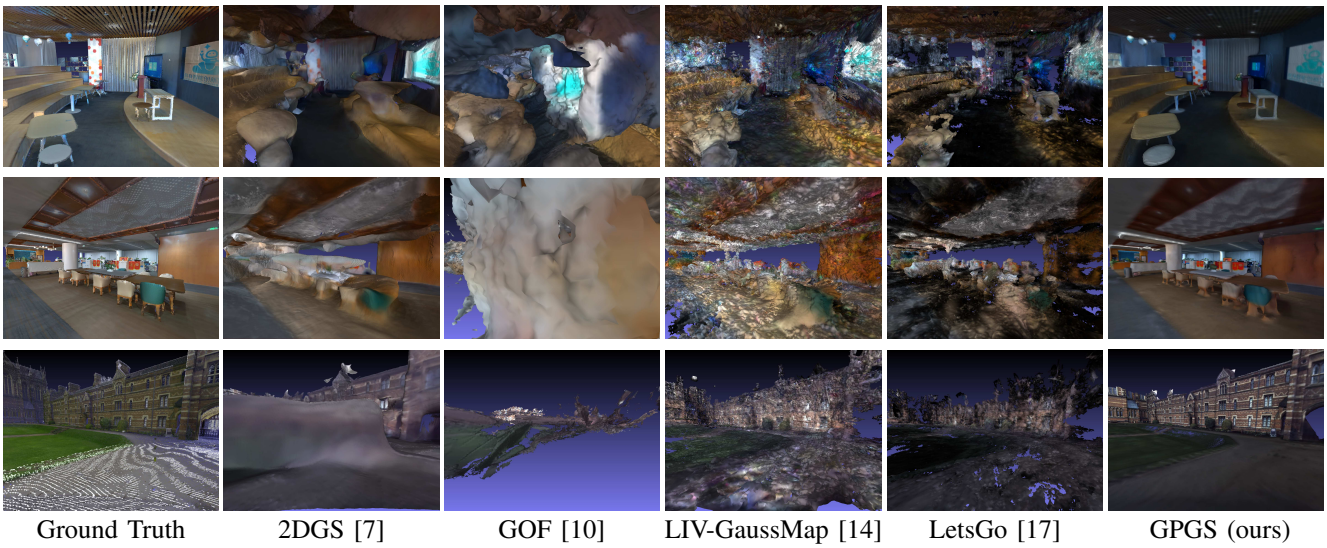


Fig. 4. Geometric quality comparison of various state-of-the-art methods on real-world datasets.

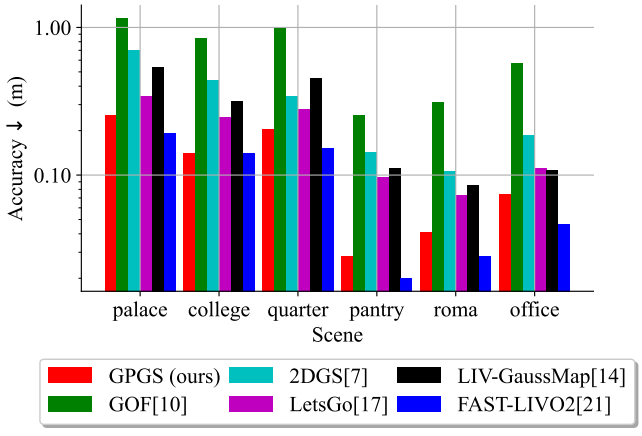


Fig. 5. Quantitative comparisons of all Gaussian-based methods and FAST-LIVO2 [21] in terms of geometric accuracy.

LIVO2 [21]. As depicted in Fig. 5, FAST-LIVO2 exhibits the best performance in terms of geometric accuracy. All Gaussian-based methods initialized with point cloud maps from FAST-LIVO2 experience a decline in geometric performance. This is attributed to the utilization of the photometric loss term during Gaussian optimization. Nevertheless, among these Gaussian-based methods, our GPGS achieves the best visual results with the least loss of geometric accuracy.

C. Ablation Study

We evaluate the effects of major components proposed in our optimization phase by disabling each component and compare the reconstruction results quantitatively. Overall, the results of evaluation on the scene *roma* show that our full framework enjoys a great balance between rendering and geometric quality, while the geometric quality outperforms all ablation studies. The details are listed in Tab. IV.

1) Divide-and-conquer Geometric Priors: By disabling the planar constraint, all Gaussian primitives are considered as non-planar and supervised by non-planar constraints. Although a slight improvement on rendering quality is observed, as shown in Fig. 6(a) and 6(b), the flatness of planar structures degenerates, which causes a significant reduction

TABLE III

COMPARISON RESULTS ON REAL-WORLD DATASETS.

Methods	Metrics	Scene					
		palace	college	quarter	pantry	roma	office
2DGS [7]	PSNR↑	23.062	24.365	21.675	25.741	22.787	22.194
	SSIM↑	0.793	0.774	0.746	0.887	0.797	0.794
	LPIPS↓	0.446	0.369	0.438	0.302	0.387	0.400
	Acc.↓	0.694	0.435	0.342	0.143	0.105	0.185
	Comp.↓	0.712	0.414	2.658	0.056	0.081	1.156
	Pre.↑	0.266	0.547	0.365	0.858	0.869	0.715
	Re.↑	0.307	0.565	0.191	0.971	0.938	0.530
	F1.↑	0.285	0.556	0.251	0.911	0.902	0.609
GOF [10]	PSNR↑	17.141	20.233	17.263	25.798	23.717	20.223
	SSIM↑	0.751	0.736	0.703	0.886	0.813	0.777
	LPIPS↓	0.464	0.402	0.468	0.303	0.357	0.424
	Acc.↓	1.157	0.840	0.992	0.253	0.309	0.573
	Comp.↓	7.041	3.676	4.536	0.094	0.119	0.428
	Pre.↑	0.127	0.216	0.125	0.605	0.474	0.252
	Re.↑	0.018	0.061	0.023	0.908	0.859	0.295
	F1.↑	0.031	0.095	0.039	0.726	0.611	0.272
LIV-GaussMap [14]	PSNR↑	24.087	25.606	22.978	26.885	21.906	22.867
	SSIM↑	0.807	0.798	0.770	0.897	0.785	0.808
	LPIPS↓	0.425	0.330	0.399	0.276	0.414	0.374
	Acc.↓	0.537	0.313	0.448	0.111	0.085	0.108
	Comp.↓	0.767	0.639	2.838	0.038	0.050	1.134
	Pre.↑	0.211	0.479	0.282	0.906	0.921	0.895
	Re.↑	0.382	0.608	0.166	0.988	0.983	0.645
	F1.↑	0.272	0.536	0.209	0.945	0.951	0.750
LetsGo [17]	PSNR↑	24.067	24.721	22.212	25.104	22.523	21.974
	SSIM↑	0.821	0.794	0.768	0.882	0.798	0.799
	LPIPS↓	0.388	0.322	0.387	0.320	0.369	0.377
	Acc.↓	0.343	0.244	0.278	0.096	0.073	0.111
	Comp.↓	1.102	0.604	2.939	0.040	0.052	1.131
	Pre.↑	0.462	0.532	0.499	0.912	0.943	0.874
	Re.↑	0.467	0.551	0.249	0.983	0.967	0.635
	F1.↑	0.465	0.541	0.333	0.946	0.955	0.735
GPGS (ours)	PSNR↑	29.673	31.111	27.742	34.483	32.032	29.574
	SSIM↑	0.892	0.914	0.863	0.961	0.940	0.906
	LPIPS↓	0.288	0.202	0.288	0.151	0.163	0.216
	Acc.↓	0.252	0.139	0.202	0.028	0.041	0.074
	Comp.↓	0.593	0.252	0.338	0.031	0.038	0.060
	Pre.↑	0.596	0.838	0.652	1.000	0.985	0.942
	Re.↑	0.620	0.785	0.618	0.992	0.995	0.976
	F1.↑	0.608	0.811	0.639	0.996	0.990	0.959

Best results are highlighted as 1st, 2nd.

TABLE IV

COMPARISON RESULTS OF ABLATION STUDY ON ROMA SQUARE

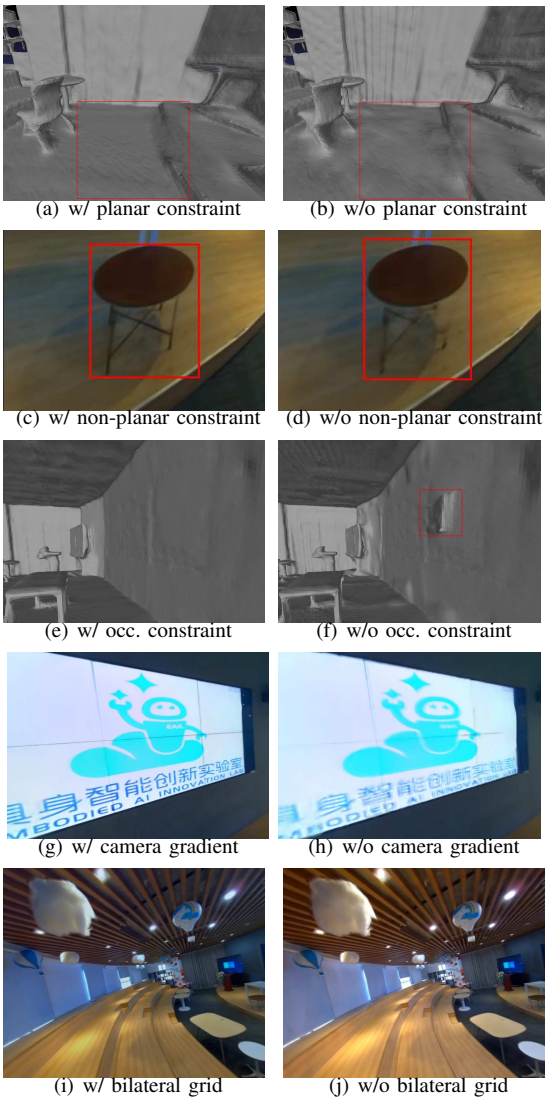
Methods	Metrics							
	PSNR↑	SSIM↑	LPIPS↓	Acc.↓	Comp.↓	Pre.↑	Re.↑	F1.↑
w/o planar constraint	32.168	0.942	0.158	0.066	0.051	0.954	0.993	0.973
w/o non-planar constraint	31.500	0.936	0.172	0.046	0.039	0.967	0.994	0.980
w/o occ. constraint	31.967	0.940	0.163	0.045	0.040	0.978	0.994	0.986
w/o camera gradient	26.139	0.849	0.284	0.049	0.048	0.976	0.993	0.985
w/o bilateral grid	29.995	0.936	0.169	0.058	0.045	0.960	0.994	0.977
Ours full	32.032	0.940	0.163	0.041	0.038	0.985	0.995	0.990

Best results are highlighted as 1st, 2nd.

in terms of geometric quality. Oppositely, disabling the non-planar constraint causes all Gaussian primitives to be supervised by estimated plane parameters. As shown in Fig. 6(c) and 6(d), the rendering and geometric quality in non-planar regions are negatively affected by the inaccuracy in reference point cloud measurements, shows the enhancement of our full approach in detail structures.

2) Occupation-aware Global Density Control: Disabling the occupation prior allows the Gaussian primitives to be located and grown in free-space. As shown in Fig. 6(f), while supervised by over-exposed images in real-world datasets, photometric-guided training of Gaussian primitives tend to produce large blobs for fitting these over-exposure thus causes artifacts. In contrast, our approach restricts all Gaussians in occupied voxels thus eliminates floaters in free-space, which is shown in Fig. 6(e).

3) Uncertainty in real-world datasets: We consider two types of uncertainties in real-world scenes: camera trajectory



(a) w/ planar constraint (b) w/o planar constraint (c) w/ non-planar constraint (d) w/o non-planar constraint (e) w/ occ. constraint (f) w/o occ. constraint (g) w/ camera gradient (h) w/o camera gradient (i) w/ bilateral grid (j) w/o bilateral grid

Fig. 6. Ablation study on the effect of major optimization components.

errors and photometric variation across views. As shown in Tab. IV and Fig. 6(h), disabling the camera pose optimization introduces pose uncertainty into optimization process thus causes artifacts. It is noteworthy that geometric quality is only slightly decreased compared to our full approach, which shows the robustness of our geometric regularization strategy and, in contrast, the sensitivity of other state-of-art Gaussian-based methods on pose uncertainty. By enabling the bilateral grid guided optimization, our approach tend to learn a average appearance from input image set which is disentangled from per-view exposure. We remove the bilateral grid for inference to ensure a multi-view consistent rendering, results in a biased appearance compared to the training view, as shown Fig. 6(i). Refer to Tab. IV, our full approach enjoys better quality in terms of rendering and geometry compared to the ablation, shows the significance of this component in real-world experiments.

V. APPLICATIONS WITH GPGS

GPGS provides a reconstruction result with spatially aligned high-fidelity rendered images and mesh model, en-

abling the construction of a digital-twin simulation environment for embodied agents. We develop software utilities using Open 3D Engine[2] to build the digital-twin, which utilizes meshes as collision model and Gaussian maps for sensor data synthesis.

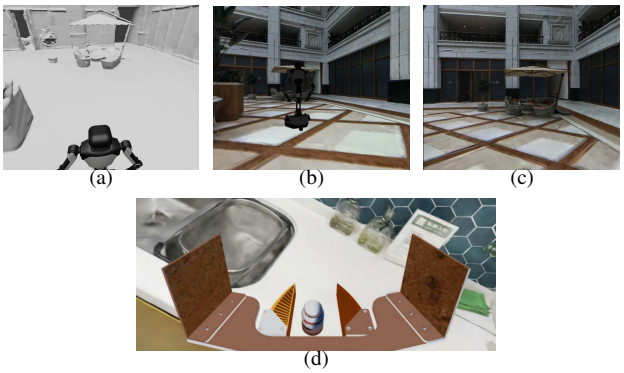


Fig. 7. Simulation environment built from GPGS. (a) The humanoid robot runs on a surface mesh. (b) User observes the robot moving in rendered Gaussian map from third-view. (c) The robot observes rendered images from Gaussians. (d) Training of grasping policies using the result of GPGS.

Figs. 7(a)-7(c) show an navigation demo developed using the outputs of GPGS. While the humanoid robot physically operates on the surface provided by mesh, it observes the high-fidelity RGB and depth images from Gaussian map, enables a visual-guided path planning. Fig. 7(d) shows the training scene of grasping policies using GPGS as background model. The photorealistic observations and accurate geometry provided by GPGS allows the policies to be trained in an environment which exhibits high fidelity to real-world conditions, thus enhances training efficiency and reduces sim-to-real gap.

VI. Conclusions

In this paper, we proposed a novel approach that directly regulates Gaussians in 3D space using Geometric Priors extracted from the voxelized point cloud map. We design a divide-and-conquer strategy and leverage different geometric priors to regulate Gaussians located in the planar and non-planar voxels separately. It leads to improvements in geometric performance. Additionally, we introduce an occupancy-aware density control method to restrict all Gaussians within occupied voxels. It reduces artifacts thus improve visual appearance. We extensively compare our method with state-of-the-art approaches on real-world datasets. Our method can concurrently achieve superior visual appearance and higher geometric accuracy.

References

[1] P. Liu, Y. Orru, C. Paxton, N. M. M. Shafiullah, and L. Pinto, "Ok-robot: What really matters in integrating open-knowledge models for robotics," *arXiv preprint: 2401.12202*, 2024.

[2] N. Yokoyama, S. Ha, D. Batra, J. Wang, and B. Bucher, "Vlfm: Vision-language frontier maps for zero-shot semantic navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2024.

[3] H. Zhou, L. Lin, J. Wang, Y. Lu, D. Bai, B. Liu, Y. Wang, A. Geiger, and Y. Liao, "Hugsim: A real-time, photo-realistic and closed-loop simulator for autonomous driving," *arXiv preprint: 2412.01718*, 2024.

[4] H. Zhou, J. Shao, L. Xu, D. Bai, W. Qiu, B. Liu, Y. Wang, A. Geiger, and Y. Liao, "Hugs: Holistic urban 3d scene understanding via gaussian splatting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2024.

[5] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: representing scenes as neural radiance fields for view synthesis," *Commun. ACM,*, 2021.

[6] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *TOG*, 2023.

[7] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2d gaussian splatting for geometrically accurate radiance fields," in *ACM SIGGRAPH 2024 Conference Papers*, 2024.

[8] P. Dai, J. Xu, W. Xie, X. Liu, H. Wang, and W. Xu, "High-quality surface reconstruction using gaussian surfels," in *ACM SIGGRAPH 2024 Conference Papers*, 2024.

[9] D. Chen, H. Li, W. Ye, Y. Wang, W. Xie, S. Zhai, N. Wang, H. Liu, H. Bao, and G. Zhang, "Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction," *arxiv preprint: 2406.06521*, 2024.

[10] Z. Yu, T. Sattler, and A. Geiger, "Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes," *ACM Trans. Graph.*, 2024.

[11] C. Yan, D. Qu, D. Xu, B. Zhao, Z. Wang, D. Wang, and X. Li, "Gs-slam: Dense visual slam with 3d gaussian splatting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2024.

[12] H. Matsuki, R. Murai, P. H. J. Kelly, and A. J. Davison, "Gaussian Splatting SLAM," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2024.

[13] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat, track & map 3d gaussians for dense rgb-d slam," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2024.

[14] S. Hong, J. He, X. Zheng, and C. Zheng, "Liv-gaussmap: Lidar-inertial-visual fusion for real-time 3d radiance field map rendering," *IEEE Robot. Autom. Lett.*, 2024.

[15] H. Zhao, W. Guan, and P. Lu, "Lvi-gs: Tightly-coupled lidar-visual-inertial slam using 3d gaussian splatting," *arXiv preprint: 2411.02703*, 2024.

[16] X. Lang, L. Li, C. Wu, C. Zhao, L. Liu, Y. Liu, J. Lv, and X. Zuo, "Gaussian-lic: Real-time photo-realistic slam with gaussian splatting and lidar-inertial-camera fusion," *arXiv preprint: 2404.06926*, 2024.

[17] J. Cui, J. Cao, F. Zhao, Z. He, Y. Chen, Y. Zhong, L. Xu, Y. Shi, Y. Zhang, and J. Yu, "Letsgo: Large-scale garage modeling and rendering via lidar-assisted gaussian primitives," *ACM Trans. Graph.*, 2024.

[18] C. Jiang, R. Gao, K. Shao, Y. Wang, R. Xiong, and Y. Zhang, "Li-gs: Gaussian splatting with lidar incorporated for accurate large-scale reconstruction," *IEEE Robot. Autom. Lett.*, 2025.

[19] K. Goel, N. Michael, and W. Tabib, "Probabilistic point cloud modeling via self-organizing gaussian mixture models," *IEEE Robot. Autom. Lett.*, 2023.

[20] Z. Liu and F. Zhang, "Balm: Bundle adjustment for lidar mapping," *IEEE Robot. Autom. Lett.*, 2021.

[21] C. Zheng, W. Xu, Z. Zou, T. Hua, C. Yuan, D. He, B. Zhou, Z. Liu, J. Lin, F. Zhu, Y. Ren, R. Wang, F. Meng, and F. Zhang, "Fast-livo2: Fast, direct lidar–inertial–visual odometry," *IEEE Trans. Robot.*, 2025.

[22] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanaes, "Large scale multi-view stereopsis evaluation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014.

[23] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2022.

[24] V. Ye, R. Li, J. Kerr, M. Turkulainen, B. Yi, Z. Pan, O. Seiskari, J. Ye, J. Hu, M. Tancik, and A. Kanazawa, "gsplat: An open-source library for gaussian splatting," *arXiv preprint: 2409.06765*, 2024.

[25] Y. Wang, C. Wang, B. Gong, and T. Xue, "Bilateral guided radiance field processing," *ACM Trans. Graph.*, 2024.

[26] Y. Tao, M. Ángel Muñoz-Bañón, L. Zhang, J. Wang, L. F. T. Fu, and M. Fallon, "The oxford spires dataset: Benchmarking large-scale lidar-visual localisation, reconstruction and radiance field methods," *arXiv preprint: 2411.10546*, 2024.

[27] V. S. Aleksandr, H. Dirk, and T. Sebastian, "Generalized-icp," in *Proc. Robot.: Sci. Syst.*, 2009.

[2]https://o3de.org/