This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Outdoor Monocular SLAM with Global Scale-Consistent 3D Gaussian Pointmaps

Chong Cheng[1*] Sicheng Yu[1*] Zijian Wang[1] Yifan Zhou[1] Hao Wang[1†]

[1]The Hong Kong University of Science and Technology (Guangzhou)

ccheng735@connect.hkust-gz.edu.cn yusch@mail2.sysu.edu.cn haowang@hkust-gz.edu.cn

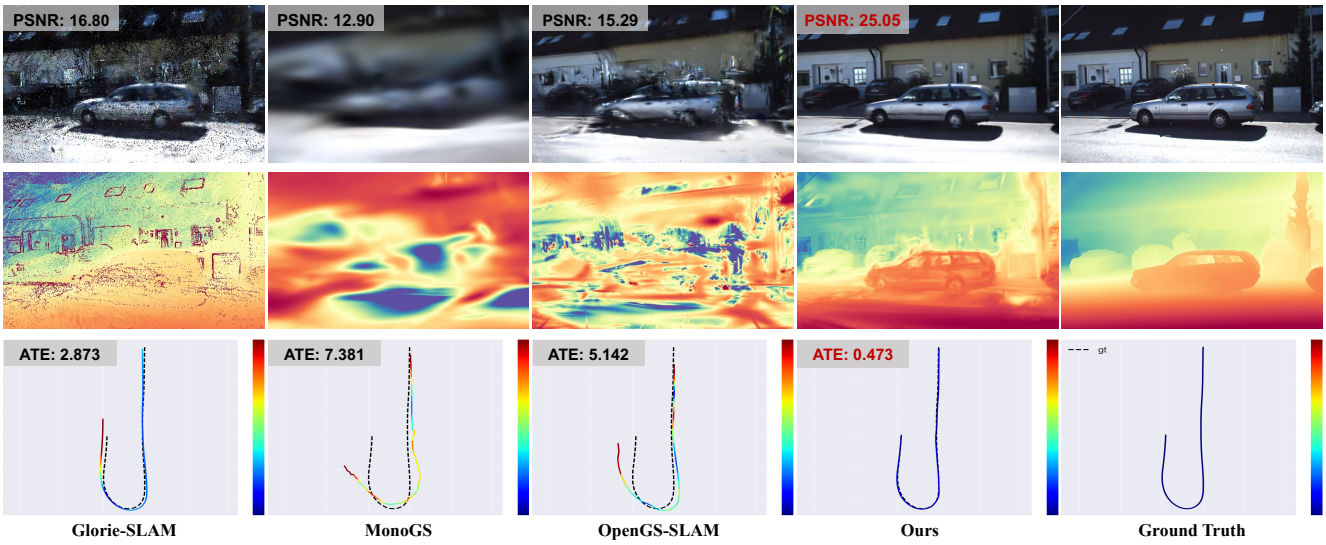


Figure 1. **Localization and novel view synthesis results on KITTI.** Our method S3PO-GS maintains robust tracking and high-quality novel view synthesis even in cases of large-angle turns. This is achieved through our self-consistent 3DGS pointmap tracking and the patch-based pointmap dynamic mapping module.

Abstract

3D Gaussian Splatting (3DGS) has become a popular solution in SLAM due to its high-fidelity and real-time novel view synthesis performance. However, some previous 3DGS SLAM methods employ a differentiable rendering pipeline for tracking, ***lack geometric priors*** *in outdoor scenes. Other approaches introduce separate tracking modules, but they accumulate errors with significant camera movement, leading to* ***scale drift****. To address these challenges, we propose a robust RGB-only outdoor 3DGS SLAM method: S3PO-GS. Technically, we establish a self-consistent tracking module anchored in the 3DGS pointmap, which avoids cumulative scale drift and achieves more precise and robust tracking with fewer iterations. Additionally, we design a patch-based pointmap dynamic mapping module, which introduces geometric priors while avoiding scale ambiguity. This significantly enhances tracking accuracy and the quality of scene reconstruction, making it particularly suitable for complex outdoor environments. Our experiments on the Waymo, KITTI, and DL3DV datasets demonstrate that S3PO-GS achieves state-of-the-art results in novel view synthesis and outperforms other 3DGS SLAM methods in tracking accuracy. Project page:* https://3dagentworld.github.io/S3PO-GS/.

1. Introduction

Visual Simultaneous Localization and Mapping (SLAM), a core problem in fields like autonomous driving, robotics, and virtual reality (VR), has received substantial attention. Within this area, 3D scene representation has become a primary research focus, resulting in the development of numerous sparse [3, 16, 24, 25] and dense [26, 27, 42] representation methods, which advance localization accuracy. However, these methods still face significant challenges in

* Equal contribution.
† Corresponding author.

novel view synthesis (NVS) capabilities.

Given the photorealistic visual effects offered by 3D Gaussian Splatting (3DGS) [14] scene representations, recent research has focused on integrating 3DGS with SLAM [9, 11, 13, 21, 38, 41]. However, existing 3DGS SLAM methods still face two key challenges in outdoor RGB-only scenarios: **lack of geometric priors** and **scale drift** issues.

On one hand, some previous RGB-only 3DGS SLAM methods, such as those proposed by [21], perform pose estimation via differentiable rendering pipelines. However, this approach **lacks geometric priors** and struggles with convergence in complex environments, particularly in outdoor settings, where the model is prone to getting stuck in local minima.

On the other hand, to enforce geometric constraints, some methods [11, 41, 43] introduce independent tracking modules and pre-trained models to supplement geometric information, enhancing the robustness of pose estimation. Yet, this strategy requires maintaining scale alignment between external modules and the 3DGS map. In scenarios with large rotations and displacements, accumulated errors can easily lead to **scale drift** in SLAM system, degrading subsequent pose estimation and map reconstruction quality.

To address the challenges above, we propose a robust 3D Gaussian Splatting SLAM method—**S3PO-GS**. Our approach leverages pre-trained pointmap models to compensate for the lack of geometric priors in RGB-only scenarios. By anchoring 3DGS-rendered pointmaps, we establish 2D-3D correspondences, enabling scale self-consistent pose estimation. Through a patch-based design, we align the scale of the pre-trained pointmap with the current 3DGS scene. This allows us to incorporate geometric priors while effectively avoiding the issue of scale drift.

Technically, we first design a self-consistent 3DGS pointmap tracking module that estimates poses through pixel-wise 2D-3D correspondences between the input frame and 3DGS-rendered pointmap. The pre-trained model serves solely as a bridge for correspondence without participating in the pose estimation, inherently avoiding scale alignment issues. Combined with the 3DGS differentiable pipeline to optimize poses, even in complex outdoor environments, this approach can achieve more accurate and robust tracking with only 10% of the iterations required.

Furthermore, to address the lack of geometric priors in monocular SLAM, we design a patch-based pointmap dynamic mapping. This approach employs a patch-scale alignment algorithm to achieve local geometric calibration between the pre-trained pointmap and the 3DGS scene. A dynamic pointmap replacement mechanism is designed to reduce reconstruction errors. These strategies introduce geometric priors and resolve the issue of scale ambiguity, enabling high-quality scene mapping.

Experiments on Waymo [33], KITTI [8], and DL3DV [19] datasets show that S3PO-GS outperforms existing 3DGS SLAM methods. It also sets new benchmarks in tracking accuracy and novel view synthesis. Our main contributions include:

- We propose a self-consistent 3DGS pointmap tracking module that introduces priors while avoiding scale alignment issues, enhancing tracking accuracy and robustness with a significant reduction in iterations.
- Our proposed patch-based pointmap dynamic mapping module leverages a pre-trained model to dynamically adjust the 3DGS pointmap while mitigating scale ambiguities, significantly improving scene reconstruction quality.
- Evaluations on multiple datasets demonstrate that our method establishes state-of-the-art performance in tracking accuracy and novel view synthesis within the 3DGS SLAM framework.

2. Related work

2.1. Classical SLAM

Classical SLAM methods commonly use sparse feature representations. For example, methods in the ORB-SLAM series [3, 24, 25] combine the FAST corner detector and BRIEF descriptor, tracking and updating only a small number of key points. Similarly, SIFT [20] and SURF [1] also rely on feature points for camera pose estimation. Based on this efficient feature tracking idea, PTAM [15] first parallelizes tracking and mapping, marking the beginning of real-time keypoint-based SLAM research. However, the resulting maps are typically sparse, serving primarily for navigation and localization rather than detailed scene modeling.

Dense SLAM [26, 27, 42] generates detailed 3D maps, contrasting with sparse methods focused on pose estimation, and is well-suited for augmented reality and robotics. It includes frame-centered approaches, which are efficient but struggle with global consistency, and map-centered approaches using voxel grids or point clouds to enhance tracking and system compactness [28, 37]. Recent advancements like iMAP [32] integrate neural networks for enhanced detail capture but face significant computational demands, limiting real-time applications. GlORIE-SLAM [42] utilizes a flexible neural point cloud representation, improving real-time performance without the need for costly backpropagation. However, it still does not achieve photorealistic novel view synthesis.

2.2. NeRF-based and 3DGS-based SLAM

NeRF [22] uses a Multi-Layer Perceptron (MLP) to sample along viewing rays and generate high-quality novel view synthesis via volume rendering, significantly outperforming traditional sparse SLAM methods in reconstruction accuracy [12, 31, 34, 40]. In the SLAM framework, NeRF optimizes the MLP using multi-view geometric information

to achieve high-fidelity scene representation [30, 44, 45]. However, its long training time limits applicability in real-time SLAM [7]. Recent research introduce explicit structures like multi-resolution voxel grids or hash encodings to improve rendering speed and efficiency [10, 23], yet they still struggle with achieving real-time rendering.

3DGS-based [14] SLAM methods [9, 13, 21, 38, 41] employ explicit 3D Gaussian representations for modeling and rendering scenes. Compared to traditional point cloud representations, they enable real-time scene reconstruction and provide high-fidelity view synthesis [4]. However, this method lacks geometric priors and struggles in outdoor environments with only RGB, requiring numerous iterations and often failing to converge. Additionally, some 3DGS-SLAM methods [11, 41, 43] decouple camera tracking from scene modeling, using an independent model for pose estimation while relying on a 3D Gaussian distribution for reconstruction. However, these methods require maintaining a scale factor, which easily accumulates scale errors in outdoor scenes with large angular movements. This leads to scale drift, degrading both localization accuracy and reconstruction quality.

We propose a 3DGS-based approach that enables efficient, accurate, and robust tracking with RGB-only input, while achieving high-fidelity novel view synthesis.

3. Method

Our method comprises three main parts: 3D Gaussian Splatting (3DGS), Self-Consistent 3DGS Pointmap Tracking and Patch-based Pointmap Dynamic Mapping, as illustrated in Fig. 2. These components together form our 3DGS SLAM pipeline.

3.1. 3D Gaussian Splatting

We employ a 3DGS [14] scene representation, where the scene is modeled using a set of Gaussians centered at points μ, each defined by its covariance matrix Σ:

$$\Sigma = RSS^TR^T, \tag{1}$$

where R is a rotation matrix and S is a scaling matrix. By projecting these 3D Gaussians onto a 2D plane and applying tile-based rasterization, we achieve efficient and differentiable rendering on a CUDA pipeline. Unlike the original 3DGS method, we do not employ spherical harmonics; instead, we directly compute the color of pixel x' using:

$$C(x') = \sum_{i\in N} c_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{2}$$

where N is the set of Gaussians affecting x', and α_i is opacity. This method allows our SLAM system to optimize all Gaussian parameters, including position, rotation, scale, opacity, and color.

3.2. Self-Consistent 3DGS Pointmap Tracking

3.2.1. Pointmap Anchored Pose Estimation

Previous 3DGS SLAM methods [41], relying on pre-trained modules for direct pose estimation, often encounter cumulative scale drift despite scale alignment techniques. To address this challenge, particularly in outdoor scenes with large rotation angles and displacements, we propose a novel pose tracking method. Inspired by visual localization [2, 29, 35, 39], we introduce a differentiable pointmap rendering pipeline within the 3DGS framework. This pipeline captures normalized 3D shape and viewpoint information through 3DGS-rendered pointmaps, establishing a basis for a scale self-consistent tracking module.

Our core innovation lies in estimating poses directly from the 3DGS scene's own scale, through the pixel-to-point 2D-3D correspondence between the 3DGS-rendered pointmaps and new input frame. Notably, the pre-trained pointmap model [18, 35] is used solely to establish these correspondences and does not directly contribute to the estimation process.

Specifically, starting from the adjacent keyframe I_{ak}, we leverage the 3DGS rasterization mechanism to build a differentiable pointmap rendering pipeline. Using the adjacent keyframe's viewpoint $\mathbf{T}_{ak} \in SE(3)$, we render a depth map $D_{ak} \in \mathbb{R}^{W\times H}$. The depth value at pixel (i, j) is computed by alpha-blending Gaussian primitives along the ray:

$$D_{ak}(i,j) = \sum_{k\in N} z_k\alpha_k \prod_{l=1}^{k-1}(1-\alpha_l), \tag{3}$$

where z_k is the distance from point u_i to the camera center, N denotes the set of Gaussian primitives along the ray sorted by z_k, and α_k represents the opacity. The rendered pointmap X^r_{ak} is derived from the depth map D_{ak}, by applying the inverse of the camera intrinsic matrix K to each pixel's depth-scaled coordinates:

$$X^r(i,j) = K^{-1}\begin{bmatrix} iD(i,j) \\ jD(i,j) \\ D(i,j)\end{bmatrix} = \begin{bmatrix} \frac{1}{f_x} & 0 & -\frac{c_x}{f_x} \\ 0 & \frac{1}{f_y} & -\frac{c_y}{f_y} \\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix} iD(i,j) \\ jD(i,j) \\ D(i,j)\end{bmatrix}, \tag{4}$$

simplifying this multiplication yields:

$$X^r(i,j) = \begin{bmatrix} \frac{iD(i,j)-c_xD(i,j)}{f_x} \\ \frac{jD(i,j)-c_yD(i,j)}{f_y} \\ D(i,j)\end{bmatrix}, \tag{5}$$

where f_x,f_y are the focal lengths, and c_x,c_y are the optical centers of camera. From this, we construct the rendered pointmap X^r_{ak}, which store viewpoint and shape information. Using D_{ak} as a bridge, we establish point-to-pixel correspondences between the pointmap and image:

$$X^r_{ak} \leftrightarrow D_{ak} \leftrightarrow I_{ak}, \tag{6}$$

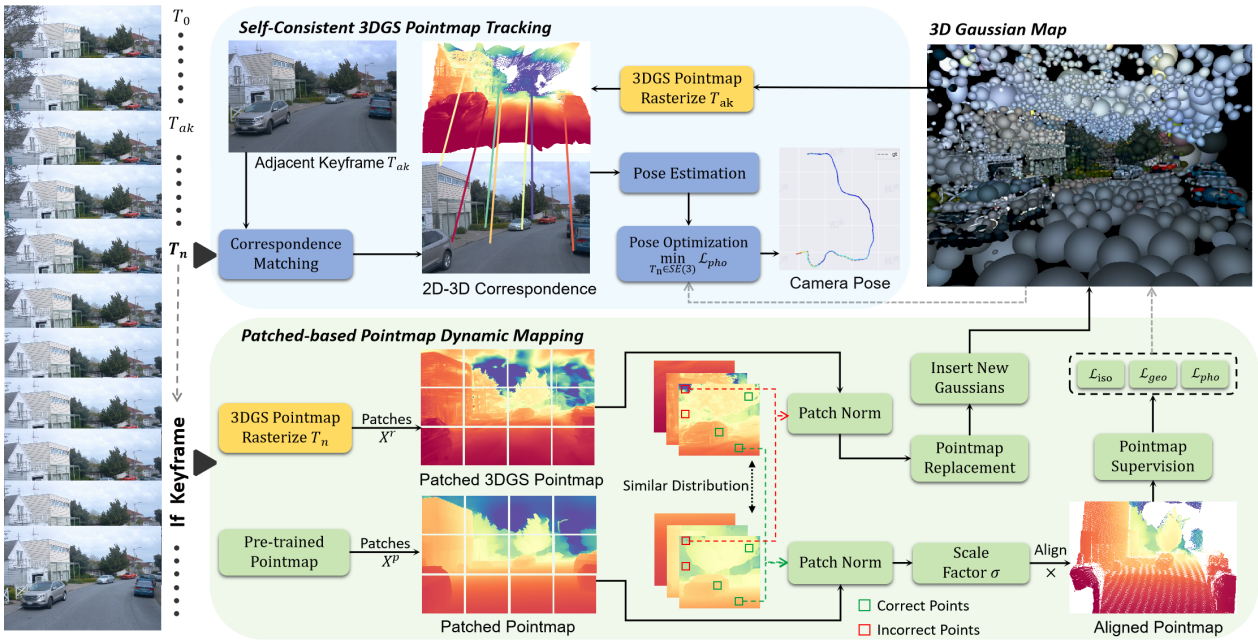


Figure 2. **S3PO-GS pipeline for SLAM.** The system begins by initializing a 3D Gaussian map (optimizing MASt3R's pointmap for 1000 steps). For new input frame T_n, we rasterize the 3DGS pointmap of the adjacent keyframe T_{ak}, match it with the input image, and establish 2D-3D correspondences to estimate scale self-consistent pose. The estimated pose is further refined using photometric loss. If T_n is selected as keyframe, we obtain its rendered pointmap X^r and pre-trained pointmap X^p, then crop both into patches with similar distributions. After patch normalization, the correct points are selected to compute a scaling factor, which is then used to adjust X^p. Once the incorrect points are replaced, X^r is used to insert new Gaussians. Finally, the aligned pre-trained pointmap is used to jointly optimize the 3D Gaussian map, enabling precise and robust localization and mapping.

where the scale of X^r_{ak} originates from the 3DGS map, independent of external dependencies.

Meanwhile, we input the current frame image I_n and the adjacent keyframe image I_{ak} into the pre-trained model [18, 35] to obtain two sets of pointmaps X^p_{ak}, X^p_n at the same scale, along with confidence scores c. Based on this, we establish pointmap correspondences $X^p_{ak} \leftrightarrow X^p_n$ by minimizing the pointmap distance and filter out low-confidence points [18]:

$$(i,j) \leftrightarrow (u,v) \;\Big|\; \underset{(i,j),(u,v)}{\arg\min} \sum_{\substack{c(i,j)\geq t \\ c(u,v)\geq t}} \|X^p_{ak}(i,j) - X^p_n(u,v)\|, \tag{7}$$

where $X^p(u,v) \in \mathbb{R}^3$ denotes the 3D coordinate at pixel (u,v), t is used to filter low-confidence points. Since the generated pointmaps are per-pixel, we also obtain pixel correspondences between images $I_{ak} \leftrightarrow I_n$. Propagating from the previously established Eq. (6), we construct 2D-3D correspondences between the rendered pointmap of the last keyframe and the current frame image:

$$X^r_{ak} \leftrightarrow I_{ak} \leftrightarrow I_n \tag{8}$$

Using these correspondences, we estimate the relative pose $\mathbf{T}^{\text{rel}}_n$ between the current frame and the adjacent keyframe via RANSAC [6] and PnP [17]. The key advantage here is that the 3D coordinates in X^r_{ak} come directly from the 3DGS model, ensuring strict scale consistency with the reconstructed scene. Consequently, the PnP solution inherently preserves the correct scale.

Finally, the pose of current frame n is computed as:

$$\mathbf{T}_n = \mathbf{T}^{\text{rel}}_n \, \mathbf{T}_{ak}. \tag{9}$$

3.2.2. Pose Optimization

To achieve precise camera pose, we leverage the 3DGS differentiable rendering pipeline to generate images and optimize the pose T_n by minimizing the photometric loss:

$$L_{\text{pho}} = \left\| I(\mathcal{G}, T) - \bar{I} \right\|_1, \tag{10}$$

where I represents a per-pixel differentiable rendering function, generating images through Gaussian $\mathcal{G}$ and camera pose T, and $\bar{I}$ is the ground truth image.

To avoid the overhead of automatic differentiation, similar to [41], we linearize the camera pose $T \in SE(3)$ into its corresponding Lie algebra $\mathfrak{se}(3)$ and explicitly incorporate the gradient of T within the 3DGS CUDA pipeline:

$$\nabla_T L_{\text{pho}} = \frac{\partial L_{\text{pho}}}{\partial r} \cdot \left(\frac{\partial r}{\partial \mu_I} \cdot \frac{\partial \mu_I}{\partial T} + \frac{\partial r}{\partial \Sigma_I} \cdot \frac{\partial \Sigma_I}{\partial T} \right), \tag{11}$$

where r denotes the rasterization function, the derivatives of mean μ_I and covariance Σ_I are derived from [21].

In particular, to enhance accuracy and focus on details in pose optimization, we penalize non-edge and invalid region.

Based on the Self-Consistent 3DGS Pointmap Tracking, our proposed method enables more precise and robust tracking from 3DGS scene with fewer iterations.

3.3. Patch-based Pointmap Dynamic Mapping

In monocular RGB-only SLAM, the lack of geometric information leads to inaccurate scene reconstruction. One solution is to introduce monocular depth priors, but depth estimation suffers from scale drift across frames. This is particularly problematic in unbounded outdoor scenes, where complex environments cause increasing instability in scale. Some works [42, 43] align depth scales to sparse point clouds from independent tracking modules, but their performance is limited by point cloud quality and they do not form an end-to-end pipeline. Recent work [41] align scale to the initial frame by establishing correspondences between consecutive frames, but this introduces cumulative errors.

We address this issue by using pre-trained pointmap [18] as geometric prior and propose a patch-based method to dynamically align its scale to the current Gaussian scene. Through pointmap replacement, we achieve optimal Gaussian insertion at keyframes. With the aligned pre-trained pointmap, we perform geometric supervision optimization on the Gaussian scene.

3.3.1. Patch-Based Scale Alignment

After optimizing the current frame's pose, we rasterize a 3DGS pointmap X^r from the current viewpoint T_n and obtain the pre-trained pointmap X^p. The scale of the X^r is consistent with the scene scale but is usually less precise than the X^p. Our approach is to identify reliable pixels in the X^r as "correct points" and use them to calculate a scaling factor to align the X^p's scale to the scene.

However, finding "correct point" when there is a scale discrepancy is challenging, necessitating preliminary alignment of the two pointmaps. Direct normalization might lead to incorrect identification of "correct points" due to inconsistent value distributions or outliers in X^r. To address this, we segment the entire pointmap into small patches and select patches with similar distributions for normalization, ensuring accurate point selection from X^r.

We initially segments X^p and X^r into patches of size $P \times P$, then calculates the mean μ and standard deviation σ for each patch. Patches are selected for normalization if they satisfy $|\mu_r - \mu_p| < \delta_\mu \times \mu_p$ and $|\sigma_r - \sigma_p| < \delta_\sigma \times \sigma_p$. For these candidate patches, pointmap values are normalized:

$$X_N(x) = \frac{X(x) - \mu(X)}{\sigma(X)}, \tag{12}$$

Points satisfying $|X^r_N(x) - X^p_N(x)| < \epsilon_r$ are selected as "correct points" set CP, and scale factor is calculated:

$$\sigma' = \frac{\mu(X^r[CP])}{\mu(X^p[CP])}, \tag{13}$$

and applied to adjust X^p. This process is iterated until the scale factor stabilizes or the iteration limit is reached, eventually outputting the aligned $\hat{X}^p = \sigma \times X^p$ for pointmap replacement and supervision. If the number of "correct points" is insufficient, the scale factor estimation might be erroneous, introducing additional biases in scene reconstruction. In this case, we establish point correspondences with the pre-trained pointmap $\hat{X}^p_{ak}$ from adjacent keyframe and compute a remedial scale factor. Since $\hat{X}^p_{ak}$ is already aligned with the scene, it can serve as a reference. For detailed algorithm and pseudocode, see the supplementary material.

3.3.2. Pointmap Replacement

We insert new Gaussians into the scene at keyframes. To minimize scale drift, we initialize the Gaussians based on the rendered pointmap X^r from the current frame. However, the rendered pointmap often contains poorly reconstructed areas, and using it directly may introduce additional errors. We use the aligned pre-trained pointmap $\hat{X}^p$ as a reference and replace the "incorrect points" in X^r:

$$\hat{X}^r(x) = \begin{cases} X^r(x), & \text{if } |X^r(x) - \hat{X}^p(x)| \le \epsilon_m \times \hat{X}^p(x) \\ \hat{X}^p(x), & \text{if } |X^r(x) - \hat{X}^p(x)| > \epsilon_m \times \hat{X}^p(x) \end{cases}. \tag{14}$$

We perform random sparse downsampling on $\hat{X}^r(x)$ to effectively control the number of 3D Gaussians, ensuring high-quality mapping while reducing processing time.

3.3.3. Map Optimization with Pointmap Supervision

To achieve efficient viewpoint coverage and introduce multi-view constraints, inspired by [5, 21], we jointly refine camera poses and the Gaussian map within the current local keyframe window $\mathcal{W}$. See the supplementary material for details.

To improve the scene geometry, we introduce a pointmap-based geometry loss:

$$L_{geo} = \left\| X^r - \hat{X}^p \right\|_1. \tag{15}$$

To mitigate excessive stretching of the ellipsoids and reduce artifacts, we employ isotropic regularization [21]:

$$L_{\text{iso}} = \sum_{i=1}^{|\mathcal{G}|} \left\| s_i - \tilde{s}_i \cdot \mathbf{1} \right\|_1, \tag{16}$$

where $\tilde{s}_i$ denotes the mean of the scaling s_i. Combining the photometric loss, geometry loss, and isotropic regularization, the map optimization task can be summarized as:

$$\min_{\substack{T_k \in SE(3) \\ \forall k \in \mathcal{W}}, \mathcal{G}} \sum_{\forall k \in \mathcal{W}} \alpha L^k_{\text{pho}} + (1-\alpha) L^k_{geo} + \lambda_{\text{iso}} L_{\text{iso}}. \tag{17}$$

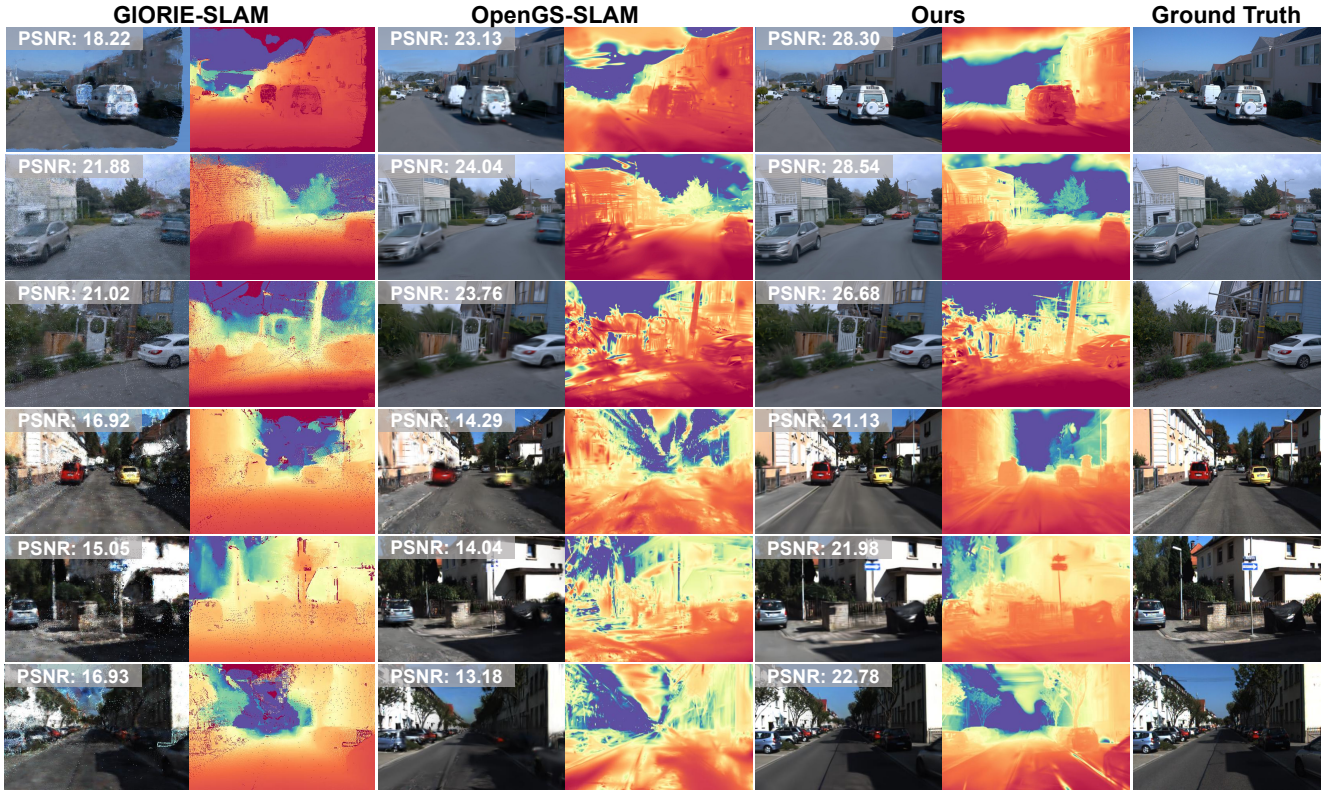


Figure 3. **Novel View Synthesis Results on Waymo (top three rows) and KITTI (bottom three rows) scenes, including Rendered RGB and Depth Maps.** Our method produces high-fidelity images that capture intricate details of vehicles, streets, and buildings. The rendered depth maps are more accurate in regions with complex depth variations, such as tree branches and roadside vehicles.

Method	Waymo [33]				KITTI [8]				DL3DV [19]			
	ATE↓	PSNR↑	SSIM↑	LPIPS↓	ATE↓	PSNR↑	SSIM↑	LPIPS↓	ATE↓	PSNR↑	SSIM↑	LPIPS↓
NeRF-SLAM [30]	97.36	10.12	0.597	0.781	63.55	11.33	0.441	0.774	4.41	12.21	0.505	0.517
NICER-SLAM[45]	19.59	12.22	0.622	0.726	18.55	12.69	0.450	0.701	3.57	14.19	0.551	0.507
Photo-SLAM [11]	19.95	17.73	0.741	0.674	17.62	15.39	0.523	0.674	2.78	16.74	0.560	0.499
GlORIE-SLAM [42]	**0.589**	18.83	0.702	0.572	1.134	15.49	0.594	0.684	0.492	16.20	0.669	0.515
MonoGS [21]	8.529	21.80	0.780	0.577	9.493	14.78	0.486	0.759	0.274	24.99	0.766	0.322
OpenGS-SLAM [41]	0.839	23.99	0.800	0.434	3.224	15.61	0.495	0.492	0.141	24.75	0.788	0.192
Ours	<u>0.622</u>	**26.73**	**0.845**	**0.360**	**1.048**	**20.03**	**0.646**	**0.398**	**0.032**	**29.97**	**0.893**	**0.108**

Table 1. **Comparison of all methods on three datasets.** ATE RMSE (m) for tracking, and PSNR, SSIM, LPIPS for Novel View Synthesis. Best results are in **bold**, second-best in <u>underlined</u>. Our method achieves NVS SOTA performance across all datasets, with the best tracking accuracy on KITTI and DL3DV, and comparable tracking accuracy to GlORIE-SLAM on Waymo.

4. Experiments

4.1. Implementation and Experiment Setup

Datasets. We conduct experiments on the Waymo Open Dataset [33], KITTI Dataset [8], and DL3DV Dataset [19] to evaluate tracking accuracy and novel view synthesis performance in outdoor environment. Specifically, we select nine 200-frame sequences from Waymo, eight 200-frame sequences from KITTI, and three 300-frame sequences from DL3DV. The selected scenes are all static and feature significant camera viewpoint changes.

Metrics. To assess novel view synthesis performance, we use PSNR, SSIM [36], and LPIPS metrics, calculated on frames excluding keyframes (i.e., training frames). For tracking accuracy, we use ATE RMSE (m) for evaluation.

Baseline Methods. We compare our method with SLAM approaches that support monocular RGB-only input and novel view synthesis. These include NeRF-based meth-

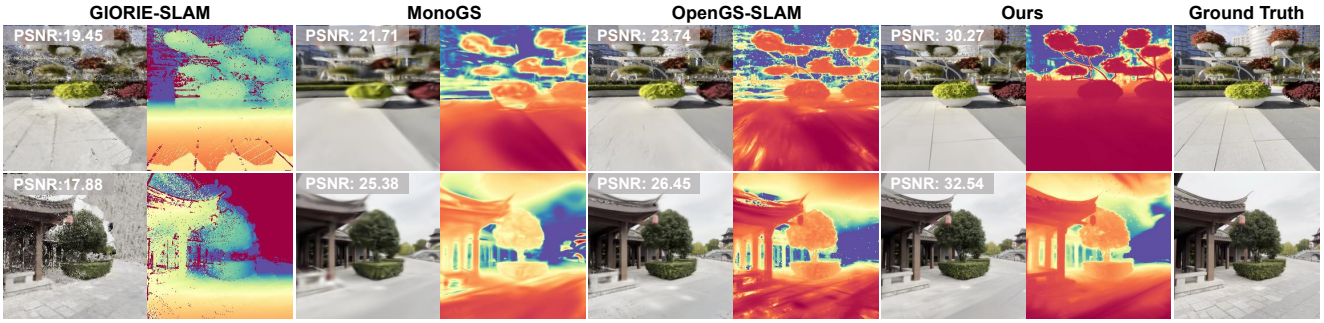


Figure 4. **Novel View Synthesis Results on DL3DV scenes, including Rendered RGB and Depth Maps.** Our method renders clearer images with better reconstruction of details such as flowerbeds, eaves, and lanterns. The depth maps more accurately capture interwoven objects like columns and show greater stability and smoothness in ground regions.

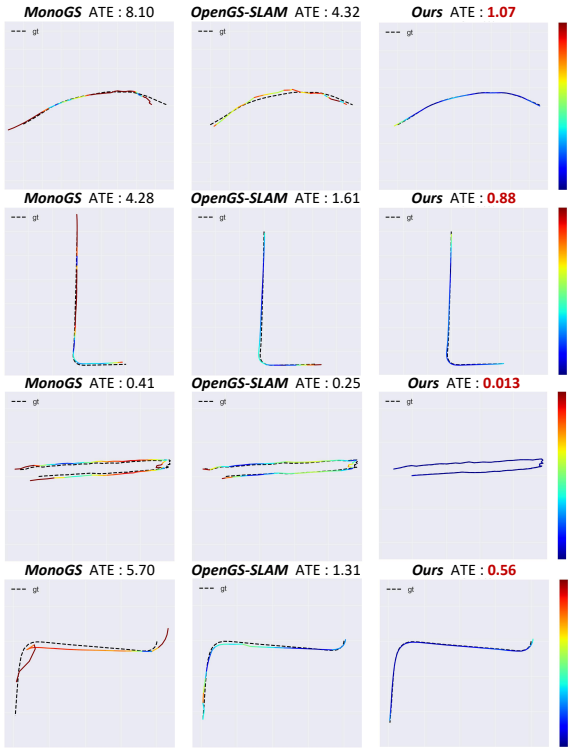


Figure 5. **Comparison of Tracking Trajectories with OpenGS-SLAM and MonoGS.** Under large viewpoint changes, MonoGS struggles to track, while OpenGS-SLAM exhibits instability. In contrast, our method achieves superior robustness.

ods NeRF-SLAM [30] and NICER-SLAM [45], implicit encoding point cloud-based GlORIE-SLAM [42], 3DGS-based methods MonoGS [21] and Photo-SLAM [11], and OpenGS-SLAM [41], which is specifically designed for outdoor environments.

Implementation Details. Experiments are conducted on an NVIDIA RTX A6000 GPU. Similar to MonoGS, Gaussian attributes and camera pose rasterization and gradient computations are implemented using CUDA. The remainder of the SLAM pipeline is developed in PyTorch. More details are provided in the supplementary material.

Method	Iterations				
	5	15	30	50	100
MonoGS	12.6	9.87	8.95	2.98	1.70
OpenGS-SLAM	4.17	2.52	1.10	0.85	0.80
Ours	**0.55**	**0.55**	**0.51**	**0.49**	**0.46**

Table 2. **Tracking error (ATE RMSE) on Waymo_405841 under different iteration counts.**

Method	ATE	PSNR	SSIM	LPIPS
MonoGS+MASt3R	3.84	23.09	0.792	0.439
Ours	**0.62**	**26.73**	**0.845**	**0.360**

Table 3. **Comparison with direct incorporation of pre-trained pointmap information in 3DGS-based SLAM (e.g., MonoGS).** We report the average results on Waymo.

4.2. Experiment Results

Camera Tracking. Table 1 presents the tracking results on three datasets. Our method achieves state-of-the-art performance on the KITTI and DL3DV datasets, and comparable performance to GlORIE-SLAM on the Waymo dataset. While GlORIE-SLAM tracks the camera by leveraging the relationship between consecutive image frames, our method, like MonoGS and OpenGS-SLAM, performs visual localization from a single image to the scene for pose estimation, a more challenging task. Despite this, our method achieves comparable tracking accuracy to GlORIE-SLAM on the Waymo and KITTI datasets, and significantly outperforms it on DL3DV. Compared to OpenGS-SLAM, which is also 3DGS-based and designed specifically for outdoor environments, we reduce tracking error by **67.5%** on KITTI and by **77.3%** on DL3DV.

Figure 5 shows the comparison of tracking trajectories. In the presence of significant camera displacement and rotation, both MonoGS and OpenGS-SLAM exhibit unsta-

Method	Iterations				
	5	15	30	50	100
Ours (w/o PAPE)	16.9	8.57	5.61	4.76	3.48
Ours	**0.55**	**0.55**	**0.51**	**0.49**	**0.46**

Table 4. **Ablation study of Pointmap Anchored Pose Estimation (PAPE) module on Waymo_405841.** We present the tracking errors across different iteration counts.

Method	ATE	PSNR	SSIM	LPIPS
w/o Pose Optimization	1.79	24.45	0.805	0.376
w/o Scale Alignment	3.50	23.49	0.784	0.411
w/o Point Replacement	1.35	25.59	0.825	0.406
w/o Geometry Loss $\mathcal{L}_{geo}$	3.73	25.70	0.829	0.402
Ours	**0.62**	**26.73**	**0.845**	**0.360**

Table 5. **Ablation study on key modules.** We report the average results on Waymo.

ble convergence and poor perception of displacement, while our method demonstrates superior accuracy and robustness. These results highlight the superiority of our method.

Novel View Synthesis. As shown in Tab. 1, our method achieves the best novel view synthesis performance across all three datasets. Compared to the current best 3DGS-based SLAM methods, PSNR is significantly improved: **+2.73** on Waymo, **+4.42** on KITTI, and **+4.98** on DL3DV.

Figures 3 and 4 show the rendered images and depth maps for the three datasets. For outdoor scenes, our method generates high-fidelity images with better reconstruction of vehicle, building, and street details. Additionally, our rendered depth maps are more accurate in regions with complex depth variations and exhibit more reasonable relative positioning between objects. This demonstrates our method's strong geometric understanding of outdoor scenes and reflects the stability of pointmap scale during training.

Pose Optimization Convergence. Table 2 shows the impact of different pose optimization iteration counts on tracking error in the Waymo scene. MonoGS fails to converge with fewer than 50 iterations, while OpenGS-SLAM shows a noticeable accuracy drop below 30 iterations. In contrast, our method achieves results comparable to 100 iterations with only 5 iterations, demonstrating the robustness of our self-consistent 3DGS pointmap tracking module.

4.3. Ablation Study

Self-Consistent 3DGS Pointmap Tracking. Table 4 shows that without the Pointmap Anchored Pose Estimation module, camera tracking fails to converge, especially with fewer optimization iterations, leading to a rapid drop in accuracy. The first row of Tab. 5 indicates that without pose optimization, both tracking and reconstruction performance degrade. This demonstrates that pose estimation ensures convergence in complex outdoor environments, while pose optimization further refines pose accuracy.



Figure 6. **Comparison with direct incorporation of pre-trained pointmap information in 3DGS-based SLAM (e.g., MonoGS).** Scale drift leads to noticeable geometric blurring.

Patch-based Pointmap Dynamic Mapping. As shown in Tab. 5, the absence of pointmap replacement leads to erroneous Gaussian insertions in keyframes, degrading both tracking and reconstruction performance. Removing $\mathcal{L}_{geo}$ results in a significant increase in tracking error due to the lack of geometric supervision, while reconstruction quality only slightly declines. This is because pointmap replacement provides geometric priors, allowing reasonable reconstruction quality even when relying solely on photometric loss. However, this compromises the system's displacement awareness. Removing patch-based scale alignment causes a substantial performance drop, as misaligned pre-trained pointmaps introduce incorrect supervision.

Pre-trained Pointmap Processing. Table 3 shows that directly incorporating pre-trained pointmap supervision into 3DGS SLAM (e.g., MonoGS) without our proposed pointmap processing modules results in significantly degraded performance. As seen in Fig. 6, scale drift causes substantial blurring. This demonstrates that pre-trained geometric priors alone are insufficient for handling outdoor scenes, and the proposed designs are the key to our method's success.

5. Conclusion

In this work, we introduce S3PO-GS, a monocular outdoor 3D Gaussian Splatting SLAM framework with a scale self-consistent pointmap, addressing the challenges of scale drift and lack of geometric priors in outdoor scenes. By using a self-consistent 3DGS pointmap tracking module, we reduce pose estimation iterations to 10% of traditional methods, achieving centimeter-level tracking accuracy on complex datasets like Waymo. A patch-based dynamic mapping mechanism, based on local patch matching, resolves monocular depth scale ambiguity and enhances reconstruction quality. Experiments show our method sets new benchmarks in tracking accuracy and novel view synthesis for 3DGS SLAM. Future work will explore loop closure and large-scale scene optimization to expand its application boundaries in outdoor SLAM.

Acknowledgment

This research is supported by the National Natural Science Foundation of China (No. 62406267), Guangzhou-HKUST(GZ) Joint Funding Program (Grant No.2025A03J3956 & Grant No.2023A03J0008), the Guangzhou Municipal Science and Technology Project (No. 2025A04J4070), and the Guangzhou Municipal Education Project (No. 2024312122).

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*, pages 404–417. Springer, 2006. 2

[2] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac - differentiable ransac for camera localization, 2018. 3

[3] Carlos Campos, Richard Elvira, Juan J. Gomez Rodriguez, Jose M. M. Montiel, and Juan D. Tardos. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6): 1874–1890, 2021. 1, 2

[4] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024. 3

[5] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. 5

[6] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 4

[7] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14346–14355, 2021. 3

[8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013. 2, 6

[9] Jiarui Hu, Xianhao Chen, Boyin Feng, Guanglin Li, Liangjing Yang, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Cg-slam: Efficient dense rgb-d slam in a consistent uncertainty-aware 3d gaussian field. *arXiv preprint arXiv:2403.16095*, 2024. 2, 3

[10] Tao Hu, Shu Liu, Yilun Chen, Tiancheng Shen, and Jiaya Jia. Efficientnerf efficient neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12902–12911, 2022. 3

[11] Huajian Huang, Longwei Li, Hui Cheng, and Sai-Kit Yeung. Photo-slam: Real-time simultaneous localization and photo-realistic mapping for monocular stereo and rgb-d cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21584–21593, 2024. 2, 3, 6, 7

[12] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17408–17419, 2023. 2

[13] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat track & map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21357–21366, 2024. 2, 3

[14] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2, 3

[15] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007. 2

[16] Henning Lategahn, Andreas Geiger, and Bernd Kitt. Visual slam for autonomous ground vehicles. In *2011 IEEE International Conference on Robotics and Automation*, pages 1732–1737. IEEE, 2011. 1

[17] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Ep n p: An accurate o (n) solution to the p n p problem. *International journal of computer vision*, 81:155–166, 2009. 4

[18] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024. 3, 4, 5

[19] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024. 2, 6

[20] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004. 2

[21] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18039–18048, 2024. 2, 3, 4, 5, 6, 7

[22] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020. 2

[23] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 3

[24] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 1, 2

[25] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1, 2

[26] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. 1, 2

[27] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011. 1, 2

[28] Victor Adrian Prisacariu, Olaf Kähler, Ming Ming Cheng, Carl Yuheng Ren, Julien Valentin, Philip HS Torr, Ian D Reid, and David W Murray. A framework for the volumetric integration of depth images. *arXiv preprint arXiv:1410.0925*, 2014. 2

[29] Jerome Revaud, Yohann Cabon, Romain Brégier, JongMin Lee, and Philippe Weinzaepfel. Sacreg: Scene-agnostic coordinate regression for visual localization, 2023. 3

[30] Antoni Rosinol, John J. Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields, 2022. 3, 6, 7

[31] Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. Point-slam: Dense neural point cloud-based slam, 2023. 2

[32] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6229–6238, 2021. 2

[33] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 6

[34] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13293–13302, 2023. 2

[35] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024. 3, 4

[36] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. 6

[37] Thomas Whelan, Stefan Leutenegger, Renato Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. 2015. 2

[38] Chi Yan, Delin Qu, Dan Xu, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Gs-slam: Dense visual slam with 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19595–19604, 2024. 2, 3

[39] Luwei Yang, Ziqian Bai, Chengzhou Tang, Honghua Li, Yasutaka Furukawa, and Ping Tan. Sanet: Scene agnostic network for camera localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 42–51, 2019. 3

[40] Xingrui Yang, Hai Li, Hongjia Zhai, Yuhang Ming, Yuqian Liu, and Guofeng Zhang. Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 499–507. IEEE, 2022. 2

[41] Sicheng Yu, Chong Cheng, Yifan Zhou, Xiaojun Yang, and Hao Wang. Rgb-only gaussian splatting slam for unbounded outdoor scenes. *arXiv preprint arXiv:2502.15633*, 2025. 2, 3, 4, 5, 6, 7

[42] Ganlin Zhang, Erik Sandström, Youmin Zhang, Manthan Patel, Luc Van Gool, and Martin R Oswald. Glorie-slam: Globally optimized rgb-only implicit encoding point cloud slam. *arXiv preprint arXiv:2403.19549*, 2024. 1, 2, 5, 6, 7

[43] Pengcheng Zhu, Yaoming Zhuang, Baoquan Chen, Li Li, Chengdong Wu, and Zhanlin Liu. Mgs-slam: Monocular sparse tracking and gaussian mapping with depth smooth regularization. *arXiv preprint arXiv:2405.06241*, 2024. 2, 3, 5

[44] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12786–12796, 2022. 3

[45] Zihan Zhu, Songyou Peng, Viktor Larsson, Zhaopeng Cui, Martin R Oswald, Andreas Geiger, and Marc Pollefeys. Nicer-slam: Neural implicit scene encoding for rgb slam. In *2024 International Conference on 3D Vision (3DV)*, pages 42–52. IEEE, 2024. 3, 6, 7