SPAR-3D
#6

StaRGS-SLAM: Stable and Robust Gaussian SLAM via Coverage-Aware Fixed-Topology Initialization

Anonymous SPAR-3D submission

Paper ID 6

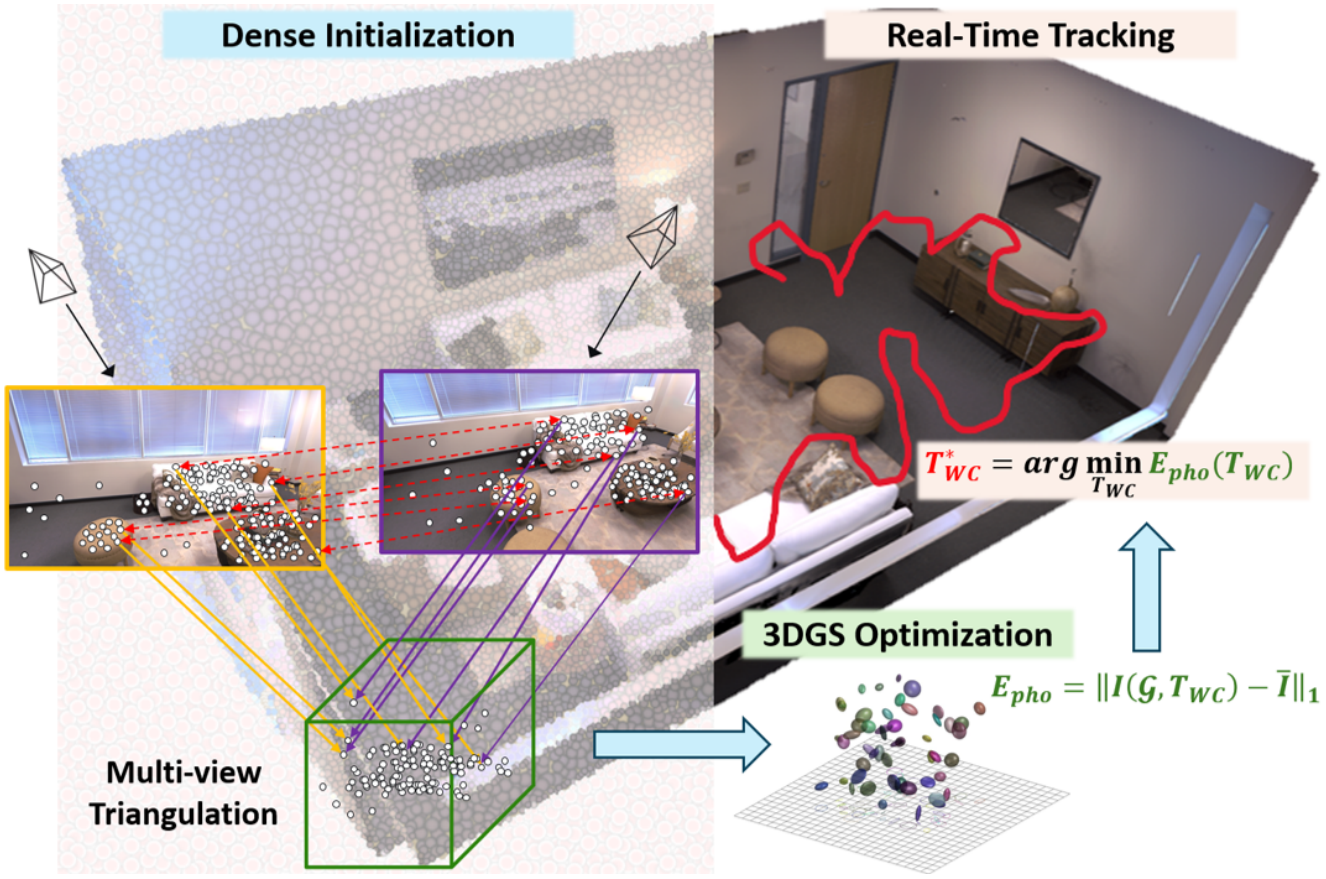

Figure 1. Overview of the proposed StaRGS-SLAM pipeline. The system integrates dense feature matching and multi-view triangulation for one-shot Gaussian initialization, followed by differentiable 3DGS optimization and real-time tracking.

Abstract

Reliable 3D scene modeling requires a mapping process that remains stable when early geometry is incomplete or visually ambiguous. We present StaRGS-SLAM, a Gaussian splatting SLAM system that improves robustness and efficiency through a training-free correspondence-guided Gaussian initialization scheme. Instead of relying on residual-driven densification during optimization, StaRGS-SLAM constructs a geometry-aware prior before iterative updates. From keyframes, it extracts dense correspondences from DINOv3 features, suppresses unreliable matches with confidence-aware inlier filtering, and triangulates the filtered observations to produce a well-covered Gaussian representation in a single initialization stage. This design provides stronger geometric support for early mapping, reduces sensitivity to unstable optimization behavior, and shortens convergence time by about 20%. Experiments on TUM RGB-D and Replica show that StaRGS-SLAM achieves competitive or superior localization and reconstruction performance compared with recent Gaussian-based and point-based SLAM methods, while maintaining real-time mapping throughput of up to 925 FPS.

1. Introduction

Recent advances in 3D Gaussian Splatting (3DGS) have enabled high-quality view synthesis and real-time mapping. However, most pipelines still rely on residual-driven densification, where Gaussians are iteratively spawned and merged as errors are detected. This causes non-stationary objectives, unstable convergence, and sensitivity to texture-rich or cluttered regions due to delayed coverage and uneven geometry.

We take a different approach by initializing from a complete and well-distributed Gaussian set rather than growing it incrementally. Using Dense Feature Matching (DFM), we obtain confidence-weighted correspondences within a short keyframe window, triangulate them into structure-aware matches, and instantiate the Gaussian set before optimization begins. Subsequent updates refine means, covariances, opacities, and colors while keeping topology fixed, resulting in stable and stationary optimization with strong spatial support even in high-frequency regions.

Integrated into monocular SLAM, this single-step seeding shortens time to usable maps, stabilizes early pose and shape estimation, and removes the need for additional networks or losses. It directly replaces densification with a brief feature-matching pass at keyframes while leaving the rest of the pipeline unchanged. An overview of the StaRGS-SLAM framework, including the initialization pipeline, is shown in Figure 1. Our contributions are summarized below.

- Single-Step Dense Initialization. A one-shot triangulation replaces residual-driven densification within the standard GS-SLAM pipeline, enabling stationary optimization and 20% faster convergence.
- Improved Localization Accuracy. Confidence-weighted correspondences stabilize early pose estimation, reducing drift by over 30%.
- Lightweight and Efficient Mapping. Spatially balanced Gaussians lower computation and memory, achieving 20% higher rendering throughput in real time.
- High-Fidelity Reconstruction. Dense seeding enhances early coverage and geometric consistency, yielding about 20% better reconstruction accuracy and completeness.

2. Related Work

3D Gaussian Splatting and Densification. 3D Gaussian splatting (3DGS) enables real-time view synthesis with anisotropic splats and a visibility-aware renderer, yet most systems rely on residual-driven densification [8]. We instead seed a fixed topology from dense multi-view correspondences and then refine only splat parameters.

Gaussian Splats for SLAM. SLAM systems mapping with Gaussians include GS-SLAM, MonoGS, SplaTAM, and Gauss-SLAM [7, 11, 20]. They rely on densification, causing early non-stationarity. Our training-free dense seed removes this stage and drops into MonoGS with minimal modifications.

Differentiable Rendering and Real-Time SLAM. Photo-SLAM, GLORIE-SLAM, and Point-SLAM couple differentiable rendering with pose optimization for fast updates via analytic/lightweight gradients [4, 12, 23]. We retain this in a Gaussian renderer and use a stationary initialization so early steps do not change topology.

Feature Matching and Dense Correspondence. SuperPoint/SuperGlue remain strong baselines [1, 13]. Detector-free transformers (LoFTR, LightGlue) extend coverage in low-texture regions, and dense matchers (DKM) provide broad two-view coverage with confidence for geometry [2, 9, 17]. We aggregate confidence-weighted dense correspondences over a short keyframe window into an explicit Gaussian seed.

Initialization and Training-Free Priors. SfM and multi-view stereo stabilize early optimization via geometric priors and learned depth [14, 22]. In Gaussian splatting, schedules and regularizers typically retain densification. Our correspondence-to-Gaussian initialization follows training-free priors and yields a stationary objective from the start.

3. Method

3.1. Gaussian Splatting Representation

We map the scene with a set of anisotropic Gaussians $\mathcal{G} = \{G_i\}$. Each G_i carries optical properties, a color vector c_i and an opacity $\alpha_i \in [0, 1]$, and geometric properties, a mean $\mu_i^W \in \mathbb{R}^3$ and a symmetric positive definite covariance $\Sigma_i^W \in \mathbb{R}^{3\times3}$ expressed in world coordinates. For brevity we describe color as a single vector and later allow spherical harmonics for view dependence.

Let $T_{WC} = [R \mid t]$ be the world–to–camera pose of the current view and let $\pi(\cdot)$ be the calibrated perspective projection. A 3D Gaussian $N(\mu_i^W, \Sigma_i^W)$ induces a 2D Gaussian on the image plane through first–order linearization of the projection around μ_i^W. The projected mean and covariance are

$$\mu_i^I = \pi\big(T_{WC}\,\mu_i^W\big), \qquad \Sigma_i^I = J_i\,R\,\Sigma_i^W\,R^\top J_i^\top, \tag{1}$$

where $J_i = \frac{\partial\,\pi(RX+t)}{\partial X}\big|_{X=\mu_i^W}$ is the Jacobian of the projection at μ_i^W. This relates the ellipsoid in 3D to an ellipse on the sensor.

Rendering is performed by rasterizing Gaussians instead of ray marching. For pixel p we collect the front–to–back ordered set $N(p)$ of screen–space Gaussians whose 2D footprints overlap p. The pixel color is obtained by α–compositing

$$C_p = \sum_{i\in N(p)} c_i\,\alpha_i \prod_{j=1}^{i-1}\big(1-\alpha_j\big), \tag{2}$$

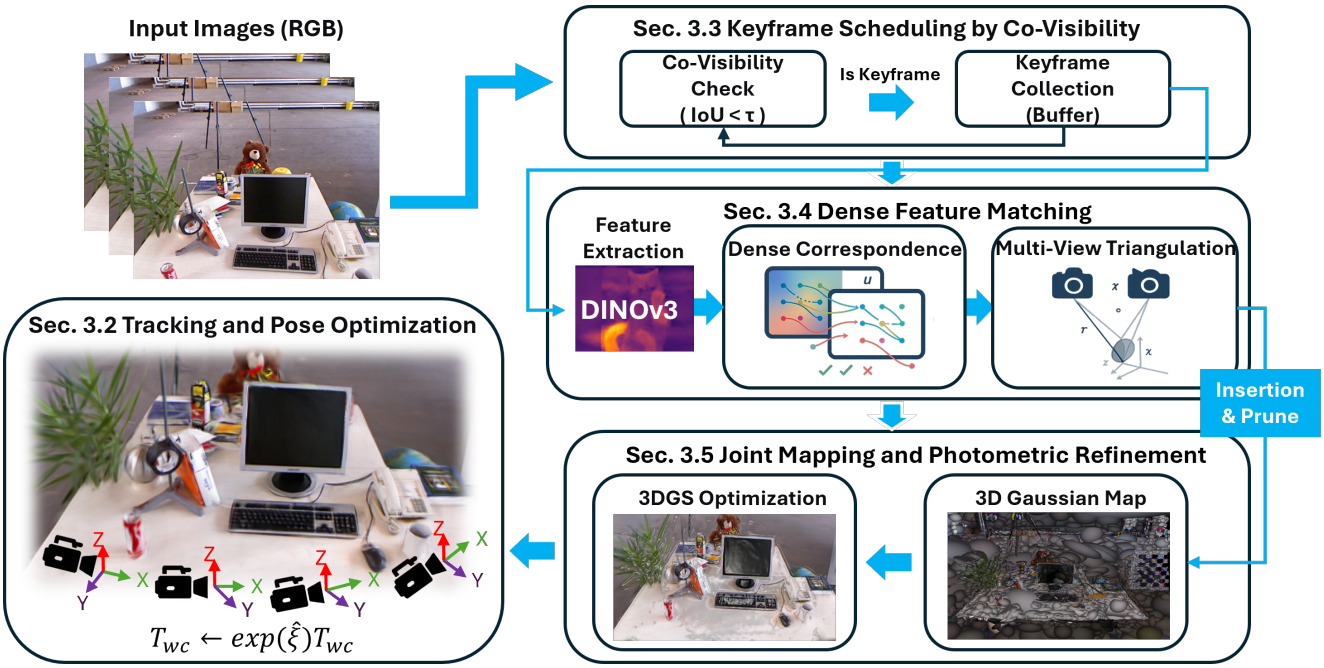

Figure 2. Detailed StaRGS-SLAM pipeline. Each keyframe triggers dense multi-view triangulation that yields a one-shot Gaussian initialization, subsequently refined through joint tracking and mapping within a differentiable 3DGS renderer using analytic SE(3) Jacobians.

where $\alpha_i \in [0,1]$ denotes the *screen–space* opacity at pixel p, obtained by modulating the primitive opacity with the value of the 2D Gaussian density at p using the parameters (μ_i^I, Σ_i^I) (the dependence on p is omitted for brevity). Empty space does not contribute because the renderer iterates over primitives that actually cover the pixel.

Equations (1) and (2) are differentiable in color, opacity, mean, covariance, and pose. Gradients flow through the splat weights and the projection Jacobian, which allows first–order optimizers to refine both optical and geometric parameters until the rendered image matches the observation with high fidelity. In subsequent sections we will initialize $\{\mu_i^W, \Sigma_i^W, \alpha_i, c_i\}$ densely in a single-step and then refine them under this differentiable renderer.

3.2. Tracking and Camera Pose Optimization

3.2.1. Objective and Per-Frame Update

For each incoming frame, we estimate the camera pose by minimizing an image-domain objective under the 3DGS renderer in Figure 2. Let $I(\mathcal{G}, T_{WC}) = \mathcal{S}(\mathcal{G}, T_{WC})$ be the rendered image and $\bar{I}$ the observation. The photometric residual is

$$E_{\text{pho}} = \left\| I(\mathcal{G}, T_{WC}) - \bar{I} \right\|_1, \tag{3}$$

augmented with an affine brightness model to absorb exposure changes, i.e., we jointly estimate gain and bias and substitute $g\, I(\cdot) + b$ into Eq. (3). We optimize $E = E_{\text{pho}}$. Pixels with low screen-space opacity or low image gradient are downweighted to reduce the influence of textureless regions. In practice we perform tens of gradient steps per frame to reach a stable update.

3.2.2. Alpha Compositing for Color

Rendering is carried out by rasterizing Gaussians in screen space and composing them front to back. For a pixel p, let $N(p)$ be the set of overlapping Gaussians sorted from near to far. The color follows the standard α-compositing in Eq. (2), which naturally handles occlusion via transmittance $\prod_{j<i}(1-\alpha_j)$. No depth map is produced or used in our tracking objective.

3.2.3. Minimal Pose Jacobians on SE(3)

We update the world-to-camera pose by a left-multiplicative twist SE(3),

$$T_{WC} \leftarrow \exp(\hat{\xi})\, T_{WC}, \tag{4}$$

where we differentiate the objective with respect to ξ in minimal coordinates. Let μ^W be a 3D Gaussian mean in world coordinates and $\mu^C = R\,\mu^W + t$ its camera-space position for $T_{WC} = [R \mid t]$. The 3D point Jacobian with respect to the pose twist is the standard 3×6 form

$$\frac{\partial \mu^C}{\partial \xi} = \begin{bmatrix} I & -[\mu^C]_\times \end{bmatrix}, \tag{5}$$

where $[\cdot]_\times$ is the skew-symmetric matrix. With calibrated projection π, the image-plane mean $\mu^I = \pi(\mu^C)$ has Jaco-

bian

$$\frac{\partial \mu^I}{\partial \xi} = J_\pi(\mu^C)\,[\,I \quad -[\mu^C]_\times\,], \tag{6}$$

where J_π is the 2×3 projection Jacobian evaluated at μ^C. The screen-space covariance Σ^I from Eq. (1) depends on both the projection Jacobian and the rotation, using the chain rule,

$$\frac{\partial \Sigma^I}{\partial \xi} = \frac{\partial \Sigma^I}{\partial J}\frac{\partial J}{\partial \mu^C}\frac{\partial \mu^C}{\partial \xi} + \frac{\partial \Sigma^I}{\partial R}\frac{\partial R}{\partial \xi}, \tag{7}$$

with $\partial R/\partial \xi$ obtained from the Lie algebra relation $\delta R \approx [\delta\omega]_\times R$ for an infinitesimal rotation $\delta\omega$. These analytic Jacobians remove the overhead of generic autodiff and match the degrees of freedom of the pose, which is essential for fast and stable tracking under a tight per-frame budget.

3.2.4. Optimization Solver and Weighting Scheme

We minimize the photometric objective in Eq. (3) using a first-order optimizer with a cosine learning rate schedule, and apply a robust penalty to per-pixel residuals. The per-pixel weight combines exposure correction, edge awareness, and visibility (via screen-space opacity) so that informative regions dominate the update. Because the 3DGS renderer and the pose Jacobians are fully analytic, gradients propagate through Eq. (2) and Eq. (6) without resorting to expensive automatic differentiation.

3.3. Keyframe Scheduling by Co-Visibility

Given the last accepted keyframe $I_{k^\star}$, we measure co-visibility between the current frame I_k and $I_{k^\star}$ by the intersection-over-union of visible Gaussians

$$\mathrm{IoU}(I_k, I_{k^\star}) = \frac{|V(I_k) \cap V(I_{k^\star})|}{|V(I_k) \cup V(I_{k^\star})|}, \tag{8}$$

where $V(I)$ collects Gaussians whose screen-space opacity exceeds a small threshold on a sufficient fraction of pixels. A new keyframe is created when $\mathrm{IoU}(I_k, I_{k^\star})$ is less than τ and the inter-view parallax is above a bound. Accepted keyframes are stored in a bounded buffer $\mathcal{B}$ that provides neighbours for multi-view initialization.

3.4. Dense Feature Matching

We extract dense visual descriptors using DINOv3 [15], which provide semantically consistent features across views. These descriptors are used to establish multi-view dense correspondences, replacing the residual-driven densification process in GS-SLAM. A confidence-aware inlier classifier is then applied to filter unreliable matches, ensuring stable multi-view geometry. Finally, a one-shot triangulation is performed to initialize a uniformly distributed set of 3D Gaussian seeds.

Dense Correspondence. Let I_r be the current keyframe and let $\mathcal{N}_r \subset \mathcal{B}$ be K neighbours selected by pose proximity and parallax. A dense matcher outputs, for each pair (r, n) with $n \in \mathcal{N}_r$, a displacement field $\mathbf{u}_{r\to n}(p)$ on I_r and a confidence map $\kappa_{r\to n}(p) \in [0, 1]$. A correspondence is represented as the pixel pair

$$\big(p,\ p + \mathbf{u}_{r\to n}(p)\big), \tag{9}$$

and low-confidence matches are filtered by a symmetric epipolar test and spatial blue-noise thinning. We aggregate per-view confidence for each retained reference pixel by

$$\bar{\kappa}(p) = \frac{1}{|\mathcal{N}_r|}\sum_{n\in\mathcal{N}_r} \kappa_{r\to n}(p). \tag{10}$$

Multi-view Triangulation. For each retained pixel p and neighbour $n \in \mathcal{N}_r$, we solve a two-view linear triangulation. Let $P_r, P_n \in \mathbb{R}^{3\times4}$ be the camera projection matrices and $\tilde{x}_r, \tilde{x}_n \in \mathbb{P}^2$ the homogeneous pixel coordinates. We form

$$A = \begin{bmatrix} \tilde{x}_r^x P_r^{3\top} - P_r^{1\top} \\ \tilde{x}_r^y P_r^{3\top} - P_r^{2\top} \\ \tilde{x}_n^x P_n^{3\top} - P_n^{1\top} \\ \tilde{x}_n^y P_n^{3\top} - P_n^{2\top} \end{bmatrix}, \quad \tilde{X} = \arg\min_{\|\tilde{X}\|=1} \|A\tilde{X}\|_2, \tag{11}$$

then $\hat{X} = \tilde{X}_{1:3}/\tilde{X}_4$ (obtained as the right singular vector of A associated with the smallest singular value). Among all neighbours we keep the hypothesis with the lowest reprojection error, breaking ties by larger baseline angle. Candidates with small parallax or large reprojection error are rejected.

Gaussian Parameter Initialization. Each valid triangulation spawns one Gaussian G_i with world mean

$$\mu_i^W = \hat{X}(p). \tag{12}$$

Construct a local orthonormal frame $U_i = [\mathbf{t}_1, \mathbf{t}_2, \mathbf{v}]$, where $\mathbf{v}$ is the surface normal estimated by a plane fit over neighbouring triangulated points, and $\mathbf{t}_1, \mathbf{t}_2$ span the tangent plane. Initialize the covariance as an anisotropic ellipsoid aligned with this frame

$$\Sigma_i^W = U_i\,\mathrm{diag}\big(s_\perp^2,\ s_\perp^2,\ s_\parallel^2\big)\,U_i^\top, \tag{13}$$

where $s_\perp$ is obtained by back-projecting a one-pixel image uncertainty via the projection Jacobian at the reference view, and $s_\parallel$ is set larger to encode depth uncertainty that increases when the baseline angle is small or the triangulation residual is high. The color c_i is the median of bilinearly sampled RGB values across supporting views after applying the exposure parameters estimated by tracking. The initial opacity α_i is a monotone mapping of the aggregated correspondence confidence $\bar{\kappa}(p)$ so that unreliable candidates remain visually weak at insertion. Finally, Gaussians are subsampled to maintain uniform spatial coverage before being inserted into the map.

3.5. Joint Mapping and Photometric Refinement

At each accepted keyframe, we first perform the single-step dense initialization (Sec. 3.4) to generate Gaussian seeds from multi-view correspondences. The surviving seeds are immediately inserted into $\mathcal{G}$ *without* any densification stage. Each newly inserted Gaussian participates in mapping right away.

Insertion, Lightweight Merging, and Pruning. Each Gaussian G_i tracks its observation count m_i, cumulative visibility v_i, and an exponential moving average of its screen-space footprint. To keep memory bounded and remove unstable outliers, we apply a lightweight periodic cleanup and prune splats that violate

$$\begin{aligned} m_i < m_{\min}, \qquad & v_i < v_{\min}, \\ \operatorname{tr}\big(\Sigma_i^W\big) > \sigma_{\max}^2, \qquad & \alpha_i < \alpha_{\min}. \end{aligned}$$

Neighbouring Gaussians with highly overlapping footprints and similar colors are merged, retaining a visibility-weighted mean of color and covariance to avoid overpopulation.

Sliding-Window Photometric Refinement. Let $\mathcal{W}$ denote a window around the latest keyframe. We jointly refine $\{T_{WC}\}_{I\in\mathcal{W}}$ and $\{c_i, \alpha_i, \mu_i^W, \Sigma_i^W\}$ by minimizing

$$\mathcal{L} = \sum_{I\in\mathcal{W}} \lambda_{\text{pho}} \left\| g_I\, \mathcal{S}(\mathcal{G}, T_{WC}) + b_I - \bar{I} \right\|_1 + \mathcal{R}, \quad (13)$$

where g_I, b_I compensate exposure changes. The regularizer

$$\begin{aligned} \mathcal{R} = \lambda_{\text{iso}} \sum_i \left\| \Sigma_i^W - \tfrac{\operatorname{tr}(\Sigma_i^W)}{3} I_3 \right\|_F \\ + \lambda_\alpha \sum_i \psi(\alpha_i) + \lambda_\mu \sum_i \left\| \mu_i^W - \bar{\mu}_i^W \right\|_2^2 \end{aligned} \quad (14)$$

discourages needle-shaped ellipsoids, avoids degenerate transmittance, and stabilizes early iterations via an EMA anchor $\bar{\mu}_i^W$. We alternate pose-only updates and full map updates with robust per-pixel weights. Gradients propagate through the analytic α-compositing in Eq. (2) and the pose Jacobians in Eq. (6).

3.6. System Schedule and Computational Profile

Each incoming frame is tracked for K_t gradient steps using the photometric objective in Eq. (3). When the co-visibility test Eq. (8) accepts a keyframe, we select K neighbours from $\mathcal{B}$ and execute the single-step dense initialization of Sec. 3.4 (dense correspondence, weighted multi-view triangulation, and parameter initialization) in one pass. The resulting Gaussians are immediately inserted into $\mathcal{G}$, followed by K_m mapping iterations over the current window $\mathcal{W}$ optimizing Eq. (13), and a lightweight cleanup as described in Sec. 3.5. Replacing iterative densification with this single-step initialization reduces the drift of newly added parameters and lowers the number of mapping iterations required to reach the same photometric fidelity, improving wall-clock throughput without changing the objective or renderer.

4. Experiments

4.1. Experimental Setup

Datasets. We evaluate on TUM RGB-D and Replica. TUM RGB-D is evaluated in both monocular and RGB-D settings. Replica is employed for photometric map evaluation on *room0–2* and *office0–4*, matching the splits used in our tables to ensure comparability of rendering metrics and throughput.

Implementation. Gaussian rasterization and gradients are implemented in CUDA, and the remaining pipeline is in PyTorch. Mixed precision is enabled where beneficial. Tracking runs in real time, while mapping executes asynchronously within a bounded local window. Non-standard hyperparameters (learning rate schedule, window sizes, keyframe and culling thresholds) are provided in the supplementary.

Evaluation Metrics. Tracking accuracy uses RMSE of Absolute Trajectory Error (ATE) on keyframes. Photometric quality adopts PSNR [3], SSIM [19], and LPIPS [24]. Reconstruction quality reports *Acc.* [cm]↓, *Comp.* [cm]↓, and *Comp.Ratio* (%)↑. Unless specified, we uniformly sample 50K surface points, set $\tau = 5$ cm, and average per scene. Photometric metrics are computed on every fifth frame excluding keyframes. Reconstruction metrics use the same sampling protocol. Each experiment is repeated three times, and the mean results are reported.

Baseline Methods. We compare with iMAP [16], NICE-SLAM [26], Vox-Fusion [21], ESLAM [6], Point-SLAM [12], Co-SLAM [18], SplaTAM [7], Gauss-SLAM [20], and MonoGS [11]. We also include SNI-SLAM [25] for reconstruction and Photo-SLAM [4], GLORIE-SLAM [23], RK-SLAM [10] for rendering. RGB-D–only methods run in RGB-D, with monocular results reported only when supported. Hyperparameters follow official documented defaults on the same splits.

4.2. Training and Convergence Analysis

The system optimizes scene specific Gaussian parameters and camera poses using the same optimizer, schedule, and window size as MonoGS, with densification removed. Each frame is tracked for K_t steps. Each accepted keyframe triggers the single-step dense initialization followed by K_m mapping steps with exposure compensation. Wall clock time is measured on identical hardware and stopping criteria. Results are summarized in Table 1. On TUM RGB-D the average training time decreases from 14.8 to 12 minutes while maintaining localization and rendering quality.

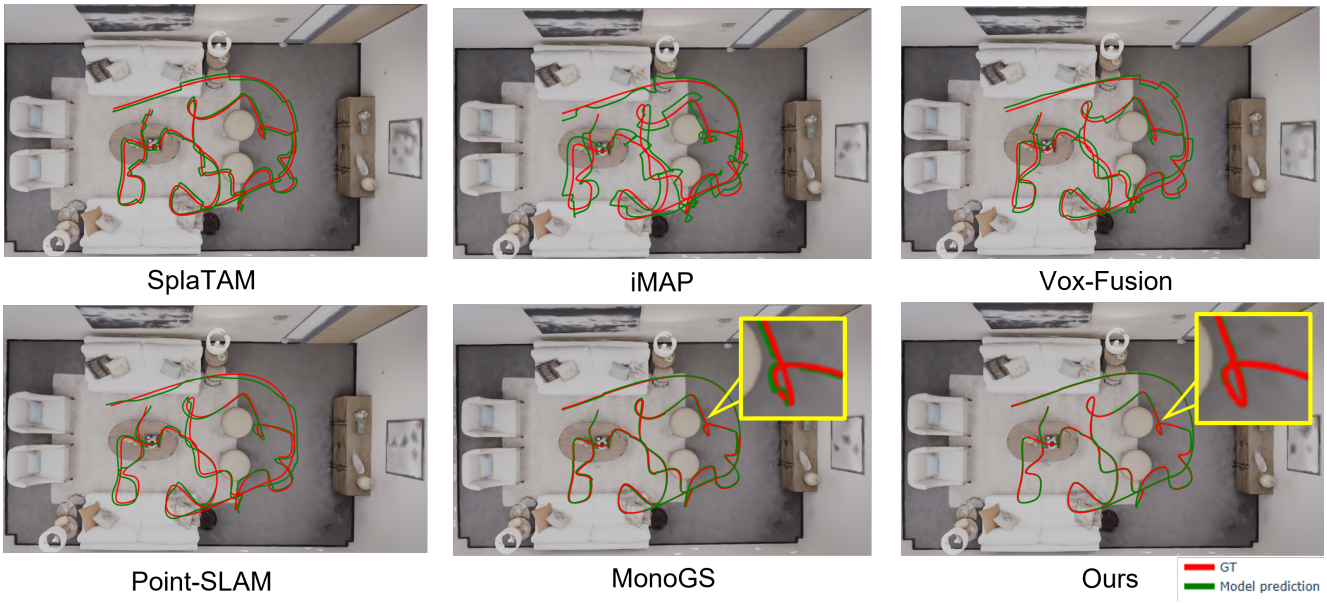

Figure 3. Trajectory comparison on a living-room scene. The red line indicates the ground-truth path and the green line shows the estimated trajectory. Our method aligns more closely with the ground-truth and exhibits fewer large drifts than previous systems.

Table 1. Optimization time (min) on TUM RGB-D sequences *fr1/desk*, *fr2/xyz*, and *fr3/office*.

Method	*fr1/desk*	*fr2/xyz*	*fr3/office*	Avg.
MonoGS [11]	6.4	20.6	17.5	14.8
Ours (RGB)	**4.9**	**16.1**	**15.0**	**12.0**

4.3. Localization Accuracy

We evaluate trajectory accuracy on Replica and TUM (Tables 2, 3). On Replica, the average ATE is **0.61 cm** across *r0–r2, o0–o4*, outperforming iMAP (2.58), NICE-SLAM (1.07), Vox-Fusion (3.09), and ESLAM (0.90 cm), while remaining competitive with Point-SLAM and MonoGS. On TUM RGB-D, the average ATE is **1.02 cm** on *fr1/desk*, *fr2/xyz*, and *fr3/office*, achieving better results than MonoGS and surpassing the same baselines. Figure 3 shows a trajectory comparison on a living-room scene from the Replica dataset, where the red line indicates the ground-truth (GT) path and the green line represents the estimated trajectory. Among existing methods, SplaTAM, iMAP, Vox-Fusion, and Point-SLAM exhibit large localization drift, with evident deviations from the GT path. MonoGS and our method perform significantly better. In this visualization, the red line is drawn above the green line, so greater overlap, where the red line covers the green one, intuitively reflects smaller localization error. Our method achieves a higher overlap ratio, indicating closer adherence to the GT trajectory and better pose consistency. The zoom-in comparison further shows smoother alignment and reduced drift compared with MonoGS, demonstrating stronger robustness in long-term tracking and loop-closure maintenance.

Table 2. Camera tracking results on the Replica dataset under the RGB-D setting. Reported values denote RMSE of ATE across *room0–2* and *office0–4*.

Method	room0	room1	room2	office0	office1	office2	office3	office4	Avg.
iMAP [16]	3.12	2.54	2.31	1.69	1.03	3.99	4.05	1.93	2.58
NICE-SLAM [26]	0.97	1.31	1.07	0.88	1.00	1.06	1.10	1.13	1.07
Vox-Fusion [21]	1.37	4.70	1.47	8.48	2.04	2.58	1.11	2.94	3.09
ESLAM [6]	0.71	0.70	0.52	0.57	0.55	0.58	0.72	**0.63**	0.63
Point-SLAM [12]	0.61	0.41	**0.37**	**0.38**	**0.48**	0.54	0.69	0.72	**0.53**
MonoGS [11]	0.62	0.62	0.77	0.44	0.52	**0.23**	0.62	2.25	0.76
Ours (RGB)	**0.45**	**0.51**	0.53	0.52	0.78	1.03	**0.45**	**0.63**	0.61

Table 3. Camera tracking results on the TUM RGB-D dataset. Values denote RMSE of ATE over *fr1/desk*, *fr2/xyz*, and *fr3/office*.

Method	fr1/desk	fr2/xyz	fr3/office	Avg.
iMAP [16]	4.90	2.00	5.80	4.23
NICE-SLAM [26]	4.26	6.19	3.87	4.77
DI-Fusion [5]	4.40	2.00	5.80	4.07
Vox-Fusion [21]	3.52	1.49	26.01	10.34
ESLAM [6]	2.47	1.11	2.42	2.00
Co-SLAM [18]	2.40	1.70	2.40	2.17
Point-SLAM [12]	4.34	1.31	3.48	3.04
MonoGS [11]	1.50	1.44	1.49	1.47
Ours (RGB)	**1.02**	**0.98**	**1.05**	**1.02**

Figure 4. Rendering results on the TUM dataset. The proposed keyframe-triggered single-step initialization produces sharper edges, fewer transparency artifacts, and more consistent colors than residual-driven densification.

4.4. Rendering Quality and Throughput

We report fidelity and throughput in Table 4 and 5. On Replica, our initializer averages **925 FPS**, exceeding MonoGS (769 FPS) while maintaining competitive PSNR [3], SSIM [19], and LPIPS [24] across *room0–2* and *office0–4*. On TUM, the system runs in real time (2.5–3.2 FPS) with PSNR/SSIM comparable to SplaTAM, Photo-SLAM, and GLORIE-SLAM, and low LPIPS. The throughput gain arises from the keyframe-triggered single-step dense initialization, which fixes Gaussian topology upfront and removes residual-driven densification, reducing per-frame cost. Qualitative results in Figure 4 show sharper edges, fewer transparency artifacts, and more consistent colors than residual-driven baselines.

Table 4. Rendering quality results on the Replica dataset across room0–2 and office0–4.

Method (FPS)	Metric	room0	room1	room2	office0	office1	office2	office3	office4	Avg.
NICE-SLAM [26]	PSNR[dB]↑	22.12	22.47	24.52	29.07	30.34	19.66	22.23	24.94	24.42
(6.54)	SSIM↑	0.689	0.757	0.814	0.874	0.868	0.797	0.801	0.856	0.809
	LPIPS↓	0.330	0.271	0.208	0.229	0.181	0.235	0.209	0.198	0.233
Vox-Fusion [21]	PSNR[dB]↑	22.39	22.36	23.92	27.79	29.83	20.33	23.47	25.21	24.41
(2.17)	SSIM↑	0.683	0.751	0.798	0.857	0.876	0.794	0.803	0.847	0.801
	LPIPS↓	0.303	0.269	0.234	0.241	0.184	0.243	0.213	0.199	0.236
Point-SLAM [12]	PSNR[dB]↑	32.40	34.08	35.50	38.26	39.16	33.98	33.48	33.49	35.17
(1.33)	SSIM↑	0.974	**0.977**	0.979	0.982	0.986	0.962	0.960	**0.979**	**0.975**
	LPIPS↓	0.113	0.116	0.111	0.100	0.118	0.156	0.132	0.142	0.124
Co-SLAM [18]	PSNR[dB]↑	28.88	28.51	29.37	35.44	34.63	26.56	28.79	32.16	28.42
	SSIM↑	0.892	0.843	0.851	0.854	0.826	0.814	0.866	0.856	0.837
	LPIPS↓	0.213	0.205	0.215	0.177	0.161	0.172	0.163	0.176	0.185
SplaTAM [7]	PSNR[dB]↑	32.49	33.72	34.65	38.29	39.04	31.91	30.05	31.83	30.98
	SSIM↑	**0.975**	0.970	0.980	0.982	0.982	0.965	0.952	0.949	0.953
	LPIPS↓	0.072	0.096	0.078	0.086	0.093	0.100	0.110	0.150	0.179
Gauss-SLAM [20]	PSNR[dB]↑	29.57	31.61	33.46	38.39	39.62	32.91	33.62	34.26	30.90
	SSIM↑	0.944	0.952	0.973	**0.985**	**0.991**	**0.974**	**0.982**	**0.979**	0.972
	LPIPS↓	0.197	0.184	0.148	0.099	0.097	0.158	0.123	0.138	0.229
MonoGS [11]	PSNR[dB]↑	**34.83**	**36.43**	**37.49**	**39.95**	**42.09**	**36.24**	**36.70**	**36.07**	**37.50**
(769)	SSIM↑	0.954	0.959	0.965	0.971	0.974	0.964	0.963	0.957	0.960
	LPIPS↓	**0.068**	**0.076**	**0.075**	**0.072**	**0.055**	**0.078**	**0.065**	0.099	**0.070**
Ours (RGB)	PSNR[dB]↑	**35.95**	33.55	32.45	34.45	35.45	34.87	34.02	35.85	34.57
(925)	SSIM↑	0.852	0.945	**0.985**	0.952	0.925	0.952	0.855	0.961	0.928
	LPIPS↓	0.085	0.092	0.112	0.088	0.096	**0.078**	0.101	**0.096**	0.093

Table 5. Rendering quality results on the TUM RGB-D dataset.

Method (FPS)	Metric	fr1/desk	fr2/xyz	fr3/office	Avg.
Point-SLAM [12]	PSNR[dB]↑	13.79	17.62	18.29	16.57
	SSIM↑	0.625	0.710	0.749	0.695
	LPIPS↓	0.545	0.584	0.452	0.527
Photo-SLAM [4]	PSNR[dB]↑	20.97	21.07	19.59	20.54
	SSIM↑	0.740	0.730	0.690	0.720
	LPIPS↓	**0.230**	0.170	0.240	0.213
MonoGS [11]	PSNR[dB]↑	19.67	16.17	20.63	18.82
	SSIM↑	0.730	0.720	0.770	0.740
	LPIPS↓	0.330	0.310	0.340	0.327
GLORIE-SLAM [23]	PSNR[dB]↑	20.26	**25.62**	21.21	22.36
	SSIM↑	0.790	0.720	0.720	0.743
	LPIPS↓	0.310	**0.090**	0.320	0.240
SplaTAM [7]	PSNR[dB]↑	21.49	25.06	21.17	22.57
	SSIM↑	0.839	**0.950**	**0.861**	**0.883**
	LPIPS↓	0.255	0.099	0.221	**0.192**
RK-SLAM [10]	PSNR[dB]↑	22.31	22.47	20.67	21.82
	SSIM↑	0.741	0.729	0.710	0.727
	LPIPS↓	0.254	0.220	0.251	0.242
Ours (RGB)	PSNR[dB]↑	**23.11**	24.85	**23.59**	**23.85**
	SSIM↑	**0.853**	0.896	0.801	0.850
	LPIPS↓	0.232	0.198	**0.219**	0.216

4.5. Reconstruction Fidelity

Geometric fidelity is evaluated using accuracy, completeness, and completeness ratio in Table 6, computed on aligned point clouds under standard thresholds. Our method attains **1.537 cm** accuracy and **1.477 cm** completeness with a **97.843%** completeness ratio. Relative to SNI-SLAM, accuracy improves by 20.9% and completeness by 13.2%, together with a 1.22-point gain in completeness ratio. The margins over ESLAM and Vox-Fusion are larger, including a 42% reduction in completeness error against Vox-Fusion. The improvements are consistent across scenes with thin structures and clutter, where coverage gaps and over-regularization commonly inflate geometric error. Qualitative inspection shows reduced truncation at object boundaries, cleaner reconstruction of high-frequency edges, and better recovery of small appendages. We attribute these outcomes to anisotropic Gaussian primitives with visibility-aware α-compositing, which sharpen depth gradients and limit color bleeding, and to a bounded, keyframe-related optimization that preserves spatial coverage without topology changes. By keeping the Gaussian set fixed after dense seeding, the optimization remains stationary and avoids late-map artifacts, which stabilizes surface inference and suppresses oversmoothing during refinement.

Table 6. Reconstruction results on the Replica dataset. Lower is better for Acc./Comp., higher for Comp.Ratio.

Methods	Reconstruction		
	Acc. [cm] ↓	Comp. [cm] ↓	Comp.Ratio (%) ↑
iMAP [16]	3.624	4.934	80.515
NICE-SLAM [26]	2.373	2.645	91.137
Vox-Fusion [21]	1.882	2.563	90.936
Co-SLAM [18]	2.104	2.082	93.435
ESLAM [6]	2.082	1.754	96.427
SNI-SLAM [25]	1.942	1.702	96.624
Ours	**1.537**	**1.477**	**97.843**

4.6. Ablation Study

Effect of Dense Initialization. Consistent rendering gains on TUM RGB, with higher PSNR/SSIM and lower LPIPS/RMSE across all sequences. On *fr1.desk*, *fr2.xyz*, and *fr3.office*, PSNR improves to 23.11, 24.85, and 23.59 with SSIM gains and LPIPS/RMSE drops (Table 7). Distributed Gaussian seeds, whose multi-view triangulation stabilizes mapping under larger motion and maintains coverage in low-parallax segments. This yields faster convergence and fewer artifacts on thin structures and cluttered regions. Without dense initialization, residual driven densification converges slowly, exhibits early spatial inconsistency, and tends to over-smooth before adequate coverage is established.

Table 7. Impact of DFM on the TUM RGB-D dataset.

	Method	PSNR ↑	SSIM ↑	LPIPS ↓	RMSE ↓
fr1_desk	w/o DFM	19.67	0.73	0.33	1.5
	Ours	**23.11**	**0.853**	**0.232**	**1.02**
fr2_xyz	w/o DFM	16.17	0.72	0.31	1.44
	Ours	**24.85**	**0.896**	**0.198**	**0.98**
fr3_office	w/o DFM	20.63	0.77	0.34	1.49
	Ours	**23.59**	**0.801**	**0.219**	**1.05**

Effect of Gaussian Count per Keyframe on Tracking. We vary the number of newly triangulated 3D points per keyframe, with each verified point instantiated as a Gaussian primitive, so the abscissa in Figure 5 corresponds to the count of Gaussians. Increasing the budget from 200 to 1000 points reduces the tracking error sharply, reaching about **0.7cm** at 1000. Beyond this regime the curve plateaus and improvements are marginal, approaching roughly 0.6 cm at 2000. We therefore adopt 1000 points per keyframe as the default trade-off between accuracy, memory, and runtime.

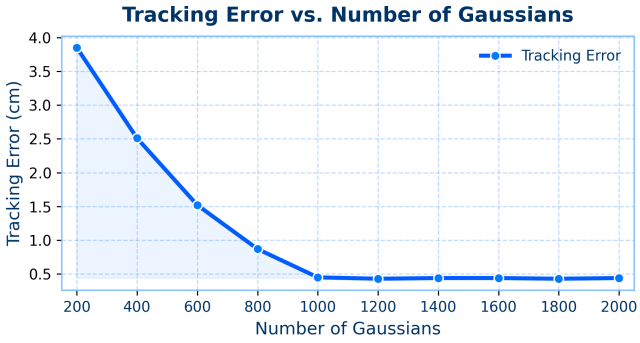

Figure 5. Tracking error versus Gaussian count. Error decreases rapidly with denser seeding and plateaus near 1000 Gaussians, indicating diminishing returns beyond this density.

5. Conclusion

We presented StaRGS-SLAM, a Gaussian splatting SLAM framework that replaces residual-driven densification with a keyframe-based correspondence-guided initialization strategy. Through dense multi-view matching and triangulation, the method forms a geometry-aware Gaussian prior that strengthens early mapping and improves the reliability of subsequent optimization under analytic $\mathrm{SE}(3)$ Jacobians. Experiments on Replica and TUM RGB-D show that this design reduces computational overhead while preserving strong localization accuracy and rendering quality. These results suggest that a more structured initialization stage can make 3D scene modeling more stable and dependable in visually challenging conditions, while remaining compatible with existing Gaussian splatting SLAM pipelines.

References

[1] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018. 2

[2] Johan Edstedt, Ioannis Athanasiadis, Mårten Wadenbäck, and Michael Felsberg. DKM: Dense kernelized feature matching for geometry estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[3] Alain Hore and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2010. 5, 7

[4] Huajian Huang, Longwei Li, Hui Cheng, and Sai-Kit 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 (CVPR)*, 2024. 2, 5, 7

[5] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu. DI-Fusion: Online implicit 3D reconstruction with deep priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 6

[6] 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 (CVPR)*, 2023. 5, 6, 8

[7] 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 (CVPR)*, 2024. 2, 5, 7

[8] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 2023. 2

[9] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[10] Xiasheng Ma, Ci Song, Yimin Ji, and Shanlin Zhong. Related keyframe optimization Gaussian–simultaneous localization and mapping: A 3D Gaussian Splatting-based simultaneous localization and mapping with related keyframe optimization. *Applied Sciences*, 2025. 5, 7

[11] Hidenobu Matsuki, Riku Murai, Paul Kelly, and Andrew J. Davison. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 5, 6, 7

[12] Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. Point-SLAM: Dense neural point cloud-based SLAM. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2, 5, 6, 7

[13] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[14] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[15] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025. 4

[16] 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 (ICCV)*, 2021. 5, 6, 8

[17] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[18] 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 (CVPR)*, 2023. 5, 6, 7, 8

[19] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 5, 7

[20] Chao Yan, Zirui Wang, Zhiqiang Li, Wei Gao, Hao Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. GS-SLAM: Dense visual SLAM with 3D Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 2, 5, 7

[21] 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 *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 499–507, 2022. 5, 6, 7, 8

[22] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[23] 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. 2, 5, 7

[24] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of

deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5, 7

[25] Siting Zhu, Guangming Wang, Hermann Blum, Jiuming Liu, Liang Song, Marc Pollefeys, and Hesheng Wang. SNI-SLAM: Semantic neural implicit SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21167–21177, 2024. 5, 8

[26] 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 (CVPR)*, 2022. 5, 6, 7, 8