LINGGS: DECOUPLING FEED-FORWARD GEOMETRY PREDICTION FROM CAMERA TRACKING FOR HIGH-FIDELITY GAUSSIAN SLAM

Anonymous authors
Paper under double-blind review

ABSTRACT

Achieving high-quality reconstruction and responsive camera tracking remains challenging for monocular RGB Gaussian splatting SLAM under a limited optimization budget. We present an online system that addresses this challenge by combining learned geometric priors with explicit geometric camera tracking, which can substantially outperform learned camera prediction in terms of tracking accuracy. A frozen streaming 3D foundation model serves as a virtual RGB-D sensor, providing depth estimation from monocular video. An explicit geometric solver then estimates camera motion through feature reprojection, and a streaming Gaussian mapper integrates the resulting observations into a dense scene representation. We further show that geometric tracking improves camera pose accuracy across representative streaming geometry backbones, demonstrating the broader applicability of our approach. Extensive experiments on indoor and outdoor datasets show that our method significantly improves reconstruction quality over existing approaches, achieving $\sim$7 FPS on a single RTX 4090 and delivering 2–3× the throughput of the evaluated prior SLAM systems.

1 INTRODUCTION

Monocular Gaussian SLAM aims to estimate the camera pose and build a renderable scene map online using only an RGB image stream. This capability supports immersive interaction, live scene capture, and remote interaction in AR/VR and robotics. 3D Gaussian splatting (GS) (Kerbl et al., 2023) provides an explicit scene representation with efficient rendering and differentiable mapping, making it a useful representation for dense SLAM. However, unlike RGB-D Gaussian SLAM methods (Keetha et al., 2024; Yan et al., 2024; Yugay et al., 2023; Peng et al., 2024; Ha et al., 2024; Zhu et al., 2025), without measured depth, the monocular GS SLAM system must jointly recover camera pose, scene geometry, and the appearance, within a limited computational budget.

Methods that rely on sparse feature points or incremental map optimization struggle to achieve high-fidelity mapping Matsuki et al. (2024); Huang et al. (2024). Learned dense priors help fill this gap, but integrating them into a consistent map introduces further challenges. Dense depth prediction and camera joint optimization add online computational cost (Homeyer et al., 2025; Cheng et al., 2025), while per-frame depth predictions do not ensure cross-frame scale consistency and require additional scale recovery and geometric alignment (Deng et al., 2025; Yu et al., 2026). Consistent depth and camera poses are both essential to this task. Depth determines where Gaussians are initialized, while camera poses determine whether observations from different frames align. Poor initialization requires additional map refinement, and inaccurate poses introduce conflicting observations that can blur the reconstructed scene. The challenge is therefore to obtain consistent dense depth and camera poses together within a limited online computational budget.

Streaming 3D foundation models offer a promising starting point. By maintaining temporal context, they predict consistent dense depth and camera poses as images arrive (Su et al., 2026; Zhuo et al., 2026; Chen et al., 2026b;a). Their depth predictions provide dense surface estimates for the Gaussian initialization, reducing the need to recover the scene structure from sparse feature points. Yet this advantage depends on the quality of the accompanying poses. We observe that directly using the models' predicted poses can lead to cross-frame misalignment, limiting reconstruction quality even when useful depth estimates are available. To this end, we introduce an explicit geometry pose

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

solver that combines 2D feature points with predicted depth and utilizes them to jointly constrain camera pose estimation.

Our system requires no additional training and runs at $\sim$7 FPS on a single RTX 4090 GPU. It achieves higher rendering quality than the previous baselines across indoor and outdoor datasets, with the lowest depth error and competitive tracking error. Overall, our main contributions are:

- **Depth-assisted geometric tracking for streaming 3D models.** We introduce an explicit geometry solver for camera tracking that combines 2D feature points with predicted depth. Intensive experiments show that our geometry solver can provide better tracking than the learned streaming camera head.
- **An efficient monocular Gaussian SLAM system.** We build a faster high-fidelity monocular GS SLAM system combining the geometric solver, a frozen streaming model, and a Gaussian mapper, leading to higher appearance quality. Across evaluated datasets, our system can improve rendering PSNR by up to 3.1 dB.
- **Generalization of geometry solver across streaming models.** We apply our geometry solver to multiple representative streaming models. We show that the geometry solver reduces trajectory error by up to $\sim$97% relative to the models' native camera head, demonstrating its effectiveness across different backbones and providing insight for future streaming model design.

2 Related Work

Visual tracking without depth prior. These methods recover scene geometry from RGB observations through visual tracking and map optimization. MonoGS (Matsuki et al., 2024) estimates poses by photometric alignment with the Gaussian map, making tracking dependent on the quality of the evolving reconstruction. Photo-SLAM (Huang et al., 2024) combines ORB-SLAM3 (Campos et al., 2021) with Gaussian mapping. CaRtGS (Feng et al., 2025), RP-SLAM (Bai et al., 2026), and SEGS-SLAM (Wen et al., 2025) further develop this architecture through improvements to computation allocation, map updates, and scene modeling. MGSO (Hu et al., 2025a) instead initializes Gaussians from geometry recovered by direct photometric odometry. Without learned dense geometric priors, surface recovery relies on multi-view image observations and parallax. Regions with weak visual constraints may remain insufficiently reconstructed at initialization, requiring interpolation or further geometric refinement during mapping.

Learned geometry for tracking and mapping. Learned depth and pointmap predictions provide richer surface information for tracking and mapping. Splat-SLAM (Sandström et al., 2024), DROID-Splat (Homeyer et al., 2025), and HI-SLAM2 (Zhang et al., 2025) integrate learned dense depth and trajectory refinement. MGS-SLAM (Zhu et al., 2024) uses multi-view depth supervision, while SplatMAP (Hu et al., 2025b) updates Gaussian positions using refined poses and depths from dense SLAM, and uses reliability masks to prune or add Gaussians. Some methods strengthen geometric consistency through map-to-tracker feedback or optical flow constraints (Zhu et al., 2026; Seo et al., 2026), and suppress unreliable observations in dynamic scenes (Zheng et al., 2025; Li et al., 2025a). These designs improve geometric estimation, but dense tracking and iterative refinement add computation to the online pipeline, limiting their application.

Feed-forward prediction offers a way to reduce this burden. StreamGS (Li et al., 2025b) and Flash-Mono (Zhang et al., 2026) predict poses and Gaussian attributes before limited map refinement, but require intensive model retraining. Pointmap and multi-view foundation models also support tracking and mapping in OpenGS-SLAM (Yu et al., 2025), ARTDECO (Li et al., 2026) and S3PO-GS (Cheng et al., 2025), but with a high tracking cost. GeoGS-SLAM (Gao et al., 2026), VGGT-GS SLAM (Han et al., 2026), and M^3 (Ren et al., 2026) combine learned predictions with geometric or photometric optimization to refine poses and scene estimates by processing frames in batches. OTF (Meuleman et al., 2025) explores efficient incremental Gaussian construction, but loses the consistent geometry reconstruction.

Combining predicted geometry with explicit pose estimation is particularly relevant to our work. GigaSLAM (Deng et al., 2025) and KiloGS-SLAM (Yu et al., 2026) use predicted depth with geometric solvers, while KiloGS-SLAM additionally activates learned pixel matching when needed,

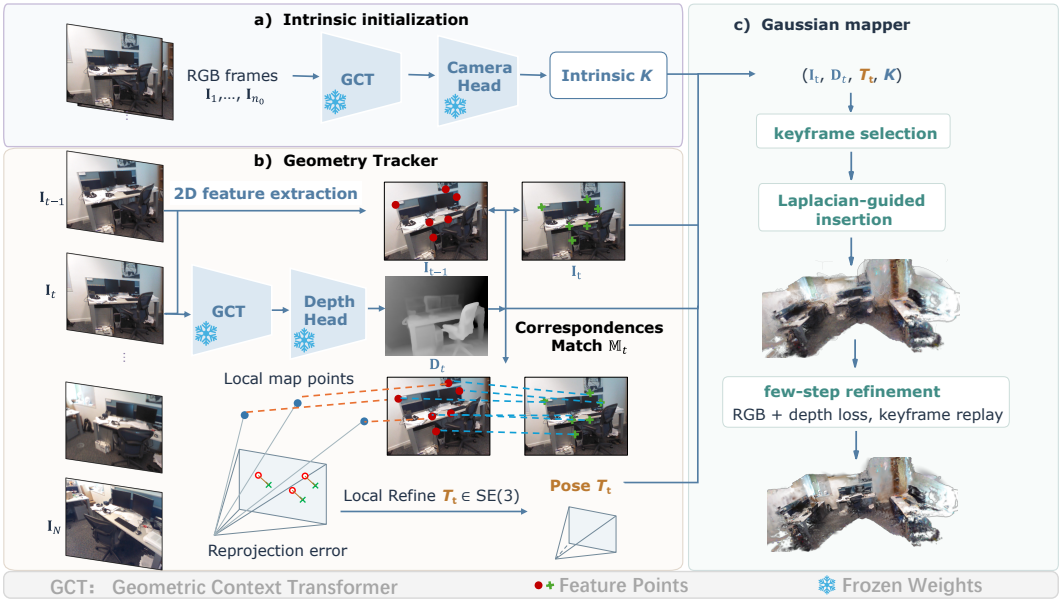


Figure 1: Overview of our online Gaussian splatting SLAM system. (a) A frozen streaming foundation model estimates camera intrinsics from the initial RGB frames. (b) Predicted depth and image features support geometric camera tracking through correspondence matching and reprojection error minimization. (c) The Gaussian mapper integrates RGB frames, predicted depth, and estimated poses through keyframe selection, Laplacian-guided Gaussian insertion, and 30-step refinement using RGB and depth losses with keyframe replay.

but they suffer from the scale drift. MonoEM-GS (Kruzhkov & Behnke, 2026) uses the iterative closest point (ICP) to align predicted pointmap with the Gaussian representation. These approaches demonstrate the value of learned geometry for explicit camera estimation. Our work focuses on using streaming consistent depth predictions for camera tracking and Gaussian mapping.

3 METHOD

As illustrated in Figure 1, given a monocular RGB sequence $\{\mathbf{I}_t\}_{t=1}^{N}$, our system combines learned geometric priors with explicit geometric camera tracking to incrementally estimate camera poses and build a Gaussian scene representation. The LingBot-Map (Chen et al., 2026a) first establishes a fixed camera calibration $\boldsymbol{K}$ from the initial frames and subsequently provides depth $\mathbf{D}_t$ as a virtual depth sensor. The geometric solver (Sec. 3.1) combines these geometric measurements with image features to estimate camera poses $\boldsymbol{T}_t$ by minimizing reprojection error. The Gaussian mapper (Sec. 3.2) integrates $(\mathbf{I}_t, \mathbf{D}_t, \boldsymbol{T}_t, \boldsymbol{K})$ through keyframe selection, Laplacian-guided Gaussian insertion, and 30 refinement steps using RGB and depth losses with keyframe replay.

3.1 GEOMETRY SOLVER

Online Gaussian mapping requires a camera pose for each incoming frame to integrate its observations into a common reconstruction. We build our geometry solver on ORB-SLAM3 (Campos et al., 2021; Huang et al., 2024), using the depth predicted by the LingBot-Map as a virtual RGB-D measurement. At frame t, the solver receives the current RGB image $\mathbf{I}_t$, the predicted depth map $\mathbf{D}_t$. Together with the sparse map established from previous observations, these inputs are used to estimate the world-to-camera pose $\boldsymbol{T}_t \in SE(3)$, which transforms a world point $\boldsymbol{x}^{\mathrm{w}} = [x^{\mathrm{w}}, y^{\mathrm{w}}, z^{\mathrm{w}}, 1]^\top$ into the current camera frame as $\boldsymbol{x}_t^{\mathrm{c}} = \boldsymbol{T}_t \boldsymbol{x}^{\mathrm{w}}$, where $\boldsymbol{x}_t^{\mathrm{c}} = [x_t^{\mathrm{c}}, y_t^{\mathrm{c}}, z_t^{\mathrm{c}}, 1]^\top$. At initialization, the first camera defines the reconstruction frame, with $\boldsymbol{T}_1 = \mathbf{I}_4$. Feature locations with valid predicted depth are back-projected to initialize the sparse map. We retain ORB-SLAM3's feature extraction, matching, and local bundle adjustment, while specifying how predicted depth contributes to pose estimation. In particular, depth enters the tracking objective as a direct observation with a constant relative-error noise model.

Camera pose estimation with a direct depth observation. Each frame extracts N_f=1200 oriented FAST and rotated BRIEF (ORB) image features on an 8-level pyramid. Following ORB-SALM3, we obtain a set of 2D–3D correspondences $\mathbb{M}_t$ between image features and local map points using projection-guided matching. Keeping the map points fixed, we refine the current pose by minimizing a robust combination of image reprojection and depth residuals:

$$\boldsymbol{T}_t = \arg\min_{\boldsymbol{T}\in SE(3)} \sum_{j\in\mathbb{M}_t} \rho\big(\boldsymbol{r}_j(\boldsymbol{T})^\top \boldsymbol{\Sigma}_j^{-1} \boldsymbol{r}_j(\boldsymbol{T})\big)\,, \qquad \boldsymbol{r}_j(\boldsymbol{T}) = \begin{bmatrix} u_j - \pi_u(\boldsymbol{T}\boldsymbol{x}_j^{\mathrm{w}}) \\ v_j - \pi_v(\boldsymbol{T}\boldsymbol{x}_j^{\mathrm{w}}) \\ z_j - [\boldsymbol{T}\boldsymbol{x}_j^{\mathrm{w}}]_z \end{bmatrix}. \tag{1}$$

where π is the pinhole projection with the frozen $\boldsymbol{K}$, $[\cdot]_z$ extracts the third coordinate, $z_j = \mathbf{D}_t(u_j, v_j)$ is the model's depth at the keypoint, ρ is the Huber kernel, and

$$\boldsymbol{\Sigma}_j = \mathrm{diag}\big(\sigma_j^2,\ \sigma_j^2,\ (\varepsilon\, z_j)^2\big), \qquad \varepsilon = 0.05, \tag{2}$$

with σ_j the pyramid-level pixel standard deviation. Pose estimation alternates between optimization and outlier re-classification. In each of four outer rounds, the solver refines the current pose using the active feature correspondences, recomputes their residuals and updates their inlier status according to the weighted squared errors. This procedure reduces the influence of incorrect matches on subsequent optimization rounds. We retain the image-plane reprojection residuals and their analytical pose Jacobians from ORB-SLAM3. ORB-SLAM3 represents RGB-D measurements as virtual stereo observations, allowing it to reuse the stereo observation model and optimization framework, but this modeling choice becomes less natural when depth is supplied directly by a feed-forward geometry model rather than inferred from stereo disparity; more details are in Appendix A.1. For the predicted depth, we instead adopt a constant-relative-uncertainty model and use a direct depth residual with the corresponding uncertainty weight. Our modification replaces the third, virtual-stereo residual with a direct depth residual and updates its Jacobian and information weight accordingly. We therefore describe only the modified depth component below.

For a matched map point $\boldsymbol{x}_j^{\mathrm{w}}$, let $z_j = \mathbf{D}_t(u_j, v_j)$ denote the observed valid depth at its matched 2D image correspondence. Given a camera pose $\boldsymbol{T}$, the reprojected camera-space depth is $[\boldsymbol{T}\boldsymbol{x}_j^{\mathrm{w}}]_z$. The direct depth residual is

$$r_{z,j}(\boldsymbol{T}) = z_j - [\boldsymbol{T}\boldsymbol{x}_j^{\mathrm{w}}]_z. \tag{3}$$

To linearize this residual around the current pose, we introduce a local pose increment $\delta\boldsymbol{\xi} \in \mathbb{R}^6$ and apply a left perturbation $\boldsymbol{T}(\delta\boldsymbol{\xi}) = \exp((\delta\boldsymbol{\xi})^\wedge)\,\boldsymbol{T}$, where

$$\delta\boldsymbol{\xi} = \begin{bmatrix} \delta\boldsymbol{\omega} \\ \delta\boldsymbol{v} \end{bmatrix}, \qquad (\delta\boldsymbol{\xi})^\wedge = \begin{bmatrix} [\delta\boldsymbol{\omega}]_\times & \delta\boldsymbol{v} \\ \mathbf{0}^\top & 0 \end{bmatrix} \in \mathfrak{se}(3). \tag{4}$$

Here, $\delta\boldsymbol{\omega}$ and $\delta\mathbf{v}$ respectively denote the rotational and translational increments, following (Solà et al., 2018; Campos et al., 2021) . The operator $(\cdot)^\wedge$ maps a six-dimensional increment to its matrix representation in $SE(3)$, and $[\cdot]_\times$ denotes the skew-symmetric cross-product matrix. The observed depth and map point remain fixed during pose-only optimization, so only the reprojected camera-space depth depends on the increment. The corresponding Jacobian is therefore

$$\boldsymbol{J}_{z,j} = \left.\frac{\partial r_{z,j}\big(\boldsymbol{T}(\delta\boldsymbol{\xi})\big)}{\partial\delta\boldsymbol{\xi}}\right|_{\delta\boldsymbol{\xi}=\mathbf{0}} = -\left.\frac{\partial[\boldsymbol{T}(\delta\boldsymbol{\xi})\boldsymbol{x}_j^{\mathrm{w}}]_z}{\partial\delta\boldsymbol{\xi}}\right|_{\delta\boldsymbol{\xi}=\mathbf{0}}. \tag{5}$$

This Jacobian describes how a small correction to the current pose changes the depth residual. The depth Jacobian is combined with the image-plane reprojection Jacobians to form the full residual Jacobian. Together with the information matrix $\boldsymbol{\Sigma}_j^{-1}$ and the Huber robust weights, these Jacobians define the linearized pose optimization problem. After solving for the increment $\delta\boldsymbol{\xi}$, we update the pose as $\boldsymbol{T} \leftarrow \exp((\delta\boldsymbol{\xi})^\wedge)\,\boldsymbol{T}$. This update is repeated within the optimization rounds described above, while keeping the map points and depth observations fixed like ORB-SLAM3.

3.2 Gaussian Mapper

The mapper incrementally reconstructs the scene as a set of 3D Gaussians,

$$\mathbb{G} = \{(\boldsymbol{\mu}_i, \boldsymbol{q}_i, \boldsymbol{s}_i, o_i, \boldsymbol{c}_i)\}_{i=1}^{N_{\mathrm{G}}}, \tag{6}$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^3$ is the center, $\boldsymbol{q}_i \in \mathbb{R}^4$ is a unit quaternion, $\boldsymbol{s}_i \in \mathbb{R}^3$ contains the principal-axis scales, $o_i \in (0,1)$ is the opacity, and $\boldsymbol{c}_i \in (0,1)^3$ is the RGB color. The map is updated at selected keyframes through render-aware Gaussian insertion followed by joint refinement with keyframe replay following OTF (Meuleman et al., 2025). All updates use only the current observation and previously stored keyframes.

Laplacian-guided Gaussian insertion. New Gaussians are allocated preferentially to image structure that is insufficiently represented by the current map. Let $\hat{\mathbf{I}}_t$ denote the rendered RGB image at pose $\boldsymbol{T}_t$, and $\boldsymbol{p} = (u, v)^\top$ denote the pixel. We use $\ell(\hat{\mathbf{I}}_t)$ to denote the absolute discrete Laplacian of the grayscale image, normalized by its maximum, with a zero output for a zero Laplacian. The insertion weights are

$$w_t(\boldsymbol{p}) = \Big[\ell(\mathbf{I}_t)(\boldsymbol{p}) - \ell(\hat{\mathbf{I}}_t)(\boldsymbol{p})\Big]^+, \tag{7}$$

where $[x]^+ = \max(0, x)$. This score favors valid pixels whose observed high-frequency response exceeds that of the rendering. It therefore provides a heuristic for identifying missing or under-resolved detail, rather than a complete measure of photometric disagreement. For an empty map, $\hat{\mathbf{I}}_t$ is zero, and the weights reduce to the observed image Laplacian restricted to valid depth.

Geometric initialization. Each sampled pixel $\boldsymbol{p}$ with depth $z = \mathbf{D}_t(\boldsymbol{p})$ initializes a Gaussian. Let $\bar{\boldsymbol{p}} = (u, v, 1)^\top$ denote its homogeneous image coordinate. The Gaussian center is obtained by back-projecting the depth and transforming the resulting point into the reconstruction frame. The remaining attributes are initialized as

$$\boldsymbol{q} = (1,0,0,0)^\top, \qquad \boldsymbol{s} = \frac{\kappa z}{f_x}\mathbf{1}_3, \qquad \kappa = 2, \qquad o = 0.07, \qquad \boldsymbol{c} = \mathbf{I}_t(u, v, :). \tag{8}$$

The scale follows the pixel-footprint initialization of OTF, with a nominal image-space scale of two pixels. Although initialized isotropically, the three scale components are optimized independently. The low initial opacity permits newly inserted primitives to make a limited contribution before refinement.

Refinement with keyframe replay. After inserting new Gaussians, the mapper refines the Gaussian map for $n = 30$ optimization steps. At each step, the newest keyframe is selected with probability $p_{\text{last}} = 0.2$ through a dedicated sampling branch; otherwise, a keyframe is sampled uniformly from all stored keyframes, including the newest.

For the sampled keyframe j, the differentiable `gsplat` rasterizer (Ye et al., 2025) produces a color image $\hat{\mathbf{I}}_j$ and a rendered depth map $\hat{\mathbf{D}}_j$. We minimize

$$\mathcal{L}_j = 0.8\,\mathcal{L}_{\text{rgb},j} + 0.2\,\Big(1 - \text{SSIM}(\hat{\mathbf{I}}_j, \mathbf{I}_j)\Big) + \lambda_d \mathcal{L}_{\text{depth},j}, \qquad \lambda_d = 0.01. \tag{9}$$

where $\mathcal{L}_{\text{rgb},j}$ is the mean absolute RGB reconstruction error. The depth loss compares the rendered depth with the stored model-predicted depth prior $\mathbf{D}_j$ in inverse-depth space:

$$\mathcal{L}_{\text{depth},j} = \frac{1}{|\mathbb{V}_j|} \sum_{\boldsymbol{p} \in \mathbb{V}_j} \left| \frac{1}{\hat{\mathbf{D}}_j(\boldsymbol{p})} - \frac{1}{\mathbf{D}_j(\boldsymbol{p})} \right|, \tag{10}$$

where $\mathbb{V}_j$ denotes the set of valid depth-supervision pixels for keyframe j. For small depth errors, the inverse-depth discrepancy is approximately the absolute depth error divided by the squared reference depth. This emphasizes nearby geometry relative to equally sized absolute depth errors at greater distances. Together with the photometric terms, the depth loss encourages appearance refinement to remain consistent with the predicted scene geometry.

4 Experiment

4.1 Setup and Evaluation

Datasets. We evaluate on four datasets, including indoor and outdoor datasets. *7-Scenes* (Shotton et al., 2013) has seven indoor RGB sequences with ground truth Kinect depth. *Oxford Spires* (Tao

et al., 2026) consists of six outdoor clips of 1000–2001 frames selected from outdoor sites: Keble College-(02,03,04), Observatory Quarter-01, Christ Church-02, and Bodleian Library-02, with poses and depth reprojected from the point cloud of a terrestrial laser scan. From *RE10K* (Zhou et al., 2018) we select 60 short indoor and outdoor real-estate clips with 71–279 frames. Eleven handheld sequences (fr1, fr2, fr3) are selected from the *TUM-D* (Sturm et al., 2012) dataset. All frames are processed at 518x392 resolution across all datasets. We evaluate all methods using a single NVIDIA RTX4090 GPU with an AMD Ryzen Threadripper PRO 3995WX 64-Cores CPU.

Evaluation metrics. We evaluate novel-view synthesis (NVS) quality on held-out views using PSNR, SSIM, and LPIPS. We measure global trajectory accuracy using the root-mean-square Absolute Trajectory Error (ATE) after aligning the estimated trajectory to the reference trajectory with a Sim(3) Umeyama alignment (Umeyama, 1991). We also report the Relative Pose Error (RPE) between consecutive frames, including translational error (RPE_t) and rotational error (RPE_R). To evaluate reconstruction geometry, we compute the mean L1 error between the rendered and ground-truth depth on held-out views. Finally, we report end-to-end throughput in frames per second (FPS), calculated by dividing the number of frames in each sequence by its total reconstruction time.

Baselines. We compare against MonoGS (Matsuki et al., 2024), S3PO-GS (Cheng et al., 2025), Photo-SLAM (Huang et al., 2024), and OTF (Meuleman et al., 2025). MonoGS uses 3D Gaussians for monocular tracking and mapping, while S3PO-GS targets monocular SLAM with globally scale-consistent Gaussian point maps. Photo-SLAM combines feature-based tracking with Gaussian splatting for photorealistic mapping, whereas OTF incrementally reconstructs large-scale scenes from unposed image sequences with large parallax. All baselines are evaluated using their publicly available implementations on identical input frames. To compare with Photo-SLAM fairly, we increase its mapping iterations from 8 to a maximum of 200 while retaining concurrent tracking and mapping, maintaining a processing rate of $\sim$6 FPS.

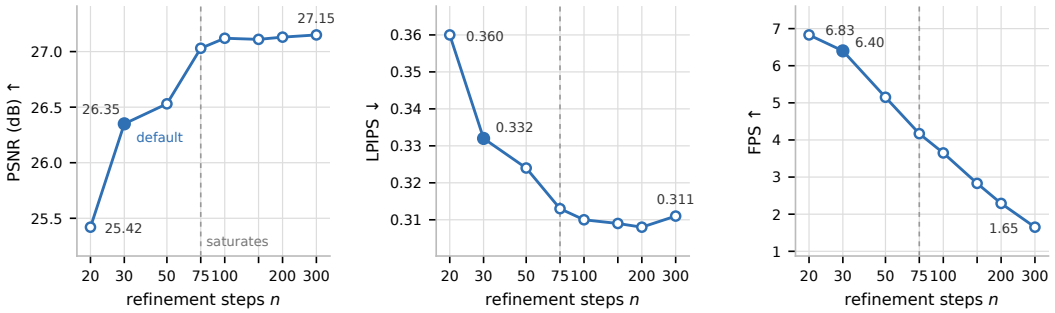


Figure 2: Ablation on refinement steps per keyframe. Means over the six Oxford Spires sequences. Rendering quality saturates at $n{\approx}75$–100 (dashed line). The filled marker is the default $n{=}30$ used in all main tables.

4.2 Novel View Synthesis and Tracking

Table 1 shows that our method consistently achieves the best rendering quality on all datasets in terms of NVS metrics. It also maintains competitive trajectory accuracy, obtaining the best ATE on Oxford Spires and TUM-D, and the second-best ATE on 7-Scenes and RE10K. The few cases in which Photo-SLAM or S3PO-GS achieves slightly lower tracking error are consistent with their larger computational budgets for pose estimation: Photo-SLAM performs extensive local-map maintenance when they have larger mapping iterations, whereas S3PO-GS employs a computationally intensive tracking procedure that aligns each frame with its Gaussian point map. The qualitative comparisons in Figure 3 further show that our reconstructions preserve finer structures and sharper appearance, while several baselines exhibit noticeable blurring, artifacts, or geometric inconsistency. For 7-Scenes, Oxford Spires and TUM-D, we additionally visualize rendered depth. The 7-Scenes and TUM-D references provide dense depth, whereas the Oxford Spires reference is obtained by projecting the scanned point cloud and is therefore sparse. For a fair comparison, the same valid-depth mask is applied to all methods. The trajectory examples in Figure 4 confirm that our estimates remain among the most accurate across three datasets, omitting RE10K because it is too short, with particularly clear improvements on the Oxford Spires and TUM-D sequences.

Table 1: Rendering quality, trajectory accuracy, and efficiency among all datasets. The **best** results are in **bold**, and the second-best results are underlined. ATE and RPE_t are reported in centimeters, and RPE_R is reported in degrees. For Photo-SLAM, it fails on the Pumpkin sequence of 7-Scenes and we only use 6 scenes for its average.

Method	(a) 7-Scenes							(b) Oxford Spires						
	PSNR↑	SSIM↑	LPIPS↓	ATE↓	RPE_t ↓	RPE_R ↓	FPS↑	PSNR↑	SSIM↑	LPIPS↓	ATE↓	RPE_t ↓	RPE_R ↓	FPS↑
MonoGS	18.94	0.681	0.501	29.90	0.776	0.290	3.28	20.40	0.635	0.571	595.47	10.58	0.321	3.21
S3PO-GS	19.76	0.713	0.470	**7.30**	**0.480**	0.213	1.74	21.63	0.699	0.506	287.81	2.802	0.134	1.31
Photo-SLAM	21.91	0.741	0.400	19.78	0.566	0.205	5.99	22.05	0.717	0.465	567.71	2.308	**0.074**	5.99
OTF	20.50	0.744	0.443	39.34	0.801	0.269	**7.57**	23.19	0.713	0.466	624.06	8.592	0.231	**8.24**
Ours	**24.48**	**0.839**	**0.293**	8.58	0.524	**0.189**	6.99	**26.35**	**0.817**	**0.332**	**73.81**	**0.943**	0.095	6.40

Method	(c) RE10K							(d) TUM-D						
	PSNR↑	SSIM↑	LPIPS↓	ATE↓	RPE_t ↓	RPE_R ↓	FPS↑	PSNR↑	SSIM↑	LPIPS↓	ATE↓	RPE_t ↓	RPE_R ↓	FPS↑
MonoGS	22.97	0.793	0.309	10.56	0.726	0.113	1.21	14.31	0.481	0.601	92.17	0.94	0.571	3.33
S3PO-GS	23.12	0.797	0.298	3.47	**0.553**	0.130	1.29	17.00	0.621	0.557	56.57	0.74	0.387	1.30
Photo-SLAM	23.86	0.802	0.287	**1.90**	0.647	**0.085**	5.94	13.68	0.496	0.601	59.96	0.89	0.330	5.99
OTF	25.20	0.859	0.245	8.79	1.064	0.150	6.35	19.58	0.708	0.462	88.55	0.94	0.323	**6.54**
Ours	**25.73**	**0.873**	**0.210**	2.84	0.645	0.088	**7.94**	**20.89**	**0.758**	**0.387**	**41.33**	**0.66**	**0.288**	5.01

Depth reconstruction accuracy. Table 2 complements the RGB evaluation by measuring the rendered depth. Our method achieves the lowest mean L1 error on all three datasets, demonstrating consistent geometric improvements across both indoor and outdoor environments. Compared with S3PO-GS, the strongest baseline across all three datasets, our method reduces the depth error by 11.8% on 7-Scenes, 42.1% on TUM-D, and 28.9% on Oxford Spires, based on the rounded values reported in the table. Compared with Photo-SLAM, the error decreases from 0.36 to 0.15 m on 7-Scenes, from 1.66 to 0.33 m on TUM-D, and from 8.54 to 3.18 m on Oxford Spires. Notably, OTF achieves competitive novel-view rendering quality, but exhibits the highest depth error on all three datasets. This discrepancy highlights a limitation of assessing 3D Gaussian maps through RGB rendering alone: visually plausible novel views do not necessarily imply geometrically accurate reconstruction, since the observed appearance can still be explained by Gaussians that are incorrectly distributed in 3D space.

Table 2: **Scale-aligned rendered-depth accuracy.** Mean depth L1 error on 7-Scenes, TUM-D, and Oxford Spires. All values are reported in meters; lower is better.

Method	7-Scenes↓	TUM-D↓	Oxford Spires↓
MonoGS	0.52	1.29	9.12
S3PO-GS	0.17	0.57	4.47
Photo-SLAM	0.36	1.66	8.54
OTF	1.00	2.35	14.12
Ours	**0.15**	**0.33**	**3.18**

4.3 Ablation study

Ablation on camera tracking. Table 3 isolates the effect of the pose on the reconstruction. Four alternatives are compared with our solver: (A) Virtual stereo, the RGB-D front-end in ORB-SLAM3, which turns the predicted depth into a virtual right-image coordinate and optimizes the stereo residual shown in Eq. 13; (B) 2D reprojection, the same front-end with depth used only to back-project the new map points, so that tracking sees the 2D reprojection residual only, like the monocular RGB ORB-SLAM3; (C) a purely geometric point-to-plane ICP of the predicted depth against a local model and (D) the backbone's own regressed camera head. Compared with virtual stereo (A), our direct-depth residual produces more accurate camera poses while preserving the same PSNR: ATE decreases from 79.5 to 73.8 cm, while translational RPE decreases from 1.123 to 0.943 cm. This result indicates that directly modeling the uncertainty of predicted depth provides a more effective geometric constraint than encoding it through a virtual-stereo observation. The comparison with 2D reprojection (B) further demonstrates the importance of explicitly including depth in pose optimization. Removing the depth residual reduces PSNR by 3.45 dB, increases ATE from 73.8 to 162.1 cm, and increases translational RPE from 0.943 to 4.723 cm. Predicted depth is therefore useful not only for initializing map points, but also for constraining the camera pose at every frame. The purely unoptimized CPU ICP variant (C) is also substantially less accurate than our solver, increasing ATE

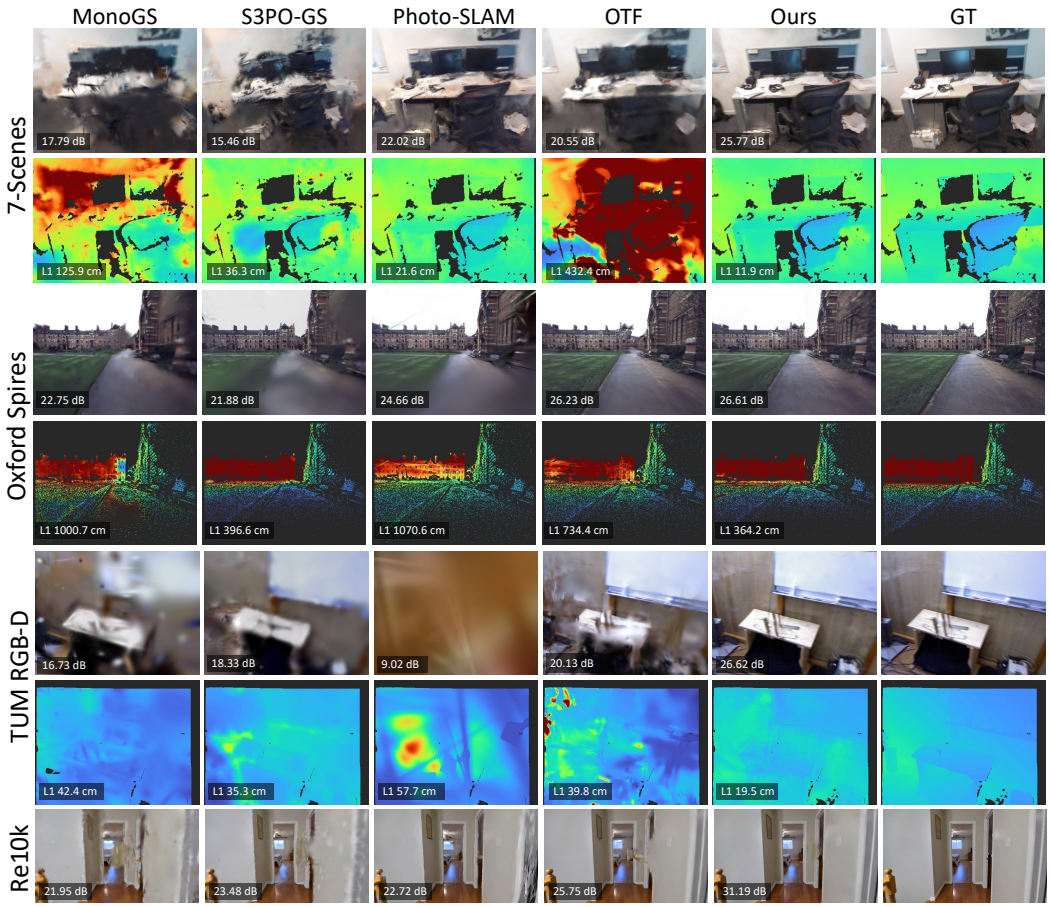


Figure 3: Qualitative rendering and depth comparison. One representative sequence is selected from each dataset. For 7-Scenes, TUM-D and Oxford Spires, we show both novel-view RGB rendering and rendered depth; RE10K is shown only in RGB because reference depth is unavailable. The 7-Scenes and TUM-D datasets provide dense reference depth, whereas the Oxford Spires reference depth is sparsely projected from the scanned point cloud. All methods are compared using the same valid-depth mask.

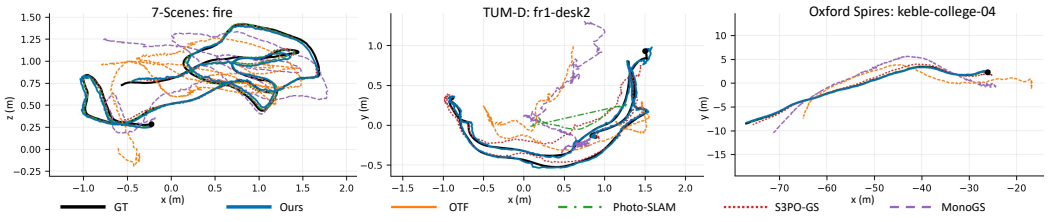


Figure 4: Qualitative trajectory comparison. We visualize one representative sequence from three datasets after trajectory alignment. Our estimated trajectories closely follow the ground truth. Because OTF is designed primarily for novel-view synthesis and estimates poses only for keyframes and held-out views, its trajectory error is computed only over this available pose subset rather than over all input frames.

from 73.8 to 307.2 cm, translational RPE from 0.943 to 15.14 cm, and rotational RPE from 0.095° to 0.482°. This result suggests that point-to-plane alignment alone is less robust to predicted-depth noise and weak geometric constraints. Finally, the backbone camera head (D) is fast but produces substantially less accurate trajectories, with 5.98 m ATE and 15.39 cm translational RPE. This sug-

Table 3: Ablation on camera pose estimation over Oxford Spires. For variant C, we use Open3D's CPU point-to-plane ICP as an unoptimized reference implementation; its runtime does not represent the computational limit of ICP.

Pose source	PSNR↑	SSIM↑	LPIPS↓	ATE (cm)↓	RPE$_t$ (cm)↓	RPE$_R$ (°)↓	Track (ms)↓	FPS↑
A: Virtual stereo	**26.35**	0.815	0.334	79.5	1.123	0.104	16.2	6.20
B: 2D reprojection	22.90	0.709	0.446	162.1	4.723	0.140	<u>13.9</u>	6.51
C: ICP	21.37	0.648	0.482	307.2	15.14	0.482	138.6	5.47
D: Camera head	20.23	0.629	0.510	598.5	15.39	0.259	**9.5**	**7.04**
Ours	**26.35**	**0.817**	**0.332**	**73.8**	**0.943**	**0.095**	16.1	6.40

Table 4: Geometry-based pose estimation across streaming backbones on Keble College-02.

Backbone	Pose source	ATE (m)↓	RPE$_t$ (cm)↓	RPE$_R$ (°)↓	Track (ms)↓
XStreamVGGT (Su et al., 2026)	Camera head	23.277	48.95	0.4	**9.6**
	Geometry solver	**0.773**	**1.44**	**0.085**	16.1
STream3R (Lan et al., 2026)	Camera head	25.560	17.67	0.319	**10.6**
	Geometry solver	**2.332**	**1.74**	**0.098**	16.1
TTT3R (Chen et al., 2026b)	Camera head	10.57	6.52	0.204	**4.2**
	Geometry solver	**7.29**	**2.33**	**0.09**	16.1
Lingbot-map (Chen et al., 2026a)	Camera head	8.224	13.74	0.202	**9.5**
	Geometry solver	**1.224**	**0.758**	**0.059**	16.1

gests that its regressed poses are not as tightly coupled to the predicted scene geometry as those obtained through explicit geometric optimization. Our solver adds only 6.6 ms of tracking time relative to the camera head, while reducing ATE by 87.6% and translational RPE by 93.9%. The end-to-end throughput remains comparable, decreasing only from 7.04 to 6.40 FPS.

Ablation on backbones. Table 4 shows that replacing the regressed camera head with the geometry solver consistently improves all trajectory metrics across the four streaming backbones on Keble College-02. The solver reduces ATE by 31.0–96.7%, translational RPE by 64.3–97.1%, and rotational RPE by 55.9–78.8%. These gains require only 16.1 ms of tracking per frame, corresponding to an additional 5.5–11.9 ms over the camera heads. The results demonstrate that our solver transfers across different backbone architectures and produces camera motion that is more consistent with their predicted scene geometry at modest computational cost.

Ablation on mapping iterations. The number of synchronous refinement steps per keyframe is the one mapper parameter that trades quality against speed. We show the ablation in Figure 2. Quality improves quickly and then saturates: $20\rightarrow30$ steps gain 0.93 dB, $30\rightarrow75$ a further $\sim$0.88 dB, and everything beyond 100 steps changes only marginally (27.11–27.15 dB). We report $n{=}30$ throughout as the speed-oriented operating point.

5 Conclusion

We present LingGS, an online monocular Gaussian SLAM system that decouples feed-forward geometry prediction from camera tracking. A frozen streaming 3D foundation model serves as a virtual RGB-D sensor, while an explicit geometric solver estimates camera poses by jointly minimizing image reprojection and direct depth residuals. Combined with incremental Gaussian mapping, this design enables high-fidelity reconstruction. Experiments across indoor and outdoor datasets demonstrate improved rendering quality and geometric accuracy while maintaining efficient processing and competitive tracking accuracy. Further experiments across multiple streaming backbones support the broader applicability of our geometric solver.

Our current system does not address loop closure or dynamic scenes. Its reliance on ORB feature extraction and matching can also compromise tracking in repetitive or weakly textured environments, where correspondences may be ambiguous or reliable features scarce. Addressing these limitations with learned prior is an important direction for improving long-term consistency and robustness. More broadly, our findings motivate future streaming 3D models that integrate learned geometric priors with explicit geometric optimization to improve pose accuracy and geometric consistency in online reconstruction.

REFERENCES

Lizhi Bai, Chunqi Tian, Jun Yang, Siyu Zhang, Masanori Suganuma, and Takayuki Okatani. RP-SLAM: Real-time photorealistic SLAM with efficient 3D Gaussian splatting. *IEEE Transactions on Visualization and Computer Graphics*, 32(2):1452–1466, 2026.

Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

Lin-Zhuo Chen, Jian Gao, Shangzhan Zhang, Yihang Chen, Ka Leong Cheng, Yipengjing Sun, Liangxiao Hu, Nan Xue, Xing Zhu, Yujun Shen, Yao Yao, and Yinghao Xu. LingBot-Map: Geometric context transformer for streaming 3D reconstruction. *arXiv preprint arXiv:2604.14141*, 2026a.

Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. TTT3R: 3D reconstruction as test-time training. In *ICLR*, 2026b.

Chong Cheng, Sicheng Yu, Zijian Wang, Yifan Zhou, and Hao Wang. Outdoor monocular SLAM with global scale-consistent 3D Gaussian pointmaps. In *ICCV*, pp. 26035–26044, 2025.

Kai Deng, Yigong Zhang, Jian Yang, and Jin Xie. GigaSLAM: Large-scale monocular SLAM with hierarchical Gaussian splats. In *SIGGRAPH Asia Conference Papers*, pp. 1–10, 2025.

Dapeng Feng, Zhiqiang Chen, Yizhen Yin, Shipeng Zhong, Yuhua Qi, and Hongbo Chen. CaRtGS: Computational alignment for real-time Gaussian splatting SLAM. *IEEE Robotics and Automation Letters*, 10(5):4340–4347, 2025.

Ruilan Gao, Letian Jin, and Yu Zhang. GeoGS-SLAM: Online monocular reconstruction using Gaussian splatting with geometric priors. *arXiv preprint arXiv:2607.11184*, 2026.

Seongbo Ha, Jiung Yeon, and Hyeonwoo Yu. RGBD GS-ICP SLAM. In *ECCV*, pp. 180–197, 2024.

Yuhang Han, Hao Wang, Jiaxi Cao, and Xingyu Liu. VGGT-GS SLAM: Uncalibrated monocular Gaussian splatting SLAM with feed-forward priors. *arXiv preprint arXiv:2609.19628*, 2026.

Christian Homeyer, Leon Begiristain, and Christoph Schnörr. Droid-splat combining end-to-end slam with 3d gaussian splatting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2025.

Yan Song Hu, Nicolas Abboud, Muhammad Qasim Ali, Adam Srebrnjak Yang, Imad Elhajj, Daniel Asmar, Yuhao Chen, and John S. Zelek. MGSO: Monocular real-time photometric SLAM with efficient 3D Gaussian splatting. In *ICRA*, pp. 11061–11067, 2025a.

Yue Hu, Rong Liu, Meida Chen, Peter Beerel, and Andrew Feng. SplatMAP: Online dense monocular SLAM with 3D Gaussian splatting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 8(1):1–18, 2025b.

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 *CVPR*, 2024.

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 *CVPR*, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph. (SIGGRAPH)*, 42(4), 2023.

Evgenii Kruzhkov and Sven Behnke. MonoEM-GS: Monocular expectation-maximization Gaussian splatting SLAM. *arXiv preprint arXiv:2604.10593*, 2026.

Yushi Lan, Yihang Luo, Fangzhou Hong, Shangchen Zhou, Honghua Chen, Zhaoyang Lyu, Bo Dai, Shuai Yang, Chen Change Loy, and Xingang Pan. Stream3r: Scalable sequential 3d reconstruction with causal transformer. In *ICLR*, pp. 42746–42768, 2026.

Guanghao Li, Kerui Ren, Linning Xu, Zhewen Zheng, Changjian Jiang, Xin Gao, Bo Dai, Jian Pu, Mulin Yu, and Jiangmiao Pang. ARTDECO: Toward high-fidelity on-the-fly reconstruction with hierarchical Gaussian structure and feed-forward guidance. In *ICLR*, pp. 102997–103029, 2026.

Mingrui Li, Yiming Zhou, Hongxing Zhou, Xinggang Hu, Florian Roemer, Hongyu Wang, and Ahmad Osman. Dy3DGS-SLAM: Monocular 3D Gaussian splatting SLAM for dynamic environments. In *ICRA*, pp. 14572–14578, 2025a.

Yang Li, Jinglu Wang, Lei Chu, Xiao Li, Shiu-hong Kao, Ying-Cong Chen, and Yan Lu. StreamGS: Online generalizable Gaussian splatting reconstruction for unposed image streams. In *ICCV*, pp. 25841–25850, 2025b.

Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian splatting SLAM. In *CVPR*, 2024.

Andreas Meuleman, Ishaan Shah, Alexandre Lanvin, Bernhard Kerbl, and George Drettakis. On-the-fly reconstruction for large-scale novel view synthesis from unposed images. *ACM Trans. Graph. (SIGGRAPH)*, 44(4), 2025.

Zhexi Peng, Tianjia Shao, Yong Liu, Jingke Zhou, Yin Yang, Jingdong Wang, and Kun Zhou. RTG-SLAM: Real-time 3D reconstruction at scale using Gaussian splatting. In *ACM SIGGRAPH Conference Papers*, 2024.

Kerui Ren, Guanghao Li, Changjian Jiang, Yingxiang Xu, Tao Lu, Linning Xu, Junting Dong, Jiangmiao Pang, Mulin Yu, and Bo Dai. M^3: Dense matching meets multi-view foundation models for monocular Gaussian splatting SLAM. *arXiv preprint arXiv:2603.16844*, 2026.

Erik Sandström, Keisuke Tateno, Michael Oechsle, Michael Niemeyer, Luc Van Gool, Martin R. Oswald, and Federico Tombari. Splat-SLAM: Globally optimized RGB-only SLAM with 3D Gaussians. *arXiv preprint arXiv:2405.16544*, 2024.

Dong-Uk Seo, Jinwoo Jeon, Eungchang Mason Lee, and Hyun Myung. GaussianFlow SLAM: Monocular Gaussian splatting SLAM guided by GaussianFlow. *IEEE Robotics and Automation Letters*, 11(6):7644–7651, 2026.

Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, pp. 2930–2937, 2013.

Joan Solà, Jeremie Deray, and Dinesh Atchuthan. A micro Lie theory for state estimation in robotics. *arXiv preprint arXiv:1812.01537*, 2018.

Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IROS*, pp. 573–580, 2012.

Zunhai Su, Weihao Ye, Hansen Feng, Keyu Fan, Jing Zhang, Dahai Yu, Zhengwu Liu, and Ngai Wong. XStreamVGGT: Extremely memory-efficient streaming vision geometry grounded transformer with KV cache compression. *Journal of the Society for Information Display*, 34(5):535–545, 2026.

Yifu Tao, Miguel Ángel Muñoz-Bañón, Lintong Zhang, Jiahao Wang, Lanke Frank Tarimo Fu, and Maurice Fallon. The Oxford Spires dataset: Benchmarking large-scale LiDAR-visual localisation, reconstruction and radiance field methods. *International Journal of Robotics Research*, 45(6): 839–857, 2026.

Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

Tianci Wen, Zhiang Liu, and Yongchun Fang. SEGS-SLAM: Structure-enhanced 3D Gaussian splatting SLAM with appearance embedding. In *ICCV*, pp. 28103–28113, 2025.

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 *CVPR*, 2024.

Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for Gaussian splatting. *Journal of Machine Learning Research*, 26(34):1–17, 2025.

Sicheng Yu, Chong Cheng, Yifan Zhou, Xiaojun Yang, and Hao Wang. RGB-only Gaussian splatting SLAM for unbounded outdoor scenes. In *ICRA*, pp. 11068–11074, 2025.

Sicheng Yu, Dongxu Shen, Beizhen Zhao, Guanzhi Ding, and Hao Wang. Robust and efficient monocular 3D Gaussian SLAM for kilometer-scale outdoor scenes. *arXiv preprint arXiv:2606.30436*, 2026.

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

Wei Zhang, Qing Cheng, David Skuddis, Niclas Zeller, Daniel Cremers, and Norbert Haala. HI-SLAM2: Geometry-aware Gaussian SLAM for fast monocular scene reconstruction. *IEEE Transactions on Robotics*, 41:6478–6493, 2025.

Zicheng Zhang, Ke Wu, Xiangting Meng, Keyu Liu, Jieru Zhao, and Wenchao Ding. Flash-Mono: Feed-forward accelerated Gaussian splatting monocular SLAM. In *ICLR*, 2026.

Jianhao Zheng, Zihan Zhu, Valentin Bieri, Marc Pollefeys, Songyou Peng, and Iro Armeni. WildGS-SLAM: Monocular Gaussian splatting SLAM in dynamic environments. In *CVPR*, 2025.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph. (SIGGRAPH)*, 37(4), 2018.

Fan Zhu, Ziyu Chen, Zhenjun Zhao, Zhisong Xu, Hui Zhu, Mingrui Li, Chunmao Jiang, and Javier Civera. Mygo-splat: Multi-objective closed-loop geometric feedback for RGB-only Gaussian SLAM. *arXiv preprint arXiv:2606.29738*, 2026.

Liyuan Zhu, Yue Li, Erik Sandström, Shengyu Huang, Konrad Schindler, and Iro Armeni. Loop-Splat: Loop closure by registering 3D Gaussian splats. In *3DV*, 2025.

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. *IEEE Robotics and Automation Letters*, 9(11):9486–9493, 2024.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *ICLR*, pp. 88055–88072, 2026.

A Appendix

A.1 Relation to ORB-SLAM3's Virtual-Stereo Model

ORB-SLAM3 handles RGB-D observations by converting depth measurements into virtual right-image coordinates and reusing its stereo reprojection model. Although the information matrix is defined in pixel coordinates and does not explicitly depend on depth, the projection model introduces a depth-dependent sensitivity to depth errors. We first elucidate the visual-stereo mechanism in ORB-SLAM3 and compare it with our direct depth residual.

Calibrated stereo geometry and observations. Consider a calibrated pinhole stereo pair with shared calibrated intrinsics (f_x, f_y, c_x, c_y) and baseline $b > 0$. The right camera center lies at $(b, 0, 0)$ in the left camera frame, and the two camera frames have aligned axes.

Let $\boldsymbol{x}_j^{\mathrm{w}}$ denote a map point in homogeneous world coordinates. Under a world-to-left-camera pose $\boldsymbol{T} \in SE(3)$, the point in left camera space is defined as:

$$\boldsymbol{x}_j^{\mathrm{c}} = \boldsymbol{T}\boldsymbol{x}_j^{\mathrm{w}} = [x_c, y_c, z_c, 1]^\top, \qquad z_c > 0. \tag{11}$$

The same point has coordinates $[x_c - b, y_c, z_c, 1]^\top$ in the right camera frame. Here, b and (x_c, y_c, z_c) are expressed in the same length unit.

The predicted image coordinates are

$$\begin{aligned}\hat{u}_{L,j} &= f_x \frac{x_c}{z_c} + c_x, \\ \hat{v}_{L,j} &= f_y \frac{y_c}{z_c} + c_y, \\ \hat{u}_{R,j} &= f_x \frac{x_c - b}{z_c} + c_x = \hat{u}_{L,j} - \frac{f_x b}{z_c}.\end{aligned} \tag{12}$$

For a matched stereo feature, let $(u_{L,j}, v_{L,j})$ denote its observed coordinates in the left image and $u_{R,j}$ its observed horizontal coordinate in the right image. Rectification makes corresponding points share the same vertical coordinate in the ideal model. ORB-SLAM3, therefore, uses the three-component observation $(u_{L,j}, v_{L,j}, u_{R,j})$ and the residual

$$\boldsymbol{r}_j^{\mathrm{st}}(\boldsymbol{T}) = \begin{bmatrix} u_{L,j} - \hat{u}_{L,j} \\ v_{L,j} - \hat{v}_{L,j} \\ u_{R,j} - \hat{u}_{R,j} \end{bmatrix}. \tag{13}$$

Explicit weighting in pixel coordinates. ORB-SLAM3 assigns the stereo observation a diagonal covariance model,

$$\boldsymbol{\Sigma}_j^{\mathrm{st}} = \sigma_j^2 \boldsymbol{I}_3, \qquad E_j^{\mathrm{st}} = \left(\boldsymbol{r}_j^{\mathrm{st}}\right)^\top \left(\boldsymbol{\Sigma}_j^{\mathrm{st}}\right)^{-1} \boldsymbol{r}_j^{\mathrm{st}}. \tag{14}$$

Here, σ_j is an assumed pixel localization standard deviation in base-level image coordinates. For a feature detected at pyramid level ℓ_j,

$$\sigma_j^2 = \sigma_0^2 s^{2\ell_j}, \qquad \sigma_0 = 1 \text{ pixel}, \tag{15}$$

where $s > 1$ is the pyramid scale factor and ℓ_j is zero-based. We use $s = 1.2$.

This covariance is assigned from the feature scale, rather than estimated from its current reprojection residual. It contains no explicit depth-dependent term. The weighted error E_j^{st} is used within the robust optimization and outlier-classification procedure. In particular, ORB-SLAM3 does not compute a separate depth variance and insert it into $\boldsymbol{\Sigma}_j^{\mathrm{st}}$.

RGB-D observations and the virtual right coordinate. For RGB-D inputs, the RGB image takes the role of the left image. Given a valid observed depth $z_j > 0$, ORB-SLAM3 constructs a virtual right-image coordinate:

$$u_{R,j} = u_{L,j} - \frac{f_x b}{z_j}. \tag{16}$$

The depth z_j denotes the coordinate along the camera's optical axis and uses the same length unit as the baseline. No physical right image is required.

Define the left-image reprojection residuals and the direct depth residual as

$$\begin{aligned} r_{u,j} &= u_{L,j} - \hat{u}_{L,j}, \\ r_{v,j} &= v_{L,j} - \hat{v}_{L,j}, \\ r_{z,j} &= z_j - z_c. \end{aligned} \tag{17}$$

The observed quantities are fixed, whereas the predicted coordinates and z_c depend on $\boldsymbol{T}$. For readability, this dependence is omitted from the scalar residual notation.

Substituting Eq. 16 into the right-image reprojection residual gives

$$\begin{aligned} r_{R,j} &= \left(u_{L,j} - \frac{f_x b}{z_j}\right) - \left(\hat{u}_{L,j} - \frac{f_x b}{z_c}\right) \\ &= r_{u,j} + f_x b \left(\frac{1}{z_c} - \frac{1}{z_j}\right) \\ &= r_{u,j} + \frac{f_x b}{z_j z_c} r_{z,j}. \end{aligned} \tag{18}$$

Writing

$$a_j(\boldsymbol{T}) = \frac{f_x b}{z_j z_c}, \tag{19}$$

the virtual-stereo residual becomes

$$\boldsymbol{r}_j^{\mathrm{VS}}(\boldsymbol{T}) = \begin{bmatrix} r_{u,j} \\ r_{v,j} \\ r_{u,j} + a_j r_{z,j} \end{bmatrix}. \tag{20}$$

Its third component combines horizontal reprojection error with an inverse-depth discrepancy. It is not a depth or disparity residual alone.

Depth sensitivity induced by the residual. Substituting Eq. 20 into the pixel-weighted cost yields the exact expression

$$E_j^{\mathrm{VS}} = \frac{1}{\sigma_j^2}\left[r_{u,j}^2 + r_{v,j}^2 + (r_{u,j} + a_j r_{z,j})^2\right]. \tag{21}$$

Expanding the square gives

$$E_j^{\mathrm{VS}} = \frac{2r_{u,j}^2 + r_{v,j}^2 + 2a_j r_{u,j} r_{z,j} + a_j^2 r_{z,j}^2}{\sigma_j^2}. \tag{22}$$

This expression explains how depth affects the cost despite the depth-independent pixel information matrix. The depth residual is multiplied by a_j before the residual is squared. Near a consistent estimate, $z_c \approx z_j \approx z$, so that

$$a_j \approx \frac{f_x b}{z^2}, \qquad \frac{a_j^2}{\sigma_j^2} \approx \frac{(f_x b)^2}{\sigma_j^2 z^4}. \tag{23}$$

Thus, at fixed calibration and feature pyramid level, the coefficient of the squared depth residual scales locally as z^{-4}.

This is a property induced by the residual parameterization, not an additional information weight explicitly assigned by ORB-SLAM3. Furthermore, Eq. 22 contains a horizontal–depth cross term. The full virtual-stereo cost therefore cannot be identified with an independent scalar depth penalty.

Comparison with direct-depth weighting. Our per-frame pose-only formulation uses the direct depth residual as the third component:

$$\boldsymbol{r}_j^{\mathrm{D}}(\boldsymbol{T}) = \begin{bmatrix} r_{u,j} \\ r_{v,j} \\ r_{z,j} \end{bmatrix}, \qquad \boldsymbol{\Sigma}_j^{\mathrm{D}} = \mathrm{diag}\big(\sigma_j^2, \sigma_j^2, (\varepsilon z_j)^2\big), \tag{24}$$

as defined in Eqs. 1 and 2. Its nominal weighted cost is

$$E_j^{\mathrm{D}} = \frac{r_{u,j}^2 + r_{v,j}^2}{\sigma_j^2} + \frac{r_{z,j}^2}{\varepsilon^2 z_j^2}. \tag{25}$$

Unlike the virtual-stereo formulation, the third residual does not contain the horizontal reprojection residual.

Here, the depth-dependent information weight is explicitly specified:

$$w_{z,j}^{\mathrm{D}} = \frac{1}{\varepsilon^2 z_j^2} \propto z_j^{-2}. \tag{26}$$

It corresponds to an assumed constant relative depth uncertainty, $\sigma_{z,j}^{\mathrm{D}}/z_j = \varepsilon$. We use $\varepsilon = 0.05$. The observed depth z_j remains fixed during pose-only optimization, so this weight is also fixed within each optimization problem.

Thus, the two formulations introduce depth dependence in different ways. Virtual stereo uses a pixel-domain weight together with a projection sensitivity that induces a local z^{-4} scaling. Our formulation directly penalizes depth error with an explicitly chosen z^{-2} weight. We evaluate direct-depth, virtual-stereo, and reprojection-only tracking in Sec. 4.3.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on research methodology and experiments and to assist with the implementation of methods. Additionally, we used generative AI tools for brainstorming, code editing, literature search, and improving the readability of the manuscript. All AI-assisted methodological suggestions, code, experimental results, and manuscript content were independently reviewed and verified by the authors. We take responsibility for the final content of this work.

ETHICS STATEMENT

This work focuses on visual SLAM and 3D scene reconstruction using publicly available research datasets. It does not involve human-subject studies or the collection of personally identifiable information. We are not aware of any specific ethical concerns beyond the general considerations associated with computer vision and 3D reconstruction technologies.

REPRODUCIBILITY STATEMENT

Our source code and configuration files will be publicly released upon acceptance of the paper. To facilitate reproducibility, we provide detailed descriptions of the proposed tracking and mapping pipelines, implementation details, experimental settings, and evaluation protocols in the main paper and appendix. Additional ablations and qualitative results are also provided to support the reported findings.