PROBEGS: CLOSED-FORM SCALE FOR ERROR-GUIDED GAUSSIAN DENSIFICATION

Anonymous authors
Paper under double-blind review

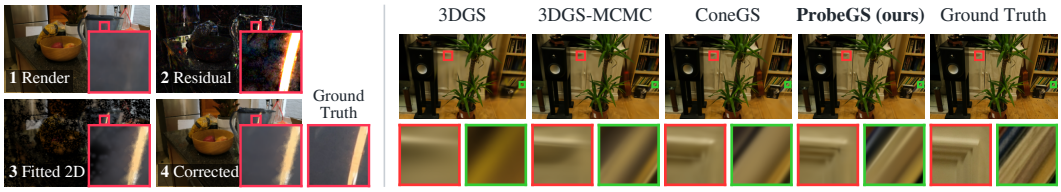

Figure 1: ProbeGS reads the scale and shape of new Gaussians from the rendering error. **Left:** on a Counter training view, 2D Gaussians (3) fitted to the residual (2) of the current render (1) correct it (4; ground truth beside its inset) and are lifted to 3D as new primitives; insets enlarge the boxed region. **Right:** full renders of a held-out Room view at a 100K budget, with the boxed regions enlarged below; ProbeGS keeps the layered cabinet molding edges and the book-spine bands that ConeGS (Baranowski et al., 2026) and non-error-guided baselines (Kerbl et al., 2023; Kheradmand et al., 2024) blur.

ABSTRACT

A small Gaussian budget makes 3D Gaussian Splatting (3DGS) viable on low-capacity devices, but only if every new Gaussian has the right position and size. The standard photometric loss averages over pixels and can leave small, high-residual regions uncovered. Yet the residual itself shows how large each correction should be: a broad residual calls for a wide Gaussian, a thin streak for a narrow, oriented one. ProbeGS therefore reads each new Gaussian's scale directly from the residual. It places corrective 2D Gaussians on the residual and lifts them to 3D. Around each pixel sampled in proportion to the residual, we measure how fast the residual decays on rings at several radii and set the standard deviation of the 2D Gaussian placed there in closed form, with no scale search. A short refinement moves only centers and colors, and a covariance lift inserts an ordinary 3DGS primitive that reproduces the fitted 2D Gaussian in its source view. We evaluate on Tanks & Temples, Deep Blending, and Mip-NeRF 360 at three requested Gaussian budgets. ProbeGS achieves state-of-the-art quality at 100K Gaussians, where it raises Deep Blending PSNR by 0.50 dB and lowers Tanks & Temples LPIPS by 0.041 over the next-best method.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) renders in real time, but its memory and rendering cost grow with the number of Gaussians. Deployment on low-capacity devices therefore fixes a small budget, and at that budget every new Gaussian must land in the right place at the right size. Standard densification splits or clones high-gradient Gaussians without locating the error or measuring its extent (Kerbl et al., 2023); later methods refine where primitives are added (Cheng et al., 2024; Zhang et al., 2024b; Rota Bulò et al., 2024; Baranowski et al., 2026) or how many are kept (Kheradmand et al., 2024).

A small budget also exposes a weakness of the training objective. The photometric loss averages over pixels, so a small region with a high residual adds little to the total and can stay uncovered after the rest of the image has converged. Correcting such a region requires knowing where the error is and how large a Gaussian it needs. Standard and error-driven densification (Kerbl et al., 2023;

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.

Rota Bulò et al., 2024) choose which Gaussians to split but derive the new Gaussians' scale from their parent. ConeGS (Baranowski et al., 2026) inserts Gaussians directly at high-error pixels but sizes each from a camera cone, independent of the residual's extent. Other error-guided methods target different settings: RTG-SLAM (Peng et al., 2024) adds Gaussians at high-error pixels during online mapping, CEM-4DGS (Kang et al., 2025) fits ellipses to clustered error pixels in dynamic scenes, and AuGS (Yang et al., 2026) fits 2D Gaussians to the residual of an already trained model. None is designed to replace densification during training, and our adaptations of the fitting-based ones take seconds per call (Table 3), too slow to run throughout training.

The residual itself shows the scale. A broad residual on a wall calls for a large isotropic Gaussian; a thin streak along a chair leg calls for a small oriented one. We therefore treat densification as placement on the residual map (Fig. 1): each corrective 2D Gaussian takes its scale and shape from the local error, and a short fit settles its position and color.

Each proposal is a 2D Gaussian centered at a pixel sampled in proportion to the residual. Robust measurements of how fast the residual decays on rings at several radii around that pixel set its scale in closed form (Sec. 4.2), with no iterative search. Where the Hessian of the log-residual is negative definite, it also elongates and orients the Gaussian. A short refinement then moves only centers and colors; this constrained fit outperforms a free-shape fit that also optimizes scale and rotation (Sec. 5.3).

We compare against ConeGS (Baranowski et al., 2026), EDGS (Kotovenko et al., 2026), and Taming 3DGS (Mallick et al., 2024) on Tanks & Temples, Deep Blending, and Mip-NeRF 360 at 100K, 500K, and 1M Gaussians. At 100K, ProbeGS raises Deep Blending PSNR by 0.50 dB and lowers Tanks & Temples LPIPS by 0.041 over the next-best method. Its fine-detail gain is largest on surfaces that few training views see, where multi-view optimization has little signal to correct a badly sized Gaussian (Sec. 5.2).

Our contributions are **(1)** a closed-form scale for each corrective 2D Gaussian, read from robust multi-scale residual decay on rings; **(2)** placement that refines only center and color, followed by a covariance lift that reproduces each fitted 2D Gaussian in its source view with an ordinary 3DGS primitive, leaving the renderer and model format unchanged; and **(3)** state-of-the-art quality at 100K Gaussians on three benchmarks.

2 Related Work

Densification and budget control. Density control decides where 3DGS adds primitives and how many it keeps. Standard 3DGS splits or clones high-gradient primitives with inherited properties (Kerbl et al., 2023). Later methods refine splitting with pixel-aware gradients, multi-view error, or geometric priors (Zhang et al., 2024b; Lyu et al., 2025; Yang et al., 2025; Cheng et al., 2024), or meet a budget by relocation, pruning, or minimal splitting (Mallick et al., 2024; Kheradmand et al., 2024; Fang & Wang, 2024; 2026; Wang et al., 2025; Hanson et al., 2025); EDGS (Kotovenko et al., 2026) replaces densification with correspondence-based initialization.

Error-guided densification. A second line adds primitives where the rendering error is high. Revising Densification (Rota Bulò et al., 2024) attributes pixel errors to primitives to guide splitting, and RTG-SLAM (Peng et al., 2024) inserts Gaussians at newly observed or high-error pixels in online mapping. ConeGS (Baranowski et al., 2026) takes scale from a cone at each high-error pixel and lifts through an Instant-NGP (Müller et al., 2022) backbone; CEM-4DGS (Kang et al., 2025) fits ellipses to clustered error pixels in dynamic scenes. AuGS (Yang et al., 2026) fits 2D Gaussians to residual images after training and lifts them with rendered depth. During training, ProbeGS reads each proposal's scale, and its shape where supported, from the residual in closed form and fits only center and color.

Image-space Gaussian fitting. 2D Gaussians fit images compactly, and some methods carry them into 3D. GaussianImage (Zhang et al., 2024a) and Image-GS (Zhang et al., 2025) use them only to represent single images. EllipssianNet (Kim et al., 2026) predicts 2D Gaussians from an image to initialize image fitting; converted to 3D Gaussians, they also initialize online SLAM. ProbeGS instead fits 2D Gaussians to the rendering residual and lifts them to 3D.

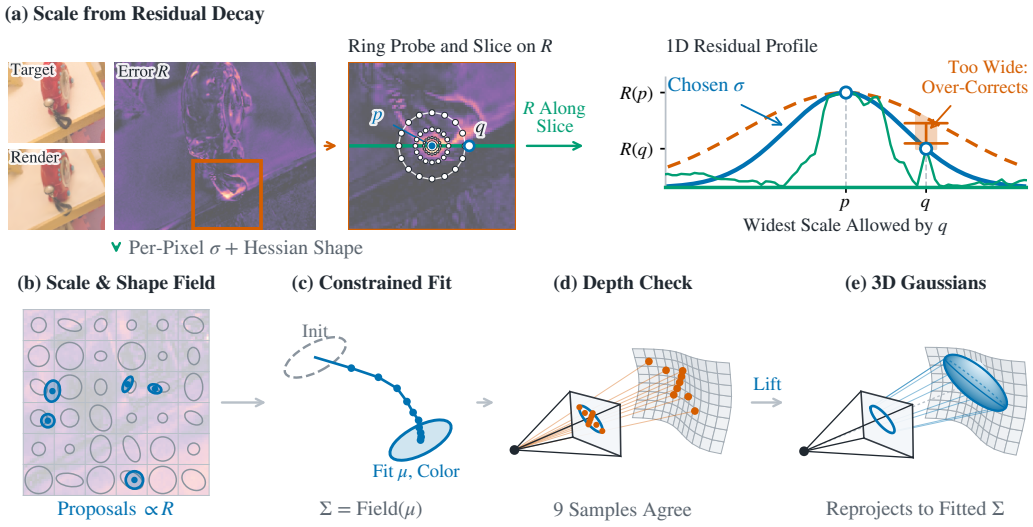

Figure 2: **ProbeGS pipeline.** (a) Residual R, shown on a held-out view of the final model (training computes it on training views), with the zoomed region boxed; the ring probe around p with a slice through a neighbor q; and R along that slice: a wider Gaussian (dashed) over-corrects the error at q; the widest one within $R(q)$ (blue) sets the scale in closed form. (b) A gated log-residual Hessian adds orientation and elongation, giving a covariance at every pixel; proposals sampled in proportion to R take it at their own pixels. (c) A $K{=}10$ fit optimizes only their centers and colors; the covariance is re-read at each moved center. (d) Depths at nine samples along each proposal's axes are checked for agreement, and (e) passing proposals become ordinary 3DGS primitives that reproduce the fitted 2D Gaussians in the source view.

3 Preliminaries

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) represents a scene as a collection of 3D Gaussians, each parameterized by a mean position $\boldsymbol{\mu} \in \mathbb{R}^3$, a covariance $\boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^\top\mathbf{R}^\top$ composed of a rotation $\mathbf{R}$ and a scale $\mathbf{S}$, opacity $\alpha \in [0,1]$, and spherical harmonic (SH) coefficients $\mathbf{c}$ for view-dependent color. Rendering projects each Gaussian onto the image plane to obtain a 2D covariance $\boldsymbol{\Sigma}' = \mathbf{J}\mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top\mathbf{J}^\top$, where $\mathbf{W}$ is the view transform and $\mathbf{J}$ the Jacobian of the projective approximation. The pixel color is computed by depth-ordered alpha blending over the sorted Gaussians:

$$\mathbf{C}(p) = \sum_{i=1}^{N} T_i\, \alpha_i\, \mathbf{c}_i, \quad T_i = \prod_{j=1}^{i-1}(1-\alpha_j), \tag{1}$$

where α_i combines the Gaussian's learned opacity with its projected evaluation at pixel p. Training minimizes a photometric loss, L1 plus D-SSIM (Wang et al., 2004), against ground-truth images, with all Gaussian parameters optimized by backpropagation through the differentiable tile-based rasterizer of 3DGS (Kerbl et al., 2023).

4 Method

4.1 Overview

ProbeGS replaces standard 3DGS densification. Figure 2 shows one *fitting call*: every $\delta = 10$ training iterations in $[0, T_{\text{dens}}]$, compute the residual $R(p) = \frac{1}{3}\|I^*(p) - \hat{I}(p)\|_1$ of a uniformly sampled training view, sample N centers, assign analytical scale and shape (Fig. 2a–b, Sec. 4.2), refine position and color (Fig. 2c, Sec. 4.3), and lift to 3D using depth (Fig. 2d–e, Sec. 4.4). The scale thus adapts to the local error extent; the fit serves placement, not 2D reconstruction. Lifted Gaussians are stashed and inserted in a batch every Δ_{dens} iterations, which we call a *drain*.

4.2 Analytical Initialization for 2D Gaussians

We read each proposal's scale from residual decay and optimize only center and color. Each proposal is a residual-weighted local feature, akin to classical interest-point detection (Harris & Stephens, 1988; Lindeberg, 1998): its Gaussian profile covers the local error but stays below the residual of lower-residual neighbors, and a secondary Hessian readout tests whether anisotropy is supported. Fig. 2a illustrates the construction.

Parameters of a 2D Gaussian. Each proposal carries three parameters: a *center* $\mu \in \Omega$, an *RGB color* $\mathbf{c} \in [0,1]^3$, and a *covariance* $\Sigma \in \mathbb{R}^{2\times 2}$. We parameterize Σ by three scalar fields,

$$\Sigma(\mu) \;=\; \mathbf{R}\big(\theta(\mu)\big)\,\mathrm{diag}\big(\sigma^2_{\text{major}}(\mu),\,\sigma^2_{\text{minor}}(\mu)\big)\,\mathbf{R}\big(\theta(\mu)\big)^\top, \tag{2}$$

where $\mathbf{R}(\theta)$ is the in-plane rotation matrix and $\sigma_{\text{major}} \geq \sigma_{\text{minor}}$ are the major- and minor-axis standard deviations in pixels. We call σ_{minor} the proposal's *scale* and the axis ratio and orientation $(\sigma_{\text{major}}/\sigma_{\text{minor}}, \theta)$ its *shape*. The three fields $(\sigma_{\text{minor}}, \sigma_{\text{major}}, \theta)$ are dense over the image grid Ω, recomputed from the current residual at each fitting call, and shared by all proposals in that call.

Centers are sampled with weight $\propto R(p)$ (Baranowski et al., 2026), concentrating on missed regions; colors start at the ground-truth pixels and are refined (Sec. 4.3). A single analytical pass supplies the robust radial minor scale (Eq. 6) and the scale-adaptive log-residual Hessian shape (Eq. 9).

Radial decay model. Consider an isotropic scalar correction centered at p with amplitude $R(p)$ and standard deviation σ. A locally compatible correction should decay toward a neighbor q whose residual is smaller. This motivates the surrogate relation

$$R(p)\,\exp\!\big(-\tfrac{\|q-p\|^2}{2\sigma^2}\big) \;\leq\; R(q), \tag{3}$$

which discourages a proposal whose tail is large relative to nearby residual. A wider Gaussian would add more correction at q than the residual there calls for, over-correcting pixels the render already matches, so the equality gives an upper bound on σ rather than a fitted width. Fig. 2a shows this bound in one direction of the ring probe.

Robust multi-scale radial field. At each native-grid radius $d \in \mathcal{D} = \{1, 2, 4, 8, 16\}$ pixels, we bicubically sample $M = 16$ points on the exact Euclidean ring. With $u_m = (\cos(2\pi m/M), \sin(2\pi m/M))$, the radial evidence is

$$e_d(p) = Q_{0.25}\Big(\{R(p + d u_m)\}_{m=0}^{M-1}\Big), \tag{4}$$

where $Q_{0.25}$ is the lower quartile. All samples lie at distance exactly d, so one noisy low-residual pixel cannot set the scale. Solving the equality corresponding to Eq. 3 gives the per-radius estimate

$$\widetilde{\sigma}_d(p) = \frac{d}{\sqrt{2\,[\ln(\max(R(p)/\max(e_d(p),\varepsilon),1))]_\varepsilon}}, \tag{5}$$

where $[x]_\varepsilon = \max(x, \varepsilon)$ and $\varepsilon = 10^{-4}$. We aggregate the five estimates robustly:

$$\sigma_{\text{minor}}(p) = \mathrm{clip}\big(Q_{0.25}(\{\widetilde{\sigma}_d(p) : d \in \mathcal{D}\}),\,\sigma_{\text{floor}},\,\sigma_{\text{rad}}\big). \tag{6}$$

The clip bounds are σ_{floor}=0.5 and σ_{rad}=16 px; the two lower-quartile operations provide spatial and cross-scale robustness. We use this radial estimate as the minor-axis initialization; the appendix gives the numerical safeguards.

Scale-adaptive Hessian for rotation and anisotropy. The same local-peak model gives the rotation through a Hessian analysis of the error map, as in affine shape adaptation for interest-point detection (Mikolajczyk & Schmid, 2004): if R is locally Gaussian around $\mu = p$, then $\ln R$ is quadratic in $q - p$ and its Hessian recovers the inverse covariance, $\nabla^2 \ln R \equiv -\Sigma^{-1}$. Eigenvectors of $-\nabla^2 \ln R$ give the ellipse axes, while the eigenvalue ratio gives the squared axis ratio. We use this ratio and orientation, not the absolute curvature magnitudes, because real residuals include a noise floor and are only locally Gaussian. We compute the Hessian on the noise-subtracted log-residual

$$L(p) \;=\; \ln\big(\max\big(R(p) - e_n,\,\varepsilon\big)\big), \quad e_n = \mathrm{median}_p\, R(p), \tag{7}$$

where e_n removes a per-image background level before taking the logarithm; the clamp at ε keeps L finite where $R(p) - e_n$ is non-positive. We compute finite-difference Hessians of L and average each component over a $(2d{+}1)^2$ window at the probe level d nearest to $\sigma_{\text{minor}}(p)$. The shape is thus measured at a support matched to the robust radial scale.

Let L_{xx}, L_{yy}, L_{xy} denote the smoothed second derivatives of L at p, and order the Hessian eigenvalues as $\lambda_1 \geq \lambda_2$. Only negative-definite local curvature is compatible with the Gaussian-peak model used for the anisotropic readout. Where both eigenvalues are negative, we set

$$\begin{aligned} r(p) &= \operatorname{clip}\left(\sqrt{\frac{-\lambda_2}{-\lambda_1}}, 1, r_{\max}\right), \\ \theta(p) &= \tfrac{1}{2}\arctan\frac{2L_{xy}}{L_{xx} - L_{yy}}, \end{aligned} \tag{8}$$

where $r(p)$ is the major-to-minor anisotropy ratio ($r_{\max}{=}5$) and $\theta(p) \in (-\frac{\pi}{2}, \frac{\pi}{2}]$ is the major-axis rotation, with the quadrant set by the signs of $2L_{xy}$ and $L_{xx}{-}L_{yy}$. At indefinite or positive-curvature pixels, we use the isotropic fallback $r(p) = 1$ and $\theta(p) = 0$. The major scale is therefore

$$\sigma_{\text{major}}(p) = \operatorname{clip}(\sigma_{\text{minor}}(p) r(p), \sigma_{\text{floor}}, \sigma_{\text{global}}), \tag{9}$$

with $\sigma_{\text{global}}{=}50$ px. The analysis produces dense fields $\sigma_{\text{minor}}(p)$, $\sigma_{\text{major}}(p)$, $\theta(p)$. During fitting, σ_{major} and σ_{minor} are bilinearly sampled at the proposal center and carry gradients back into position. Rotation follows the sampled field but is detached from the position gradient. The appendix gives pseudocode for the field construction.

4.3 2D Gaussian Fitting

2D optimization model. Each call creates a disposable model of N Gaussians, indexed $i = 1, \ldots, N$, to fit the current residual. Only normalized position $\boldsymbol{\mu}_i \in [0,1]^2$ and RGB color $\mathbf{c}_i \in [0,1]^3$ are learnable. Covariance $\boldsymbol{\Sigma}_i$ is sampled from the analytical σ_{minor}, σ_{major}, and θ fields (Sec. 4.2): scales carry position gradients; rotation tracks the field with its position gradient detached. Scale and rotation are never optimized: a moved center re-reads its covariance from the fields. The 2D model is never rendered at test time.

Rendering and compositing. The 2D layer corrects the current render rather than rebuilding the image, so a proposal must not become fully opaque by default. The Image-GS (Zhang et al., 2025) renderer normalizes Gaussian weights $w_i(p) = \exp(-\frac{1}{2}(p - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1}(p - \boldsymbol{\mu}_i))$ over the $k{=}10$ largest weights; we add a baseline reference term w_0, a kernel wider than the image with zero color and mask $m_0 = 0$ (proposals have $m_i = 1$). With this index set $\mathcal{T}_k(p) \subseteq \{0, \ldots, N\}$ and $Z(p) = \epsilon + \sum_{j \in \mathcal{T}_k(p)} w_j(p)$,

$$I_{2D}(p) = \frac{\sum_{i \in \mathcal{T}_k(p)} w_i(p)\, \mathbf{c}_i}{Z(p)}, \quad \alpha_{2D}(p) = \frac{\sum_{i \in \mathcal{T}_k(p)} w_i(p) m_i}{Z(p)} \in [0,1]. \tag{10}$$

The reference enters only Z, so coverage stays partial where no proposal dominates. Compositing with the detached current render, $\hat{I}_{\text{comp}} = I_{2D} + \hat{I}_{\text{3DGS}}(1 - \alpha_{2D})$, lets the layer replace wrong colors as well as fill gaps. The appendix gives the expanded formulation.

2D reconstruction loss. The 2D model is optimized on

$$\mathcal{L} = \mathcal{L}_1(\hat{I}_{\text{comp}}, I^*) + \lambda_{\text{SSIM}}\big(1 - \text{SSIM}(\hat{I}_{\text{comp}}, I^*)\big) + \lambda_{\text{grad}}\, \mathcal{L}_1\big(|\nabla \hat{I}_{\text{comp}}|, |\nabla I^*|\big) + \lambda_{\text{cov}}\, \mathcal{L}_{\text{cov}}, \tag{11}$$

where $\mathcal{L}_1$ and SSIM are the standard 3DGS photometric terms (Kerbl et al., 2023) on the composite output, $|\nabla \cdot|$ is the finite-difference gradient magnitude of the composite output, $\mathcal{L}_{\text{cov}}$ penalizes 2D-layer opacity α_{2D} in low-residual regions to discourage spurious placements,

$$\mathcal{L}_{\text{cov}} = \text{mean}_p[(1 - \hat{e}(p))\, \alpha_{2D}(p)], \tag{12}$$

where $\hat{e} = R / \max_p R$ is the residual normalized to $[0,1]$. We set $\lambda_{\text{SSIM}}{=}0.1$, $\lambda_{\text{grad}}{=}1$, $\lambda_{\text{cov}}{=}0.01$. With scale and rotation supplied analytically, we use $K{=}10$ reconstruction iterations of Adam; the fields are computed once per call in a field-initialization pass not counted in K. Calls run every $\delta{=}10$ training iterations across the densification window $[0, T_{\text{dens}}]$, with $T_{\text{dens}}{=}25{,}000$.

4.4 LIFTING TO 3D

Following ConeGS, we take depth from an Instant-NGP (Müller et al., 2022; Li et al., 2023) backbone trained on the training views only and lift each fitted 2D Gaussian along the ray through its subpixel center. The 3D covariance is set through the rasterizer's perspective Jacobian so that the new primitive renders with the fitted 2D covariance in its source view; opacity starts at 0.1 and color at the fitted value, and proposals without a valid depth are rejected. A depth check adapted from AuGS (Yang et al., 2026), at nine samples along each Gaussian's axes, corrects or rejects centers whose depth would create an occluder at a depth discontinuity. The appendix details the lift.

At each drain (every $\Delta_{\text{dens}}{=}100$ iterations), we prune Gaussians with opacity below 0.005 and insert surviving proposals up to the remaining headroom $C - n_t$ (budget C, current count n_t), sampling excess stash entries by fitted-center residual and discarding the rest. Between drains, standard 3DGS optimization continues.

Causal proposal scheduling (controller). Each call proposes a share of the expected pruning replacement and the remaining growth, using only completed drains and remaining capacity. With current count n_t, remaining drains D_t, pruning EMA $\bar{p}_t$, oversupply factor γ, and m calls per drain,

$$N_t = \min\Big(N_{\max},\, \max\big(N_{\min},\, \lceil \tfrac{\gamma}{m}\big(\bar{p}_t + \tfrac{C-n_t}{D_t}\big)\rceil\big)\Big). \tag{13}$$

At 100K the floor equals the cap, so the proposal count is fixed; the appendix gives the constants and the fallback used at larger budgets.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Datasets and metrics. We evaluate on Tanks & Temples (Knapitsch et al., 2017), Deep Blending (Hedman et al., 2018), and the Mip-NeRF 360 (Barron et al., 2022) subset shared by ConeGS (Baranowski et al., 2026), 3DGS-MCMC (Kheradmand et al., 2024), and Scaffold-GS (Lu et al., 2024). Every eighth COLMAP camera is held out for testing, and we report PSNR (dB), SSIM, and LPIPS (Zhang et al., 2018) (VGG) on them. The appendix lists scenes and resolutions.

Baselines. Table 1 compares five baselines at matched requested budgets: 3DGS (Kerbl et al., 2023), SfM-initialized 3DGS-MCMC (Kheradmand et al., 2024), EDGS (Kotovenko et al., 2026) with densification enabled, our count-calibrated Taming 3DGS adaptation (Mallick et al., 2024), and ConeGS (Baranowski et al., 2026). ConeGS shares our depth backbone and densification loop but sizes each new Gaussian from a camera cone, so it is our direct baseline: the same densification without residual-based scale and shape initialization. For fitting cost, we time our adaptations of the fitters in gOMP (Wang et al., 2012), CEM-4DGS (Kang et al., 2025), and AuGS (Yang et al., 2026); their iterative fits are too slow to run as part of regular densification, so we compare only this stage (Table 3). The appendix details the baseline settings and additional official native-count comparisons.

Baseline numbers marked ^{S} are the published values from the ConeGS supplement (Baranowski et al., 2026) (Tables A3, A8–A11). We use the causal proposal schedule (Eq. 13) with one field-initialization pass followed by $K{=}10$ reconstruction steps. The same settings apply to every scene and budget, and ProbeGS ends at exactly the requested count.

5.2 COMPARISON TO STATE OF THE ART

ProbeGS is best on every Deep Blending metric at every budget. On Tanks & Temples it leads all metrics at 100K and SSIM and LPIPS at 500K and 1M, where Taming 3DGS has higher PSNR. On Mip-NeRF 360 it has the highest PSNR and SSIM (SSIM tied at 100K), and LPIPS ties ConeGS at displayed precision at 100K and 500K (ConeGS leads at full precision). The PSNR advantage over ConeGS is positive on every dataset and budget and narrows at 1M on Mip-NeRF 360. Fig. 3 shows 100K renders; official checkpoints do not exist at these budgets, so qualitative baselines are trained with their official code at the same budget. Fig. 4 shows how the residual evolves over training.

Table 1: Dataset-average comparison across Gaussian budgets (100K, 500K, 1M); every ranked row uses the requested count. Unless marked ^{R}, baseline numbers are published values from the ConeGS supplement (Baranowski et al., 2026). ^{S}: published in the ConeGS supplement; ^{R}: our count-calibrated Taming adaptation. †: ConeGS adaptation with densification enabled. Grey (Reference) rows are unranked operating points at each method's native Gaussian count (row below each), placed after the largest budget that count exceeds; Deep Blending and Tanks & Temples values are published, and Mip-NeRF 360 values average our seven scenes (Mini-Splatting2: our run of the official code). **Bold**: best; underlined: second; ties at displayed precision are ranked at full precision.

	Deep Blending			Tanks & Temples			Mip-NeRF 360		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Budget: 100*K*									
3DGS^{S} (Kerbl et al., 2023)	24.65	0.827	0.412	22.38	0.774	0.334	23.61	0.692	0.413
3DGS-MCMC (SfM)^{S} (Kheradmand et al., 2024)	28.94	0.876	0.333	22.49	0.780	0.332	27.06	0.800	0.303
EDGS^{S,†} (Kotovenko et al., 2026)	28.43	0.872	0.337	22.32	0.778	0.324	27.09	0.798	0.296
Taming 3DGS^{R} (Mallick et al., 2024)	29.14	0.890	0.307	22.60	0.788	0.282	26.65	0.775	0.328
ConeGS^{S} (Baranowski et al., 2026)	29.44	0.880	0.328	23.12	0.790	0.309	27.74	0.809	**0.285**
ProbeGS (ours)	**29.94**	**0.902**	**0.278**	**23.39**	**0.815**	**0.241**	**27.86**	**0.809**	0.285
Taming Compact (Mallick et al., 2024) (Reference) Gaussian count	29.89	0.902 0.27M	0.263	23.89	0.835 0.29M	0.207			
Mini-Splatting2 (Fang & Wang, 2026) (Reference) Gaussian count				23.14	0.841 0.35M	0.186			
Budget: 500*K*									
3DGS^{S} (Kerbl et al., 2023)	29.26	0.882	0.302	23.54	0.816	0.265	28.22	0.821	0.260
3DGS-MCMC (SfM)^{S} (Kheradmand et al., 2024)	29.44	0.887	0.308	23.68	0.825	0.259	28.82	0.861	0.214
EDGS^{S,†} (Kotovenko et al., 2026)	29.33	0.889	0.286	23.46	0.831	0.229	28.82	0.865	0.193
Taming 3DGS^{R} (Mallick et al., 2024)	29.85	0.904	0.259	**24.23**	0.847	0.195	28.91	0.846	0.232
ConeGS^{S} (Baranowski et al., 2026)	29.86	0.891	0.285	23.69	0.829	0.228	29.08	0.870	**0.190**
ProbeGS (ours)	**30.03**	**0.908**	**0.241**	23.94	**0.853**	**0.164**	**29.20**	**0.871**	0.190
Taming Compact (Mallick et al., 2024) (Reference) Gaussian count							28.69	0.854 0.64M	0.221
Mini-Splatting2 (Fang & Wang, 2026) (Reference) Gaussian count	30.08	0.912 0.65M	0.240				28.91	0.875 0.62M	0.183
Budget: 1*M*									
3DGS^{S} (Kerbl et al., 2023)	28.94	0.881	0.291	23.59	0.826	0.244	28.71	0.846	0.222
3DGS-MCMC (SfM)^{S} (Kheradmand et al., 2024)	29.67	0.887	0.287	23.93	0.837	0.233	29.23	0.875	0.189
EDGS^{S,†} (Kotovenko et al., 2026)	29.43	0.889	0.271	23.66	0.843	0.201	29.18	0.878	0.168
Taming 3DGS^{R} (Mallick et al., 2024)	29.95	0.908	0.248	**24.28**	0.855	0.176	29.25	0.862	0.206
ConeGS^{S} (Baranowski et al., 2026)	29.69	0.889	0.273	23.70	0.835	0.207	29.37	0.880	0.168
ProbeGS (ours)	**30.29**	**0.910**	**0.230**	24.00	**0.859**	**0.147**	**29.45**	**0.882**	**0.167**

The advantage concentrates where few training cameras observe the surface (Fig. 5). We divide each test image into square patches. A patch's coverage is the number of training cameras that see its surface point: we take the point at the patch center from the Instant-NGP depth shared with ConeGS and count the training cameras whose image contains its projection and whose own depth places no other surface in front of it. A patch's detail error is the mean absolute difference between the high-frequency components (the image minus a Gaussian-blurred copy) of the render and of the ground truth, and the gain over a baseline is the baseline's detail error minus that of ProbeGS. On average over the test views of all scenes, the gain over ConeGS is largest in patches seen by at most two training views, falls by more than half at 6–10 views, and disappears beyond ten; the gains over 3DGS-MCMC and 3DGS follow the same trend. In low-coverage patches, multi-view optimization has little signal to correct a poorly sized splat, so the scale set at insertion matters more. The supplementary videos show the same pattern: sparsely observed regions are sharper with ProbeGS than with the baselines.

5.3 Ablation Studies

Table 2 reports PSNR at 100K and 1M.

How scale and shape are set. *Single-scale* uses only the probe level $d{=}4$, removing the multi-probe radial estimate (Eq. 6) and the scale-adaptive Hessian window, and changes the average PSNR by -0.14 dB at 100K. *Isotropic* keeps the multi-probe scale but zeros the Hessian readout ($\sigma_{\text{major}}{=}\sigma_{\text{minor}}$, $\theta{=}0$), changing PSNR by -0.01 dB at 100K and -0.11 dB at 1M. At 100K the budget mostly covers coarse structure, where a correctly sized isotropic Gaussian suffices; at 1M the

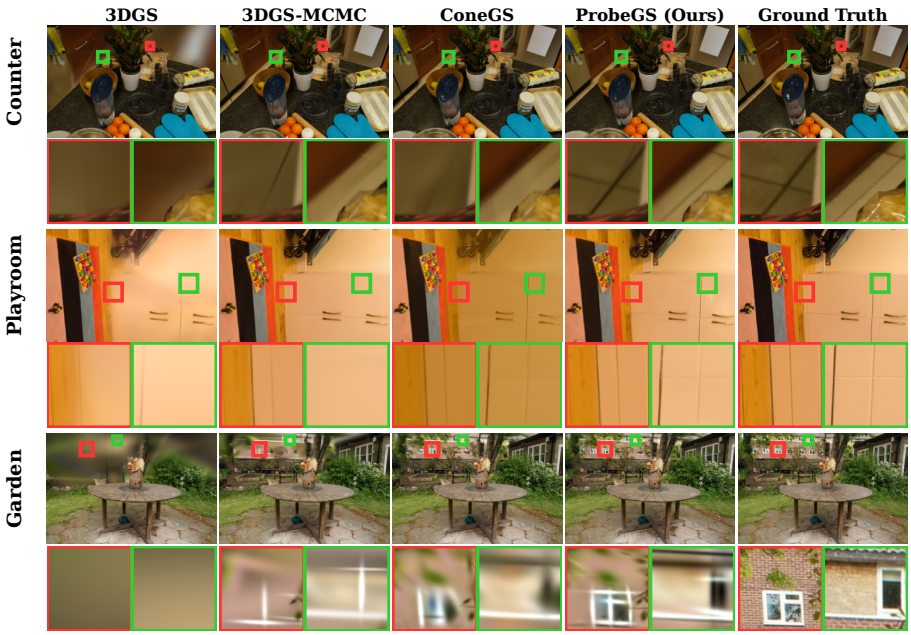

Figure 3: Qualitative comparison at 100K Gaussians on Counter (Mip-NeRF 360), Playroom (Deep Blending), and Garden (Mip-NeRF 360), against 3DGS (Kerbl et al., 2023), 3DGS-MCMC (Kheradmand et al., 2024), and ConeGS (Baranowski et al., 2026); the appendix shows 500K and 1M.

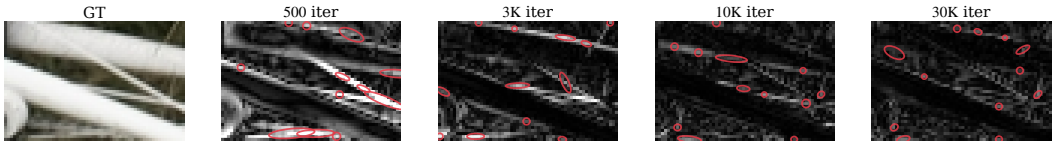

Figure 4: **Residual-shape evolution at the 100K budget.** Bicycle crop on one intensity scale; red ellipses are one-σ contours of the analytical covariance at local residual peaks.

remaining residual concentrates on thin, oriented structure, where elongation and orientation pay off. The *free-shape* path replaces the field's scale and shape with the fixed isotropic default initialization of Image-GS (Zhang et al., 2025) and jointly optimizes position, color, scale, and rotation. It changes PSNR by -0.09 dB at $K{=}10$ and -0.29 dB at $K{=}100$. At matched $K{=}100$ and 100K, the constrained path wins on 11/11 scenes in PSNR, 11/11 in SSIM, and 11/11 in LPIPS. It also lowers the per-call fitting error faster (Fig. 6).

Refinement after field initialization. Quality depends little on fit length: ten times the steps ($K{=}100$) changes PSNR by only $+0.09$ dB but raises training time from 20.6 to 66.5 min on average (ConeGS: 17.6 min; Table S1). A fitting call costs about 0.13 s even at a 20K proposal cap, 41–632$\times$ cheaper than our adaptations of greedy-pursuit, clustered, and AuGS fitters (Table 3).

6 Conclusion

We presented ProbeGS, which densifies 3DGS under a fixed Gaussian budget by reading each new Gaussian's scale from the rendering residual. The pixel-averaged loss can leave small, high-residual regions uncovered, and prior densification inherits scale from parent Gaussians, takes it from a camera cone, or fits it iteratively. ProbeGS samples pixels in proportion to the residual and sets each corrective 2D Gaussian's standard deviation in closed form from how fast the residual decays on rings at several radii, 41–632$\times$ cheaper per call than our adaptations of fitting-based alternatives. A short fit refines only centers and colors, and a covariance lift makes each fitted 2D Gaussian

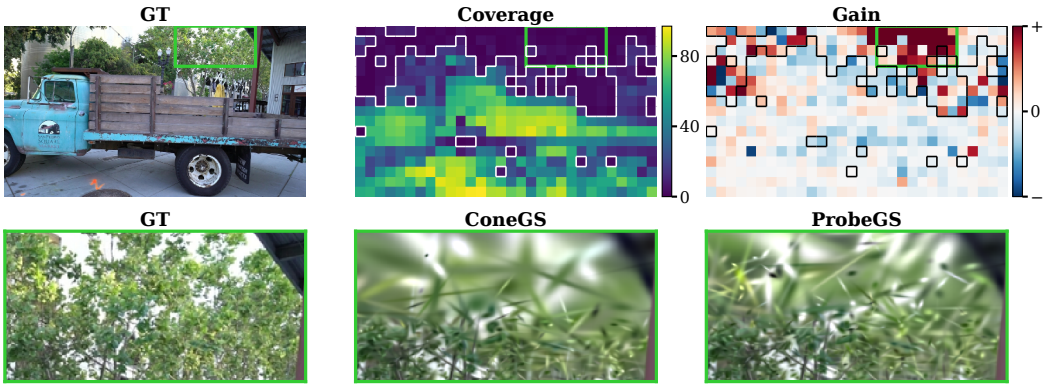

Figure 5: Per-patch training-view coverage and detail-error gain on one Truck test view at 100K, ProbeGS versus ConeGS (red: ProbeGS has the lower detail error; blue: ConeGS does). Outlines mark low-coverage patches; the bottom row crops the green box, the one with the largest gain.

Table 2: Ablation study on DB, T&T, Mip-NeRF 360 (average of dataset means) at 100K and 1M budgets; Δ: change from the full method.

	100K		1M	
Configuration	PSNR	Δ	PSNR	Δ
Analytical Shape (§4.2)				
Single-Scale	26.92	−0.14	27.68	−0.23
Isotropic	27.05	−0.01	27.80	−0.11
Refinement (§4.3)				
K=100 Constrained	27.15	+0.09	27.92	+0.01
K=10 Free-Shape	26.97	−0.09	27.76	−0.15
K=100 Free-Shape	26.77	−0.29	27.61	−0.30
K=10 **Constrained (Ours)**	27.06	—	27.91	—

Table 3: Per-call 2D fitting time (s) at each proposal cap; baselines are our adaptations of gOMP (Wang et al., 2012), CEM-4DGS (Kang et al., 2025), and AuGS (Yang et al., 2026). Component time; at 2,500 calls per run the fastest adaptation adds ∼3.6 h; their own settings call them rarely.

Fitter (cap)	2K	10K	20K
ProbeGS (ours)	0.10	0.12	0.13
gOMP-style SOMP	10.08	43.85	82.12
CEM-4DGS-style	13.93	13.86	13.70
AuGS adaptation	5.12	5.26	5.33

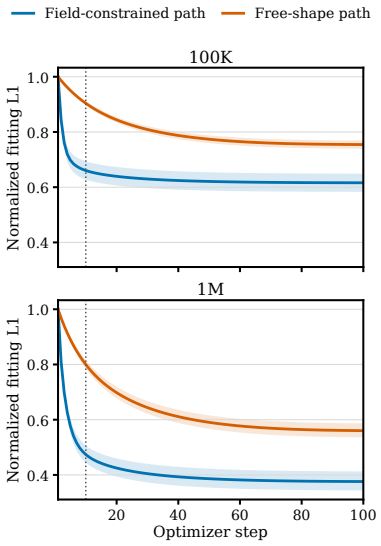

Figure 6: Per-call fitting error (L1, each path normalized by its own initial value, so the curves compare convergence rate rather than absolute error) over K=100 steps for the field-constrained and free-shape paths, averaged equally over datasets; bands are 95% scene-bootstrap intervals and the dotted line marks step 10.

one ordinary 3DGS primitive, leaving the renderer and model format unchanged. On three benchmarks, ProbeGS achieves state-of-the-art quality at 100K Gaussians, raising Deep Blending PSNR by 0.50 dB and lowering Tanks & Temples LPIPS by 0.041 over the next-best method, and its fine-detail gain concentrates on surfaces that few training views see. The appendix discusses limitations; future work includes dynamic scenes and a depth source without separate pretraining.

ETHICS STATEMENT

We adhere to the ICLR Code of Ethics. This work involves no human subjects and no new data collection: all experiments use public benchmarks under their licenses, and we process no personally identifiable information.

REPRODUCIBILITY STATEMENT

Section 4 specifies the method: the adaptive scale and shape readout (Sec. 4.2), the 2D fit (Sec. 4.3), and the lift and proposal controller (Sec. 4.4); Sec. 5.1 gives the evaluation protocol. The appendix provides implementation details and hyperparameters (App. A.1), pseudo-code for the analytical field (App. A.2), the lift and its depth-consistency check (App. A.4–A.5), the datasets, metrics, and baseline settings (App. B), and the fitting-time measurement setup (App. D.1). All experiments use the public Mip-NeRF 360, Tanks & Temples, and Deep Blending benchmarks, holding out every eighth view for testing.

AI USE STATEMENT

We used generative AI tools to aid or polish writing, for retrieval and discovery, for research ideation or execution, and to support qualitative and thematic data analysis; we did not use them for translation or synthetic data, and mathematical claims, proofs, and dataset cleaning are not applicable. We take responsibility for all content.

REFERENCES

Bartłomiej Baranowski, Stefano Esposito, Patricia Gschoßmann, Anpei Chen, and Andreas Geiger. ConeGS: Error-guided densification using pixel cones for improved reconstruction with fewer primitives. In *International Conference on 3D Vision (3DV)*, pp. 989–999, 2026. doi: 10.1109/3DV69130.2026.00099.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5460–5469, 2022. doi: 10.1109/CVPR52688.2022.00539. URL https://arxiv.org/abs/2111.12077.

Kai Cheng, Xiaoxiao Long, Kaizhi Yang, Yao Yao, Wei Yin, Yuexin Ma, Wenping Wang, and Xuejin Chen. GaussianPro: 3D Gaussian Splatting with Progressive Propagation. In *International Conference on Machine Learning (ICML)*, pp. 8123–8140, 2024. URL https://proceedings.mlr.press/v235/cheng24f.html.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of Gaussians. In *European Conference on Computer Vision (ECCV)*, pp. 165–181, 2024. doi: 10.1007/978-3-031-72980-5_10.

Guangchi Fang and Bing Wang. Efficient scene modeling via structure-aware and region-prioritized 3D gaussians. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(4):4623–4641, 2026. doi: 10.1109/TPAMI.2025.3646473. URL https://doi.org/10.1109/TPAMI.2025.3646473.

Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-Splat: Fast 3D Gaussian Splatting with Sparse Pixels and Sparse Primitives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21537–21546, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Hanson_Speedy-Splat_Fast_3D_Gaussian_Splatting_with_Sparse_Pixels_and_Sparse_CVPR_2025_paper.html.

Chris Harris and Mike Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference*, pp. 23.1–23.6, 1988. doi: 10.5244/C.2.23. URL https://doi.org/10.5244/C.2.23.

Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics*, 37(6), 2018. doi: 10.1145/3272127.3275084.

Taeho Kang, Jaeyeon Park, Kyungjin Lee, and Youngki Lee. Clustered error correction with grouped 4D Gaussian splatting. In *ACM SIGGRAPH Asia 2025 Conference Papers*, 2025. doi: 10.1145/3757377.3763858.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4), July 2023. doi: 10.1145/3592433. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Yang-Che Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3D Gaussian Splatting as Markov Chain Monte Carlo. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 80965–80986, 2024. doi: 10.52202/079017-2573. URL http://papers.nips.cc/paper_files/paper/2024/hash/93be245fce00a9bb2333c17ceae4b732-Abstract-Conference.html.

MyoungGon Kim, JeongHyeon Ahn, Seohyeon Park, Hyemi Kim, Seunghyun Park, Jung Ho Hwang, and JungHyun Han. EllipssianNet: Image-guided sampling of 2D gaussians for gaussian splatting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 2104–2113, 2026. doi: 10.1109/WACV61042.2026.00209. URL https://openaccess.thecvf.com/content/WACV2026/html/Kim_EllipssianNet_Image-guided_Sampling_of_2D_Gaussians_for_Gaussian_Splatting_WACV_2026_paper.html.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. doi: 10.1145/3072959.3073599.

Dmytro Kotovenko, Olga Grebenkova, and Björn Ommer. EDGS: Eliminating densification for efficient convergence of 3DGS. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 41065–41076, 2026. URL https://arxiv.org/abs/2504.13204.

Ruilong Li, Hang Gao, Matthew Tancik, and Angjoo Kanazawa. NerfAcc: Efficient sampling accelerates NeRFs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 18537–18546, 2023. doi: 10.1109/ICCV51070.2023.01699.

Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):79–116, 1998. doi: 10.1023/A:1008045108935.

Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-GS: Structured 3D Gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20654–20664, 2024. doi: 10.1109/CVPR52733.2024.01952.

Yanzhe Lyu, Kai Cheng, Xin Kang, and Xuejin Chen. ResGS: Residual densification of 3D Gaussian for efficient detail recovery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28093–28102, 2025.

Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Francisco Vicente Carrasco, Markus Steinberger, and Fernando De La Torre. Taming 3DGS: High-quality radiance fields with limited resources. In *ACM SIGGRAPH Asia 2024 Conference Papers*, 2024. doi: 10.1145/3680528.3687694. URL https://humansensinglab.github.io/taming-3dgs/docs/paper_lite.pdf.

Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004. doi: 10.1023/B:VISI.0000027790.02288.f2. URL https://doi.org/10.1023/B:VISI.0000027790.02288.f2.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4), 2022. doi: 10.1145/3528223.3530127. URL http://dx.doi.org/10.1145/3528223.3530127.

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 2024 Conference Papers*, 2024. doi: 10.1145/3641519.3657455.

Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in Gaussian splatting. In *European Conference on Computer Vision (ECCV)*, pp. 347–362, 2024. doi: 10.1007/978-3-031-73036-8_20.

Jian Wang, Seokbeop Kwon, and Byonghyo Shim. Generalized orthogonal matching pursuit. *IEEE Transactions on Signal Processing*, 60(12):6202–6216, 2012. doi: 10.1109/TSP.2012.2218810.

Peihao Wang, Yuehao Wang, Dilin Wang, Sreyas Mohan, Zhiwen Fan, Lemeng Wu, Ruisi Cai, Yu-Ying Yeh, Zhangyang Wang, Qiang Liu, and Rakesh Ranjan. Steepest Descent Density Control for Compact 3D Gaussian Splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 26663–26672, 2025. URL https://arxiv.org/abs/2505.05587.

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. doi: 10.1109/TIP.2003.819861. URL https://doi.org/10.1109/TIP.2003.819861.

Haosen Yang, Chenhao Zhang, Wenqing Wang, Marco Volino, Adrian Hilton, Li Zhang, and Xiatian Zhu. Improving Gaussian Splatting with Localized Points Management. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21696–21705, 2025. doi: 10.1109/CVPR52734.2025.02021.

Yixin Yang, Bojian Wu, Yang Zhou, and Hui Huang. Augmented Radiance Field: A General Framework for Enhanced Gaussian Splatting. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=IzlaRUHncO.

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)*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068. URL https://arxiv.org/abs/1801.03924.

Xinjie Zhang, Xingtong Ge, Tongda Xu, Dailan He, Yan Wang, Hongwei Qin, Guo Lu, Jing Geng, and Jun Zhang. GaussianImage: 1000 fps image representation and compression by 2D Gaussian Splatting. In *European Conference on Computer Vision (ECCV)*, pp. 327–345, 2024a. doi: 10.1007/978-3-031-72673-6_18.

Yunxiang Zhang, Bingxuan Li, Alexandr Kuznetsov, Akshay Jindal, Stavros Diolatzis, Kenneth Chen, Anton Sochenov, Anton Kaplanyan, and Qi Sun. Image-GS: Content-adaptive image representation via 2D Gaussians. In *ACM SIGGRAPH 2025 Conference Papers*, 2025. doi: 10.1145/3721238.3730596. URL https://arxiv.org/abs/2407.01866.

Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-GS: Density control with pixel-aware gradient for 3D Gaussian splatting. In *European Conference on Computer Vision (ECCV)*, pp. 326–342, 2024b. doi: 10.1007/978-3-031-72655-2_19.