DELTASPLAT: ITERATIVE GAUSSIAN REFINEMENT FOR POSE-FREE FEED-FORWARD 3D GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

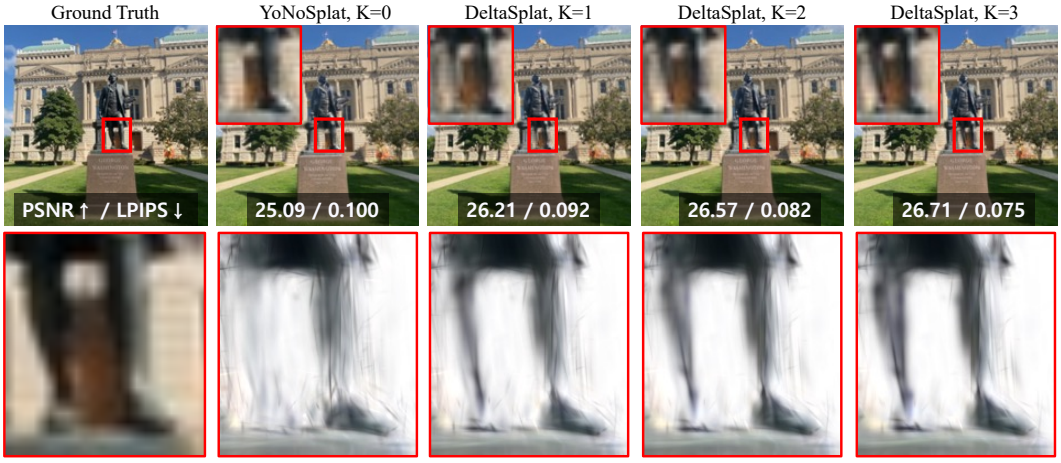


Figure 1: Iterative Gaussian refinement on a DL3DV scene in the pose-free setting. K denotes the number of refinement iterations applied. Starting from the feed-forward output of the YoNoSplat backbone ($K{=}0$), each DeltaSplat iteration progressively recovers structures missing from the initial prediction: the statue's legs, absent at $K{=}0$, gradually emerge in both the rendered views (top, red box) and the underlying Gaussian geometry (bottom, with the ground-truth image crop as reference), with monotonic gains in PSNR and LPIPS.

ABSTRACT

Pose-free feed-forward 3D Gaussian Splatting (3DGS) reconstructs a scene from sparse, unposed images in a single network pass, removing the need for camera calibration and per-scene optimization. However, camera estimation errors propagate into the predicted Gaussians and compound the geometric and photometric inaccuracies of single-pass prediction. To correct these errors, we introduce DeltaSplat, a lightweight Gaussian refinement module for pose-free feed-forward 3DGS. It iteratively renders the current Gaussians at the input context views and predicts per-Gaussian updates from the resulting residuals. A 2D residual alone, however, underdetermines the 3D correction. DeltaSplat therefore conditions each update on per-pixel Plücker rays and rendered depth as a soft geometric prior. A dual-branch convolutional mixer efficiently encodes these inputs, and per-attribute heads decode the fused features into position, opacity, and color updates. The module adds only $\sim$2.2% parameters to the backbone and remains fully feed-forward at inference. On DL3DV, DeltaSplat reaches 26.64 dB PSNR in the pose-free setting, improving its state-of-the-art backbone by 1.75 dB and surpassing even baselines supplied with ground-truth cameras; consistent gains hold across 6–24 views and all camera regimes.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) has rapidly emerged as a powerful representation for novel-view synthesis, combining an explicit set of anisotropic Gaussian primitives with differ-

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.

entiable rasterization to achieve high-quality, real-time rendering. Despite its rendering efficiency, conventional 3DGS requires accurate camera calibration and thousands of per-scene optimization steps to reconstruct every new scene. Feed-forward 3DGS (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025b; Ye et al., 2025; 2026) addresses this reconstruction bottleneck by amortizing scene-specific optimization into a learned model that directly predicts 3D Gaussians from a sparse set of input images in a single network pass. This paradigm substantially reduces reconstruction time while leveraging learned geometric and appearance priors to handle sparse observations.

Despite removing per-scene optimization, most early feed-forward 3DGS models (Chen et al., 2024; Xu et al., 2025a; Kang et al., 2026) still assume that accurate camera parameters are available. However, the dependence on accurate camera poses substantially limits the practical applicability of feed-forward reconstruction. Although calibrated rigs and controlled capture systems can provide reliable camera parameters, casually captured image collections generally contain neither known extrinsics nor precise intrinsics. Camera poses can be recovered using external sensors or SfM and SLAM pipelines, but these procedures introduce additional preprocessing cost and can be unreliable for sparse, weakly overlapping, or unstructured observations. Moreover, even small pose errors perturb the multi-view correspondences and projection geometry used to construct the Gaussians, leading to noticeable degradation in reconstruction quality. When pose estimation becomes the dominant computational cost or failure point, it also undermines the primary efficiency advantage of feed-forward 3DGS.

To address the aforementioned issue, recent pose-free feed-forward approaches jointly recover a scene representation and its camera geometry from unposed images. AnySplat (Jiang et al., 2025) employs a geometry transformer to predict 3D Gaussian primitives together with the camera intrinsics and extrinsics of all input views in a single forward pass. NoPoSplat (Ye et al., 2025) anchors one input camera coordinate system as a canonical frame and directly predicts the Gaussians of all views in this shared space, avoiding explicit transformations based on externally estimated poses. YoNoSplat (Ye et al., 2026) predicts local Gaussians and camera parameters for each view and aggregates them into a global representation using either predicted or provided poses, while its mixed-pose training strategy enables a single model to support posed, unposed, calibrated, and uncalibrated inputs.

Pose-free reconstruction, however, does not eliminate camera uncertainty; it internalizes camera estimation within the reconstruction model. Errors in the predicted cameras perturb the rays and coordinate transformations used to place the Gaussians, producing the systematic geometric displacement, ghosting, and blurred appearance seen in the unrefined prediction of Figure 1. These errors interact with the geometric and photometric inaccuracies of single-pass Gaussian prediction, which provides no explicit feedback mechanism for assessing and revising its output. Refining the predicted Gaussian representation is therefore necessary, but deriving a geometrically reliable correction from unposed observations remains challenging.

Rendering residuals provide a potentially useful correction signal, as they directly expose discrepancies between the current reconstruction and the observed context images. ReSplat (Xu et al., 2025a), for example, exploits pixel- and feature-space rendering errors to recurrently refine Gaussians under calibrated cameras. In the pose-free setting, however, uncertainty in the predicted camera geometry corrupts the correspondence between image-space residuals and individual Gaussians, thereby destabilizing the inferred 3D updates. Reliable Gaussian refinement under such camera uncertainty therefore requires additional geometric cues to disambiguate the residual-to-Gaussian mapping.

To address this challenge, we introduce DeltaSplat, a lightweight residual refinement module for pose-free feed-forward 3DGS. DeltaSplat iteratively compares the current renderings with the input context images and predicts Gaussian updates from the resulting 2D residuals. To lift these image-space errors into reliable 3D corrections, it jointly incorporates the Plücker rays (Plücker, 1865) and rendered depth associated with each Gaussian, providing a soft geometric prior for refining its position and appearance. Furthermore, a dual-branch convolutional mixer encodes those fine-grained full-resolution features while processing most features at reduced resolution, enabling more precise refinement with lower computational complexity. The encoded features are processed by parameter-specific heads to predict residual updates, which are then mapped back to the corresponding Gaussians to refine their position, opacity, and color. Repeating this process for three shared-weight iterations progressively improves the Gaussian representation. DeltaSplat adds only $\sim$2.2% to the backbone parameters and requires no per-scene optimization at inference time.

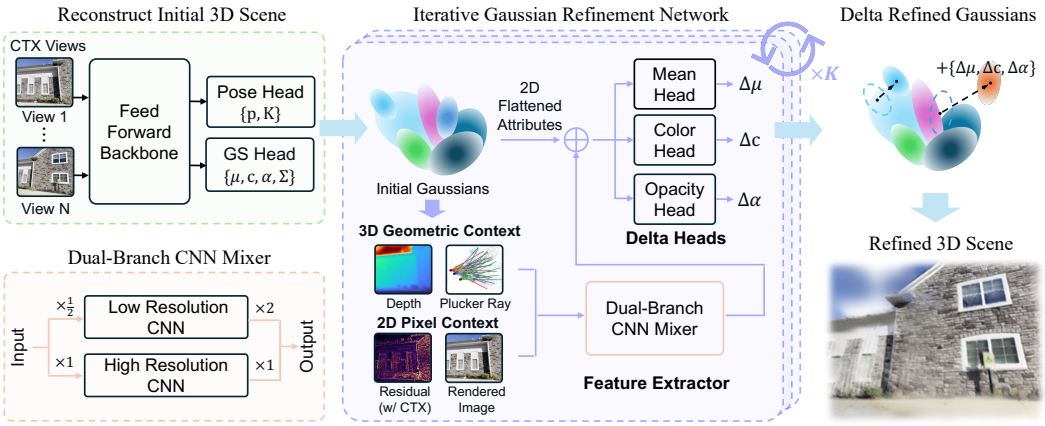


Figure 2: **Method overview.** The backbone predicts cameras and initial Gaussians; at each iteration, the refinement module encodes the render residual with the Plücker rays and the rendered depth, and decodes the per-Gaussian update $\{\Delta\mu, \Delta c, \Delta\alpha\}$, repeated $K{=}3$ times with shared weights.

Experiments on DL3DV (Ling et al., 2024) demonstrate that DeltaSplat substantially outperforms YoNoSplat (Ye et al., 2026), a state-of-the-art pose-free feed-forward 3DGS method. With six context views, predicted poses, and known intrinsics, DeltaSplat achieves 26.64 dB PSNR, improving YoNoSplat by **+1.75** dB under the same evaluation protocol, and surpassing every baseline evaluated with ground-truth cameras. The margins persist at 12 and 24 views and in the fully uncalibrated setting, and DeltaSplat likewise improves the backbone by about 2 dB when ground-truth cameras are supplied. These results demonstrate the effectiveness of geometrically guided residual refinement for high-quality pose-free reconstruction without per-scene optimization.

2 Related Work

2.1 Feed-Forward 3DGS

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) models 3D scenes as explicit, anisotropic Gaussians optimized on hundreds of calibrated images, whose camera poses are estimated via Structure-from-Motion (Schönberger & Frahm, 2016) (SfM). Although it yields high-fidelity renderings, obtaining them requires accurate camera poses and thousands of per-scene optimization steps. Subsequent efforts improve both training and rendering efficiency (Mallick et al., 2024; Hanson et al., 2025; Fang & Wang, 2024; Lee et al., 2025; Chen et al., 2025a; Ren et al., 2026; Dong et al., 2026), yet optimizing each scene from scratch remains inherently time-consuming. To overcome this bottleneck, feed-forward (FF) 3DGS amortizes per-scene optimization into a single forward pass, predicting Gaussians directly from sparse context images (typically 2–12 views). Recent architectures inject specialized inductive biases into this feed-forward prediction: pixelSplat (Charatan et al., 2024) utilizes epipolar attention, MVSplat (Chen et al., 2024) and MVSGaussian (Liu et al., 2024) incorporates cost volumes, and DepthSplat (Xu et al., 2025b) leverages pre-trained depth priors. While these approaches achieve sub-second inference, they address only the optimization overhead and remain strictly dependent on ground-truth camera poses.

2.2 Pose-Free Feed-Forward 3DGS

To eliminate the reliance on ground-truth cameras, pose-free FF 3DGS leverage geometric foundation models (Wang et al., 2024; Leroy et al., 2024; Wang et al., 2025; 2026) to construct 3D representations directly from uncalibrated images. Specifically, early methods map Gaussians onto explicit 3D points (Smart et al., 2024) or relative canonical frames (Ye et al., 2025; Huang et al., 2025). Although these methods eliminate extrinsic pose supervision, they still require known camera intrinsics. Recent frameworks achieve high reconstruction fidelity even in fully uncalibrated settings. For instance, AnySplat (Jiang et al., 2025) voxelizes pixel-wise primitives for dense captures, YoNoSplat (Ye et al., 2026) aggregates per-view local Gaussians to improve scalability with intrin-

sic conditioning module, and 2Xplat (Jeong et al., 2026) decouples camera estimation and Gaussian synthesis into specialized experts. However, predicting full camera geometry together with Gaussian attributes inevitably compounds pose inaccuracies with feed-forward synthesis limits, resulting in suboptimal 3D representations.

2.3 Refining Feed-Forward Gaussians

Standard feed-forward models operate open-loop, leaving a quality gap relative to per-scene optimization. InstantSplat (Fan et al., 2024) reduces this gap with test-time optimization, but its iterative fitting incurs substantial latency. Diff3R (Liu et al., 2026) instead trains the network to predict an initialization that downstream test-time optimization can refine efficiently, yet it too relies on this iterative optimization at inference. Learned refiners avoid this overhead by directly predicting parameter updates: SplatFormer (Chen et al., 2025b) and ReSplat (Xu et al., 2025a) refine Gaussians using rendering residuals, yet both strictly assume ground-truth poses to establish exact 2D to 3D correspondences.

In pose-free settings, camera inaccuracies distort rays and decouple 2D residuals from true 3D locations. While concurrent work GIFSplat (Chen et al., 2026) addresses this spatial ambiguity using generative diffusion priors, we aim to resolve it geometrically by leveraging per-pixel ray directions and rendered depth alongside color residuals to guide Gaussian updates, effectively compensating for pose errors and enhancing visual fidelity.

3 Method

3.1 Preliminaries

Given a set of N images $\{I_i \in \mathbb{R}^{H\times W\times 3}\}_{i=1}^{N}$, we partition them into context views $\mathcal{C} = \{1,\dots,N_{\text{ctx}}\}$ and target views $\mathcal{T} = \{N_{\text{ctx}}+1,\dots,N\}$. A pose-free feed-forward predictor f_θ directly maps the context images to a global 3D scene representation and its associated camera geometry:

$$\big(\mathcal{G}_{\text{ff}},\ \{p_i\}_{i\in\mathcal{C}}\big) = f_\theta\big(\{I_i\}_{i\in\mathcal{C}}\big), \tag{1}$$

$$\mathcal{G}_{\text{ff}} = \big\{(\mu_g,\ \alpha_g,\ q_g,\ s_g,\ c_g^{(0)})\big\}_{g=1}^{G}. \tag{2}$$

The predicted collection $\mathcal{G}_{\text{ff}}$ comprises G 3D Gaussians, where each primitive g is characterized by its spatial mean $\mu_g \in \mathbb{R}^3$, opacity $\alpha_g \in (0,1)$, and anisotropic shape defined via rotation $q_g \in \mathbb{S}^3$ and scale $s_g \in \mathbb{R}^3_+$. Additionally, each Gaussian carries a degree-0 spherical harmonic (SH) coefficient $c_g^{(0)}$ to capture its view-independent color. These primitives are pixel-aligned with each context view, resulting in a total count of $G = N_{\text{ctx}}HW$. Each camera $p_i = [K_i, R_i, t_i]$ encapsulates the intrinsics $K_i \in \mathbb{R}^{3\times 3}$ and the extrinsic pose composed of a rotation $R_i \in \mathrm{SO}(3)$ and a translation $t_i \in \mathbb{R}^3$; both the intrinsics and the pose are predicted by f_θ, though either may instead be supplied when calibration is available.

To synthesize a novel view, a differentiable rasterizer $\mathcal{R}$ renders the Gaussian collection $\mathcal{G}$ at a target camera p_i expressed in the coordinate frame of the prediction:

$$\hat{I}_i = \mathcal{R}\big(\mathcal{G};\ p_i\big) \qquad \text{for } i \in \mathcal{T}. \tag{3}$$

3.2 Iterative Gaussian Refinement

To bridge the quality gap caused by single-pass prediction and camera estimation errors, we introduce a refinement module h_ϕ that progressively improves the Gaussian representation, as illustrated in Figure 2. We define $\mathcal{G}^{(k)}$ as the Gaussian set at the k-th refinement step, initializing the process with the backbone's feed-forward output, $\mathcal{G}^{(0)} = \mathcal{G}_{\text{ff}}$.

At each iteration $k \in \{0,\dots,K-1\}$, the current state $\mathcal{G}^{(k)}$ is rendered from the predicted context cameras p_i, yielding a rendering residual $r_i^{(k)}$ that captures the pixel-wise discrepancy between the

current rendered image $\hat{I}_i^{(k)}$ and the input context image I_i:

$$\hat{I}_i^{(k)} = \mathcal{R}\big(\mathcal{G}^{(k)}; p_i\big), \tag{4}$$

$$r_i^{(k)} = I_i - \hat{I}_i^{(k)}. \tag{5}$$

Conditioned on the residuals $r_i^{(k)} \in \mathbb{R}^{H\times W\times 3}$, the renderings $\hat{I}_i^{(k)} \in \mathbb{R}^{H\times W\times 3}$, the rendered depths $d_i^{(k)} \in \mathbb{R}^{H\times W}$, and the Plücker ray maps (Plücker, 1865) $\ell_i \in \mathbb{R}^{H\times W\times 6}$ of the context cameras, together with the current Gaussian state, the module predicts one update per Gaussian:

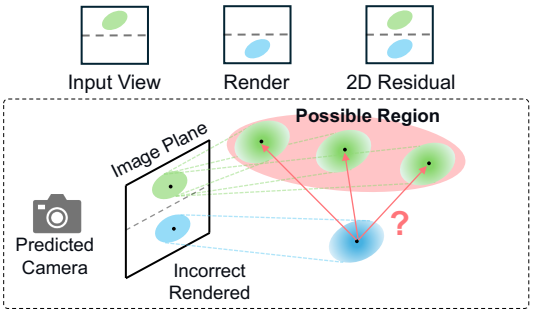
$$\Delta^{(k)} = h_\phi\big(\{r_i^{(k)}, \hat{I}_i^{(k)}, d_i^{(k)}, \ell_i\}_{i\in\mathcal{C}}, \mathcal{G}^{(k)}\big), \tag{6}$$

where each per-Gaussian update $\Delta_g^{(k)} \in \mathbb{R}^7$ consists of a 3D position shift $\Delta\mu_g \in \mathbb{R}^3$, an opacity adjustment $\Delta\alpha_g \in \mathbb{R}$, and a color correction $\Delta c_g^{(0)} \in \mathbb{R}^3$. The Gaussian state is then updated as

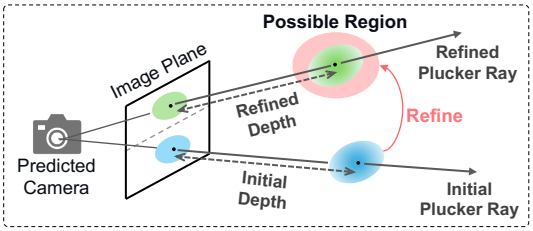
$$\mathcal{G}^{(k+1)} = \mathcal{G}^{(k)} \oplus \Delta^{(k)}, \tag{7}$$

where $\oplus$ adds each predicted increment to its respective attribute, while the scales s_g and rotations q_g remain fixed throughout the refinement, preserving the local structure established by the backbone. As in prior iterative estimation methods (Teed & Deng, 2020), the module weights are shared across all K iterations, and repeating this process K times yields the refined set $\mathcal{G}^{(K)}$.

3.3 Geometry-Guided Residual Encoding



(a) 2D Residual Only



(b) 2D Residual w/ Plucker Ray and Depth

Figure 3: **Geometry-guided residual encoding.** (a) The 2D residual alone underdetermines the position update $\Delta\mu$: displacements along the viewing ray yield identical residuals, leaving a large possible region (red). (b) The per-pixel Plücker ray and rendered depth encode the line of sight and the current prediction's position along it, a prior on the current geometry under which the possible region contracts.

Inferring a 3D positional update from 2D image-space residuals is inherently underdetermined. Because a 2D residual constrains corrections only within the image plane, displacing a Gaussian along its viewing ray yields an identical 2D projection, leaving the residual unchanged regardless of 3D spatial displacement (Figure 3 (a)). To resolve this spatial ambiguity, we explicitly condition the refinement on the 3D geometry of the current prediction by incorporating per-pixel Plücker rays and rendered depth, which describe the line of sight and the current prediction's position along it (Figure 3 (b)). This geometric context acts as a soft prior that regulates the direction and scale of the predicted corrections.

For each context view, we construct a 13-channel feature map by concatenating the image residual r_i, the current rendering $\hat{I}_i$, the Plücker-ray map ℓ_i, and the depth map d_i rendered alongside $\hat{I}_i$ by the same rasterizer. The entry $\ell_{i,u}$ of ℓ_i at pixel $u = (x, y)$ is given by

$$\ell_{i,u} = \big(\mathbf{d}_{i,u},\ \mathbf{o}_i \times \mathbf{d}_{i,u}\big) \in \mathbb{R}^6, \tag{8}$$

where $\mathbf{o}_i \in \mathbb{R}^3$ is the center of camera p_i and $\mathbf{d}_{i,u} \in \mathbb{R}^3$ the unit direction of the ray from $\mathbf{o}_i$ through pixel u.

Each input carries distinct information: the residual r_i measures the per-pixel photometric discrepancy, ℓ_i specifies for each pixel the line of sight along which the positional correction is ambiguous, and d_i locates the current prediction along this line. Conditioning on ℓ_i and d_i thus exposes the geometric structure of the ambiguity directly in the input, rather than requiring the network to recover it from appearance. Furthermore, the rays and the depth are computed from exactly

Table 1: Main results on the DL3DV testset (140 scenes, 224^2) at 6 / 12 / 24 context views. p, K: ✓ means the ground-truth pose / intrinsics are supplied, blank means predicted by the model.

			6 views			12 views			24 views		
Method	p	K	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
MVSplat (Chen et al., 2024)	✓	✓	22.66	0.760	0.173	21.29	0.709	0.224	19.98	0.662	0.269
DepthSplat (Xu et al., 2025b)	✓	✓	23.42	0.797	0.136	21.91	0.753	0.179	20.09	0.690	0.240
YoNoSplat (Ye et al., 2026)	✓	✓	24.72	0.817	0.139	23.29	0.773	0.177	22.67	0.758	0.192
DeltaSplat	✓	✓	**26.70**	**0.866**	**0.113**	**25.33**	**0.833**	**0.142**	**24.72**	**0.824**	**0.151**
NoPoSplat (Ye et al., 2025)		✓	22.77	0.743	0.179	19.38	0.563	0.318	17.86	0.495	0.397
YoNoSplat (Ye et al., 2026)		✓	24.89	0.819	0.138	23.15	0.758	0.183	22.35	0.731	0.205
DeltaSplat		✓	**26.64**	**0.862**	**0.115**	**25.05**	**0.820**	**0.148**	**24.21**	**0.795**	**0.167**
AnySplat (Jiang et al., 2025)			19.03	0.554	0.235	18.94	0.549	0.262	19.70	0.596	0.249
YoNoSplat (Ye et al., 2026)			24.53	0.804	0.142	22.93	0.746	0.187	22.17	0.720	0.209
DeltaSplat			**25.69**	**0.831**	**0.126**	**24.34**	**0.791**	**0.159**	**23.66**	**0.771**	**0.177**

the cameras used to place and render the Gaussians. Whether these cameras are externally calibrated or estimated by the backbone, all inputs to h_ϕ are thus expressed in the coordinate frame of the current prediction: an error in the estimated cameras displaces the residual and the geometric channels together, leaving them mutually consistent. Refinement therefore remains well-defined in the pose-free setting, where no ground-truth cameras are available.

3.4 Delta Decoder

The decoder transforms the geometry-guided features into per-Gaussian updates in two stages: an image-space dual-branch mixer followed by per-attribute heads.

Dual-branch mixer. The update requires spatial context at two scales: per-pixel detail at full resolution and broader structure that only a wide receptive field provides. The mixer must also remain fast, because it runs at every refinement iteration and its cost enters the total inference time K times. It therefore processes the encoding through two branches operating at different resolutions. The coarse branch rearranges the input from $H \times W \times C$ to $\frac{H}{2} \times \frac{W}{2} \times 4C$ via pixel-unshuffle (Shi et al., 2016), a lossless space-to-depth reshaping, and applies dilated convolutions (Yu & Koltun, 2016) at this half resolution; the number of spatial positions drops to a quarter while dilation further enlarges the receptive field, so the branch covers broad structural context at a fraction of the full-resolution cost. The detail branch instead stays at the full $H \times W$ resolution and applies a few lightweight convolutions, preserving the high-frequency residual signal. The two branch outputs are fused additively into a single feature map, from which the per-Gaussian features of the next stage are sampled. The resulting mixer surpasses a full-resolution single-branch variant in quality at $1.84\times$ lower time overhead, since most of its computation runs on the quarter-area grid.

Per-attribute head. To map the image-space features to Gaussian updates, each Gaussian samples the fused feature map at its projected mean in each context view, and the per-view samples are aggregated by an average weighted by the Gaussian's rasterizer blending weights. The aggregated feature is concatenated with the Gaussian's current parameters and decoded into the update. The position, opacity, and color updates differ substantially in numerical scale, so decoding them with a single shared MLP lets the largest signal dominate the others; we therefore employ three parallel MLPs, one per attribute. All output layers are zero-initialized so that refinement starts exactly at $\Delta = 0$ (Zhang et al., 2023).

3.5 Training and Inference

The backbone f_θ and the refinement module h_ϕ are trained jointly in an end-to-end manner by supervising the output of the final refinement iteration: the loss is computed after all K refinement iterations, and its gradients propagate through the rasterizer and the entire unrolled sequence of updates. The objective combines a pixel-wise mean squared error with a perceptual

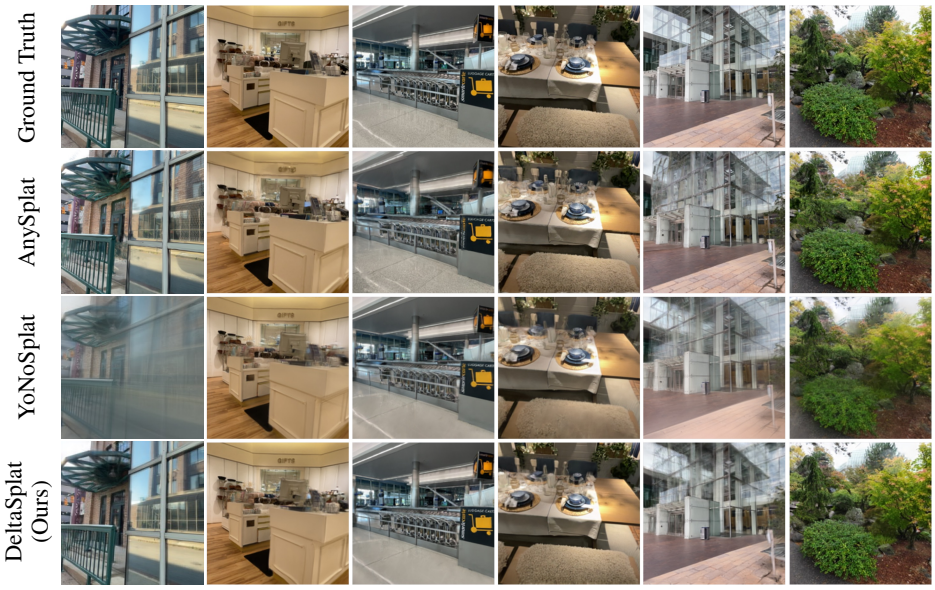


Figure 4: Qualitative results on DL3DV scenes.

LPIPS (Zhang et al., 2018) term over all rendered views:

$$\mathcal{L} = \sum_{i \in \mathcal{C} \cup \mathcal{T}} \Big(\big\| \hat{I}_i^{(K)} - I_i \big\|_2^2 + \lambda_{\text{lp}} \, \text{LPIPS}\big(\hat{I}_i^{(K)}, \, I_i\big) \Big), \tag{9}$$

where $\hat{I}_i^{(K)}$ is the rendering of the final Gaussian set $\mathcal{G}^{(K)}$ at camera p_i and $\| \cdot \|_2^2$ denotes the mean squared error over pixels. The context terms anchor the refined Gaussians to the observed input images, whereas the target terms supervise generalization to novel viewpoints. We set $\lambda_{\text{lp}} = 0.1$ with the VGG backbone.

Training runs for 50 k steps with $K{=}3$ and shared weights across iterations. At test time, h_ϕ is applied $K{=}3$ times to its own output, consistent with training; further implementation details are provided in Appendix A.

4 Experiments

4.1 Experimental Setup

Dataset. We train on the DL3DV-10K (Ling et al., 2024) trainset and evaluate on 140-scene testset. All images are processed at a resolution of 224×224, and experiments are conducted with 6, 12, and 24 context views.

Model configuration. We use YoNoSplat (Ye et al., 2026), pre-trained from a π^3 model (Wang et al., 2026), as the backbone. Our model is initialized from its official DL3DV-trained checkpoint and trained for 50 k steps with a batch size of one scene per step on a single NVIDIA RTX PRO 6000 GPU.

Evaluation metrics. We report PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) (VGG), averaged over the test scenes. We follow the evaluation protocol of YoNoSplat (Ye et al., 2026), where target camera poses in the pose-free settings are estimated by photometric optimization before rendering. Unless otherwise specified, results use the refined Gaussians after $K{=}3$ iterations.

4.2 Main Results

Pose-free comparison. Table 1 summarizes the main results. In the pose-free setting with known intrinsics, DeltaSplat reaches 26.64 dB PSNR at 6 views, improving its feed-forward backbone by

Table 2: Cross-dataset generalization to ScanNet++ (Yeshwanth et al., 2023) at 224^2 with 32 / 64 / 128 context views and known intrinsics. All models are trained on DL3DV.

	32 views			64 views			128 views		
Method	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
YoNoSplat	17.94	0.659	0.380	18.83	0.688	0.342	19.28	0.701	0.325
DeltaSplat	**18.62**	**0.670**	**0.371**	**19.76**	**0.711**	**0.328**	**20.62**	**0.741**	**0.298**

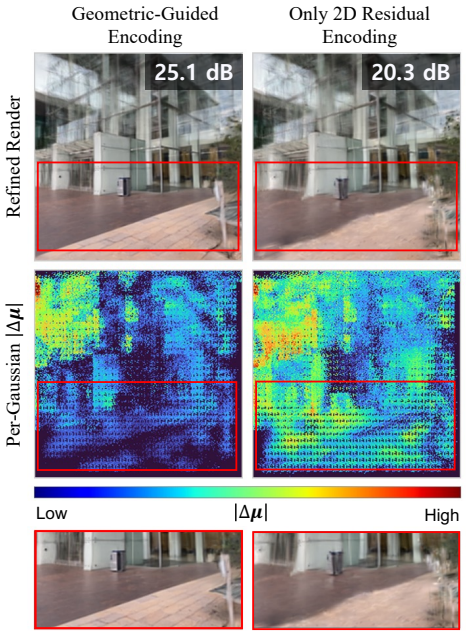


Figure 5: **Effect of the 3D geometry prior.** Columns compare the geometry-aimed encoding with an encoding built from the 2D residual alone; rows show the refined render, the per-Gaussian $|\Delta\mu|$, and a zoom of the boxed region. Without the Plücker-ray and depth channels the same residual drives larger and more scattered position updates, and PSNR falls from 25.1 to 20.3 dB.

Table 3: Generalization to RE10K (Zhou et al., 2018) (224^2, 6 context views). Baselines are trained on RE10K, whereas DeltaSplat is transferred zero-shot from DL3DV. p, K: ✓ means the ground-truth pose / intrinsics are supplied.

Method	p	K	PSNR↑	SSIM↑	LPIPS↓
DepthSplat	✓	✓	24.16	0.846	0.145
YoNoSplat	✓	✓	25.04	0.848	0.134
NoPoSplat		✓	22.18	0.750	0.207
YoNoSplat		✓	25.40	0.857	0.131
DeltaSplat		✓	**26.30**	**0.865**	**0.129**
YoNoSplat			24.57	0.823	0.144
DeltaSplat			**24.87**	**0.830**	**0.142**

Table 4: Component ablations on the DL3DV test set in the pose-free setting with known intrinsics (6 context views). Each row removes a single component from the full model.

Variant	PSNR↑	SSIM↑	LPIPS↓
Full model	**26.64**	**0.862**	**0.115**
w/o geometric conditioning	25.27	0.823	0.146
w/o dual-branch mixer (single-branch mixer)	26.05	0.846	0.132
w/o per-attribute MLP (single MLP)	26.22	0.851	0.127

+1.75 dB, with consistent margins of +1.90 and +1.86 dB at 12 and 24 views; SSIM and LPIPS improve accordingly. Notably, DeltaSplat also surpasses every baseline supplied with ground-truth cameras (26.64 vs. at best 24.72 dB at 6 views): the gain from refinement exceeds what the backbone gains from access to exact cameras.

Fully uncalibrated setting. When the intrinsics are also predicted, DeltaSplat attains 25.69 dB at 6 views, again above every baseline supplied with ground-truth cameras, and the margin over the backbone widens with more views (+1.16 to +1.49 dB).

Qualitative results. Figure 4 compares DeltaSplat with the baselines on DL3DV scenes.

Efficiency. The refinement module adds ∼2.2% to the backbone parameters, and the K=3 refinement iterations add ∼0.11 s per scene on top of the backbone forward pass. By contrast, the optional post-optimization of YoNoSplat (Ye et al., 2026) requires 17.7–165 s per scene depending on the view count, while DeltaSplat remains entirely feed-forward.

Table 5: Effect of the number of refinement iterations K applied at test time, with per-scene refinement time, on the DL3DV test set in the pose-free setting with known intrinsics (6 context views). $K{=}0$ denotes the unrefined backbone output; the model is trained with $K{=}3$.

	Backbone	Refinement iterations					
K	0	1	2	3	4	5	6
PSNR↑	24.80	26.02	26.45	**26.64**	26.48	26.32	26.17
SSIM↑	0.818	0.847	0.857	**0.862**	0.858	0.853	0.848
LPIPS↓	0.140	0.126	0.118	**0.115**	0.119	0.124	0.130
Time (ms)↓	54	90	125	**160**	194	230	266

4.3 Ablation Study

Geometric conditioning. Omitting the Plücker-ray and depth channels forces the network to infer 3D corrections solely from 2D features, with two consequences shown in Table 4. First, overall visual quality degrades: removing the geometric channels reduces PSNR by 1.37 dB, canceling about three quarters of the 1.84 dB gain that refinement provides over the unrefined backbone output. Second, positional updates lose spatial directionality: the magnitude of spatial corrections $|\Delta\mu|$ becomes substantially larger and more dispersed (Figure 5), indicating that corrections applied without explicit geometric frames lack directional precision. The geometric channels thus supply exactly the information that the 2D residual underdetermines: the line of sight of each pixel and the current prediction's position along it.

Mixer and head design. Table 4 also compares the dual-branch mixer against a single-branch variant that processes the encoding entirely at full resolution, and the per-attribute heads against a single shared MLP. Removing the dual-branch design costs 0.59 dB, as broad structural context must then be gathered entirely at full resolution. Collapsing the per-attribute heads into a single MLP reduces PSNR by 0.42 dB, since the update channels differ in numerical scale and a shared decoder lets the largest signal dominate.

Refinement iterations. Table 5 varies the number of refinement iterations applied at test time. Quality improves monotonically over the first three iterations and peaks at the training setting $K{=}3$, while the refinement time grows linearly from 54 ms for the unrefined backbone output to 266 ms at $K{=}6$. Iterating beyond the training horizon degrades quality only gradually without destabilizing the prediction, and yields no further gains, so $K{=}3$ offers the best quality–cost trade-off.

Cross-dataset generalization. Tables 2 and 3 evaluate DeltaSplat beyond the DL3DV training distribution. On ScanNet++ (Yeshwanth et al., 2023), the DL3DV-trained models are evaluated without any fine-tuning, using 32, 64, and 128 context views with known intrinsics. DeltaSplat outperforms the backbone at every view count, and the margin grows from $+0.68$ dB at 32 views to $+1.34$ dB at 128 views, following the same trend as on DL3DV. On RE10K (Zhou et al., 2018) with 6 context views, the baselines are trained on RE10K while DeltaSplat is transferred zero-shot from DL3DV. DeltaSplat attains 26.30 dB with known intrinsics and 24.87 dB in the fully uncalibrated setting, surpassing the backbone in both settings. The refinement thus transfers to unseen datasets without any adaptation.

5 Conclusion

We introduced DeltaSplat, a lightweight Gaussian refinement module for pose-free feed-forward 3DGS. Conditioning image-space residuals on per-pixel Plücker rays and rendered depth, it lifts 2D evidence into 3D updates that progressively refine the position, opacity, and color of the predicted Gaussians. With $\sim 2.2\%$ additional parameters and a fully feed-forward inference path, DeltaSplat consistently improves a state-of-the-art backbone across camera regimes, view counts, and datasets, offering a general plug-in quality stage for pose-free feed-forward reconstruction. A natural next step is to extend the refinement to higher resolutions and to other feed-forward backbones.

AI USE STATEMENT

We did not use generative AI tools for any task that requires disclosure under the ICLR 2027 AI policy: the research idea, methodology, experimental design, implementation, experiments, and interpretation of results are the authors' own work, and the paper text was written by the authors. We used a large language model (LLM) assistant for tasks with recommended disclosure: paraphrasing author-written sentences to improve readability, cross-checking the numbers reported in the text against the tables, formatting the manuscript and the references in the ICLR template, and drafting the reproducibility statement. All AI-assisted edits were reviewed and approved by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 3.5 specifies the training objective and schedule, and Section 4.1 the datasets, backbone initialization, and evaluation protocol. Appendix A lists the optimizer settings and the architecture configuration of the mixer and heads, Appendix B the timing protocol, and Appendix D the evaluation-time pose alignment applied identically to all pose-free methods. All datasets used (DL3DV, RE10K, and ScanNet++) are publicly available.

REFERENCES

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D Gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *CVPR*, 2024.

Tianyu Chen, Wei Xiang, Kang Han, Yu Lu, Di Wu, Gaowen Liu, and Ramana Rao Kompella. GIF-Splat: Generative prior-guided iterative feed-forward 3D Gaussian splatting from sparse views. In *CVPR*, 2026.

Youyu Chen, Junjun Jiang, Kui Jiang, Xiao Tang, Zhihao Li, Xianming Liu, and Yinyu Nie. Dash-Gaussian: Optimizing 3D Gaussian splatting in 200 seconds. In *CVPR*, 2025a.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D Gaussian splatting from sparse multi-view images. In *ECCV*, 2024.

Yutong Chen, Marko Mihajlovic, Xiyi Chen, Yiming Wang, Sergey Prokudin, and Siyu Tang. Splat-Former: Point transformer for robust 3D Gaussian splatting. In *ICLR*, 2025b.

Zheng Dong, Daifei Qiu, Pinxuan Dai, Ke Xu, Jiamin Xu, Lili He, Rynson WH Lau, and Weiwei Xu. TurboGS: Accelerating 3D Gaussian splatting via error-guided sparse pixel sampling and optimization. In *ICML*, 2026.

Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, Zhangyang Wang, and Yue Wang. InstantSplat: Unbounded sparse-view pose-free Gaussian splatting in 40 seconds, 2024.

Guangchi Fang and Bing Wang. Mini-Splatting: Representing scenes with a constrained number of Gaussians. In *ECCV*, 2024.

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 *CVPR*, 2025.

Ranran Huang et al. No pose at all: Self-supervised pose-free 3D Gaussian splatting from sparse views, 2025.

Hwasik Jeong, Seungryong Lee, Gyeongjin Kang, Seungkwon Yang, Xiangyu Sun, Seungtae Nam, and Eunbyung Park. 2Xplat: Two experts are better than one generalist, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. AnySplat: Feed-forward 3D Gaussian splatting from unconstrained views. *ACM Transactions on Graphics*, 44(6), 2025.

Gyeongjin Kang, Seungkwon Yang, Seungtae Nam, Younggeun Lee, Jungwoo Kim, and Eunbyung Park. Multi-view pyramid transformer: Look coarser to see broader. In *CVPR*, 2026.

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

Joo Chan Lee, Jong Hwan Ko, and Eunbyung Park. Optimized minimal 3D Gaussian splatting. In *NeurIPS*, 2025.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3D with MASt3R. In *ECCV*, 2024.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. DL3DV-10K: A large-scale scene dataset for deep learning-based 3D vision. In *CVPR*, 2024.

Tianqi Liu, Guangcong Wang, Shoukang Hu, Liao Shen, Xinyi Ye, Yuhang Zang, Zhiguo Cao, Wei Li, and Ziwei Liu. MVSGaussian: Fast generalizable Gaussian splatting reconstruction from multi-view stereo. In *ECCV*, 2024.

Yueh-Cheng Liu, Jozef Hladký, Matthias Nießner, and Angela Dai. Diff3R: Feed-forward 3D Gaussian splatting with uncertainty-aware differentiable optimization, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Markus Steinberger, Francisco Vicente Carrasco, and Fernando De La Torre. Taming 3DGS: High-quality radiance fields with limited resources. In *SIGGRAPH Asia*, 2024.

Julius Plücker. XVII. On a new geometry of space. *Philosophical Transactions of the Royal Society of London*, 155:725–791, 1865.

Shiwei Ren, Tianci Wen, Yongchun Fang, and Biao Lu. FastGS: Training 3D Gaussian splatting in 100 seconds. In *CVPR*, 2026.

Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016.

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016.

Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3R: Zero-shot Gaussian splatting from uncalibrated image pairs, 2024.

Zachary Teed and Jia Deng. RAFT: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *CVPR*, 2025.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *CVPR*, 2024.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, et al. π^3: Scalable permutation-equivariant visual geometry learning. In *ICLR*, 2026.

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.

Haofei Xu, Daniel Barath, Andreas Geiger, and Marc Pollefeys. ReSplat: Learning recurrent Gaussian splatting, 2025a.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting Gaussian splatting and depth. In *CVPR*, 2025b.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3D Gaussian splats from sparse unposed images. In *ICLR*, 2025.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. YoNoSplat: You only need one model for feedforward 3D Gaussian splatting. In *ICLR*, 2026.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. ScanNet++: A high-fidelity dataset of 3D indoor scenes. In *ICCV*, 2023.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018.

A Implementation Details

Training. The model is trained for 50 k steps on DL3DV with AdamW (Loshchilov & Hutter, 2019) (peak learning rate 4×10^{-5}, linear warmup for 100 steps followed by cosine decay to zero, weight decay 10^{-5}, gradient clipping at 1.0), using a batch size of one scene per step without gradient accumulation. Training unrolls the $K{=}3$ refinement iterations and applies the loss of Eq. (9) to the renders produced after the final iteration; gradients propagate through the differentiable rasterizer and the unrolled sequence of updates. The forward pass runs in bf16 autocast, while losses and the backward pass are computed in fp32. The evaluated checkpoint stores color at spherical-harmonic degree 0, so the $\Delta c^{(0)}$ update acts on the full color.

Architecture. The coarse branch of the dual-branch mixer operates at half resolution via pixel-unshuffle ($s{=}2$) with 768 channels and dilation rates $(1, 1, 2, 2)$, while the detail branch applies three 64-channel convolutions at full resolution. Each per-attribute head is a three-layer MLP with hidden dimension 256 and SiLU activations, decoding the position (3), opacity (1), and color (3) updates; all output layers are zero-initialized. The complete configuration is provided with the released code.

B Inference Time Breakdown

Table 6 breaks down the per-scene reconstruction time at $K{=}3$ on DL3DV (6 context views, 224^2, bf16), measured as the median over 20 scenes after three warm-up scenes, with CUDA synchronization around every segment. The final iteration is cheaper than the first two because no context re-rendering is required after the last update. The unrefined backbone corresponds to the encoder and context-rendering segments, and the totals are consistent with Table 5.

C Per-Scene Analysis

Figure 6 compares DeltaSplat against its backbone scene by scene on the 140 DL3DV test scenes (6 context views, pose-free with known intrinsics). DeltaSplat improves every one of the 140 scenes, with an average gain of $+1.75$ dB, and the improvement is spread across the full quality range rather than driven by a few outlier scenes.

Table 6: Per-scene reconstruction time on DL3DV (6 context views, 224^2).

Segment	Time (ms)
Backbone encoder	43.1
Context-view rendering	11.1
Refinement iteration 1	38.4
Refinement iteration 2	38.3
Refinement iteration 3	27.4
Total reconstruction (K=3)	158.3

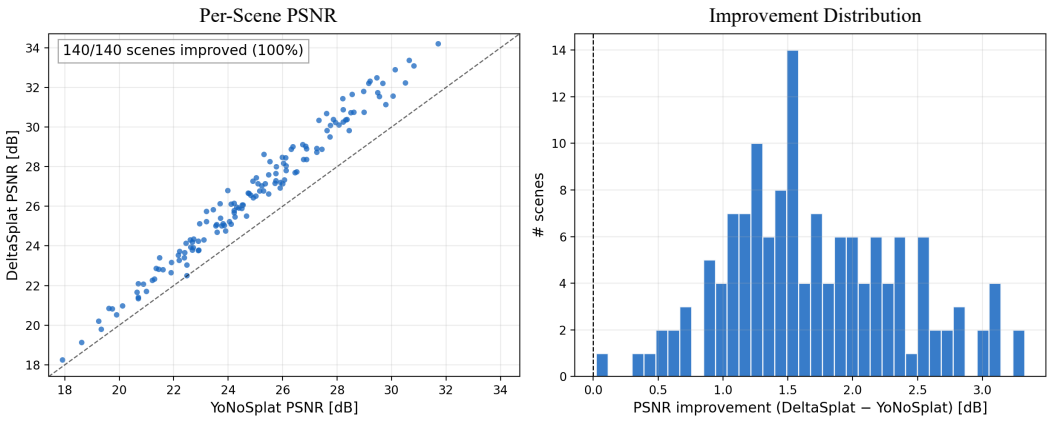


Figure 6: Per-scene PSNR of DeltaSplat versus the YoNoSplat backbone on DL3DV (140 scenes, 6 context views, pose-free with known intrinsics).

D Results without Evaluation-Time Pose Alignment

In the pose-free settings, the scene is reconstructed in the camera frame predicted by the model, which is a valid coordinate system of its own but differs from the SfM-derived frame in which the dataset expresses the target poses. Rendering a target view therefore requires expressing its camera in the predicted frame. Following the evaluation protocol of YoNoSplat (Ye et al., 2026), the target extrinsics are aligned by optimizing a photometric loss against the held-out target image before rendering (a per-view SE(3) correction, optimized with Adam for 100 steps at the official default learning rates), identically for all pose-free baselines and for our method.

Table 7 reports the pose-free results of Table 1 without this evaluation-time pose alignment, i.e., rendering directly at the dataset-provided target extrinsics. In this setting the metric conflates reconstruction quality with the discrepancy between the predicted camera frame and the ground-truth frame; LPIPS, which is less sensitive to small misalignments, still improves over the backbone in all settings.

E Applying DeltaSplat to Other Backbones

To assess whether the refinement generalizes beyond its primary backbone, we attach DeltaSplat to NoPoSplat (Ye et al., 2025) and AnySplat (Jiang et al., 2025), keeping each backbone frozen and training only the refinement module on DL3DV. Table 8 reports paired evaluations on the DL3DV test set in the pose-free setting, with each backbone evaluated in its native configuration (NoPoSplat: two context views with known intrinsics; AnySplat: six context views). DeltaSplat improves PSNR and SSIM for both backbones. The gain is also complementary to backbone adaptation: fine-tuning the AnySplat backbone on DL3DV reaches 18.04 dB, and applying the refinement on top still adds a further $+0.18$ dB.

Table 7: Pose-free results without evaluation-time pose alignment (DL3DV, 224^2). K: ✓ means ground-truth intrinsics are supplied.

		6 views		12 views		24 views	
Method	K	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓
YoNoSplat	✓	23.32	0.156	20.95	0.210	20.22	0.235
DeltaSplat	✓	24.04	0.139	21.36	0.197	20.68	0.219
YoNoSplat		22.29	0.173	20.38	0.229	19.71	0.255
DeltaSplat		22.41	0.165	20.45	0.217	19.81	0.240

Table 8: DeltaSplat applied to other feed-forward backbones on the DL3DV test set (pose-free). Base numbers are our own paired re-evaluation under each backbone's native configuration; the backbone is frozen in the +DeltaSplat rows.

Method	PSNR↑	SSIM↑	LPIPS↓
NoPoSplat	22.18	0.716	0.182
+ DeltaSplat	**22.77**	**0.739**	**0.179**
AnySplat	16.85	0.495	**0.368**
+ DeltaSplat	**17.19**	**0.501**	0.371

F Additional Qualitative Results

Figures 7 and 8 provide additional qualitative comparisons under the pose-free evaluation protocol. Across ScanNet++ with 32 to 128 context views and RE10K, the refinement sharpens blurred regions and restores structures that the unrefined backbone misses.

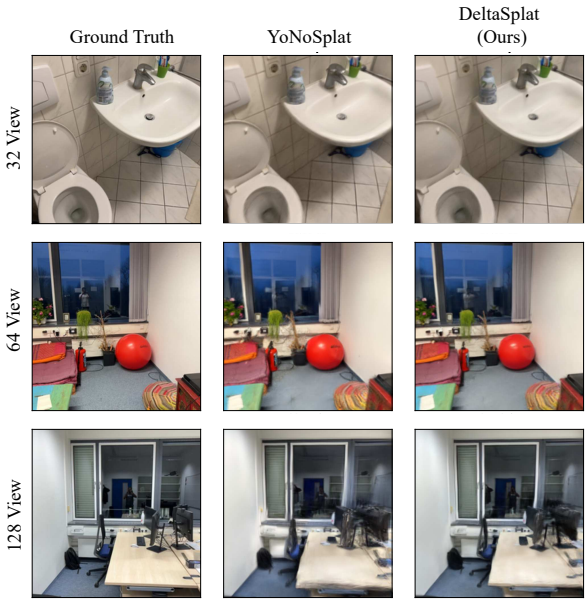


Figure 7: Qualitative cross-dataset generalization on ScanNet++ with 32, 64, and 128 context views (models trained on DL3DV, known intrinsics).

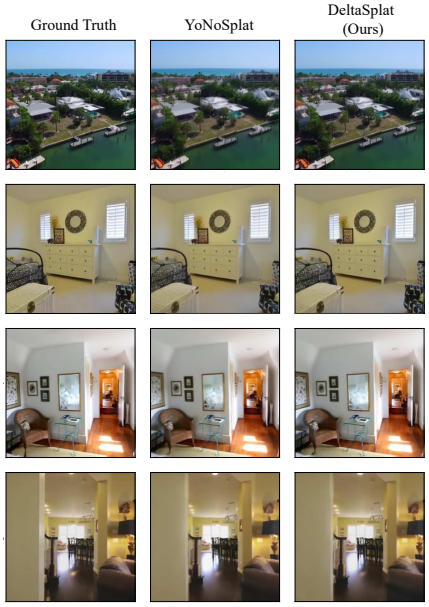


Figure 8: Additional qualitative results on RE10K (6 context views, pose-free).