SAMPLESPLAT: EFFICIENT FEED-FORWARD 3D GAUSSIAN SPLATTING VIA SCENE-AWARE SPATIAL SAMPLING

Anonymous authors
Paper under double-blind review

ABSTRACT

Feed-forward 3D Gaussian Splatting enables rapid scene reconstruction, but pixel-aligned Gaussian allocation couples representation capacity to input resolution and view count, preventing adaptive allocation of the Gaussian budget according to scene complexity. To address this limitation, we present **SampleSplat**, a feed-forward framework that decouples Gaussian generation from the input pixel grid through **scene-aware probabilistic sampling**. Given 3D anchors constructed from image features and estimated geometry, our sampling module jointly learns anchor selection and spatial offsets, enabling Gaussians to be adaptively allocated and positioned under a shared budget. The module is trained end-to-end using only reconstruction supervision, without handcrafted complexity measures or auxiliary allocation labels. As a result, SampleSplat supports **variable-view inputs, including monocular input**, and **flexible input resolutions within a single model**, while enabling **adaptive allocation** under a controllable Gaussian budget Experiments on DL3DV-10K demonstrate state-of-the-art novel view synthesis performance across these settings, with monocular reconstruction taking 90.97 ms at the native resolution of 540×960.

1 INTRODUCTION

Rapidly reconstructing renderable 3D scenes from one or multiple images is essential for novel view synthesis and 3D content creation. 3D Gaussian Splatting (3DGS) achieves high-quality real-time rendering through explicit scene representation and efficient rasterization (Kerbl et al., 2023), but its iterative per-scene optimization incurs substantial reconstruction costs. Feed-forward methods (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025) learn to map images directly to Gaussian representations in a single forward pass, enabling efficient scene reconstruction. NoPoSplat (Ye et al., 2025) and SPFSplat (Huang & Mikolajczyk, 2025) further reduce the reliance on known input camera poses. AnySplat (Jiang et al., 2025) and YoNoSplat (Ye et al., 2026a) enable scene reconstruction from uncalibrated images, extending its applicability to casual capture settings.

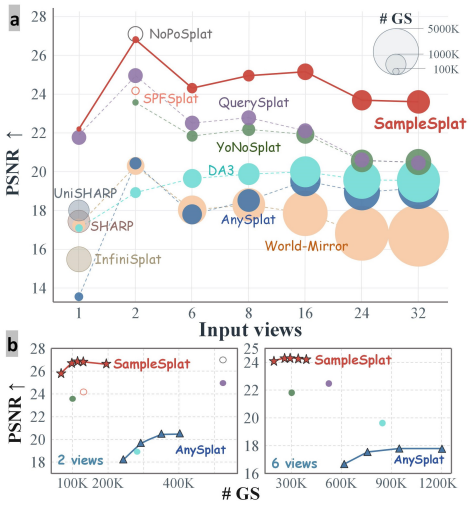
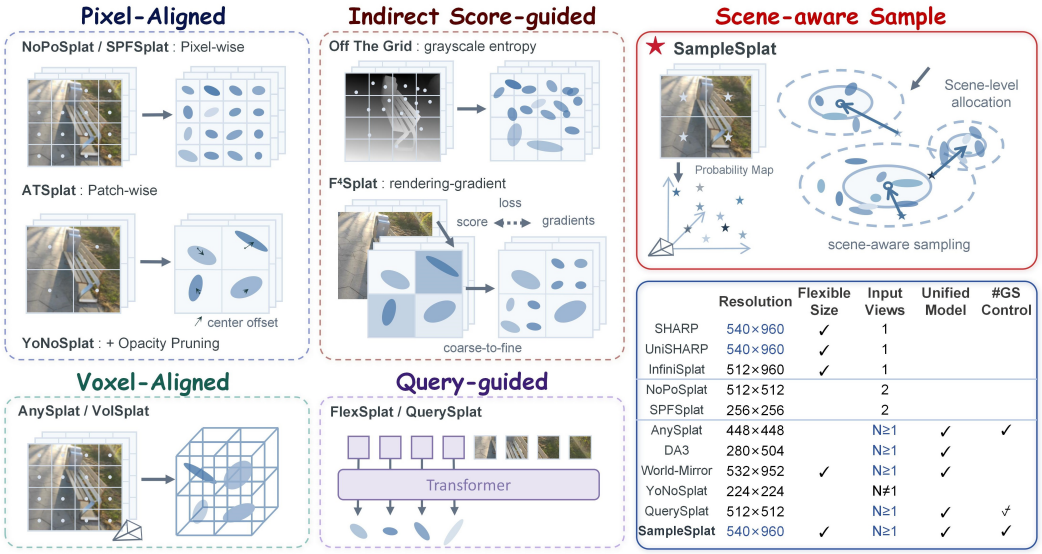


Figure 1: **(a) Performance and Gaussian efficiency under variable-view inputs. (b) Rendering quality vs. #GS.** SampleSplat maintains high rendering quality while using fewer Gaussians across both settings.

Many feed-forward methods generate Gaussians by assigning a predefined number of primitives to each pixel or local image region (Charatan et al., 2024; Chen et al., 2024; Ye et al., 2025; Singhal et al., 2026). This design provides direct geometric initialization but rigidly ties representation capacity to input resolution and the number of views. When the number, resolution, and aspect ratio of in-

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.



	Resolution	Flexible Size	Input Views	Unified Model	#GS Control
SHARP	540×960	✓	1		
UniSHARP	540×960	✓	1		
InfiniSplat	512×960	✓	1		
NoPoSplat	512×512		2		
SPFSplat	256×256		2		
AnySplat	448×448		N≥1	✓	✓
DA3	280×504		N≥1	✓	
World-Mirror	532×952	✓	N≥1	✓	
YoNoSplat	224×224		N≠1		
QuerySplat	512×512		N≥1	✓	✓
SampleSplat	540×960	✓	N≥1	✓	✓

Figure 2: **Overview of existing strategies for flexible Gaussian organization.** The comparison of model capabilities and settings highlights SampleSplat as the most flexible framework, supporting flexible input sizes, variable-view inputs, unified modeling, and explicit Gaussian-budget control.

put images vary, many existing methods cannot accommodate these configurations within a single model, limiting their applicability across capture settings (Ye et al., 2025; Mescheder et al., 2026; Jiang et al., 2025; Ye et al., 2026a). Moreover, **the number of Gaussians needed is not necessarily proportional to the input pixel count**: flat regions and complex boundaries have different modeling requirements, while uniform sampling across overlapping views can introduce artifacts when their estimated poses are imperfect. Therefore, increasing Gaussian counts in proportion to input size can lead to **inefficient budget use, high computational costs, and additional noise**.

As shown in Figure 2, prior works explore more flexible Gaussian organization beyond pixel grids. AnySplat (Jiang et al., 2025) aggregates per-pixel predictions through differentiable voxelization, while QuerySplat (Li et al., 2026) generates Gaussians with learnable queries. Beyond spatial flexibility, recent methods also adapt Gaussian density to local reconstruction complexity. Off The Grid (Moreau et al., 2026) uses patch-level grayscale entropy to assign density levels, whereas F4Splat (Kim et al., 2026) and ATSplat (In et al., 2026) learn densification and uncertainty scores supervised by rendering gradients and reconstruction errors, respectively. These methods demonstrate the value of allocating more capacity to challenging regions, but rely on handcrafted complexity measures or explicitly constructed auxiliary scoring targets. Therefore, it is worth exploring how to **learn regional Gaussian allocation directly through reconstruction supervision**, using scene features to distribute a prescribed budget according to local modeling needs.

Accordingly, we propose **SampleSplat**, a feed-forward framework that reconstructs 3D Gaussian scenes from one or multiple uncalibrated images through scene-aware spatial sampling. As shown in Figure 3, we construct 3D anchors using geometric features and camera estimates from DA3 (Lin et al., 2025), together with local appearance and learned GS depth, to support scene-aware Gaussian sampling. The core of SampleSplat is a **scene-aware probabilistic sampling** module that jointly learns how to allocate and position Gaussians under a shared budget. The model predicts selection probabilities across anchors and offset means and standard deviations for each anchor, then samples Gaussian centers and decodes their attributes. Trained solely through reconstruction supervision, the module requires no explicit complexity labels.

On DL3DV-10K (Ling et al., 2024), SampleSplat achieves state-of-the-art performance across monocular, sparse-view, and dense-view settings. As shown in Figure 1, it attains high rendering quality with substantially fewer Gaussians while supporting flexible control over the Gaussian budget. For monocular reconstruction at 540×960 resolution, SampleSplat requires only **90.97 ms**,

with learned sampling adaptively allocating more Gaussians to object boundaries and regions with complex textures. Our main contributions are as follows:

- We propose SampleSplat, **a unified feed-forward framework** for uncalibrated 3D scene reconstruction. One model supports variable-view inputs, including monocular input, accommodates flexible image sizes and allows explicit control over the Gaussian budget.
- We design an anchor selection and conditional spatial sampling module that jointly allocates and positions Gaussians according to local reconstruction demands under a controllable budget. By fully utilizing the representational capacity of each Gaussian, the module enables **fast**, **high-fidelity** reconstruction with **far fewer gaussians** than existing approaches.
- We achieve **state-of-the-art** performance across monocular, sparse-view, and dense-view settings on DL3DV-10K across a range of evaluation metrics, reaching 26.40 **dB** with only two input views. At 540×960 resolution, monocular reconstruction takes **only** 90.97 **ms**.

2 Related Work

2.1 Feed-Forward 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) enables high-quality, real-time novel view synthesis but requires time-consuming per-scene optimization. Feed-forward methods accelerate reconstruction by predicting Gaussian parameters from images. While pixelSplat, MVSplat, and DepthSplat (Charatan et al., 2024; Chen et al., 2024; Xu et al., 2025) rely on known camera parameters, NoPoSplat (Ye et al., 2025) removes the need for input camera poses, and AnySplat and YoNoSplat (Jiang et al., 2025; Ye et al., 2026a) support uncalibrated images. However, two limitations remain. First, methods such as pixelSplat and MVSplat employ fixed per-pixel Gaussian allocation (Charatan et al., 2024; Chen et al., 2024), coupling representation size to image resolution and view count. This allocation does not adapt to scene complexity or cross-view redundancy, limiting representation efficiency and scalability. Second, existing methods exhibit limitations in input flexibility and resolution, such as supporting only specific view counts (Ye et al., 2025; Huang & Mikolajczyk, 2025; Mescheder et al., 2026), lacking single-image reconstruction capabilities (Ye et al., 2026a; 2025), or being restricted to fixed low resolutions and aspect ratios (Jiang et al., 2025; Ye et al., 2026a; Li et al., 2026). These constraints limit their applicability to diverse input settings.

2.2 3D Foundation Models

Through large-scale pretraining, 3D visual foundation models learn transferable geometric priors for spatial perception and scene reconstruction. DUSt3R (Wang et al., 2024) simplifies 3D reconstruction by directly regressing point maps from uncalibrated image pairs, while MASt3R (Leroy et al., 2024) improves image matching by combining geometric prediction with local feature learning. Subsequent models, including VGGT (Wang et al., 2025), Pi3 (Wang et al., 2026b), and DA3 (Lin et al., 2025), extend this direction towards more general feed-forward multi-view geometry prediction, recovering depth, camera parameters, and scene structure. These geometric priors also support downstream reconstruction and novel view synthesis. However, translating them into compact, high-quality Gaussian representations while accommodating variable-view inputs, and flexible input resolutions remains an important research question. Our work explores the integration of 3D foundation models with adaptive Gaussian reconstruction towards this goal.

3 SampleSplat Framework

To enable flexible and adaptive 3D reconstruction from uncalibrated images, we propose **SampleSplat**, a unified feed-forward framework that learns geometry-aware scene representations together with scene-aware probabilistic sampling. Given variable-view inputs, including monocular input, SampleSplat accommodates flexible image sizes and produces high-fidelity 3D Gaussian representations under an explicitly controllable Gaussian budget (Figure 3). Section 3.1 first constructs geometry-aware 3D anchors and scene features. Section 3.2 then uses them to allocate, sample, and

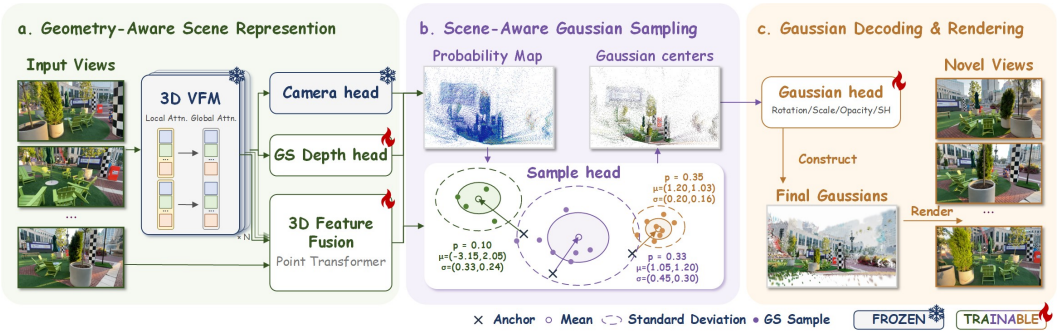


Figure 3: **Overview of SampleSplat.** (a) Geometry-Aware Scene Representation: Geometric features from a 3D VFM are fused with local appearance features to form a unified scene representation. (b) Scene-Aware Gaussian Sampling: Sampling probabilities, offset means, and standard deviations are predicted to adaptively sample Gaussian centers. (c) Gaussian Decoding and Rendering: Gaussian attributes are decoded to construct a 3D Gaussian representation for novel-view rendering.

decode Gaussians under a shared budget. Section 3.3 finally explains how reconstruction supervision jointly trains the sampling distributions and Gaussian representation.

3.1 Geometry-Aware Scene Representation

Pretrained 3D VFM Features We adopt Depth Anything 3 (DA3) (Lin et al., 2025) as our pretrained visual geometry model. Its multi-view geometric priors and 3D spatial representations support the reconstruction network in capturing cross-view correspondences and scene structure. During training, we freeze the DA3 geometry backbone and camera head while optimizing the feature adapters and GS depth head. Multi-level single-view and multi-view features preserve local image structure and cross-view geometric information, respectively, and are mapped to a shared spatial grid by the adapters. Initialized with pretrained depth-head weights, the GS depth head learns reconstruction-oriented depth $\widehat{D}_i$ and confidence $\widehat{C}_i$ through reconstruction supervision. These predictions provide geometric cues for local anchor positioning and aggregation and are combined with RGB inputs and adapted features for Gaussian prediction.

3D Feature Fusion Given N input images $\mathcal{I}_{\text{in}} = \{I_i\}_{i=1}^{N}$, we align DA3 and multi-scale CNN features to a shared $h \times w$ grid and fuse them with RGB, GS depth, and confidence to obtain anchor features $\mathbf{z}_j$. This design is motivated by prior studies showing that self-attention and convolution capture complementary global/low-frequency and local/high-frequency information (Park & Kim, 2022; Chen et al., 2022; Peng et al., 2021), and follows the common practice in feed-forward Gaussian splatting of combining Transformer-based geometric features with CNN or RGB shortcuts to preserve fine-grained appearance details (Ye et al., 2025; Jiang et al., 2025; Wang et al., 2026a). For each anchor, confidence-guided attention aggregates pixel coordinates and GS depth within its associated image region. The predicted cameras then map the aggregated coordinates and depth to its 3D position $\mathbf{a}_j$. A Point Transformer (Zhao et al., 2021; Xu et al., 2026) enriches the features with spatial and cross-view information across all $L = Nhw$ anchors:

$$\mathbf{f}_j = \mathbf{z}_j + \mathrm{PT}\left(\{(\mathbf{a}_k, \mathbf{z}_k)\}_{k=1}^{L}\right)_j . \tag{1}$$

The resulting position–feature pairs guide subsequent scene-aware Gaussian sampling.

3.2 Scene-Aware Gaussian Sampling

To break the rigid coupling between Gaussian count and input pixels while avoiding redundant primitives across views, we propose scene-aware Gaussian sampling that jointly learns where to allocate Gaussians and how to place them in 3D under a unified, controllable budget shared across variable-view uncalibrated inputs.

Sampling Probability A probability head predicts a score $s_j = f_{\text{prob}}(\mathbf{f}_j)$ for each anchor. We normalize scores across all L anchors and mix the learned probabilities with a uniform component:

$$p_j = (1-\eta)\frac{\exp(s_j)}{\sum_{k=1}^{L}\exp(s_k)} + \frac{\eta}{L}. \tag{2}$$

Here, η controls uniform mixing, preserving sampling opportunities for low-probability anchors. The probabilities are learned through reconstruction supervision without explicit complexity labels.

Spatial Sampling For each anchor, a geometry head predicts a normalized offset mean $\boldsymbol{\delta}_j$ and standard deviation $\boldsymbol{\sigma}_j$. Given $M = \text{round}(\rho(N)L)$, where $\rho(N)$ is a view-dependent budget ratio, we select anchors with replacement and sample Gaussian centers through reparameterization:

$$\begin{aligned} &z_k \sim \text{Categorical}(p_1, \ldots, p_L), \qquad \boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \\ &\mathbf{o}_k = \boldsymbol{\delta}_{z_k} + \boldsymbol{\sigma}_{z_k} \odot \boldsymbol{\epsilon}_k, \\ &\boldsymbol{\mu}_k = \mathbf{a}_{z_k} + \ell_{z_k}\mathbf{o}_k. \end{aligned} \tag{3}$$

Here, z_k indexes the source anchor, and $\ell_j = |\overline{d}_j|/\sqrt{(f_{x,j}w)(f_{y,j}h)}$ converts normalized offsets to 3D displacements using the anchor depth $\overline{d}_j$ and normalized focal lengths $f_{x,j}, f_{y,j}$ of its source view. The expected number of samples from anchor j is $\mathbb{E}[n_j] = Mp_j$: sampling probabilities control allocation, while offset means and standard deviations control local position and dispersion.

Together, these steps define a scene-conditioned mixture distribution over Gaussian centers:

$$p(\mathbf{x} \mid \mathcal{I}_{\text{in}}) = \sum_{j=1}^{L} p_j \, \mathcal{N}\big(\mathbf{x}; \mathbf{a}_j + \ell_j\boldsymbol{\delta}_j, \ell_j^2 \,\text{diag}(\boldsymbol{\sigma}_j^2)\big)\,. \tag{4}$$

This formulation allows SampleSplat to adapt both Gaussian allocation and spatial coverage to local reconstruction demands, using a controllable budget more efficiently. As shown in Figure 1, SampleSplat achieves high rendering quality with substantially fewer Gaussians across different input-view settings, while allowing explicit control over the quality–efficiency trade-off.

Gaussian Decoding We inject each sampled offset into its source anchor feature through a lightweight residual encoder:

$$\widetilde{\mathbf{f}}_k = \mathbf{f}_{z_k} + f_{\text{offset}}(\mathbf{o}_k). \tag{5}$$

A shared Gaussian head decodes rotation, scale, opacity, and spherical harmonic coefficients from $\widetilde{\mathbf{f}}_k$, while centers are determined by sampling. The decoded Gaussians constitute the scene representation $\mathcal{G}$, from which novel views are synthesized via differentiable Gaussian rasterization.

3.3 Model Training

Reconstruction supervision is applied directly to the feed-forward output to jointly learn Gaussian attributes and sampling distributions.

Distribution Learning Spatial offsets receive gradients through reparameterization. To train the probability head despite discrete source selection, we introduce an opacity-based gradient surrogate:

$$g_k = \exp[\beta\,(\log p_{z_k} - \text{sg}(\log p_{z_k}))]\,, \qquad \widetilde{\alpha}_k = \alpha_k g_k, \tag{6}$$

where sg denotes stop-gradient. Since $g_k = 1$ in the forward pass, rendering opacity is unchanged, while the backward pass provides a learning signal to the probability head. This is a biased gradient surrogate. We also detach ℓ_j in the offset branch, while preserving gradients through anchor positions and sampled offsets.

Loss Functions We supervise input-view reconstruction and target-view synthesis with a rendering loss $\mathcal{L}_{\text{render}}$ combining pixel-wise L1 and perceptual terms, and regularize normalized inverse depth with an edge-aware smoothness loss $\mathcal{L}_{\text{smooth}}$. The overall objective is

$$\mathcal{L} = \mathcal{L}_{\text{render}}(\mathcal{S}_{\text{in}}) + \mathcal{L}_{\text{render}}(\mathcal{S}_{\text{tgt}}) + \lambda_{\text{smooth}}\mathcal{L}_{\text{smooth}}, \tag{7}$$

where $\mathcal{S}_{\text{in}}$ and $\mathcal{S}_{\text{tgt}}$ denote the input and target view sets. Detailed loss definitions and weights are provided in Appendix A.3.

Table 1: **Quantitative evaluation of sparse-view novel-view synthesis.** PSNR, SSIM, LPIPS, and DISTS are reported under different context-view settings. The best and second-best results are highlighted for each metric and setting within each pose group.

Method	2 views				6 views				8 views				Avg.			
	PSNR ↑	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**
Model-predicted Tgt. Pose																
AnySplat	20.4296	0.6081	0.2808	0.1179	17.7976	0.5177	0.3742	0.1578	18.5095	0.5438	0.3581	0.1579	18.9122	0.5565	0.3377	0.1445
SPFSplat	24.1690	0.7490	0.2800	0.1679	-	-	-	-	-	-	-	-	-	-	-	-
World-Mirror	20.2996	0.6039	0.2847	0.0904	18.0239	0.5205	0.4222	0.1783	18.2896	0.5331	0.4217	0.1863	18.8710	0.5525	0.3762	0.1517
DA3	18.9184	0.5645	0.4251	0.2072	19.6471	0.5992	0.4013	0.2025	19.8720	0.6045	0.3914	0.1955	19.4792	0.5894	0.4059	0.2018
QuerySplat	24.9523	0.7625	0.2803	0.1604	22.5033	0.6871	0.3606	0.2036	22.7817	0.6950	0.3568	0.2018	23.4124	0.7149	0.3325	0.1886
SampleSplat	25.0869	0.7813	0.2566	0.1018	22.7937	0.7100	0.3022	0.1115	23.2350	0.7216	0.2881	0.1049	23.7052	0.7376	0.2823	0.1061
GT Tgt. Pose																
NoPoSplat	26.1056	0.8284	0.1534	0.0761	-	-	-	-	-	-	-	-	-	-	-	-
YoNoSplat	21.5974	0.6319	0.3662	0.2029	19.7154	0.5683	0.4110	0.2199	19.8954	0.5738	0.4037	0.2164	20.4027	0.5913	0.3936	0.2131
SampleSplat	25.9471	0.8088	0.2507	0.1015	22.5389	0.6994	0.3071	0.1121	22.8296	0.7089	0.2930	0.1055	23.7719	0.7390	0.2836	0.1064
GT Tgt. Pose + op																
NoPoSplat	27.3379	0.8622	0.1445	0.0755	-	-	-	-	-	-	-	-	-	-	-	-
YoNoSplat	23.5682	0.7090	0.3382	0.1997	21.8336	0.6626	0.3731	0.2146	22.1791	0.6732	0.3637	0.2111	22.5270	0.6816	0.3583	0.2085
SampleSplat	26.8498	0.8350	0.2406	0.0984	24.3269	0.7753	0.2817	0.1082	24.9457	0.7905	0.2664	0.1021	25.3741	0.8003	0.2629	0.1029

Table 2: **Quantitative evaluation of dense-view novel-view synthesis.** PSNR, SSIM, LPIPS, and DISTS are reported under different context-view settings. The best and second-best results are highlighted for each metric and setting within each pose group.

Method	16 views				24 views				32 views				Avg.			
	PSNR ↑	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**
Model-predicted Tgt. Pose																
AnySplat	19.5267	0.5852	0.3424	0.1536	18.9490	0.5623	0.3720	0.1706	19.1401	0.5700	0.3683	0.1710	19.2052	0.5725	0.3609	0.1651
World-Mirror	17.8508	0.5258	0.4554	0.2184	16.8180	0.4869	0.5048	0.2528	16.6980	0.4869	0.5121	0.2613	17.1223	0.4999	0.4908	0.2442
DA3	19.9914	0.6041	0.3865	0.1898	19.5734	0.5900	0.4062	0.2004	19.5481	0.5896	0.4051	0.1976	19.7043	0.5946	0.3993	0.1959
QuerySplat	22.1057	0.6677	0.3871	0.2212	20.6022	0.6136	0.4440	0.2604	20.4524	0.6092	0.4486	0.2632	21.0534	0.6302	0.4266	0.2483
SampleSplat	23.6824	0.7414	0.2538	0.0942	22.2651	0.6918	0.2917	0.1110	22.0092	0.6839	0.2958	0.1116	22.6523	0.7057	0.2805	0.1056
GT Tgt. Pose																
YoNoSplat	19.4427	0.5564	0.4145	0.2220	18.2772	0.5189	0.4562	0.2471	18.2001	0.5171	0.4568	0.2479	18.6400	0.5308	0.4425	0.2390
SampleSplat	22.5594	0.6918	0.2730	0.0962	21.1138	0.6432	0.3120	0.1133	20.9945	0.6403	0.3124	0.1133	21.5559	0.6585	0.2991	0.1076
GT Tgt. Pose + op																
YoNoSplat	21.9148	0.6655	0.3690	0.2155	20.5551	0.6225	0.4091	0.2403	20.4984	0.6229	0.4102	0.2411	20.9894	0.6370	0.3961	0.2323
SampleSplat	25.1504	0.7958	0.2586	0.0987	23.6828	0.7571	0.2924	0.1143	23.5985	0.7572	0.2917	0.1143	24.1439	0.7700	0.2809	0.1091

3.4 Optional Test-Time Pose Alignment

To assess the effect of camera errors on view-synthesis evaluation, we optionally refine target poses after feed-forward reconstruction. All Gaussian attributes, network parameters, and camera intrinsics are fixed; only target rotations and translations are optimized. For a world-to-camera transform E_v, we apply incremental updates,$E_v^{(t+1)} = \exp(\boldsymbol{\xi}_v^{\wedge})E_v^{(t)}$, where $\boldsymbol{\xi}_v \in \mathfrak{se}(3)$. We minimize image MSE plus VGG-LPIPS weighted by 0.05 for 100 steps. This optional procedure requires target images and does not modify the reconstructed Gaussians. Results with and without alignment are reported separately.

4 Experiments

4.1 Setup

Training Settings We initialize from DA3-GIANT-1.1 (Lin et al., 2025), keeping its geometry backbone and camera head frozen. Training uses variable image resolutions, with input views sampled uniformly from $\{1, 2, 4, 6, 8\}$ and six target views per sample. We train for 60K iterations on four NVIDIA H100 80GB GPUs with a batch size of one per GPU. Detailed configurations are provided in Appendix A.1.

Datasets We train on the official training splits of DL3DV-10K (Ling et al., 2024) and a subset of UCO3D (Liu et al., 2025), sampling them at an 80% and 20% ratio. DL3DV-10K contains diverse indoor and outdoor scene videos, while UCO3D provides object-centric multi-view videos. We evaluate on the official test splits of DL3DV-10K, and RealEstate10K (Zhou et al., 2018), a real-estate video dataset.

Baseline We compare our method with representative approaches to RGB-only NVS under both multi-view and monocular input settings. **(1) For multi-view NVS,** we compare with AnyS-

plat (Jiang et al., 2025), NoPoSplat (Ye et al., 2025), YoNoSplat (Ye et al., 2026b), SPFSplat (Huang & Mikolajczyk, 2025), Depth Anything 3 (Lin et al., 2025), and WorldMirror (Liu et al., 2026). **(2) For monocular NVS,** we compare with AnySplat (Jiang et al., 2025), WorldMirror (Liu et al., 2026), SHARP (Mescheder et al., 2026), UniSHARP (Song et al., 2026), InfiniSplat (Wang et al., 2026a), and Depth Anything 3 (Lin et al., 2025).

Metrics. We assess the quality of novel-view synthesis using PSNR, SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018), and DISTS (Ding et al., 2020).

4.2 Main Results

Multi-view Novel View Synthesis We evaluate SampleSplat on the 140-scene DL3DV-10K benchmark under three target-pose protocols, excluding target views from reconstruction. Baseline configurations and evaluation protocols are detailed in Appendix B. Table 1 compares sparse-view synthesis with 2, 6, and 8 context views. With predicted poses, SampleSplat leads in PSNR, SSIM, and LPIPS throughout and in DISTS with 6 and 8 views, averaging 23.71 dB PSNR. With ground-truth poses, it leads across all metrics with 6 and 8 views, whereas NoPoSplat leads with 2. Optional alignment raises its average PSNR from 23.77 to 25.37 dB. Notably, NoPoSplat reconstructs a field of view different from that of the input and is evaluated against cropped ground truth; as shown in Figure 5, its reconstruction exhibits **a substantial loss of information from the original field of view**. Table 2 evaluates dense-view synthesis with 16, 24, and 32 context views. SampleSplat leads on every metric and view count under all three protocols. Its average PSNR exceeds the strongest baseline by 1.60 dB with predicted poses, 2.92 dB with ground-truth poses, and 3.15 dB with alignment. These results demonstrate that SampleSplat effectively uses varying amounts of contextual information for high-quality novel-view synthesis across sparse and dense inputs.

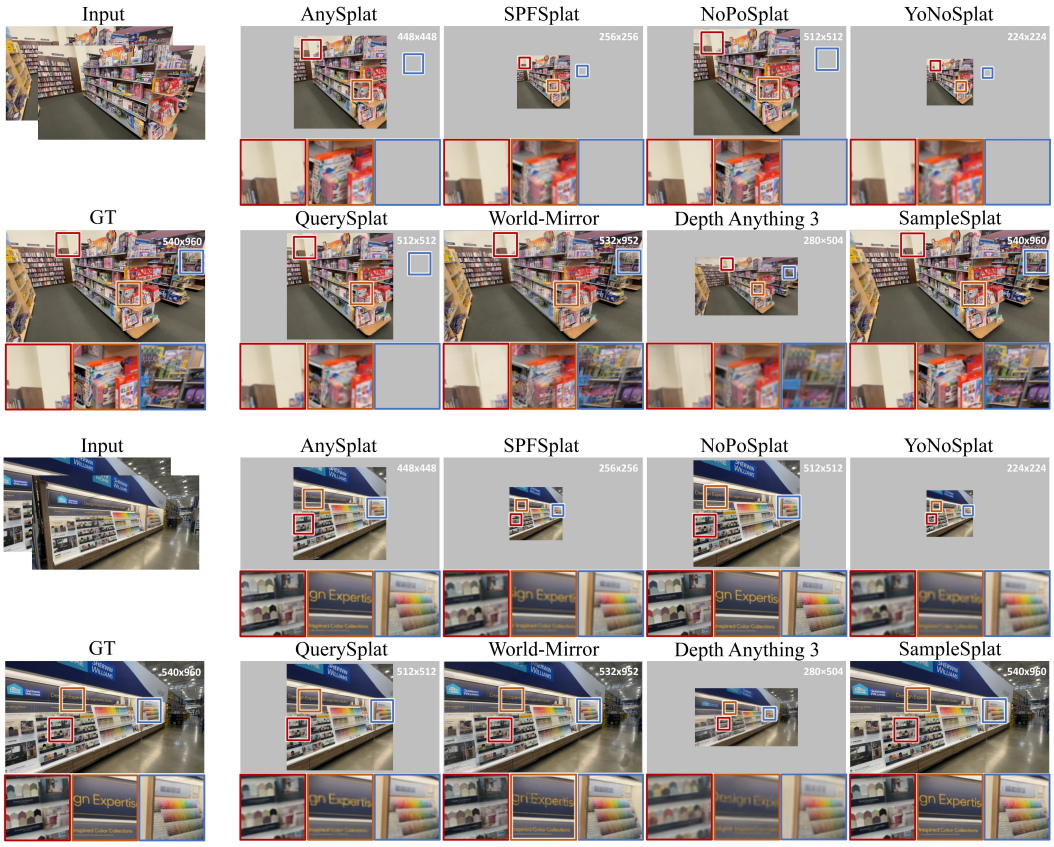


Figure 4: **Qualitative comparison.** Qualitative comparison of novel-view synthesis methods on DL3DV with two context views. SampleSplat produces renderings with finer textures and sharper object boundaries while preserving the original input resolution.

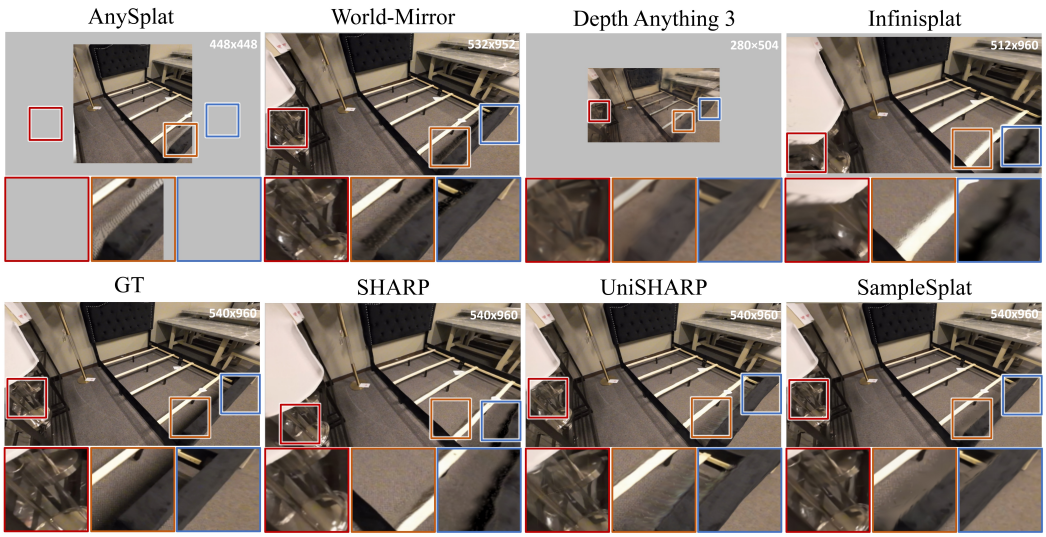


Figure 5: **Qualitative comparison.** Qualitative comparison of monocular novel-view synthesis results across different methods on DL3DV. SampleSplat produces renderings with finer textures and sharper object boundaries while preserving the original input resolution.

Monocular View Synthesis Table 3 compares monocular novel-view synthesis on DL3DV. Our method achieves the best PSNR, SSIM, and LPIPS (22.0145, 0.7039, and 0.3215), surpassing UniSHARP by 4.01 dB in PSNR while maintaining competitive DISTS (0.1263). It runs at 90.97 ms per image, substantially faster than SHARP, UniSHARP, and InfiniSplat, without requiring ground-truth camera intrinsics.

Table 3: **Quantitative evaluation of Monocular view synthesis on DL3DV.** The best and second-best results are highlighted.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	DISTS ↓
AnySplat	13.5546	0.4537	0.4776	0.1960
World-Mirror	17.2398	0.5202	0.3854	0.1090
SHARP	17.4346	0.5417	0.3937	0.1270
UniSHARP	17.9997	0.5741	0.3480	0.1003
InfiniSplat	15.4821	0.5091	0.4683	0.1678
DA3	17.0894	0.5282	0.5290	0.2612
QuerySplat	21.9168	0.6710	0.3563	0.1849
SampleSplat	22.0145	0.7039	0.3215	0.1263

Qualitative Comparison Figure 4 and Figure 5 present qualitative comparisons of novel-view synthesis under multi-view and monocular input settings, respectively. SampleSplat preserves fine textures, reduces ghosting, and maintains sharp boundaries in the multi-view setting, while producing visually coherent novel views from a single input image in the monocular setting.

Ablation Studies Table 4 shows that the full model performs best across all metrics and view settings. Removing GS Depth causes the largest PSNR drop (6.09–6.93 dB), highlighting the importance of accurate geometric anchors. Removing the Point Transformer reduces PSNR by 1.67–2.23 dB, while removing sampling or replacing it with random sampling also consistently degrades quality, supporting the effectiveness of learned scene-aware sampling. Moreover, standalone DA3 performs substantially worse than SampleSplat, supporting that our gains extend beyond the pre-trained DA3 priors and arise from our reconstruction and scene-aware sampling design.

Table 4: **Ablation study on DL3DV.** Results are reported with 2, 6, and 8 context views. PT denotes the Point Transformer.

Variant	2 Views				6 Views				8 Views			
	PSNR ↑	SSIM ↑	LPIPS ↓	DISTS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	DISTS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	DISTS ↓
DA3	18.9184	0.5645	0.4251	0.2072	19.6471	0.5992	0.4013	0.2025	19.8720	0.6045	0.3914	0.1955
w/o PT	23.7559	0.7490	0.2997	0.1316	20.9398	0.6608	0.3656	0.1541	21.3544	0.6739	0.3542	0.1491
w/o Sampling	23.6646	0.7195	0.3048	0.1284	22.2199	0.6793	0.3365	0.1404	22.6032	0.6903	0.3237	0.1345
Random Sampling	23.2904	0.7130	0.3067	0.1250	21.9627	0.6687	0.3381	0.1334	22.3403	0.6795	0.3254	0.1278
w/o GS Depth	19.3305	0.5706	0.4281	0.1731	16.4493	0.4815	0.5424	0.2634	16.6622	0.4884	0.5407	0.2774
SampleSplat	**25.0869**	**0.7813**	**0.2566**	**0.1018**	**22.7937**	**0.7100**	**0.3022**	**0.1115**	**23.2350**	**0.7216**	**0.2881**	**0.1049**

Table 5: Quality and efficiency comparison for single-image Gaussian Splatting reconstruction. The best and second-best results are highlighted.

Method	Resolution	PSNR ↑	SSIM ↑	LPIPS ↓	Time (ms) ↓	# Gaussians
AnySplat	448 × 448	13.5087	0.4330	0.4534	79.44	183,944
WorldMirror	532 × 952	17.2398	0.5202	0.3854	75.51	419,504
SHARP	540 × 960 *orig.*	17.4346	0.5417	0.3937	3203.08	1,179,648
UniSHARP	540 × 960 *orig.*	17.9997	0.5741	0.3480	238.89	1,036,800
InfiniSplat	512 × 960	15.4821	0.5091	0.4683	246.06	1,479,792
DA3	280 × 504	17.4923	0.4869	0.4221	66.10	113,758
QuerySplat	512 × 512	21.9168	0.6710	0.3563	697.13	524,288
SampleSplat	540 × 960 *orig.*	22.0145	0.7039	0.3215	90.97	64,800

4.3 Efficiency

As shown in Table 5, SampleSplat achieves the best reconstruction quality across all three metrics using only 64,800 Gaussians, the fewest among the compared methods. It processes a 540×960 image in 90.97 ms, approximately $2.6\times$ faster than UniSHARP while using $16\times$ fewer Gaussians, demonstrating a favorable balance between quality, inference time, and representation size.

4.4 Visual Analysis

Figure 6 visualizes SampleSplat's learned spatial sampling. GS Depth guides Gaussian placement, while Sampling Probability favors textures and boundaries over flat regions. The offset mean and standard deviation control sample locations and dispersion, concentrating samples in complex regions and spreading them across flat areas. GS Density follows this pattern, with fewer Gaussians in flat regions and more near boundaries and complex textures. Despite stochastic fluctuations and uniform mixing, these results demonstrate that **SampleSplat adaptively allocates Gaussians and adjusts their spatial sampling distributions according to local scene complexity**.

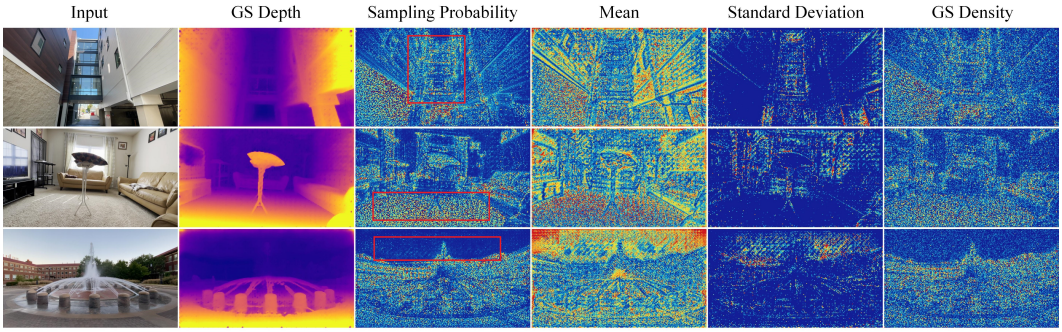


Figure 6: **Visualization of scene-aware sampling in SampleSplat.** The richly textured ground in the third row receives higher sampling probabilities and more Gaussians, while the largely uniform sky in the fourth row receives fewer, illustrating adaptive budget allocation according to local scene complexity.**This demonstrates SampleSplat's ability to adaptively allocate Gaussians according to local scene complexity.**

5 Conclusion

We presented SampleSplat, a unified feed-forward framework for uncalibrated 3D scene reconstruction from one or multiple images. Its geometry-aware scene representation supports scene-aware probabilistic sampling, which jointly learns Gaussian allocation and 3D placement under a shared budget through reconstruction supervision. One model supports monocular, sparse-view, and dense-view inputs across varying resolutions and aspect ratios, achieving state-of-the-art novel-view synthesis on DL3DV-10K with efficient inference and budget-efficient Gaussian allocation adapted to local reconstruction demands. However, SampleSplat relies on the representation quality of the pretrained 3D VFM, and its total Gaussian budget is determined by input size and predefined view-dependent ratios. The framework currently handles only static scenes and does not support 4D Gaussian Splatting.

AI USE STATEMENT

We used generative AI tools to assist with translation and grammar checking of the manuscript. All AI-assisted content was reviewed by the authors for accuracy and consistency with the intended meaning. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the model architecture, training configuration, datasets, evaluation protocols, and implementation details in the main paper and appendix. All experiments follow the official dataset splits and clearly specified evaluation metrics to facilitate consistent comparison. We will release the source code and pretrained models to further support reproducibility.

REFERENCES

David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D gaussian splats from image pairs for scalable generalizable 3D reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

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 *Computer Vision – ECCV 2024*. Springer, 2024. doi: 10.1007/978-3-031-72664-4_21.

Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022.

Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2020.

Ranran Huang and Krystian Mikolajczyk. No pose at all: Self-supervised pose-free 3D Gaussian Splatting from sparse views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://ieeexplore.ieee.org/document/11444651/.

Cho In, Jeonghwan Cho, Mijin Yoo, Gim Hee Lee, and Seon Joo Kim. Atsplat: Compact feed-forward 3d gaussian splatting with adaptive token expansion. *arXiv preprint arXiv:2607.20417*, 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 (TOG)*, 44(6):1–16, 2025.

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. doi: 10.1145/3592433.

Injae Kim, Chaehyeon Kim, Minseong Bae, Minseok Joo, and Hyunwoo J Kim. F4splat: Feed-forward predictive densification for feed-forward 3d gaussian splatting. *arXiv preprint arXiv:2603.21304*, 2026.

Vincent Leroy, Yohann Cabon, and Jerome Revaud. Grounding image matching in 3d with mast3r. In *Computer Vision – ECCV 2024*, pp. 71–91. Springer, 2024. doi: 10.1007/978-3-031-73220-1_5.

Yinglong Li, Donghui Shen, Xiaoyu Zhang, Zhichao Ye, Hongyu Wu, Aimin Hao, Guofeng Zhang, and Haomin Liu. Querysplat: Decoupling geometry and appearance representations in 3dgs prediction. *arXiv preprint arXiv:2608.01186*, 2026.

Haotong Lin, Sili Chen, Junhao Liew, Donny Y Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22160–22169. IEEE, 2024.

Xingchen Liu, Piyush Tayal, Jianyuan Wang, Jesus Zarzar, Tom Monnier, Konstantinos Tertikas, Jiali Duan, Antoine Toisoul, Jason Y Zhang, Natalia Neverova, et al. Uncommon objects in 3d. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14102–14113. IEEE, 2025.

Yifan Liu, Zhiyuan Min, Zhenwei Wang, Junta Wu, Tengfei Wang, Yixuan Yuan, Yawei Luo, and Chunchao Guo. WorldMirror: Universal 3D world reconstruction with any-prior prompting. In *International Conference on Machine Learning*, 2026.

Lars Mescheder, Wei Dong, Shiwei Li, Xuyang Bai, Marcel Santos, Peiyun Hu, Bruno Lecouat, Mingmin Zhen, Amaël Delaunoy, Tian Fang, Yanghai Tsin, Stephan R. Richter, and Vladlen Koltun. Sharp monocular view synthesis in less than a second. In *International Conference on Learning Representations*, 2026. URL `https://openreview.net/forum?id=yx3g4sF70y`.

Arthur Moreau, Richard Shaw, Michal Nazarczuk, Jisu Shin, Thomas Tanay, Zhensong Zhang, Songcen Xu, and Eduardo Pérez-Pellitero. Off the grid: Detection of primitives for feed-forward 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11756–11766, 2026.

Namuk Park and Songkuk Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022.

Zhiliang Peng, Wei Huang, Shanzhi Gu, Lingxi Xie, Yaowei Wang, Jianbin Jiao, and Qixiang Ye. Conformer: Local features coupling global representations for visual recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 367–376, 2021.

Badrinath Singhal, Sreehari Iyer, Ankit Dhiman, Venkatesh Babu Radhakrishnan, et al. Adaptivesplat: Texture aware controllable 3d gaussian allocation for feed-forward reconstruction. *arXiv preprint arXiv:2607.04256*, 2026.

Meixi Song, Dizhe Zhang, Hao Ren, Ruiyang Zhang, Bo Du, Ming-Hsuan Yang, and Lu Qi. Unisharp: Universal sharp monocular view synthesis. *arXiv preprint arXiv:2606.07514*, 2026.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025.

Jiawei Wang, Hao Yu, Yongzhen Hu, Xinyi Yang, Tao Ni, Xin Zhan, Junbo Chen, Xiaowei Zhou, Ruizhen Hu, and Sida Peng. Infinisplat: Implicit gaussian decoding for large-baseline monocular view synthesis. *arXiv preprint arXiv:2608.02437*, 2026a.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709. IEEE, 2024.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations*, volume 2026, pp. 10481–10497, 2026b.

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, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. DepthSplat: Connecting gaussian splatting and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Haofei Xu, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Resplat: Learning recurrent gaussian splatting. In *European Conference on Computer Vision*, pp. 364–382. Springer, 2026.

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 *International Conference on Learning Representations*, volume 2025, pp. 54009–54033, 2025.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. Yonosplat: You only need one model for feedforward 3d gaussian splatting. In *International Conference on Learning Representations*, volume 2026, pp. 39852–39871, 2026a.

Botao Ye, Boqi Chen, Haofei Xu, Daniel Barath, and Marc Pollefeys. YoNoSplat: You only need one model for feedforward 3D Gaussian Splatting. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=ImRhA9xmay.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16259–16268, 2021.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018.

A Implementation Details

This section complements the main method description with the training configuration, sampling schedule, and loss formulation. Evaluation protocols and additional experimental results are presented in Sections B and C, respectively.

A.1 Training Details

We jointly optimize the feature adapters, GS depth head, fusion modules, Point Transformer, sampler, and Gaussian head.

The temporal sampling window is progressively enlarged from 24–40 to 30–60 frames during the first 10K iterations. For multi-view inputs, additional input views are selected by farthest-point sampling over camera positions, and six target frames are sampled uniformly between the earliest and latest input frames. For single-view inputs, target frames are sampled symmetrically on either side at temporal distances of 1–3, 4–6, and 7–11 frames.

DL3DV samples are randomly processed at 540×960, 480×720, 480×640, or 360×640, while UCO3D samples use 960×540, 720×480, or 640×480.

A.2 Sampling Configuration

For input view counts $N \in \{1, 2, 4, 6, 8\}$, the Gaussian budget ratios $\rho(N)$ are 2.0, 1.75, 1.5, 1.33, and 1.25, respectively. We initialize offset means to zero, standard deviations to 0.1, and the final layers of the probability and offset-feature heads to zero. Sampling is uniform for the first 1000 steps, followed by a 1000-step linear transition using $\mathrm{softmax}(\gamma(t)\mathbf{s})$, with $\gamma(t)$ increasing from zero to one. We use $\eta = \beta = 0.1$, $\sigma_{\min} = 10^{-4}$, and $\sigma_{\max} = 1$. At test time, we use the learned distribution with a fixed random seed.

A.3 Training Objective

Let $\widehat{I}_v = \mathcal{R}(\mathcal{G}, \widehat{T}_v, \widehat{K}_v)$ denote the rendered image. We supervise both input-view reconstruction and target-view synthesis. For a view set $\mathcal{S}$, the rendering loss is

$$\mathcal{L}_{\mathrm{render}}(\mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{v \in \mathcal{S}} \Big[\|\widehat{I}_v - I_v\|_1 + \lambda_{\mathrm{perc}} \mathcal{L}_{\mathrm{perc}}(\mathcal{D}_{1/2}(\widehat{I}_v), \mathcal{D}_{1/2}(I_v)) \Big]. \tag{8}$$

Here, $\mathcal{D}_{1/2}$ downsamples images by a factor of two, and $\mathcal{L}_{\mathrm{perc}}$ measures weighted L1 differences in image space and frozen VGG-19 features. Half-resolution perceptual supervision reduces memory usage.

We additionally regularize normalized inverse depth, $\widetilde{d}_i = \widehat{D}_i^{-1} / \operatorname{mean}(\widehat{D}_i^{-1})$, with an edge-aware smoothness loss:

$$\mathcal{L}_{\mathrm{smooth}} = \frac{1}{N} \sum_{i=1}^{N} \left\langle |\partial_x \widetilde{d}_i| e^{-\|\partial_x I_i\|_{1,c}} + |\partial_y \widetilde{d}_i| e^{-\|\partial_y I_i\|_{1,c}} \right\rangle, \tag{9}$$

where $\langle \cdot \rangle$ denotes spatial averaging and $\| \cdot \|_{1,c}$ denotes the channel-wise mean absolute value. The total objective is given in Eq. equation 7.

We use $\lambda_{\mathrm{perc}} = 0.5$ and $\lambda_{\mathrm{smooth}} = 0.01$.

B Evaluation Protocol

We describe the baseline preprocessing and target-pose settings used for the quantitative and qualitative comparisons.

Table 6: **Cross-dataset evaluation on RealEstate10K.** Results are reported with 6 and 16 context views. Avg. denotes the arithmetic mean across the two view settings. The best and second-best results are highlighted for each metric and view count within each target-pose group.

Method	6 views				16 views				Avg.			
	PSNR ↑	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**	**PSNR ↑**	**SSIM ↑**	**LPIPS ↓**	**DISTS ↓**
Model-predicted Tgt. Pose												
AnySplat	18.9960	0.6396	0.2890	0.1223	21.5464	0.7306	0.2181	0.1019	20.2712	0.6851	0.2536	0.1121
World-Mirror	20.6803	0.7197	0.2783	0.1183	22.2109	0.7830	0.2460	0.1147	21.4456	0.7514	0.2622	0.1165
DA3	19.4242	0.6768	0.3207	0.1774	20.0889	0.6997	0.2913	0.1578	19.7566	0.6883	0.3060	0.1676
QuerySplat	23.6947	0.7755	0.2577	0.1589	25.1994	0.8162	0.2336	0.1476	24.4471	0.7959	0.2457	0.1533
SampleSplat	24.4507	0.8114	0.2678	0.1260	26.4902	0.8577	0.2217	0.1059	25.4705	0.8346	0.2448	0.1160
GT Tgt. Pose												
YoNoSplat	24.0617	0.7926	0.2203	0.1269	25.8972	0.8482	0.1843	0.1186	24.9795	0.8204	0.2023	0.1228
SampleSplat	25.0146	0.8274	0.2473	0.1187	26.5223	0.8580	0.2205	0.1053	25.7685	0.8427	0.2339	0.1120

B.1 Baseline Configuration and Image Processing

We compare with recent pose-free 3DGS methods using official pretrained weights and default inference pipelines. Baseline inputs are resized with preserved aspect ratios and cropped as required. Predictions are rescaled to the original image's shorter side and evaluated against ground-truth crops with matching fields of view. SampleSplat supports commonly used resolutions without cropping.

B.2 Target-Pose Settings

We report two groups of results according to the source of target-view poses. In the first group, each method predicts target poses without ground-truth camera poses. In the second group, rendering uses ground-truth target poses. For comparison with YoNoSplat and NoPoSplat in this group, we replace only the target poses used by SampleSplat for rendering at inference, retaining the same checkpoint without additional training or fine-tuning. Target views are excluded from scene reconstruction in both settings to prevent information leakage.

Since NoPoSplat is trained with two context views and has limited generalization to larger view counts, we report its results only in the two-view setting.

C Additional Results and Analysis

We first evaluate generalization to RealEstate10K, then present additional rendering comparisons on DL3DV and examine how the learned sampler distributes Gaussians. Together, these results connect reconstruction quality with the spatial allocation of the Gaussian budget.

C.1 Cross-Dataset Generalization

We evaluate SampleSplat on RealEstate10K (Zhou et al., 2018) with 6 and 16 context views to assess generalization beyond its training datasets. SampleSplat is trained on DL3DV-10K and UCO3D without training or fine-tuning on RealEstate10K. Table 6 reports results grouped by target-pose source. QuerySplat and both GT-pose results use test-time pose alignment; YoNoSplat is trained on RealEstate10K, whereas SampleSplat is not. Alignment uses target images as described in Section 3.4, so these results include test-time optimization.

With model-predicted target poses, SampleSplat achieves the highest PSNR and SSIM at both view counts, exceeding QuerySplat by 0.76 and 1.29 dB in PSNR with 6 and 16 views, respectively. QuerySplat achieves lower LPIPS with 6 views, while AnySplat achieves lower LPIPS with 16 views. World-Mirror and AnySplat obtain the lowest DISTS with 6 and 16 views, respectively.

With ground-truth target poses and alignment, SampleSplat surpasses YoNoSplat in PSNR, SSIM, and DISTS at both view counts, improving PSNR by 0.95 and 0.63 dB with 6 and 16 views, respectively. YoNoSplat, which is trained on RealEstate10K, achieves lower LPIPS. These results demonstrate that SampleSplat generalizes to an unseen dataset while maintaining strong reconstruction quality across view counts.

For baselines requiring cropping, evaluation uses ground-truth crops with matching fields of view, so the reported metrics do not capture information lost outside the cropped region. SampleSplat preserves the original field of view without cropping, retaining **scene information excluded from cropped evaluations**.

C.2 Qualitative Comparisons on DL3DV

Figures 7 and 9 present additional qualitative comparisons of novel-view synthesis on the DL3DV test set with two and eight context views, respectively. SampleSplat preserves fine textures, reduces ghosting, and maintains sharp object boundaries across both input settings.

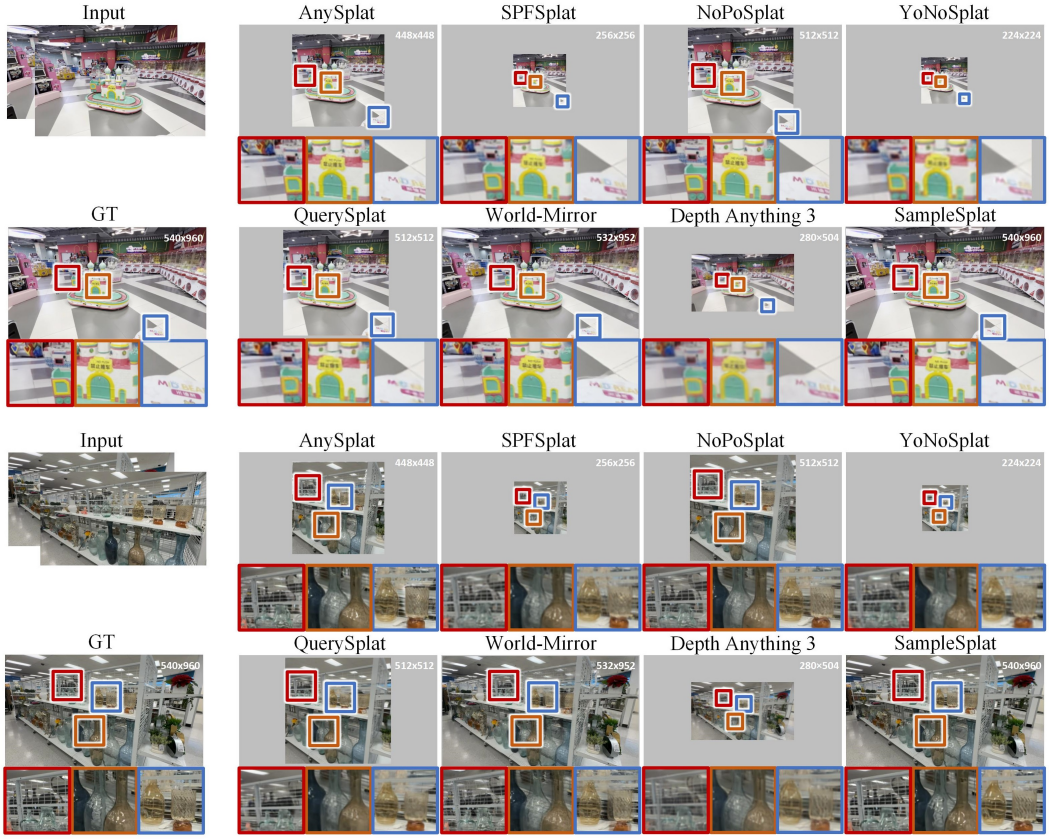


Figure 7: **Qualitative comparison with 2 context views.** Comparison of novel-view synthesis methods on the DL3DV. SampleSplat produces renderings with finer textures and sharper object boundaries while preserving the original input resolution.

C.3 Analysis of Scene-Aware Sampling

To complement the rendering comparisons, Figure 8 examines SampleSplat's learned spatial sampling. GS Depth guides Gaussian placement, while Sampling Probability favors regions with pronounced geometric or appearance variations. The offset mean and standard deviation control sample locations and dispersion, and GS Density reflects the resulting allocation.

In the second row, SampleSplat concentrates sampling around the boundaries of tables and chairs, allocating Gaussians more finely along these contours. This spatially focused allocation helps represent object boundaries where geometry and appearance change abruptly.

SampleSplat also responds to lighting variations on otherwise smooth surfaces. In the third and fourth rows, reflective regions on smooth walls attract sampling attention despite their limited geometric complexity. These examples suggest that the learned distribution responds to appearance

variations as well as geometric boundaries, directing Gaussians toward regions with changing illumination and reflections. Despite stochastic fluctuations and uniform mixing, the visualizations illustrate how **SampleSplat adapts Gaussian allocation and spatial sampling distributions to local geometric and appearance complexity**.

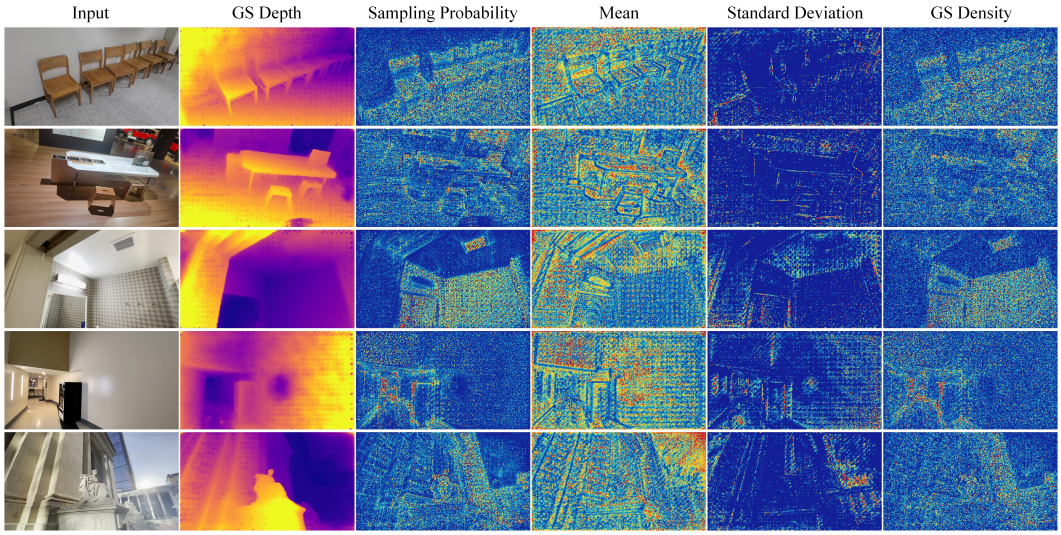


Figure 8: **Additional visualization of scene-aware sampling in SampleSplat.** GS Density illustrates **adaptive Gaussian allocation according to local scene complexity**, with more Gaussians near boundaries and complex textures and fewer in flat regions.

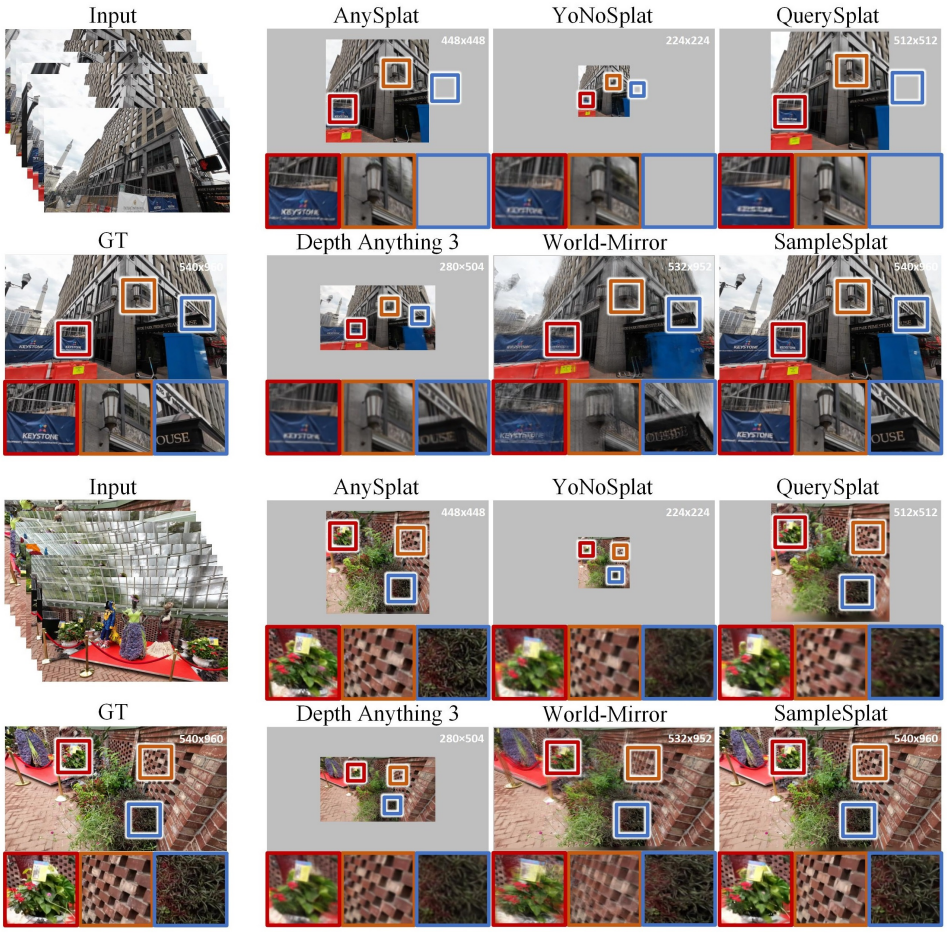


Figure 9: **Qualitative comparison with 8 context views.** Comparison of novel-view synthesis methods on the DL3DV. SampleSplat preserves fine textures, reduces ghosting, and maintains sharp object boundaries while preserving the original input resolution.