AdaptiveSplat: Texture Aware Controllable 3D Gaussian Allocation for Feed-Forward Reconstruction

Badrinath Singhal[1], Srihari K G[1], Sreehari Iyer[1], Ankit Dhiman[1,2], and Venkatesh Babu Radhakrishnan[1]

[1] Vision and AI Lab, Indian Institute of Science, Bangalore
[2] Samsung R & D Institute India - Bangalore
{badrinaths, sriharig, sreeharid, ankitd, venky}@iisc.ac.in
https://badrinaths.github.io/projects/adaptive-splat/

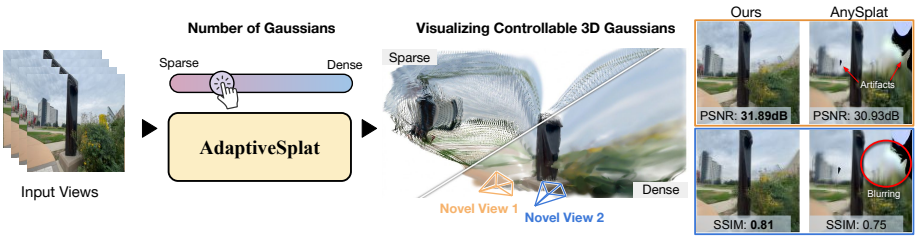

Fig. 1: Overview: We demonstrate **AdaptiveSplat**'s ability to maintain scene fidelity while controlling the allocation of primitives. Given input images and a target Gaussian budget, our feed-forward approach produces sparse yet high-quality reconstructions. In contrast, prior methods exhibit artifacts and over-smoothing at high sparsity.

Abstract. Current feed-forward 3D reconstruction methods predict pixel aligned Gaussian primitives, resulting in highly redundant representations. A natural solution is to prune the redundant Gaussians, but naive pruning introduces severe artifacts and often requires inference time fine-tuning, breaking the feed-forward paradigm. Based on previous works, high frequency regions require more Gaussian primitives, while low frequency regions can be represented with significantly fewer primitives. Motivated by this, we propose a novel approach to explicitly control the number of Gaussians by leveraging local texture information. Our approach achieves this through three key components: (1) texture estimation to capture spatial variation in scene detail, (2) texture-aware pruning that removes redundant Gaussians from low frequency regions, and (3) an adaptive Gaussian head that predicts the modified attributes of the retained primitives without breaking the feed-forward paradigm. Experiments on RE10K, ACID, DL3DV, Tanks and Temples, and DTU demonstrate the effectiveness of our approach, while ablation studies validate the contributions of its key components.

1 Introduction

Feed-forward models for 3D reconstructions [4, 7, 23, 31, 51, 53, 54, 62, 70, 75, 80] generate 3D Gaussian representations [26] directly from input views. These approaches predict pixel-aligned Gaussians, where each input pixel produces the parameters of a corresponding 3D Gaussian in the scene. While this design enables efficient feed-forward inference, it allocates Gaussians uniformly across the image without considering the underlying scene complexity. A straightforward way to reduce this redundancy is to prune the Gaussians. However, naive pruning strategies introduce severe artifacts in rendered novel views, often producing empty regions that disrupt scene continuity and degrade rendering quality.

Several works have explored compression techniques for 3D Gaussian representations [2, 14, 17, 42, 44, 45]. However, these approaches primarily focus on post-hoc quantization or encoding and typically require dense multi-view inputs along with scene-specific optimization. Such reliance on iterative optimization limits their applicability to the challenging feed-forward setting. Other works attempt to prune pixel-aligned Gaussians directly within feed-forward pipelines using geometry-driven heuristics such as overlap removal [75], opacity thresholding [80], or enforcing a single Gaussian per voxel [23]. While computationally simple, these strategies are largely agnostic to scene complexity and offer little control over the pruned Gaussians. Consequently, they may remove primitives that are necessary to represent high-frequency structures [43, 54]. These limitations raise an important question: **Can we enable controllable 3D Gaussian density in feed-forward models without sacrificing their fidelity?**

We draw inspiration from well-established statistical properties of natural images, whose power spectral density (PSD) follows an approximate power-law decay ($PSD \propto \frac{1}{f^\alpha}$, where f denotes spatial frequency and $\alpha \approx 2$) [56]. This observation implies that most visual energy is concentrated in low-frequency components, while high-frequency structures are sparse and localized. Consequently, different regions of a scene require different levels of representation density: smooth areas with low-frequency content can be reconstructed with relatively few Gaussians, whereas highly textured regions containing fine structural details require denser Gaussian allocation. Prior work [43, 54] empirically supports this hypothesis, demonstrating that high-frequency regions benefit from increased Gaussian density for accurate reconstruction. However, existing feed-forward architectures largely ignore this requirement due to their fixed pixel-aligned prediction strategy. To further illustrate this observation, we conduct a simple experiment shown in Fig. 2.

Motivated by this key insight, we propose `AdaptiveSplat`, a feed-forward framework that enables controllable 3D Gaussian allocation while preserving reconstruction fidelity. Our method focuses on capturing the variance in visual complexity. This is achieved by estimating the texture of the scene. Based on this signal, we perform texture-aware pruning that removes redundant Gaussians more aggressively in smooth regions while preserving those necessary for detailed structures. To compensate for the removed primitives, we introduce a novel adaptive Gaussian head that re-predicts the attributes of the remaining

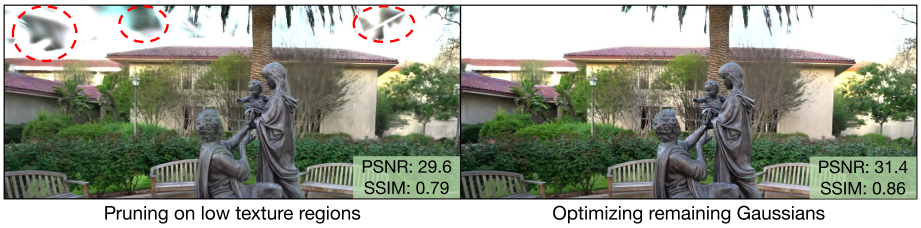

Fig. 2: Toy experiment demonstrating core insight for **AdaptiveSplat**. (Left) A rendered view shows visible artifacts, such as "black patches" that are indicative of naive pruning in low-texture regions. (Right) The same scene is rendered after refining the remaining Gaussians. The rendering quality is significantly restored as the surviving Gaussians in the low-texture areas have adaptively expanded to fill the gaps, thereby eliminating the observed artifacts.

Gaussians within the same feed-forward pipeline. This design enables flexible control over the Gaussian budget without requiring test-time optimization or post-pruning fine-tuning. Importantly, our approach can be seamlessly integrated into existing feed-forward 3D Gaussian pipelines without requiring architectural modifications. In summary, our contributions are as follows:

- *Texture-Based Scene Energy Estimation*: We introduce a texture-driven scene energy estimation mechanism that quantifies the amount of high-frequency content present in different image regions, enabling identification of areas that require higher representational capacity.
- *Texture-Aware Selective Pruning:* Building on the estimated scene energy, we propose a principled pruning strategy that selectively removes Gaussians from low-texture regions while preserving those in high-detail areas.
- *Adaptive Gaussian Head:* To mitigate artifacts introduced by pruning, we design an adaptive Gaussian head that re-predicts the attributes of the retained Gaussians, allowing the representation to adjust to the modified Gaussians.
- We perform extensive experiments for various feed-forward models on datasets like RE10K, ACID, DL3DV, Tanks and Temples, and DTU.

2 Related Work

2.1 Radiance Fields and Multi View Reconstruction

Reconstructing 3D scenes from multi-view images is a core computer vision problem. Modern radiance field methods like NeRF [9, 41] and 3D Gaussian Splatting (3DGS) [8, 26] achieve high-quality novel view synthesis, with 3DGS offering notable advantages in rendering speed through rasterization. [38] and [6] use structural anchors and contextual compression to reduce redundancy. These approaches have found success in applications such as 3D asset generation [16, 27, 32, 36, 40, 68, 76, 78], scene editing [5, 10, 15, 19, 20, 22, 52, 63, 69], and dynamic scene reconstruction [26, 35, 39, 47, 64]. However, they often require

dense multi-view images which limit adaptability and efficiency in dynamic, real-time environments. Sparse-view reconstruction methods [4, 29, 36, 46, 62, 65, 71] leverage learned priors from datasets or foundational models [49]. Recently, Feed-forward methods [25, 30, 57–59] offer a compelling alternative, directly inferring 3D representations with superior generalization across diverse scenes, bypassing the computationally expensive steps of Structure-from-Motion (SfM) [50] and radiance field optimization by directly predicting dense 3D scene representations from limited views [24, 43, 60, 67] but are rigid in number of 3D Gaussian predictions. These methods have been extended to dynamic scenes [73], multi-view consistency [3], and estimating 3DGS representations [51, 54, 55, 70, 72, 77], even from just a few views [23, 80].

2.2 Compression in 3D Gaussian Splatting

Due to its explicit nature, 3DGS often requires a large number of Gaussians to represent a scene, leading to higher storage, processing time, and slow rendering. To address these challenges, several works focus on compressing 3DGS representations. EAGLES [17] prunes based on scene-wide influence, targeting low-contribution Gaussians (e.g., small scale, low opacity, occlusion) and Light-Gaussian [13] prunes based on estimated importance (ray hits, volume), PUP-3DGS [18] uses uncertainty estimation to estimate Gaussians' importance by approximating second order gradient of loss. EfficientGS [37] estimates importance of Gaussians per pixel by selecting Top-K Gaussians to be retained, and SOG-3DGS [42] applies image compression techniques. Other methods like [2] remove Gaussians with low gradients followed by fine-tuning, while [14] proposes pruning and densification based on alpha blending importance. However, a common limitation across these compression techniques is their reliance on quantization or their inability to dynamically modify the attributes of remaining Gaussians post-pruning, which can lead to visible artifacts. Moreover, all existing methods require dense multi-view images of the scene to compress which fail in the sparse-view feed-forward setting. Interestingly, previous work like [66] has explored using wavelet based frequency decomposition to achieve high quality synthesis for multiview images but no such method exists for 3D Gaussian Splatting representation.

3 Method

We introduce a controllable feed forward 3D Gaussian generation pipeline. Fig. 3 shows the overall pipeline of our method.

Problem Formulation Existing feed-forward 3D Gaussian prediction models generate a pixel-aligned set of 3D Gaussians $\mathcal{G}_k$ for each input view. Given m views $\{(I_k, v_k)\}_{k=1}^{m}$, where $I_k \in \mathbb{R}^{H\times W\times 3}$ denotes the RGB image and v_k its associated camera pose, such models produce a set of $N = mHW$ primitives without

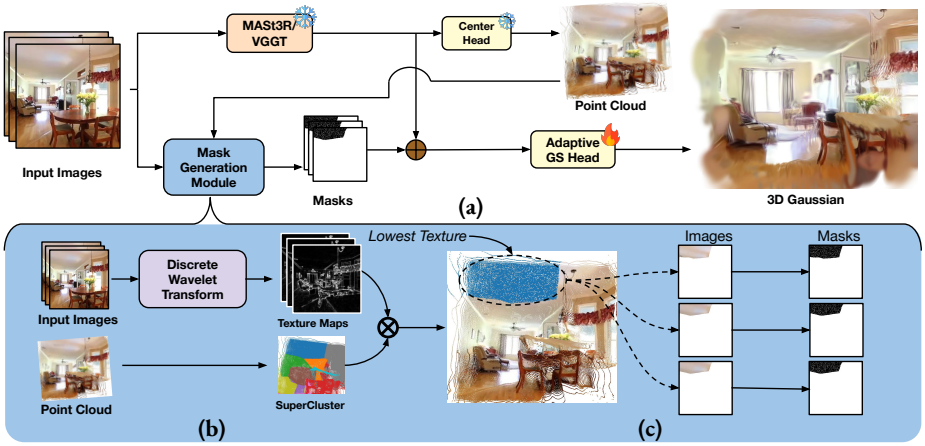

Fig. 3: Our pipeline allocates the desired number of 3D Gaussians for real-time, novel-view synthesis. (a) **Initial Representation:** We infer a 3D point cloud from context images using the MASt3R/VGGT backbone. (b) **SuperCluster Formation and Texture Analysis:** The point cloud is partitioned into SuperClusters, and texture information is simultaneously analyzed using DWT to identify low-texture regions. (c) **Gaussian prediction:** Based on the texture, masks are generated that are used in Adaptive Gaussian Head to predict optimized attributes of remaining 3D Gaussian primitives.

explicit control over representation size. In contrast, we introduce explicit control over the number of retained Gaussians. Under a user-defined pruning budget $\beta \in [0, 1)$, our objective is to predict a compact subset $\mathcal{G} = \{g_i\}_{i=1}^{B}$ where $B = \lfloor (1-\beta)N \rfloor$. Each Gaussian is parameterized as $g_i = \{\boldsymbol{\mu}_i, \boldsymbol{s}_i, \boldsymbol{q}_i, \boldsymbol{\alpha_i}, \boldsymbol{\rho}_i\}$, where $\boldsymbol{\mu}_i \in \mathbb{R}^3$ denotes the 3D mean, $\boldsymbol{s}_i \in \mathbb{R}^3$ anisotropic scale, $\boldsymbol{q}_i \in \mathbb{R}^4$ a unit quaternion encoding rotation, $\alpha_i \in [0, 1]$ opacity, and $\boldsymbol{\rho}_i \in \mathbb{R}^h$ spherical harmonics coefficients modeling view-dependent appearance.

3.1 Texture-Based Scene Energy Estimation

We estimate spatially localized scene complexity using the Discrete Wavelet Transform (DWT) applied to the input images. Given images $\{I_k\}_{k=1}^{m}$, its single-level 2D DWT is computed as $W_{ij}^{d} = \sum_u \sum_v I_k(u, v) \cdot \psi^d(i-u, j-v)$, where ψ^d denotes the directional wavelet basis. This decomposes the image into three high-frequency detail components corresponding to horizontal, vertical, and diagonal orientations. We define the local texture energy at pixel (i, j) in view k as $E_{i,j}^{(k)} = \sum_{d \in h,v,diag} |W_{ij}^{d}|$, which aggregates directional high-frequency responses.

3.2 Texture-Aware Selective Pruning

Using a pretrained multi-view backbone, we extract a dense pointmap $\mathcal{P}_k = \{p_i\}_{i=1}^{HW}$ from each input view I_k where each point $p \in \mathbb{R}^6$ encodes its 3D location and RGB color. We aggregate all views to form a global point cloud

$\mathcal{P} = \bigcup_{k=1}^{m} \mathcal{P}_k$, where $\mathcal{P} = \{p_i\}_{i=1}^{N}$. To obtain homogeneous regions, we partition $\mathcal{P}$ into K SuperClusters using K-means clustering. Clustering in (x, y, z, r, g, b) space encourages points that are spatially proximate and photometrically consistent to be grouped together. Such regions correspond to locally smooth surfaces that can be represented with fewer but spatially broader Gaussians. Formally, each SuperCluster SC_i is defined as $SC_i = \{p \in \mathcal{P} \mid |p - c_i|^2 \leq |p - c_j|^2, \forall j \neq i\}$ where the centroid is $c_i = \frac{1}{|SC_i|} \sum_{p \in SC_i} p$. The resulting clusters $\{SC_i\}_{i=1}^{K}$ form a disjoint partition of the point cloud such that $\bigcup_{i=1}^{K} SC_i = \mathcal{P}$ and $SC_i \cap SC_j = \emptyset$ for $i \neq j$.

We assign texture energy to each SuperCluster by averaging the wavelet coefficients associated with its constituent points p. Let $\pi_k(p)$ denote the projection of point p onto image I_k via pointmap $\mathcal{P}_k$. The SuperCluster energy is computed as $E_{SC_i} = \frac{1}{|SC_i|} \sum_{p \in SC_i} \bar{E}(\pi(p))$ where $\bar{E}(p)$ denotes the wavelet coefficient of the pixel corresponding to the point p. These SuperCluster-level energies provide a structured measure of regional complexity and guide subsequent Gaussian allocation under the pruning budget.

Binary Mask Construction:

$\{SC_1, \ldots, SC_K\}$ denote the regions ordered by increasing energy E_{SC_j}. Each region SC_j has a nominal retention of $\gamma|SC_j|$ Gaussians, where $\gamma = 0.1$ (empirically validated, ablation provided). We choose the largest index ℓ such that $\sum_{j=1}^{\ell} \gamma|SC_j| \leq B$. This ensures that we can effectively cover the entire low-texture region with 3D Gaussians with fewer retained Gaussians while pruning the rest.

For all regions $j < \ell$, we retain the $\gamma|SC_j|$ Gaussians. For the final region SC_ℓ, we retain only $|R_\ell| = B - \sum_{j=1}^{\ell-1} \gamma|SC_j|$. Let $\mathcal{G}_{\text{ret}}$ denote the retained set of Gaussians. The corresponding binary mask entries for each context view $i \in \{1, \ldots, m\}$ are given by $M_{i,j} = \mathbb{1}[G_j \in \mathcal{G}_{\text{ret}} \cap \mathcal{G}_i], \quad j \in \{1, \ldots, HW\}$ (see Fig. 4).

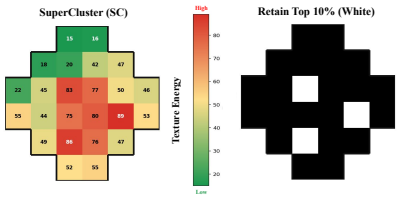

Fig. 4: Within each SuperCluster, Gaussians are ranked by texture energy, and the top 10% ($\gamma = 0.1$) are kept while the rest are masked out. This rule is applied only to low-energy clusters; high-energy clusters are fully retained.

3.3 Adaptive Gaussian Head

Given the features F and binary masks $M_1, \ldots M_m$, the Adaptive Gaussian Head f_θ predicts refined Gaussian attributes for the retained indices:

$f_\theta(F, M) \rightarrow \{\boldsymbol{s}_i, \boldsymbol{q}_i, \boldsymbol{\alpha}_i, \boldsymbol{\rho}_i\}_{i=1}^{B}$. The Adaptive Gaussian Head is built on the DPT architecture [48] with additional convolutional layers that fuse ViT [11] features extracted from the multiview feature map F with binary masks $M_1, \ldots, M_m$.

3.4 Training and Inference

Training: To ensure that the model's performance is stable across diverse budget rates β at test time, it is important that it generalizes well to a broad spectrum of pruning percentages during training. To achieve this, we train the Gaussian Head with $\beta \sim \mathcal{U}[0, 1)$. This improves robustness of the method to different budgets β, enabling adaptation of the retained 3D Gaussians.

The network is trained using photometric loss on the rendered image from the retained 3D Gaussians at the target camera pose and the corresponding ground truth image using the following loss formulation $\mathcal{L}$ as follows:

$$\mathcal{L} = \frac{1}{|v_t|} \sum_{v \in v_t} \left(\|I_v - \hat{I}_v\|_2^2 + \lambda \cdot \text{LPIPS}\left(I_v, \hat{I}_v\right) \right), \tag{1}$$

Here v_t represents the set of target camera views, I_v denotes the target ground truth image and $\hat{I}_v$ denotes the rendered image from the target view v. 3D Gaussians are obtained from Gaussian Head f_θ using context view decoder features along with its corresponding masks $M_1, M_2, \ldots M_m$. The weighting coefficient λ controls the perceptual regularization strength.

Inference: At inference, the user selects a desired sparsity level $\beta \in [0, 0.8]$. The system computes DWT from images, forms masks, and directly predicts adapted Gaussian attributes via the Gaussian Head.

4 Experiments

4.1 Experimental Setup

Implementation Details Our approach builds upon the architecture of MASt3R [30] for sparse-view inputs and VGGT [57] for dense-views with a DPT [48] Gaussian Head to predict 3D Gaussian primitives similar to [23, 70], which integrates a Gaussian prediction head into the MASt3R/VGGT pipeline. This allows for extraction of 3D Gaussian attributes for each point in the point cloud, providing an initial 3D representation. These methods don't require camera poses as input; thus, we also implement our approach on MVSplat [7] backbone which also takes camera pose as input. We choose $K = 300$ and $\gamma = 10\%$ for our pipeline (ablations provided on these hyperparameters) and λ is set to 0.001. We trained our model on 1 NVIDIA A100 40GB GPU for 48 GPU hours on each dataset. We use available pretrained checkpoints of the baseline feed-forward models for the evaluation.

Datasets We train and evaluate our method on diverse real-world datasets following the experimental setup of pixelSplat [4] and AnySplat [23]. Our primary training and evaluation data is derived from RealEstate10K [79] (indoor, multi-view videos), ACID [34] (large-scale, dynamic outdoor drone scenes) and DL3DV [33] (large scale scene) datasets. These datasets provide multi-view images with

COLMAP-computed [50] camera poses, and we adhere to their official train-test splits. To assess zero-shot generalization, we also evaluate on the DTU dataset [1, 21], comprising high-quality, object-centric scans. We further report results on the Tanks and Temples dataset [28] in the supplementary material.

Evaluation Criteria We benchmark our method on novel view synthesis using widely adopted image quality metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [61], and Learned Perceptual Image Patch Similarity (LPIPS) [74]. Beyond these measures, we explicitly compare against β used for 3D scene representation. This metric directly demonstrates our approach's efficiency in reducing redundancy while maintaining superior visual fidelity.

4.2 Comparisons with Baselines

Comparisons with existing feed-forward methods are challenging because they lack controllable Gaussian allocation and are designed for dense, per-pixel prediction, unlike our approach. To enable fair comparison, we construct baselines by applying established pruning techniques on top of state-of-the-art feed-forward methods. We consider four representative sparse-view feed-forward methods: pixelSplat [4], MVSplat [7], NoPoSplat [70], and HiSplat [54], as well as Gaussian Graph Network (GGN) [75], AnySplat [23], a recent method designed for dense-view settings. On top of the Gaussian primitives predicted by these models, we apply four representative pruning strategies: EAGLES [17], LightGaussian [13], EfficientGS [37], and PUP3DGS [18].

To compare with existing methods fairly, baselines must also operate under the feed-forward setting. However, directly applying pruning to the predicted Gaussians is suboptimal, as most pruning strategies assume per-scene optimization. While this optimization can partially recover reconstruction quality, it breaks the feed-forward assumption. To account for this trade-off, we evaluate baselines under two settings: (1) pruning followed by per-scene fine-tuning, which we refer to as with fine-tuning, and (2) direct pruning, which preserves the feed-forward property and is referred to as without fine-tuning. We evaluate our approach in both sparse-view and dense-view feed-forward settings. For sparse-view evaluation, we use two input views, following the protocol of pixelSplat [4]. For dense-view settings we evaluate using 6, 9, 16, and 32 view inputs following the protocol of AnySplat [23]. We report results for the best-performing backbone, while detailed comparisons across all backbones pixelSplat, MVSplat, HiSplat, NoPoSplat, GGN, InstantSplat [12] and AnySplat are provided in the supplementary.

4.3 Inference Time Analysis

To highlight the practical benefits of controllable pruning, we report rendering throughput under different pruning strengths. We measure rendering speed by synthesizing 1000 novel views for each pruning strength and reporting the Frames Per Second (FPS).

4.4 Ablation Study

We analyze the impact of the key design choices in our method. Each ablation isolates a specific component to evaluate its contribution to the overall performance.

• *Texture Ablation:* We analyse the importance of texture for ranking SuperClusters. In our method, texture energy determines the order in which SuperClusters are selected for pruning. In this ablation, we remove the texture-based ranking and train the model end-to-end while selecting SuperClusters randomly for pruning to evaluate the importance of texture guidance.

• *Importance of Hyperparameter K and γ:* We provide ablations on various choices of K and γ, which control the number of SuperClusters and the retention ratio within each SuperCluster.

• *Adaptive vs Fixed Gaussian Head:* Here, we replace the Adaptive Gaussian Head with a fixed, pretrained Gaussian head and directly select the retained Gaussians from its predictions without further adaptation. This allows us to evaluate the importance of learning adaptive Gaussian attributes after pruning.

• *Binary Mask:* To validate the importance of the binary mask, we train the Adaptive Gaussian Head with the mask as input and compare it with another Adaptive Gaussian head trained end-to-end without the mask.

Collectively, these ablations validate the importance of the key components of our method in achieving superior performance.

4.5 Qualitative Texture Analysis

The experiments so far focus on novel view synthesis to evaluate reconstruction quality. However, our method explicitly leverages texture signals to guide Gaussian allocation. To analyze this component more directly, we perform qualitative analysis with baselines focusing on the trends in the texture structure of the reconstructed scenes.

5 Results

5.1 Comparison with Baselines

Quantitative Analysis Tab. 1 presents a detailed comparison of AdaptiveSplat against established baselines on the ACID and RE10K datasets with finetuning for sparse view input (additional evaluations without finetuning included in supplementary) and Tab. 2 presents results on the DL3DV dataset with finetuning on dense view (additional evaluations without finetuning included in supplementary). Performance is reported across $\beta = \{0.4, 0.6, 0.8\}$. Our method consistently outperforms existing pruning strategies across all datasets and β values. On ACID, AdaptiveSplat improves PSNR from $19.43dB$ (EAGLES) to $22.55dB$ at $\beta = 0.4$, while reducing LPIPS from 0.620 to 0.299. On RE10K, it achieves 22.29 PSNR compared to 16.60 from LightGaussian and 16.53 from EAGLES. Improvements are even larger on DL3DV, for example with 6 views and $\beta = 0.4$,

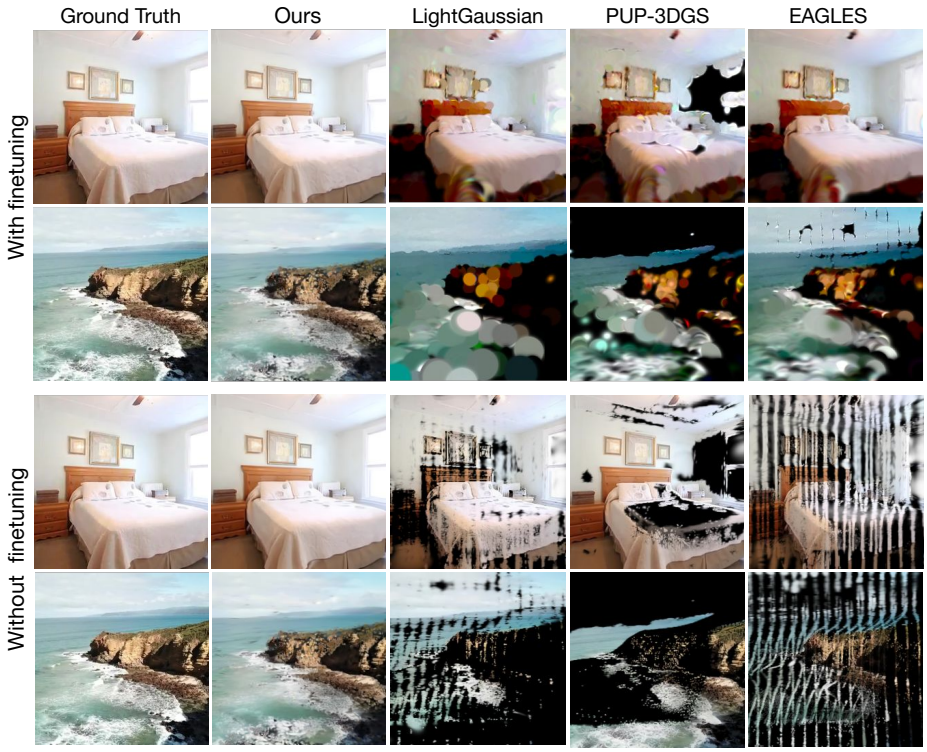

Fig. 5: Comparison of results on RE10K and ACID scenes using $\beta = 0.4$ and $\beta = 0.8$ of target budget. We allocate fewer but larger 3D Gaussians to low texture SuperClusters by the feed-forward backbone.

AdaptiveSplat reaches $20.45dB$ PSNR versus $11.36dB$ from LightGaussian. Notably, while baseline methods degrade significantly at high pruning ($\beta = 0.8$), our method maintains strong performance across all view counts.

Overall, the trend remains the same, highlighting the superiority of the proposed approach. We report only the best-performing baselines here, while the remaining baseline comparisons, wider pruning ranges, ablation studies (with and without fine-tuning across various backbones), and cross-domain evaluation on the DTU dataset are deferred to the supplementary material.

Qualitative Analysis We present a qualitative comparison of our method in Fig. 5 and Fig. 6. In all these results, we highlight two important issues with the baseline methods: (a) For non-finetuned cases, “black patch" artifacts can be consistently seen, which is indicative of empty spaces within the 3D scene where Gaussians were removed without any readjustment. Because of “black patches" artifacts, we see lower metrics across all baselines compared, whereas our results show consistent results without any such artifacts and simultaneously provide high fidelity. (b) For finetuned cases, results struggle with “blobby Gaussian”

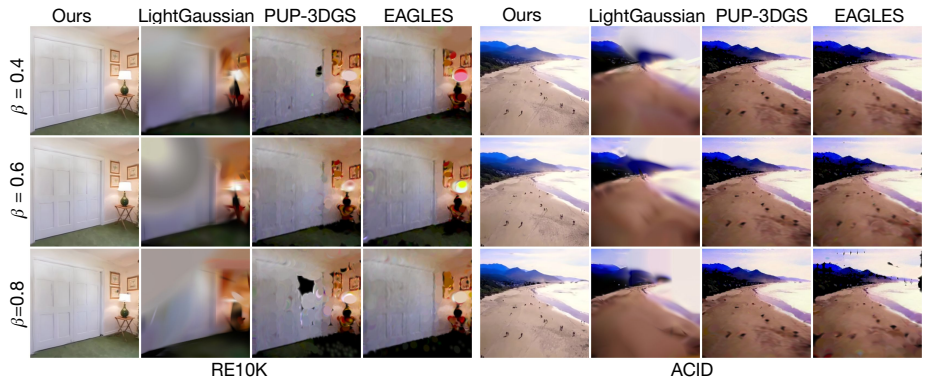

Fig. 6: Qualitative comparison of our method with baseline methods (with finetuning) on a single scene from RE10K and ACID datasets. Our method faithfully represents a scene even with $\beta = 0.8$ (80% pruning)

Table 1: Quantitative Results on ACID and RE10K Datasets. Comparison of pruning methods with NoPoSplat and HiSplat as backbones across different pruning ratios β.

			$\beta = 0.4$			$\beta = 0.6$			$\beta = 0.8$		
		Method	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**
ACID	NoPoSplat	EAGLES	11.115	0.592	0.344	10.834	0.617	0.316	10.316	0.656	0.265
		LightGauss.	10.835	0.632	0.350	10.706	0.636	0.346	10.611	0.638	0.343
		PUP-3DGS	9.672	0.640	0.266	8.616	0.672	0.203	7.722	0.705	0.137
		EfficientGS	9.171	0.665	0.193	9.290	0.661	0.196	9.303	0.663	0.195
	HiSplat	EAGLES	19.433	0.620	0.555	19.769	0.605	0.565	19.464	0.618	0.556
		LightGauss.	19.785	0.604	0.566	19.594	0.612	0.560	18.660	0.628	0.541
		PUP-3DGS	19.382	0.617	0.558	18.529	0.643	0.538	18.003	0.659	0.527
		EfficientGS	19.762	0.607	0.565	19.677	0.608	0.564	19.310	0.619	0.556
		Ours	**22.549**	**0.299**	**0.640**	**22.310**	**0.310**	**0.627**	**21.055**	**0.342**	**0.593**
RE10K	NoPoSplat	EAGLES	11.017	0.573	0.415	10.846	0.588	0.399	10.530	0.614	0.367
		LightGauss.	10.185	0.625	0.393	10.282	0.625	0.397	10.373	0.623	0.397
		PUP-3DGS	10.298	0.599	0.371	9.737	0.623	0.332	8.772	0.656	0.263
		EfficientGS	8.742	0.658	0.262	8.841	0.655	0.266	8.801	0.658	0.263
	HiSplat	EAGLES	16.526	0.598	0.556	16.404	0.606	0.552	16.243	0.614	0.547
		LightGauss.	16.598	0.596	0.558	16.421	0.604	0.552	16.056	0.618	0.542
		PUP-3DGS	16.472	0.601	0.558	16.083	0.618	0.543	15.588	0.638	0.529
		EfficientGS	16.525	0.598	0.556	16.404	0.606	0.551	16.242	0.613	0.546
		Ours	**22.294**	**0.235**	**0.735**	**22.110**	**0.243**	**0.726**	**20.740**	**0.272**	**0.692**

artifact. This happens as existing pruning methods completely remove all Gaussians from a local region without replacing them. To compensate for the empty space, Gaussians from neighboring areas must expand, which distorts their shape and negatively impacts the quality of the overall representation. This effect persists even after further finetuning, as the fundamental problem of sub-optimal Gaussian allocation remains.

Table 2: Multiview Results on DL3DV Dataset. Comparison of pruning methods using the AnySplat base model across varying view counts and pruning ratios β.

		$\beta = 0.4$			$\beta = 0.6$			$\beta = 0.8$		
	Method	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**
6 Views	EAGLES	10.464	0.675	0.241	9.206	0.688	0.193	7.693	0.703	0.129
	LightGauss.	11.361	0.671	0.262	10.295	0.683	0.223	9.169	0.697	0.177
	PUP-3DGS	10.746	0.672	0.249	9.695	0.683	0.206	8.424	0.695	0.157
	Ours	**20.448**	**0.334**	**0.601**	**20.307**	**0.366**	**0.582**	**20.049**	**0.408**	**0.556**
9 Views	EAGLES	11.384	0.662	0.289	9.813	0.678	0.234	8.147	0.695	0.159
	LightGauss.	13.252	0.652	0.325	12.272	0.665	0.294	10.874	0.682	0.244
	PUP-3DGS	12.236	0.654	0.305	10.753	0.672	0.260	9.178	0.686	0.198
	Ours	**19.678**	**0.354**	**0.569**	**19.582**	**0.381**	**0.555**	**19.437**	**0.413**	**0.536**
16 Views	EAGLES	6.460	0.683	0.074	5.773	0.711	0.031	9.002	0.692	0.206
	LightGauss.	14.580	0.646	0.366	13.806	0.661	0.343	12.375	0.678	0.295
	PUP-3DGS	13.847	0.655	0.353	12.399	0.664	0.314	10.493	0.682	0.252
	Ours	**19.125**	**0.392**	**0.524**	**19.082**	**0.410**	**0.512**	**18.990**	**0.435**	**0.498**
32 Views	EAGLES	13.944	0.665	0.373	12.828	0.672	0.346	10.817	0.691	0.276
	LightGauss.	14.362	0.663	0.375	13.835	0.669	0.361	12.531	0.683	0.322
	PUP-3DGS	14.335	0.661	0.376	13.630	0.668	0.355	11.938	0.685	0.312
	Ours	**17.677**	**0.446**	**0.476**	**17.653**	**0.461**	**0.467**	**17.607**	**0.481**	**0.456**

We also show effects of increasing β in Fig. 6, where we pick three broad pruning rates ($\beta = \{0.4, 0.6, 0.8\}$) and compare AdaptiveSplat with the baselines. As can be seen, baselines degrade the scene quality and completely fail to adapt existing Gaussian attributes at higher β value. In contrast, AdaptiveSplat optimally re-adjusts 3D Gaussians to adapt to higher pruning rates, consistently maintaining the high scene quality. We show more qualitative results in the supplementary.

5.2 Ablation Study

Tab. 3 evaluates the contribution of key components of our method at a fixed pruning budget $\beta = 0.4$. Our analysis reveals that removing the Adaptive GS Head training significantly degrades performance, reducing PSNR from 22.55 dB to 17.44 dB on ACID and from 22.29 dB to 19.81 dB on RE10K, which indicates that learning to adapt Gaussian attributes post-pruning is critical for recovering reconstruction quality. Similarly, excluding the DWT-based texture signal leads to noticeable performance decreases specifically from 22.55 dB to 21.31 dB on ACID confirming the importance of texture guidance in ranking SuperClusters and allocating Gaussians effectively. Finally, removing the contextual mask further degrades RE10K performance to 20.06 dB, highlighting its vital role in providing spatial context for the Adaptive Gaussian Head. The impact of hyperparameters K and γ is evaluated in Tab. 4, where increasing K generally improves reconstruction quality by producing finer SuperClusters that better identify low-texture regions and enable more precise allocation of larger Gaussians. However, these quality gains become marginal beyond $K = 300$ while

Table 3: Ablation study. We evaluate the impact of training the GS Head, texture features, and the contextual mask at a fixed budget of $\beta = 0.4$.

	ACID			RE10K		
Ablation Variant	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**PSNR↑**	**LPIPS↓**	**SSIM↑**
w/o Texture	21.31	0.326	0.623	20.73	0.260	0.713
w/o Adaptive GS Head	17.44	0.445	0.490	19.81	0.399	0.617
w/o Binary Mask	22.52	0.318	**0.642**	20.06	0.339	0.656
Ours	**22.55**	**0.299**	0.639	**22.29**	**0.235**	**0.735**

training costs continue to rise. Regarding the retention ratio, we observe that retaining $\gamma = 0.10$ of Gaussians achieves the best performance, particularly under high pruning scenarios such as $\beta = 0.9$. Larger γ values tend to introduce redundant Gaussians in the same regions, which ultimately degrades reconstruction quality, consistent with observations in prior works [13, 14, 17]. Overall, these results demonstrate that each individual component contributes significantly to the final performance, with their combination yielding the highest reconstruction quality across both the ACID and RE10K datasets.

Table 4: Effect of the number of clusters k (left) and retention percentage γ (right) on reconstruction quality across scenes.

A. Varying number of clusters k				**B. Varying retention percentage γ**			
Config	**PSNR↑**	**LPIPS↓**	**SSIM↑**	**Config**	**PSNR↑**	**LPIPS↓**	**SSIM↑**
$k = 50$	22.416	0.303	0.636	$\gamma = 0.05$	21.796	0.347	0.608
$k = 100$	22.488	0.301	0.638	$\gamma = 0.10$	**22.549**	**0.299**	**0.640**
$k = 300$	22.549	0.299	0.640	$\gamma = 0.15$	21.587	0.352	0.589
$k = 400$	**22.559**	**0.298**	**0.641**	$\gamma = 0.20$	22.262	0.304	0.629

5.3 Qualitative Texture Analysis

We provide additional analysis focusing on scene texture preservation. Our method explicitly targets low-texture regions for pruning, allowing the model to allocate fewer but larger Gaussians in such areas while preserving fine details in highly textured regions. Fig. 7 presents a qualitative comparison of texture preservation on a representative scene across different pruning ratios. We observe that baseline methods tend to oversmooth textured regions under pruning, leading to noticeable loss of high-frequency details. In comparison, our approach better maintains scene fidelity by leveraging texture-aware clustering and natural scene statistics. Furthermore, Fig. 8 provides additional qualitative comparisons across multiple scenes at $\beta = 0.4$, demonstrating that while baselines suffer from

blurred edges, over-smoothing, and structural distortion, our method maintains sharp high-frequency details and structural integrity.

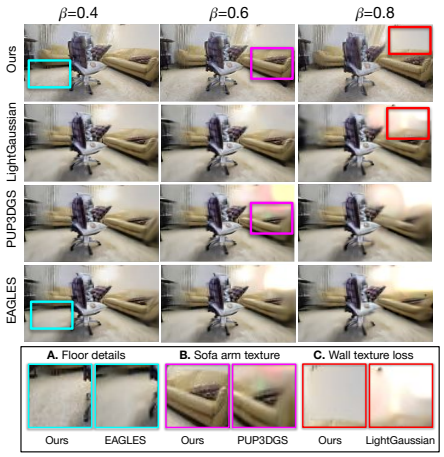

Fig. 7: Texture Analysis (fixed scene): Baselines smooth out the textures whereas we retain texture of the scene across the regions.

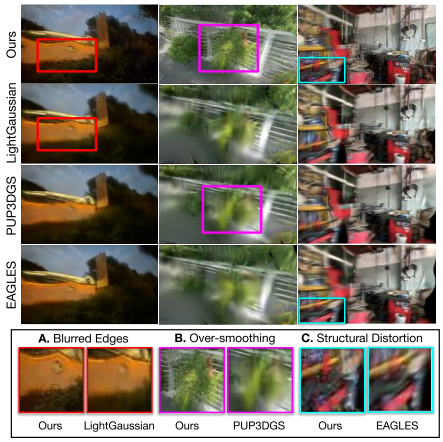

Fig. 8: Texture Analysis (varying scene): Across the scenes we observe oversmoothed regions whereas we retain the texture of the scene.

These results consistently demonstrate that competing methods introduce excessive smoothing in high-texture areas, whereas our method preserves structural details more effectively under increasing pruning strengths. Fig. 9 shows the histogram of mean scale values demonstrating our insight empirically that mean scale values of 3D Gaussians corresponding to low texture regions are predicted larger in our method to retain the quality of scene; thus, redundant 3D Gaussians can be pruned.

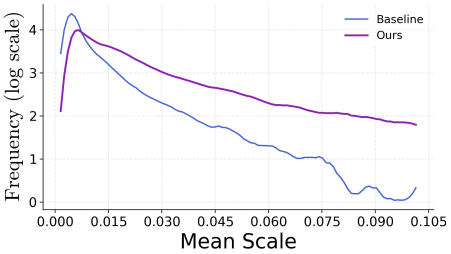

Fig. 9: Plot of mean scale values with log_{10} frequency: Predicted 3D Gaussians from our method are larger than traditional 3D Gaussian representation.

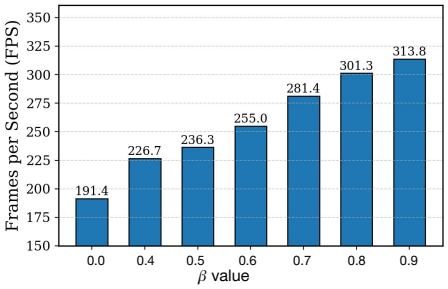

Fig. 10: Plot of rendering speed in Frame Per Second (FPS) with increasing pruning strength.

5.4 Inference Time Analysis

Fig. 10 summarizes the FPS achieved across pruning strengths. As pruning strength increases, the reduced number of active Gaussians leads to faster rendering speeds and lower computational overhead, demonstrating the effectiveness of pruning for balancing reconstruction quality and runtime efficiency. We use the same GPU memory load as the base model [23, 70]. Fig. 11 presents a Pareto plot comparing inference time and quality of ours with other methods. Our method occupies the top-left region, achieving the highest PSNR at substantially lower latency than the finetuned baselines. Further, we include additional discussions and comparisons on inference latency analysis of ours with baselines in the supplementary.

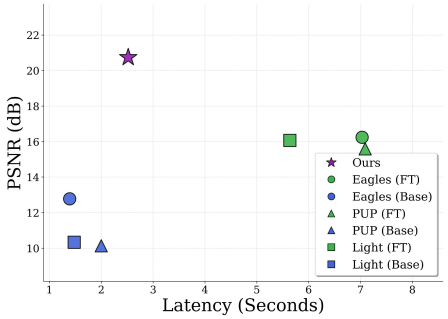

Fig. 11: Pareto Plot for inference time and quality comparison for RE10K at $\beta = 0.8$

6 Conclusions and Future Work

We introduce **AdaptiveSplat**, a novel, feed-forward framework for controllable 3D Gaussian pruning. Our method's core novelty is texture-aware Gaussian control that significantly reduces primitives by targeting low-texture regions. By eliminating the need for expensive per-scene optimization, AdaptiveSplat provides a practical solution for controllable 3D reconstruction, achieving a superior quality-efficiency trade-off even at extreme pruning rates. On smooth surfaces with high-frequency texture (such as intricate wall patterns) or specular highlights, our method suffers from the same limitation as any pixel-aligned feed-forward model. We briefly discuss these limitations in the supplementary. Future work will focus on integrating a more nuanced understanding of both texture and geometry to enable more robust pruning decisions, paving the way for truly efficient 3D scene representation.

Acknowledgement

We sincerely thank Anudeep for careful review of our manuscript and for providing valuable feedback and suggestions that significantly improved the quality of the article. This work is supported by Google and Kotak IISc AI-ML Centre (KIAC). Badrinath Singhal is supported by TCS Research Scholarship and Qualcomm Innovation Fellowship (QIF).

References

1. Aanæs, H., Jensen, R.R., Vogiatzis, G., Tola, E., Dahl, A.B.: Large-scale data for multiple-view stereopsis. International Journal of Computer Vision pp. 1–16 (2016)
2. Ali, M.S., Qamar, M., Bae, S.H., Tartaglione, E.: Trimming the fat: Efficient compression of 3d gaussian splats through pruning. arXiv preprint arXiv:2406.18214 (2024)
3. Asim, M., Wewer, C., Wimmer, T., Schiele, B., Lenssen, J.E.: Met3r: Measuring multi-view consistency in generated images (2024)
4. Charatan, D., Li, S.L., Tagliasacchi, A., Sitzmann, V.: 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. pp. 19457–19467 (2024)
5. Chen, M., Laina, I., Vedaldi, A.: Dge: Direct gaussian 3d editing by consistent multi-view editing. arXiv preprint arXiv:2404.18929 (2024)
6. Chen, Y., Wu, Q., Lin, W., Harandi, M., Cai, J.: Hac: Hash-grid assisted context for 3d gaussian splatting compression (2024), https://arxiv.org/abs/2403.14530
7. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. arXiv preprint arXiv:2403.14627 (2024)
8. Dhiman, A., Lu, T., Srinath, R., Arslan, E., Xing, A., Xiangli, Y., Radhakrishnan, V.B., Sridhar, S.: Turbo-gs: Accelerating 3d gaussian fitting for high-resolution radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15454–15464 (2026)
9. Dhiman, A., Srinath, R., Rangwani, H., Parihar, R., Boregowda, L.R., Sridhar, S., Babu, R.V.: Strata-nerf : Neural radiance fields for stratified scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 17603–17614 (October 2023)
10. Dong, J., Wang, Y.X.: 3DGS-drag: Dragging gaussians for intuitive point-based 3d editing. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=7JUrBLDjCq
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
12. Fan, Z., Cong, W., Wen, K., Wang, K., Zhang, J., Ding, X., Xu, D., Ivanovic, B., Pavone, M., Pavlakos, G., et al.: Instantsplat: Sparse-view gaussian splatting in seconds. arXiv preprint arXiv:2403.20309 (2024)
13. Fan, Z., Wang, K., Wen, K., Zhu, Z., Xu, D., Wang, Z.: Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps (2023)
14. Fang, G., Wang, B.: Mini-splatting: Representing scenes with a constrained number of gaussians. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 165–181. Springer (2024)
15. Fang, J., Wang, J., Zhang, X., Xie, L., Tian, Q.: Gaussianeditor: Editing 3d gaussians delicately with text instructions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
16. Gao, R., Holynski, A., Henzler, P., Brussee, A., Martin-Brualla, R., Srinivasan, P., Barron, J.T., Poole, B.: Cat3d: Create anything in 3d with multi-view diffusion models. Advances in Neural Information Processing Systems (2024)

17. Girish, S., Gupta, K., Shrivastava, A.: Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 54–71. Springer (2024)
18. Hanson, A., Tu, A., Singla, V., Jayawardhana, M., Zwicker, M., Goldstein, T.: Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5949–5958 (June 2025), https://pup3dgs.github.io/
19. Haque, A., Tancik, M., Efros, A., Holynski, A., Kanazawa, A.: Instruct-nerf2nerf: Editing 3d scenes with instructions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2023)
20. Hyung, J., Hwang, S., Kim, D., Lee, H., Choo, J.: Local 3d editing via 3d distillation of clip knowledge. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12674–12684 (2023)
21. Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., Aanaes, H.: Large scale multi-view stereopsis evaluation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2014)
22. Jiakai, Z., Xinhang, L., Xinyi, Y., Fuqiang, Z., Yanshun, Z., Minye, W., Yingliang, Z., Lan, X., Jingyi, Y.: Editable free-viewpoint video using a layered neural representation. In: ACM SIGGRAPH (2021)
23. Jiang, L., Mao, Y., Xu, L., Lu, T., Ren, K., Jin, Y., Xu, X., Yu, M., Pang, J., Zhao, F., et al.: Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. ACM Transactions on Graphics (TOG) **44**(6), 1–16 (2025)
24. Keetha, N., Karhade, J., Jatavallabhula, K.M., Yang, G., Scherer, S., Ramanan, D., Luiten, J.: Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2024)
25. Keetha, N., Müller, N., Schönberger, J., Porzi, L., Zhang, Y., Fischer, T., Knapitsch, A., Zauss, D., Weber, E., Antunes, N., Luiten, J., Lopez-Antequera, M., Bulò, S.R., Richardt, C., Ramanan, D., Scherer, S., Kontschieder, P.: MapAnything: Universal feed-forward metric 3D reconstruction. In: International Conference on 3D Vision (3DV). IEEE (2026)
26. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/
27. Kim, J., Koo, J., Yeo, K., Sung, M.: Synctweedies: A general generative framework based on synchronized diffusions. arXiv:2403.14370 (2024)
28. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics **36**(4) (2017)
29. Koo, J., Park, C., Sung, M.: Posterior distillation sampling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
30. Leroy, V., Cabon, Y., Revaud, J.: Grounding image matching in 3d with mast3r. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 71–91. Springer (2024)
31. Li, X., Wang, T., Gu, Z., Zhang, S., Guo, C., Cao, L.: Flashworld: High-quality 3d scene generation within seconds (2025)
32. Lin, C., Pan, P., Yang, B., Li, Z., Mu, Y.: Diffsplat: Repurposing image diffusion models for scalable 3d gaussian splat generation. In: International Conference on Learning Representations (ICLR) (2025)
33. Ling, L., Sheng, Y., Tu, Z., Zhao, W., Xin, C., Wan, K., Yu, L., Guo, Q., Yu, Z., Lu, Y., et al.: Dl3dv-10k: A large-scale scene dataset for deep learning-based

3d vision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22160–22169 (2024)
34. Liu, A., Tucker, R., Jampani, V., Makadia, A., Snavely, N., Kanazawa, A.: Infinite nature: Perpetual view generation of natural scenes from a single image (2021), https://arxiv.org/abs/2012.09855
35. LIU, Q., Liu, Y., Wang, J., Lyu, X., Wang, P., Wang, W., Hou, J.: MoDGS: Dynamic gaussian splatting from casually-captured monocular videos. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=2prShxdLkX
36. Liu, R., Wu, R., Hoorick, B.V., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object (2023)
37. Liu, W., Guan, T., Zhu, B., Xu, L., Song, Z., Li, D., Wang, Y., Yang, W.: Efficientgs: Streamlining gaussian splatting for large-scale high-resolution scene representation. IEEE MultiMedia **32**(1), 61–71 (2025). https://doi.org/10.1109/MMUL.2025.3543224
38. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., Dai, B.: Scaffold-gs: Structured 3d gaussians for view-adaptive rendering (2023), https://arxiv.org/abs/2312.00109
39. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In: 3DV (2024)
40. Meng, X., Wang, C., Lei, J., Daniilidis, K., Gu, J., Liu, L.: Zero-1-to-g: Taming pretrained 2d diffusion model for direct 3d generation. arXiv preprint arXiv:2501.05427 (2024)
41. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: Proceedings of the European Conference on Computer Vision (ECCV) (2020)
42. Morgenstern, W., Barthel, F., Hilsmann, A., Eisert, P.: Compact 3d scene representation via self-organizing gaussian grids. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 18–34. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-73013-9_2, https://fraunhoferhhi.github.io/Self-Organizing-Gaussians/
43. Nam, S., Sun, X., Kang, G., Lee, Y., Oh, S., Park, E.: Generative densification: Learning to densify gaussians for high-fidelity generalizable 3d reconstruction. arXiv preprint arXiv:2412.06234 (2024)
44. Navaneet, K., Meibodi, K.P., Koohpayegani, S.A., Pirsiavash, H.: Compact3d: Smaller and faster gaussian splatting with vector quantization. arXiv preprint arXiv:2311.18159 (2023)
45. Niedermayr, S., Stumpfegger, J., Westermann, R.: Compressed 3d gaussian splatting for accelerated novel view synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10349–10358 (June 2024)
46. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv (2022)
47. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-NeRF: Neural Radiance Fields for Dynamic Scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2020)
48. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 12179–12188 (2021)

49. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)
50. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2016)
51. Smart, B., Zheng, C., Laina, I., Prisacariu, V.A.: Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs (2024), https://arxiv.org/abs/2408.13912
52. Srinivasan, P.P., Deng, B., Zhang, X., Tancik, M., Mildenhall, B., Barron, J.T.: Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
53. Szymanowicz, S., Rupprecht, C., Vedaldi, A.: Splatter image: Ultra-fast single-view 3d reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
54. Tang, S., Ye, W., Ye, P., Lin, W., Zhou, Y., Chen, T., Ouyang, W.: Hisplat: Hierarchical 3d gaussian splatting for generalizable sparse-view reconstruction. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=SBzIbJojs8
55. Tang, Y., Guo, Y., Li, D., Peng, C.: Spars3r: Semantic prior alignment and regularization for sparse 3d reconstruction. arXiv preprint arXiv:2411.12592 (2024)
56. Torralba, A., Oliva, A.: Statistics of natural image categories. In: LBMV07 Presentations. Carnegie Mellon (2007)
57. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: Vggt: Visual geometry grounded transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2025)
58. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: Dust3r: Geometric 3d vision made easy. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20697–20709 (June 2024)
59. Wang, Y., Zhou, J., Zhu, H., Chang, W., Zhou, Y., Li, Z., Chen, J., Pang, J., Shen, C., He, T.: π^3: Permutation-equivariant visual geometry learning. arXiv preprint arXiv:2507.13347 (2025)
60. Wang, Y., Wu, Q., Xu, D.: F3d-gaus: Feed-forward 3d-aware generation on imagenet with cycle-aggregative gaussian splatting. arXiv preprint arXiv:2501.06714 (2025)
61. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)
62. Wewer, C., Raj, K., Ilg, E., Schiele, B., Lenssen, J.E.: latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. In: arXiv (2024)
63. Wu, J., Bian, J.W., Li, X., Wang, G., Reid, I., Torr, P., Prisacariu, V.: GaussCtrl: Multi-View Consistent Text-Driven 3D Gaussian Splatting Editing. Proceedings of the European Conference on Computer Vision (ECCV) (2024)
64. Wu, R., Gao, R., Poole, B., Trevithick, A., Zheng, C., Barron, J.T., Holynski, A.: CAT4D: Create Anything in 4D with Multi-View Video Diffusion Models. arXiv:2411.18613 (2024)
65. Wu, R., Mildenhall, B., Henzler, P., Park, K., Gao, R., Watson, D., Srinivasan, P.P., Verbin, D., Barron, J.T., Poole, B., et al.: Reconfusion: 3d reconstruction with diffusion priors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21551–21561 (2024)

66. Xu, M., Zhan, F., Zhang, J., Yu, Y., Zhang, X., Theobalt, C., Shao, L., Lu, S.: Wavenerf: Wavelet-based generalizable neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18195–18204 (2023)
67. Xu, Y., Shi, Z., Yifan, W., Peng, S., Yang, C., Shen, Y., Gordon, W.: Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. arxiv: 2403.14621 (2024)
68. Yang, C., Li, S., Fang, J., Liang, R., Xie, L., Zhang, X., Shen, W., Tian, Q.: Gaussianobject: High-quality 3d object reconstruction from four views with gaussian splatting. ACM Transactions on Graphics **43**(6) (December 2024)
69. Yao, Y., Zeng, Z., Gu, C., Zhu, X., Zhang, L.: Reflective gaussian splatting. arXiv preprint (2024)
70. Ye, B., Liu, S., Xu, H., Xueting, L., Pollefeys, M., Yang, M.H., Songyou, P.: No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. arXiv preprint arXiv:2410.24207 (2024)
71. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
72. Yu, J.J., Aumentado-Armstrong, T., Forghani, F., Derpanis, K.G., Brubaker, M.A.: Polyoculus: Simultaneous multi-view image-based novel view synthesis. In: Proceedings of the European Conference on Computer Vision (ECCV) (2024)
73. Zhang, J., Herrmann, C., Hur, J., Jampani, V., Darrell, T., Cole, F., Sun, D., Yang, M.H.: Monst3r: A simple approach for estimating geometry in the presence of motion. arXiv preprint arxiv:2410.03825 (2024)
74. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
75. Zhang, S., Fei, X., Liu, F., Song, H., Duan, Y.: Gaussian graph network: Learning efficient and generalizable gaussian representations from multi-view images. Advances in Neural Information Processing Systems **37**, 50361–50380 (2024)
76. Zhou, J., Zhang, W., Liu, Y.S.: Diffgs: Functional gaussian splatting diffusion. In: Advances in Neural Information Processing Systems (NeurIPS) (2024)
77. Zhou, S., Chang, H., Jiang, S., Fan, Z., Zhu, Z., Xu, D., Chari, P., You, S., Wang, Z., Kadambi, A.: Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21676–21685 (2024)
78. Zhou, S., Fan, Z., Xu, D., Chang, H., Chari, P., Bharadwaj, T., You, S., Wang, Z., Kadambi, A.: Dreamscene360: Unconstrained text-to-3d scene generation with panoramic gaussian splatting. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 324–342. Springer (2024)
79. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. In: SIGGRAPH (2018)
80. Ziwen, C., Tan, H., Zhang, K., Bi, S., Luan, F., Hong, Y., Fuxin, L., Xu, Z.: Longlrm: Long-sequence large reconstruction model for wide-coverage gaussian splats. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2025)