GEOGSPLAT: GEOMETRY-GROUNDED NON-PIXEL-ALIGNED GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review



Figure 1: **GeoGSplat** introduces a geometry-grounded non-pixel-aligned Gaussian Splatting paradigm. It achieves high-quality 3D reconstruction and novel view synthesis by incorporating reliable geometric priors through complementary Anchor and Triplane representations. Larger bubble size indicates a larger number of Gaussians.

ABSTRACT

Feed-forward Gaussian Splatting enables efficient novel view synthesis, but pixel-aligned prediction makes the primitive count grow with image resolution and view count, leading to substantial redundancy and increased computational cost. Non-pixel-aligned methods address this limitation by predicting a fixed number of Gaussians. Without explicit ray and depth constraints, these methods yield misplaced Gaussian centers, causing rendering artifacts under large viewpoint changes. To address this issue, we introduce **GeoGSplat**, a geometry-grounded feed-forward model that explicitly incorporates geometric priors into the non-pixel-aligned framework. A dual-branch encoder constructs *Anchors* and *Triplane* to capture coarse 3D structure and fine-grained scene features, respectively. Subsequently, a hierarchical decoder progressively decodes dense 3D points and Gaussian primitives, enabling 3D reconstruction and novel view synthesis. GeoGSplat surpasses both state-of-the-art pixel-aligned and non-pixel-aligned Gaussian Splatting methods on DTU and NRGBD in terms of novel view synthesis and reconstruction. The advantage of our geometry-grounded paradigm further increases as target viewpoints move farther from the input cameras, demonstrating stronger generalization to large viewpoint changes.

1 INTRODUCTION

Feed-forward Gaussian Splatting (Charatan et al., 2024; Chen et al., 2024; Zhang et al., 2024; Ren et al., 2026) aims to directly predict Gaussian primitives from a set of images, supporting novel view synthesis (NVS) and avoiding expensive per-scene optimization (Kerbl et al., 2023). Most existing methods (Charatan et al., 2024; Chen et al., 2024) conduct pixel-aligned predictions, where each input pixel is associated with a predicted Gaussian primitive. Such methods inherit explicit geometric constraints from per-pixel depth prediction, which restricts Gaussian centers to a geometrically meaningful 3D surface and thus reduces the ambiguity of appearance learning during rasterization. However, this paradigm couples Gaussian primitives to both image resolution and the number of input views, resulting in considerable computational overhead. As illustrated in Fig. 2, the number of predicted primitives grows rapidly with the number of input views. Moreover, overlapping regions across images produce multiple Gaussians for the same scene content, introducing

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.

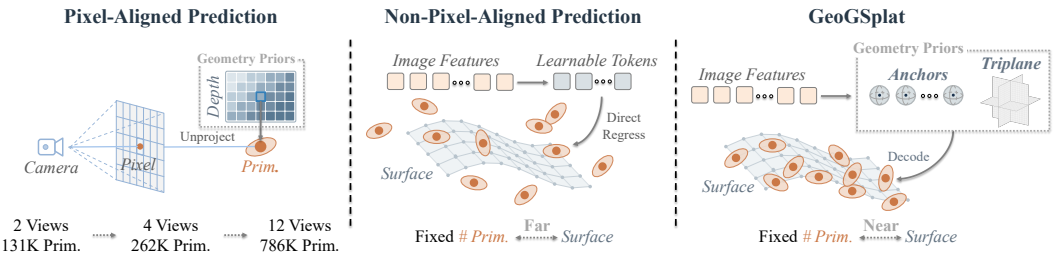


Figure 2: **Comparison of pixel-aligned, non-pixel-aligned, and our GeoGSplat paradigms**. Pixel-aligned methods rely on explicit geometry priors but produce an increasing number of primitives (Prim.) with more input views. Non-pixel-aligned methods predict a fixed number of primitives but lack geometric priors. GeoGSplat combines fixed-size prediction with explicit geometric priors.

substantial redundancy. Some recent methods (Ren et al., 2026; Li et al., 2026a;b), adopt a non-pixel-aligned paradigm that predicts a fixed number of Gaussians. These methods typically decode Gaussians from a set of learnable tokens that aggregate multi-view image features through attention. While this formulation removes the scaling dependency of pixel-aligned prediction, it also discards the geometric constraints provided by per-pixel depth. Primitives are learned from photometric supervision, leaving the Gaussian centers underconstrained and resulting in inaccurate geometry and overfitting to the observed viewpoints. Such Gaussians may produce plausible renderings near the input cameras, but they tend to yield strong rendering artifacts under larger viewpoint changes.

To address this limitation, we introduce *GeoGSplat*, a geometry-grounded non-pixel-aligned framework consisting of a *dual-branch structured encoder* and a *geometry-guided hierarchical decoder*. The encoder captures complementary scene properties through *Anchors* and *Triplane*. Anchors comprise sparse 3D spheres with learnable centers and adaptive radii, providing coarse geometric constraints on the scene structure. Triplane comprises three orthogonal 2D feature maps that encode fine-grained high-dimensional scene features queryable at arbitrary 3D coordinates. Building on the two complementary representations, the decoder progressively converts sparse geometric priors into dense 3D points and Gaussian primitives. We first perform *Anchor-to-Point* decoding by sampling Triplane features within each Anchor region to predict a set of dense 3D points. We then apply *Point-to-Gaussian* decoding, re-querying the Triplane at the decoded 3D points to predict corresponding Gaussian primitives. This hierarchical design establishes interaction between Anchors and Triplane, using Anchor geometry to constrain point reconstruction and employing fine-grained Triplane features to enable geometric refinement and appearance learning.

We conduct experiments on DTU (Jensen et al., 2014) and NRGBD (Azinović et al., 2022) for object-level and scene-level evaluations, respectively. Our GeoGSplat outperforms both non-pixel-aligned and pixel-aligned competitors on these datasets in terms of NVS and 3D reconstruction. Furthermore, we evaluate GeoGSplat under a challenging setting by progressively moving the target cameras farther from the input cameras. Its performance advantage becomes increasingly pronounced under larger viewpoint changes, showing stronger robustness and generalization under large viewpoint changes enabled by our geometric priors. Our contributions are summarized below.

- We introduce a **geometry-grounded non-pixel-aligned paradigm** for feed-forward Gaussian Splatting that incorporates explicit geometric priors, enabling high-quality NVS and 3D reconstruction in an efficient manner.
- We propose a **dual-branch structured encoder** that constructs Anchors and Triplane to capture complementary coarse 3D structure and fine-grained high-dimensional scene features, respectively.
- We develop a **geometry-guided hierarchical decoder** that progressively decodes dense 3D points and Gaussian primitives by querying Triplane features under Anchor-based geometric constraints.

2 Related Works

Pixel-aligned feed-forward Gaussian Splatting. Significant progress has been made in feed-forward Gaussian Splatting (Charatan et al., 2024; Chen et al., 2024; Zhang et al., 2024; Ren et al., 2026; Szymanowicz et al., 2024), predicting Gaussian primitives from images without per-scene optimization. Most existing approaches follow a pixel-aligned mechanism (Chen et al., 2024; Xu et al., 2025; Charatan et al., 2024), predicting a Gaussian primitive for each image pixel. Such predictions benefit from explicit geometric constraints imposed by the corresponding camera rays and per-pixel depths, which restrict Gaussian centers to plausible 3D locations. However, the pixel-aligned pipeline ties the number of Gaussians to the input image resolution and the number of views, leading to considerable redundancy and computational cost.

Non-pixel-aligned models. Non-pixel-aligned methods remove correspondences between image pixels and Gaussian primitives, instead predicting Gaussians from aggregated image features or scene representations (Wang et al., 2026; Zhang et al., 2026; Li et al., 2026a; Ren et al., 2026; Zhao et al., 2026; Moreau et al., 2026; Li et al., 2026b). However, removing pixel-to-Gaussian correspondences also eliminates the geometric constraints provided by per-pixel ray and depth. In this context, Gaussian centers tend to be inaccurately placed, leading to suboptimal solutions that overfit the observed viewpoints. While these primitives may still produce plausible renderings for novel views close to the input cameras, their geometric errors become increasingly exposed under large viewpoint changes, resulting in strong rendering artifacts. To tackle this issue, we introduce a new non-pixel-aligned paradigm that injects explicit geometric priors, enabling more robust NVS.

Scene Representations. Existing feed-forward methods adopt various scene representations (Yao et al., 2018; Gu et al., 2020; Yu et al., 2021; Wang et al., 2021; Chen et al., 2021; Jiang et al., 2025a; Johari et al., 2022; Wang et al., 2024; Leroy et al., 2024; Wang et al., 2025; Xiang et al., 2026). These representations play a critical role in Gaussian prediction. For instance, voxels and anchors constrain the spatial locations at which primitives are predicted (Lu et al., 2024; Fridovich-Keil et al., 2022); Triplane (Hong et al., 2024; Zhang et al., 2024; Chan et al., 2022) provides structured representations, where fine-grained high-dimensional scene features can be encoded and queried. Motivated by the complementarity of these two representations, GeoGSplat combines Anchors and Triplane, using Anchors to provide coarse geometric constraints and Triplane to supply fine-grained scene features.

3 Method

3.1 Preliminaries

Given N_I posed images $\{(\mathbf{I}_i, \boldsymbol{\pi}_i)\}_{i=1}^{N_I}$, feed-forward Gaussian Splatting predicts a set of 3D Gaussian primitives in a single pass. Here, $\mathbf{I}_i \in \mathbb{R}^{H\times W\times 3}$ is an RGB image and $\boldsymbol{\pi}_i$ denotes its camera parameters. The predicted Gaussian set $\mathcal{G}$ is defined as:

$$\mathcal{G} = \{\mathbf{g}_j\}_{j=1}^{N}, \qquad \mathbf{g}_j = (\boldsymbol{\mu}_j, \alpha_j, \mathbf{s}_j, \mathbf{q}_j, \mathbf{c}_j), \tag{1}$$

where N denotes the number of predicted Gaussians, $\boldsymbol{\mu}_j$, α_j, $\mathbf{s}_j$, $\mathbf{q}_j$, and $\mathbf{c}_j$ represent the center, opacity, scale, rotation, and spherical-harmonic coefficients of $\mathbf{g}_j$ (Kerbl et al., 2023).

Pixel-aligned prediction. For each pixel $\mathbf{u} = (x, y)$ in image $\mathbf{I}_i$, pixel-aligned methods (Chen et al., 2024) predict a depth z and obtain its Gaussian center $\boldsymbol{\mu}$ by camera unprojection $\boldsymbol{\mu} = \mathrm{Unproj}(\mathbf{u}, z, \boldsymbol{\pi}_i)$, where $\mathrm{Unproj}(\cdot)$ indicates the 2D to 3D unprojection. Since each pixel is mapped to one Gaussian, the total number of Gaussians is $N = N_I \times H \times W$ and therefore grows with both the input resolution and the number of views.

Non-pixel-aligned prediction. Non-pixel-aligned methods (Ren et al., 2026; Li et al., 2026b) aggregate the input images into multi-view features $\mathbf{F}_{\mathrm{img}}$ and use N_Q learnable tokens $\mathbf{Q}$ to directly decode Gaussians as $\mathcal{G} = f(\mathbf{F}_{\mathrm{img}}, \mathbf{Q})$, where $N = N_Q \times N_{\mathcal{G}/Q}$ and $N_{\mathcal{G}/Q}$ denotes the number of Gaussians predicted per token. This fixed-count formulation enables efficient prediction, but discards the geometric priors that have proven effective in pixel-aligned models.

To retain a fixed Gaussian count while keeping explicit geometric priors, we introduce GeoGSplat, a geometry-grounded non-pixel-aligned prediction paradigm. As shown in Fig. 3, GeoGSplat first

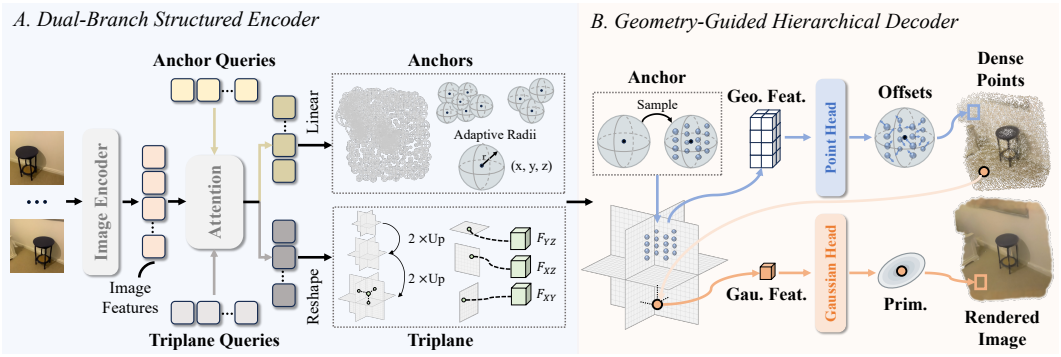


Figure 3: **Pipeline of GeoGSplat.** Image features are encoded in parallel into *Anchors* and *Triplane*. Anchors comprise sparse 3D spheres with learnable centers and adaptive radii, accounting for the coarse scene structure; Triplane consists of three orthogonal 2D feature maps that encode fine-grained high-dimensional scene features. *Anchor-to-Point Decoding* (blue) gathers Triplane features within each Anchor region to predict offsets and recover dense 3D points; *Point-to-Gaussian Decoding* (orange) re-queries the Triplane at these points to predict the corresponding Gaussian primitives.

constructs M Anchors $\mathcal{A}$ and a Triplane $\mathcal{T}$ from $\mathbf{F}_{\text{img}}$ through a dual-branch structured encoder. Subsequently, a geometry-guided hierarchical decoder is proposed to expand each Anchor into K dense 3D points and decode the corresponding K Gaussians. As M and K are fixed, $N = M \times K$ remains independent of (N_I, H, W).

3.2 Dual-Branch Structured Encoder

Image Feature Extraction. For each view, RGB and their corresponding Plücker ray (Plucker, 1865) embeddings are summed and processed by an image encoder, implemented as a standard Vision Transformer (Dosovitskiy et al., 2021):

$$\mathbf{F}_{\text{img}} = \text{Encoder}\Big([\text{PE}_{\text{rgb}}(\mathbf{I}_i) + \text{PE}_{\text{ray}}(\text{Ray}(\boldsymbol{\pi}_i))]_{i=1}^{N_I}\Big), \quad \mathbf{F}_{\text{img}} \in \mathbb{R}^{N_I n_p \times C}. \tag{2}$$

where n_p is the number of patches per view, $\text{PE}_{\text{rgb}}(\cdot)$ and $\text{PE}_{\text{ray}}(\cdot)$ indicate standard ViT patch embedding modules applied to RGB patches and Plücker rays, respectively.

Dual-Branch Feature Aggregation. To extract features for Anchor and Triplane construction, we introduce two sets of learnable queries, denoted as $\mathbf{Q}^A \in \mathbb{R}^{M \times C}$ and $\mathbf{Q}^T \in \mathbb{R}^{n_T \times C}$, respectively, where $n_T = 3S_0^2$ and S_0 is the initial plane resolution. For the Triplane branch, we additionally introduce spatial positional embeddings $\mathbf{P}^T \in \mathbb{R}^{n_T \times C}$ to encode the 2D grid coordinates and identities of the three orthogonal planes. The two sets of queries aggregate the image features through a shared attention module $\text{Att}(\cdot)$ (Carion et al., 2020) as

$$\mathbf{F}^A = \text{Att}(\mathbf{Q}^A, \mathbf{F}_{\text{img}}), \quad \mathbf{F}^A \in \mathbb{R}^{M \times C}, \quad \mathbf{F}^T = \text{Att}(\mathbf{Q}^T + \mathbf{P}^T, \mathbf{F}_{\text{img}}), \quad \mathbf{F}^T \in \mathbb{R}^{n_T \times C}. \tag{3}$$

The resulting $\mathbf{F}^A$ and $\mathbf{F}^T$ are subsequently used to construct Anchors and Triplane.

Anchor Construction. To provide coarse geometric constraints on the scene structure, we construct M Anchors $\mathcal{A} = \{(\mathbf{p}_i, r_i)\}_{i=1}^M$ from $\mathbf{F}^A$. Each Anchor is represented by a 3D sphere with center $\mathbf{p}_i$ and adaptive radius r_i. Specifically, we initialize the Anchor centers from a set of reference points $\mathcal{Z} = \{\mathbf{z}_i\}_{i=1}^M$ with broad spatial coverage, then refine them according to $\mathbf{F}^A$. The reference points $\mathcal{Z}$ are randomly sampled once from the normalized 3D volume and kept fixed throughout training. We predict an offset from $\mathbf{F}_i^A$ with a linear projection $Linear(\cdot)$. The offset is then applied to the corresponding reference point $\mathbf{z}_i$ to obtain the center $\mathbf{p}_i$:

$$\mathbf{p}_i = \tanh\big(\mathbf{z}_i + \text{Linear}(\mathbf{F}_i^A)\big), \qquad \mathbf{p}_i \in (-1, 1)^3, \tag{4}$$

where $\tanh(\cdot)$ keeps the predicted centers within the normalized scene volume. Given the refined Anchor centers, we further determine their spatial extents using adaptive radii to accommodate the non-uniform spatial distribution of 3D scene content. We initialize the radius r_i using the median

distance from $\mathbf{p}_i$ to the four nearest Anchor centers. The radius is then adjusted with a bounded factor predicted from $\mathbf{F}_i^A$ as:

$$r_i = d_i(\gamma_{\min} + (\gamma_{\max} - \gamma_{\min})\,\sigma(\text{Linear}(\mathbf{F}_i^A))), \qquad r_i \in \mathbb{R}_+, \tag{5}$$

where d_i defines the median distance, $(\gamma_{\min}, \gamma_{\max})$ bound the learned multiplier, and $\sigma(\cdot)$ is the sigmoid function. d_i adapts the Anchor extent to the spatial distribution of scene geometry, while the learned factor makes it responsive to image content. Together, these Anchors form a coarse 3D representation of the scene observed in the input images.

Triplane Construction. To complement the coarse structure with fine-grained features, we build Triplane from $\mathbf{F}^T$ through progressive refinement. First, we reshape $\mathbf{F}^T \in \mathbb{R}^{3S_0^2 \times C}$ to an initial Triplane $\mathbf{T}_{S_0} \in \mathbb{R}^{3\times S_0 \times S_0 \times C'}$. Two refinement stages are then applied to increase the plane resolution from S_0 to $S = 4S_0$. Each refinement comprises bilinear upsampling by a factor of two $\text{Up}_2(\cdot)$ and a Triplane refinement block $\mathcal{R}_s(\cdot,\cdot)$:

$$\mathbf{T}_{2s} = \mathcal{R}_s(\text{Up}_2(\mathbf{T}_s), \mathbf{F}_{\text{img}}), \qquad \mathbf{T}_{2s} \in \mathbb{R}^{3\times 2s\times 2s\times C'}, \tag{6}$$

where s and C' denote the plane resolution and feature dimension, respectively. $\mathcal{R}_s(\cdot,\cdot)$ combines residual convolution (He et al., 2016), cross-plane feature fusion (Wang et al., 2023), and image-conditioned cross-attention. The final Triplane is defined as $\mathcal{T} = \{\mathbf{T}^{xy}, \mathbf{T}^{xz}, \mathbf{T}^{yz}\}, \mathbf{T} \in \mathbb{R}^{S\times S\times C_T}$. Given 3D coordinates $\mathbf{x} \in [-1, 1]^3$, we project them onto each plane and sample features at the corresponding 2D locations. The Triplane features $\phi(\mathbf{x})$ are then obtained via concatenation. Such a query process is defined as

$$\phi_a(\mathbf{x}) = \text{Bilinear}(\mathbf{T}^a, \text{Proj}(\mathbf{x})), \quad \phi(\mathbf{x}) = [\phi_{xy}(\mathbf{x}); \phi_{xz}(\mathbf{x}); \phi_{yz}(\mathbf{x})] \in \mathbb{R}^{3C_T}. \tag{7}$$

where $a \in \{xy, xz, yz\}$ identifies a plane, $\text{Proj}(\cdot)$ maps the 3D coordinates to 2D coordinates on a plane, and $\text{Bilinear}(\cdot,\cdot)$ samples the corresponding features by bilinear interpolation. The queried features are utilized in the following decoder for reconstruction and Gaussian primitive prediction.

3.3 Geometry-Guided Hierarchical Decoder

To enforce geometric priors throughout decoding, we propose a hierarchical decoder with two stages: Anchor-to-Point decoding reconstructs dense 3D geometry, and Point-to-Gaussian decoding builds Gaussian primitives upon the recovered geometry for appearance modeling.

Anchor-to-Point Decoding. To decode dense points within each Anchor, we initialize reference points and then refine them using scene features from the Triplane. Similar to Anchor construction, these points are randomly initialized once from the normalized 3D volume and kept fixed throughout training. For each Anchor, the reference points $\bar{\mathbf{X}}_i$ are established within its local region defined by r_i. We then query the Triplane to obtain geometry features as $\mathbf{F}_i^{\text{geo}} = \phi(\bar{\mathbf{X}}_i) \in \mathbb{R}^{K\times 3C_T}$, where $\phi(\cdot)$ denotes the Triplane sampling function in Eq. (7). To refine $\bar{\mathbf{X}}_i$ based on $\mathbf{F}_i^{\text{geo}}$, we introduce K learnable tokens $\mathbf{Q}^{\text{off}}$ and apply cross-attention to predict coordinate offsets:

$$\Delta\mathbf{X}_i = \text{CrossAtt}\left(\mathbf{Q}_i^{\text{off}}, \mathbf{F}_i^{\text{geo}}, \mathbf{F}_i^{\text{geo}}\right). \tag{8}$$

We scale the offsets by r_i and obtain K points $\mathbf{X}_i$ as $\mathbf{X}_i = \bar{\mathbf{X}}_i + r_i\Delta\mathbf{X}_i$, where $\mathbf{X}_i \in \mathbb{R}^{K\times 3}$. The final dense points are defined as $\mathcal{P} = [\mathbf{X}_1; \ldots; \mathbf{X}_M] \in \mathbb{R}^{N\times 3}$ with $N = M \times K$.

Point-to-Gaussian Decoding. To further decode $\mathcal{G}$ from the dense points $\mathcal{P}$, we start by querying $\mathcal{T}$ at each refined point $\mathbf{x}_j \in \mathcal{P}$. A shared MLP is then employed to produce the Gaussian features as $\mathbf{z}_j = \text{MLP}([\mathbf{x}_j; \phi(\mathbf{x}_j)])$. The corresponding $\mathbf{g}_j$ is predicted from $\mathbf{z}_j$ through a Gaussian head as

$$(\Delta\boldsymbol{\mu}_j, \alpha_j, \mathbf{s}_j, \mathbf{q}_j, \mathbf{c}_j) = \text{Head}_G(\mathbf{z}_j). \tag{9}$$

The final Gaussian center is computed as $\boldsymbol{\mu}_j = \mathbf{x}_j + \Delta\boldsymbol{\mu}_j$.

3.4 Training Loss

The geometric priors provided by the aforementioned Anchors and dense points allow us to enhance the supervision by utilizing ground-truth geometry. Therefore, we employ a geometry loss defined as the symmetric Chamfer Distance $\text{CD}(\cdot,\cdot)$:

$$\mathcal{L}_{\text{geo}} = \text{CD}(\mathcal{P}_A, \mathcal{P}_{\text{gt}}^A) + \text{CD}(\mathcal{P}, \mathcal{P}_{\text{gt}}), \tag{10}$$

where $\mathcal{P}_A$ denotes the Anchor centers, $\mathcal{P}$ indicates the decoded dense 3D points, and $(\mathcal{P}_{\text{gt}}^A, \mathcal{P}_{\text{gt}})$ are the ground-truth counterparts. We unproject and fuse the valid depth points from the input views. We then apply FPS (Qi et al., 2017) to obtain $\mathcal{P}_{\text{gt}} \in \mathbb{R}^{N\times 3}$ and $\mathcal{P}_{\text{gt}}^A \in \mathbb{R}^{M\times 3}$, matching the predicted points $\mathcal{P}$ and Anchor centers $\mathcal{P}_A$, respectively. This geometry supervision enables reliable scene-wide Anchor placement and local point refinement.

Furthermore, we render the predicted Gaussians at the input cameras and compute the standard photometric loss (Wang et al., 2004; Zhang et al., 2018) as

$$\mathcal{L}_{\text{render}} = \mathcal{L}_{\text{rgb}} + \lambda_s \mathcal{L}_{\text{SSIM}} + \lambda_p \mathcal{L}_{\text{LPIPS}}. \tag{11}$$

The complete training loss is defined as $\mathcal{L} = \mathcal{L}_{\text{geo}} + \mathcal{L}_{\text{render}}$. In practice, we first train with geometric supervision alone for 15K iterations, and then add the rendering loss. This allows the model to establish stable geometry before jointly optimizing geometry and appearance.

4 Experiments

4.1 Experimental Setup

Implementation details. The training resolution is set to 256×256. The image encoder contains 8 ViT blocks, and the shared attention module contains 12 attention blocks. We use $M = 4096$ Anchor tokens and predict $K = 16$ dense points per Anchor, yielding a fixed set of $65{,}536$ Gaussians. Triplane is progressively upsampled from resolution 64 to 128 and then 256. We train for 320K iterations using AdamW (Loshchilov & Hutter, 2017), with a peak learning rate of 4×10^{-4} and BF16 precision. Training is performed on 128 NVIDIA L20 GPUs with a global batch size of 128. During training, the network takes 4 input images, and the renderings are produced at the input cameras, without sampling additional target views.

Datasets. We train GeoGSplat on five public datasets: ScanNet++ (Yeshwanth et al., 2023), ARKitScenes (Dehghan et al., 2021), BlendedMVS (Yao et al., 2020), CO3D (Reizenstein et al., 2021), and WildRGB-D (Xia et al., 2024). We perform evaluations on DTU (Jensen et al., 2014) and NRGBD (Azinović et al., 2022) for object- and scene-level reconstruction, respectively. For novel view synthesis, each evaluation sample contains 4 novel views. We use 4 context views by default and also report results with 2 and 12 input views in the Appendix A.2.

Evaluation protocol. We evaluate NVS using PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018). The input and target resolutions are set to 256×256 for all methods except AnySplat, whose input images are resized to 448×448 to match its input resolution requirement. 3D reconstruction is evaluated using Chamfer distances (Tenenbaum et al., 1977) from predicted points to ground truth (Pred→GT) and from ground truth to predicted points (GT→Pred), as well as their average, denoted as CD. For GeoGSplat, these metrics are computed on the predicted dense points, while for the other methods, they are computed on the predicted Gaussian centers. Notably, the results of GeoGSplat computed on dense points and Gaussian centers are nearly identical, as reported in Appendix A.4. For a fair comparison, we define ground-truth point clouds in a normalized coordinate system. We then transform predicted point clouds from the camera coordinate system to the same coordinate system for CD computation. All evaluated methods use ground-truth camera poses for the input images, except AnySplat, which uses predicted poses by default. Notably, we also report results of AnySplat with ground-truth poses, denoted as AnySplat*.

4.2 Novel View Synthesis and Geometric Reconstruction

Tables 1 and 2 report NVS and reconstruction results with 4 input views, and Fig. 4 provides qualitative comparisons. We compare our GeoGSplat against pixel-aligned methods MVSplat (Chen et al., 2024), DepthSplat (Xu et al., 2025), and AnySplat (Jiang et al., 2025b). Although AnySplat aggregates Gaussians through voxelization, it relies on per-pixel depth prediction, and the Gaussian count is still coupled to both image resolution and the number of input views. We also compare against non-pixel-aligned methods TokenGS (Ren et al., 2026) and VolSplat (Wang et al., 2026).

Novel view synthesis. Under the same Gaussian budget, GeoGSplat substantially improves NVS quality over TokenGS on both datasets. On DTU, GeoGSplat improves PSNR from 17.71 to 20.48 dB and reduces LPIPS from 0.354 to 0.137. On NRGBD, PSNR improves from 23.82 to

Table 1: **Novel view synthesis with four input views.** NUM denotes the number of Gaussians used for rendering. The best PSNR, SSIM, and LPIPS are shown in bold.

Method	DTU				NRGBD			
	PSNR↑	SSIM↑	LPIPS↓	NUM	PSNR↑	SSIM↑	LPIPS↓	NUM
MVSplat	14.44	0.520	0.401	158K	18.53	0.609	0.319	255K
DepthSplat	15.46	0.599	0.331	158K	22.98	0.792	0.170	255K
AnySplat	16.60	0.630	0.166	170K	20.12	0.690	0.138	643K
AnySplat*	17.54	0.700	0.148	166K	20.52	0.756	0.140	624K
VolSplat	15.86	0.602	0.306	152K	22.22	0.757	0.216	246K
TokenGS	17.71	0.643	0.354	65K	23.82	0.844	0.173	65K
Ours	**20.48**	**0.771**	**0.137**	65K	**24.78**	**0.847**	**0.120**	65K

24.78 dB and LPIPS decreases from 0.173 to 0.120. Notably, even compared with pixel-aligned approaches that predict considerably more Gaussians, GeoGSplat still achieves the best PSNR, SSIM, and LPIPS. For example, AnySplat* uses 624K Gaussians on NRGBD, nearly $10\times$ more than GeoGSplat, yet attains lower NVS quality.

Table 2: **3D reconstruction with four input views.** Pred→GT and GT→Pred denote Chamfer distances from predicted points to ground-truth points and vice versa, respectively. CD indicates their average. The best results are shown in bold.

Method	DTU			NRGBD		
	Pred→GT↓	GT→Pred↓	CD↓	Pred→GT↓	GT→Pred↓	CD↓
MVSplat	0.271	0.061	0.166	0.091	0.080	0.085
DepthSplat	0.130	0.066	0.098	0.093	0.058	0.076
AnySplat	0.048	0.032	0.040	0.063	0.046	0.055
AnySplat*	0.033	**0.016**	0.025	0.046	0.038	0.042
VolSplat	0.091	0.035	0.063	0.061	0.059	0.060
TokenGS	0.108	0.057	0.082	0.515	0.039	0.277
Ours	**0.019**	**0.016**	**0.018**	**0.017**	**0.017**	**0.017**

3D reconstruction. The NVS improvement is accompanied by significantly more accurate geometric reconstruction. As shown in Table 2, GeoGSplat reduces CD from 0.082 to 0.018 on DTU and from 0.277 to 0.017 on NRGBD compared with TokenGS. It also achieves the lowest CD among all compared methods on both datasets. Notably, TokenGS obtains a Pred→GT distance of 0.515 despite a GT→Pred distance of only 0.039 on NRGBD, indicating that many predicted centers are displaced from the ground-truth geometry. GeoGSplat reduces both directional distances to 0.017. Compared with pixel-aligned baselines, such as AnySplat*, GeoGSplat also achieves lower CD, demonstrating accurate reconstruction without relying on the expensive pixel-aligned predictions.

Qualitative comparison. Fig. 4 provides a visual comparison of the synthesized novel views. TokenGS (Ren et al., 2026) produces blurred object markings on DTU and blurred textures on NRGBD. AnySplat* (Jiang et al., 2025b) yields sharper renderings, but suffers from duplicated or distorted structures, particularly on NRGBD, due to inconsistent depth predictions across views. In contrast, GeoGSplat preserves both fine appearance details and structurally faithful scene geometry. Such observations show the effectiveness of our geometry-guided non-pixel-aligned paradigm. Our explicit geometric constraints improve geometry over TokenGS, reducing blurred appearance, and non-pixel-aligned prediction implicitly aligns information across views, avoiding duplicated structures caused by the inconsistent depth.

4.3 How Geometry Affects Novel View Synthesis

Considering that inaccurate geometry may still produce plausible renderings from viewpoints close to the input views, we examine whether the impact of geometric errors becomes more apparent under larger viewpoint changes. We keep the context views fixed and compare GeoGSplat and

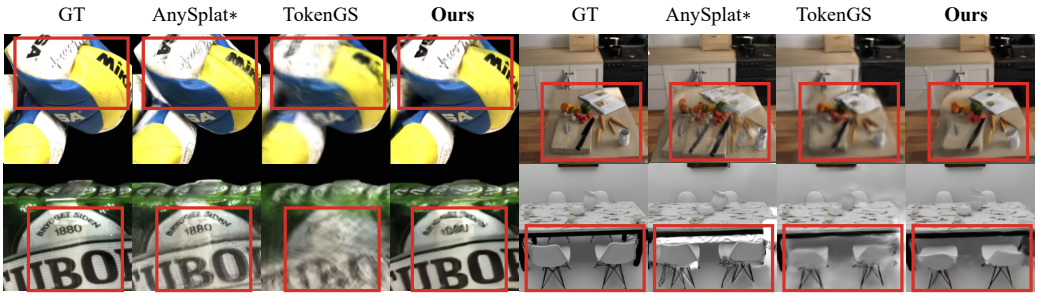


Figure 4: **Qualitative novel view synthesis with four input views.** Columns show ground truth, AnySplat*, TokenGS, and GeoGSplat from left to right. Red boxes highlight blurred details in TokenGS and shifted or duplicated structures in AnySplat. In contrast, GeoGSplat preserves finer details and more accurate scene structures in these regions.

TokenGS along the same target-view trajectories on NRGBD. Please refer to the Appendix B.6 for more details.

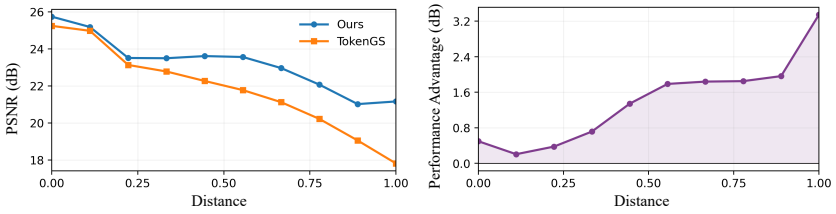


Figure 5: **NVS under varying viewpoint changes on NRGBD.** The plots show PSNR (left) and the performance advantage over TokenGS (right) at different target-context distances. The performance gap between GeoGSplat and TokenGS becomes larger as the target-context distance increases.

As shown in Fig. 5, both methods achieve similar performance for target views close to the context views. As the distance increases, the performance gap between GeoGSplat and TokenGS becomes larger, with GeoGSplat maintaining higher PSNR. These results suggest that TokenGS can produce plausible renderings near the input views despite inaccurate geometry, but its geometric errors become increasingly apparent as the viewpoint changes. In contrast, GeoGSplat benefits from reliable geometry, leading to more robust NVS under larger viewpoint changes. Fig. 6 provides a detailed

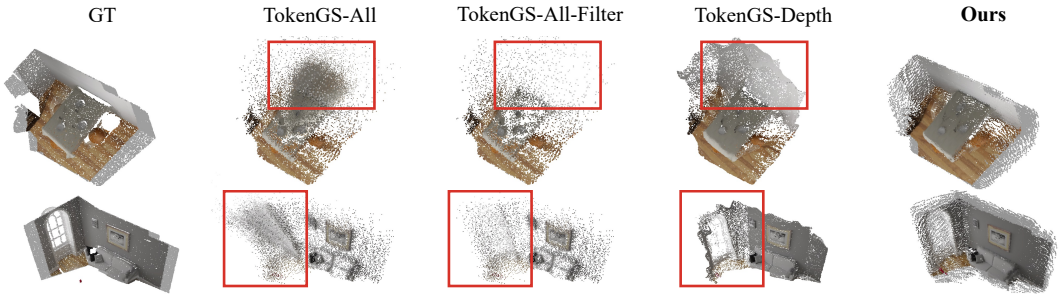


Figure 6: **3D reconstruction on NRGBD.** From left to right: ground truth, TokenGS Gaussian centers, opacity-filtered TokenGS centers, points unprojected from depth rendered by TokenGS, and GeoGSplat dense points. TokenGS contains substantial geometric outliers that remain after opacity filtering, while its depth-based reconstruction is also displaced from the ground truth. GeoGSplat predicts points closer to the ground-truth geometry and provides more complete scene coverage.

comparison of the recovered geometry on NRGBD. TokenGS predicts Gaussian centers that are substantially far from the ground-truth points, consistent with its high Pred→GT distance reported in

Sec. 4.2. These outliers remain after removing low-opacity Gaussians, indicating that they are not simply caused by low-contribution primitives. Points unprojected from TokenGS rendered depth also remain displaced from the ground truth. In contrast, GeoGSplat reconstructs points that are much closer to the ground-truth geometry.

4.4 Ablation Studies

We conduct comprehensive ablations on the key components of GeoGSplat and report the results in Table 3. All variants use the same data and evaluation protocol, ensuring fair comparisons. For computational efficiency, we conduct all ablation studies using a smaller GeoGSplat model. Please refer to the Appendix B.7 for the detailed model architecture.

Table 3: **Ablation studies**. Full uses both representations, both decodings, and both geometry losses. The best result in each group is shown in bold.

Variant	DTU				NRGBD			
	PSNR↑	SSIM↑	LPIPS↓	CD↓	PSNR↑	SSIM↑	LPIPS↓	CD↓
w/o Anchor	18.67	0.700	0.188	0.031	22.46	0.782	0.185	0.030
w/o Triplane	18.63	0.663	0.253	0.031	22.47	0.733	0.229	0.031
Full	**19.79**	**0.728**	**0.183**	**0.019**	**23.70**	**0.798**	**0.164**	**0.019**
w/o Anchor-to-Point Decoding	18.32	0.653	0.276	0.032	21.81	0.703	0.276	0.041
w/o Point-to-Gaussian Decoding	18.71	0.720	0.186	0.031	22.35	0.768	0.189	0.031
Full	**19.79**	**0.728**	**0.183**	**0.019**	**23.70**	**0.798**	**0.164**	**0.019**
w/o Geometry Losses	13.60	0.518	0.343	0.218	18.53	0.600	0.301	0.135
w/o Dense-Point Loss	18.76	0.705	0.187	0.026	23.42	0.791	0.166	0.025
w/o Anchor Loss	19.60	0.724	**0.178**	0.020	**23.82**	**0.801**	**0.161**	0.022
Full	**19.79**	**0.728**	0.183	**0.019**	23.70	0.798	0.164	**0.019**

Dual-Branch Structured Encoder. We first investigate the effectiveness of the dual-branch encoder. w/o Anchor and w/o Triplane remove the Anchor and Triplane constructions, respectively, and directly predict points and Gaussians from $\mathbf{F}^A$ and $\mathbf{F}^T$. The two variants incur noticeable performance drops on DTU and NRGBD. In particular, PSNR drops by more than 1 dB on both datasets. These results validate the complementary roles of Anchors and Triplane in our encoder.

Geometry-Guided Hierarchical Decoder. We next examine the progressive decoding process. w/o Anchor-to-Point Decoding directly regresses dense points from $\mathbf{F}^A$ without querying Triplane. w/o Point-to-Gaussian Decoding removes the query process at the Gaussian prediction stage and predicts Gaussians from $\mathbf{F}^A$. Both variants perform worse than *Full*, demonstrating the benefit of progressively refining predictions through the two-stage Triplane querying mechanism.

Geometry Supervision. Since our framework is grounded in explicit geometric priors, we further investigate the effect of geometric supervision. Removing the geometry losses causes severe performance drops, demonstrating the importance of geometric supervision for maintaining reliable geometric priors in GeoGSplat. Removing either loss also degrades performance, with the dense-point loss having a larger impact on both NVS and geometry. *Full* achieves the lowest CD, showing that supervising both Anchor centers and dense points contributes to accurate geometric reconstruction.

5 Conclusion

We presented GeoGSplat, a geometry-grounded non-pixel-aligned framework for feed-forward Gaussian Splatting that maintains a fixed Gaussian budget while incorporating explicit geometric priors. GeoGSplat combines a dual-branch structured encoder, which represents the scene through complementary Anchors and Triplane, with a geometry-guided hierarchical decoder that progressively reconstructs dense 3D points before predicting Gaussian primitives. Experiments on DTU and NRGBD demonstrate consistent improvements over both pixel-aligned and non-pixel-aligned baselines in NVS and 3D reconstruction, with the advantage becoming more significant under large viewpoint changes. These results highlight the importance of explicit geometric grounding for robust and generalizable feed-forward Gaussian Splatting.

AI USE STATEMENT

In this work, we used generative AI tools for code debugging, programming assistance, and translation assistance. We have not used generative AI tools for generating synthetic data sets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, cleaning or reformatting datasets, supporting qualitative or thematic data analysis, or interpreting results. Additionally, we used generative AI tools for editing the manuscript to improve readability. We have reviewed all AI-assisted work. The AI-assisted code was tested and verified by the authors for correctness and reproducibility, and the AI-assisted text edits were reviewed and revised by the authors to ensure accuracy and clarity. Generative AI was not used for ideation, hypothesis generation, or research design. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REFERENCES

Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6280–6291, 2022.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5460–5469, 2022.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pp. 213–229, 2020.

Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16123–16133, 2022.

David Charatan, Sizhe Lester 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*, pp. 19457–19467, 2024.

Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14104–14113, 2021.

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 *European Conference on Computer Vision*, pp. 370–386, 2024.

Afshin Dehghan, Gilad Baruch, Zhuoyuan Chen, Yuri Feigin, Peter Fu, Thomas Gebauer, Daniel Kurz, Tal Dimry, Brandon Joffe, Arik Schwartz, et al. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. *NeurIPS Datasets and Benchmarks*, 2(6):16, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*, 2021.

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5491–5500, 2022.

Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2492–2501, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. In *International Conference on Learning Representations*, volume 2024, pp. 50678–50702, 2024.

Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 406–413, 2014.

Hanwen Jiang, Hao Tan, Peng Wang, Haian Jin, Yue Zhao, Sai Bi, Kai Zhang, Fujun Luan, Kalyan Sunkavalli, Qixing Huang, et al. Rayzer: A self-supervised large view synthesis model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4918–4929, 2025a.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Transactions on Graphics*, 44(6):1–16, 2025b.

Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18344–18347, 2022.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, George Drettakis, et al. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139–1, 2023.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, pp. 71–91, 2024.

Wenyu Li, Sidun Liu, Tongrui Hu, Peng Qiao, and Yong Dou. Locusgs: Spatially grounded tokens for feed-forward 3d gaussian splatting. *arXiv preprint arXiv:2608.12825*, 2026a.

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*, 2026b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20654–20664, 2024.

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.

Julius Plucker. I. on a new geometry of space. *Proceedings of the Royal Society of London*, pp. 53–58, 1865.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10881–10891, 2021.

Jiawei Ren, Michal Tyszkiewicz, Jiahui Huang, and Zan Gojcic. Tokengs: Decoupling 3d gaussian prediction from pixels with learnable tokens. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10208–10217, 2024.

RC Bolles HG Barrow JM Tenenbaum, RC Bolles, HC Wolf, et al. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 659–663, 1977.

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

Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4690–4699, 2021.

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

Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, et al. Rodin: A generative model for sculpting 3d digital avatars using diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4563–4573, 2023.

Weijie Wang, Yeqing Chen, Zeyu Zhang, Hengyu Liu, Haoxiao Wang, Zhiyuan Feng, Wenkang Qin, Feng Chen, Jia-Wang Bian, Zheng Zhu, et al. Volsplat: Rethinking feed-forward 3d gaussian splatting with voxel-aligned prediction. In *European Conference on Computer Vision*, pp. 1–19, 2026.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. Rgbd objects in the wild: Scaling real-world 3d object learning from rgb-d videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22378–22389, 2024.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. Native and compact structured latents for 3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14419–14429, 2026.

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*, pp. 16453–16463, 2025.

Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *European Conference on Computer Vision*, pp. 785–801, 2018.

Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1787–1796, 2020.

Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12–22, 2023.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4576–4585, 2021.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. In *European Conference on Computer Vision*, pp. 1–19, 2024.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

Xiaoxue Zhang, Xiaoxu Zheng, Yixuan Yin, Tiao Zhao, Kaihua Tang, Michael Bi Mi, Zhan Xu, and Dave Zhenyu Chen. Anchorsplat: Feed-forward 3d gaussian splatting with 3d geometric priors. *arXiv preprint arXiv:2604.07053*, 2026.

Qitao Zhao, Hao Tan, Qianqian Wang, Sai Bi, Kai Zhang, Kalyan Sunkavalli, Shubham Tulsiani, and Hanwen Jiang. E-rayzer: Self-supervised 3d reconstruction as spatial visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7525–7535, 2026.

A More Experiments

A.1 Comparison with Retrained TokenGS

The evaluation of TokenGS in Sec. 4.2 uses the publicly released model, which differs from GeoGSplat in training data. To further isolate the effect of our geometry-grounded design, we retrain TokenGS under the same setting as GeoGSplat. Both variants are trained under the same training data, input resolution, Gaussian budget, and training schedule. *TokenGS-R* uses the original photometric loss, while *TokenGS-R + Geometry loss* additionally uses the geometry loss on the Gaussian centers. The results are reported in Table 4.

Table 4: **Comparison with retrained TokenGS under four input views.** Both TokenGS variants use the same setting as GeoGSplat and differ only in whether the geometry loss is applied.

Variant	DTU				NRGBD			
	PSNR↑	SSIM↑	LPIPS↓	CD↓	PSNR↑	SSIM↑	LPIPS↓	CD↓
TokenGS-R	13.75	0.571	0.386	0.277	22.48	0.728	0.234	0.180
TokenGS-R + Geometry loss	14.41	0.608	0.304	0.044	24.17	0.797	0.158	0.029
GeoGSplat	**20.48**	**0.771**	**0.137**	**0.018**	**24.78**	**0.847**	**0.120**	**0.017**

Both retrained TokenGS variants perform substantially worse than GeoGSplat on both datasets. Adding the geometry loss improves both NVS and geometric accuracy, but a substantial gap remains. The results indicate that the gains of GeoGSplat do not simply come from adding training data or geometric supervision, but from the geometry-grounded architecture itself.

A.2 Generalization Across Input View Counts

We extend the evaluation by testing the methods on 2 and 12 input views without retraining, and report the results in Table 5. With 2 input views, GeoGSplat achieves the highest PSNR among the compared methods on both datasets, and the lowest CD on NRGBD. With 12 input views, GeoGSplat maintains its leading performance, further improving PSNR to 22.53 dB on DTU and 25.21 dB on NRGBD, and reducing CD to 0.014 and 0.015, respectively. It is worth noting that GeoGSplat keeps a fixed set of 65K Gaussians for both input settings. These results demonstrate that GeoGSplat can generalize across varying numbers of input views without retraining, while maintaining accurate geometry and high-quality NVS.

Table 5: **Evaluation under varying input view counts on DTU and NRGBD.** All methods are tested with 2 and 12 input views without retraining. We report PSNR (P) for novel view synthesis, Chamfer distance (CD) for 3D reconstruction, and the number of predicted Gaussians (NUM).

(a) DTU

Method	2 Views			12 Views		
	P↑	CD↓	NUM	P↑	CD↓	NUM
DepthSplat	14.01	0.224	78K	17.06	0.035	480K
AnySplat*	15.94	**0.034**	91K	18.71	0.019	352K
TokenGS	14.57	0.155	65K	18.30	0.073	65K
Ours	**17.53**	0.037	65K	**22.53**	**0.014**	65K

(b) NRGBD

Method	2 Views			12 Views		
	P↑	CD↓	NUM	P↑	CD↓	NUM
DepthSplat	20.16	0.102	127K	22.12	0.090	766K
AnySplat*	17.28	0.045	325K	22.37	0.035	1665K
TokenGS	21.84	0.330	65K	23.90	0.250	65K
Ours	**22.01**	**0.024**	65K	**25.21**	**0.015**	65K

A.3 Experiments on Mip-NeRF 360

We conducted additional experiments on Mip-NeRF 360 (Barron et al., 2022), a large-scale scene-level dataset. As shown in Table 6, GeoGSplat achieves the best PSNR of 21.59 dB and the lowest CD of 0.015 on Mip-NeRF 360, showing that our geometry-grounded design generalizes well to large-scale scenes.

Table 6: **Comparison on Mip-NeRF 360 with 12 input views.** The best results are shown in bold.

	MVSplat	DepthSplat	AnySplat*	VolSplat	TokenGS	**Ours**
PSNR↑	16.24	16.06	18.65	19.79	21.51	**21.59**
CD↓	0.106	0.063	0.081	0.058	0.132	**0.015**

A.4 Chamfer Distance Comparison between Dense Points and Gaussian Centers

We compare the Chamfer distance computed on the dense points and the final Gaussian centers after Point-to-Gaussian decoding. As shown in Table 7, the two measurements are nearly identical across all settings. The predicted Gaussian offsets adjust Gaussian centers for better rendering quality, while preserving the high-quality geometry reconstructed by the dense points.

Table 7: **Consistency between dense points and Gaussian centers.** We report Chamfer distance computed on the dense points from Anchor-to-Point decoding and the final Gaussian centers after Point-to-Gaussian decoding.

Dataset	Views	Dense Points CD↓	Gaussian Centers CD↓
DTU	2V	0.037148	0.037130
DTU	4V	0.017984	0.017879
DTU	12V	0.014296	0.014189
NRGBD	2V	0.023718	0.023623
NRGBD	4V	0.017406	0.017485
NRGBD	12V	0.015325	0.015472

A.5 Additional Visualizations

We provide more visualizations in Fig. 7, where we can observe that our GeoGSplat reconstructs cleaner Gaussian centers and renders higher-quality images than TokenGS.

B Implementation Details

B.1 Shared Attention Module

The attention module $\mathrm{Att}(\cdot)$ used in Eq. 3 consists of a cross-attention layer, a self-attention layer, and a feed-forward network (FFN). Given query tokens $\mathbf{Q}$ and image features $\mathbf{F}_{\mathrm{img}}$, we first perform cross-attention to aggregate image features:

$$\mathbf{H}_1 = \mathbf{Q} + \mathrm{MHA}\left(\mathbf{Q}, \mathbf{F}_{\mathrm{img}}, \mathbf{F}_{\mathrm{img}}\right), \tag{12}$$

where $\mathrm{MHA}(\cdot,\cdot,\cdot)$ denotes multi-head attention. We then apply self-attention among the query tokens:

$$\mathbf{H}_2 = \mathbf{H}_1 + \mathrm{MHA}\left(\mathbf{H}_1, \mathbf{H}_1, \mathbf{H}_1\right). \tag{13}$$

The output is further refined by an FFN:

$$\mathbf{H}_3 = \mathbf{H}_2 + \mathrm{FFN}\left(\mathbf{H}_2\right), \tag{14}$$

where $\mathrm{FFN}(\cdot)$ consists of two linear layers. The same attention module is shared between the Anchor and Triplane branches, while the two branches use independent query tokens.

B.2 Triplane Refinement Block

Each refinement stage in Eq. (6) consists of three operations: a residual convolution that refines each plane locally, a cross-plane fusion that exchanges information between the three planes, and an image-conditioned cross-attention that re-injects image evidence at the new resolution.

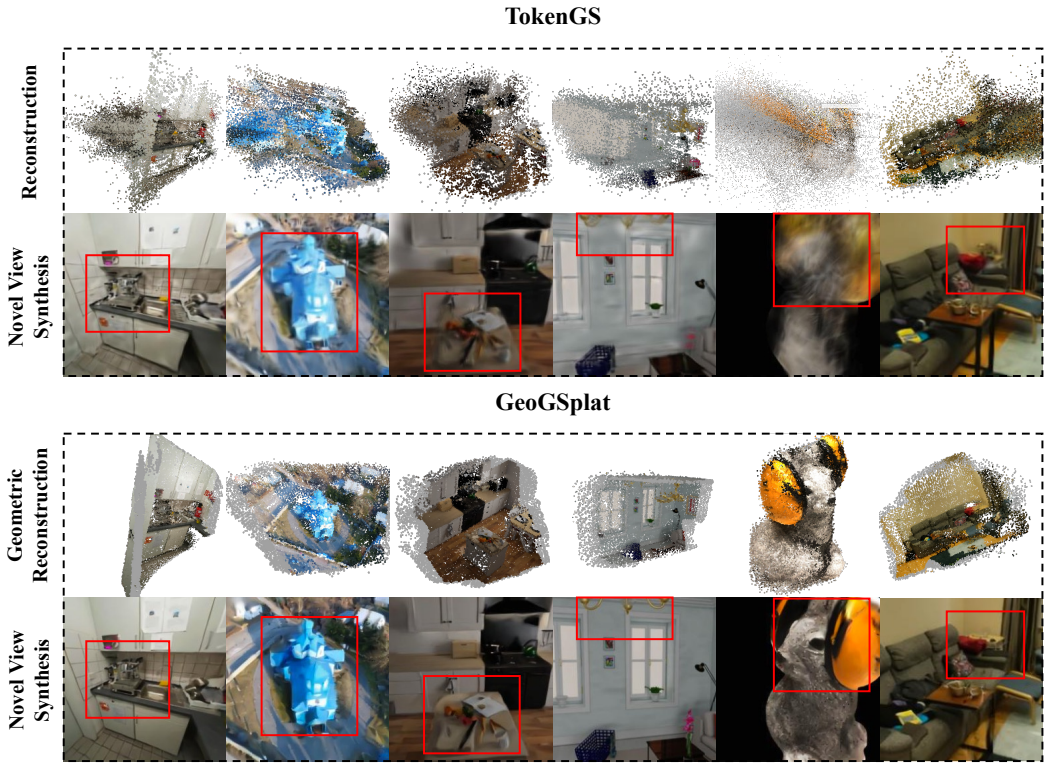


Figure 7: **Comparison of reconstruction and novel view synthesis.**

First, the Triplane resolution is doubled, followed by two residual convolutional blocks:

$$\mathbf{Z}^a \leftarrow \mathbf{Z}^a + \mathrm{Conv}_{3\times3}\left(\mathrm{Conv}_{3\times3}(\mathbf{Z}^a)\right), \tag{15}$$

where $\mathbf{Z}^a$ denotes the feature map of plane $a \in \{xy, xz, yz\}$, and $\mathrm{Conv}_{3\times3}$ denotes a 3×3 convolution. We then perform cross-plane fusion by exchanging features between the three orthogonal planes. For instance, the xy and xz planes share the x-axis. We average the features of the xz plane along the z-axis and replicate the resulting features along the y-axis to match the xy plane:

$$\mathbf{P}^{xz\to xy} = \mathrm{Expand}_y\left(\mathrm{Mean}_z(\mathbf{Z}^{xz})\right). \tag{16}$$

where Mean_z averages along the z-axis, and Expand_y replicates the resulting features along the y-axis. The same operation is applied to the other pairs of orthogonal planes. The pooled features from the other two planes are concatenated with the target plane and fused by convolution:

$$\mathbf{Z}^a \leftarrow \mathbf{Z}^a + \mathrm{Conv}\left(\left[\mathbf{Z}^a; \mathbf{P}^{b\to a}; \mathbf{P}^{c\to a}\right]\right), \tag{17}$$

where $\{b, c\} = \{xy, xz, yz\} \setminus \{a\}$. Finally, image features are re-injected at each resolution through image-conditioned cross-attention:

$$\mathbf{Z} \leftarrow \mathbf{Z} + \mathrm{CrossAtt}\left(\mathbf{Z}, \mathbf{F}_{\mathrm{img}}, \mathbf{F}_{\mathrm{img}}\right), \tag{18}$$

where $\mathbf{Z}$ stacks the features of the three planes. The refined features are reshaped into three orthogonal planes to obtain $\mathbf{T}_{2s}$.

Across the refinement stages, the Triplane is progressively refined from 64^2 to 128^2 and 256^2, with channel dimensions of 1024, 768, and 384, respectively.

B.3 Reference Initialization

We use fixed spatial references for both Anchor center prediction and Anchor-to-Point prediction. For Anchor centers, we randomly sample M reference centers $\{\mathbf{p}_i^0\}_{i=1}^M$ from the normalized scene

space. The reference centers are sampled once at initialization and then kept fixed throughout training.

For Anchor-to-Point prediction, we randomly sample K normalized reference offsets $\{\delta_k\}_{k=1}^{K}$ within the Anchor region. The same set of offsets is sampled once at initialization and shared across all Anchors and training samples. The corresponding reference points are defined as

$$\mathbf{q}_{ik} = \mathbf{p}_i + r_i \delta_k. \tag{19}$$

B.4 Dataset-Specific View Sampling

The five training datasets have different view organizations, and we therefore use dataset-specific sampling strategies to construct four input views. For ScanNet++ and ARKitScenes, which consist of continuous indoor camera trajectories, we first randomly select a reference view and then sample three additional views within a predefined pose-distance range, avoiding views that are either too close or too far from the reference. For BlendedMVS, we first define minimum and maximum temporal gaps, then randomly select a value within this range to determine the interval between the first and last frames; views are then randomly sampled from within this interval. For CO3D and WildRGB-D, where each sequence contains views of a single object from different viewpoints, we uniformly sample four distinct views from the same sequence.

B.5 Normalized Geometric Evaluation

We evaluate geometry in a common normalized coordinate system defined by the context cameras and ground-truth geometry. The context camera poses determine the orientation and origin of this coordinate system, while the ground-truth point cloud determines its scale. We then transform the ground-truth point cloud and all context cameras into this frame, and normalize the scene to fit within $[-1, 1]^3$. For each method, the predicted points are transformed into the same frame using the known correspondence between its context cameras and the ground-truth context cameras. Thus, both ground-truth and predicted points are evaluated in a shared coordinate system.

B.6 Target-Context Distance

We use the *target-context distance* to characterize the viewpoint change in Sec. 4.3. For each target view, it is defined as the Euclidean distance between the target camera center and the nearest context camera center.

As the distance increases, the target view may contain regions that are unseen in all context views. These regions are not observed in the input views. Thus, we compute PSNR only on target pixels that are visible in at least one context view, so that the metric directly reflects the rendering quality of reconstructed regions.

B.7 Detailed Model Architecture

We use two model settings in our experiments. The smaller model is used for all ablation studies to reduce computational cost, while the main-table model is used for comparison with prior methods. Their detailed architectures are summarized in Table 8.

Table 8: **Architecture comparison between the smaller model used for ablations and the model used in the main tables.**

Component	Ablation model	Main-table model
Input resolution	256×256	256×256
Patch size	8×8	8×8
Image encoder blocks	6	8
Shared Attention blocks	8	12
Triplane resolutions	$64 \to 128 \to 256$	$64 \to 128 \to 256$
Anchors $\times$ points per Anchor	4096×16	4096×16
Global batch size	64	128