# ARGEOSPLAT: AUTOREGRESSIVE GEOMETRY GENERATION AND REFINEMENT FOR SPARSE-VIEW 3D RECONSTRUCTION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

A few posed photographs leave much of a scene unseen. Multiview generation can fill in missing views, but plausible RGB images alone do not specify a renderable 3D scene. We extend a next-scale autoregressive multiview generator to produce RGB images and camera-frame pointmaps together. Separate bitwise tokenizers and prediction heads represent appearance and geometry, while a shared transformer conditions each scale on both modalities at earlier scales. We train the geometry predictions with decoded XYZ supervision through a straight-through estimator and a ground-truth-guided cross-view consistency loss. The generated pointmaps then provide proposals for a feed-forward Gaussian predictor. A multiview cost volume searches near each proposed depth to correct local errors, and training on free-running generator outputs exposes the predictor to the errors it will see at inference. With two or three input views, our $256 \times 256$ Gaussian renders exceed EscherNet in PSNR by 1.47–4.02 dB on GSO and OO3D and reduce cross-view photometric error by 0.16–0.24. In ablations, geometry supervision lowers XYZ L2 from 0.0194 to 0.0144, while local depth refinement raises render PSNR from 21.11 to 23.12.

## 1 INTRODUCTION

Reconstructing a 3D scene from only a few posed images is under-constrained: large portions of the scene may be unobserved, and the available views often overlap too little for reliable geometric reasoning. Recent multiview generative models can ease this limitation by synthesizing additional views conditioned on the sparse observations and target cameras. These generated views can reveal appearance that is missing from the input set, but using them for 3D reconstruction raises a stricter requirement than image synthesis alone. The generated observations must look plausible and must also agree on the geometry of a common 3D scene. Even small geometric inconsistencies can lead to incorrect surface placement and visible artifacts once the generated views are fused into an explicit representation.

To make the geometry of generated views explicit, we pair each RGB image with a pointmap. Each valid pixel then has a 3D coordinate in its camera frame, which serves as a surface proposal for reconstruction (Wang et al., 2024). Bolt3D has shown that jointly generated RGB images and pointmaps can support feed-forward Gaussian scene prediction, using a diffusion model and a dedicated geometry autoencoder (Szymanowicz et al., 2025). Our starting point is NAMVIS, which generates posed multiview RGB images through next-scale autoregression: it processes all target views together at each scale, then proceeds from coarse to fine (Khafizov et al., 2026). We extend that discrete formulation to generate pointmaps alongside RGB and use them to predict a Gaussian scene.

Our generator uses separate lookup-free quantized tokenizers and output heads for RGB and geometry. Their embeddings enter a shared multiview transformer, so predictions at a finer scale depend on the color and pointmap codes generated at earlier scales. Within a scale, the bit heads predict all views in parallel; no same-scale sampled bit is fed back to predict another. A pointmap tokenizer trained on masked XYZ, spatial gradients, normals, and validity preserves geometric detail that an RGB tokenizer is not trained to retain. Only posed RGB source images are required at inference.



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.

Bitwise classification alone does not measure the 3D effect of a geometry mistake. We therefore decode hard predicted geometry bits during training and supervise the resulting 3D coordinates, using a straight-through estimator to propagate the loss to the logits. A second loss compares predictions at cross-view matches verified by ground-truth geometry. These objectives link the discrete generator to the pointmaps that the Gaussian predictor will use.

The predictor must also tolerate generator errors. Later autoregressive scales receive the model's own earlier predictions at inference, whereas training approximates those errors with bitwise self-correction. We use separate, scale-dependent corruption schedules for RGB and geometry, but independent bit flips cannot reproduce every structured error in a free-running rollout. We therefore train the Gaussian predictor with generated observations among its inputs.

Generated pointmaps provide useful surface locations, yet their depths can still be wrong. We build a multiview cost volume around each proposed point and use cross-view feature matches to predict a bounded local depth correction. Direct relative-depth supervision trains this correction alongside the rendering objective. The corrected points anchor the predicted Gaussian centers, and a bounded projected-scale parameterization controls primitive size and memory use during rendering. Generative completion is thus paired with local geometric verification, and the generated pointmap is not taken as final geometry.

We report novel-view synthesis and cross-view consistency on GSO and OmniObject3D. With two or three observed views, our Gaussian renders exceed EscherNet in PSNR by 1.47–4.02 dB while reducing cross-view photometric error. Controlled ablations isolate the effects of decoded XYZ supervision, cross-view geometry supervision, and local cost-volume depth refinement.

Our contributions are:

- A next-scale autoregressive model that generates multiview RGB and camera-frame pointmaps from sparse posed RGB observations.
- A geometry-specific discrete autoencoder with lookup-free quantization, together with decoded-XYZ supervision through a straight-through estimator and ground-truth-guided cross-view consistency for autoregressive geometry prediction.
- A feed-forward Gaussian predictor that uses generated pointmaps as proposals and locally refines their depth through multiview matching before constructing the final 3D representation.

## 2 Related Work

**Feed-forward 3D Gaussian reconstruction.** Recent methods predict 3D Gaussian representations directly from one or a few images, avoiding per-scene optimization. Splatter Image predicts pixel-aligned Gaussians with a feed-forward image network and extends the formulation to multiple views through cross-view attention (Szymanowicz et al., 2024). pixelSplat reconstructs Gaussian scenes from image pairs by predicting distributions over Gaussian locations (Charatan et al., 2024). MVSplat introduces cost-volume-based multiview matching for feed-forward Gaussian reconstruction, and DepthSplat combines such matching with strong monocular depth features (Chen et al., 2024; Xu et al., 2025). Other work relaxes camera requirements: NoPoSplat predicts Gaussians in a canonical coordinate frame directly from unposed sparse views (Ye et al., 2025). Our reconstruction stage instead receives pointmaps predicted by a generative model and uses multiview matching to locally refine these geometric proposals.

**Generative multiview synthesis for 3D reconstruction.** When the observations are extremely sparse, generative models can supply information about regions that are not directly visible in the input images. Camera-conditioned diffusion models such as Zero123++ generate consistent novel views from a single input image (Shi et al., 2023). CAT3D scales this idea to scene reconstruction by using a multiview diffusion model to synthesize additional posed images before passing the expanded view set to a conventional 3D reconstruction system (Gao et al., 2024). More recently, Bolt3D jointly generates multiview appearance and pointmaps with a latent diffusion model and predicts a Gaussian scene from the resulting representations (Szymanowicz et al., 2025). These works show that generated observations help under-constrained reconstruction. We also augment sparse

observations, but we replace iterative diffusion with next-scale autoregressive generation and tie discrete geometry prediction directly to decoded XYZ accuracy. In addition, we do not accept generated point locations as final geometry; we treat them as proposals and refine them with multiview correspondence.

**Next-scale autoregressive visual generation.** Autoregressive image models traditionally factorize images into sequences of tokens, but recent work replaces raster-order prediction with coarse-to-fine generation. VAR introduces next-scale prediction, in which progressively finer visual token maps are predicted autoregressively (Tian et al., 2024). Infinity extends this formulation with bitwise token prediction and bitwise self-correction, enabling scalable high-resolution generation (Han et al., 2025). NAMVIS adapts this approach to posed multiview synthesis by conditioning next-scale autoregression on camera geometry and predicting all target-view bits in parallel with shared multiview attention (Khafizov et al., 2026). Our work builds directly on NAMVIS and extends its RGB-only formulation with an explicit geometry pathway. We introduce a geometry-specific discrete autoencoder with lookup-free quantization and autoregressive geometry head, fuse RGB and geometry representations inside the shared multiview transformer, and supervise hard geometry predictions after decoding them into continuous camera-frame pointmaps. Next-scale autoregression can then produce novel-view appearance together with the explicit 3D evidence that the downstream Gaussian reconstruction stage needs.

Pointmap regression has also become an effective representation for dense 3D reasoning. DUSt3R directly predicts pixel-aligned 3D points and aligns them across multiple views, while MASt3R augments this representation with features optimized for image matching (Wang et al., 2024; Leroy et al., 2024). These works motivate pointmaps as a convenient interface between image-space prediction and explicit 3D geometry. In our setting, however, pointmaps are generated for unobserved target views instead of being reconstructed solely from observed images.

# 3 METHOD

## 3.1 PROBLEM FORMULATION

Let $\mathcal{S} = \{(I_i, K_i, T_i)\}_{i=1}^{N_s}$ be a set of source RGB images with known intrinsics $K_i$ and camera-to-world poses $T_i$. Given target cameras $\mathcal{C} = \{(K_j, T_j)\}_{j=1}^{N_t}$, we first generate an RGB image $\hat{I}_j$ and a pointmap $\hat{P}_j$ for each target view. A pointmap assigns an XYZ coordinate in the target camera frame to each valid pixel. We then convert the generated observations into a set of 3D Gaussians $\mathcal{G}$, which can be rendered at cameras outside $\mathcal{C}$. At inference the generator needs only source RGB images and camera parameters, not source pointmaps.

## 3.2 GEOMETRY-SPECIFIC DISCRETE REPRESENTATION

For RGB, we use the pretrained discrete autoencoder from Infinity and keep it frozen throughout training. Pointmaps require a different representation: reconstructing camera-frame geometry should preserve 3D coordinates and local surface structure rather than perceptual image appearance. We therefore train a separate geometry autoencoder with the same bitwise LFQ interface expected by the autoregressive model.

Given a target-camera pointmap, the geometry autoencoder encodes XYZ together with a binary validity (alpha) mask into multiscale lookup-free quantized (LFQ) codes, with 32 binary decisions per token. The decoder reconstructs both the XYZ coordinates and the alpha mask. Invalid point coordinates are set to zero before encoding. We train this autoencoder with

$$\mathcal{L}_{\text{geo}} = \lambda_1 \mathcal{L}_{\text{XYZ}} + \lambda_\nabla \mathcal{L}_\nabla + \lambda_n \mathcal{L}_{\text{normal}} + \lambda_\alpha \mathcal{L}_{\text{alpha}} + \mathcal{L}_{\text{LFQ}}. \tag{1}$$

Here, $\mathcal{L}_{\text{XYZ}}$ is a masked $\ell_1$ reconstruction loss. $\mathcal{L}_\nabla$ compares spatial finite differences over valid neighboring pixels, $\mathcal{L}_{\text{normal}}$ compares surface normals computed from the reconstructed and ground-truth pointmaps, and $\mathcal{L}_{\text{alpha}}$ is binary cross-entropy on the alpha mask, which indicates whether each pixel contains valid geometry. The LFQ term regularizes the discrete representation.

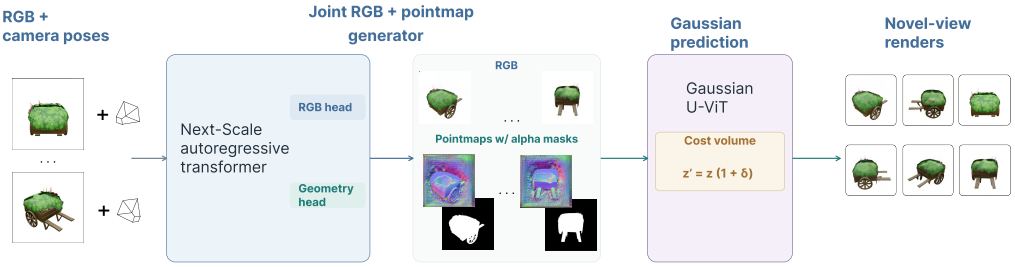


Figure 1: **ARGeoSplat architecture.** Posed source RGB images condition a next-scale autoregressive transformer that jointly predicts RGB images and pointmaps for target cameras. The generated pointmaps and validity masks provide depth proposals; a multiview cost volume refines their depths. A UViT then predicts Gaussian attributes from the generated observations and corrected geometry, producing a 3D representation that can be rendered from new viewpoints.

Appearance and geometry thus have separate discrete representations, and both autoencoders expose the same bitwise token interface to the autoregressive transformer. Both are frozen when training the autoregressive model.

### 3.3 Joint next-scale RGB and pointmap generation

For each target camera, the RGB and geometry autoencoders provide token maps $b^I_{j,s}$ and $b^P_{j,s}$ at scales $s = 1, \dots, S$. We extend the camera-conditioned transformer of NAMVIS to model

$$p_\theta(\mathbf{b}^I_{1:S}, \mathbf{b}^P_{1:S} \mid \mathcal{S}, \mathcal{C}) = \prod_{s=1}^{S} \prod_{m \in \{I,P\}} \prod_{j,u,d} p_\theta\left(b^m_{j,s,u,d} \mid \mathbf{b}^I_{<s}, \mathbf{b}^P_{<s}, \mathcal{S}, \mathcal{C}\right), \tag{2}$$

where $u$ indexes token positions and $d$ indexes bits. Bold symbols collect all target views. Source RGB features and camera geometry condition the transformer. At a given scale, RGB and pointmap embeddings are projected separately and added before the shared blocks. Two output heads predict the bits of their respective modalities. Conditioned on earlier scales, these bit predictions factorize across positions, views, and modalities; their shared hidden states and the codes generated at earlier scales couple the modalities across the coarse-to-fine process. The predicted LFQ codes are decoded by their respective autoencoders to obtain $\hat{I}_j$, $\hat{P}_j$, and the pointmap validity map.

During teacher-forced training, we apply bitwise self-correction to the earlier-scale context. RGB and geometry have separate maximum corruption rates $\eta^I_s$ and $\eta^P_s$ for each context scale, increasing toward finer scales. We sample a severity $u_s \sim \mathcal{U}(0,1)$ at each scale and flip context bits at rates $u_s\eta^I_s$ and $u_s\eta^P_s$. The severity is shared across modalities, while their flip masks are independent. The corrupted codes are requantized before they condition the next-scale prediction.

The base generator objective is the sum of bitwise cross-entropies for RGB and geometry. To align discrete geometry prediction with coordinate error, we also decode hard predicted geometry bits during training. If $p$ is the predicted probability of a geometry bit being one, we use $\tilde{b} = p + \mathrm{stopgrad}(\mathbf{1}[p > 1/2] - p)$: the forward pass uses a hard bit, while the backward pass uses the gradient of $p$. The frozen geometry decoder then produces $\tilde{P}_j = D_P(\tilde{\mathbf{b}}^P_j)$, and we apply a masked Smooth $\ell_1$ loss on XYZ coordinates, $\mathcal{L}^{\mathrm{AR}}_{\mathrm{XYZ}}$.

We additionally encourage agreement between predicted pointmaps at overlapping target views. For each pair of views, we project valid ground-truth points from one view into the other and keep matches whose sampled ground-truth 3D positions agree within a threshold. After converting the predicted pointmaps to a common world frame, a Smooth $\ell_1$ loss penalizes disagreement at these GT-guided matches. Ground truth selects correspondences; the loss itself compares predictions. The full generator objective is

$$\mathcal{L}_{\mathrm{gen}} = \mathcal{L}^I_{\mathrm{bit}} + \lambda_P \mathcal{L}^P_{\mathrm{bit}} + \lambda_X \mathcal{L}^{\mathrm{AR}}_{\mathrm{XYZ}} + \lambda_C \mathcal{L}_{\mathrm{cons}}. \tag{3}$$

The consistency term is applied only when at least two target views and valid GT-guided matches are available.

### 3.4 POINTMAP-GUIDED GAUSSIAN PREDICTION

The second stage takes generated RGB, XYZ, the predicted alpha mask, and camera rays for a set of predictor views. We first move each generated point onto its pixel's camera ray while preserving its camera depth $z$. This gives a geometrically valid proposal, but does not assume that its predicted depth is exact.

We refine the proposal with a local multiview cost volume. For each valid reference pixel, we consider relative offsets $r_k \in [-\epsilon, \epsilon]$ and depths

$$z_k = z(1 + r_k), \qquad k = 1, \dots, D. \tag{4}$$

Each candidate point is projected into the other predictor views. A feature encoder and a cross-view transformer extract RGB features at one-quarter image resolution. Correlations at the projected locations are combined over valid support views, then regularized by a cost-volume U-Net. A softmax depth head produces candidate probabilities $q_k$, from which we compute

$$\delta = \sum_k q_k r_k, \qquad z' = z(1 + \delta). \tag{5}$$

We upsample the scalar correction to pointmap resolution and move the corrected 3D point along the reference camera ray. Pixels with no valid cross-view support retain their original depth. The search stays near the generated pointmap, so cross-view matching corrects the proposed geometry without replacing it.

A feed-forward UViT receives the generated RGB, corrected XYZ, ray directions, and ray moments. Gaussian centers are fixed by the corrected points; the network predicts opacity, rotation, color coefficients, and anisotropic scale. To control Gaussian growth, each predicted scale component is bounded in projected pixel units before conversion to world units:

$$\sigma_{\mathrm{px}} = \sigma_{\min} + (\sigma_{\max} - \sigma_{\min}) \operatorname{sigmoid}(a), \qquad \sigma_{\mathrm{world}} = \frac{z'}{f_{\mathrm{px}}} \sigma_{\mathrm{px}}, \tag{6}$$

where $a$ is the network output and $f_{\mathrm{px}}$ is focal length in pixels. This limits the projected extent of each primitive while allowing its world-space size to vary with depth.

### 3.5 TRAINING THE GAUSSIAN PREDICTOR ON GENERATED INPUTS

We train the Gaussian stage with free-running generator outputs among its inputs so that the predictor encounters their structured appearance and geometry errors. Rendering the predicted Gaussians at input and held-out cameras gives an image loss combining pixel error, DSSIM, and LPIPS, with a small opacity penalty.

The depth corrector also receives direct supervision on generated pointmaps. For a generated depth $z$ and reference depth $z^\star$, its relative target is $\delta^\star = (z^\star - z)/z$. We use only pixels with valid geometry in both maps and $|\delta^\star| < \epsilon$, since larger errors lie outside the correction range. A masked Smooth $\ell_1$ loss compares $\delta$ with $\delta^\star$. The Gaussian stage is trained with the rendering and depth-correction losses:

$$\mathcal{L}_{\mathrm{GS}} = \mathcal{L}_{\mathrm{render}} + \lambda_D \mathcal{L}_{\mathrm{depth}}. \tag{7}$$

Key training settings for both stages are given with the experiments.

## 4 EXPERIMENTS

### 4.1 EXPERIMENTAL SETUP

**Datasets.** We train on Objaverse-XL (Deitke et al., 2023) and report quantitative comparisons on Google Scanned Objects (GSO) (Downs et al., 2022) and OmniObject3D (OO3D) (Wu et al., 2023). GSO and OO3D are not used during training and therefore evaluate cross-dataset generalization.

**Implementation details.** We train our joint RGB–geometry generator at $256 \times 256$ on Objaverse-XL with frozen RGB and geometry tokenizers; the geometry tokenizer uses 32-bit codes. AdamW uses a learning rate of $2 \times 10^{-4}$; geometry bitwise, decoded XYZ, and cross-view loss weights are $1$, $0.25$, and $1$. The Gaussian predictor is trained at the same resolution. Its cost volume samples 33 depths within $\pm 10\%$ of the proposed depth. AdamW learning rates are $5 \times 10^{-5}$ for the predictor and $2 \times 10^{-4}$ for the cost volume; input-view, novel-view, and direct depth-loss weights are $0.1$, $0.9$, and $1$. The evaluated Gaussian checkpoint was saved after 3,000 updates.

**Evaluation protocol.** We consider sparse-view reconstruction from $N_s \in \{1, 2, 3\}$ posed RGB input images. The same source views and held-out evaluation cameras are used for all methods whenever their input requirements permit a direct comparison. Given the sparse observations, the autoregressive generator produces posed RGB views and camera-frame pointmaps that are used to construct the input to the Gaussian predictor.

We evaluate all methods at $256 \times 256$ where supported. Splatter Image is evaluated using its released object-centric model at $128 \times 128$, and we mark this resolution difference explicitly in the comparison tables.

**Runtime.** We report inference time per object with batch size 1, measured on a single NVIDIA A40 GPU. For our method, "Ours (generated)" measures autoregressive RGB–geometry generation, while "Ours" additionally includes feed-forward Gaussian prediction. Model loading and rendering of evaluation views are excluded. All baseline runtimes are measured under the same hardware setup.

**Image quality.** All methods are evaluated at held-out target cameras using PSNR, SSIM, and LPIPS. For image-generation methods, these metrics are computed directly on their generated target views. For methods that reconstruct an explicit 3D representation, including ours, they are computed on novel views rendered from the reconstructed scene. Higher PSNR and SSIM and lower LPIPS indicate better agreement with the ground-truth target images.

**Cross-view consistency.** Per-view image metrics do not directly measure whether predictions from different viewpoints describe the same underlying surface consistently. We therefore also evaluate cross-view appearance consistency, using ground-truth geometry only to establish correspondences.

For a pair of target views $i$ and $j$, we back-project valid pixels from view $i$ using the ground-truth pointmap and transform the resulting 3D points into view $j$. We retain correspondences that are valid and mutually visible according to the ground-truth geometry. Given the resulting correspondence set $\mathcal{M}_{ij}$, we define

$$E_{\text{cons}} = \frac{1}{|\mathcal{M}_{ij}|} \sum_{(u,v)\in\mathcal{M}_{ij}} \left\| \hat{I}_i(u) - \hat{I}_j(v) \right\|_1 . \tag{8}$$

We average the score across target-view pairs and scenes. Lower values indicate stronger cross-view appearance consistency. The metric does not use geometry predicted by the evaluated method, so the same correspondences are used for image-generation and explicit-3D approaches.

**Baselines.** We compare against generative novel-view synthesis methods and feed-forward 3D reconstruction methods. The image-generation baselines include Zero123-XL (Liu et al., 2023), EscherNet (Kong et al., 2024), and Wonder3D (Long et al., 2024). For explicit 3D reconstruction, we compare against Splatter Image. Methods are evaluated only in source-view settings supported by their released models.

### 4.2 Comparison with Prior Work

Tables 1 and 2 compare image-generation and explicit-3D methods using PSNR, SSIM, LPIPS, and cross-view consistency. For our method, we report both direct generations and final 3DGS renders.

Table 1: Comparison on **GSO**. All methods are evaluated at $256 \times 256$ where supported; Splatter Image uses its released $128 \times 128$ model. Cons. denotes cross-view photometric consistency. All runtimes are measured on an NVIDIA A40. Best and second-best results within each input-view setting are shown in bold and underlined, respectively.

| Method | Output | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Cons. ↓ | Time (s) ↓ |
|---|---|---|---|---|---|---|
| **1 input view** | | | | | | |
| Zero123-XL | Images | 19.49 | **0.824** | **0.140** | 0.495 | 2.6 |
| Wonder3D | Images | 14.16 | 0.777 | 0.285 | 0.523 | 2.0 |
| EscherNet | Images | 17.32 | 0.784 | 0.213 | 0.497 | 3.2 |
| Splatter Image | 3DGS | 18.48 | 0.798 | 0.276 | **0.311** | **0.1** |
| Ours (generated) | Images | **20.02** | 0.820 | <u>0.154</u> | 0.411 | <u>0.6</u> |
| **Ours** | 3DGS | <u>19.68</u> | <u>0.822</u> | 0.196 | <u>0.325</u> | 0.8 |
| **2 input views** | | | | | | |
| EscherNet | Images | 20.43 | 0.828 | <u>0.128</u> | 0.424 | 3.2 |
| Ours (generated) | Images | **22.89** | **0.856** | **0.094** | <u>0.386</u> | **0.6** |
| **Ours** | 3DGS | <u>22.47</u> | <u>0.855</u> | 0.145 | **0.258** | <u>0.8</u> |
| **3 input views** | | | | | | |
| EscherNet | Images | 21.61 | 0.846 | <u>0.095</u> | 0.407 | 3.2 |
| Ours (generated) | Images | **23.49** | **0.863** | **0.085** | <u>0.375</u> | **0.6** |
| **Ours** | 3DGS | <u>23.09</u> | <u>0.861</u> | 0.135 | **0.247** | <u>0.8</u> |

Table 2: Comparison on **OmniObject3D (OO3D)**. All methods are evaluated at $256 \times 256$ where supported; Splatter Image uses its released $128 \times 128$ model. Cons. denotes cross-view photometric consistency. All runtimes are measured on an NVIDIA A40. Best and second-best results within each input-view setting are shown in bold and underlined, respectively.

| Method | Output | PSNR ↑ | SSIM ↑ | LPIPS ↓ | Cons. ↓ | Time (s) ↓ |
|---|---|---|---|---|---|---|
| **1 input view** | | | | | | |
| Zero123-XL | Images | <u>18.75</u> | <u>0.817</u> | <u>0.145</u> | 0.573 | 2.6 |
| Wonder3D | Images | 14.29 | 0.786 | 0.243 | 0.624 | 2.0 |
| EscherNet | Images | 14.61 | 0.764 | 0.258 | 0.603 | 3.2 |
| Splatter Image | 3DGS | 16.44 | 0.782 | 0.294 | **0.335** | **0.1** |
| Ours (generated) | Images | **18.89** | **0.821** | **0.144** | 0.491 | <u>0.6</u> |
| **Ours** | 3DGS | 18.33 | 0.811 | 0.167 | <u>0.338</u> | 0.8 |
| **2 input views** | | | | | | |
| EscherNet | Images | 16.92 | 0.798 | 0.182 | 0.508 | 3.2 |
| Ours (generated) | Images | **21.67** | **0.857** | **0.090** | <u>0.444</u> | **0.6** |
| **Ours** | 3DGS | <u>20.94</u> | <u>0.848</u> | <u>0.118</u> | **0.273** | <u>0.8</u> |
| **3 input views** | | | | | | |
| EscherNet | Images | 19.12 | 0.833 | 0.123 | 0.468 | 3.2 |
| Ours (generated) | Images | **22.50** | **0.867** | **0.078** | <u>0.424</u> | **0.6** |
| **Ours** | 3DGS | <u>21.53</u> | <u>0.853</u> | <u>0.100</u> | **0.256** | <u>0.8</u> |

For our method, we report both the images produced directly by the autoregressive generator and the novel views rendered from the final Gaussian representation.

On GSO, our Gaussian renders exceed EscherNet in PSNR by 2.04 dB with two input views and 1.47 dB with three. Their cross-view consistency error decreases from 0.424 to 0.258 and from 0.407 to 0.247, respectively. SSIM also improves, although LPIPS is higher than EscherNet's in both settings.

On OO3D, the corresponding PSNR gains are 4.02 and 2.41 dB. Cross-view consistency error decreases from 0.508 to 0.273 with two inputs and from 0.468 to 0.256 with three; LPIPS also falls from 0.182 to 0.118 and from 0.123 to 0.100. The one-view comparison is mixed: our Gaussian renders have higher PSNR than Splatter Image on both datasets, while Zero123-XL retains lower LPIPS on both and higher PSNR on OO3D.

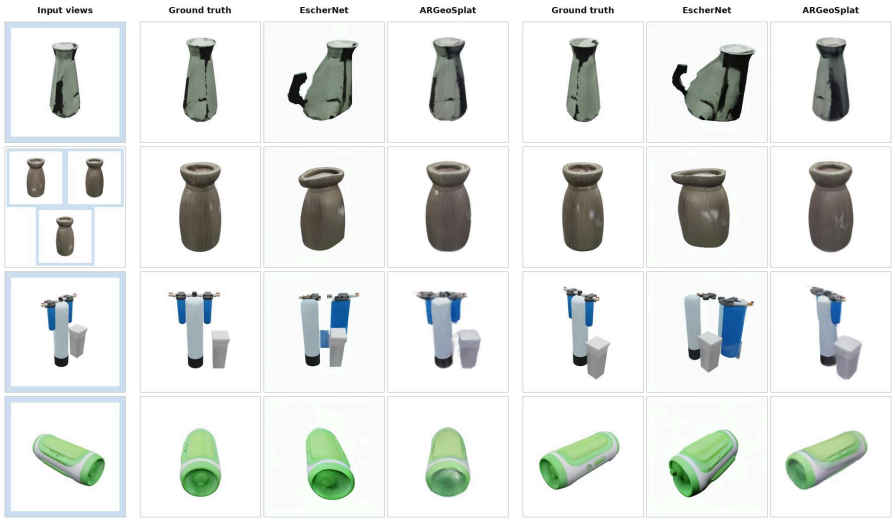


Figure 2: **Qualitative comparison with EscherNet at** $256 \times 256$**.** Selected examples compare target views generated by EscherNet and our method. Matching colors identify the source views for each scene. Examples are shown from Objaverse-XL, GSO, and OO3D.

**Inference speed.** Our autoregressive generator requires 0.6 s per object, and the complete generation-to-3DGS pipeline requires 0.8 s on an NVIDIA A40. This is substantially faster than the evaluated generative baselines, which require 2.0–3.2 s under the same hardware setup. Splatter Image is faster at 0.1 s, but it is limited to the single-view setting and achieves lower PSNR than our method on both GSO and OO3D.

Across both datasets and all input-view counts, the directly generated images have higher PSNR and lower LPIPS than our Gaussian renders. The renders have lower cross-view photometric error in every setting, showing the tradeoff between per-view image fidelity and a consistent explicit representation.

Figure 2 shows selected qualitative comparisons with EscherNet at $256 \times 256$. Figure 3 shows input views and renders of reconstructions.

### 4.3 Ablation Studies

The ablations isolate the contributions of the geometry supervision and local multiview refinement. All ablations use a held-out Objaverse subset with 1–3 input views. Geometry quality is measured by mean XYZ L2 over valid pixels:

$$E_{\mathrm{XYZ}} = \frac{1}{|\Omega|} \sum_{u \in \Omega} \left\| \hat{P}(u) - P^{\star}(u) \right\|_2 . \tag{9}$$

**Decoded-space geometry supervision.** Starting from the joint RGB–geometry autoregressive model, we first add decoded XYZ supervision through the straight-through estimator. We then add the ground-truth-guided cross-view consistency objective. This experiment measures whether supervision applied after geometry decoding improves the actual pointmaps rather than only their discrete token predictions, and whether explicit cross-view supervision further improves geometric agreement.

Table 3 shows that decoded XYZ supervision accounts for most of the reduction in mean valid-pixel XYZ L2, from 0.0194 to 0.0153. Adding ground-truth-guided cross-view supervision lowers the error further to 0.0144.



Figure 3: **Qualitative 3DGS renders.** Each row shows one object reconstructed from three posed input views. The first two columns display two of those inputs on a light blue background; the remaining six show Gaussian renders. Examples are shown from Objaverse-XL, GSO, and OO3D.

Table 3: Geometry supervision.

| Variant | XYZ L2 ↓ |
|---|---|
| Base RGB+geometry AR | 0.0194 |
| + decoded XYZ STE | 0.0153 |
| + cross-view consistency | 0.0144 |

Table 4: Local depth refinement.

| Variant | XYZ L2 ↓ | PSNR ↑ | SSIM ↑ | LPIPS ↓ |
|---|---|---|---|---|
| No CV | 0.013 | 21.11 | 0.87 | 0.14 |
| + CV | 0.010 | 23.12 | 0.90 | 0.12 |

**Local depth refinement.** We evaluate the Gaussian reconstruction stage with and without the proposed multiview cost-volume correction. In the first variant, the generated pointmaps are passed directly to the Gaussian predictor. In the second, the cost volume searches locally around each proposed depth before Gaussian prediction. We report both the resulting XYZ error and novel-view rendering quality.

As shown in Table 4, local depth refinement lowers XYZ L2 from 0.013 to 0.010 and raises render PSNR from 21.11 to 23.12 dB. SSIM also improves from 0.87 to 0.90, while LPIPS falls from 0.14 to 0.12, indicating gains in both geometry and rendered image quality.

## 5 Conclusion and Limitations

We presented ARGeoSplat, a sparse-view reconstruction framework that combines next-scale autoregressive multiview generation with explicit geometry prediction and feed-forward 3D Gaussian reconstruction. Starting from posed RGB observations, the model jointly generates RGB views and camera-frame pointmaps using separate discrete representations for appearance and geometry. Decoded XYZ supervision and ground-truth-guided cross-view consistency improve the quality of the generated pointmaps, while a local multiview cost volume refines their depths before Gaussian prediction. Experiments on GSO and OO3D show improved novel-view quality and cross-view consistency, while ablations confirm the contributions of decoded geometry supervision and depth refinement. The current method assumes known cameras and is limited to object-centric data; extending it to unposed inputs and larger scenes remains future work.

### AI USE STATEMENT

We used generative AI tools to assist with editing and improving the clarity, readability, and organization of the manuscript. All scientific content, experimental results, analyses, and conclusions were produced and verified by the authors. We reviewed all AI-assisted text and take responsibility for the final content of this work.

### REPRODUCIBILITY STATEMENT

We provide the model architecture and training objectives in Sec. 3, and describe the datasets, evaluation protocol, metrics, and ablations in Sec. 4. We plan to release the training and evaluation code upon publication.

## REFERENCES

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

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. MVSplat: Efficient 3D Gaussian splatting from sparse multi-view images. In *European Conference on Computer Vision (ECCV)*, 2024. doi: 10.1007/978-3-031-72664-4_21.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-xl: A universe of 10m+ 3d objects, 2023. URL https://arxiv.org/abs/2307.05663.

Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B. McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items, 2022. URL https://arxiv.org/abs/2204.11918.

Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole. CAT3D: Create anything in 3D with multi-view diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Jian Han, Jinlai Liu, Yi Jiang, Bin Yan, Yuqi Zhang, Zehuan Yuan, Bingyue Peng, and Xiaobing Liu. Infinity: Scaling bitwise AutoRegressive modeling for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. doi: 10.1109/CVPR52734.2025.01467.

Ramil Khafizov, Ilya Statsenko, Ruslan Rakhimov, Artem Komarichev, Peter Wonka, and Evgeny Burnaev. NAMVIS: Next-scale autoregressive multi-view image synthesis. In *Advances in Neural Information Processing Systems*, 2026. Accepted at NeurIPS 2026.

Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J. Davison. Escher-Net: A generative model for scalable view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9503–9513, 2024.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3D with MASt3R. In *European Conference on Computer Vision (ECCV)*, 2024. doi: 10.1007/978-3-031-73220-1_5.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9298–9309, 2023.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, and Wenping Wang. Wonder3D: Single image to 3D using cross-domain diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9970–9980, 2024.

Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model, 2023. URL https://arxiv.org/abs/2310.15110.

Stanislaw Szymanowicz, Christian 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 (CVPR)*, 2024.

Stanislaw Szymanowicz, Jason Y. Zhang, Pratul Srinivasan, Ruiqi Gao, Arthur Brussee, Aleksander Holynski, Ricardo Martin-Brualla, Jonathan T. Barron, and Philipp Henzler. Bolt3D: Generating 3D scenes in seconds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. doi: 10.1109/ICCV51701.2025.02304.

Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jérôme Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20697–20709, 2024. doi: 10.1109/CVPR52733.2024.01956.

Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Liang Pan Jiawei Ren, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

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 (CVPR)*, 2025.

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