Free-Range Gaussians: Non-Grid-Aligned Generative 3D Gaussian Reconstruction

Akhmedkhan Shabanov[1,2], Peter Hedman[1], Ethan Weber[1], Zhengqin Li[1], Denis Rozumny[1], Gael Le Lan[1], Naina Dhingra[1], Lei Luo[1], Andrea Vedaldi[1,3], Christian Richardt[1], Andrea Tagliasacchi[2,4], Bo Zhu[1], and Numair Khan[1]

[1] Meta Reality Labs
[2] Simon Fraser University
[3] University of Oxford
[4] University of Toronto

Abstract. We present *Free-Range Gaussians*, a multi-view reconstruction method that predicts non-pixel, non-voxel-aligned 3D Gaussians from as few as four images. This is done through flow matching over Gaussian parameters. Our generative formulation of reconstruction allows the model to be supervised with non-grid-aligned 3D data, and enables it to synthesize plausible content in unobserved regions. Thus, it improves on prior methods that produce highly redundant grid-aligned Gaussians, and suffer from holes or blurry conditional means in unobserved regions. To handle the number of Gaussians needed for high-quality results, we introduce a hierarchical patching scheme to group spatially related Gaussians into joint transformer tokens, halving the sequence length while preserving structure. We further propose a timestep-weighted rendering loss during training, and photometric gradient guidance and classifier-free guidance at inference to improve fidelity. Experiments on Objaverse and Google Scanned Objects show consistent improvements over pixel and voxel-aligned methods while using significantly fewer Gaussians, with large gains when input views leave parts of the object unobserved.

Keywords: 3D Gaussian Splatting · Sparse-View Reconstruction · Flow Matching · Generative 3D

1 Introduction

Reconstructing 3D objects from sparse images is a long-standing challenge in computer vision. 3D Gaussian splatting (3DGS) [20] has emerged as a compelling representation offering real-time differentiable rendering with high visual quality. This has spurred feed-forward methods that predict 3DGS from sparse views in a single forward pass, bypassing costly per-scene optimization.

The dominant approach is *pixel-aligned* reconstruction: each predicted Gaussian is anchored to a pixel in one of the input views. Methods such as Splatter Image [48], DepthSplat [61], LGM [50], and GS-LRM [70] all produce Gaussians that inherit the spatial structure of the input images. While effective in observed

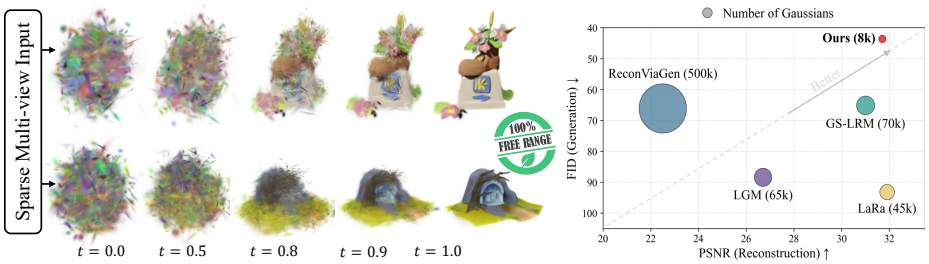

Fig. 1: Non-grid-aligned generative reconstruction. From four sparse views, our method produces a complete 3D Gaussian reconstruction via flow matching, achieving state-of-the-art quality for both observed (high PSNR) and unobserved regions (low FID) with ∼10× fewer Gaussians than grid-aligned methods.

regions, the resulting representation is highly redundant, and suffers from holes and artifacts in unobserved regions (Figure 4). A common line of work addresses this problem by synthesizing novel views to observe hidden parts, and then reconstructing from the expanded set of images [31, 34, 49]. However, identifying the novel viewpoints presupposes unavailable geometric knowledge, and inconsistent synthesized views result in low-quality reconstruction.

An alternative approach decouples the reconstruction from input pixels by using a volumetric grid [4,55]. However, the representation remains redundant, and detail is bound by grid resolution. Further, trained only with a photometric loss, these methods predict blurry conditional means for unobserved areas. Generative methods like TRELLIS [60] hallucinate sharp content in unobserved areas, but are prone to ignore the conditioning images and poses (Figure 2).

We refer to the above-mentioned approaches as grid-aligned since both anchor Gaussians to a fixed spatial grid, whether 2D image pixels or 3D voxels. We address their limitations through *generative reconstruction*: instead of deterministically regressing Gaussian parameters, we use flow matching [29] to model the conditional distribution of 3D Gaussians given sparse views. Thus, our model iteratively denoises Gaussian parameters from noise. As the output is not tied to pixels or voxels, the model can place Gaussians anywhere in 3D space, producing a more compact representation. In particular, grid-aligned baselines typically operate at resolutions on the order of 512^3 to 1536^3, whereas our method uses only 8K Gaussians, comparable to roughly 20^3 elements—several orders of magnitude fewer. Furthermore, by sampling from the conditional distribution rather than regressing its mean, our method produces sharp completions for unseen regions while staying faithful to observed views.

A key challenge is scale: high-quality reconstructions require denoising tens of thousands of Gaussian tokens. We construct a level-of-detail (LoD) hierarchy [21] over ground-truth Gaussians for coarse-to-fine training, and propose a patchification strategy that cuts the number of tokens in half while preserving locality. To recover details and improve fidelity, we combine three complementary mechanisms: (1) a photometric loss that emphasizes details at later denoising

Fig. 2: Generation vs. Reconstruction. State-of-the-art generative methods like TRELLIS [60] produce sharp output but suffer from weak input conditioning as they fail to preserve object orientation, and hallucinate geometry and appearance even in observed regions. Our results remain aligned with input-view poses, geometry, and appearance. Please see the supplementary material for additional results.

stages, (2) photometric gradient guidance [37] to steer denoising toward input views, and (3) classifier-free guidance [15]. In summary, our contributions are:

- A method to produce compact and complete non-grid-aligned 3D Gaussian reconstructions from sparse views via flow matching.
- A coarse-to-fine training and Gaussian patchification scheme based on a hierarchical LoD structure.
- Training and inference-time guidance mechanisms to recover high-frequency details and improve input fidelity.

2 Related Work

Per-Pixel Feed-Forward 3D Reconstruction. A large family of methods anchors each predicted 3D element to an input pixel or patch. Splatter Image [48] predicts one Gaussian per pixel from a single view; LGM [50] does so from four views using a U-Net. pixelSplat [3] produces per-pixel Gaussians from stereo image pairs, while GRM [63] scales the paradigm to high-resolution outputs via a large Gaussian reconstruction model. GS-LRM [70] extends LRM [16] with pixel-aligned transformer tokens and Plücker ray embeddings. Long-LRM [71] further scales feed-forward Gaussian prediction to 32 high-resolution views by replacing the quadratic-cost transformer with a hybrid Mamba2–transformer architecture. DepthSplat [61] and MVSplat [7] create Gaussians from multi-view stereo depth and cost-volume cues, respectively. MVSNeRF [5] was an early work combining plane-sweep cost volumes with volume rendering for generalizable radiance field reconstruction from multi-view stereo. AnySplat [17] extends the paradigm of feed-forward geometry foundation models like DUST3R [54], VGGT [51] and MapAnything [19] to uncalibrated collections by jointly predicting Gaussians and cameras. PF-LRM [52] addresses the pose-free setting by jointly predicting shape

and camera parameters, while Splatt3R [47] and MASt3R [24] extend DUST3R to predict Gaussians and dense stereo matches from uncalibrated image pairs. These approaches are fast and accurate in observed regions, but cannot place Gaussians in unobserved regions, limiting them to partial reconstructions.

Generate-and-Reconstruct Approaches. To cover unobserved regions, several methods first synthesize novel views and then reconstruct from the expanded set. Zero-1-to-3 [31] and SyncDreamer [34] generate the expanded set via view-conditioned diffusion for downstream 3D reconstruction. One-2-3-45++ [30] extends this pipeline with consistent multi-view generation and 3D diffusion, while InstantMesh [62] streamlines it by coupling a multi-view diffusion model with a sparse-view large reconstruction model for efficient mesh extraction. ReconFusion [58] regularizes a neural radiance field (NeRF) [36] at novel poses with a diffusion prior, but still requires per-scene optimization. Similarly, iFusion [57] leverages Zero-1-to-3 predictions within an optimization pipeline to align poses and generate novel views, but the inconsistency between synthesized views still limits reconstruction quality. Bolt3D [49] generates per-pixel Gaussian maps from a multi-view diffusion model. Wonderland [27] and Lyra [1] both leverage video diffusion models to produce 3D Gaussians – the latter using self-distillation without requiring multi-view training data. These methods offer test-time adaptivity, but selecting the viewpoints to synthesize requires a priori scene understanding, and artifacts from inconsistent generated images can propagate to the 3D output.

Full Reconstruction via Structured Representations. An alternative is to predict 3D content on a volumetric grid. Among early work in this direction, pixelNeRF [65] projects image features onto 3D query points. More recently, LRM [16] predicts tri-planes from a vision transformer, whereas LaRa [4] and VolSplat [55] predict Gaussians on voxel grids. MeshLRM [56] adapts the LRM architecture to directly produce high-quality meshes rather than radiance fields. SCube [44] combines a hierarchical sparse-voxel scaffold with latent diffusion. While decoupling the output from input pixels, the quality of these methods is bound by the grid resolution. Further, under photometric losses, they predict the conditional mean of unobserved regions, yielding complete but blurry results.

Generative 3D Models. DreamFusion [43] distills 2D diffusion priors into 3D but requires per-object optimization. Feed-forward alternatives apply diffusion directly to 3D representations such as point clouds [68], or latent-space encodings of these representations [26, 66]. Direct3D [59] applies a 3D latent diffusion transformer for scalable image-to-3D generation. Among Gaussian-based generators, GaussianCube [69] is tied to a voxel-structured representation, which inherits grid-capacity limits. The same is true of TRELLIS [60], although it encodes the grid into a highly compact latent representation. L3DG [45] performs latent diffusion with a VQ-VAE [40], but its latent construction also relies on 3D grid structure. GSD [37] uses a Gaussian diffusion prior with view guidance at sampling time, but does not train a dedicated multi-view conditional model for sparse-view reconstruction. Moreover, its guidance backpropagates the image loss through the denoiser and renderer to the noisy latent, whereas our guidance is computed directly from the rendering loss on the current noisy Gaussians. DiffSplat [28]

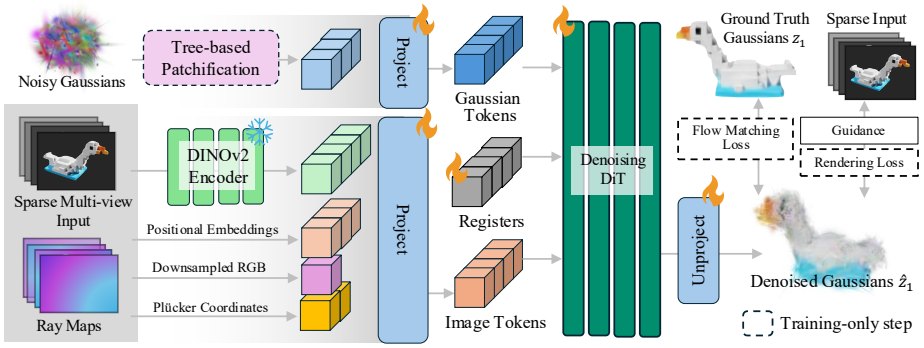

Fig. 3: Method overview. We pose reconstruction as a generative process based on flow matching. Conditioned on sparse image features, a diffusion transformer denoises a set of non-grid-aligned 3D Gaussians from random noise (Section 3.1). A level-of-detail tree enables Gaussian patchification for tractable sequence lengths (Section 3.2). A timestep-weighted rendering loss, photometric gradient guidance, and classifier-free guidance steer the output toward consistent reconstructions (Section 3.3).

and GaussianObject [64] leverage 2D diffusion priors for Gaussian generation and refinement. While generative methods produce high-quality completions in unseen regions, they are prone to ignoring or modulating the conditioning signals.

Concurrent Work. ReconViaGen [2] conditions a pretrained TRELLIS generator with VGGT features in a voxel latent space. Unlike ReconViaGen, our approach operates *directly* on Gaussian parameters, without an intermediate voxel representation or multiple large pretrained models.

3 Free-Range Gaussians

Our goal is to train a feed-forward model for non-grid-aligned 3D Gaussian reconstruction. Given a sparse set $\mathcal{I}$ of posed multi-view images of an object, we aim to predict a set $\mathcal{G}$ of 3D Gaussians that faithfully reconstructs the object. We do this without imposing any pixel- or voxel-based structure on the Gaussians by framing reconstruction as a generative process based on flow matching (Section 3.1). Starting from random noise, the model iteratively refines a set of Gaussian parameters conditioned on the input views $\mathcal{I}$. This generative formulation has two advantages. First, it allows the model to produce more plausible completions in unobserved regions than is possible with a regression loss. Second, it enables direct supervision using data that is not anchored to a voxel or pixel grid. To make the generative process computationally tractable, we organize the Gaussians into a hierarchical tree structure that reduces the number of tokens fed to the denoising model (Section 3.2). Finally, we introduce guidance mechanisms to steer generation toward outputs that are consistent with the input views (Section 3.3). Figure 3 provides an overview of our pipeline.

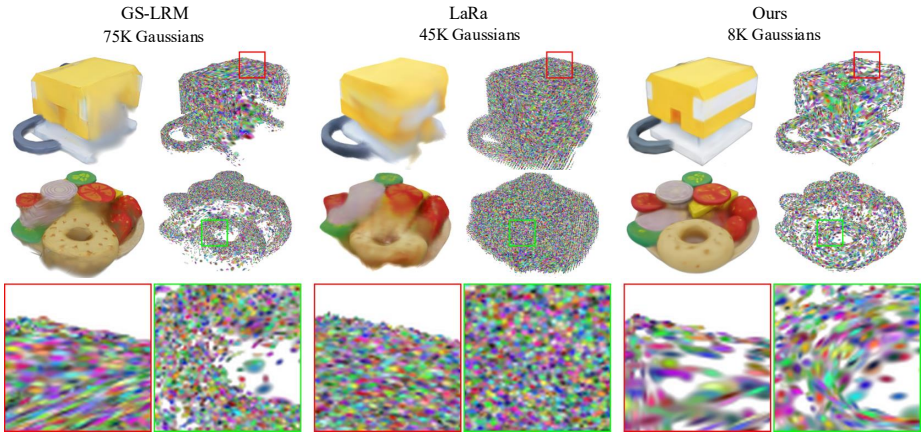

Fig. 4: Generative reconstruction is efficient and complete. We visualize reconstruction density by rendering scaled-down Gaussians with random colors. Pixel-aligned methods like GS-LRM [70] leave holes in unobserved regions despite a highly redundant representation. Voxel-aligned methods like LaRa [4] are more complete but predict blurry conditional means for unseen parts. Our generative approach produces sharp, plausible completions with far fewer non-grid-aligned Gaussians.

3.1 3D Gaussians via Generative Reconstruction

We frame sparse-view reconstruction as a generative task to address two shortcomings of existing work. First, regression-based methods trained with L2 losses [4, 5, 53, 65, 71] predict the conditional expectation $\mathbb{E}[\mathcal{G} \mid \mathcal{I}]$, which averages over all plausible completions and produces blurry results in unseen regions (Figure 4). A generative model instead samples from the conditional distribution $P(\mathcal{G} \mid \mathcal{I})$, enabling sharp completions. Second, existing feed-forward methods anchor each Gaussian to either a voxel [2, 4, 55] or an input pixel [3, 7, 48, 50, 70], producing redundant grid-aligned outputs. We instead supervise directly with ground-truth 3D Gaussians, freeing the model from any grid structure.

However, a fundamental challenge in directly regressing Gaussian parameters with a feed-forward model is establishing *correspondences* between predicted and ground-truth Gaussians. One solution is permutation-invariant losses (e.g., using Chamfer or Earth Mover's distance), but these are imperfect proxies for reconstruction quality, and they become computationally expensive for large sets of Gaussians. Inspired by Nichol et al. [38], who generate point clouds via diffusion over point coordinates, we sidestep this issue by framing reconstruction as conditional generation via flow matching [29, 32]. Correspondence is thus established *by construction*: each noisy training sample interpolates between a specific ground-truth Gaussian and its paired noise, and the model learns to recover that ground truth, so no explicit assignment between predicted and ground-truth Gaussians is needed. This is not a fixed global ordering: the Gaussian set is randomly initialized at every training iteration and inference run, so the model

cannot exploit any canonical order. In summary, a generative formulation enables detailed and plausible reconstruction in unseen regions and avoids computing expensive permutation-invariant losses to supervise non-grid-aligned Gaussians.

Describing the process in detail, we define a linear probability path between a standard normal prior $\mathcal{N}(0, \mathbf{I})$ and the data distribution of 3D Gaussians, where $\mathbf{I}$ is the identity covariance matrix. Then, given a ground-truth 3D Gaussian $\mathbf{z}_1$ and noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, the interpolated sample at time $t \in [0, 1]$ is $\mathbf{z}_t = (1-t)\,\boldsymbol{\epsilon} + t\,\mathbf{z}_1$. The model $f_{\boldsymbol{\theta}}$ predicts the clean sample directly (x-prediction [25]), yielding $\hat{\mathbf{z}}_1 = f_{\boldsymbol{\theta}}(\mathbf{z}_t, t, \mathcal{I})$, from which the velocity field $\mathbf{v}_t = (\hat{\mathbf{z}}_1 - \mathbf{z}_t)/(1-t)$ is derived. At inference, starting from pure noise, this velocity is integrated via Euler steps $\mathbf{z}_{t+\Delta t} = \mathbf{z}_t + \Delta t \cdot \mathbf{v}_t$, to obtain the final prediction. We use the standard parameterization of 3D Gaussians by mean position, log-scale, quaternion rotation, logit-opacity, and RGB color. We normalize each parameter independently using per-parameter mean and standard deviation computed over the training set.

3.2 Hierarchical Gaussians for Efficient Training

High-quality Gaussian reconstructions typically require more than 50K Gaussians per object, making direct flow matching over the full set prohibitive due to the quadratic cost of self-attention. We address this by organizing the Gaussians into a tree-based hierarchy, where each node represents a Gaussian and each level of the tree yields a progressively finer representation of the object. This allows us to train the model at a level that balances detail with tractability. Furthermore, as the tree structure provides a notion of spatial proximity, it enables us to patchify the sequence [6, 23] by grouping sibling nodes into a single token.

Following Kerbl et al. [21], we construct a tree-based hierarchy over the ground-truth Gaussians of each training object. Each parent node is a weighted combination of its two children, producing a coarser Gaussian; depth l thus yields 2^l Gaussians at a uniform level of detail (Figure 5). Only the Gaussians at a chosen depth serve as model input; parent nodes merely define the hierarchy. This enables a coarse-to-fine curriculum. Our Diffusion Transformer (DiT) operates on Gaussian tokens without positional encodings, and is agnostic to sequence length and order, so a trained model can be fine-tuned on different numbers of Gaussians. Concretely, we first train at depth 11 (2K Gaussians, 224×224 images), then fine-tune at depth 13 (8K Gaussians, 512×512 images), progressively increasing both representational resolution and image-supervision resolution.

To further reduce the sequence length, we exploit the tree structure to patchify the input Gaussians. Patchification groups neighboring elements into a single token and is standard in pixel-space generative models [6, 23]. But it does not directly apply to non-grid-aligned Gaussians, which lack a regular spatial structure. The tree provides a natural substitute: we group pairs of sibling Gaussians into single tokens, *halving* the sequence from $\mathbb{R}^{N \times C}$ to $\mathbb{R}^{(N/2) \times 2C}$, where N is the number of Gaussians at the chosen depth and C=14 is the per-Gaussian parameter dimensionality. Since siblings are spatially proximate, they form a coherent group and provide the transformer with a natural structural prior. A linear layer then maps the patched vectors to the transformer hidden

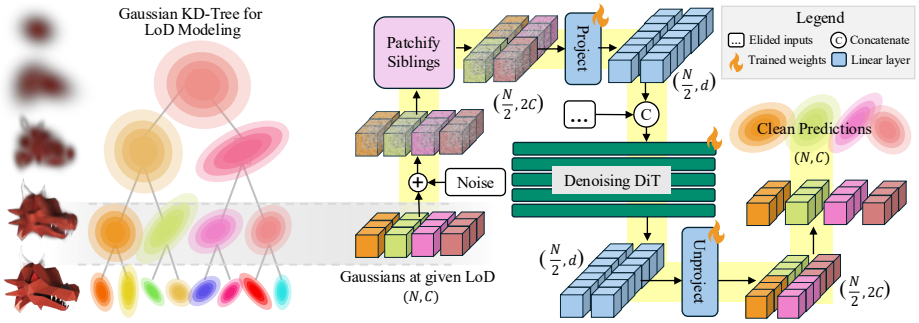

Fig. 5: Patchification with hierarchical Gaussians. A binary tree over the ground-truth Gaussians provides level-of-detail slices at each depth, allowing us to keep the Gaussian count low during training while preserving a faithful object representation. At the chosen depth, pairs of sibling nodes are merged into single transformer tokens, halving the sequence length while preserving spatial locality.

dimension $d = 1024$. At the output, the process is reversed. Thus, we use the hierarchy for (1) selecting LoD slices for coarse-to-fine training, and (2) defining pairs of tokens for patchification.

3.3 Reconstruction-Guided Flow Matching

While improving training efficiency, the LoD slicing described in the previous section provides a coarser signal than the original reconstruction. We address this via three complementary mechanisms to recover visual details in the generated output: (1) a timestep-weighted photometric loss that supervises rendered images once predictions become reliable, (2) photometric gradient guidance that steers Gaussian parameters toward lower rendering error at each denoising step, and (3) classifier-free guidance [15] to amplify the influence of the input views.

Timestep-Weighted Multi-View Rendering Loss. We add photometric supervision to sharpen details beyond what the flow matching loss alone provides. Thus, at each denoising step, we compute the L1 error between renderings of the predicted $\hat{\mathbf{z}}_1$ and both input and ground-truth images. However, when t is close to zero, $\hat{\mathbf{z}}_1$ is unreliable and a photometric loss provides noisy gradients. Therefore, we weight the rendering losses $\mathcal{R}_{\text{Held}}$ and $\mathcal{R}_{\text{Seen}}$, over held-out ground-truth and observed input views, respectively, with a monotonically increasing function

$$w(t) = 50 \cdot \left(\min\left(1, \frac{t}{0.9}\right)\right)^5$$

that concentrates supervision on later stages, when the model's prediction is close to $\mathbf{z}_1$. The combined training loss is:

$$\mathcal{L} = \mathcal{L}_{\text{FM}} + w(t) \cdot \left(\mathcal{R}_{\text{Held}} + \mathcal{R}_{\text{Seen}}\right), \text{ where } \mathcal{L}_{\text{FM}} = \left\| \left(\frac{\hat{\mathbf{z}}_1 - \mathbf{z}_t}{1-t}\right) - (\mathbf{z}_1 - \boldsymbol{\epsilon}) \right\|_2^2.$$

Photometric-Based Reconstruction Guidance. We steer the model toward predictions that better match the observed input views through explicit reconstruction feedback. At each denoising step t, we render the current noisy Gaussians $\mathbf{z}_t$ from the conditioning views, compute the seen-view photometric loss $\mathcal{R}_{\text{Seen}}$, and backpropagate it to obtain per-Gaussian gradients $\nabla_{\mathbf{z}_t}\mathcal{R}_{\text{Seen}}$. We use these gradients in two complementary ways. First, during both training and inference, we concatenate them to the Gaussian features provided to the denoising network. This rendering gradient concatenation gives the model an explicit reconstruction signal indicating how each Gaussian should be adjusted to better match the conditioning views. Second, during inference only, we use the same gradient as external guidance in the Euler update:

$$\mathbf{z}_{t+\Delta t} = \mathbf{z}_t + \Delta t \cdot \mathbf{v}_t(\nabla_{\mathbf{z}_t}\mathcal{R}_{\text{Seen}}) - \lambda_{\text{PG}}\nabla_{\mathbf{z}_t}\mathcal{R}_{\text{Seen}}. \tag{1}$$

Thus, rendering gradient concatenation incorporates reconstruction gradients inside the model as input features, whereas photometric gradient guidance applies them outside the model during sampling. In both cases, the gradients are computed only from the conditioning views; held-out and evaluation views are never used in this pathway. For all experiments, we set $\lambda_{\text{PG}} = 50$.

Classifier-Free Guidance. We use classifier-free guidance [15] to further amplify the influence of the conditioning views. Specifically, we train the model to produce unconditional output by zeroing out the image features with probability 10% during training. Then, at inference time, the guided prediction is: $\hat{\mathbf{z}}_1^{\text{guided}} = \hat{\mathbf{z}}_1^{\text{uncond}} + s \cdot (\hat{\mathbf{z}}_1^{\text{cond}} - \hat{\mathbf{z}}_1^{\text{uncond}})$, where $s > 1$ is the guidance scale.

3.4 Model Architecture

Our model is based on the DiT architecture [42]. The input images $\mathcal{I}$ are processed by a frozen DINOv2-Base [41] encoder, producing patch tokens of dimension 768. These are concatenated along the channel dimension with positional embeddings (32 channels), downsampled RGB values (3 channels), and Plücker ray coordinates (6 channels) of each view. The image tokens are projected to the hidden dimension of 1024 via a linear layer, and concatenated with the patchified noisy Gaussian tokens and 32 learnable embedding tokens acting as registers [9]. The combined sequence is processed by 18 transformer blocks. Each block consists of self-attention with 16 64-dimensional heads, AdaLN [42] conditioning on the timestep t, and a GEGLU feed-forward network. The transformer output is mapped to the input size via a linear projection. QK-normalization [13] via RMSNorm [67] is applied for training stability.

3.5 Training Strategy

We train on 3D Gaussian reconstructions of ∼140K Objaverse objects [10]. Timesteps t follow a cosine schedule [39]. As described in Section 3.2, we use a coarse-to-fine curriculum: first training with 2K Gaussians at 224×224 resolution, then fine-tuning with 8K Gaussians at 512×512. We use AdamW with a learning rate of 10^{-4}, cosine decay to 5×10^{-6}, and 2,000 warmup steps, training for 700K iterations on 64 NVIDIA H200 GPUs for 6 days.

Table 1: Quantitative comparison of partially observed objects. We evaluate the case when four input views are concentrated in a hemisphere, leaving parts of the object unobserved. The values for DreamSim (DS) [12] and DINO similarity are ×100. Our method shows strong performance on all metrics, with an order-of-magnitude fewer Gaussians (#GS). We highlight the metrics in blue, proportional to their percentile.

		Objaverse				GSO			
Method	#GS	PSNR ▲	DINO ▲	DS ▼	FID ▼	PSNR ▲	DINO ▲	DS ▼	FID ▼
LGM [50]	65K	22.88	66.46	17.05	88.43	19.65	58.01	26.41	54.96
LaRa [4]	45K	27.79	71.72	12.39	93.19	25.64	60.20	22.28	69.18
GS-LRM [70]	70K	27.90	77.57	10.33	65.06	27.57	72.20	16.11	34.13
Ours	**8K**	**29.92**	**81.80**	**9.24**	**43.58**	**28.08**	**72.24**	**15.50**	27.69

3.6 Inference

Starting from $\mathbf{z}_{t=0} = \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$, we integrate the learned velocity field with 50 Euler steps (Δt=0.02). At each step, the model predicts $\hat{\mathbf{z}}_1$, from which the velocity is computed and the state updated. Classifier-free guidance and photometric gradient guidance (Section 3.3) are applied at every step. The final $\hat{\mathbf{z}}_1$ is denormalized to obtain the predicted Gaussian parameters $\hat{\mathcal{G}}$, which can be directly rendered via the 3DGS rasterizer. Single object inference takes approximately 26 seconds on a single H200 GPU.

4 Experiments

Training Data. Following ShapeSplat [35], we created a dataset of ∼140K objects from Objaverse. Each object is rendered from 40 views at 512×512, and then reconstructed using 3DGS-MCMC optimization [22]. A hierarchical LoD tree is precomputed for each object using Kerbl et al.'s method [21].

Test Data. We evaluate on two benchmarks: (1) Objaverse [10] (500 held-out objects) processed as described above, and (2) Google Scanned Objects (GSO) [11] (1,030 objects), a dataset of real scanned household items providing an out-of-distribution test of generalization.

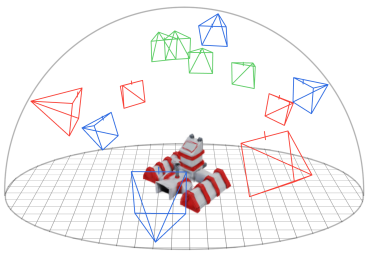

Fig. 6: Full observation, Partial observation, and Validation cameras.

Input View Configurations. We evaluate two input-view settings (Figure 6). *Full observation* follows LaRa [4] by using K-means ($K = 4$) to group the cameras, and sample one view per cluster. This ensures sufficient angular coverage of the object. *Partial*

Table 2: Quantitative comparison of fully observed objects. We evaluate the case in which four input views provide full coverage of the object. The values for SSIM and LPIPS are ×100 and ×1000, respectively. Our method achieves competitive quality with grid-aligned methods while generating an order-of-magnitude fewer Gaussians (#GS). We highlight the metrics in blue, proportional to their percentile.

		Objaverse			GSO		
Method	#GS	PSNR ▲	SSIM ▲	LPIPS ▼	PSNR ▲	SSIM ▲	LPIPS ▼
LGM [50]	65K	26.65	95.50	57.71	23.64	92.35	73.61
LaRa [4]	45K	**31.91**	97.12	43.60	29.15	**95.60**	60.70
GS-LRM [70]	70K	31.03	97.10	**32.80**	31.13	95.00	**30.93**
Ours	**8K**	31.66	**97.14**	53.15	**31.49**	95.13	77.25

observation draws all four views from a single cluster, concentrating observations in one region and leaving other parts of the object unobserved. This setting is specifically designed to test how methods handle unobserved regions. For both input settings, we use the same fixed validation view, sampled to provide broad 360° object coverage; this set is similar to the full-observation pattern but not identical.

Baselines. We compare against four feed-forward reconstruction methods spanning the pixel-aligned, volumetric, and generative paradigms: LGM [50], a U-Net predicting pixel-aligned Gaussians from four input views; GS-LRM [70], a pixel-aligned transformer with Plücker ray embeddings; LaRa [4], a voxel-based method trained with multi-view rendering loss; and ReconViaGen [2], a generative reconstruction method. As the source code for GS-LRM is not released we use our re-implementation which matches or slightly exceeds the reported results [11]).

4.1 Results

We present quantitative results on both benchmark datasets in Table 1 and Table 2; qualitative comparisons are shown in Figures 7 and 8. For full observation, we report standard pixel-level metrics (PSNR, SSIM, LPIPS); for partial observation, where unobserved regions dominate, we use perceptual and distributional metrics (DINO, DreamSim, FID) that better capture the quality of hallucinated content. Crucially, while the perceptual and distributional metrics reward plausible completions, conditioning fidelity is captured by PSNR in both settings, since our fixed validation-view set spans viewpoints both close to and far from the conditioning images, covering interpolation as well as extrapolation.

Full Observation. Under full observation (Table 2), our method performs on par with the baselines while using only 8K Gaussians. Notably, this is an order-of-magnitude fewer Gaussians than all other baselines (45K–500K).

Partial Observation. The advantage of our approach is most pronounced under partial observation (Table 1 and Figure 7), where input views are concentrated

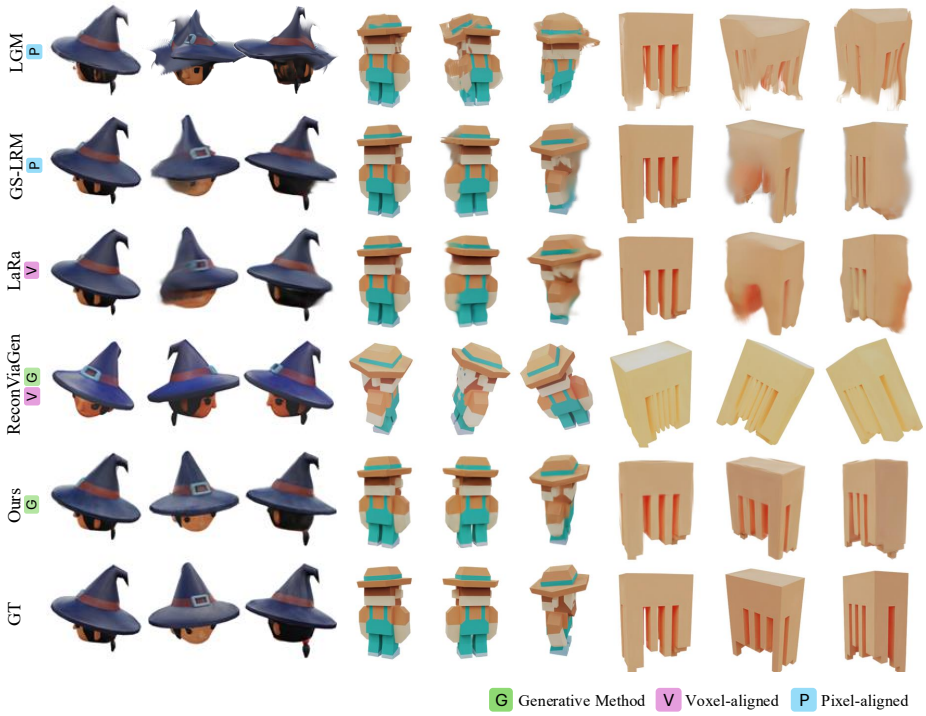

Fig. 7: Qualitative comparison on the Objaverse dataset [10]. The pixel-aligned LGM [50] and GS-LRM [70] produce severe artifacts and holes in unobserved regions, while LaRa [4] yields blurry completions. ReconViaGen's generative reconstruction [2] synthesizes realistic content but is misaligned with the input views. Our method produces sharp, input-consistent reconstructions for both observed and unobserved regions.

on one side, leaving large parts of the object unobserved. Here, our method outperforms all baselines on most metrics across both datasets, including perceptual (DreamSim [12], DINOv2-feature [41]) and distributional (FID [14]) measures. The strong FID scores reflect a key benefit of the generative formulation: our model produces perceptually realistic completions rather than blurry conditional averages. Pixel-aligned methods (LGM, GS-LRM) degrade significantly, as they cannot place Gaussians in unobserved regions. LaRa, although not pixel-aligned, yields the worst FID due to its reliance on multi-view rendering loss alone.

Concurrent pose-free baseline. ReconViaGen [2] is a concurrent method that operates in a pose-free setting. Qualitatively, ReconViaGen often produces realistic completions in unobserved regions, but in our comparisons it is less reliably aligned with the conditioning views (Figure 7).

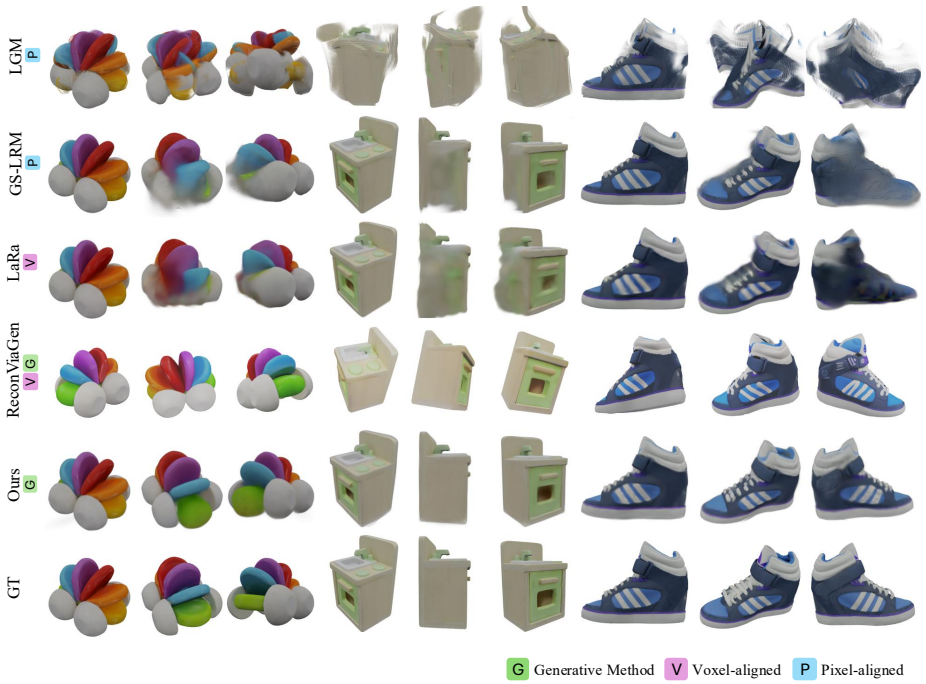

Fig. 8: Qualitative comparison on the GSO dataset. Additional examples comparing the input-fidelity and completeness of our method with the baselines.

4.2 Ablations

Table 3 ablates the key components of our method on GSO under both observation settings. Removing flow matching, i.e. replacing the DiT with a vanilla transformer and learnable latent tokens trained with photometric loss only (Figure 3), causes the largest quality drop, confirming the importance of the generative formulation. The timestep-weighted multi-view rendering loss (Section 3.3) and coarse-to-fine curriculum training (Section 3.2) are also significant; skipping the coarser 2K stage and training directly at 8K Gaussians degrades results. Removing patchification and processing all 8K Gaussians as individual tokens hurts performance because the doubled sequence length reduces training efficiency. Conversely, halving the count to 4K Gaussians (matching our token budget but without patchification) also degrades quality due to reduced capacity. Patchification thus retains high local representational capacity while halving the sequence length. Replacing our LoD tree-based patchification with random groupings further degrades results, confirming that spatially coherent pairings provide a useful inductive bias. At each step we concatenate the gradient of the rendering loss with respect to the Gaussian parameters $\nabla_{\mathbf{z}}\mathcal{R}_{\text{Seen}}$ to the noisy inputs. Removing this step also reduces performance, showing that explicit gradient cues improve input reconstruction quality and input fidelity.

Table 3: Ablation study. PSNR (dB) on GSO (1,030 objects) under full and partial observation, split into training-time and test-time ablations.

Configuration	Full (PSNR ▲)	Partial (PSNR ▲)
Full method (Ours)	31.49	28.08
Training-time ablations		
No Flow Matching	28.40 (–3.09)	26.30 (–1.78)
No MV Rendering Loss	29.66 (–1.83)	26.87 (–1.21)
No LoD Curriculum Training	30.00 (–1.49)	26.42 (–1.66)
Without Patching (4K GS)	30.47 (–1.02)	27.22 (–0.86)
Without Patching (8K GS)	30.54 (–0.95)	26.75 (–1.33)
With Random Patching	30.84 (–0.65)	27.16 (–0.92)
No Rendering Gradient Concatenation	30.89 (–0.60)	27.51 (–0.57)
Test-time ablations		
No Classifier-Free Guidance	31.30 (–0.20)	27.50 (–0.60)
No Photometric Gradient Guidance	31.10 (–0.40)	27.90 (–0.20)

Fig. 9: Qualitative Ablation results. Using a generative model as the core predictor, a timestep-weighted rendering loss, and coarse-to-fine curriculum training prove the most significant contributors to the quality of our final results.

Number of Input Views. To study robustness to the number of input views, we fine-tune a single model with the number of conditioning views sampled uniformly from 2 to 9 and evaluate this same model at each view count, avoiding any train-test mismatch. Table 4 together with the supplementary qualitative results therefore measure how well the model adapts across varying numbers of inputs. Increasing from two to four views provides the largest gain (+3.3 dB), as the added viewpoints resolve geometric ambiguities. Beyond this, returns diminish: six and nine views add +0.5 dB and +0.9 dB respectively, indicating that four views already cover most of the visible surface. Nonetheless, our method consistently benefits from additional views when available.

Table 4: Number of input views. PSNR (dB) on GSO, full observations.

# Views	PSNR ▲
2	28.22
4	31.49
6	32.04
9	32.39

5 Conclusion

We presented *Free-Range Gaussians*, a non-grid-aligned approach to sparse-view 3D Gaussian reconstruction via flow matching. Formulating reconstruction as conditional generation over Gaussian parameters overcomes the limitations of pixel-aligned methods (holes in unobserved regions), generate-and-reconstruct pipelines (dependence on synthesized-view placement), and volumetric methods (blurry conditional means from multi-view losses). Our hierarchical LoD patching scheme groups sibling Gaussians into joint tokens for efficient training, and reconstruction-guided flow matching yields accurate results despite a moderate budget of Gaussians. Experiments on Objaverse and Google Scanned Objects show consistent improvements over pixel-aligned (LGM [50], GS-LRM [70]) and voxel-based (LaRa [4]) methods, particularly under partial observation.

Limitations and Future Work. While our approach establishes a strong baseline, several avenues exist to further expand its capabilities. Our current 8K Gaussian budget ensures tractable training but limits fine detail. This can be addressed by adapting efficient attention mechanisms such as sparse [8] or linear-complexity [18] attention to scale to larger Gaussian counts, or by using our method as a coarse generator followed by an upsampling stage. Our iterative denoising (50 steps) is slower than single-pass feed-forward methods; distillation [46] or flow straightening [33] can reduce the step count while preserving quality. Our results also depend on the quality of the fitted Gaussian supervision, since heavily occluded objects can produce ground-truth Gaussians with missing geometry or artifacts. Still, these artifacts are not systematic across the ∼140K training objects, and our timestep-weighted photometric supervision (Section 3.3) helps keep predictions consistent with the observed images; extending the method with pose estimation could further enable pose-free and single-view reconstruction. Finally, operating directly in the 3D Gaussian parameter space opens avenues beyond reconstruction, including shape completion, semantic editing, and matching, bringing the versatility of 2D generative modeling into 3D.

References

1. Bahmani, S., Shen, T., Ren, J., Huang, J., Jiang, Y., Turki, H., Tagliasacchi, A., Lindell, D.B., Gojcic, Z., Fidler, S., Ling, H., Gao, J., Ren, X.: Lyra: Generative 3D scene reconstruction via video diffusion model self-distillation. In: ICLR (2026)
2. Chang, J., Ye, C., Wu, Y., Chen, Y., Zhang, Y., Luo, Z., Li, C., Zhi, Y., Han, X.: ReconViaGen: Towards accurate multi-view 3D object reconstruction via generation. In: ICLR (2026)
3. Charatan, D., Li, S., Tagliasacchi, A., Sitzmann, V.: pixelSplat: 3D Gaussian splats from image pairs for scalable generalizable 3D reconstruction. In: CVPR (2024)
4. Chen, A., Xu, H., Esposito, S., Tang, S., Geiger, A.: LaRa: Efficient large-baseline radiance fields. In: ECCV (2024)
5. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo. In: ICCV (2021)
6. Chen, S., Ge, C., Zhang, S., Sun, P., Luo, P.: PixelFlow: Pixel-space generative models with flow (2025), arXiv:2504.07963

7. Chen, Y., Xu, H., Zheng, C., Zhuang, B., Pollefeys, M., Geiger, A., Cham, T.J., Cai, J.: MVSplat: Efficient 3D Gaussian splatting from sparse multi-view images. In: ECCV (2024)
8. Child, R., Gray, S., Radford, A., Sutskever, I.: Generating long sequences with sparse transformers (2019), arXiv:1904.10509
9. Darcet, T., Oquab, M., Mairal, J., Bojanowski, P.: Vision transformers need registers. In: ICLR (2024)
10. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3D objects. In: CVPR (2023)
11. Downs, L., Francis, A., Koenig, N., Kinman, B., Hickman, R., Reymann, K., McHugh, T.B., Vanhoucke, V.: Google Scanned Objects: A high-quality dataset of 3D scanned household items. In: ICRA (2022). https://doi.org/10.1109/ICRA46639.2022.9811809
12. Fu, S., Tamir, N., Sundaram, S., Chai, L., Zhang, R., Dekel, T., Isola, P.: DreamSim: Learning new dimensions of human visual similarity using synthetic data. In: NeurIPS (2023)
13. Henry, A., Dachapally, P.R., Pawar, S.S., Chen, Y.: Query-key normalization for transformers. In: Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 4246–4253 (2020)
14. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: NIPS (2017)
15. Ho, J., Salimans, T.: Classifier-free diffusion guidance. In: NeurIPS Workshops (2021)
16. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: LRM: Large reconstruction model for single image to 3D. In: ICLR (2024)
17. Jiang, L., Mao, Y., Xu, L., Lu, T., Ren, K., Jin, Y., Xu, X., Yu, M., Pang, J., Zhao, F., Lin, D., Dai, B.: AnySplat: Feed-forward 3D Gaussian splatting from unconstrained views. ACM Trans. Graph. (2025)
18. Katharopoulos, A., Vyas, A., Pappas, N., Fleuret, F.: Transformers are RNNs: Fast autoregressive transformers with linear attention. In: International conference on machine learning. pp. 5156–5165. PMLR (2020)
19. Keetha, N., Müller, N., Schönberger, J., Porzi, L., Zhang, Y., Fischer, T., Knapitsch, A., Zauss, D., Weber, E., Antunes, N., Luiten, J., Lopez-Antequera, M., Bulò, S.R., Richardt, C., Ramanan, D., Scherer, S., Kontschieder, P.: MapAnything: Universal feed-forward metric 3D reconstruction. In: 3DV (2026)
20. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. **42**(4), 139:1–14 (2023). https://doi.org/10.1145/3592433
21. Kerbl, B., Meuleman, A., Kopanas, G., Wimmer, M., Lanvin, A., Drettakis, G.: A hierarchical 3D Gaussian representation for real-time rendering of very large datasets. ACM Trans. Graph. **43**(4), 62:1–15 (2024). https://doi.org/10.1145/3658160
22. Kheradmand, S., Rebain, D., Sharma, G., Sun, W., Tseng, J., Isack, H., Kar, A., Tagliasacchi, A., Yi, K.M.: 3D Gaussian splatting as Markov chain Monte Carlo. In: NeurIPS (2024)
23. Kim, D., Ghadiyaram, D., Gadde, R.: DDiT: Dynamic patch scheduling for efficient diffusion transformers. In: CVPR (2026)
24. Leroy, V., Cabon, Y., Revaud, J.: Grounding image matching in 3D with MASt3R. In: ECCV (2024)

25. Li, T., He, K.: Back to basics: Let denoising generative models denoise (2025), arXiv:2511.13720
26. Li, Y., Zou, Z.X., Liu, Z., Wang, D., Liang, Y., Yu, Z., Liu, X., Guo, Y.C., Liang, D., Ouyang, W., Cao, Y.P.: TripoSG: High-fidelity 3D shape synthesis using large-scale rectified flow models. IEEE Trans. Pattern Anal. Mach. Intell. (2026). https://doi.org/10.1109/TPAMI.2025.3633512
27. Liang, H., Cao, J., Goel, V., Qian, G., Korolev, S., Terzopoulos, D., Plataniotis, K.N., Tulyakov, S., Ren, J.: Wonderland: Navigating 3D scenes from a single image. In: CVPR (2025)
28. Lin, C., Pan, P., Yang, B., Li, Z., Mu, Y.: DiffSplat: Repurposing image diffusion models for scalable Gaussian splat generation. In: ICLR (2025)
29. Lipman, Y., Chen, R.T.Q., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: ICLR (2023)
30. Liu, M., Shi, R., Chen, L., Zhang, Z., Xu, C., Wei, X., Chen, H., Zeng, C., Gu, J., Su, H.: One-2-3-45++: Fast single image to 3D objects with consistent multi-view generation and 3D diffusion. In: CVPR (2024)
31. Liu, R., Wu, R., Hoorick, B.V., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3D object. In: ICCV (2023)
32. Liu, X., Gong, C., qiang liu: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: ICLR (2023)
33. Liu, X., Zhang, X., Ma, J., Peng, J., et al.: InstaFlow: One step is enough for high-quality diffusion-based text-to-image generation. In: ICLR (2024)
34. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: SyncDreamer: Generating multiview-consistent images from a single-view image. In: ICLR (2024)
35. Ma, Q., Li, Y., Ren, B., Sebe, N., Konukoglu, E., Gevers, T., Gool, L.V., Paudel, D.P.: ShapeSplat: A large-scale dataset of Gaussian splats and their self-supervised pretraining. In: 3DV (2025)
36. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020). https://doi.org/10.1007/978-3-030-58452-8_24
37. Mu, Y., Zuo, X., Guo, C., Wang, Y., Lu, J., Wu, X., Xu, S., Dai, P., Yan, Y., Cheng, L.: GSD: View-guided Gaussian splatting diffusion for 3D reconstruction. In: ECCV (2024)
38. Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., Chen, M.: Point-E: A system for generating 3D point clouds from complex prompts (2022), arXiv:2212.08751
39. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: International conference on machine learning. pp. 8162–8171. PMLR (2021)
40. van den Oord, A., Vinyals, O., Kavukcuoglu, K.: Neural discrete representation learning. In: NeurIPS (2017)
41. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning robust visual features without supervision. Transactions on Machine Learning Research (2024)
42. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: ICCV (2023)
43. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: DreamFusion: Text-to-3D using 2D diffusion. In: ICLR (2023)
44. Ren, X., Lu, Y., Liang, H., Wu, Z., Ling, H., Chen, M., Fidler, S., Williams, F., Huang, J.: SCube: Instant large-scale scene reconstruction using VoxSplats. In: NeurIPS (2024)

45. Roessle, B., Müller, N., Porzi, L., Bulò, S.R., Kontschieder, P., Dai, A., Nießner, M.: L3DG: Latent 3D Gaussian diffusion. In: SIGGRAPH Asia (2024)
46. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. In: ICLR (2022)
47. Smart, B., Zheng, C., Laina, I., Prisacariu, V.A.: Splatt3R: Zero-shot Gaussian splatting from uncalibrated image pairs (2024), arXiv:2408.13912
48. Szymanowicz, S., Rupprecht, C., Vedaldi, A.: Splatter image: Ultra-fast single-view 3D reconstruction. In: CVPR (2024)
49. Szymanowicz, S., Zhang, J.Y., Srinivasan, P., Gao, R., Brussee, A., Holynski, A., Martin-Brualla, R., Barron, J.T., Henzler, P.: Bolt3D: Generating 3D scenes in seconds. In: ICCV (2025)
50. Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., Liu, Z.: LGM: Large multi-view Gaussian model for high-resolution 3D content creation. In: ECCV (2024)
51. Wang, J., Chen, M., Karaev, N., Vedaldi, A., Rupprecht, C., Novotny, D.: VGGT: Visual geometry grounded transformer. In: CVPR (2025)
52. Wang, P., Tan, H., Bi, S., Xu, Y., Luan, F., Sunkavalli, K., Wang, W., Xu, Z., Zhang, K.: PF-LRM: Pose-free large reconstruction model for joint pose and shape prediction. In: ICLR (2024)
53. Wang, Q., Wang, Z., Genova, K., Srinivasan, P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.: IBRNet: Learning multi-view image-based rendering. In: CVPR (2021)
54. Wang, S., Leroy, V., Cabon, Y., Chidlovskii, B., Revaud, J.: DUSt3R: Geometric 3D vision made easy. In: CVPR (2024)
55. Wang, W., Chen, Y., Zhang, Z., Liu, H., Wang, H., Feng, Z., Qin, W., Zhu, Z., Chen, D.Y., Zhuang, B.: VolSplat: Rethinking feed-forward 3D Gaussian splatting with voxel-aligned prediction. In: ECCV (2026), arXiv:2509.19297
56. Wei, X., Zhang, K., Bi, S., Tan, H., Luan, F., Deschaintre, V., Sunkavalli, K., Su, H., Xu, Z.: MeshLRM: Large reconstruction model for high-quality meshes (2024), arXiv:2404.12385
57. Wu, C.H., Chen, Y.C., Solarte, B., Yuan, L., Sun, M.: iFusion: Inverting diffusion for pose-free reconstruction from sparse views. In: 3DV (2025)
58. Wu, R., Mildenhall, B., Henzler, P., Park, K., Gao, R., Watson, D., Srinivasan, P.P., Verbin, D., Barron, J.T., Poole, B., Holynski, A.: ReconFusion: 3D reconstruction with diffusion priors. In: CVPR (2024)
59. Wu, S., Lin, Y., Zhang, F., Zeng, Y., Xu, J., Torr, P., Cao, X., Yao, Y.: Direct3D: Scalable image-to-3D generation via 3D latent diffusion transformer. In: NeurIPS (2024)
60. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3D latents for scalable and versatile 3D generation. In: CVPR (2025)
61. Xu, H., Peng, S., Wang, F., Blum, H., Barath, D., Geiger, A., Pollefeys, M.: DepthSplat: Connecting Gaussian splatting and depth. In: CVPR (2025)
62. Xu, J., Cheng, W., Gao, Y., Wang, X., Gao, S., Shan, Y.: InstantMesh: Efficient 3D mesh generation from a single image with sparse-view large reconstruction models (2024), arXiv:2404.07191
63. Xu, Y., Shi, Z., Yifan, W., Chen, H., Yang, C., Peng, S., Shen, Y., Wetzstein, G.: GRM: Large Gaussian reconstruction model for efficient 3D reconstruction and generation. In: ECCV (2024)
64. Yang, C., Li, S., Fang, J., Liang, R., Xie, L., Zhang, X., Shen, W., Tian, Q.: GaussianObject: Just taking four images to get a high-quality 3D object with Gaussian splatting. ACM Trans. Graph. (2024)

65. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: CVPR (2021)
66. Zhang, B., Feng, J., Tang, H., Chen, Z.F., Li, X., Metaxas, D.: 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. In: ACM TOG (2023)
67. Zhang, B., Sennrich, R.: Root mean square layer normalization. In: NeurIPS (2019)
68. Zhang, B., Tang, J., Nießner, M., Wonka, P.: 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. ACM Trans. Graph. **42**(4), 92:1–16 (2023). https://doi.org/10.1145/3592442
69. Zhang, B., Cheng, Y., Yang, J., Wang, C., Zhao, F., Tang, Y., Chen, D., Guo, B.: GaussianCube: A structured and explicit radiance representation for 3D generative modeling. In: NeurIPS (2024)
70. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: GS-LRM: Large reconstruction model for 3D Gaussian splatting. In: ECCV (2024)
71. Ziwen, C., Tan, H., Zhang, K., Bi, S., Luan, F., Hong, Y., Fuxin, L., Xu, Z.: Long-LRM: Long-sequence large reconstruction model for wide-coverage Gaussian splats. In: ICCV (2025)