TRIFOLD: FOLDING IRREGULAR 3D GAUSSIANS INTO STRUCTURED LATENTS FOR 3D SCENE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent progress in 3D generation has motivated growing interest in compact generative representations of complex 3D scenes. For native 3D Gaussian scenes, however, learning such representations remains challenging due to widely varying primitive counts, nonuniform spatial density, and differences in geometry and appearance even among spatially neighboring primitives. In this work, we introduce **TriFold**, a native 3D Gaussian VAE that encodes Gaussian scenes into a fixed-layout triplane latent and jointly reconstructs their geometry and appearance. The key insight is to impose structural regularity on the latent space while preserving the spatial adaptivity of the underlying Gaussians. During encoding, we use cell-wise learned queries to aggregate projected Gaussians into multiple slots per cell, reducing the information loss caused by single-feature pooling. During decoding, occupancy prediction identifies active regions, where query-based set prediction and candidate selection reconstruct local Gaussian sets with spatially varying primitive counts. Building on this representation, we train a latent flow-matching model for conditional scene-level 3DGS generation, enabling point-cloud-conditioned synthesis and single-image-conditioned amodal generation. Experiments demonstrate that TriFold achieves high-fidelity reconstruction and generation from a single image that provides only a partial observation of the scene, validating its effectiveness for native 3D Gaussian representation learning.

1 INTRODUCTION

Amodal 3D scene generation aims to synthesize complete environments from partial observations, enabling exploration and embodied interaction beyond captured views (DeVries et al., 2021; Bautista et al., 2022; Meng et al., 2025). This requires a representation that captures complex 3D structure while remaining tractable for generative modeling. Recent advances increasingly rely on structured latent representations that reorganize irregular 3D signals into formats compatible with generative models (Wang et al., 2023; Li et al., 2025; Team, 2025; Xiang et al., 2025; 2026). However, extending this paradigm to native scene-level 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) poses distinct representation challenges. Scene-level Gaussian distributions are heterogeneous: globally, scenes can differ by orders of magnitude in primitive count and exhibit highly nonuniform spatial density; locally, spatially adjacent Gaussians suffer from severe attribute heterogeneity: neighboring primitives often differ sharply in geometry and appearance (Lee et al., 2025). These variations motivate a latent representation that is structured for generative modeling while preserving scene-specific Gaussian cardinality, spatial distributions, and fine-grained local attributes.

Existing methods reconcile native Gaussian scenes with latent generative modeling through diverse structural paradigms. Quantized-grid methods, such as L3DG (Roessle et al., 2024) and GaussianGPT (von Lützow et al., 2026), discretize Gaussians into voxel-aligned representations with predefined local capacity before vector quantization. Such fixed per-voxel budgets can underrepresent dense regions while wasting capacity in sparse ones. Canonical-token methods instead regularize globally. Can3Tok (Gao et al., 2025) resamples each scene into a fixed-cardinality primitive set, inadvertently imposing a uniform representation budget irrespective of structural complexity. Sparse latent methods offer topological flexibility by anchoring features to dynamic active voxels. TRELLIS (Xiang et al., 2025), for example, constructs sparse latents and decodes geometry via a cascaded pipeline. While highly effective for bounded, object-centric assets with comprehensive visual coverage, extending this paradigm to expansive scene-level 3DGS is non-trivial: complex occlusions

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.

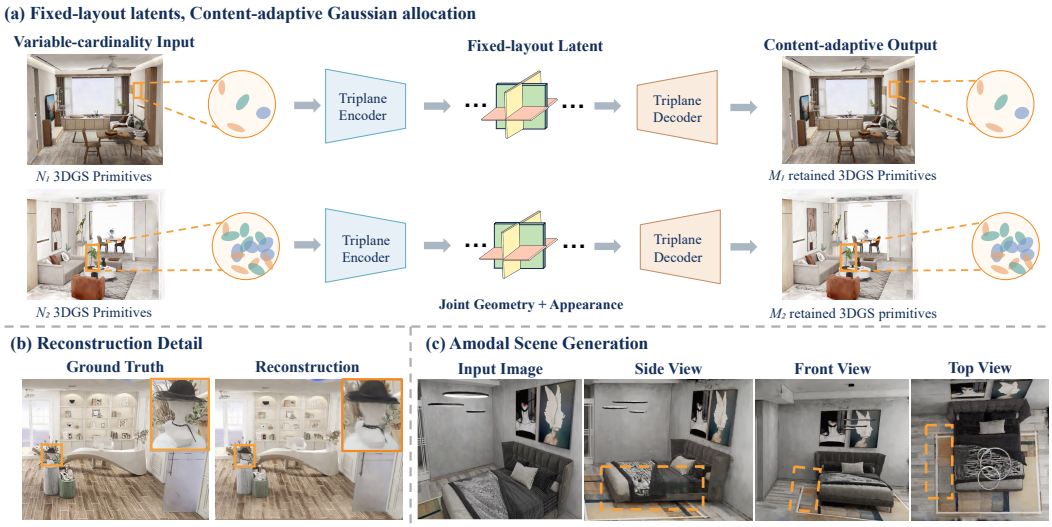


Figure 1: **Overview of TriFold.** (a) TriFold encodes variable-cardinality Gaussian scenes into fixed-layout triplane latents that jointly represent geometry and appearance, and decodes content-adaptive Gaussian sets with variable local densities. (b) The learned representation supports detailed scene reconstruction. (c) Our single-image-conditioned amodal generation preserves observed content while inferring plausible geometry and appearance for unobserved regions.

across broad scenes complicate exhaustive multiview feature anchoring, while scaling sparse voxel grids and multi-stage pipelines over unconstrained volumes introduces notable computational and representation overhead. Finally, continuous-field methods, such as DiffGS (Zhou et al., 2024), reparameterize Gaussians as continuous spatial fields via feature interpolation; yet, this bias toward spatial smoothness can conflict with the high-frequency variations in attributes such as scale and rotation across neighboring primitives. Consequently, an open challenge remains: how can we provide generative backbones with a regular, fixed-layout latent space, while preserving the unconstrained, variable-length nature of native Gaussian representations at both the input and output?

To bridge this gap, we present **TriFold**, a variational autoencoder that operates directly on native 3D Gaussian primitives for scene-level generative modeling. Our key design principle is to impose structural regularity in the latent space that jointly represents geometry and appearance, and preserve the flexible, content-adaptive organization of native Gaussian primitives. Specifically, the encoder maps variable-cardinality, non-uniformly distributed Gaussian sets into fixed-layout triplane latents. It first extracts context-aware primitive features with a point transformer. Since multiple features may project to the same triplane cell, we use multiple learned queries per cell to aggregate local information. Then, we use Cross-plane attention to integrate information across planes through shared Gaussian projections, followed by a variational bottleneck that produces triplane latents.

To recover native 3DGS from this structured latent, our decoder adopts a set-based decoding paradigm without rigid 3D discretization. An occupancy predictor first localizes active regions, while an optimal-transport-guided set decoder dynamically predicts and filters Gaussian candidates within each region. Because the predefined query count serves only as a local capacity upper bound rather than a fixed allocation, the spatial allocation and primitive cardinality remain adaptive to scene content. We supervise Gaussian centers through optimal-transport matching, while the remaining geometric and appearance attributes are constrained by a full-scene rendering loss. Together, these objectives enable accurate Gaussian reconstruction with high rendering fidelity, providing a regular tensor interface for generative modeling while retaining the flexible organization of native 3DGS.

Building on this structured latent space, we train a latent flow-matching model for conditional scene-level 3DGS generation. We consider two conditioning settings: point-cloud-conditioned synthesis from complete spatial observations and single-image-conditioned amodal generation. In the latter, a single image provides only a partial observation of the scene, requiring the learned scene priors to preserve observed content while inferring plausible geometry and appearance in unobserved regions. Experiments demonstrate high-fidelity reconstruction and generation in both settings, demonstrating the effectiveness of structured triplane latents for native 3D Gaussian scene generation.

In summary, our main contributions are:

- **A native scene-level 3D Gaussian variational autoencoder.** We introduce TriFold, which maps variable-cardinality Gaussian scenes into a fixed-layout triplane latent and decodes them back into content-adaptive native 3DGS.
- **Unified geometry-appearance modeling.** Our unified representation jointly encodes scene geometry and appearance, and we reconstruct both from the same representation through an occupancy predictor and a query-based set decoder.
- **Conditional scene-level 3DGS generation based on TriFold.** We train a latent flow-matching model on the learned triplane latents, enabling both point-cloud-conditioned synthesis and single-image-conditioned amodal generation with plausible geometry and appearance in unobserved regions.

2 Related Work

3D Representations and Generative Models. The choice of scene representation shapes the fidelity and scalability of 3D generative models. Neural radiance fields represent scenes as continuous functions and support high-quality novel-view synthesis, but require repeated neural evaluations during rendering (Mildenhall et al., 2020). Meshes represent explicit surfaces through vertices and faces, while voxel grids discretize space into volumetric cells storing occupancy or other attributes, with dense grids incurring cubic memory growth with resolution. Representative generative approaches include π-GAN for implicit radiance fields (Chan et al., 2021), SDFusion for volumetric SDF grids (Cheng et al., 2023), GET3D and MeshGPT for explicit meshes (Gao et al., 2022; Siddiqui et al., 2024), and 3DShape2VecSet, Michelangelo, and TripoSG for generation in learned 3D latent spaces (Zhang et al., 2023; Zhao et al., 2023; Li et al., 2025).

These properties make 3DGS attractive for generative modeling. Native 3DGS, however, represents each scene as an unordered set whose cardinality and spatial density adapt to scene content. This organization is mismatched with latent diffusion and flow-matching backbones, which typically operate on fixed-shape tensors (Rombach et al., 2022; Esser et al., 2024). Triplanes offer a regular layout with a practical balance between spatial expressiveness and computational efficiency and have been widely used for 3D-aware generation (Chan et al., 2022; Wu et al., 2024a). The central challenge is therefore to map adaptive Gaussian sets into such fixed-layout latents while retaining their scene-dependent cardinality and spatial density during reconstruction.

Latent Representations for 3D Gaussians. We focus on methods that regularize Gaussian representations or construct generative 3D latents with Gaussian decoders. This differs from image-conditioned 3DGS methods, which directly predict Gaussian representations from one or more images for reconstruction or content creation (Charatan et al., 2024; Zou et al., 2024; Tang et al., 2024; Xu et al., 2024). Within this generative latent setting, existing approaches differ primarily in how they impose structural regularity. Grid-based methods impose explicit spatial organization: GaussianCube rearranges a fixed number of Gaussians onto voxel grids through optimal transport, whereas L3DG and GaussianGPT adopt voxel-aligned Gaussian representations with vector-quantized autoencoders (Zhang et al., 2024; Roessle et al., 2024; von Lützow et al., 2026). Canonical-token methods such as Can3Tok convert a scene into a fixed-cardinality token set (Gao et al., 2025). Sparse approaches retain scene-dependent spatial support: TRELLIS constructs sparse latent features from multiview observations and generates spatial structure and latent features in two stages, whereas GS-Voxel reorganizes fitted Gaussian scenes into sparse voxel structures with scene-dependent support (Xiang et al., 2025; Qian et al., 2026). Continuous-field methods such as DiffGS decode Gaussian attributes from interpolated spatial features (Zhou et al., 2024).

These approaches enable Gaussian-based generation through different constraints on scene representation, including fixed global or local capacity, occupancy-dependent support, and continuous functional parameterization. TriFold instead places structural regularity in a fixed-layout triplane latent. It directly encodes fitted Gaussian sets with scene-dependent cardinality and density, jointly represents geometry and appearance, and decodes native Gaussian primitives with adaptive spatial allocation and local counts. This design provides a regular tensor interface for generative modeling while retaining the adaptive organization of native 3DGS.

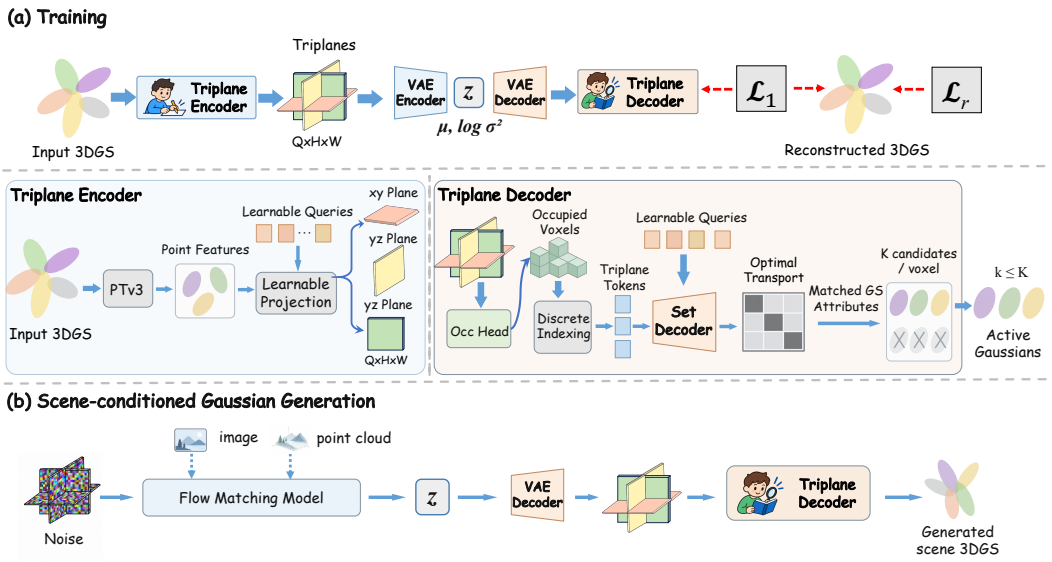


Figure 2: Overview of TriFold. The triplane encoder maps a variable-cardinality Gaussian set to a fixed-layout variational latent, and the triplane decoder reconstructs spatially adaptive local Gaussian sets. The learned latent further supports image-conditioned scene generation with flow matching.

3 METHOD

3.1 UNIFIED GEOMETRY–APPEARANCE GAUSSIAN LATENT SPACE

A 3DGS scene is an unordered, variable-cardinality set $\mathcal{G} = \{g_i\}_{i=1}^{N}$, where $g_i = (x_i, \ell_i, q_i, \alpha_i, \mathbf{c}_i)$ denotes position, log-scale, rotation, opacity, and spherical-harmonic appearance. We learn a Gaussian-to-triplane VAE,

$$\mathbf{z} \sim q_\phi(\mathbf{z} \mid \mathcal{G}), \qquad \widehat{\mathcal{G}} = D_\psi(\mathbf{z}), \qquad \mathbf{z} \in \mathbb{R}^{3C_z \times Q \times R_p \times R_p}. \tag{1}$$

The difficulty is asymmetric: several spatially distinct Gaussians can project to one plane cell, while one decoded cell can correspond to zero, one, or many Gaussians. Our encoder therefore retains multiple features per projected cell, and our decoder predicts a local Gaussian set only where the scene is occupied. The fixed latent jointly represents geometry and appearance while the decoded set remains spatially adaptive. We use $Q = 4$, $C_z = 16$, and $R_p = 256$.

3.2 TRIPLANE ENCODER

We use a modified PTv3 backbone (Wu et al., 2024b), augmented with 3D rotary positional encoding, to extract a context-aware feature $\mathbf{h}_i$ for each Gaussian g_i. For each plane $p \in \{xy, xz, yz\}$, we project the center of Gaussian onto the plane, assign the projection to grid cell $\mathbf{j}_i^p$, and concatenate $\mathbf{h}_i$ with its continuous projected coordinates, yielding the plane-conditioned feature $\widetilde{\mathbf{h}}_i^p$. For any cell $\mathbf{j}$, let $\mathcal{I}_\mathbf{j}^p = \{i : \mathbf{j}_i^p = \mathbf{j}\}$ denote the indices of Gaussians projected to cell $\mathbf{j}$. Rather than pooling these features into a single descriptor, a query-based set aggregator produces Q feature slots:

$$\mathbf{T}_\mathbf{j}^p = \mathrm{Agg}_Q\big(\{\widetilde{\mathbf{h}}_i^p\}_{i \in \mathcal{I}_\mathbf{j}^p}\big) \in \mathbb{R}^{Q \times C}. \tag{2}$$

Within the aggregator, local geometric contextualization captures the relational topology among the projected features, after which Q learned queries adaptively extract these features into discrete slots independently on each plane. The spatial projections of each primitive inherently induce geometric correspondences across the xy, xz, and yz cells, enabling a sparse cross-plane attention module to propagate contextual information across corresponding slots prior to plane densification. Finally, a convolutional variational bottleneck predicts $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ and samples $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$. This yields a regular tensor interface without collapsing multiple Gaussians projected to the same cell into a single feature. Architectural details are provided in Appendix A.2.

3.3 TRIPLANE DECODER

The decoder reverses this mapping in two stages: support localization and Gaussian reconstruction. It first samples and aggregates features from the three planes at each voxel center $\mathbf{x}_v$, yielding context features $\mathbf{C}_v$, predicting occupancy $\rho_v = f_{\text{occ}}(\mathbf{C}_v, \mathbf{x}_v)$. Only voxels in $\widehat{\mathcal{V}} = \{v \mid \rho_v > \tau_{\text{occ}}\}$ are decoded during inference. For each selected voxel, a DETR-style set decoder (Carion et al., 2020) uses K learned queries conditioned on $\mathbf{C}_v$ to predict Gaussian candidates $\{\widehat{g}_{vk}\}_{k=1}^{K}$ and their objectness probabilities p_{vk}^{obj}. The queries are refined through intra-voxel self-attention and sparse inter-voxel attention across neighboring voxels. Because the local target is unordered and variable-sized, we use entropy-regularized Optimal Transport (OT) to softly match candidates to target Gaussians or a dummy "no-Gaussian" bin, with the matched target mass providing supervision for objectness. At inference, candidates are retained according to their objectness-opacity scores:

$$\widehat{\mathcal{G}} = \{\widehat{g}_{vk} \mid v \in \widehat{\mathcal{V}},\ p_{vk}^{\text{obj}} \widehat{\alpha}_{vk} > \tau_{\text{g}}\}. \tag{3}$$

Thus, K bounds only the local candidate capacity rather than fixing cardinality: both the spatial allocation of selected voxels and the number of retained primitives per voxel adapt to scene content, preserving a variable-cardinality native Gaussian representation. Appendix A.2 and A.3 provide the details of the architecture and OT.

3.4 JOINT SUPERVISION FOR GAUSSIAN VAE TRAINING

We jointly train the encoder and decoder to answer three coupled questions: *where* Gaussian decoding should occur, *what* local Gaussian set should be reconstructed, and *which* predicted candidates should remain active at inference.

Occupancy supervision. Occupancy supervision should balance spatial coverage with sparsity. Positive-only supervision can lead to an unnecessarily dense active set, whereas treating uncertain boundary samples as hard negatives may suppress thin structures. We therefore combine pointwise occupancy regression with a voxel-center margin objective:

$$\mathcal{L}_{\text{occ}} = \lambda_{\text{reg}}\, \mathbb{E}_{\mathbf{x}} |\rho(\mathbf{x}) - o(\mathbf{x})| + \lambda_{\text{m}} \big(\mathbb{E}_{v \in \mathcal{V}_+} [m_+ - \rho_v]_+^2 + \mathbb{E}_{v \in \mathcal{V}_-} [\rho_v - m_-]_+^2 \big)\,. \tag{4}$$

Here $o(\mathbf{x})$ is the reference occupancy signal and ρ_v is the predicted occupancy at voxel center v. The sets $\mathcal{V}_+$ and $\mathcal{V}_-$ are occupied and confidently empty voxel centers, respectively. The margins $m+$ and m_- encourage occupied centers to score above m_+ and empty centers below m_-.

Gaussian decoder supervision. For each occupied voxel v, we use an unordered set of $M_v \leq K$ target Gaussians for set-level supervision, while the decoder predicts K candidates. A detached OT plan P_v^* softly matches the predicted candidates to these targets and defines the position loss

$$\mathcal{L}_{\text{pos}} = \frac{1}{|\mathcal{V}_+|} \sum_{v \in \mathcal{V}_+} \frac{K}{M_v} \sum_{k=1}^{K} \sum_{j=1}^{M_v} [P_v^*]_{kj}\, \frac{\|\widehat{\mathbf{x}}_{vk} - \mathbf{x}_{vj}\|_1}{h}, \tag{5}$$

where h denotes the voxel size. Under this correspondence, we directly supervise only Gaussian centers; scale, rotation, appearance, and opacity are instead constrained through rendering supervision. As in set-based object prediction, we use an objectness loss $\mathcal{L}_{\text{obj}}$ to predict whether each Gaussian corresponds to a target primitive, using the non-dummy transport mass as a soft target. During rendering supervision, we render the selected Gaussian candidates from sampled cameras and compare them with target views rendered from the complete ground truth Gaussian scene:

$$\mathcal{L}_{\text{render}} = (1-\eta)\|\widehat{I}^r - I^r\|_1 + \eta\big(1 - \text{SSIM}(\widehat{I}^r, I^r)\big). \tag{6}$$

where η balances the pixel-wise and structural similarity terms. We stop gradients from $\mathcal{L}\text{render}$ through the predicted Gaussian centers, so that it constrains scale, rotation, spherical-harmonic appearance, and opacity without directly optimizing their positions. When a voxel contains more than K primitives, matched ℓ_1 supervision covers only a subset. Rendering against the full ground-truth scene instead encourages the decoder's K candidates to preserve the overall target appearance. For occupied voxels, the decoder objective is

$$\mathcal{L}_{\text{GS}} = \lambda_{\text{pos}}\mathcal{L}_{\text{pos}} + \lambda_{\text{obj}}\mathcal{L}_{\text{obj}} + \lambda_{\text{render}}\mathcal{L}_{\text{render}} \tag{7}$$

Appendix A.3 provides further details on the complete OT transport marginals, threshold-aligned activation objective, and rendering protocol.



Figure 3: **Qualitative VAE reconstruction comparison.** Compared with GaussianGPT, our method better captures scene geometry and appearance.

Phantom-voxel supervision. Training the set decoder only on ground-truth occupied voxels introduces a discrepancy: at inference, false-positive occupancy proposals are also passed to the decoder and may produce ghost Gaussians. We therefore mine predicted voxels that do not overlap ground-truth support as *phantom voxels*. Since these voxels contain no target primitives, we apply

$$\mathcal{L}_{\mathrm{ph}} = \mathbb{E}_{v \in \mathcal{V}_{\mathrm{ph}}} \left[\frac{1}{K} \sum_{k=1}^{K} \left(\lambda_{\mathrm{ph,obj}} \, \mathrm{BCE}(p_{vk}^{\mathrm{obj}}, 0) + \lambda_{\mathrm{ph},\alpha} |\widehat{\alpha}_{vk}| \right) \right]. \tag{8}$$

These terms teach the decoder to reject proposer errors rather than assuming that every decoded voxel contains a primitive. We exclude a small halo around occupied voxels to avoid treating uncertain boundaries as negatives; sampling details are given in Appendix A.3.

The KL term regularizes the latent for generative modeling. The complete objective is

$$\mathcal{L}_{\mathrm{VAE}} = \mathcal{L}_{\mathrm{occ}} + \mathcal{L}_{\mathrm{GS}} + \mathcal{L}_{\mathrm{ph}} + \lambda_{\mathrm{KL}} \mathcal{L}_{\mathrm{KL}}. \tag{9}$$

More detailed weights and the optimization schedule are reported in Appendix A.6.

3.5 Scene Generation over Gaussian Latents

We freeze the VAE and train a conditional latent flow-matching model. For image conditioning, dense DINO features are lifted with estimated depth and camera geometry, splatted onto the three latent-aligned planes, and concatenated with patch embeddings of the noisy scene latent. Given a normalized clean latent $\mathbf{z}_0$, noise $\boldsymbol{\epsilon}$, and $\sigma \in [0, 1]$, we use the linear path $\mathbf{z}_\sigma = (1 - \sigma)\mathbf{z}_0 + \sigma\boldsymbol{\epsilon}$ and optimize

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}\left[\frac{1}{D} \left\| v_\theta(\mathbf{z}_\sigma, \sigma, \mathbf{c}) - (\boldsymbol{\epsilon} - \mathbf{z}_0) \right\|_2^2 \right]. \tag{10}$$

where D denotes the dimensionality of the vectorized latent. The same latent interface also supports point-cloud-conditioned completion. Appendix A.5 details how image and point-cloud observations are converted into aligned condition tokens and injected into the flow transformer.

4 Experiments

4.1 Datasets and Evaluation Protocol

Datasets and View Sampling. We evaluate on InteriorGS (Miao et al., 2026), which contains large-scale structured indoor 3D Gaussian scenes, and the Aria Synthetic Environments (ASE)



Figure 4: **Adaptive 3DGS allocation.** Visualization of the target scene, VAE reconstruction, and retained Gaussian count per occupied voxel.

Table 2: **Conditional generation evaluation.** For single-image conditioning, appearance is evaluated separately in regions observed and unobserved from the conditioning view. Full-point-cloud conditioning covers the complete scene and therefore has no unobserved split. Detailed evaluation protocols are provided in Appendix A.6.

	Observed Appearance		Unobserved Appearance		Unobserved Geometry	
Condition	FID ↓	KID ($\times 10^3$) ↓	FID ↓	KID ($\times 10^3$) ↓	COV-CD (%) ↑	MMD-CD ($\times 10^3$) ↓
Full point cloud	54.12	16.76	N/A (complete conditioning)			
Single image	60.38	18.20	67.81	21.43	85.24	12.75

3DGS subset of SceneSplat-49K (Ma et al., 2025). Since InteriorGS scenes span large spatial regions, we crop them into smaller training samples following the procedure in Appendix 4, yielding over 19,600 crops. Aria scenes are used directly without cropping, yielding approximately 8,600 samples. InteriorGS and Aria scenes are split 9:1 into training and test sets before cropping to avoid scene overlap. To construct single-image-to-amodal-3DGS pairs, we sample candidate conditioning views and use Qwen3-VL-8B Bai et al. (2025) to filter out uninformative views.

Implementation Details The triplane encoder uses $Q = 4$ tokens per plane cell, and the variational latent has $C_z = 16$ channels per plane at a spatial resolution of 256^2. The decoder predicts at most $K = 24$ candidates per selected voxel on a 256^3 occupancy grid. We train the Gaussian VAE with Adam and a learning rate of 2×10^{-4}. After convergence, we freeze the VAE, normalize its latents with training-set statistics, and train the flow model with AdamW, learning rate of 10^{-4}, and weight decay 0.01. Both stages use gradient clipping and cosine learning-rate schedules with linear warmup. We train the VAE for 80K steps and the latent flow model for 100K steps on AMD MI308X GPUs, with a per-GPU batch size of 1. Complete data preprocessing and optimization settings are provided in Appendices A.1 and A.6, respectively.

4.2 Scene Reconstruction

In this section, we evaluate how well the Gaussian VAE preserves scene appearance and geometry. We compare against Can3Tok (Gao et al., 2025), DiffGS (Zhou et al., 2024), and GaussianGPT (von Lützow et al., 2026), which respectively use canonical tokens, continuous feature fields, and voxel-quantized representations. Table 1 reports rendering fidelity (PSNR, SSIM, and LPIPS) and geometric reconstruction (CD). Our method achieves the best results across all metrics, improving PSNR over GaussianGPT by 5.15 dB while reducing LPIPS and CD. Figure 3 further shows that our VAE better preserves scene structure and appearance. Additional reconstructions are provided in Appendix Figure 8, and Appendix A.9 evaluates our VAE's latent smoothness and robustness. Using official implementations under the same scene-level training and evaluation protocol, adapted Can3Tok and DiffGS baselines converge to low-detail reconstructions, mapping distinct input scenes to similar coarse structures (Appendix Figure 9).

Table 1: Quantitative comparison with existing 3D Gaussian representation methods. Chamfer Distance (CD) values are multiplied by 10^3. Best results are in **bold**, second-best are underlined.

Method	PSNR↑	SSIM↑	LPIPS↓	CD↓
Can3Tok	4.53	0.012	0.920	10.96
DiffGS	12.43	0.719	0.374	3.21
GaussianGPT	<u>21.40</u>	<u>0.856</u>	<u>0.157</u>	<u>2.62</u>
TriFold (Ours)	**26.55**	**0.913**	**0.088**	**2.23**



Figure 5: **Point-cloud-conditioned generation.** Given a full point cloud, the latent flow model generates native 3DGS with coherent geometry and appearance across novel views.



Figure 6: **Single-image amodal generation.** Given one input image, the latent flow model generates plausible room structure and appearance across novel views.

Adaptive Gaussian Allocation. Figure 4 compares the target and reconstruction and shows the retained candidates per occupied voxel. The decoder predicts $K = 24$ candidate Gaussians per voxel and retains only those with sufficient objectness, allowing the number of output Gaussians to vary with local content. Thus, K specifies an upper bound rather than a fixed output count. We choose $K = 24$ based on the InteriorGS statistics in Figure 7, where 97.8% of occupied voxels contain at most 24 Gaussians on the same 256^3 grid. For datasets with higher local densities, the decoder can be configured with a larger K without changing the latent layout.

4.3 Scene Generation

To evaluate the generative utility of the learned TriFold latent, we train latent flow-matching models under full-point-cloud and single-image conditioning. Our latent supports generation from complete spatial observations and amodal completion from a single view, while decoding to content-adaptive native 3DGS. As existing 3DGS generators typically adopt different conditioning protocols, direct comparisons require method-specific adaptation; we focus on evaluating generation within our conditional setup rather than benchmarking across generative methods. Qualitatively, Figures 5 and 6 show plausible scene geometry and appearance across widely separated views. In particular, the single-image model synthesizes plausible room structure and content beyond the input view. Additional samples are provided in Figures 13 and 14. Quantitatively, Table 2 shows better observed-region appearance under full-point-cloud conditioning, while the single-image model achieves an unobserved-region FID of 67.81 and a COV-CD of 85.24%. Since a single image only partially observes the scene, plausible completions may differ from the reference in unobserved regions.

4.4 Ablation Studies

Data Scaling and Cross-dataset Generalization. Unless otherwise specified, all ablations are trained on the InteriorGS training split. Under this setting, our model transfers to Aria without any Aria training data (Table 4), demonstrating cross-dataset generalization. Separately, Table 3 shows consistent gains from scaling the training data, with further improvements when Aria is included.

Table 3: **Training data composition.** We compare models trained with different amounts of data and evaluate all variants on the same held-out test set.

Training data	PSNR↑	SSIM↑	LPIPS↓	CD↓
+ InteriorGS	25.83	0.904	0.096	2.27
+ Aria	**26.55**	**0.913**	**0.088**	**2.23**

Table 4: **Cross-dataset generalization.** Models are evaluated on the Aria test set *without* any Aria training data; the last row is an in-domain reference trained with its training set.

Training data	PSNR↑	SSIM↑	LPIPS↓	CD↓
InteriorGS	22.00	0.720	0.33	2.21
+ Aria (in-domain)	**24.26**	**0.781**	**0.32**	**2.17**

Table 5: **Component analysis.** Each row adds or replaces one model or training component while keeping the remaining evaluation protocol fixed. For the $\mathcal{L}_{\text{pos}}$-only baseline, we use the same correspondence to directly supervise all Gaussian attributes.

ID	Architecture		Training objective			Rendering quality			Gaussian reconstruction
	Triplane Encoder	Triplane Decoder	Phantom	$\mathcal{L}_{\text{pos}}$	$\mathcal{L}_{\text{render}}$	PSNR↑	SSIM↑	LPIPS↓	Chamfer Distance (CD) ↓
baseline				✓		12.10	0.617	0.491	3.06
1	✓			✓		14.72	0.752	0.378	2.58
2		✓		✓		13.53	0.669	0.330	2.77
3	✓	✓		✓		17.81	0.803	0.231	2.48
4	✓	✓			✓	22.64	0.885	0.134	2.41
5	✓	✓		✓	✓	24.39	0.892	0.107	2.36
Ours	✓	✓	✓	✓	✓	**25.83**	**0.904**	**0.096**	**2.27**

Component Ablations. We next isolate the contribution of individual model components while keeping the training data and evaluation protocol fixed. Table 5 analyzes the model in three stages. First, the multi-query encoder and query-based set decoder each improve the baseline, while combining them yields further gains, demonstrating their complementary roles in Gaussian encoding and decoding. Second, rendering supervision substantially improves visual fidelity, and combining it with the position loss further benefits both appearance and geometry. Finally, phantom-voxel training provides additional gains across all metrics, resulting in the full model.

Architectural Design Choices. Beyond the component-level ablation, Table 6 compares alternative encoder and decoder implementations with the rest of the pipeline fixed. For Gaussian encoding, point-based backbones outperform the sparse 3D CNN, with PTv3 performing best and capturing richer geometric information, as illustrated in Figure 10. For aggregation, QFormer improves over mean pooling, supporting query-based aggregation for resolving many-to-one collisions in triplane cells. For decoding, the set decoder substantially outperforms the fixed-output MLP, validating adaptive set prediction for recovering variable-cardinality Gaussians. Together, these results show that the gains arise from architectural choices tailored to the irregular and variable-density structure of 3DGS.

Table 6: **Architectural design choices.** Comparison of alternative encoder, projection, and decoding designs under the same evaluation settings.

Variant	PSNR↑	SSIM↑	LPIPS↓	CD↓
Encoder Head				
Sparse 3D CNN	23.16	0.830	0.139	2.48
PointNet	24.37	0.878	0.117	2.41
PTv3	**25.83**	**0.904**	**0.096**	**2.27**
Aggregation				
Mean pooling	24.60	0.857	0.102	2.39
QFormer	**25.83**	**0.904**	**0.096**	**2.27**
Decoder Head				
MLP	23.09	0.885	0.132	2.55
Set decoder	**25.83**	**0.904**	**0.096**	**2.27**

5 Conclusion

We introduced **TriFold**, a native scene-level 3D Gaussian VAE that maps Gaussian sets with adaptive cardinality and density into fixed-layout triplane latents. Rather than regularizing the Gaussian representation itself, the encoder captures fine-grained structure using cell-wise learned queries and primitive-anchored cross-plane attention, while the decoder predicts local Gaussian sets with scene-dependent allocations. The resulting latent jointly represents geometry and appearance and provides a regular interface for flow-based generative modeling. Experiments demonstrate high-fidelity reconstruction across diverse and complex scenes and show that the learned latent supports conditional scene generation, including amodal completion from partial observations.

AI USE STATEMENT

Generative AI tools assisted with drafting and editing the abstract, introduction, related work, and method sections; inspecting the implementation for consistency with the manuscript; identifying relevant literature; and formatting references. The authors are responsible for independently checking all AI-assisted text, technical claims, citations, and mathematical notation against the implementation, experiments, and cited sources. The authors take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Sections 3 and 4 describe the model formulation, training objectives, datasets, metrics, and evaluation protocol. The appendix provides further implementation details, including data preprocessing and spatial normalization (Appendix A.1), the VAE architecture and set decoder (Appendices A.2 and A.3), baseline adaptations (Appendix A.4), condition construction and latent flow matching (Appendix A.5), and training, sampling, and evaluation settings (Appendix A.6). Together, these sections specify the model dimensions, loss weights, optimization schedules, sampling parameters, dataset splits, and evaluation procedures used in our experiments. Appendix A.10 further discusses limitations related to the scale and diversity of the available 3DGS training data and the fixed-scale spatial normalization adopted in the current framework.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Miguel Angel Bautista, Pengsheng Guo, Samira Abnar, Walter Talbott, Alexander Toshev, Zhuoyuan Chen, Laurent Dinh, Shuangfei Zhai, Hanlin Goh, Daniel Ulbricht, Afshin Dehghan, and Joshua Susskind. GAUDI: A neural architect for immersive 3d scene generation. In *Advances in Neural Information Processing Systems*, volume 35, 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*, 2020.

Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5795–5805. IEEE, 2021.

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, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2022.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19457–19467. IEEE, 2024.

Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4456–4465, 2023.

Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W. Taylor, and Joshua M. Susskind. Unconstrained scene generation with locally conditioned radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances in neural information processing systems*, 35:31841–31854, 2022.

Quankai Gao, Iliyan Georgiev, Tuanfeng Y. Wang, Krishna Kumar Singh, Ulrich Neumann, and Jae Shin Yoon. Can3Tok: Canonical 3d tokenization and latent modeling of scene-level 3d gaussians. In *ICCV*, 2025.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 2023.

Joo Chan Lee, Jong Hwan Ko, and Eunbyung Park. Optimized minimal 3d gaussian splatting. *Advances in Neural Information Processing Systems*, 38:135864–135888, 2025.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

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

Mengjiao Ma, Qi Ma, Yue Li, Jiahuan Cheng, Runyi Yang, Bin Ren, Nikola Popovic, Mingqiang Wei, Nicu Sebe, Luc Van Gool, Theo Gevers, Martin R. Oswald, and Danda Pani Paudel. Scenesplat++: A large dataset and comprehensive benchmark for language gaussian splatting. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS 2025)*, 2025.

Quan Meng, Lei Li, Matthias Nießner, and Angela Dai. LT3SD: Latent trees for 3d scene diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 650–660, 2025.

Bingchen Miao, Rong Wei, Zhiqi Ge, Xiaoquan sun, Shiqi Gao, Jingzhe Zhu, Renhan Wang, Siliang Tang, Jun Xiao, Rui Tang, and Juncheng Li. Towards physically executable 3d gaussian for embodied navigation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=HB6KvsqcAn.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, 2020.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Ming Qian, Zijian Wang, Minchao Sun, Jincheng Xiong, Hang Zhang, Mu Xu, Chi Wang, and Baoquan Chen. GS-Voxel: Fitting-free structured latents for large-scale 3dgs generation. *arXiv preprint arXiv:2608.17988*, 2026.

Barbara Roessle, Norman Müller, Lorenzo Porzi, Samuel Rota Bulò, Peter Kontschieder, Angela Dai, and Matthias Nießner. L3dg: Latent 3d gaussian diffusion. In *SIGGRAPH Asia 2024 Conference Papers*, pp. 1–11, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Yawar Siddiqui, Antonio Alliegro, Alexey Artemov, Tatiana Tommasi, Daniele Sirigatti, Vladislav Rosov, Angela Dai, and Matthias Nießner. Meshgpt: Generating triangle meshes with decoder-only transformers. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19615–19625. IEEE, 2024.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pp. 1–18. Springer, 2024.

Tencent Hunyuan3D Team. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details, 2025. URL https://arxiv.org/abs/2506.16504.

Nicolas von Lützow, Barbara Rössle, Katharina Schmid, and Matthias Nießner. GaussianGPT: Towards autoregressive 3d gaussian scene generation. In *European Conference on Computer Vision*, 2026.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

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 *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 4563–4573. IEEE, 2023.

Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Jingxi Xu, Philip Torr, Xun Cao, and Yao Yao. Direct3d: Scalable image-to-3d generation via 3d latent diffusion transformer. *Advances in Neural Information Processing Systems*, 37:121859–121881, 2024a.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer V3: Simpler, faster, stronger. In *CVPR*, 2024b.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *CVPR*, 2025.

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 Conference on Computer Vision and Pattern Recognition*, pp. 14419–14429, 2026.

Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. In *European Conference on Computer Vision*, pp. 1–20. Springer, 2024.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.

Bowen Zhang, Yiji Cheng, Jiaolong Yang, Chunyu Wang, Feng Zhao, Yansong Tang, Dong Chen, and Baining Guo. GaussianCube: A structured and explicit radiance representation for 3d generative modeling. In *NeurIPS*, 2024.

Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, Bin Fu, Tao Chen, Gang Yu, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. *Advances in neural information processing systems*, 36:73969–73982, 2023.

Junsheng Zhou, Weiqi Zhang, and Yu-Shen Liu. DiffGS: Functional gaussian splatting diffusion. In *NeurIPS*, 2024.

Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10324–10335, 2024.

A APPENDIX

In this appendix, we provide additional implementation and training details, along with extended experimental evaluations and qualitative results. These additions complement the main paper and further demonstrate the effectiveness and robustness of our approach.

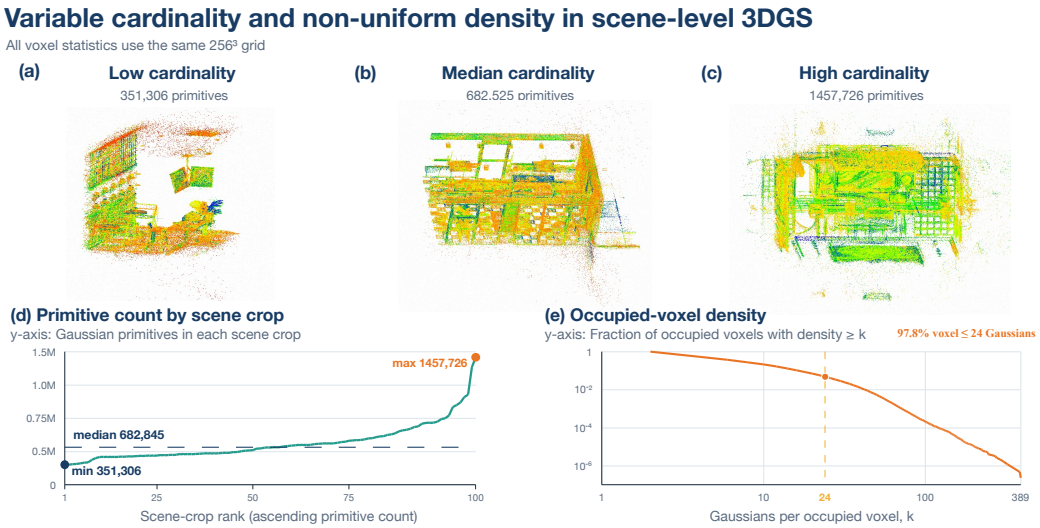


Figure 7: **Variable cardinality and nonuniform spatial density in scene-level 3DGS.** (a-c) Representative InteriorGS scene crops with different primitive counts, visualized using Gaussian centers voxelized on a shared 256^3 grid. (d) Across 100 random scene crops, the number of primitives ranges from 351,306 to 1,457,726, with a median of 682,845. (e) The complementary cumulative distribution of occupied-voxel density shows a pronounced long tail: 97.8% of occupied voxels contain at most 24 Gaussians, while the maximum density can even reach 389 Gaussians per voxel. These statistics motivate a representation that accommodates both scene-dependent cardinality and non-uniform local Gaussian allocation.

Challenges of Native Scene-level 3DGS Modeling. As illustrated in Fig. 7, scene-level 3DGS exhibits substantial variations in both global primitive cardinality and local spatial density. Across scene crops, total primitive counts vary by more than a factor of four, while the number of Gaussians within occupied voxels follows a pronounced heavy-tailed distribution. Compounded by severe clutter and complex occlusions inherent to indoor environments, these structural irregularities are inherently mismatched with the fixed-dimensional tensor structures required by standard latent generative models. A useful latent representation should therefore regularize the representation layout while preserving content-adaptive Gaussian allocation in the decoded scene.

By *native 3DGS* modeling, we specifically refer to a Gaussian-to-latent-to-Gaussian representation interface: our autoencoder directly ingests pre-fitted Gaussian sets and reconstructs content-adaptive Gaussian outputs, without converting the scene into an intermediate 3D representation or entangling representation learning with 2D-to-3D visual inference. This paradigm is fundamentally distinct from image-to-3DGS VAEs, whose latent representations are jointly shaped by the image-conditioned 3D reconstruction problem.

For computational tractability, our local set decoder uses $K = 24$ candidate queries per selected voxel. Here, K defines the maximum local candidate capacity rather than a fixed output cardinality: each selected voxel retains a content-dependent subset of candidates, while the number and spatial distribution of selected voxels remain scene-dependent. On the same 256^3 grid, 97.8% of occupied voxels in the original fitted scenes already contain at most 24 Gaussians. For the remaining high-density tail, the set-level matching target retains at most K primitives using an opacity- and scale-aware selection strategy. Importantly, this local selection does not limit the rendering target: rendering supervision is computed against views rendered from the complete fitted Gaussian scene

prior to local primitive selection. The decoder is therefore supervised not only by local primitive correspondence, but also by the collective rendering effect of the complete Gaussian representation.

A.1 Data Preparation

Gaussian Scene Normalization. Each input scene is represented as an unordered set of fitted Gaussian primitives. A primitive consists of 59 values: a 3D center, 48 spherical-harmonic coefficients for degree-3 appearance, one opacity logit, three anisotropic log-scales, and a four-dimensional rotation quaternion.

We use dataset-specific spatial normalization while maintaining a common canonical domain for the Gaussian VAE. For InteriorGS, each sample is constructed from a predefined 3D crop with a fixed spatial extent of $B = 8$. We translate the Gaussian centers by the corresponding crop center and apply the fixed transformation T_B that maps this crop to the canonical VAE domain. Because B is fixed, the spatial scale of this transformation is identical across all InteriorGS scenes. For Aria, which is used for Gaussian VAE evaluation but not for conditional diffusion training, no fixed $B = 8$ crop is applied; we instead determine a per-scene normalization scale from robust statistics of the Gaussian centers to map the valid scene extent into the same canonical domain.

In both cases, the model operates on a padded cubic domain of side length 1.1, discretized at a resolution of 256^3. The corresponding spatial transformation is applied consistently to Gaussian centers and occupancy-sample coordinates, and Gaussian log-scales are adjusted by the logarithm of its scale factor. Opacity, appearance, and rotation remain unchanged.

For set-level matching during Gaussian VAE training, an over-populated voxel retains at most 24 target primitives using an opacity- and scale-aware selection heuristic. This local selection is used only to construct the bounded set-level target; rendering supervision is computed from the complete fitted Gaussian scene before this selection.

Occupancy Targets Caching. The occupancy predictor is supervised by sampled 3D locations rather than by dense enumeration of all 256^3 voxels. For each scene, we construct a nearest-center distance field from the preprocessed Gaussian centers and draw samples across four distinct regions: exact Gaussian centers, a thin inner shell, a near-surface transition shell, and far-field points. Center and inner-shell samples represent occupied space, the transition shell decays linearly toward zero, and far-field samples serve as negatives. In our experimental configuration, the two shell radii correspond to 0.47 and 1.19 voxels at resolution 256, and the four groups account for 15%, 20%, 35%, and 30% of the cached samples, respectively. Samples outside the padded model domain are removed after applying the corresponding scene scale. At each training iteration, we randomly sample 250,000 cached locations for occupancy supervision. This hierarchical sampling strategy sharpens the decision boundary while maintaining sufficient coverage of empty space.

A.2 Gaussian VAE Architecture

Multi-slot Writing and Cross-plane Aggregation. The point features are orthographically projected onto the xz, xy, and yz planes at a resolution of $R_p = 256$. Within each occupied cell, a four-layer, eight-head QFormer employs $Q = 4$ learned queries to read the unordered set of projected primitives. Prior to attention, each projected feature is encoded as:

$$\widetilde{\mathbf{h}}_i^p = \mathrm{MLP}([\,\mathbf{h}_i + \mathbf{e}_p,\ \gamma(\mathbf{u}_i^p),\ \gamma(\boldsymbol{\delta}_i^p)\,])\,, \qquad \boldsymbol{\delta}_i^p = R_p\mathbf{u}_i^p - \lfloor R_p\mathbf{u}_i^p \rfloor, \tag{11}$$

where $\mathbf{u}_i^p \in [0,1)^2$ is the normalized in-plane coordinate, $\boldsymbol{\delta}_i^p$ is its within-cell offset, $\mathbf{e}_p$ is a learned plane embedding, and γ represents an eight-frequency Fourier encoding. The learned queries also receive the Fourier encoding of the cell center. Collectively, these terms encode the specific plane, cell, and location of the primitive, while the PTv3 feature $\mathbf{h}_i$ provides the necessary 3D context. The multiple queries capture diverse summaries of the primitives colliding within a cell, preventing the information bottleneck caused by single-vector pooling.

Subsequently, the encoder performs a single sparse cross-plane aggregation step. For each primitive, its three projected cells constitute an aligned triplet; tokens sharing the same query index attend across this triplet, and updates common to multiple primitives are averaged. This mechanism allows the orthogonal planes to reconcile complementary views of the same 3D geometry without requiring

dense attention across all plane cells. Finally, the VAE maps each plane to $C_z = 16$ variational channels using a base width of 384, channel multipliers $[1, 2]$, and three residual blocks per resolution. Stacking the three planes along the channel dimension yields a $48 \times 4 \times 256 \times 256$ latent tensor.

Occupancy-guided Support Proposal. At each candidate voxel center $\mathbf{x}_v$, the predictor retrieves Q features from each plane using the exact same integer cell indexing as the Gaussian decoder. For every slot q, it concatenates the corresponding xz, xy, and yz features and projects the resulting $3C$-dimensional vector back down to C dimensions. The predictor then flattens the Q fused features, appends $\mathbf{x}_v$ alongside its six-frequency Fourier encoding, and processes the result through a skip-connected MLP containing two four-layer blocks. A sigmoid activation maps the output logit to the occupancy probability ρ_v. During inference, we evaluate the 256^3 voxel centers in chunks and forward cells with $\rho_v > 0.4$ to the set decoder. This shared indexing guarantees that the occupancy predictor and Gaussian decoder consistently access the identical three plane cells at any given voxel.

DETR-style set decoder. Since the decoder must decode a regular triplane into an irregular point set of unknown local cardinality, we formulate the decoding process as a set prediction problem rather than fixed-index regression. First, an occupancy predictor identifies candidate voxels. For each candidate, 24 learned queries cross-attend to the three sampled plane contexts to explain the local Gaussian set. Separate task heads predict the primitive's offset, scale, rotation, spherical-harmonic appearance, opacity, and objectness. Sinkhorn matching establishes an order-independent correspondence to the target set (Sec. A.3); unmatched queries are routed to a dummy target. At inference time, inactive candidates are dynamically pruned based on their predicted objectness and opacity.

A.3 Objectness-Aware Set Prediction and OT Matching

Objectness and Differentiable Selection Supervision. Objectness indicates the probability that a query should successfully instantiate a Gaussian, whereas opacity defines that instantiated Gaussian's rendering contribution. Decoupling these roles ensures that unused query slots aren't penalized solely via near-zero opacity. The soft objectness target is derived from the non-dummy mass assigned by the Sinkhorn algorithm and is supervised via a balanced binary cross-entropy loss. During rendering and inference, query k is ranked by a unified score:

$$s_k = p_k^{\mathrm{obj}}\, \alpha_k, \tag{12}$$

where α_k represents its predicted opacity. In our reported VAE configuration, the selection threshold is set to 0.1. A differentiable activation loss forces at least one matched query in every occupied voxel to surpass a margin of 0.15, while suppressing unmatched queries below the selection boundary. During training, the renderer utilizes the union of OT-matched queries and those queries whose predicted scores naturally exceed the threshold. Consequently, a false positive selected by the model is directly exposed to the rendering objective, rather than being shielded by the target assignment.

Because candidate enumeration inherently involves a discrete thresholding step, we do not differentiate through the set membership operation itself. Instead, the occupancy, objectness, opacity, and activation scores governing this set are jointly optimized using differentiable surrogate losses aligned with the inference rule. As a result, the threshold functions as a deployment decision based on learned scores, rather than a manually tuned, post-hoc pruning heuristic.

Optimal-transport matching. Within each occupied voxel v (of size h), the decoder predicts a fixed set of K candidate Gaussians to reconstruct a variable-sized, unordered target set of $M_v \le K$ Gaussians. To absorb candidate slots that do not correspond to any target primitive, we append an additional dummy "no-Gaussian" entry. We define the pairwise matching cost solely based on Gaussian centers:

$$[\mathbf{A}_v]_{kj} = \begin{cases} \frac{1}{h}\,\|\widehat{\mathbf{x}}_{vk} - \mathbf{x}_{vj}\|_1, & 1 \le j \le M_v, \\ 0, & j = M_v + 1, \end{cases} \tag{13}$$

where $\widehat{\mathbf{x}}_{vk}$ and $\mathbf{x}_{vj}$ denote the centers of the k-th candidate and the j-th target Gaussian in voxel v, respectively. We obtain a soft correspondence by solving an entropy-regularized optimal transport problem:

$$P_v^* = \arg\min_{P \in \Pi(\mathbf{a}_v, \mathbf{b}_v)} \langle \mathbf{A}_v, P \rangle_F - \varepsilon\, \mathcal{H}(P), \tag{14}$$

where $\mathcal{H}(P) = -\sum_{k,j} P_{kj} \log P_{kj}$ and $\Pi(\mathbf{a}_v, \mathbf{b}_v)$ denotes the set of transport plans subject to the marginals

$$\mathbf{a}_v = \frac{1}{K}\mathbf{1}_K, \qquad \mathbf{b}_v = \left[\frac{1}{K}\mathbf{1}_{M_v}^\top,\ 1 - \frac{M_v}{K}\right]^\top. \tag{15}$$

By construction, each target Gaussian receives a mass of $1/K$, while the dummy entry absorbs the residual mass $1 - M_v/K$ from unmatched candidate slots. We solve for P_v^* with Sinkhorn iterations and stop gradients through it, i.e., the transport plan is treated as constant during backpropagation.

Under this correspondence, we directly supervise only Gaussian centers; scale, rotation, appearance, and opacity are instead constrained through rendering supervision. The same transport plan also provides a soft objectness target. Since each candidate has a row mass of $1/K$, its total non-dummy transport mass yields

$$t_{vk} = K \sum_{j=1}^{M_v} [P_v^*]_{kj} = 1 - K\,[P_v^*]_{k,M_v+1}. \tag{16}$$

Candidates assigned to real Gaussians approach a target of 1, whereas those routed to the dummy entry approach 0. We optimize the corresponding objectness logits with a binary cross-entropy loss, balanced by the number of positive and negative candidates within each batch. Thus, a single center-based OT correspondence serves two purposes: it assigns each candidate a target Gaussian for center supervision, and it indicates whether the candidate represents a valid primitive, eliminating any separate matching heuristic for objectness.

Phantom Voxels Regularization. If the decoder were trained exclusively on ground-truth occupied voxels, it would never encounter false positives, even though the learned occupancy proposer will inevitably produce them during inference. This train-test domain gap would cause queries at erroneous locations to decode "ghost" Gaussians with high objectness or opacity. We bridge this gap by deliberately feeding selected false-positive proposals to the decoder as *phantom voxels*. Because they contain no target Gaussians, all queries within a phantom voxel receive negative objectness supervision, forcing their activation and opacity to be heavily suppressed. Phantom training commences after the initial occupancy warm-up phase; we subsequently alternate between ground-truth and predicted occupancy supports, enabling the decoder to master both accurate local reconstruction and robust recovery from proposer errors. Although sampling the phantom set is non-differentiable, all losses evaluated on the selected queries are strictly differentiable with respect to the decoder and its prediction heads. This mechanism operates analogously to hard-negative mining, directly optimizing the network against failure cases it will realistically face at test time.

A.4 Baseline Implementation and Training

For Can3Tok, DiffGS, and GaussianGPT, we use the official implementations released by the respective authors. To ensure a controlled comparison, we adapt only their data-loading pipelines as necessary to train and evaluate all methods on the same scene-level 3DGS dataset and train/test split used by TriFold. Unless otherwise stated, we retain the original model architectures, representation designs, training objectives, and method-specific optimization settings, including the optimizer, learning-rate schedule, batch size, and loss weights, from the corresponding official training configurations.

Can3Tok's official implementation requires a fixed primitive budget of 40,000 Gaussians per scene. We therefore downsample each pre-fitted 3DGS scene to 40,000 Gaussians before tokenization and keep this budget fixed across all scenes. All baseline models are trained from scratch for 80K optimization steps, using the same number of optimization steps as our Gaussian VAE. Evaluation is performed on the same held-out scenes using the same rendering views and evaluation protocol for all methods.

Under this setting, GaussianGPT converges to meaningful scene-level reconstructions, whereas the adapted Can3Tok and DiffGS models converge to substantially lower-fidelity solutions. In particular, distinct input scenes are frequently reconstructed as similar low-detail Gaussian structures, as illustrated in Figure 9. We report the resulting models directly rather than excluding these degraded cases from the comparison. These results characterize the behavior of the released methods under

our scene-level adaptation and evaluation setting and should not be interpreted as general limitations of the original methods in the domains for which they were designed.

Geometric evaluation of degenerate predictions. We evaluate center-wise Chamfer Distance (CD) for all methods using the same preprocessing and the common normalized scene coordinate frame, without method-specific post-hoc alignment. Under our scene-level adaptation, the lower-fidelity baseline solutions can exhibit degenerate Gaussian attributes and substantial drift of the predicted centers from the reference geometry. We retain these errors in the geometric evaluation rather than realigning the predictions or excluding degraded scenes, such that CD directly reflects the resulting geometric discrepancy.

Accordingly, the large CD values of the affected baselines reflect their substantial geometric degradation rather than differences in the evaluation protocol. Rendering-based metrics (PSNR/SSIM/LPIPS) are evaluated independently using the same rendering views for all methods. Thus, both geometric and rendering metrics are computed under a common evaluation protocol, and degraded baseline predictions are reported directly rather than removed or assigned special-case values.

A.5 Conditional Latent Flow Matching

Condition Injection. The image-to-GS and point-cloud-to-GS models share a unified context-conditioning architecture. Each input observation is first converted into a dense spatial representation aligned with the Gaussian VAE latent grid. These representations are then embedded as condition tokens and concatenated with the noisy scene tokens, and the combined sequence is processed via joint self-attention. To distinguish their roles, condition tokens are modulated using the clean timestep ($\sigma = 0$) via AdaLN, whereas scene tokens are modulated by the sampled noise level. Both token sets share the same two-dimensional RoPE coordinates at corresponding spatial locations but use separate patch embeddings to distinguish their modalities.

Image-condition preprocessing and coordinate alignment. Our conditional diffusion model is trained only on InteriorGS; Aria is used for evaluating the Gaussian VAE and is not included in diffusion training. Accordingly, the fixed $B = 8$ spatial extent described below applies specifically to InteriorGS.

For image-to-GS conditioning, we resize each conditioning image to 720×1280 and extract dense DINOv3 ViT-H+/16 features Siméoni et al. (2025), yielding a 45×80 patch grid with 1280 feature channels. We additionally estimate monocular depth using Depth Anything v3 Lin et al. (2025). Since monocular depth is defined only up to a global scale, we rescale the raw prediction $\widetilde{d}_n$ as

$$d_n = \gamma_d \widetilde{d}_n, \tag{17}$$

where γ_d is a single global depth-scale factor shared across all InteriorGS scenes and fixed throughout training and inference. This factor resolves the global scale ambiguity of monocular depth and is independent of the target 3DGS.

We discard the lowest-confidence 10% of valid depth pixels and back-project the remaining DINO features using the corresponding camera intrinsics and extrinsics. For a valid sample n with homogeneous image coordinate $\widetilde{\mathbf{u}}_n$, the back-projected 3D location is

$$\mathbf{x}_n^{\text{world}} = R^\top \left(d_n K^{-1} \widetilde{\mathbf{u}}_n - t \right), \tag{18}$$

where (R, t) denotes the world-to-camera transformation. After applying γ_d, the predicted depth and camera translation are expressed in the same metric units.

All InteriorGS samples used for conditional diffusion training and evaluation are constructed from predefined 3D crops with a fixed spatial extent of $B = 8$. Let $\mathbf{c}$ denote the center of the corresponding crop. We first express each lifted point in crop-local coordinates,

$$\mathbf{x}_n^{\text{crop}} = \mathbf{x}_n^{\text{world}} - \mathbf{c}, \tag{19}$$

and then apply the fixed crop-to-canonical transformation

$$\mathbf{x}_n^{\text{tri}} = T_B \left(\mathbf{x}_n^{\text{crop}} \right). \tag{20}$$

Because the crop extent B is fixed, T_B uses the same spatial scale for every InteriorGS scene; only the crop translation $\mathbf{c}$ varies with the predefined crop location. Crucially, the same crop translation and transformation T_B are applied to the Gaussian scene before encoding it into the target VAE latent. The lifted image features and target Gaussian latent therefore share the same canonical coordinate frame.

Both $\mathbf{c}$ and B are specified by the InteriorGS data construction and are available independently of the target Gaussian representation. Constructing the image condition at inference therefore requires only the conditioning RGB image, its predicted depth, the associated camera parameters, the fixed depth-scale factor γ_d, and the predefined crop transform. No target 3DGS coordinates or target-derived spatial statistics are used.

We orthographically project each lifted DINO feature $\mathbf{f}_n$ at $\mathbf{x}_n^{\mathrm{tri}}$ onto the xz, xy, and yz planes. For plane p, let $\mathcal{P}_{\mathbf{j}}^p$ denote the set of lifted samples whose projection falls into cell $\mathbf{j}$. Their features are aggregated by mean splatting:

$$\mathbf{S}_{\mathbf{j}}^p = \frac{1}{|\mathcal{P}_{\mathbf{j}}^p|} \sum_{n \in \mathcal{P}_{\mathbf{j}}^p} \mathbf{f}_n, \tag{21}$$

with empty cells zero-filled. This produces three condition planes spatially aligned with the target Gaussian triplane latent. We concatenate the three planes along the channel dimension and apply a dedicated 2×2 convolutional patch embedding to obtain $128^2 = 16{,}384$ spatial condition tokens.

Point-cloud conditioning. For point-cloud-to-GS conditioning, we apply the same spatial preprocessing used to construct the target Gaussian VAE latent. We retain only the Gaussian positions and colors, while replacing opacity, scale, and rotation with fixed neutral values before passing the resulting set through the frozen Stage-1 triplane encoder and VAE encoder. Specifically, we use an opacity logit of 40, an isotropic log-scale of $\log(0.07h)$ (with voxel size $h = 1.1/256$), and an identity quaternion.

The resulting condition latent has shape $48 \times 4 \times 256 \times 256$. We standardize it using the same training-set statistics as for the target VAE latents, flatten the four slots and 48 channels into a 192-channel tensor, and apply a separate 2×2 patch embedding to obtain $16{,}384$ spatially aligned condition tokens. For both conditioning modes, the complete condition is replaced by zeros with probability 10% during training for classifier-free guidance.

A.6 Training and Evaluation Details

Gaussian VAE training. We jointly train the PTv3 encoder, the variational triplane autoencoder, the occupancy predictor, and the local Gaussian set decoder end to end. The latent representation comprises four slots, each with 16 channels per plane, at a spatial resolution of 256^2. The decoder predicts $K = 24$ candidate Gaussians per selected voxel, with the active subset determined by predicted objectness. The overall objective combines occupancy, center reconstruction, objectness, rendering, and phantom-voxel suppression losses, together with KL regularization. As detailed in the main paper, optimal-transport matching operates solely on Gaussian centers and provides both the correspondence for center supervision and the soft objectness targets. Among the Gaussian attributes, only the normalized center coordinates are directly supervised under this correspondence, with a loss weight of 5. Scale, rotation, appearance, and opacity receive no direct matched-attribute supervision and are instead constrained through differentiable rendering, with additional opacity suppression on phantom voxels.

For the remaining objectives, the point-sampled and voxel-level occupancy losses use weights of 1 and 0.5, respectively; the objectness BCE and threshold-aligned activation losses use weights of 1 and 0.3; and the phantom-voxel objectness and opacity suppression terms use weights of 0.5 and 0.2. Direct spherical-harmonic regression is disabled in the reported configuration. We train the network with Adam at a peak learning rate of 2×10^{-4}. The KL weight is linearly warmed up to 2×10^{-4} over the first 6k steps, and gradient norms are clipped to 1.

For completeness, we describe the foreground-aware RGB loss used for rendering supervision. Rendered training views often contain a large black background, so a uniform full-image average would disproportionately weight background pixels. We therefore compute the RGB L_1 error separately



Figure 8: **Additional VAE Reconstructions.** Under the identical evaluation setup, our VAE preserves more visible layout, appearance, and fine-grained local structures in these examples.

over foreground and background regions and downweight the latter. Pixels whose target alpha exceeds 0.05 are treated as foreground. Let $\mathcal{F}^r$ and $\mathcal{B}^r$ denote the foreground and background pixel sets in view r, respectively, and let

$$\ell^r_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{p \in \mathcal{S}} \left\| \widehat{I}^r_p - I^r_p \right\|_1 \tag{22}$$

denote the mean RGB L_1 error over a pixel set $\mathcal{S}$. The RGB loss in our implementation is then

$$\ell^r_{\mathrm{rgb}} = \frac{w_{\mathrm{fg}}\, \ell^r_{\mathcal{F}} + w_{\mathrm{bg}}\, \ell^r_{\mathcal{B}}}{w_{\mathrm{fg}} + w_{\mathrm{bg}}}, \qquad w_{\mathrm{fg}} = 1, \quad w_{\mathrm{bg}} = 0.1. \tag{23}$$

Thus, the RGB term $\mathcal{L}_1$ in the main paper is implemented as this foreground-aware L_1 loss rather than a uniform full-image average. The complete rendering objective is

$$\mathcal{L}_{\mathrm{render}} = \mathbb{E}_r \left[(1-\eta)\, \ell^r_{\mathrm{rgb}} + \eta \big(1 - \mathrm{SSIM}(\widehat{I}^r, I^r)\big) \right], \qquad \eta = 0.2. \tag{24}$$

At each training iteration, we randomly sample 32 camera viewpoints per scene and render the full decoded Gaussian set at 640×640 resolution, comparing the renderings against the corresponding target images. For rendering supervision, all decoded candidates are rendered before objectness-based primitive selection, providing scene-level supervision beyond the capacity-limited center targets used for set-level matching.

Gaussian Generation Training. We train a Wan-style latent transformer Wan et al. (2025) consisting of 30 blocks, a hidden width of 1536, a feed-forward width of 8960, and 12 attention heads. The four-slot latent is patchified using non-overlapping 2×2 spatial patches, yielding $128 \cdot 128 = 16{,}384$ scene tokens. The DINO-splat tokens are concatenated with the noisy scene tokens and fed into the transformer via a shared self-attention stream. We optimize the flow-matching objective in Eq. 10 using AdamW, with a learning rate of 10^{-4}, weight decay of 0.01, 5k warm-up

Table 7: **Local Gaussian matching.** We compare how predicted and target Gaussians are matched inside each decoded voxel.

Setting	PSNR↑	SSIM↑	LPIPS↓	CD↓
Hard matching	25.61	**0.908**	0.103	2.29
Sinkhorn OT (Ours)	**25.83**	0.904	**0.096**	**2.27**

Table 8: **Triplane resolution.** R_p denotes the plane-grid resolution prior to latent compression.

Setting	PSNR↑	SSIM↑	LPIPS↓	CD↓
$R_p = 64$	21.07	0.860	0.155	2.65
$R_p = 128$	24.15	0.882	0.118	2.42
$R_p = 256$ (Ours)	**25.83**	**0.904**	**0.096**	**2.27**

steps, and cosine decay toward 10^{-5}. We apply conditional dropout with a probability of 0.1, maintain an exponential moving average (EMA) with a decay of 0.9999, and clip gradients to unit norm. The reported generator is trained for 100k steps.

Generation Inference and Evaluation. For all generation experiments, we use the pretrained parameters of the latent flow model. At inference, we integrate the learned flow from $\sigma = 1$ to $\sigma = 0$ using UniPC with 50 function evaluations and a classifier-free guidance scale of 1.0. For single-image-conditioned generation, each held-out test scene is conditioned on one input image, which provides a partial observation of the complete 3D scene. We generate one sample per condition using fixed random seeds. Generated and reference 3DGS scenes are rendered from the same 16 evaluation cameras at a resolution of 720×480.

Appearance metrics. We evaluate rendered appearance using Fréchet Inception Distance (FID) and Kernel Inception Distance (KID), computed from standard Inception-v3 features using the PyTorch-based metric implementation. For single-image conditioning, we separately evaluate regions observed and unobserved from the conditioning view. Specifically, we back-project valid pixels of the conditioning image into 3D using the corresponding depth and camera parameters, and reproject these 3D samples into each evaluation view. The resulting projection footprint defines the observed region, while the remaining valid region is treated as unobserved. The same region masks are applied to the generated and reference renderings when computing FID and KID. The masks are used exclusively for evaluation and are never provided to the generative model.

Geometry metrics. We evaluate geometric generation using Coverage (COV) and Minimum Matching Distance (MMD) under Chamfer Distance (CD). Each generated and reference scene is represented by 2,048 points sampled from its Gaussian centers. For single-image conditioning, we use the same geometric projection procedure to distinguish observed and unobserved scene geometry. Specifically, the conditioning observation is lifted into 3D using its depth and camera parameters, and its projection, together with the corresponding projected depth, determines the portion of the scene covered by the input observation. Geometry outside this observed support is treated as unobserved. We sample 2,048 points from the corresponding unobserved geometry and compute COV-CD and MMD-CD between generated and reference scene sets under the same spatial normalization.

A.7 Additional VAE Reconstruction Results

Figure 8 provides additional qualitative comparisons with GaussianGPT. In all three scenes, our method can preserve both the global room layout and fine-grained local structures. While GaussianGPT captures the coarse spatial envelope, it struggles with thin structures and details, often producing blurred or fragmented geometry. Although our reconstructions are not flawless, regions with sparse observations near scene boundaries appear less sharp than the ground truth; they still preserve the high-fidelity scene-specific geometry and appearances.

Figure 9 further examines a representative failure mode of the adapted Can3Tok and DiffGS baselines. Both methods tend to decode visually distinct input scenes into similar, low-detail Gaussian structures. They recover only a rudimentary spatial envelope, discarding most of the scene-specific layouts, appearances, and object structures. We present these cases as diagnostic examples to understand representational limits, rather than as a universal characterization of either baseline.





Figure 9: **Representative reconstruction failures of the adapted Can3Tok and DiffGS baselines.** Both baselines are trained using their officially released implementations on the same scene-level training split and with the same optimization budget used in our comparison. Under this setting, distinct input scenes can be reconstructed as similar low-detail structures. These examples illustrate the observed behavior under our scene-level adaptation and are not intended to characterize the methods outside this setting.

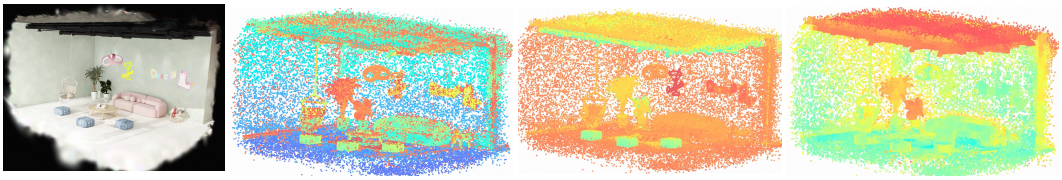


Figure 10: **Encoder feature comparison.** We visualize features extracted by PTv3, PointNet, and a sparse 3D CNN from the same Gaussian input. PTv3 retains clearer spatial variation around surfaces, objects, and scene boundaries in this example; Table 6 reports the corresponding quantitative comparison.

A.8 Expanded Ablation Results and Analysis

We complement the component ablations in the main paper with controlled studies on local Gaussian matching and representation capacity. Tables 7 and 8 examine the matching objective and triplane resolution, while Tables 9 and 10 vary the number of encoder slots and latent channels. Figures 10 and 11 provide qualitative evidence supporting the necessity of contextual point features and multiple local slots.

For all reconstruction ablations, PSNR, SSIM, and LPIPS are computed on the same held-out renderings, while Chamfer Distance (CD) is evaluated on decoded Gaussian geometry under identical normalization. Each hyperparameter variant is trained from scratch using the same data and evaluation split.

Figure 10 highlights the importance of the point backbone prior to triplane projection. PointNet Qi et al. (2017) extracts largely uniform features across the scene, resulting in limited local discriminability. While the sparse 3D CNN captures broad spatial variations, these variations mostly correspond to coarse, scene-level regions. In contrast, PTv3 features exhibit sharp transitions around floors, walls, furniture, and object boundaries. This preserves critical local structures, allowing the encoder to more accurately assign features to triplane cells. This qualitative observation aligns with

Table 9: **Encoder token slots.** Q denotes the number of feature slots per triplane cell.

Setting	PSNR↑	SSIM↑	LPIPS↓	CD↓
$Q = 1$	24.40	0.883	0.114	2.38
$Q = 2$	25.26	0.895	0.105	2.32
$Q = 4$ (Ours)	25.83	0.904	0.096	2.27
$Q = 8$	**26.11**	**0.910**	**0.090**	**2.23**

Table 10: **Latent channel width.** C_z specifies the number of latent channels per plane.

Setting	PSNR↑	SSIM↑	LPIPS↓	CD↓
$C_z = 8$	25.16	0.892	0.102	2.34
$C_z = 16$ (Ours)	25.83	0.904	0.096	2.27
$C_z = 32$	**25.89**	**0.905**	**0.094**	**2.25**

the architectural comparisons in Table 6 and justifies contextualizing Gaussian primitives before local aggregation.

Local Matching and Spatial Resolution. Table 7 reveals a metric-dependent trade-off between hard assignment and Sinkhorn OT. Sinkhorn improves PSNR from 25.61 to 25.83, reduces LPIPS from 0.103 to 0.096, and reduces CD from 2.29 to 2.27, while hard matching yields a slightly higher SSIM (0.908 versus 0.904). Overall, the transport-based objective yields superior perceptual and geometric reconstructions, even if it does not dominate every single metric. Table 8 exhibits a clearer trend: increasing R_p from 64 to 256 improves PSNR by 4.76 dB and SSIM by 0.053, while reducing LPIPS by 0.059 and CD by 0.38. These consistent gains demonstrate that a coarse grid prematurely discards spatial details prior to latent compression.

Slot Capacity and Latent Width. Increasing the encoder capacity from one to four slots improves all four metrics (Table 9), including a 1.43 dB increase in PSNR and a 0.11 reduction in CD. This confirms that utilizing multiple tokens per projected cell is crucial when multiple Gaussian primitives collapse into the same local region. Increasing the slot count from four to eight yields further but diminishing gains, revealing a quality–capacity trade-off. By comparison, widening the latent representation from $C_z = 16$ to $C_z = 32$ changes PSNR by only 0.06 dB, SSIM by 0.001, LPIPS by 0.002, and CD by 0.02 (Table 10). This indicates that the representation benefits significantly more from structured local capacity (slots) than from a brute-force increase in latent channels. We use $Q = 4$ and $C_z = 16$ as a compact operating point; however, $Q = 8$ remains a viable option when reconstruction quality is strictly prioritized over representation size.

Local Gaussian Decoder Capacity. We further study the effect of the maximum number of Gaussian candidates K predicted within each selected voxel. K defines a local capacity upper bound rather than a fixed scene-level output cardinality: each selected voxel retains a content-dependent subset of the K candidates, while the number and spatial distribution of selected voxels remain scene-dependent. As shown in Table 11, increasing K from 8 to 24 generally improves reconstruction quality, with particularly consistent gains in the appearance metrics. Increasing the capacity further to $K = 32$ yields only marginal improvements ($+0.13$ dB PSNR, $+0.003$ SSIM, and -0.004 LPIPS), while CD remains unchanged at the reported precision. These results indicate that $K = 24$ captures most of the useful local representation capacity in our setting.

Table 11: Effect of local Gaussian decoder capacity. K is the maximum number of Gaussian candidates per selected voxel, while rendering supervision always uses the complete target scene. CD values are multiplied by 10^3. Best results are in **bold**, second-best are underlined.

Setting	PSNR↑	SSIM↑	LPIPS↓	CD↓
$K = 8$	25.06	0.887	0.112	2.31
$K = 16$	25.61	0.895	0.106	<u>2.33</u>
$K = 24$ (Ours)	<u>25.83</u>	<u>0.904</u>	<u>0.096</u>	**2.27**
$K = 32$	**25.96**	**0.907**	**0.092**	**2.27**

This behavior is consistent with the local-density statistics in Figure 7: 97.8% of occupied voxels in the original fitted scenes contain at most 24 Gaussians on the same 256^3 grid. For the remaining high-density tail, the set-level correspondence target is truncated to at most K primitives, whereas the rendering target remains untruncated and is computed from the complete fitted Gaussian scene before local primitive selection. Consequently, although the decoder cannot reproduce more than K individual primitives within a single voxel, the retained Gaussians can jointly adjust their non-positional attributes under full-scene rendering supervision to approximate the image-space effect of the denser target representation. The small performance gap between $K = 24$ and $K = 32$

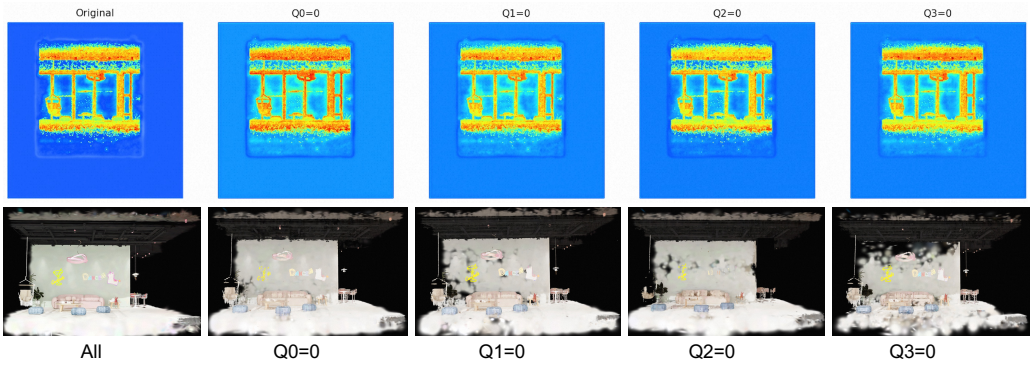


Figure 11: **Single-slot removal.** This intervention complements the slot-count ablation in Table 9. We zero one of the Q encoder slots while keeping all remaining latent values fixed. The top row visualizes decoded feature energy on the xz plane, and the bottom row shows the corresponding Gaussian reconstruction. Different removals produce distinct changes, indicating that the slots are not identical copies of the same local summary.

empirically suggests that this bounded local capacity is not a major bottleneck for reconstruction fidelity under our data distribution.

Figure 11 probes the four-slot representation without retraining the decoder. While masking out any single slot keeps the global room layout recognizable, it alters the feature response and degrades distinct local structures in the reconstructed Gaussians. The degraded regions and artifact patterns vary depending on which slot is removed, rather than presenting a uniform failure mode. Together with the monotonic slot-count improvements in Table 9, this demonstrates that the slots learn complementary local information. While this does not imply a strictly disentangled semantic meaning for each slot, it strongly supports the presence of non-redundant local decomposition within the projected cells.

A.9 Latent Representation Analysis

To evaluate the structural integrity and generative suitability of the learned VAE latent space, we analyze its smoothness and robustness through interpolation and perturbation. For interpolation, we linearly interpolate between two distinct scene latents, $\mathbf{z}_\alpha = (1-\alpha)\mathbf{z}_A + \alpha\mathbf{z}_B$. As shown in Fig. 12(a), the decoded triplane features exhibit smooth, continuous semantic transitions along the interpolation path, confirming a well-regularized latent manifold.

To assess robustness, we systematically inject Gaussian noise, apply element-wise dropout, and enforce structured masking on the latent codes, subsequently quantifying the resulting degradation relative to the unperturbed reconstruction. We further visualize the decoded triplane features under increasing masking ratios, where selected latent components are zeroed out. As shown in Fig. 12(b), the representation remains highly resilient under moderate perturbations: the decoded scene is largely preserved under Gaussian noise up to $\sigma = 0.5$, gracefully degrading only at extreme noise levels ($\sigma = 1.0$).

A.10 Additional Conditional Generation Results

We provide additional qualitative results for both point-cloud- and single-image-conditioned generation. These visual examples complement the quantitative evaluations in Sec. 4.3 by demonstrating the structural diversity of the generated 3D Gaussian scenes.

Point-cloud Conditioned Generation. Figure 13 presents supplementary scenes generated from full point-cloud conditions. These samples span a wide variety of indoor layouts that encompass diverse arrangements of structures. Notably, the generated Gaussian sets successfully assemble into holistic, scene-scale environments rather than degenerating into isolated objects or disconnected fragments. The results show that major surfaces and semantic arrangements are robustly generated.

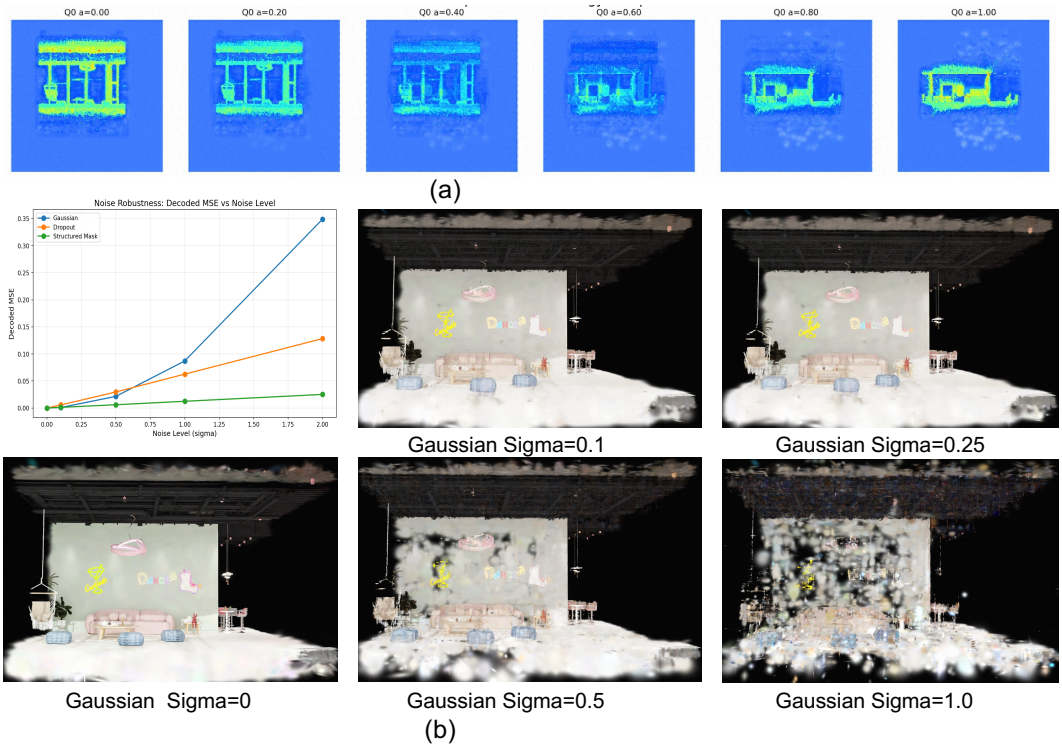


Figure 12: **Latent interpolation and perturbation analysis.** (a) Linear interpolation produces smooth changes in decoded triplane features. (b) The latent representation remains stable under



Figure 13: **Additional point-cloud-conditioned generations.** Each panel shows a Gaussian scene sampled from a different point-cloud condition. The examples illustrate the diversity of generated indoor layouts and scene content.

Single-image Conditioned Generation. Figure 14 shows some examples of single-image-conditioned generation. In each row, the input view is followed by top, front, and side renderings of a single sampled Gaussian scene. The generated scenes preserve prominent cues from the input, including wall decorations, window placements, cabinets, and furniture configurations. Crucially, these elements remain geometrically consistent across novel viewpoints: planar surfaces maintain their relative spatial arrangements, and scattered objects (e.g., tables and chairs) are grounded coherently within the shared 3D space.



Figure 14: **Single-image-conditioned generation.** Each row shows the conditioning view followed by top, front, and side renderings of one sampled Gaussian scene. The novel views assess whether input-specific scene cues are preserved within a coherent 3D completion.

Limitations. Our current framework is primarily limited by the scale and diversity of available scene-level 3DGS training data. In contrast to image and video VAEs and generative models, which can benefit from substantially larger and more diverse training corpora, our Gaussian VAE and latent flow model are trained on a relatively limited collection of 3D scenes. This restricts the diversity of the learned scene prior and may limit generalization to scene distributions that differ substantially from the training data. We expect larger and more diverse 3DGS datasets to further improve both representation learning and generative modeling. A second limitation concerns physical scene scale. To obtain a consistent spatial coordinate system, we construct all training samples using the same fixed-size 3D crop and map them to a canonical triplane domain. While this fixed-scale normalization stabilizes representation learning, it also ties the learned representation to the physical scale used during training, and scenes at substantially different spatial scales may not be represented optimally without additional normalization. Extending TriFold to scale-conditioned or multi-scale representations, together with training on larger 3DGS corpora, is therefore an important direction for improving its generalization to more diverse real-world scenes.