GLoD-LVSM: Learning Geometry-Grounded Continuous-LoD Scene Representations for Large View Synthesis Models

Anonymous authors
Paper under double-blind review

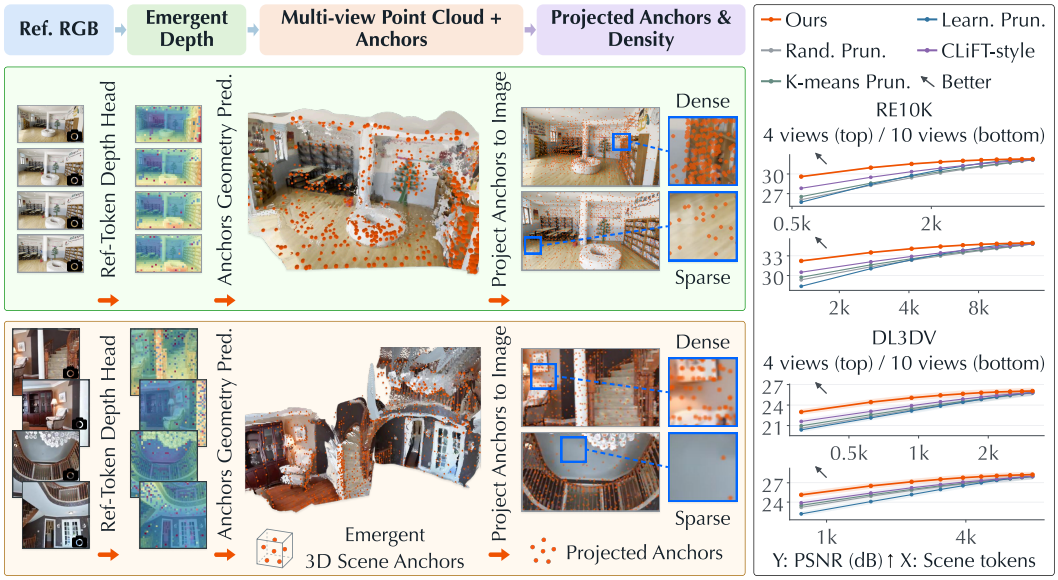

Fig. 1: GLoD-LVSM learns a compact, geometry-grounded scene representation (anchors) for novel view synthesis. Left: View-synthesis supervision and compression alone yield emergent depth and self-organized, surface-aligned scene anchors whose density adapts to local texture complexity. Right: GLoD-LVSM achieves the best quality–scene-token trade-off across reference-view counts.

Abstract

Large view synthesis models (LVSMs) reconstruct scenes from sparse views and render novel views quickly, but they incur unnecessary storage and rendering costs due to token redundancy. We present GLoD-LVSM, a compact, geometry-grounded scene representation for LVSMs with continuous levels of detail. First, instead of learning a fixed set of scene anchor initializations, GLoD-LVSM learns a continuous field that initializes a flexible number of anchors to match the desired budget. A condenser then aggregates scene information into these anchors, with both the field and condenser trained end-to-end. Second, the condenser infers reference-token and anchor geometry to guide compression through geometry-grounded attention, improving rendering quality and enabling frustum culling during decoding. Compared with existing LVSM scene-compression baselines, GLoD-LVSM achieves the best trade-off between rendering quality and scene token count. Compared with the uncompressed base model, GLoD-LVSM reduces the scene-token count by 33.4% and accelerates rendering by $1.23\times$ with no quality loss. Under a relaxed quality-loss budget, the reduction increases to 85.5%, with $3.00\times$ rendering speedup. Further analysis reveals that view-synthesis supervision yields emergent scene depth and self-organized anchors that align with surfaces and adapt their density to local texture complexity. Beyond rendering, we further show that our scene representation also supports scene-semantic understanding, highlighting its potential for broader downstream tasks.

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.

1 INTRODUCTION

Large view synthesis models (LVSMs) learn a reconstruction prior for novel view synthesis (NVS) (Jin et al., 2025; Kim et al., 2026; Szymanowicz et al., 2026), reducing scene reconstruction to a single feed-forward pass and enabling high-quality rendering from sparse reference views. Recent high-quality LVSMs commonly adopt a highway representation, where each reference-image patch is directly mapped to a scene token (Szymanowicz et al., 2026; Kim et al., 2026). While this design preserves detailed appearance information, the resulting scene-token count grows with the number of reference views, increasing both storage and decoding costs. This tight coupling between image and scene-representation sizes becomes a bottleneck with more reference views.

In addition to scaling poorly with reference view count, highway representations allocate scene capacity inefficiently in two ways. First, overlapping 3D regions observed from different views are stored as separate tokens rather than consolidated. Second, every image patch receives the same one-token capacity regardless of texture complexity. An ideal LVSM representation should therefore (1) support flexible scene-token count independent of the input-view count, (2) consolidate multi-view observations into a unified, *scene-centric* 3D representation, and (3) allocate capacity adaptively to scene complexity. Thus, we propose GLoD-LVSM, which provides two key features: (1) a geometry-grounded scene representation that supports (2) continuous level-of-detail. We refer to scene tokens in GLoD-LVSM as *anchors* to emphasize their scene-centric 3D nature.

Different storage and rendering budgets require different scene-token numbers, which correspond to different levels of scene detail. To support arbitrary scene-token counts and thus continuous levels of detail (CLoD), we first design an anchor initialization scheme that can generate a flexible number of tokens. Specifically, we introduce a CLoD scene-anchor initialization that parameterizes anchor initialization as a continuous field (Sec. 3.2). Sampling this field at different densities generates different numbers of anchors. This decouples the scene-token count from the reference-view count and allows our model to operate under different storage and rendering budgets.

A compact scene representation should consolidate multi-view observations and allocate capacity where it is most useful. To achieve this, we (1) use unified, scene-centric 3D anchors to consolidate redundant multi-view tokens, and (2) learn the spatial allocation of these anchors end-to-end. Specifically, we use an end-to-end learned, geometry-grounded condenser that aggregates reference-view tokens into 3D anchors (Sec. 3.3). The condenser predicts geometry for both views and anchors and uses their geometric relationships to guide cross-attention. The resulting anchors have 3D positions, which further enable target-view frustum culling to reduce decoding cost (Sec. 3.4). Both the geometry and scene-complexity-aware anchor allocation emerge from the NVS objective (Sec. 4.6).

We evaluate GLoD-LVSM across datasets, reference-view counts and scene-token budgets. Quantitatively, GLoD-LVSM consistently achieves the best quality–token-count trade-off among the baselines. Qualitatively, the learned anchors emerge near scene surfaces and allocate higher density to regions with richer texture, suggesting that NVS supervision and compression alone can induce a geometry- and complexity-aware efficient scene representation. Beyond rendering, we further evaluate the learned representation on scene-semantic understanding, demonstrating its potential as a 3D scene representation for broader downstream tasks. Our contributions are threefold:

- **Continuous-LoD scene representation.** We introduce a scene-conditioned continuous query field that enables flexible scene anchor budgets independent of the reference-view count.
- **Geometry-grounded scene condensation.** We learn scene-centric 3D anchors end-to-end, enabling efficient multi-view consolidation and scene-adaptive capacity allocation.
- **Efficient and generalizable scene anchors.** GLoD-LVSM achieves the best quality–scene-token trade-off for rendering and supports scene semantic understanding.

2 RELATED WORK

2.1 NOVEL VIEW SYNTHESIS

Novel view synthesis (NVS) is a widely studied problem and can be broadly categorized into per-scene optimization methods and feed-forward generalizable methods. NeRF (Mildenhall et al., 2020)

and 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) achieve photorealistic novel-view synthesis by optimizing a radiance field or a set of Gaussian primitives for each scene. However, they typically require time-consuming per-scene optimization and dense image capture to achieve high rendering quality. In contrast, generalizable NVS methods (Sajjadi et al., 2022; Chen et al., 2024; Xu et al., 2025), including recent Transformer-based approaches such as LVSMs (Jin et al., 2025; Kim et al., 2026; Sun et al., 2026; Szymanowicz et al., 2026), learn reconstruction priors from large datasets and can reconstruct unseen scenes from sparse input views in a single feed-forward pass, avoiding costly dense image capture and per-scene optimization. Our work builds on Transformer-based generalizable NVS and focuses on learning a compact, geometry-grounded scene representation that reduces redundancy among scene tokens, thereby lowering storage and rendering costs.

2.2 Compact Scene Representation

Compact scene representations have been widely studied for efficient NVS. For example, 3DGS methods prune or compress Gaussian primitives to reduce representation size while preserving visual quality (Fan et al., 2024; Fang & Wang, 2024; Lin et al., 2025). Some methods further support multiple or continuous levels of detail (CLoD), allowing representation complexity to adapt to runtime rendering budgets (Milef et al., 2025; Cheng et al., 2026). Compact scene-token representations have also been explored for LVSMs. LVSM-ED (Jin et al., 2025) uses a fixed latent bottleneck and cannot support flexible compression rates. CLiFT (Wang et al., 2025) uses K-means to select representative reference-view tokens, relying on handcrafted clustering that may be suboptimal. Our work also targets compact scene representations for LVSMs. Unlike prior methods, we support CLoD and learn scene anchors end-to-end, enabling flexible compression with less handcrafted design.

3 GLoD-LVSM: Geometry-Grounded CLoD Scene Representation

Fig. 2 illustrates GLoD-LVSM. Given V reference images and their camera parameters, GLoD-LVSM constructs a compact scene representation with a chosen budget of K anchors. A highway scene encoder first extracts reference-view tokens (Sec. 3.1). Our anchor initializer then generates K anchors from a continuous field (Sec. 3.2). These anchors are passed to our condenser, which produces scene-centric 3D anchors (Sec. 3.3). The anchors will be used to render the RGB images given the target views (Sec. 3.4). We train GLoD-LVSM end-to-end using the NVS loss (Sec. 3.5).

3.1 Highway Scene Encoder

We use the encoder from LagerNVS (Szymanowicz et al., 2026). The inputs are $\mathcal{X} = \{(\mathbf{I}_v, \mathcal{P}_v)\}_{v=1}^{V}$, where $\mathbf{I}_v$ is the v-th reference image and $\mathcal{P}_v$ is its camera parameters. The encoder patchifies each reference image, embeds each patch as a token, embeds $\mathcal{P}_v$ into a camera token $\mathbf{t}_v^{\text{cam}}$, and appends four register tokens $\mathbf{t}_{1:4}^{\text{reg}}$ to each view (Darcet et al., 2024). For each view, input token sequence $\mathbf{T}_v^{\text{in}} = [\mathbf{t}_v^{\text{cam}}, \mathbf{t}_{1:4}^{\text{reg}}, \text{PatchEmbed}(\mathbf{I}_v)]$. The resulting tokens are jointly encoded across all V views:

$$\mathbf{H} = \text{Encoder}(\{\mathbf{T}_v^{\text{in}}\}_{v=1}^{V}), \qquad \boldsymbol{Z} = \mathbf{H}_{\text{patch}}\mathbf{W} \in \mathbb{R}^{N \times D}. \tag{1}$$

Here, $\mathbf{W}$ linearly maps the encoded patch tokens to a defined D-dimensional space, and N is the total number of input patches across all views. We refer to the output $\boldsymbol{Z}$ as *reference-view tokens* and retain the encoded register and camera tokens for anchor initialization (Sec. 3.2). This one-patch-one-token highway design causes storage and decoding costs to grow with reference-view count.

3.2 Contextualized Continuous Field for Anchor Initialization

Given reference-view tokens $\boldsymbol{Z}$, our condenser compresses them into scene anchors. The first challenge is anchor initialization. Fixed learned vectors (Jin et al., 2025) support only a predefined anchor count. Clustering-based initialization (Wang et al., 2025) supports variable counts, but relies on handcrafted clustering that may be suboptimal and remains image-centric rather than scene-centric. We instead learn a scene-conditioned continuous initialization field end-to-end with minimal handcrafted design (Fig. 2(a)). Sampling the field at different densities yields different anchor budgets, enabling continuous levels of detail under varying storage and rendering constraints.

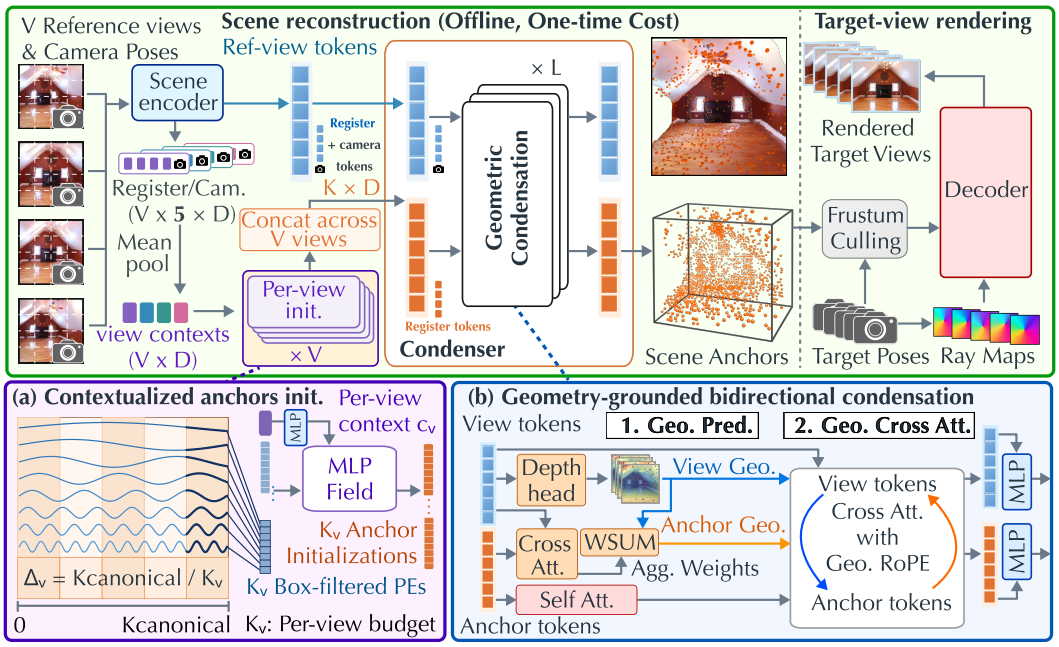

Fig. 2: Overview of GLoD-LVSM. A highway encoder extracts reference-view tokens and per-view context. (a) A scene-conditioned field uses box-filtered positional encodings to initialize the chosen number of anchors. (b) The condenser predicts geometry for both reference-view tokens and anchors and uses geometry-grounded bidirectional attention to condense reference-view tokens into anchors. The same scene anchors are reused to render different target views.

Continuous field parameterization. Given a total anchor budget K, we distribute K evenly across the V reference views, assigning K_v anchors to view v. For each view, our goal is to learn a D-dimensional continuous field that maps query indices $i \in \{0, \dots, K_v - 1\}$ to K_v vectors in the D-dimensional anchor space, with D set to the highway model's scene-token dimension. We parameterize this continuous field as an MLP conditioned jointly on the anchor indices and the current scene/view context. Specifically, each anchor index i is mapped to a continuous coordinate $u_{v,i}$ and encoded with our anti-aliased positional encoding $\bar{\gamma}(u_{v,i})$ (Eqn. 3). To condition the field on the current input, we mean-pool the encoded registers and camera tokens (Sec. 3.1) of view v and map them through $\mathrm{MLP}_{\text{context}}$ to obtain a view- and scene-specific context vector $\mathbf{c}_v$. $\bar{\gamma}(u_{v,i})$ and $\mathbf{c}_v$ are then concatenated and passed to $\mathrm{MLP}_{\text{field}}$ to produce the D-dimensional anchor initialization $\boldsymbol{a}_{v,i}$:

$$\boldsymbol{a}_{v,i} = \mathrm{MLP}_{\text{field}}\left([\bar{\gamma}(u_{v,i}), \mathbf{c}_v]\right), \qquad \mathbf{c}_v = \mathrm{MLP}_{\text{context}}\left(\mathrm{MeanPool}([\mathbf{H}^{\text{reg}}_{v,1:4}, \mathbf{H}^{\text{cam}}_{v}])\right). \quad (2)$$

Finally, we concatenate the initializations from all views to obtain $K = \sum_v K_v$ anchor initializations and feed them to the condenser. The field is optimized end-to-end through the NVS objective.

Anti-Aliased Positional Encoding Across Anchor Budgets. To sample the same continuous field at different anchor budgets, we map each discrete query index i to a continuous coordinate $u_{v,i}$ in a fixed canonical domain $[0, K_{\text{canonical}}]$, where $K_{\text{canonical}} = 2000$. For each view with budget K_v, we divide the domain into K_v intervals. Changing K_v therefore changes the sampling density while preserving coverage of the domain. To help the field capture high-frequency variation, we follow NeRF-style positional encoding (PE) (Mildenhall et al., 2020) and encode each coordinate as $\gamma(u) = \frac{1}{\sqrt{F}}[\cos(2\pi \mathbf{f} u), \sin(2\pi \mathbf{f} u)] \in \mathbb{R}^{2F}$. Here, $\mathbf{f} = (f_0, \dots, f_{F-1})$ is logarithmically spaced between $f_{\min} = 1/20000$ and $f_{\max} = 0.5$. We set $F = D/2$, where D is the dimension of anchor space. At small budgets, sparse sampling can alias the high-frequency components of PE. We therefore integrate PE over the sampling interval, analogous to box filtering (Barron et al., 2021):

$$\bar{\gamma}_{v,i} = \frac{1}{\Delta_v} \int_{I_{v,i}} \gamma(u)\, du, \quad I_{v,i} = [u_{v,i}, u_{v,i} + \Delta_v], \quad u_{v,i} = i\Delta_v, \quad \Delta_v = \frac{K_{\text{canonical}}}{K_v}. \quad (3)$$

As the anchor budget decreases, the intervals become wider and more strongly attenuate high-frequency components. The sine and cosine integrals have closed-form solutions, given in Supple-

mentary A.1. The resulting $\bar{\gamma}(u_{v,i}) \in \mathbb{R}^D$ provides a D-dimensional encoding of each anchor index whose frequency content reflects the anchor budget, with lower budgets inducing stronger attenuation.

3.3 Geometry-Grounded Bidirectional Condensation

As illustrated in Fig. 2(b), our condenser maps K initialized anchors and N reference-view tokens $\boldsymbol{Z}$ to K compact scene anchors. It uses $L = 8$ bidirectional Transformer blocks (Szymanowicz et al., 2026), where the two streams attend to each other. Content-only attention can mix spatially unrelated information because it lacks 3D awareness. We therefore predict geometry for both streams (Sec. 3.3.1) and use it to modulate the bidirectional cross-attention (Sec. 3.3.2). In the condenser, we retain the reference-view register and camera tokens and add four anchor registers. These tokens also participate in attention, but with no geometry modulation.

3.3.1 Predicting View and Anchor Geometry

View Geometry. We predict geometry for each reference-view token directly from its feature. Following RayRoPE (Wu et al., 2026), a linear depth head maps token $\boldsymbol{z}_j$ to a log-depth center μ_j and an uncertainty scale σ_j. Using the known camera parameters, we back-project the four corners of the corresponding image patch at depth $d_j = e^{\mu_j}$ to obtain its spatial span in 3D space $\mathbf{p}_{j,1:4}$. We also use the unit center-ray direction $\mathbf{v}_j$ to encode its viewing direction. Together, $\{\mathbf{p}_{j,1:4}, \mathbf{v}_j, \sigma_j\}$ describes the spatial support and viewing direction of reference-view token j. The geometry is recomputed from the updated reference-view features at every condenser layer.

Anchor Geometry. Rather than directly regressing unconstrained 3D coordinates for each anchor, we derive its geometry by aggregating the geometry of the reference-view tokens. This introduces a useful prior: it constrains anchors to the convex hull of the observed geometry rather than allowing unconstrained 3D positions. Specifically, we use a lightweight single-head cross-attention to compute aggregation weights β_{ij} between anchor i and reference-view token j:

$$\beta_{ij} = \mathrm{softmax}_j\left(\frac{\hat{\mathbf{q}}_i^\top \hat{\mathbf{k}}_j}{\sqrt{64}}\right), \qquad \mathbf{p}_i^A = \sum_j \beta_{ij}\bar{\mathbf{p}}_j, \qquad \mathbf{v}_i^A = \sum_j \beta_{ij}\mathbf{v}_j, \tag{4}$$

where $\hat{\mathbf{q}}_i$ and $\hat{\mathbf{k}}_j$ are 64-dimensional query and key projections, and $\bar{\mathbf{p}}_j = \frac{1}{4}\sum_{c=1}^{4} \mathbf{p}_{j,c}$ is the center of the back-projected view-token patch. The same weights aggregate both position and viewing direction, giving each anchor a 3D position $\mathbf{p}_i^A$ and a viewing-direction descriptor $\mathbf{v}_i^A$. β_{ij} is normalized over the reference-view tokens. Importantly, we use the anchor geometry predicted at the final condenser block as the anchor's final geometry for subsequent rendering.

3.3.2 Geometry-Grounded Bidirectional Attention

After geometry prediction, each condenser block first applies self-attention to the anchor stream, followed by bidirectional cross-attention between anchors and reference-view tokens and MLPs. We inject the inferred geometry into this cross-attention using geometry-dependent RoPE.

We define the geometric descriptors for the two token streams as follows. A reference-view token j corresponds to an image patch and is represented by its four back-projected patch corners $\mathbf{p}_{j,1:4}$, viewing direction $\mathbf{v}_j$, and depth uncertainty σ_j. An anchor i, in contrast, has a single 3D position $\mathbf{p}_i^A$ and viewing-direction descriptor $\mathbf{v}_i^A$. To match the geometric layout of the reference-view tokens, we repeat $\mathbf{p}_i^A$ four times for each anchor. Following RayRoPE (Wu et al., 2026), we represent a 3D point using $\pi(\mathbf{p})$, which first transforms the point into the reference camera frame to obtain (X, Y, Z), then computes $\pi(\mathbf{p}) = (X/r, Y/r, 1/|Z|)$ with $r = \sqrt{X^2 + Y^2 + Z^2}$. The four points therefore provide 12 coordinates. Appending the 3D viewing-direction descriptor gives a 15-dimensional geometric descriptor for both reference-view tokens and anchors:

$$\mathbf{g}_j^R = [\pi(\mathbf{p}_{j,1}), \ldots, \pi(\mathbf{p}_{j,4}), \mathbf{v}_j], \qquad \mathbf{g}_i^A = [\pi(\mathbf{p}_i^A), \ldots, \pi(\mathbf{p}_i^A), \mathbf{v}_i^A] \in \mathbb{R}^{15}. \tag{5}$$

For reference-view token geometry $\mathbf{g}_j^R$, we use RayRoPE's expected RoPE as encoding to account for the predicted depth uncertainty σ_j (Sec. 3.3.1). For anchor geometry $\mathbf{g}_i^A$, we use standard RoPE

encoding (Su et al., 2024). The encodings are then used to rotate the query, key, value, and output features in the bidirectional cross-attention, following the rotation mechanism in RayRoPE.

A key challenge is choosing the RoPE coordinate frame. When an anchor interacts with tokens from a reference view, we express RoPE in that view's frame for both cross-attention directions. This avoids dependence on the arbitrary world coordinate system, similar to conventional graphics pipelines that transform scene geometry into camera space before rasterization (Akenine-Möller et al., 2016).

3.4 Runtime Culling and Rendering

Culling. After scene reconstruction, the same set of anchors can be reused across different target views. We use the anchor geometry from the final condenser layer as the geometry of each anchor. Given a target camera $\mathcal{P}_t$, we project each anchor position $\mathbf{p}_i^A$ into the target camera frame and obtain its normalized image coordinate (u_i^t, v_i^t) and camera-space depth z_i^t. We retain anchors that lie in front of the camera and inside a slightly padded image region, $\mathcal{K}_t = \{i : z_i^t > 0,\ (u_i^t, v_i^t) \in [-\delta, 1+\delta]^2\}$, with $\delta = 0.1$. The padding retains anchors close to the image boundary. Only anchors in $\mathcal{K}_t$ are passed to the decoder, reducing the number of scene tokens processed for each target view.

Novel-View Rendering. We use the bidirectional decoder of LagerNVS (Szymanowicz et al., 2026), modified with our geometry-grounded RoPE for anchor–target-token cross-attention (Sec. 3.3.2). The target camera is converted into a ray map and patch-embedded into target tokens, which predict their view geometry with a linear head (as for reference-view tokens) and interact with anchors through bidirectional cross-attention. Anchors use the fixed geometry from the final condenser layer. The target camera frame is used as the RoPE coordinate frame. Finally, a linear projection maps target tokens to RGB patches, which form the rendered image $\hat{\mathbf{I}}_t$.

3.5 Training

We freeze the pretrained scene encoder and jointly optimize the continuous query field, condenser, and decoder. The decoder is supervised by the LagerNVS rendering loss $\mathcal{L}_{\text{render}}$ (Szymanowicz et al., 2026). We additionally regularize the predicted depth uncertainty σ_j^ℓ of reference-view token j at condenser layer ℓ (Sec. 3.3.1) to encourage sharp, localized geometry:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \lambda_\sigma \mathcal{L}_{\text{decay}}, \qquad \mathcal{L}_{\text{decay}} = \frac{1}{LN} \sum_{\ell=0}^{L-1} \sum_{j=1}^{N} |\sigma_j^\ell|, \qquad \lambda_\sigma = 10^{-3}. \tag{6}$$

We first train without culling, then enable culling and perform culling-aware fine-tuning. During training, we sample 1–10 reference views and uniformly sample the anchor budget from 256 to the corresponding highway-token count N. No explicit depth or anchor-position supervision is used.

4 Evaluation

4.1 Experimental Setup

Datasets. We evaluate on RealEstate10K (Zhou et al., 2018) and DL3DV (Ling et al., 2024) at resolutions of 256×256 and 256×448, respectively. We study varying-reference-view settings from 2 to 10 views. Since the official protocols cover only 2 views for RealEstate10K and 2/4/6 views for DL3DV, we construct the higher-view settings using the method described in Supplementary A.2.

Training. We fine-tune the pretrained general LagerNVS checkpoint (Szymanowicz et al., 2026) on each dataset using its 1–10-view training protocol, obtaining LagerNVS-FT, which is our base model. We initialize our decoder from LagerNVS-FT and train the condenser and decoder, followed by culling-aware fine-tuning. See Supplementary A.3 for training details.

Baselines. We compare against two groups of baselines:

- **System-level baselines.** We compare GLoD-LVSM against CLiFT (Wang et al., 2025), LVSM (Jin et al., 2025), MVSplat (Chen et al., 2024), and DepthSplat (Xu et al., 2025).

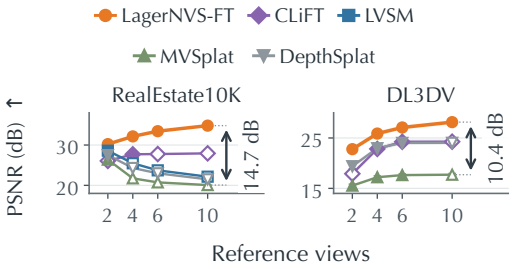

Fig. 3: PSNR across datasets and view counts. Shaded bands show 95% CI.

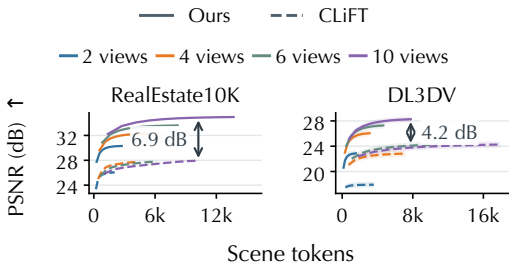

Fig. 4: PSNR vs. scene-token counts. Shaded bands show 95% CI.

- **Condenser baselines.** Alternative condensers using the same base model and training schedule:
  - RANDOM PRUNING: randomly selects scene tokens to match the desired budget.
  - K-MEANS PRUNING: sets the number of clusters to the desired token budget, clusters the reference-view token features, and retains only the token closest to each cluster centroid.
  - LEARNED PRUNING: uses a trained token pruner to select 90% of the token budget, then fills the remaining 10% through feature-diversity-based sampling (Takezoe et al., 2026).
  - CLIFT-STYLE: uses a K-means-based Transformer condenser adapted from CLIFT (Wang et al., 2025), with the number of blocks increased to eight to match our condenser.

Metrics. We report novel-view PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) under different reference view counts and scene token budgets. For all quality metrics, we report the mean and 95% confidence intervals (CIs) from 2,000 scene-level resamples with replacement.

4.2 COMPARING AGAINST OTHER SYSTEMS

Our base model is strong. To show that we build on a strong highway backbone, we first compare our base model, LAGERNVS-FT, against other systems. Fig. 3 shows the results, where filled markers denote view counts seen during training and hollow markers denote unseen counts. LAGERNVS-FT consistently achieves substantially higher PSNR than all compared systems[1]; SSIM and LPIPS show similar trends (Supplementary Fig. S4). It also improves as more reference views are provided. [2]

GLOD-LVSM achieves the best quality–token trade-offs. We compare the quality–scene-token trade-off with CLIFT because all other systems do not support flexible scene-token counts. Fig. 4 shows the results for different reference views. GLOD-LVSM achieves a better PSNR–token-count trade-off than CLIFT. The advantage comes from both our strong base model and our better condenser design. SSIM and LPIPS show similar trends (see Supplementary Fig. S3).

4.3 COMPARING AGAINST DIFFERENT CONDENSER DESIGNS

To isolate the effect of the strong base model, we conduct controlled experiments using the same base model and training schedule across different condenser designs. Because the baseline condensers do not produce geometry-grounded tokens, applying frustum-culling is not straightforward. For a fair comparison, we disable culling for all methods and report the compressed scene-token count. Fig. 5 compares quality–token trade-offs with 2 and 6 reference views on both datasets; 4- and 10-view results are shown in Fig. 1. Our condenser achieves a better PSNR–token-count trade-off across datasets and reference-view counts. SSIM and LPIPS show similar trends (see Supplementary Figs. S5–S8). We also visualize novel-view predictions from all condensers in Fig. 6, which visually confirms our method's quality advantage. More visualizations across scenes, reference-view counts, and compression rates are provided in Supplementary Figs. S13–S25.

[1] LVSM is evaluated only on RealEstate10K because no official DL3DV checkpoint is available.

[2] DL3DV results in Fig. 3 are not directly comparable across reference-view counts because the official target set is defined as the complement of the reference views and therefore changes slightly with view count.

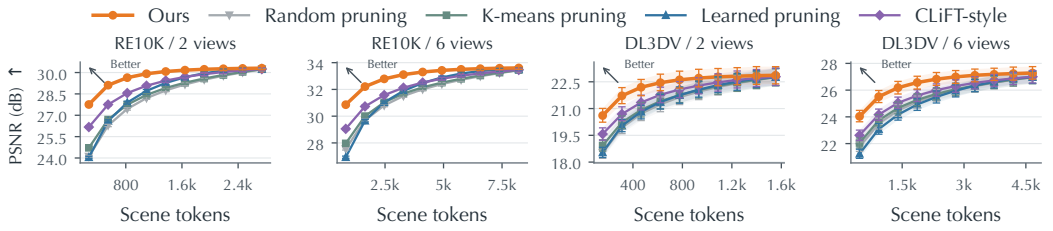

Fig. 5: Comparison of condenser quality–token trade-offs using the same base model. Left: RealEstate10K with 2 and 6 reference views. Right: DL3DV with 2 and 6 reference views. The x-axis denotes the pre-culling scene-token count. Shaded bands indicate 95% confidence intervals.

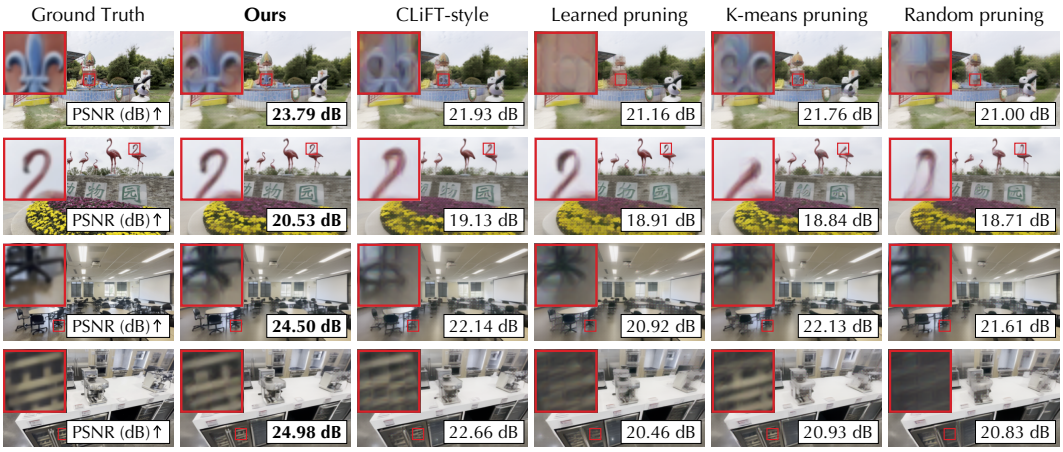

Fig. 6: Visualization of novel-view predictions from different condensers using two reference views at the same budget of 311 scene tokens (80% reduction), shown on four DL3DV scenes.

4.4 Token Reduction & Rendering Speedup over the Base Model

We now evaluate the token reduction and rendering speedup against the base model. Efficiency gains must be considered in the context of rendering quality. Using LaGeRNVS-FT as the quality reference, we measure scene-token reduction under PSNR losses of $X \in \{2, 1, 0.5, 0\}$ dB. Fig. 7 shows the results. On 10-view RealEstate10K, GLoD-LVSM reduces the scene-token count by 85.5% at a 2 dB quality loss. Even with no quality loss (0 dB), it still reduces the token count by 33.4%, which we attribute to the more effective geometry-grounded representation. The corresponding rendering speedups are $3.00\times$ and $1.23\times$ over LaGeRNVS-FT, respectively. With culling, the estimated speedups further increase to $3.20\times$ and $1.55\times$. More speed results and evaluation details are provided in Supplementary Fig. S1 and Supplementary A.5.

4.5 Scaling Quality with Views and Beyond the Highway Token Count

We now highlight two properties of GLoD-LVSM. First, unlike highway designs whose scene tokens grow with reference-view count, GLoD-LVSM can exploit more views at a *fixed scene-token budget*. Fig. 8 shows that, at a fixed token budget, GLoD-LVSM generally improves with more views. For example, on RealEstate10K with a 2048-token budget, 10 views outperform 2 views by 2.8 dB. In contrast, a 10-view highway model requires over 13K tokens. At small capacity, such as 512 tokens, gains saturate/degrade beyond 6 views. See Supplementary Fig. S9 for more results.

Second, because GLoD-LVSM learns a continuous anchor-initialization field (Sec. 3.2), it can sample more anchors (K) than the original highway representation tokens (N), which baseline condensers cannot support. Fig. 9 reports the PSNR gain relative to GLoD-LVSM at the uncompressed budget. Although GLoD-LVSM exceeds the base-model quality at the same token budget, its quality continues to improve for $K > N$. See Supplementary Fig. S10 for more results.

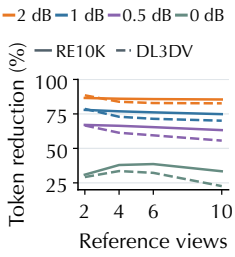

Fig. 7: Scene-token reductions at different PSNR loss budgets.

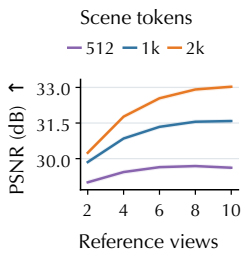

Fig. 8: Scaling quality with reference views on RealEstate10K.

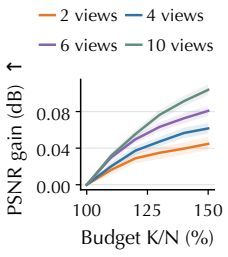

Fig. 9: Scaling beyond the base-model token budget on DL3DV.

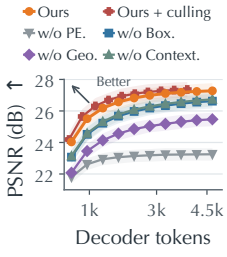

Fig. 10: DL3DV ablation studies using 6 reference views.

4.6 Emergent Geometry and Anchor Allocation

To understand GLoD-LVSM, we visualize its representation using four reference views with 2048 scene anchors (Fig. 1). Without explicit geometry supervision, consistent scene depth emerges across reference views, while the learned anchors concentrate near scene surfaces. Projecting the anchors back onto the reference images further reveals scene-adaptive allocation: anchors are denser in texture-rich regions and sparser in textureless regions. These results show that the NVS loss can induce an efficient geometry- and texture-aware scene representation. More visualizations are in Supplementary Figs. S26–S31. We process raw depth before point-cloud visualization; see Supplementary A.4.

4.7 Ablation Studies

Fig. 10 compares quality–decoder-token trade-offs for our ablations on 6-view DL3DV. Ours denotes GLoD-LVSM without culling. Ours + Culling enables our frustum-culling (Sec. 3.4). w/o Geo. removes geometry-dependent RoPE (Sec. 3.3.2) from both the condenser and decoder. w/o Box. removes anti-aliasing from the positional encoding (Eqn. 3). w/o PE. removes positional encoding and instead feeds normalized anchor indices to the field MLP. w/o Context.[3] removes scene- and view-specific conditioning from the anchor initialization field (Eqn. 2). The results show that each component contributes to the quality–token trade-off of GLoD-LVSM. Results with other reference-view counts show similar trends and are provided in Supplementary Figs. S11–S12.

4.8 GLoD-LVSM Supports Scene Semantic Understanding

We further examine whether the learned anchors capture scene semantics beyond rendering. We use VLM-generated annotations for RealEstate10K (He et al., 2025; Li et al., 2023) to categorize scenes into eight semantic classes. For each scene, GLoD-LVSM compresses 10 reference views into 1,024 anchors, which are mean-pooled into a single 768-dimensional *scene vector*. We train only a linear probe on this vector for scene classification and achieve 80.40%/91.73% top-1/top-2 accuracy. This result shows that learned anchors already contain scene semantics. See Supplementary A.6 for evaluation protocol and Supplementary Fig. S2 for the confusion matrix.

5 Conclusion

We propose GLoD-LVSM, a compact, geometry-grounded LVSM representation with continuous levels of detail. GLoD-LVSM achieves a strong quality–token trade-off through two key designs: (1) an end-to-end learned continuous field with minimal handcrafted design, and (2) a condenser that consolidates multi-view observations into scene-centric, geometry-grounded anchors. Trained with the NVS objective, GLoD-LVSM learns geometry- and texture-aware anchor allocation. We further show its utility on scene semantic understanding, highlighting the potential beyond rendering.

[3]Removing the context vector makes the per-view initialization identical across views. We therefore replace the per-view fields with a single global initialization field and set $K_{\text{canonical}} = 20{,}000$.

AI USE STATEMENT

We did not use generative AI tools to generate experimental results or independently derive scientific claims. We used AI tools for language polishing. All AI-assisted content was reviewed and verified by the authors.

REPRODUCIBILITY STATEMENT

The main paper and supplementary material provide details of the method, training procedure, and evaluation setup. We also plan to release the training and evaluation code, model checkpoints, and relevant configuration files upon acceptance.

ETHICS STATEMENT

We use only public datasets and conduct no new data collection or human-subject studies; we are not aware of additional ethical concerns specific to our method.

REFERENCES

T. Akenine-Möller, E. Haines, and N. Hoffman. *Real-Time Rendering*. CRC Press, 2016. ISBN 9781498785631. URL https://books.google.com/books?id=Xcb1CwAAQBAJ.

Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5835–5844. IEEE, 2021.

Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *European conference on computer vision*, pp. 370–386. Springer, 2024.

Zhigang Cheng, Mingchao Sun, Liu Yu, Zengye Ge, Luyang Tang, Mu Xu, Yangyan Li, and Peng Pan. Clod-gs: Continuous level-of-detail via 3d gaussian splatting. In *International Conference on Learning Representations*, volume 2026, pp. 89275–89299, 2026.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International conference on learning representations*, volume 2024, pp. 2632–2652, 2024.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *Advances in neural information processing systems*, 37:140138–140158, 2024.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In *European conference on computer vision*, pp. 165–181. Springer, 2024.

Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for video diffusion models. In *International Conference on Learning Representations*, volume 2025, pp. 100433–100464, 2025.

Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. Lvsm: A large view synthesis model with minimal 3d inductive bias. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=QQBPWtvtcn.

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

Evan Kim, Hyunwoo Ryu, Thomas W Mitchel, and Vincent Sitzmann. Scaling view synthesis transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28893–28902, 2026.

Dongxu Li, Junnan Li, Hung Le, Guangsen Wang, Silvio Savarese, and Steven C.H. Hoi. LAVIS: A one-stop library for language-vision intelligence. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 31–41, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.acl-demo.3.

Weikai Lin, Yu Feng, and Yuhao Zhu. Metasapiens: Real-time neural rendering with efficiency-aware pruning and accelerated foveated rendering. In *Proceedings of the 30th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 1*, pp. 669–682, 2025.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22160–22169. IEEE, 2024.

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 *ECCV*, 2020.

Nicholas Milef, Dario Seyb, Todd Keeler, Thu Nguyen-Phuoc, A Božič, Sushant Kondguli, and Carl Marshall. Learning fast 3d gaussian splatting rendering using continuous level of detail. In *Computer Graphics Forum*, volume 44, pp. e70069. Wiley Online Library, 2025.

Mehdi SM Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lučić, Daniel Duckworth, Alexey Dosovitskiy, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6219–6228. IEEE, 2022.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Cheng Sun, Jaesung Choe, Min-Hung Chen, Ryo Hachiuma, and Yu-Chiang Frank Wang. Dvsm: Decoder-only view synthesis model done right. *arXiv preprint arXiv:2605.29891*, 2026.

Stanislaw Szymanowicz, Minghao Chen, Jianyuan Wang, Christian Rupprecht, and Andrea Vedaldi. Lagernvs: Latent geometry for fully neural real-time novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Rinyoichi Takezoe, Yaqian Li, Zi-Hao Bo, Anzhou Hou, Mo Guang, and Kaiwen Long. Learnpruner: Rethinking attention-based token pruning in vision language models. In *International Conference on Learning Representations*, volume 2026, pp. 66381–66400, 2026.

Zhengqing Wang, Yuefan Wu, Jiacheng Chen, Fuyang Zhang, and Yasutaka Furukawa. Clift: Compressive light-field tokens for compute efficient and adaptive neural rendering. *Advances in Neural Information Processing Systems*, 38:8759–8787, 2025.

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

Yu Wu, Minsik Jeon, Jen-Hao Rick Chang, Oncel Tuzel, and Shubham Tulsiani. Rayrope: Projective ray positional encoding for multi-view attention. In *European Conference on Computer Vision (ECCV)*, 2026.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16453–16463. IEEE, 2025.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 586–595. IEEE, 2018.

Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018.