DVSM: THE SIMPLEST DECODER-ONLY VIEW SYNTHESIS MODEL PERFORMS BEST

Anonymous authors
Paper under double-blind review

ABSTRACT

Regression-based view-synthesis models have explored encoder-decoder, decoder-only, and foundation-model designs, but it remains unclear which composition is most effective. We revisit this design space through controlled experiments and find that the simplest choice works best: one decoder should both reconstruct a reusable scene state and render novel views. DVSM prefills a per-layer KV cache from the context views, then renders each target through unidirectional cross-attention using the same weights. Every tested form of weight decoupling reduces quality despite increasing the parameter count. Our feature-space analysis provides a consistent explanation: for the same viewpoints, complete weight sharing keeps reconstruction and camera-only rendering features aligned and enables the renderer to recover the corresponding oracle features. Foundation features can be injected during context prefill as an orthogonal enhancement. DVSM establishes state-of-the-art results on DL3DV and zero-shot out-of-distribution benchmarks, outperforming both geometry-free and geometry-based baselines without explicit 3D inductive biases. On most benchmarks, our best-performing variant is also the smallest model, showing that the gains come from our design rather than model scale.

1 INTRODUCTION

Novel view synthesis renders images from unobserved viewpoints given a set of input views. Traditional methods first reconstruct an explicit 3D representation, such as a mesh, a neural radiance field (NeRF), or 3D Gaussian Splatting (3DGS), and then render target views through a graphics-based operation (Mildenhall et al., 2020; Kerbl et al., 2023). Large Reconstruction Models remove per-scene optimization by predicting these 3D representations with a feed-forward network (Hong et al., 2024b).

Regression-based view-synthesis models take a more direct approach: a Transformer predicts target images with few explicit 3D assumptions (Jin et al., 2025). Recent methods have explored several ways to compose their familiar building blocks, including global or per-view context encoders, final latent tokens or per-layer KV caches, and joint, bidirectional, or unidirectional novel-view decoding (Jin et al., 2025; Jia et al., 2026; Kim et al., 2026; Szymanowicz et al., 2026). These choices are often evaluated together with changes in model size, training, or pretrained backbones, making it difficult to determine which architectural decisions are actually responsible for the improvement.

We therefore ask a deliberately simple empirical question: which composition works best? Rather than introducing a new Transformer primitive, we compare these choices under controlled settings. Our experiments identify a simple design: encode all context views with global attention, store the reconstructed scene as a per-layer KV cache, keep this cache fixed during unidirectional novel-view decoding, and share all weights between the two stages. We instantiate these findings in DVSM, a decoder-only View Synthesis Model. DVSM processes the context once to prefill the cache, then uses the same tokenizer and decoder to render each target view. We use the terms *reconstruction* and *rendering* for cache construction and cache querying, respectively; DVSM does not reconstruct an explicit 3D representation.

The controlled experiments consistently favor this design. Global cross-view attention is crucial for context encoding, and every tested form of weight decoupling reduces quality despite increasing the parameter count. We further analyze the retrieved features when reconstruction and rendering

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.

use the same viewpoints. Reconstruction observes RGB and camera information, while rendering receives only the cameras and must recover appearance from the cache. The reconstruction features therefore provide an oracle target. Complete weight sharing keeps the two feature spaces aligned and enables rendering to recover the corresponding oracle features, whereas separate encoder and decoder weights increasingly diverge in later layers. This analysis provides empirical evidence for the shared design rather than a formal connection to differentiable rendering.

Foundation features can be injected during context prefill as an orthogonal enhancement. The resulting models establish state-of-the-art performance on DL3DV and zero-shot out-of-distribution benchmarks, outperforming both geometry-free and geometry-based baselines without explicit 3D inductive biases. On most benchmarks, our best-performing variant is also the smallest model in the comparison. The improvement therefore comes from the design rather than model scale.

Our contributions are:

- We conduct a controlled empirical study of regression-based view-synthesis architectures and identify a simple, effective composition: global context encoding, a reusable per-layer KV cache, unidirectional novel-view decoding, and complete weight sharing.
- We show that every tested form of weight decoupling reduces quality despite adding parameters. A feature-space analysis further shows that the shared model better aligns reconstruction and rendering and recovers the corresponding oracle features.
- DVSM achieves state-of-the-art results on DL3DV and zero-shot out-of-distribution benchmarks, including against geometry-based methods. On most benchmarks, its best-performing variant is also the smallest compared model, demonstrating that the gains come from our design rather than scale.

2 Related work

Per-scene optimization. Optimization-based methods dominate high-quality novel-view synthesis. They fit a scene representation by reproducing observed images through differentiable rendering, then render new viewpoints of the same scene. NeRF represents a scene with a coordinate-based MLP and uses volume rendering (Mildenhall et al., 2020; Max, 1995). Later work develops faster representations and optimization methods (Sun et al., 2022; Fridovich-Keil et al., 2022; Chen et al., 2022; Müller et al., 2022; Reiser et al., 2021). 3DGS combines Gaussian primitives with a CUDA rasterizer to improve both speed and quality (Kerbl et al., 2023; Zwicker et al., 2001). Other extensions use ray tracing or alternative scene primitives (Moenne-Loccoz et al., 2024; Sun et al., 2025a; von Lützow & Nießner, 2025; Govindarajan et al., 2025; Tobiasz et al., 2025). These methods work best with dense observations, and their iterative optimization can take minutes or hours. We instead study a feed-forward model that also works with sparse input views.

Geometry-based feed-forward models. Feed-forward methods learn reconstruction priors from training data. Early methods build cost volumes and predict multiplane images, often for forward-facing scenes (Yao et al., 2018; Wang et al., 2024; Guo et al., 2019; Yang et al., 2020; Gu et al., 2019; Mildenhall et al., 2019). Methods that predict NeRFs support a wider range of viewpoints (Chen et al., 2021; Yu et al., 2021; Wang et al., 2021; Liu et al., 2022). Following 3DGS, many models predict pixel-aligned Gaussian attributes (Kerbl et al., 2023; Charatan et al., 2024; Xu et al., 2025; Chen et al., 2024; Szymanowicz et al., 2024; Tang et al., 2024a). BTimer uses decoder-only LVSM features with KV caching as test-time augmentation for predicting explicit Gaussians; unlike DVSM, its final scene representation and renderer are geometry-based (Liang et al., 2025). Large Reconstruction Models (LRMs) use large Transformers to predict triplanes, meshes, or Gaussians (Hong et al., 2024b; Vaswani et al., 2017; Chan et al., 2021; Wei et al., 2024; Zhang et al., 2024). LongLRM, LongLRM++, and tttLRM scale to dozens of input views by using Mamba2 or test-time training layers (Chen et al., 2025b;a; Wang et al., 2026; Dao & Gu, 2024; Sun et al., 2025b; Zhang et al., 2025). These models are efficient, but errors in the predicted geometry can still limit their treatment of translucent materials and mirror reflections. Our geometry-free model instead learns to synthesize target images directly.

Geometry-free feed-forward models. Another line of work learns a neural rendering function without explicitly reconstructing 3D geometry. Some methods sample features from reference views along epipolar lines, while others predict color directly from camera rays (Du et al., 2023; Suhail

et al., 2022; 2021; Sitzmann et al., 2021; Rombach et al., 2021; Sajjadi et al., 2022). LVSM uses a large Transformer to improve model capacity and image quality (Jin et al., 2025). Efficient LVSM, LaCT, and SVSM further modify the architecture or scene representation (Jia et al., 2026; Zhang et al., 2025; Kim et al., 2026). LagerNVS initializes its encoder from a geometry-pretrained reconstruction model but renders directly from the resulting latent features (Szymanowicz et al., 2026). Other extensions handle unknown camera geometry (Jiang et al., 2025; Zhao et al., 2025; Mitchel et al., 2025; Wang et al., 2025a). We focus on the core LVSM architecture for calibrated images.

Camera-aware generative models. Recent generative models also support camera control (Kong et al., 2024; Liu et al., 2024; Long et al., 2024; Tang et al., 2023; 2024b; Zhou et al., 2025; Yu et al., 2024; Cao et al., 2025; Bai et al., 2025; Ren et al., 2025). Our setting differs in four ways: target views have no required temporal order, camera control is continuous and precise, reconstruction may use many context views, and target content is assumed to be observable from the context views. Because DVSM is regression-based, it does not generate unseen regions, but it provides more accurate interpolation within the observed scene. Combining LVSMs with generative models is an interesting direction for future work.

3 Approach

We first define the task and review the LVSM framework in Section 3.1. We then describe DVSM and the weight-sharing design in Section 3.2. Finally, we introduce foundation-feature injection in Section 3.4.

3.1 Preliminaries

Given calibrated input images, our task is to synthesize images at novel viewpoints. The input contains V images $\{\mathbf{I}_i \in \mathbb{R}^{H\times W\times 3}\}_{i=1}^{V}$, camera poses $\{\mathbf{P}_i \in \mathbb{SE}(3)\}_{i=1}^{V}$, and intrinsics $\{\mathbf{K}_i \in \mathbb{R}^{3\times 3}\}_{i=1}^{V}$. For each target camera $(\mathbf{P}', \mathbf{K}')$, the model synthesizes one novel view. Following common practice, we do not assume an ordering among target cameras, and each target view is rendered independently. We focus on reconstruction rather than generation: most target content is assumed to be observable in the input views.

Conventional methods recover an explicit 3D representation through per-scene optimization (*e.g.*, NeRF or 3DGS) or a generalizable network such as an LRM (Mildenhall et al., 2020; Kerbl et al., 2023; Hong et al., 2024b). LVSMs instead render target views with a feed-forward model and do not explicitly predict geometry (Jin et al., 2025). We write the reconstruction and rendering processes as

$$\bar{\mathbf{I}} = \mathrm{Rend}(\mathbf{P}', \mathbf{K}', S; \phi), \quad S = \mathrm{Recon}(\mathbf{I}, \mathbf{P}, \mathbf{K}; \theta), \tag{1}$$

where $\mathrm{Recon}(\cdot)$ and $\mathrm{Rend}(\cdot)$ are neural networks with parameters θ and ϕ, $\bar{\mathbf{I}}$ is the synthesized target view, and S is an implicit scene representation. Depending on the model, S may consist of tokens, a KV cache, or updated fast weights (Jin et al., 2025; Kim et al., 2026; Jia et al., 2026; Zhang et al., 2025). During training, we minimize the photometric and perceptual differences from the ground-truth target image $\mathbf{I}'$:

$$\mathcal{L} = \mathrm{MSE}(\bar{\mathbf{I}}, \mathbf{I}') + \lambda \cdot \mathrm{Percep}(\bar{\mathbf{I}}, \mathbf{I}'), \tag{2}$$

where λ weights the perceptual loss Percep (Johnson et al., 2016).

3.2 Sharing one decoder for reconstruction and rendering

Recent encoder-decoder LVSMs use separate Vision Transformers (ViTs) for $\mathrm{Recon}(\cdot)$ and $\mathrm{Rend}(\cdot)$. Our main empirical finding is that one decoder with weights shared across reconstruction and rendering—including the input patch embeddings—improves quality. We first describe DVSM, then compare it with related LVSM architectures (Jia et al., 2026; Kim et al., 2026; Dosovitskiy et al., 2021).

The reconstruction tokens combine color and camera information, whereas the rendering tokens contain only the target-camera information:

$$\mathbf{X}^{(\mathrm{recon})} = \mathrm{LayerNorm}\left(\mathrm{PE}^{(\mathrm{ray})}(\mathrm{Plücker}(\mathbf{P}, \mathbf{K})) + \mathrm{PE}^{(\mathrm{rgb})}(\mathbf{I})\right), \tag{3a}$$

$$\mathbf{X}^{(\mathrm{rend})} = \mathrm{LayerNorm}\left(\mathrm{PE}^{(\mathrm{ray})}(\mathrm{Plücker}(\mathbf{P}', \mathbf{K}'))\right). \tag{3b}$$

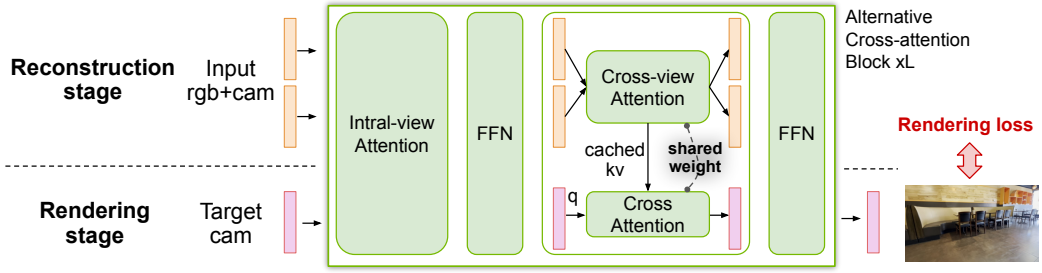

Figure 1: **Our architecture.** The reconstruction and rendering stages share all weights, including the input tokenizer. Reconstruction caches the keys and values from cross-view attention; rendering queries this cache with target-camera tokens.

Table 1: **Architecture comparison.** The scene representation is the target-independent state computed from the context views and reused across novel-view queries. M is the fixed number of latent tokens, P is the total number of patches across all context views, L is the number of layers, and D is the feature dimension.

Method	Context encoding	Scene repr.	Repr. size	Novel-view decoding	Shared weights
Encoder-Decoder LVSM	Global attn.	Latent tokens	$M \times D$	Joint full self-attn.	No
Decoder-Only LVSM	None	None	0	Joint full self-attn.	N/A
Efficient LVSM	Intra-view attn.	KV cache	$2L \times P \times D$	Unidirectional cross-attn.	No
SVSM	Global attn.	Latent tokens	$P \times D$	Unidirectional cross-attn.	No
LagerNVS	VGGT	Latent tokens	$P \times D$	Bidirectional cross-attn.	No
DVSM (ours)	Global attn.	KV cache	$2L \times P \times D$	Unidirectional cross-attn.	Yes

Here, $\mathrm{Plücker}(\cdot)$ converts the camera parameters into Plücker ray maps at the image resolution (Plucker, 1865). The functions $\mathrm{PE}^{(\mathrm{rgb})}(\cdot)$ and $\mathrm{PE}^{(\mathrm{ray})}(\cdot)$ are linear patch embeddings with patch size p. The context tokens are $\mathbf{X}^{(\mathrm{recon})} \in \mathbb{R}^{V \times \frac{HW}{p^2} \times D}$ for latent dimension D, and the target-view tokens are $\mathbf{X}^{(\mathrm{rend})} \in \mathbb{R}^{\frac{HW}{p^2} \times D}$. The two branches share the weights of $\mathrm{PE}^{(\mathrm{ray})}(\cdot)$ and $\mathrm{LayerNorm}(\cdot)$.

As shown in Figure 1, one decoder ViT processes both $\mathbf{X}^{(\mathrm{recon})}$ and $\mathbf{X}^{(\mathrm{rend})}$. It contains L repeated blocks. Following VGGT, each block alternates intra-view and cross-view attention, with an MLP after each attention layer (Wang et al., 2025b). We use residual connections and QK normalization throughout the decoder (He et al., 2016; Henry et al., 2020).

During reconstruction, we cache the keys and values from every cross-view attention layer (Pope et al., 2023). This cache acts as an implicit scene representation for later target-view queries. During rendering, $\mathbf{X}^{(\mathrm{rend})}$ passes through the same decoder. The only difference is that each cross-view attention layer queries the cached keys and values instead of recomputing them. Finally, the last-layer target tokens $\mathbf{X}_L^{(\mathrm{rend})}$ are mapped to an image by

$$\bar{\mathbf{I}} = \mathrm{PixShuf}(\mathrm{Linear}(\mathrm{LayerNorm}(\mathbf{X}_L^{(\mathrm{rend})}))) \,, \tag{4}$$

where pixel shuffle $\mathrm{PixShuf}(\cdot)$ rearranges each output token into an image patch (Shi et al., 2016).

3.3 Architectural Design Choices

Table 1 organizes the closest architectures by four design choices. We discuss each choice in turn rather than describing every method separately.

Context encoding. Our controlled ablation in Table 5 shows that global cross-view attention is crucial: replacing it with independent intra-view attention causes a large quality drop. We therefore retain global attention when encoding the context. A pretrained foundation encoder such as VGGT is a promising alternative because it provides strong 3D-aware features (Wang et al., 2025b; Szymanowicz et al., 2026). However, directly combining a large foundation encoder with our complete weight-sharing design would require running the same backbone again for every novel-view

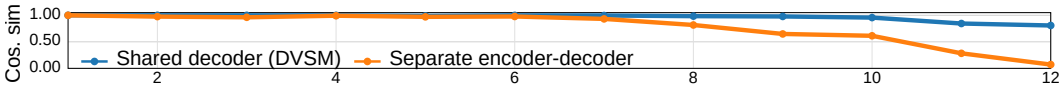

Figure 2: **Reconstruction–rendering feature alignment.** We use the same 32 viewpoints for both stages: reconstruction tokens receive RGB and camera information, while rendering tokens receive only the corresponding cameras and retrieve appearance from the reconstructed cache. At each cross-attention layer, we report the cosine similarity between the attended features softmax$(QK^T)V$ for corresponding tokens. Sharing the decoder weights keeps these features closely aligned, whereas separate encoder and decoder weights cause them to diverge in later layers.

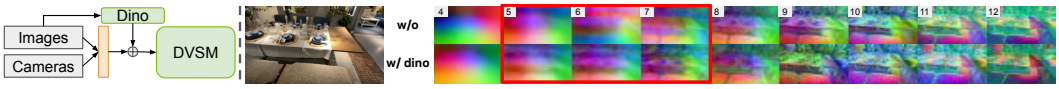

Figure 3: **Foundation feature injection.** We add DINOv3 features to the reconstruction tokens (left) (Sim'eoni et al., 2025). Target-view queries retrieve this information from the KV cache, producing sharper features in earlier rendering layers (red box).

query, which exceeds our resource budget. Integrating shared reconstruction and rendering into a VGGT-based NVS model is promising future work.

Scene representation. Most related methods summarize the context as a set of final latent tokens that become the input to a subsequent decoder (Jin et al., 2025; Kim et al., 2026; Szymanowicz et al., 2026). DVSM instead prefills the keys and values at every cross-view attention layer. This per-layer KV cache lets each target-view block read the corresponding reconstructed features directly, without first transforming a single final token set.

Novel-view decoding. Joint full self-attention and bidirectional cross-attention update the scene representation separately for every target-frame query (Jin et al., 2025; Szymanowicz et al., 2026). We view this repeated update as inefficient and unnecessary: conventional view-synthesis pipelines reconstruct a scene once and keep it fixed while rendering multiple viewpoints. DVSM follows the same principle through unidirectional cross-attention. Target tokens read the fixed scene cache, while the reconstructed scene representation is not modified.

Weight sharing. DVSM uses the same parameters for context encoding and novel-view decoding. This sharing does not require re-encoding the context: the cache is constructed once and queried by the same decoder for every target view. Our controlled ablations show that separating any tested component reduces quality despite adding parameters. To analyze why sharing helps, we reconstruct a scene from 32 views and then render the same viewpoints. Reconstruction tokens contain RGB and camera information, whereas rendering tokens contain only camera information and must retrieve appearance from the reconstructed cache. Because the queried views are identical to the reconstruction views, this is an oracle setting for testing whether rendering recovers the reconstruction features. At every cross-attention layer, we measure the cosine similarity between the retrieved features of corresponding reconstruction and rendering tokens. As shown in Figure 2, the shared decoder retrieves nearly identical features in the two stages through most of the network, whereas separate encoder and decoder weights increasingly diverge in later layers. One parameter set participates in both cache construction and cache querying, receives gradients through both computations, and keeps the two stages in the same feature space. In contrast, separate encoder and decoder parameters can specialize independently without a direct constraint that their intermediate features remain aligned. Together, the controlled ablations and feature-space analysis show that complete weight sharing improves both feature alignment and final rendering quality.

3.4 Foundation-feature injection

Visual foundation models have received limited study in LVSMs (Caron et al., 2021; Oquab et al., 2024; Sim'eoni et al., 2025; Ranzinger et al., 2023; Heinrich et al., 2024). We inject their pretrained features into the reconstruction branch of DVSM. Specifically, we extend the reconstruction patch embedding in Equation (3a) as

$$\mathbf{X}^{(\text{recon})} = \text{LayerNorm}\left(\text{PE}^{(\text{ray})}(\text{Plücker}(\mathbf{P}, \mathbf{K})) + \text{PE}^{(\text{rgb})}(\mathbf{I}) + \text{Prior}(\mathbf{I})\right), \tag{5}$$

Table 2: **Two input views on Re10K at 256×256 resolution.** Recon-time is the processing time per scene, and Rend-time is the rendering time per target frame, both measured on an H100 GPU. A darker tone is better.

Method	PSNR↑	SSIM↑	LPIPS↓	#params↓	Recon-time↓	Rend-time↓
LVSM (Jin et al., 2025)	28.58	0.893	0.114	165M	9.00ms	7.06ms
Dec.-only LVSM (Jin et al., 2025)	29.67	0.906	0.098	165M	–	11.47ms
Eff. LVSM (Jia et al., 2026)	28.93	0.895	0.102	199M	7.06ms	8.15ms
SVSM (Kim et al., 2026)	30.01	0.910	0.096	416M	9.74ms	9.05ms
LagerNVS (Szymanowicz et al., 2026)	31.39	0.928	0.078	997M	58.85ms	21.03ms
DVSM (ours) ps8	29.98	0.912	0.096	163M	8.08ms	6.90ms
DVSM (ours) ps8+dino	31.23	0.925	0.085	454M	17.63ms	7.14ms

Table 3: **Multi-view evaluation on DL3DV.** The short-sequence, low-resolution setting uses 2, 4, or 6 input views at 256×448, while the full-sequence, high-resolution setting uses 16, 32, or 64 input views at 540×960. Size is the model size in millions of parameters. Ctx. is the context-processing time per scene, and Rend. is the per-frame rendering time. All runtimes are measured without batching. A darker tone is better.

		2 input views (256×448)					4 input views (256×448)					6 input views (256×448)				
Method	Size	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓
Geometry-based methods																
MVSplat (Chen et al., 2024)	11M	17.54	0.529	0.402	100ms	3ms	21.63	0.721	0.233	150ms	3ms	22.93	0.775	0.193	230ms	4ms
DepthSplat (Xu et al., 2025)	115M	19.20	0.610	0.307	100ms	3ms	22.16	0.734	0.188	110ms	4ms	23.00	0.764	0.161	130ms	4ms
TokenGS (Ren et al., 2026)	211M	20.07	0.636	0.420	140ms	2ms	23.11	0.746	0.319	170ms	2ms	24.09	0.776	0.297	220ms	2ms
Geometry-free methods																
LVSM (Jin et al., 2025)	165M	21.55	0.678	0.287	25ms	18ms	26.00	0.825	0.159	44ms	18ms	27.42	0.857	0.137	67ms	18ms
LaCT (Zhang et al., 2025)	379M	17.64	0.516	0.433	205ms	53ms	23.41	0.753	0.212	333ms	53ms	27.29	0.864	0.135	475ms	53ms
LagerNVS (Szymanowicz et al., 2026)	1052M	20.97	0.635	0.269	919ms	17ms	22.89	0.705	0.188	910ms	17ms	23.51	0.726	0.165	926ms	18ms
DVSM (ours) ps16+dino	454M	21.72	0.677	0.281	86ms	10ms	27.03	0.851	0.134	44ms	10ms	28.90	0.890	0.105	47ms	10ms
DVSM (ours) ps8	163M	22.65	0.723	0.243	42ms	10ms	28.32	0.882	0.106	29ms	10ms	30.70	0.922	0.074	44ms	10ms
		16 input views (540×960)					32 input views (540×960)					64 input views (540×960)				
Method	Size	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓
Geometry-based methods																
3DGS (Kerbl et al., 2023)		21.20	0.708	0.264	> 600 s	< 5 ms	23.60	0.779	0.213	> 600 s	< 5 ms	26.43	0.854	0.167	> 600 s	< 5 ms
LongLRM (Chen et al., 2025b)	174M	21.30	0.713	0.332	0.8s	5ms	23.67	0.770	0.279	1.3s	7ms	23.65	0.787	0.267	2.8s	12ms
iLRM (Kang et al., 2026a)	177M	22.75	0.757	0.303	0.3s	5ms	24.12	0.794	0.265	0.8s	7ms	24.28	0.811	0.248	2.7s	11ms
tttLRM (Wang et al., 2026)	381M	23.59	0.783	0.258	2.6s	8ms	25.05	0.820	0.220	4.6s	12ms	25.92	0.842	0.200	9.1s	17ms
MVP (Kang et al., 2026b)	230M	23.58	0.789	0.248	0.1s	5ms	25.73	0.839	0.197	0.2s	7ms	27.44	0.873	0.165	0.4s	11ms
Geometry-free methods																
LVSM (Jin et al., 2025)	165M	21.64	0.666	0.365	3.8s	48ms	21.73	0.664	0.365	14.7s	48ms	21.46	0.651	0.377	59.4s	48ms
LaCT (Zhang et al., 2025)	379M	24.95	0.803	0.223	1.2s	53ms	27.22	0.845	0.184	2.3s	53ms	28.66	0.865	0.169	4.6s	53ms
LagerNVS (Szymanowicz et al., 2026)	1052M	21.73	0.663	0.349	1.8s	26ms	22.17	0.671	0.332	2.6s	41ms	21.91	0.661	0.330	4.4s	73ms
DVSM (ours) ps16+dino	454M	26.22	0.836	0.183	0.4s	18ms	28.21	0.875	0.144	0.9s	28ms	29.83	0.900	0.120	2.8s	48ms
DVSM (ours) ps8	163M	26.86	0.852	0.170	2.4s	143ms	29.13	0.893	0.129	8.8s	263ms	31.11	0.920	0.104	35.4s	505ms

where Prior$(\cdot)$ is a pretrained foundation model followed by a linear projection to the DVSM latent dimension. We retain $\mathrm{PE}^{(\mathrm{rgb})}(\cdot)$ because foundation models are often designed to be insensitive to color changes, while view synthesis requires accurate appearance. The prior is evaluated only during reconstruction, so it does not change the per-view rendering cost. Its features enrich the cached keys and values and help the rendering tokens recover sharper features in earlier layers, as shown in Figure 3.

4 Experiments

Implementation and evaluation details are provided in Section A.

4.1 Two-view evaluation

Benchmark. We evaluate two-view synthesis on Re10K, which contains 10K indoor and outdoor real-estate videos (Khakhulin et al., 2022). Following LVSM, we use the train-test split from pixelSplat (Jin et al., 2025; Charatan et al., 2024). The main comparison focuses on LVSM, decoder-only LVSM, Efficient LVSM, SVSM, and LagerNVS, while the supplementary material provides a broader comparison with geometry-based and generative methods (Jin et al., 2025; Jia et al., 2026; Kim et al., 2026; Szymanowicz et al., 2026; Yu et al., 2021; Charatan et al., 2024; Xu et al., 2025; Chen et al., 2024; Zhang et al., 2024; Yu et al., 2024; Zhou et al., 2025; Suhail et al., 2022; Du et al., 2023).

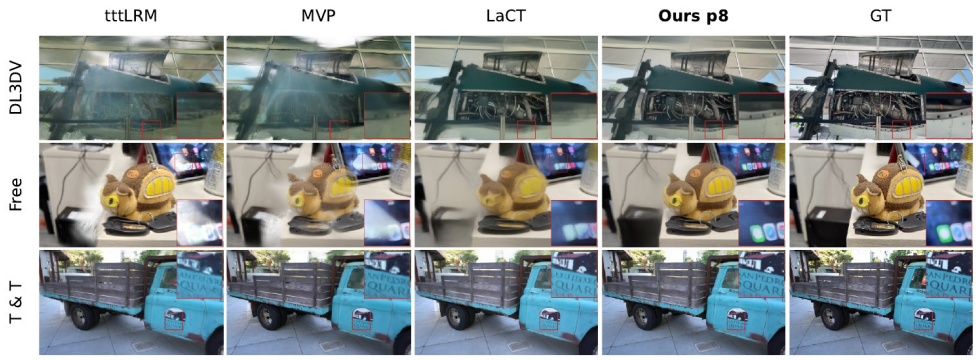

Figure 4: **Qualitative comparison.** We compare DVSM (ps8) with tttLRM, MVP, and LaCT on DL3DV and on the zero-shot Free and Tanks & Temples benchmarks (Wang et al., 2026; Kang et al., 2026b; Zhang et al., 2025; Ling et al., 2024; Wang et al., 2023; Knapitsch et al., 2017). DVSM preserves sharper text and scene details. The supplementary material contains more images and videos.

Our setup follows LVSM where possible, but the cross-paper comparison is not fully controlled. For example, Efficient LVSM uses 1,024 latent channels, while SVSM trains for 170K iterations and uses projective camera encoding; both use only one feed-forward layer per block (Jia et al., 2026; Kim et al., 2026; Li et al., 2025). LagerNVS initializes its encoder from VGGT, which is pretrained with explicit 3D supervision (Szymanowicz et al., 2026; Wang et al., 2025b). We therefore use the matched experiments in Section 4.4 as the primary evidence for our architectural claims and report the broader comparison to place the final model in context.

Results. Table 2 reports image quality and computational cost at 256×256 resolution. The DINO-enhanced ps8 model achieves the second-best PSNR, SSIM, and LPIPS, only slightly behind LagerNVS while using less than half as many parameters. Its context processing and per-frame rendering are also substantially faster. It outperforms LVSM, decoder-only LVSM, Efficient LVSM, and SVSM across all three quality metrics. Without DINO features, the ps8 model is the smallest and fastest-rendering method in the main comparison, yet its quality is on par with the much larger SVSM. These results show that DVSM's performance does not come simply from increasing model size. The supplementary material provides an expanded comparison and qualitative results.

4.2 Dense-view evaluation

Benchmark. We evaluate multi-view synthesis on DL3DV, which contains about 10K diverse training scenes and 140 held-out test scenes (Ling et al., 2024). We consider two complementary settings. The short-sequence setting uses 2, 4, or 6 context views at 256×448 resolution and compares with MVSplat, DepthSplat, TokenGS, LVSM, LaCT, and LagerNVS (Chen et al., 2024; Xu et al., 2025; Ren et al., 2026; Jin et al., 2025; Zhang et al., 2025; Szymanowicz et al., 2026). The full-sequence setting uses the evaluation views provided by LongLRM: every eighth frame is a target, and K-means selects 16, 32, or 64 context views from the remaining frames at 540×960 resolution (Chen et al., 2025b). This setting compares with per-scene optimization using 3DGS, geometry-based feed-forward models, and geometry-free models (Kerbl et al., 2023; Chen et al., 2025b; Kang et al., 2026a; Wang et al., 2026; Kang et al., 2026b; Jin et al., 2025; Zhang et al., 2025; Szymanowicz et al., 2026). Because LVSM does not report the full-sequence setting, we evaluate it using our data pipeline (Jin et al., 2025).

Results. Table 3 reports quality, model size, and runtime for both settings. The ps8 model achieves the best PSNR, SSIM, and LPIPS at every evaluated view count. In the short-sequence setting, it clearly outperforms the strongest baseline with 2 views, and its advantage becomes substantial as more views are provided. In the full-sequence setting, it consistently surpasses all baselines, with a larger advantage at the highest view count. It also substantially outperforms per-scene-optimized 3DGS in image quality, although 3DGS retains much faster rasterization.

The ps16 model with DINO features provides a more efficient operating point. It also outperforms every baseline across all three quality metrics at 16, 32, and 64 views; at the highest view count, it is clearly better than LaCT while being slightly faster in both context processing and per-frame

Table 4: **Zero-shot evaluations.** The 16- and 64-view evaluations use 540×960 resolution. Size is the model size in millions of parameters. Ctx. is the context-processing time per scene, and Rend. is the per-frame rendering time. A darker tone is better.

		Mip-NeRF 360 (16 views)					Free (16 views)					Tanks & Temples (16 views)				
Method	Size	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓
Geometry-based methods																
LongLRM (Chen et al., 2025b)	174M	19.34	0.478	0.499	1.3s	5ms	20.32	0.633	0.377	0.3s	4ms	17.45	0.603	0.437	0.3s	4ms
iLRM (Kang et al., 2026a)	177M	20.35	0.497	0.495	0.3s	6ms	21.17	0.642	0.393	0.3s	4ms	18.23	0.629	0.422	0.3s	4ms
tttLRM (Wang et al., 2026)	381M	21.25	0.546	0.417	3.0s	6ms	22.38	0.698	0.309	1.9s	5ms	18.90	0.668	0.356	1.9s	7ms
MVP (Kang et al., 2026b)	230M	20.71	0.559	0.417	0.1s	6ms	22.12	0.706	0.285	0.1s	4ms	18.79	0.671	0.352	0.1s	4ms
Geometry-free methods																
LaCT (Zhang et al., 2025)	379M	20.62	0.523	0.418	1.2s	53ms	21.39	0.653	0.331	1.1s	53ms	18.78	0.644	0.362	1.1s	53ms
DVSM (ours) ps16+dino	454M	20.36	0.518	0.391	0.4s	17ms	20.64	0.674	0.321	0.3s	17ms	18.94	0.664	0.342	0.3s	17ms
DVSM (ours) ps8	163M	22.29	0.609	0.314	2.4s	143ms	22.98	0.734	0.240	2.4s	143ms	20.34	0.711	0.286	2.4s	143ms

		Mip-NeRF 360 (64 views)					Free (64 views)					Tanks & Temples (64 views)				
Method	Size	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓	PSNR↑	SSIM↑	LPIPS↓	Ctx.↓	Rend.↓
Geometry-based methods																
LongLRM (Chen et al., 2025b)	174M	21.64	0.536	0.423	2.3s	9ms	23.67	0.720	0.289	2.2s	10ms	20.02	0.681	0.350	2.2s	9ms
iLRM (Kang et al., 2026a)	177M	22.01	0.556	0.430	2.6s	10ms	23.84	0.712	0.309	2.6s	9ms	20.25	0.696	0.350	2.6s	8ms
tttLRM (Wang et al., 2026)	381M	23.65	0.633	0.336	8.2s	11ms	25.76	0.789	0.222	8.1s	13ms	21.55	0.751	0.274	8.1s	13ms
MVP (Kang et al., 2026b)	230M	24.27	0.692	0.283	0.4s	9ms	26.50	0.820	0.178	0.4s	11ms	22.02	0.779	0.233	0.4s	9ms
Geometry-free methods																
LaCT (Zhang et al., 2025)	379M	23.93	0.612	0.332	4.4s	53ms	26.02	0.746	0.242	4.4s	53ms	22.61	0.739	0.270	4.4s	53ms
DVSM (ours) ps16+dino	454M	23.64	0.651	0.268	2.8s	47ms	24.31	0.752	0.228	2.8s	47ms	23.05	0.784	0.206	2.8s	47ms
DVSM (ours) ps8	163M	26.06	0.746	0.198	35.6s	505ms	27.55	0.835	0.143	35.8s	506ms	24.35	0.821	0.171	35.2s	503ms

rendering. The ps8 model is the smallest reported model in the full-sequence comparison but delivers the highest quality. Thus, the gains are not explained by model scale or an explicit geometric representation.

Our dense-view LVSM evaluation remains nearly flat as the number of context views increases. Because this behavior may reflect limitations of either the fixed latent-token design or our dense-view training setup, we do not use this baseline alone to support the architectural claim; the matched ablations in Section 4.4 provide the controlled evidence.

Limitations. LVSM-based methods render more slowly than geometry-based methods such as LongLRM (Chen et al., 2025b). DVSM also cannot process all 200–400 input views within GPU memory. Future work could use more efficient attention, merge tokens, or otherwise reduce rendering cost (Shazeer, 2019; Ainslie et al., 2023; Bolya et al., 2023; Wang et al., 2025d).

Qualitative results. Figure 4 compares DVSM with tttLRM, MVP, and LaCT on DL3DV and two zero-shot benchmarks (Wang et al., 2026; Kang et al., 2026b; Zhang et al., 2025). DVSM generally preserves more high-frequency detail. The supplementary material includes additional images and videos.

4.3 Cross-dataset generalization

Benchmark. We evaluate models trained on DL3DV without finetuning on Mip-NeRF 360, Free, and Tanks & Temples (Ling et al., 2024; Barron et al., 2022; Wang et al., 2023; Knapitsch et al., 2017). For each dataset, we evaluate 16 and 64 context views at a fixed 540×960 resolution. We compare with geometry-based LongLRM, iLRM, tttLRM, and MVP, as well as the geometry-free LaCT (Chen et al., 2025b; Kang et al., 2026a; Wang et al., 2026; Kang et al., 2026b; Zhang et al., 2025). LaCT and both DVSM variants are evaluated under the same protocol. The ps8 variant does not use DINO features, whereas the ps16 variant uses DINO features. The supplementary material provides qualitative comparisons for every dataset and view count.

Results. Table 4 shows that the ps8 model achieves the best PSNR, SSIM, and LPIPS in all six dataset and view-count settings. With 16 views, it consistently outperforms the next-best entry, with the clearest advantage on Tanks & Temples. With 64 views, it substantially outperforms the strongest geometry-based baseline across all three datasets, again with the largest advantage on Tanks & Temples. DVSM therefore generalizes better than geometry-based methods that explicitly encode 3D inductive biases, despite synthesizing views directly from a learned feature cache. Moreover, the best-performing ps8 model is smaller than every baseline and substantially smaller than the ps16 model. The generalization gain is therefore not a consequence of a larger model or foundation-feature injection.

Table 5: **Ablation experiments.** All variants use the same data, training schedule, and evaluation protocol; only the component listed in the "Ablation" column changes. Size denotes trainable parameters. See Section A.4 for the experimental setup. Colors indicate results that are better, worse, or similar to (a).

# Ablation	Size(M)↓	PSNR↑	SSIM↑	LPIPS↓	Recon-time↓	Rend-fps↑
(a) =DVSM (ours)	164	24.60	0.773	0.223	0.39s	56.1
Weight sharing						
(b)=(a)+decouple inp. proj.	166	24.20	0.757	0.240	0.39s	56.1
(c)=(a)+decouple intra-view attn.	191	24.34	0.763	0.229	0.39s	56.2
(d)=(a)+decouple cross-view attn.	178	24.06	0.754	0.240	0.40s	55.7
(e)=(a)+decouple ffn	268	23.89	0.749	0.240	0.39s	56.3
(f) =(a)+decouple entire decoder	308	23.19	0.720	0.268	0.39s	56.1
Cross-view attn. in reconstruction stage						
(g)=(a)+recon. no cross-view attn.	164	22.98	0.719	0.267	0.16s	56.9
(h)=(g)+decouple entire decoder	308	21.89	0.673	0.305	0.16s	57.3
Foundation feature injection						
(i) =(a)+frozen DINOv3	164	25.12	0.789	0.206	0.65s	55.7
(j) =(a)+tunable DINOv3	454	25.64	0.803	0.186	0.65s	55.8
Block composition						
(k)=(a)+no ffn in-between	110	24.05	0.751	0.246	0.37s	61.2
(l) =(a)+no intra-view attn.	83	23.32	0.725	0.276	0.32s	75.9

Limitations. The ps8 model trades efficiency for quality: its context-processing and per-frame rendering costs grow substantially at the highest view count. All methods also perform worse on these unseen datasets than on the in-domain DL3DV evaluation in Table 3. Training on a mixture of datasets, as done by geometry-based foundation models, may further improve robustness (Wang et al., 2025b;c; Lin et al., 2025).

4.4 Controlled experiments

We first present controlled experiments that isolate the design choices in DVSM.

Weight sharing. The fully shared DVSM is configuration (a) in Table 5; configurations (b)–(f) progressively decouple components from the input projection to the entire decoder. Every form of decoupling reduces quality despite adding parameters. Fully separating the decoder in (f) causes the largest drop, -1.41 dB PSNR, while using the most parameters.

Cross-view attention during reconstruction. Configurations (g)–(h) replace reconstruction-stage cross-view attention with the intra-view attention used by Efficient LVSM (Jia et al., 2026). This reduces reconstruction time by about 60% but causes a large quality drop, while rendering speed is nearly unchanged. Weight sharing remains beneficial, with a 1.09 dB PSNR gap between (g) and (h).

Additional ablations. Frozen and tunable DINOv3 features improve PSNR by 0.52 and 1.04 dB, respectively, without materially changing rendering speed (Sim'eoni et al., 2025). Smaller blocks are faster and use fewer parameters, but substantially reduce quality. Further discussion appears in Section A.4.

5 Conclusion

Controlled experiments show that a simple decoder-only design works best for regression-based view synthesis: global context encoding, a reusable KV cache, unidirectional decoding, and complete weight sharing. Decoupling weights consistently reduces quality, while shared weights align reconstruction and rendering features and enable oracle-feature recovery. With optional foundation features, DVSM achieves state-of-the-art results on DL3DV and zero-shot benchmarks without explicit 3D inductive biases. Because its best-performing variant is also the smallest model on most benchmarks, the gains come from the design rather than model scale.

AI USE STATEMENT

In this work, we used generative AI tools to implement and debug new features specified by the authors and to assist with interpreting and presenting experimental results. The original codebase, the proposed method, and all experiments were designed by the authors. We did not use generative AI to propose hypotheses, design the research methodology or experiments. Additionally, we used generative AI to draft and edit the manuscript, organize the LaTeX source, locate and summarize relevant literature, prepare tables and scientific figures, and write analysis and plotting scripts. The first author traced and manually verified all AI-assisted code changes and checked every AI-suggested citation against its original source. All co-authors proofread the complete manuscript. The authors take responsibility for the final content of this work, including text, claims, code, and artifacts produced with the aid of generative AI.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebr'on, and Sumit K. Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *EMNLP*, 2023.

Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, and Di Zhang. Recammaster: Camera-controlled generative rendering from a single video. In *ICCV*, 2025.

Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *ICLR*, 2023.

Chenjie Cao, Jingkai Zhou, Shikai Li, Jingyun Liang, Chaohui Yu, Fan Wang, Xiangyang Xue, and Yanwei Fu. Uni3c: Unifying precisely 3d-enhanced camera and human motion controls for video generation. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, 2025.

Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Eric Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, Jonathan Tremblay, S. Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3d generative adversarial networks. In *CVPR*, 2021.

David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. Pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *CVPR*, 2024.

Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *ICCV*, 2021.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, 2022.

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 *ECCV*, 2024.

Ziwen Chen, Hao Tan, Peng Wang, Zexiang Xu, and Fuxin Li. Long-lrm++: Preserving fine details in feed-forward wide-coverage reconstruction. *arXiv preprint arXiv:2512.10267*, 2025a.

Ziwen Chen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Fuxin Li, and Zexiang Xu. Long-lrm: Long-sequence large reconstruction model for wide-coverage gaussian splats. In *ICCV*, 2025b.

Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. In *ICML*, 2024.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024.

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

Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *CVPR*, 2023.

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

Shrisudhan Govindarajan, Daniel Rebain, Kwang Moo Yi, and Andrea Tagliasacchi. Radiant foam: Real-time differentiable ray tracing. In *ICCV*, 2025.

Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *CVPR*, 2019.

Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *CVPR*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Greg Heinrich, Michael Ranzinger, Hongxu Yin, Yao Lu, Jan Kautz, Andrew Tao, Bryan Catanzaro, and Pavlo Molchanov. Radiov2.5: Improved baselines for agglomerative vision foundation models. In *CVPR*, 2024.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *EMNLP*, 2020.

Sunghwan Hong, Jaewoo Jung, Heeseong Shin, Jiaolong Yang, Seungryong Kim, and Chong Luo. Unifying correspondence, pose and nerf for generalized pose-free novel view synthesis. In *CVPR*, 2024a.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: large reconstruction model for single image to 3d. In *ICLR*, 2024b.

Xiaosong Jia, Yihang Sun, Junqi You, Songbur Wong, Zichen Zou, Junchi Yan, Zuxuan Wu, and Yu-Gang Jiang. Efficient-lvsm: Faster, cheaper and and better large view synthesis model via decoupled co-refinement attention. *arXiv preprint arXiv:2602.06478*, 2026.

Hanwen Jiang, Hao Tan, Peng Wang, Haian Jin, Yue Zhao, Sai Bi, Kai Zhang, Fujun Luan, Kalyan Sunkavalli, Qixing Huang, and Georgios Pavlakos. Rayzer: A self-supervised large view synthesis model. In *ICCV*, 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 *ICLR*, 2025.

Justin Johnson, Alexandre Alahi, and Fei-Fei Li. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

Gyeongjin Kang, Seungtae Nam, Seungkwon Yang, Xiangyu Sun, Sameh Khamis, Abdelrahman Mohamed, and Eunbyung Park. iLRM: An iterative large 3d reconstruction model. In *CVPR*, pp. 37332–37342, 2026a.

Gyeongjin Kang, Seungkwon Yang, Seungtae Nam, Younggeun Lee, Jungwoo Kim, and Eunbyung Park. Multi-view pyramid transformer: Look coarser to see broader. In *CVPR*, pp. 37380–37390, 2026b.

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

Taras Khakhulin, Denis Korzhenkov, Pavel Solovev, Gleb Sterkin, Andrei-Timotei Ardelean, and Victor Lempitsky. Stereo magnification with multi-layer images. In *CVPR*, 2022.

Evan Kim, Hyunwoo Ryu, Thomas W. Mitchel, and Vincent Sitzmann. Scaling view synthesis transformers. *arXiv preprint arXiv:2602.21341*, 2026.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM TOG*, 2017.

Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J. Davison. Eschernet: A generative model for scalable view synthesis. In *CVPR*, 2024.

Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as relative positional encoding. In *Adv. Neural Inform. Process. Syst.*, 2025.

Hanxue Liang, Jiawei Ren, Ashkan Mirzaei, Antonio Torralba, Ziwei Liu, Igor Gilitschenski, Sanja Fidler, Cengiz Oztireli, Huan Ling, Zan Gojcic, and Jiahui Huang. Feed-forward bullet-time reconstruction of dynamic scenes from monocular videos. *Adv. Neural Inform. Process. Syst.*, 2025.

Haotong Lin, Sili Chen, Jun Hao 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.

Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, Xuanmao Li, Xingpeng Sun, Rohan Ashok, Aniruddha Mukherjee, Hao Kang, Xiangrui Kong, Gang Hua, Tianyi Zhang, Bedrich Benes, and Aniket Bera. DL3DV-10K: A large-scale scene dataset for deep learning-based 3d vision. In *CVPR*, 2024.

Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *CVPR*, 2022.

Yuan Liu, Chu-Hsing Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. In *ICLR*, 2024.

Xiaoxiao Long, Yuanchen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, and Wenping Wang. Wonder3d: Single image to 3d using cross-domain diffusion. In *CVPR*, 2024.

Nelson L. Max. Optical models for direct volume rendering. *IEEE TVCG*, 1995.

Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: practical view synthesis with prescriptive sampling guidelines. *SIGGRAPH*, 2019.

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.

Thomas W. Mitchel, Hyunwoo Ryu, and Vincent Sitzmann. True self-supervised novel view synthesis is transferable. *arXiv preprint arXiv:2510.13063*, 2025.

Nicolas Moenne-Loccoz, Ashkan Mirzaei, Or Perel, Riccardo de Lutio, Janick Martinez Esturo, Gavriel State, Sanja Fidler, Nicholas Sharp, and Zan Gojcic. 3d gaussian ray tracing: Fast tracing of particle scenes. *ACM TOG*, 2024.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM TOG*, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russ Howes, Po-Yao (Bernie) Huang, Shang-Wen Li, Ishan Misra, Michael G. Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *TMLR*, 2024.

Julius Plucker. On a new geometry of space. *Philosophical Transactions of the Royal Society of London*, 1865.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In *Machine Learning and Systems*, 2023.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, et al. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *Adv. Neural Inform. Process. Syst.*, 2025.

Michael Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo Molchanov. Am-radio: Agglomerative vision foundation model reduce all domains into one. In *CVPR*, 2023.

Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *ICCV*, 2021.

Jiawei Ren, Michal Jan Tyszkiewicz, Jiahui Huang, and Zan Gojcic. TokenGS: Decoupling 3d gaussian prediction from pixels with learnable tokens. In *CVPR*, pp. 15365–15375, 2026.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Muller, Alexander Keller, Sanja Fidler, and Jun Gao. Gen3c: 3d-informed world-consistent video generation with precise camera control. In *CVPR*, 2025.

Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-free view synthesis: Transformers and no 3d priors. In *ICCV*, 2021.

Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas A. Funkhouser, and Andrea Tagliasacchi. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *CVPR*, 2022.

Noam Shazeer. Fast transformer decoding: One write-head is all you need. *arXiv preprint arXiv:1911.02150*, 2019.

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *CVPR*, 2016.

Oriane Sim'eoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michael Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Herv'e J'egou, Patrick Labatut, and Piotr Bojanowski. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Vincent Sitzmann, Semon Rezchikov, William T. Freeman, Joshua B. Tenenbaum, and Frédo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In *Adv. Neural Inform. Process. Syst.*, 2021.

Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs. *arXiv preprint arXiv:2408.13912*, 2024.

Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. *IEEE TPAMI*, 2021.

Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Generalizable patch-based neural rendering. In *ECCV*, 2022.

Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022.

Cheng Sun, Jaesung Choe, Charles Loop, Wei-Chiu Ma, and Yu-Chiang Frank Wang. Sparse voxels rasterization: Real-time high-fidelity radiance field rendering. In *CVPR*, 2025a.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Ge Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Oluwasanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): Rnns with expressive hidden states. In *ICML*, 2025b.

Stanislaw Szymanowicz, Eldar Insafutdinov, Chuanxia Zheng, Dylan Campbell, João F. Henriques, Christian Rupprecht, and Andrea Vedaldi. Flash3d: Feed-forward generalisable 3d scene reconstruction from a single image. *3DV*, 2024.

Stanislaw Szymanowicz, Minghao Chen, Jianyuan Wang, Christian Rupprecht, and Andrea Vedaldi. Lagernvs: Latent geometry for fully neural real-time novel view synthesis. In *CVPR*, 2026.

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 *ECCV*, 2024a.

Shitao Tang, Fuyang Zhang, Jiacheng Chen, Peng Wang, and Yasutaka Furukawa. Mvdiffusion: Enabling holistic multi-view image generation with correspondence-aware diffusion. In *NeurIPS*, 2023.

Shitao Tang, Jiacheng Chen, Dilin Wang, Chengzhou Tang, Fuyang Zhang, Yuchen Fan, Vikas Chandra, Yasutaka Furukawa, and Rakesh Ranjan. Mvdiffusion++: A dense high-resolution multi-view diffusion model for single or sparse-view 3d object reconstruction. In *ECCV*, 2024b.

Rafal Tobiasz, Grzegorz Wilczy'nski, Marcin Mazur, Slawomir Konrad Tadeja, and Przemysław Spurek. Meshsplats: Mesh-based rendering with gaussian splatting initialization. In *AAAI*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Nicolas von Lützow and Matthias Nießner. Linprim: Linear primitives for differentiable volumetric rendering. In *Adv. Neural Inform. Process. Syst.*, 2025.

Chen Wang, Hao Tan, Wang Yifan, Zhiqin Chen, Yuheng Liu, Kalyan Sunkavalli, Sai Bi, Lingjie Liu, and Yiwei Hu. tttlrm: Test-time training for long context and autoregressive 3d reconstruction. *arXiv preprint arXiv:2602.20160*, 2026.

Haoru Wang, Kai Ye, Yangyan Li, Wenzheng Chen, and Baoquan Chen. The less you depend, the more you learn: Synthesizing novel views from sparse, unposed images without any 3d knowledge. *arXiv preprint arXiv:2506.09885*, 2025a.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotný. VGGT: visual geometry grounded transformer. In *CVPR*, 2025b.

Peng Wang, Yuan Liu, Zhaoxi Chen, Lingjie Liu, Ziwei Liu, Taku Komura, Christian Theobalt, and Wenping Wang. F2-nerf: Fast neural radiance field training with free camera trajectories. In *CVPR*, 2023.

Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P. Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas A. Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021.

Xingzheng Wang, Kaiqiang Chen, Zixuan Wang, and Wenhao Huang. Pmsnet: Parallel multi-scale network for accurate low-light light-field image enhancement. In *Transactions on Multimedia*, 2024.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Scalable permutation-equivariant visual geometry learning. *arXiv preprint arXiv:2507.13347*, 2025c.

Zhengqing Wang, Yuefan Wu, Jiacheng Chen, Fuyang Zhang, and Yasutaka Furukawa. Clift: Compressive light-field tokens for compute-efficient and adaptive neural rendering. In *NeurIPS*, 2025d.

Xinyue Wei, Kai Zhang, Sai Bi, Hao Tan, Fujun Luan, Valentin Deschaintre, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Meshlrm: Large reconstruction model for high-quality mesh. *arXiv preprint arXiv:2404.12385*, 2024.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *ICML*, 2020.

Haofei Xu, Songyou Peng, Fangjinhua Wang, Hermann Blum, Daniel Barath, Andreas Geiger, and Marc Pollefeys. Depthsplat: Connecting gaussian splatting and depth. In *CVPR*, 2025.

Jiayu Yang, Wei Mao, José Manuel Álvarez, and Miaomiao Liu. Cost volume pyramid based depth inference for multi-view stereo. In *CVPR*, 2020.

Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, 2018.

Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. In *ICLR*, 2025.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021.

Wangbo Yu, Jinbo Xing, Li Yuan, Wenbo Hu, Xiaoyu Li, Zhipeng Huang, Xiangjun Gao, Tien-Tsin Wong, Ying Shan, and Yonghong Tian. Viewcrafter: Taming video diffusion models for high-fidelity novel view synthesis. *IEEE TPAMI*, 2024.

Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. GS-LRM: large reconstruction model for 3d gaussian splatting. In *ECCV*, 2024.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T Freeman, and Hao Tan. Test-time training done right. *arXiv preprint arXiv:2505.23884*, 2025.

Qitao Zhao, Hao Tan, Qianqian Wang, Sai Bi, Kai Zhang, Kalyan Sunkavalli, Shubham Tulsiani, and Hanwen Jiang. E-rayzer: Self-supervised 3d reconstruction as spatial visual pre-training. *arXiv preprint arXiv:2512.10950*, 2025.

Jensen Zhou, Hang Gao, Vikram S. Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models. In *ICCV*, 2025.

Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus H. Gross. EWA volume splatting. In *Visualization Conference*, 2001.

SUPPLEMENTARY MATERIAL

We provide implementation and dataset details in Section A and additional results in Section B.

A ADDITIONAL DETAILS

A.1 COMMON IMPLEMENTATION DETAILS

Unless stated otherwise, all final models use 768 latent channels and 12 alternating cross-attention blocks (Figure 1). We use AdamW with weight decay 0.05, momentum $(0.9, 0.95)$, and a cosine learning-rate schedule with 2.5K linear warm-up steps. The peak learning rate is 4×10^{-4} at the lowest training resolution and 1×10^{-5} during higher-resolution finetuning. The perceptual-loss weight is $\lambda = 0.2$. All models are trained on 64 A100 GPUs. We use BFloat16 mixed precision for the learned layers.

The standard $\mathrm{Plücker}$ function embeds camera information as a six-dimensional vector $(\mathbf{r}_o \times \mathbf{r}_d, \mathbf{r}_d)$ at each pixel. Here, $\mathbf{r}_o$ is the camera position, $\mathbf{r}_d$ is the unit ray direction, and $\times$ denotes the cross product. We normalize the context-view poses following GS-LRM (Zhang et al., 2024).

All Transformer blocks use pre-normalization and QK normalization (Xiong et al., 2020; Henry et al., 2020). Each attention layer has 12 heads, and each MLP expands the hidden dimension by a factor of four. We remove the bias from all linear projections.

A.2 RE10K TWO-VIEW DETAILS

The reported models use patch size 8; one variant injects DINOv3 features, while the other does not (Sim'eoni et al., 2025). For each GPU and iteration, we sample 32 scenes with two context views and three target views per scene. The frame gap between the context views is sampled uniformly from $[25, 192]$. Candidate targets include the frames between the two context views and extrapolated frames within 25 frames of either context view. Models are trained for 100K iterations at 256×256 with a peak learning rate of 4×10^{-4} and finetuned for 10K iterations at 512×512 with a peak learning rate of 1×10^{-4}.

A.3 DL3DV DENSE-VIEW DETAILS

During training, we sample 2, 4, 8, 16, or 32 context views per scene and use the same number of target views. We dynamically adjust the number of scenes so that each GPU processes 32 context views per iteration. The gap between adjacent context views is sampled from $[1, 16]$. If C is the set of sampled context-frame indices, target frames are sampled from $[\min(C) - 2, \max(C) + 2] \setminus C$. We remove the small subset of scenes that do not use a 16:9 aspect ratio or a 960p source resolution. Training uses 100K iterations at 340p, followed by 10K iterations each at 480p, 720p, and 540×960. The ps8 model does not use DINO features because of memory constraints.

A.4 CONTROLLED-EXPERIMENT DETAILS

All variants use the same training schedule and data sampler. We train on 8 A100 GPUs with 32 context views, 720p final resolution, 768 latent channels, 12 cross-view attention layers, and patch size 16. Each model is trained from scratch with a resolution curriculum: 100K iterations at 340p, 10K at 480p, and 10K at 720p. We use DL3DV, which contains a wide range of scenes (Ling et al., 2024). For each of its 140 held-out scenes, we select every eighth frame as a target and use K-means to sample 32 context views from the remaining frames, which contain about 300 images per scene on average. Efficiency is measured on one A100 GPU.

For foundation-feature injection, DINOv3-L16 is evaluated once on the context views and fused into the reconstruction branch as described in Section 3.4 (Sim'eoni et al., 2025). Frozen and tunable DINOv3 improve PSNR by 0.52 dB and 1.04 dB, respectively. Rendering cost remains nearly unchanged, while reconstruction takes 66% longer. For block composition, configuration (k) removes the feed-forward layer between intra-view and cross-view attention, and (l) removes all intra-view

attention. Both variants are faster and smaller but substantially less accurate, so we retain the original block.

The architecture study does not exhaust all general Transformer improvements. Register tokens, projective RoPE, and gated attention are promising extensions that we do not evaluate because of time and compute constraints (Darcet et al., 2024; Li et al., 2025; Qiu et al., 2025).

A.5 Zero-shot evaluation details

We evaluate the DL3DV-trained ps8 model from Section 4.3; this model does not use DINO features. The evaluation uses Mip-NeRF 360, Free, and Tanks & Temples (Barron et al., 2022; Wang et al., 2023; Knapitsch et al., 2017). Every eighth frame is used as a target, and K-means samples 16 or 64 context views from the remaining frames. We will release these splits for reproducibility. Context and target images are resized to 540×960, and metrics are computed at this resolution. We apply the same preprocessing to every method.

B Additional results

COLMAP-free extension. We test a simple extension that replaces COLMAP poses with a feed-forward camera estimator. For comparison with COLMAP-free methods on Re10K, we use the DVSM ps8 model with DINO features and estimate camera poses with the off-the-shelf π^3 model (Wang et al., 2025c). Following NoPoSplat, we use ground-truth camera intrinsics (Ye et al., 2025). The test scenes match Table 2, while the view sampling follows NoPoSplat.

Table 7 shows that this direct combination outperforms the dedicated baselines in the reported setup. This is not a jointly trained pose-estimation method; it demonstrates that DVSM can accept estimated cameras. Joint pose and view-synthesis training remains future work.

Qualitative results. Figure 5 retains the failure case from the original appendix. Figures 6 to 11 compare all evaluated input-view settings on DL3DV. Figures 12 to 17 provide zero-shot comparisons on Free, Mip-NeRF 360, and Tanks & Temples. Figure 18 presents the Re10K comparison at the end of the appendix.

Figure 5 highlights a failure under strong viewpoint extrapolation. The lounge chair is rendered correctly in the first row but disappears from another viewpoint in the second row. Similar inconsistencies occur in other LVSMs when the implicit geometry is incorrect. Better training-view sampling and explicit consistency constraints may improve extrapolation.

Across Re10K, DL3DV, Free, Mip-NeRF 360, and Tanks & Temples, DVSM generally produces sharper views and more legible details, including text. These examples are consistent with the quantitative comparisons in the main paper.

Table 6: **Expanded comparison with two input views on Re10K at 256×256 resolution.**

Method	PSNR↑	SSIM↑	LPIPS↓
pixelNeRF (Yu et al., 2021)	20.43	0.589	0.550
AttnRend (Du et al., 2023)	24.78	0.820	0.213
ViewCrafter (Yu et al., 2024)	21.63	0.642	0.175
GPNR (Suhail et al., 2022)	24.11	0.793	0.255
PixelSplat (Charatan et al., 2024)	26.09	0.863	0.136
MVSplat (Chen et al., 2024)	26.39	0.869	0.128
GS-LRM (Zhang et al., 2024)	28.10	0.892	0.114
DepthSplat (Xu et al., 2025)	27.46	0.889	0.115
SEVA (Zhou et al., 2025)	25.66	0.841	0.139
LVSM (Jin et al., 2025)	28.58	0.893	0.114
Dec.-only LVSM (Jin et al., 2025)	29.67	0.906	0.098
Eff. LVSM (Jia et al., 2026)	28.93	0.895	0.102
SVSM (Kim et al., 2026)	30.01	0.910	0.096
LagerNVS (Szymanowicz et al., 2026)	31.39	0.928	0.078
DVSM (ours) $_{\text{ps8}}$	29.98	0.912	0.096
DVSM (ours) $_{\text{ps8+dino}}$	31.23	0.925	0.085

Table 7: **COLMAP-free setup.** We replace COLMAP poses with camera poses predicted by π^3. Every method in the table estimates camera poses rather than using COLMAP.

Method	PSNR↑	SSIM↑	LPIPS↓
Splatt3R (Smart et al., 2024)	15.32	0.490	0.436
CoPoNeRF (Hong et al., 2024a)	18.94	0.619	0.388
NoPoSplat (Ye et al., 2025)	25.03	0.838	0.160
DVSM (ours) $_{+\pi^3}$ (Wang et al., 2025c) poses	27.55	0.860	0.124

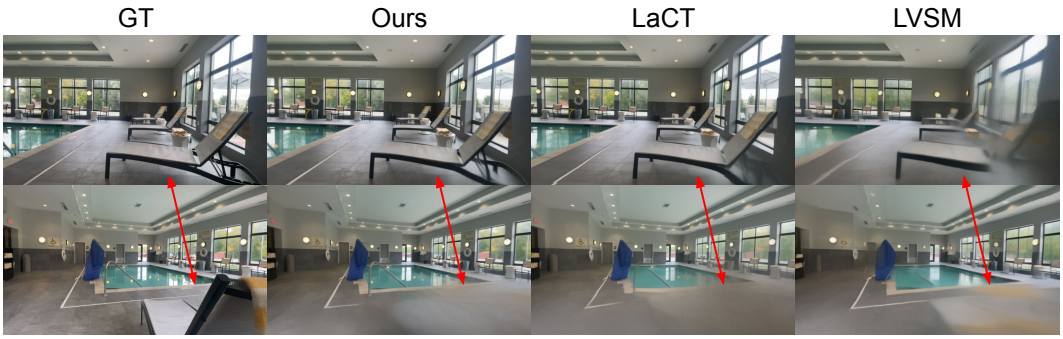

Figure 5: **Failure case.**

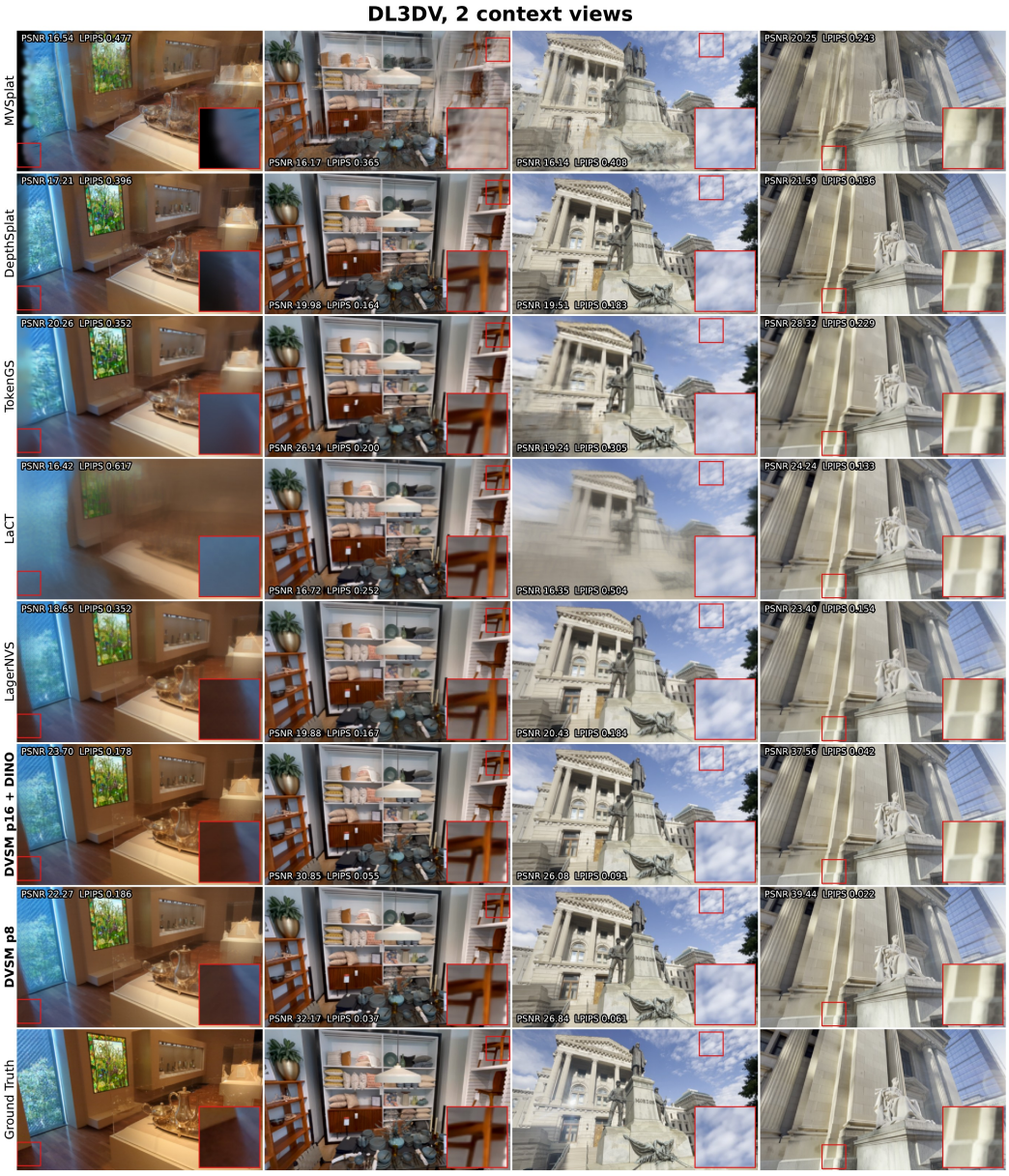

Figure 6: **Qualitative comparison on DL3DV using two context views (Ling et al., 2024).**

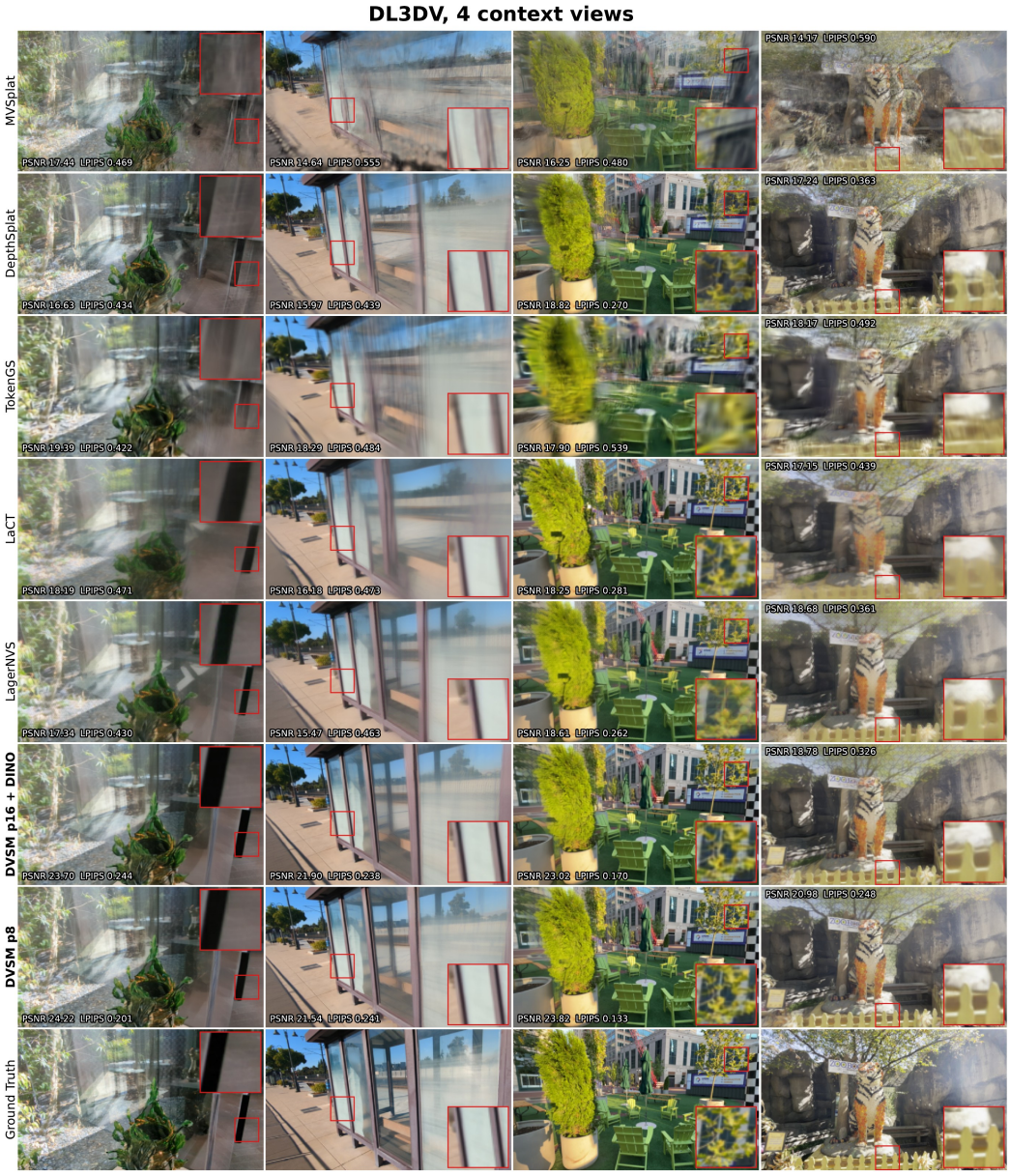

Figure 7: **Qualitative comparison on DL3DV using four context views (Ling et al., 2024).**

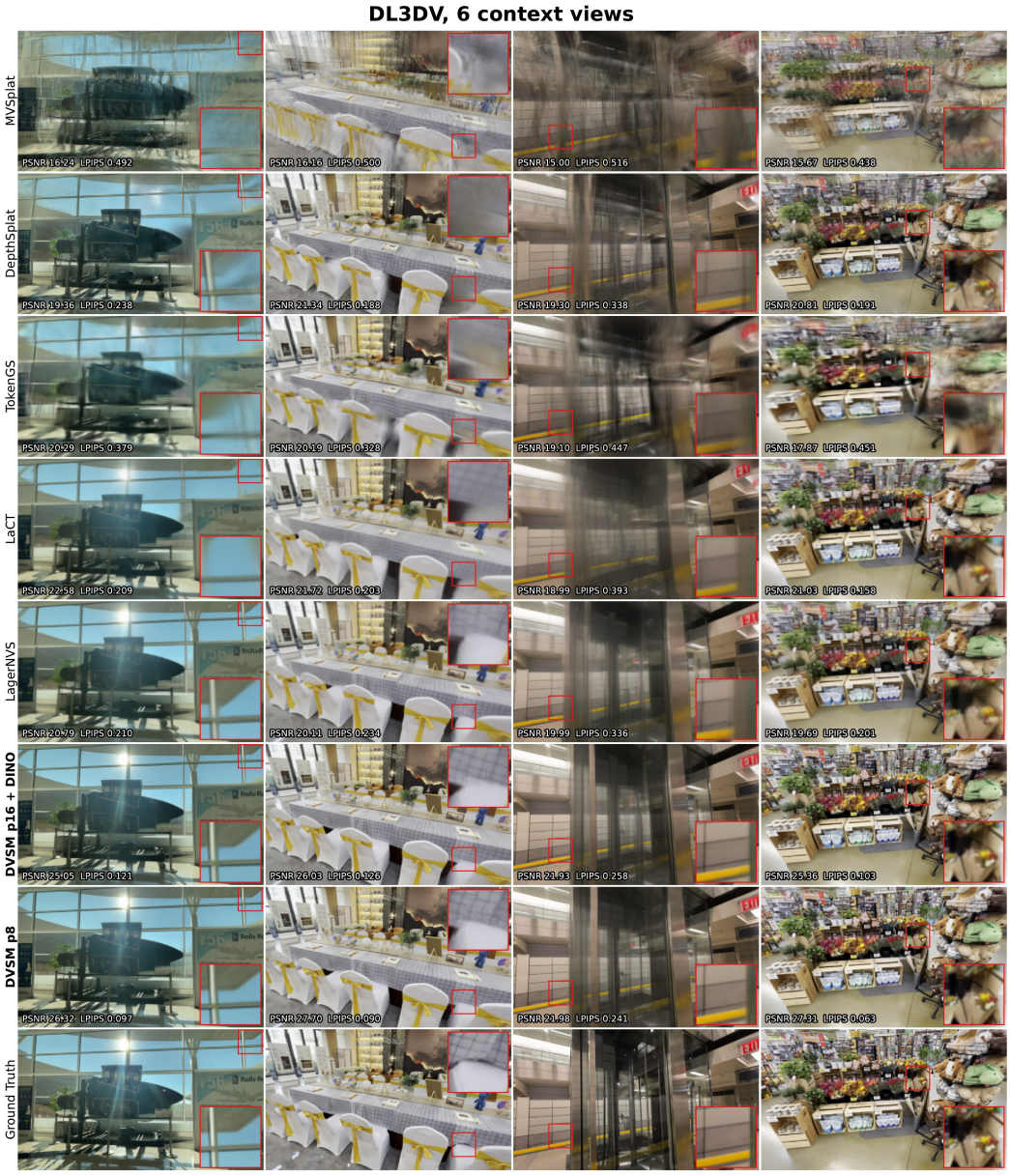

Figure 8: **Qualitative comparison on DL3DV using six context views (Ling et al., 2024).**



Figure 9: **Qualitative comparison on DL3DV using 16 context views (Ling et al., 2024).**

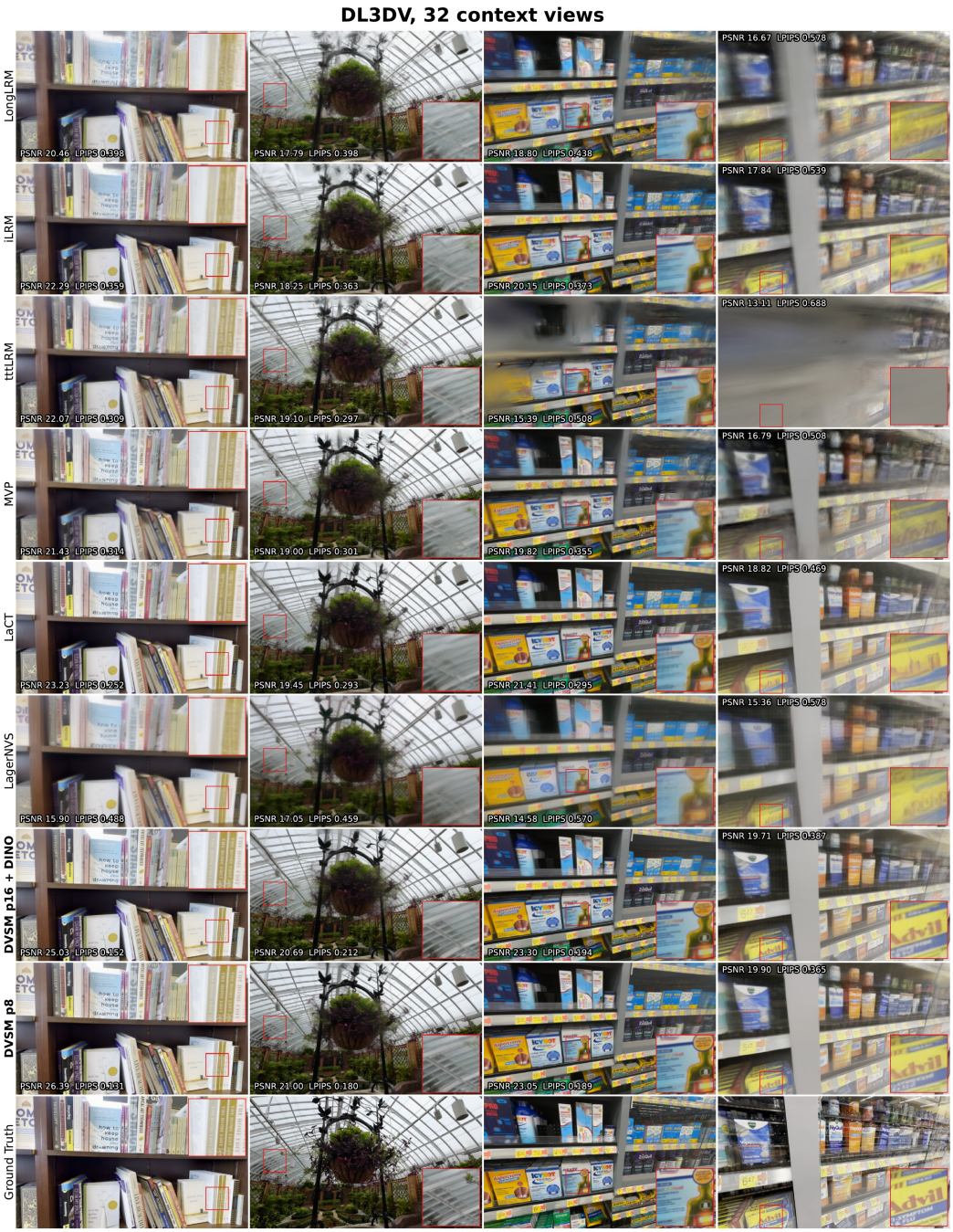

Figure 10: **Qualitative comparison on DL3DV using 32 context views (Ling et al., 2024).**

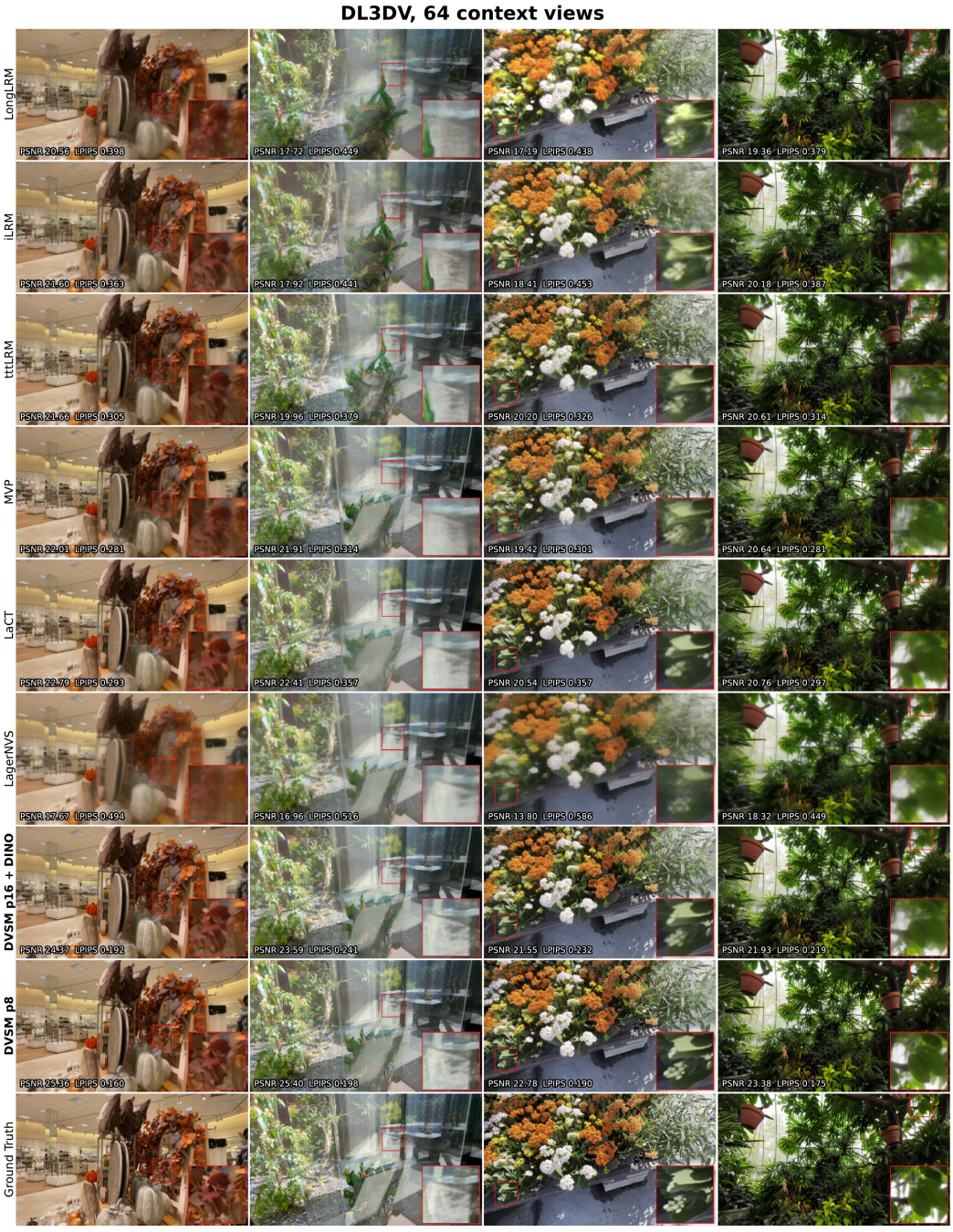

Figure 11: **Qualitative comparison on DL3DV using 64 context views (Ling et al., 2024).**

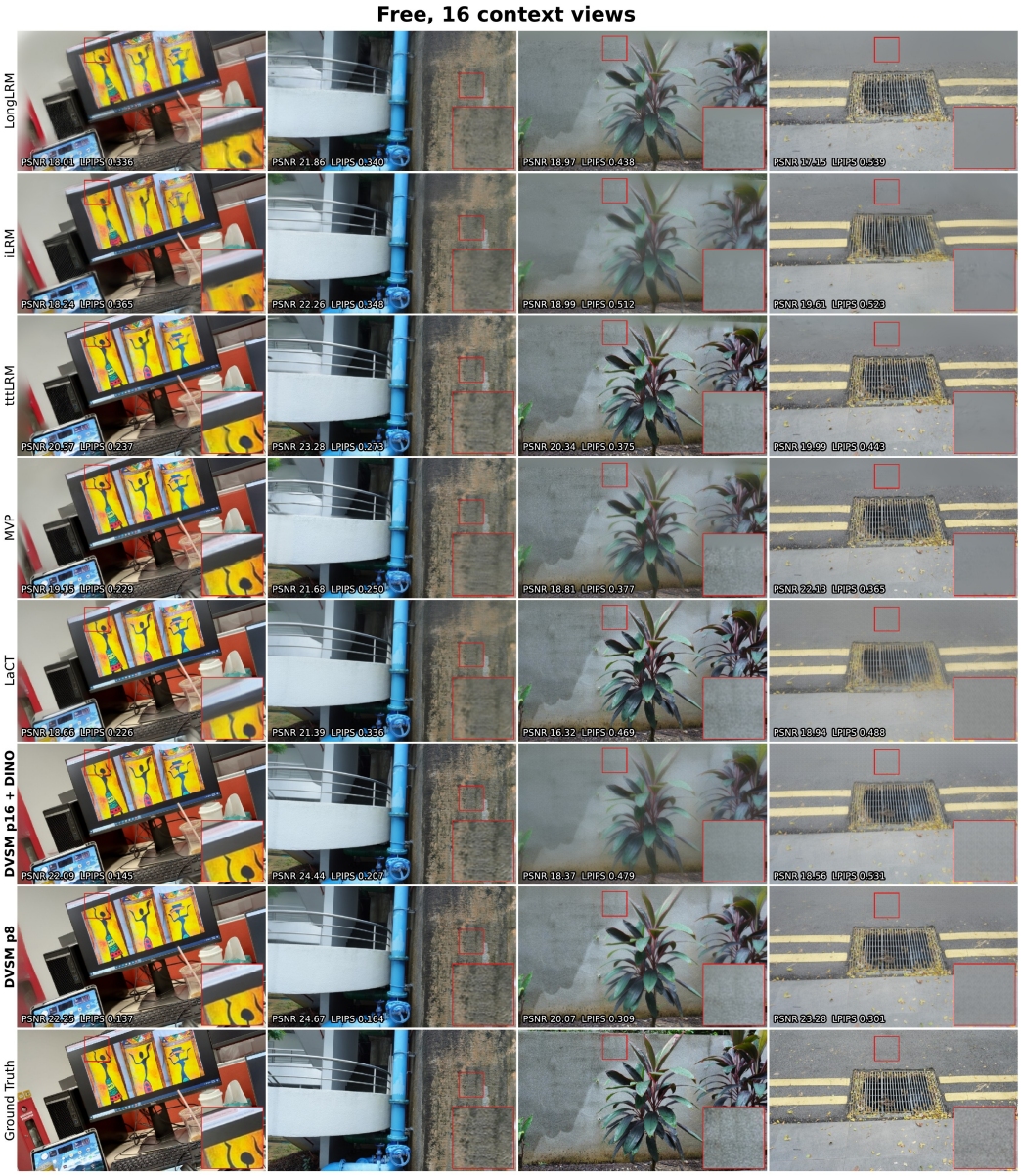

Figure 12: **Zero-shot qualitative comparison on Free using 16 context views (Wang et al., 2023).**

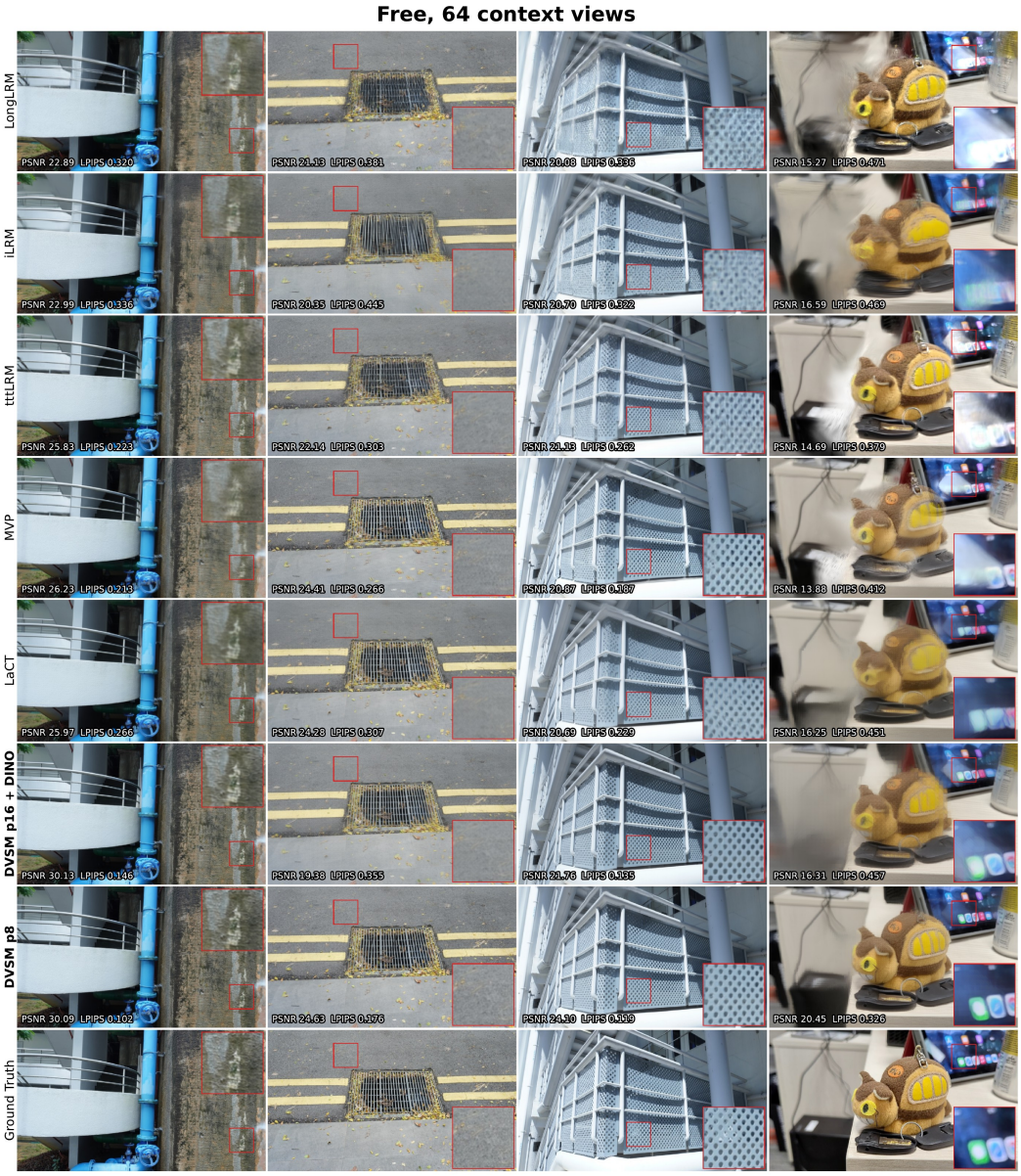

Figure 13: **Zero-shot qualitative comparison on Free using 64 context views (Wang et al., 2023).**

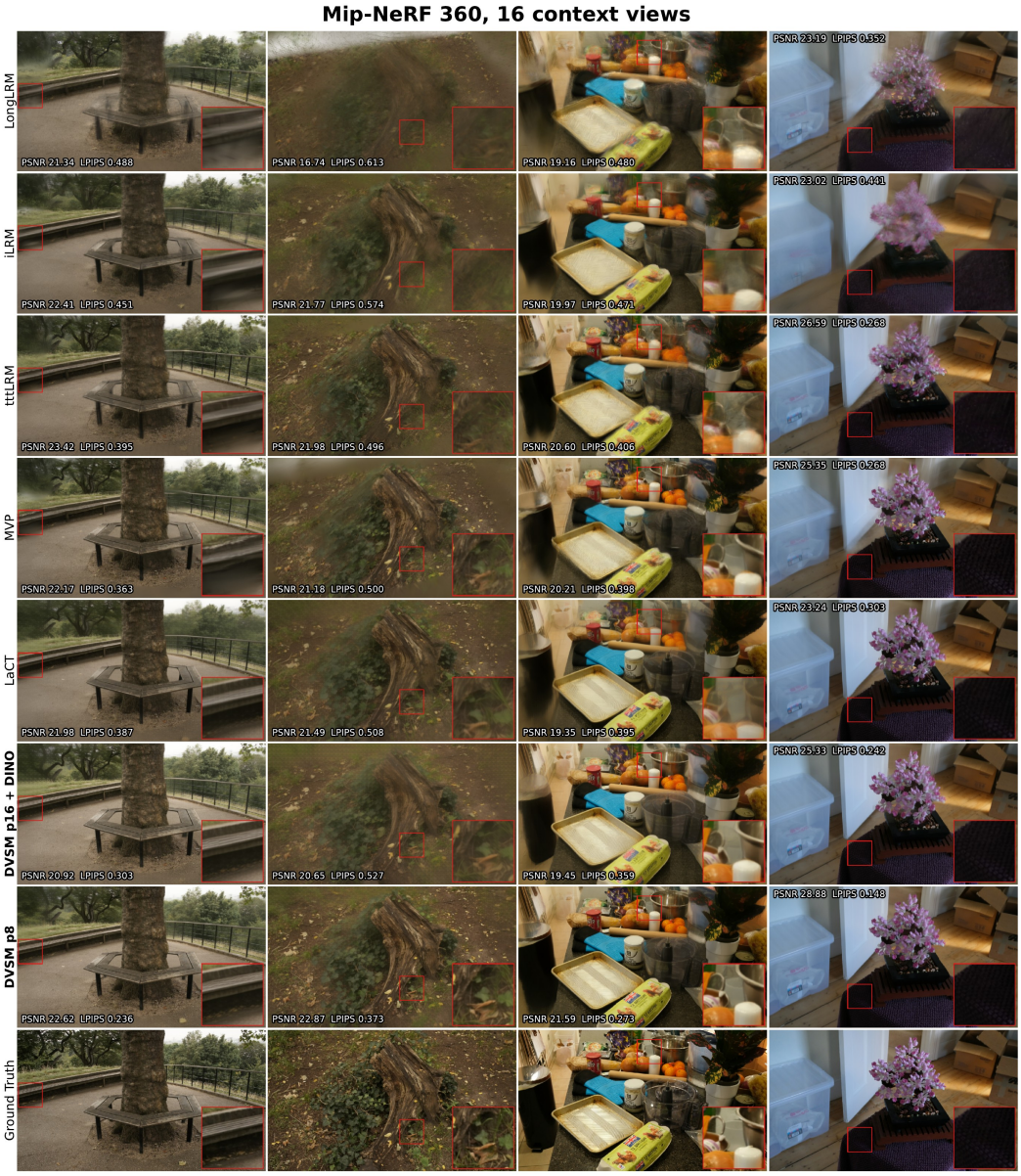

Figure 14: **Zero-shot qualitative comparison on Mip-NeRF 360 using 16 context views (Barron et al., 2022).**

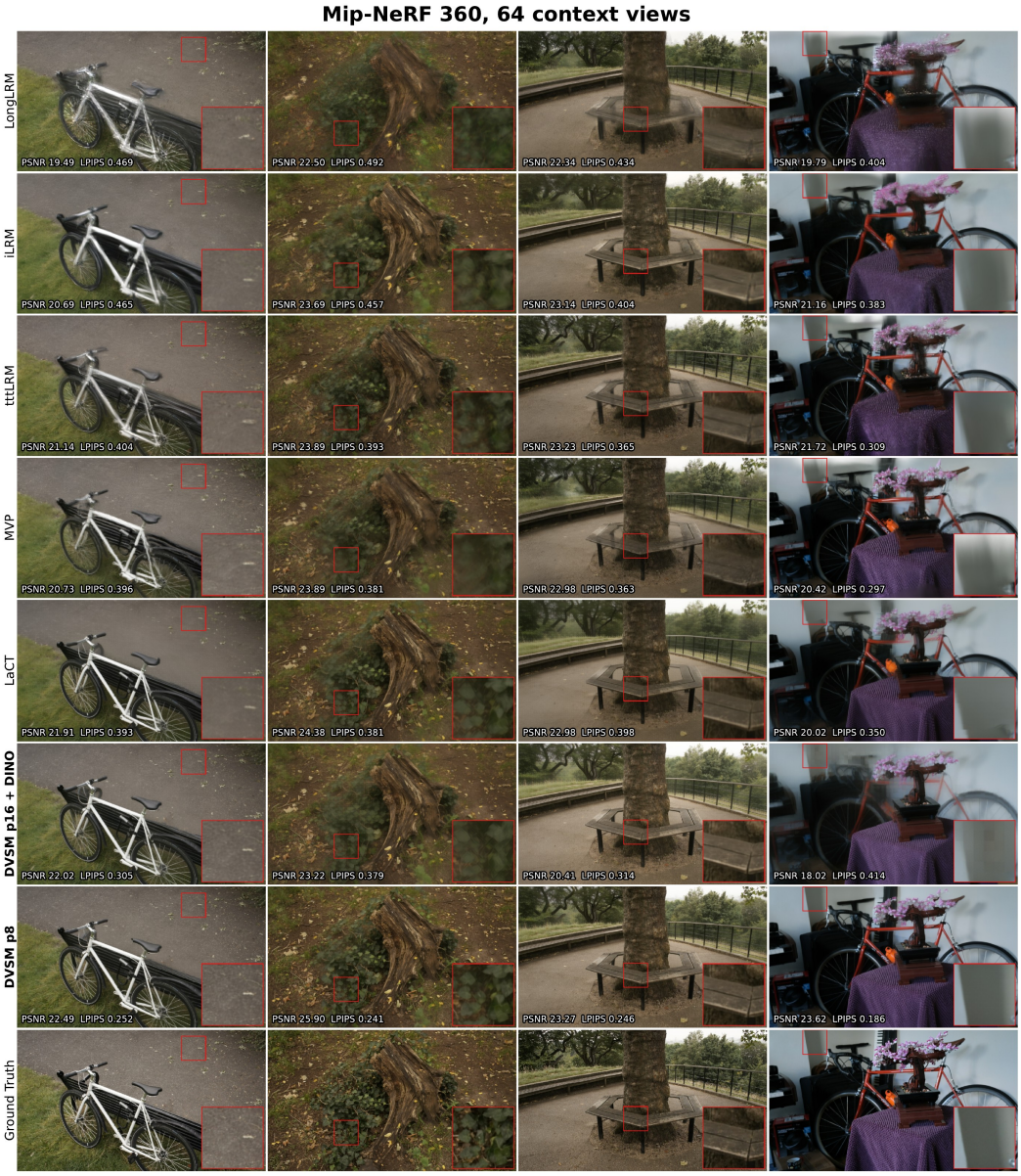

Figure 15: **Zero-shot qualitative comparison on Mip-NeRF 360 using 64 context views (Barron et al., 2022).**

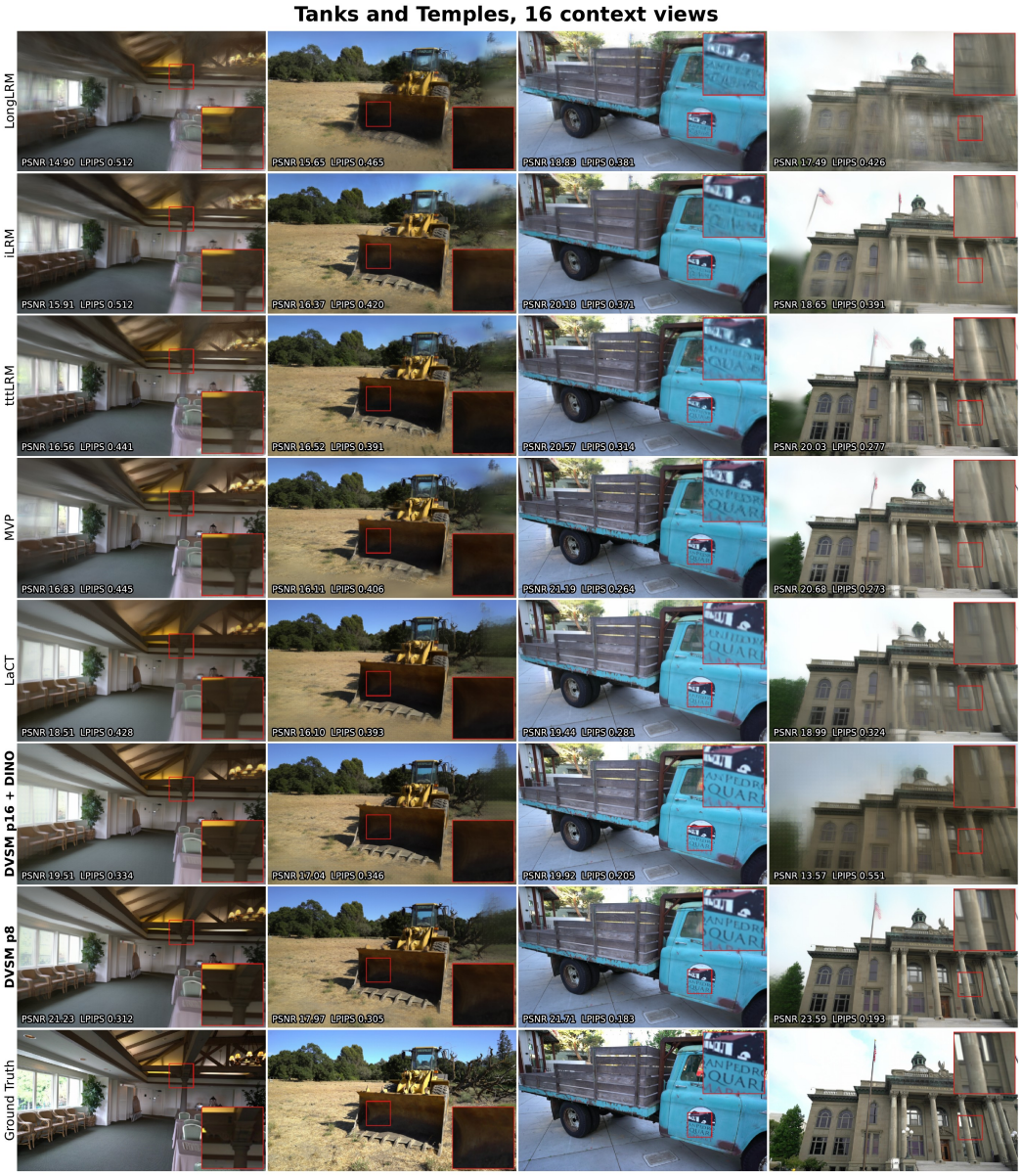

Figure 16: **Zero-shot qualitative comparison on Tanks & Temples using 16 context views (Knapitsch et al., 2017).**

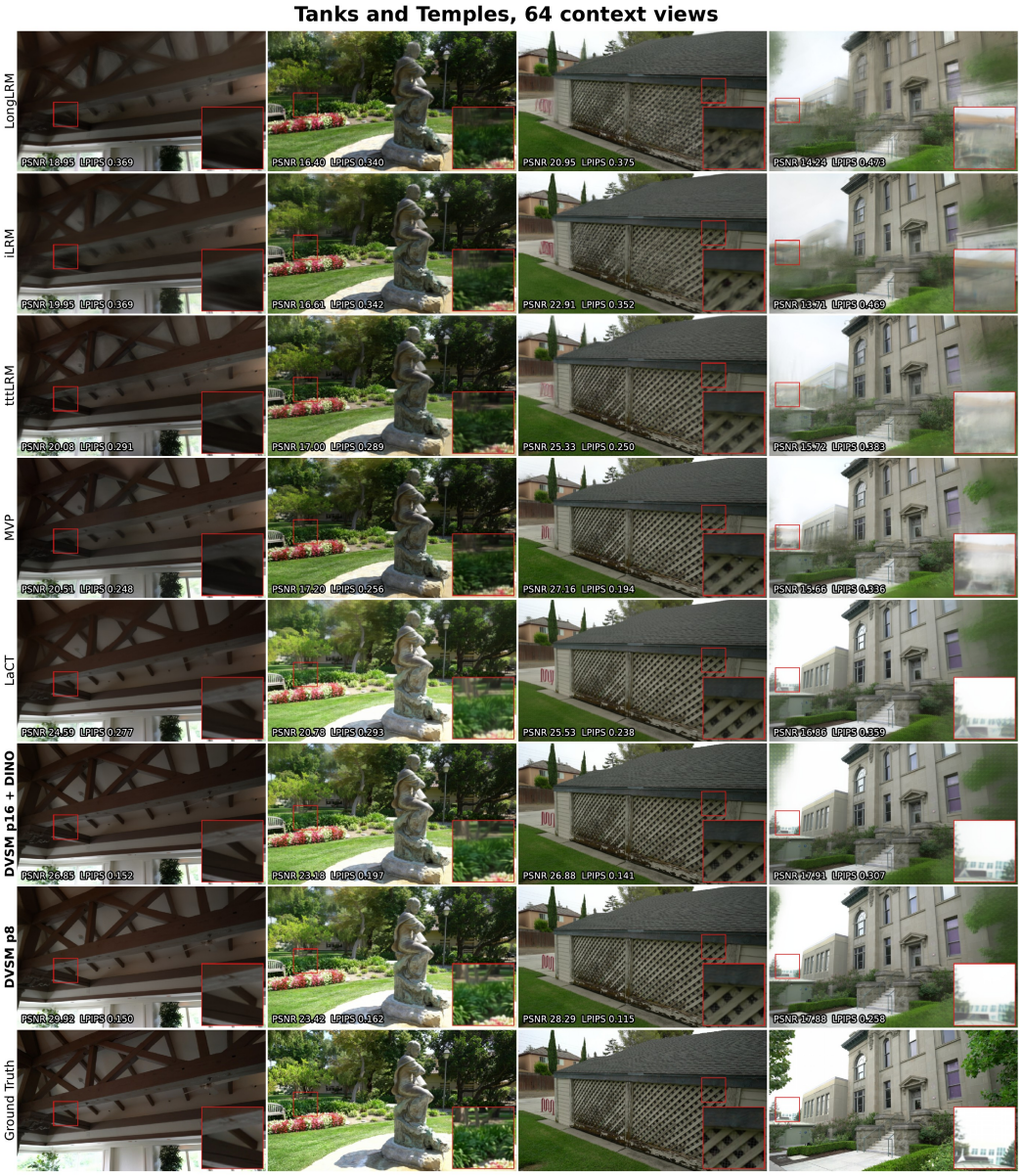

Figure 17: **Zero-shot qualitative comparison on Tanks & Temples using 64 context views (Knapitsch et al., 2017).**



Figure 18: **Qualitative results on Re10K with two context views (Khakhulin et al., 2022).**