GROUNDING SELF-SUPERVISED LATENT NOVEL VIEW SYNTHESIS WITH CROSS-VIEW CONSISTENCY

Anonymous authors
Paper under double-blind review

ABSTRACT

A self-supervised novel view synthesis objective offers a promising approach to learning latent scene representations from unlabeled images. Despite remarkable progress, inferring geometrically grounded representations from unposed multi-view images remains a fundamental challenge. Latent representation methods achieve strong photorealism but suffer from shortcut learning, and fail to recover the underlying camera motion. Explicit-representation methods resolve this ambiguity by rendering through 3D Gaussians, but their non-convex optimisation converges only under a carefully scheduled training curriculum. In this work, we ask whether geometry can be encoded into features while preserving the expressive capacity of a latent space. We introduce a feed-forward model that bakes geometry into features through a photometric reprojection objective. Our proposed framework comprises a permutation-equivariant camera estimator, a ray-conditioned renderer, and a dense depth decoder that facilitate reconstructing every view through a warping operation via predicted depth, intrinsics, and relative pose. We evaluate both photorealism and raw pose accuracy under interpolation and extrapolation settings. Experiments show that, in the extrapolation setting, our method outperforms prior self-supervised methods in PSNR, LPIPS, and strict raw pose accuracy.

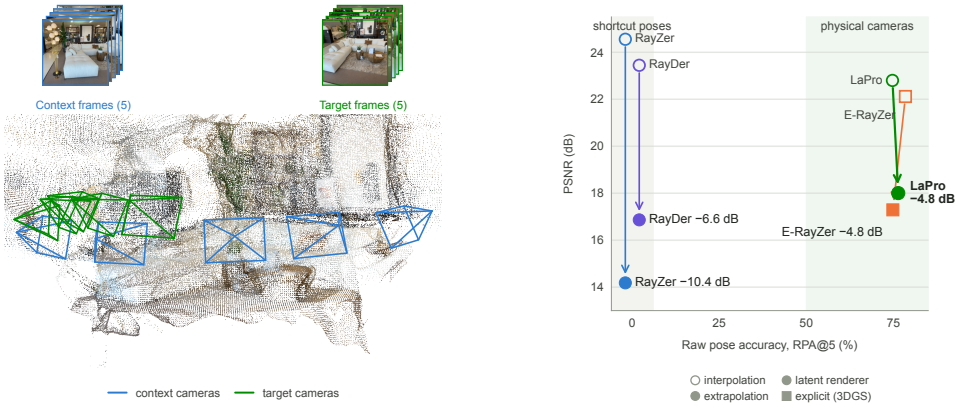

Figure 1: **Synthesizing novel views beyond the input trajectory. Left:** point cloud with blue context and green target cameras predicted by our model. **Right:** interpolation (hollow) to extrapolation (filled) performance, respectively; annotations show the PSNR change. Circles denote latent renderers; squares denote explicit 3DGS.

1 INTRODUCTION

Self-supervised novel view synthesis has emerged as a strong framework for learning scene representations from large amounts of unlabeled images: a model predicts the cameras of unposed views and learns by rendering held-out views, without 3D annotation. By learning to synthesise novel viewpoints, recent feed-forward methods such as RayZer (Jiang et al., 2025) and RayDer (Prestel et al.,

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.

2026) infer expressive, unconstrained latent scene representations that aim to encode the underlying 3D structure of a scene, and achieve striking photorealism. However, despite this photorealism, there remains a gap in learning latent representations that are geometrically grounded. Latent NVS models are prone to shortcut learning: they learn to interpolate the input frames rather than reason about viewpoint, and their rendering quality degrades once the query viewpoint moves beyond the input trajectory (Mitchel et al., 2026), precisely the setting that novel view synthesis is designed to address (Fig. 1).

Geometric grounding enables a novel view synthesis model to be controllable: a user-specified camera should produce the corresponding viewpoint, including viewpoints off the input trajectory, and two scenes traversed with identical camera motion should yield equivalent cameras regardless of their pixel content (Mitchel et al., 2026). Surprisingly, rendering quality alone does not reveal whether a model has this property. Prior latent-based approaches attain the highest rendering quality on the input trajectory, yet their predicted cameras fail to recover the underlying camera motion. This stems from the rendering objective being under-constrained: it permits learning $SE(3)$ poses and latent scene representations that only need to satisfy the rendering objective, rather than recover the scene structure and camera motion. Consequently, latent pose representations can become temporal interpolation cues that a learned decoder exploits to shortcut rather than rely on geometric transformations.

Recent methods address this shortcoming differently. E-RayZer (Zhao et al., 2025) and Pensieve (Wang et al., 2025b) reintroduce explicit 3D Gaussians, constraining rendering directly in 3D space. This ties the model to explicit primitives whose optimisation depends on a training curriculum: E-RayZer schedules samples by visual overlap precomputed with a pretrained matcher, and its pose accuracy collapses without it. XFactor (Mitchel et al., 2026) instead trains on pose-preserving augmentations of the same sequence, which separates pose from appearance, blurring renderings. LagerNVS (Szymanowicz et al., 2026) that 3D-grounded features benefit latent rendering, but obtains them from VGGT (Wang et al., 2025a), a foundation model trained with 3D annotations. In each case, geometry is restored by changing the representation, the data, or the backbone, rather than the training signal itself. We instead ask: *can we preserve the expressive capacity of a latent space while enabling the model to learn physically grounded, transferable camera poses, when multi-view geometry enters only through the training signal?*

We introduce LaPro, a self-supervised latent novel view synthesis framework with cross-view projective consistency. A permutation-equivariant camera estimator predicts relative poses and shared intrinsics, and a lightweight decoder predicts dense depth. We employ a cross-view warping operation that reconstructs every input view from every other, providing a pixel-dense signal to camera and depth networks. The predicted cameras must therefore explain how one real image projects into another, and since the warp is not learned, the renderer cannot compensate for a shortcut camera. We first train cameras and depth under this cross-view warping objective alone, with a narrow-to-wide curriculum that gradually expands the temporal baseline of the sampled windows, and then add a transformer-based latent renderer and train the system jointly. LaPro is the only self-supervised latent renderer we evaluate that recovers physical cameras, and that it achieves the best rendering quality and strict pose accuracy when the query leaves the input trajectory.

Our key contributions can be summarised as follows:

1. We propose a self-supervised framework that preserves the expressive latent space of feed-forward NVS, comprising a permutation-equivariant camera estimator, a dense depth decoder, and a Plücker ray-conditioned renderer.
2. We introduce a cross-view inverse-warping objective that provides a dense learning signal through the reprojective consistency of the input images to ground the predicted cameras in multi-view geometry.
3. We show that a simple narrow-to-wide curriculum suffices to train the end-to-end system without precomputed visual-overlap scores.

2 Related Work

We focus on feed-forward novel view synthesis, as opposed to per-scene optimisation methods such as NeRF (Mildenhall et al., 2020) and 3DGS (Kerbl et al., 2023). Within this setting, methods differ

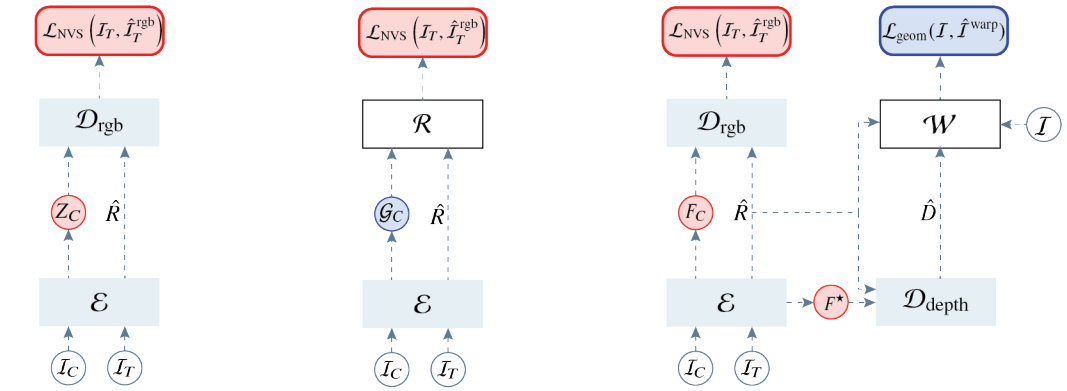

Figure 2: $\mathcal{I}_C$ and $\mathcal{I}_T$ are the unposed context and target images, and $\mathcal{I}$ is the union set; $\hat{\mathcal{R}}$ denotes the predicted cameras, $\hat{D}$ the predicted depth, $\hat{I}_T^{\text{rgb}}$ the rendered target, and $\hat{I}^{\text{warp}}$ reprojected images. Grey blocks are learned modules; white blocks are fixed, differentiable image-formation operators. Red denotes rendering objective, $\mathcal{L}_{\text{NVS}}$; blue denotes geometry and its loss $\mathcal{L}_{\text{geom}}$. **(a) Geometry left to the renderer** methods constrained only by $\mathcal{L}_{\text{NVS}}$. **(b) Geometry in the representation** methods that predict explicit Gaussians $\mathcal{G}_C$ constrained by a fixed rasteriser $\mathcal{R}$. **(c) Geometry through the objective (ours):** LaPro first pretrains with $\mathcal{L}_{\text{geom}}$, then adds $\mathcal{D}_{\text{rgb}}$ and trains jointly on both losses.

mainly in *where* 3D structure is imposed: in the scene representation, only implicitly through the rendering loss, in the training data or a pretrained backbone, or in the training objective (Fig. 2). To our knowledge, LaPro is the first to keep a fully latent renderer while imposing geometry solely through a self-supervised projective objective.

Geometry in the representation. pixelSplat (Charatan et al., 2024) and MVSplat (Chen et al., 2024) predict 3D Gaussians directly from posed images, and LRM (Hong et al., 2024) and GS-LRM (Zhang et al., 2024) scale this with large transformers. In the self-supervised, unposed setting, E-RayZer (Zhao et al., 2025) and NAS3R (Huang et al., 2026) render explicit Gaussians through a fixed rasteriser, and Pensieve (Wang et al., 2025b) aligns an LVSM-style implicit representation with 3D Gaussians in a second stage. The price is a commitment to explicit primitives and their non-convex optimisation, which E-RayZer counters with a visual-overlap curriculum. In contrast, LaPro never renders through explicit primitives: its predicted depth supervises the cameras during training, and the rendering process remains independent of it.

Geometry left to the renderer. SRT (Sajjadi et al., 2022) and LVSM (Jin et al., 2025) show that a purely learned renderer rivals explicit ones, but consume ground-truth cameras. RUST (Sajjadi et al., 2023), RayZer (Jiang et al., 2025), and RayDer (Prestel et al., 2026) self-predict the cameras instead; their photorealism is striking, but because the cameras are constrained only through the rendering loss, they need not correspond to physical motion and do not transfer across scenes (Mitchel et al., 2026). LaPro keeps this latent architecture and its expressivity; what changes is the training objective, which requires every predicted camera to satisfy how one view reprojects into another.

Geometry from data or a backbone. Recent work restores grounded cameras without explicit rendering by changing what the model learns from. XFactor (Mitchel et al., 2026) trains on pose-preserving augmentations of the same sequence to force pose apart from appearance, and LagerNVS (Szymanowicz et al., 2026) decodes novel views from the features of VGGT (Wang et al., 2025a), a backbone trained with 3D supervision. Unlike both, LaPro requires neither augmented training pairs nor a geometry-supervised backbone: it learns from raw, unposed multi-view sequences on top of a frozen, purely 2D self-supervised DINOv3 encoder (Siméoni et al., 2025).

Geometry through the objective. Photometric reprojection is a classical self-supervised signal: SfMLearner (Zhou et al., 2017) and Monodepth (Godard et al., 2017; 2019) learn depth and ego-motion by warping adjacent video frames. These methods treat depth and pose as the end goal and do not synthesise novel views. LaPro repurposes this objective: all-pairs reprojection over an unordered

input view set grounds the cameras of a latent renderer, with a depth serving only as an auxiliary variable that remains independent of the rendering process.

3 APPROACH

3.1 PROBLEM FORMULATION AND MODEL OVERVIEW

Problem statement. Let $\mathcal{I} = \{I_i\}_{i=1}^{V}$ be an unordered set of unposed, uncalibrated images of a static scene, with $I_i \in \mathbb{R}^{3\times H\times W}$, partitioned into disjoint context and target sets $\mathcal{I}_C$ and $\mathcal{I}_T$. Our model learns

$$\Phi : (\mathcal{I}_C, \mathcal{I}_T) \mapsto \big(\{P_i, K_i, D_i\}_{i=1}^{V}, \hat{\mathcal{I}}_T\big), \qquad P_i \in SE(3), \tag{1}$$

where P_i, K_i, and D_i denote camera-to-world pose, intrinsics, and dense depth, and $\hat{\mathcal{I}}_T$ contains the rendered targets. We train on sparse input views of five contexts and a single target image.

Why a fixed image-formation operator. As observed by Zhao et al. (2025), when the camera estimator and renderer are learned jointly, they only need to remain mutually compatible: a learned renderer can absorb an incorrect pose estimate, for example one encoding frame order rather than position, and still produce the same image. The rendering loss alone therefore does not favour physical cameras. Removing image-index embeddings does not change this: our architecture has none, yet trained on rendering alone its cameras reach 0.0% RPA@5 (Table 3). In constrast, a fixed operator cannot absorb such distortions. E-RayZer obtains one by making the renderer a rasteriser; we instead place it beside the renderer, as a differentiable warp between real views. For scenes with sufficient parallax, the warp is minimised only when poses, intrinsics, and depth jointly satisfy the photometric reprojection, which requires dense depth, hence our depth decoder, and constrains the same cameras the renderer consumes, while the novel view synthesis task remains fully latent.

Overview. LaPro jointly estimates cameras, depth and RGB from DINOv3 vision encoder features. It refines the DINOv3 tokens in a shared encoder to aggregate the multi-view information through inter-frame and local attention. Predicted cameras are then converted into pixel-aligned Plücker rays, which condition both a depth decoder and a latent RGB renderer. The depth decoder predicts a dense map for each observed view, enabling differentiable cross-warping between images. The renderer combines context appearance with context rays and synthesises a target from its predicted camera. Figure 3 depicts the pipeline.

3.2 CAMERA AND CONTEXT ENCODING

We leverage a frozen DINOv3-L/16 backbone that maps each image to patch tokens $F_i \in \mathbb{R}^{N\times d_0}$, where $N = HW/16^2$ and $d_0 = 1024$. Let Q and Z denote the learnable camera and register tokens appended to each view. Our encoder alternates within-view and inter-view attention to jointly update all three token groups:

$$(F^\star, Q^\star, Z^\star) = \mathcal{E}_{\text{cam}}(\{F, Q, Z\}; \mathcal{M}_{\text{attn}})\,. \tag{2}$$

The outputs are refined patch, camera, and register features of width $d = 768$. At selected inter-view stages, only camera and register tokens exchange information, providing an efficient route for multi-view aggregation. Other inter-view stages update all tokens. We use spatial positional encoding but no temporal or view-index embeddings, preserving permutation equivariance over the input views.

Directed information flow prevents target pixel leakage. The attention mask $\mathcal{M}_{\text{attn}}$ allows context queries to read only the context stream. Each target query token reads the context stream and attends to itself. This rule applies to patch, camera, and register tokens at every inter-view stage. Thus, the target camera prediction without passing its appearance into the context representation.

Camera gauge, intrinsics, and rays. A pose head reads each camera token together with a designated context-reference token, $[q_c^\star; q_i^\star]$, and predicts a 6D rotation representation and 3D translation. Before shuffling input frames, the canonical view is selected as the medoid in the sequence. Thus, our encoder takes as input a set of images. A second head reads $q_c^\star$ to predict a positive focal length shared across views, with $f_x = f_y$ and the principal point at the image centre. Each predicted camera (P_i, K_i) defines a dense Plücker map $R_i = [\mathbf{o}_i \times \mathbf{d}_i, \mathbf{d}_i] \in \mathbb{R}^{6\times H\times W}$.

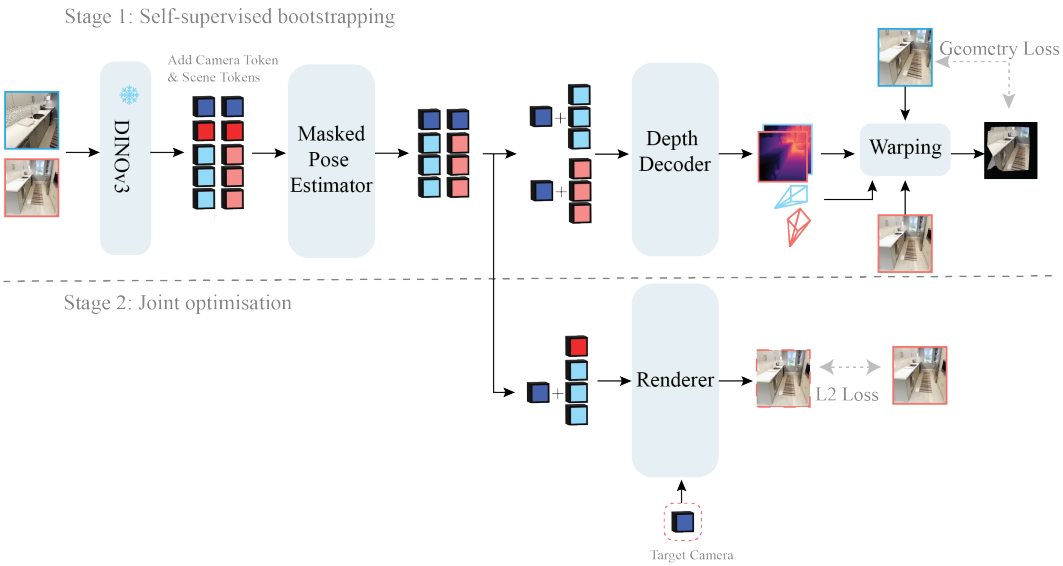

Figure 3: Overview of the LaPro training pipeline. An unordered camera estimator and dense depth decoder are first grounded through all-pairs differentiable cross-warping between observed views. A fresh ray-conditioned latent renderer is then introduced and jointly optimised with novel-view synthesis and the same projective objective. Depth supplies geometric supervision during training but is not rasterised by the final renderer.

3.3 Projective geometric supervision

Depth prediction. We add tokenized Plücker rays to the refined image features, $U_i = F_i^\star + E_R^{\text{depth}}(R_i) \in \mathbb{R}^{N\times d}$. For each view, a shallow transformer depth decoder uses these tokens as queries and the other views as context:

$$D_i = \mathcal{D}_{\text{depth}}(U_i, \{U_j\}_{j\neq i}), \qquad D_i \in \mathbb{R}^{H\times W}. \tag{3}$$

The module comprises three attention blocks followed by a DPT head that fuses their intermediate features into dense inverse-depth logits. We perform a normalization to bound the inverse-depth to stabilize the depth predictions. All observed views contribute to depth supervision; neither these depth maps nor the depth decoder's tokens are supplied to the renderer to prevent target leakage.

Cross-view inverse-warping. Our geometric supervision requires predicted cameras and depth to satisfy correspondence between observed images. Let $\mathcal{W}$ be the operator unprojects destination pixels with D_i, transforms them into camera j, and bilinearly samples I_j. For every ordered pair $i \neq j$, a differentiable inverse warp projects source image I_j into destination i:

$$\hat{I}_{i\leftarrow j} = \mathcal{W}(I_j; D_i, P_i, P_j, K_i, K_j). \tag{4}$$

Geometry objective. Following differentiable view synthesis (Zhou et al., 2017) and minimum-reprojection supervision (Godard et al., 2017; 2019), the photometric loss combines SSIM and Charbonnier residuals (Charbonnier et al., 1994), selects the lowest error among valid source projections, and masks pixels for which warping does not improve on the unwarped comparison. We also warp frozen DINO features and regularise inverse-depth smoothness:

$$\mathcal{L}_{\text{geom}} = \mathcal{L}_{\text{photo}} + \lambda_f \mathcal{L}_{\text{feat}} + \lambda_s \mathcal{L}_{\text{smooth}}. \tag{5}$$

Feature tensors remain frozen, while gradients through the sampling coordinates train predicted cameras and depth. This objective uses all observed views, including targets; the directed mask restricts appearance flow into the renderer, not the views available for geometric supervision.

3.4 Latent renderer

Context representation. We retain the refined context patches rather than compressing them into a fixed scene-token set. To do this, we gate the frozen DINO block-8 features and patch-embedded

RGB to $F_C^\star$, yielding $\widetilde{F}_C$. The camera estimator continues to use final-layer DINO features.

$$\widetilde{F}_C = F_C^\star + \gamma_8 \odot A_8(F_C^{(8)}) + \gamma_{\text{rgb}} \odot E_{\text{rgb}}(\mathcal{I}_C) \tag{6}$$

Camera conditioning. We then fuse the augmented features with embedded context rays, $X_C = \text{MLP}_{\text{fuse}}([\widetilde{F}_C; E_R^{\text{src}}(R_C)]) \in \mathbb{R}^{V_C \times N \times d}$, where brackets denote feature concatenation and the fusion module includes normalisation. Flattening X_C and appending the refined context registers $Z_C^\star$ gives source tokens x_C. All appearance inputs to this representation come exclusively from $\mathcal{I}_C$. We use PRoPE (Li et al., 2026) in renderer self- and cross-attention layers.

Rendering decoder. Our renderer is composed of 11 layers of apply self-attention applied to both context and target streams and bidirectional cross-attention. Source stream self-attention aggregates appearance information across contexts, while bidirectional updates enable the source features to adapt to the target query. Let $\mathcal{P}_C$ and $\mathcal{P}_t$ denote the predicted context and target cameras. For target t, we embed its predicted rays as $r_t = E_R^{\text{tgt}}(R_t)$. The rendering transformer updates target and source tokens:

$$(r_t^\star, x_C') = \mathcal{D}_{\text{render}}(r_t, x_C; \mathcal{P}_C, \mathcal{P}_t) \tag{7}$$

Progressive RGB decoding. We employ progressive RGB decoding for spatial refinement. We normalise the target features, followed by a linear projection that is progressively upsampled to 256x2566 resolution. Final RGB is reshaped through pixelshuffle.

$$\hat{I}_t = \mathcal{H}_{\text{rgb}}(r_t^\star), \qquad \hat{I}_t \in [0,1]^{3 \times H \times W} \tag{8}$$

3.5 Curriculum and joint optimisation

We first train the camera estimator and depth decoder with $\mathcal{L}_{\text{geom}}$, gradually expanding the sampled temporal window from nearby views to larger motion. We then introduce the rendering objective and retain the geometric objective to continue supervising depth and camera consistency as the renderer learns novel view synthesis. Exact schedules and gradient paths are specified in Appendix E. Thus, our joint objective is given as:

$$\mathcal{L}_{\text{joint}} = \underbrace{\text{MSE}(\hat{I}_t, I_t) + \lambda_p \mathcal{L}_{\text{VGG}}(\hat{I}_t, I_t)}_{\mathcal{L}_{\text{NVS}}} + \lambda_g \mathcal{L}_{\text{geom}}. \tag{9}$$

4 Experiments

4.1 Experimental setup

Data and implementation. We train on the cleaned DL3DV-10K release (Ling et al., 2024), containing 9,627 scenes, and reserve 97 complete scenes for validation. All images are square-cropped to 256^2. Each training sample contains six shuffled views, with five contexts and one target in Stage 2. We set $\lambda_f = 0.2$, $\lambda_p = 0.2$, and $\lambda_g = 2$, and train on eight B200 GPUs with an effective batch size of 256, for 80k iterations in Stage 1 and 85k in Stage 2, whose final checkpoint we evaluate. Full optimisation details are given in Appendix E.

Baselines. We compare with the released checkpoints of RayZer (Jiang et al., 2025), E-RayZer (Zhao et al., 2025), RayDer (Prestel et al., 2026), and, for cross-scene transfer, XFactor (Mitchel et al., 2026). RayDer is evaluated zero-shot, without training or fine-tuning on DL3DV. Each method runs through its own preprocessing, camera convention, and forward path on the same physical frames: every method receives the same five context views and is scored on the same five targets. The one exception is RayZer, which has a native split of 16 contexts and 8 targets.

Protocols. We evaluate in three settings. *Interpolation* draws contexts and targets from the same camera path, using all 140 DL3DV benchmark scenes. *Extrapolation* evaluation, unlike prior self-supervised latent methods, we follow the standard evaluation protocol commonly used in per-scene NeRF and 3DGS methods (Warburg et al., 2023) and condition on one trajectory to render targets from a second trajectory; we annotate the DL3DV benchmark, use only 122 scenes; 18 scenes are

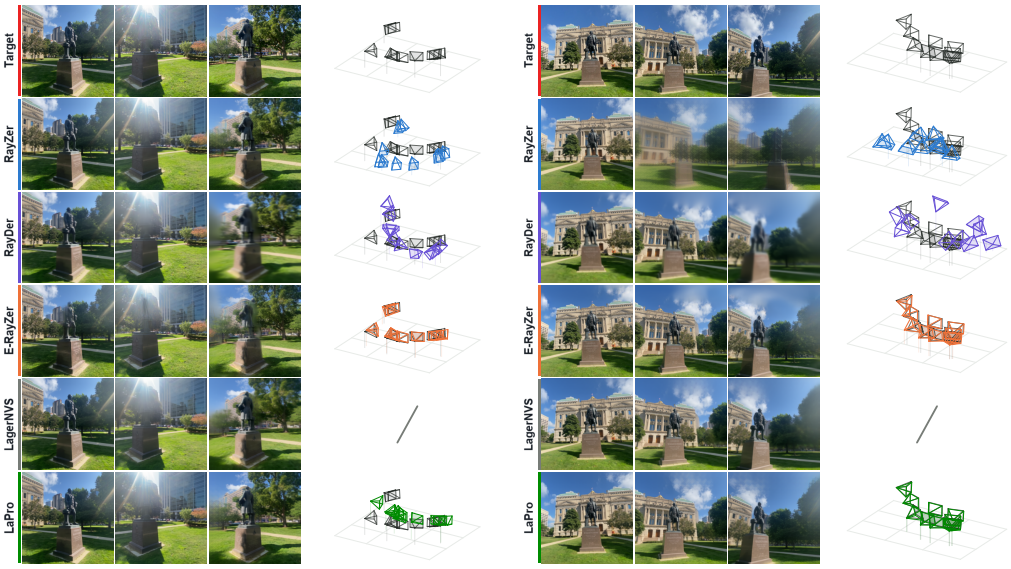

Figure 4: Image synthesis and model-predicted cameras overlaid on the ground-truth cameras: black denotes ground truth and method colors denote predictions. **Left:** interpolation; **right:** extrapolation. Only three of the five targets are displayed.

filtered out due to occlusions in distinct trajectories. This tests whether the learned renderer can synthesize novel views with predicted cameras that leave the input context trajectory. *Cross-scene transfer* follows XFactor's released TPS protocol: cameras inferred from a source scene drive the renderer of an unrelated target scene, over 4,000 ordered scene pairs.

Metrics. We report PSNR, SSIM, and LPIPS as image metrics. For camera accuracy, we evaluate the raw predicted cameras over the 45 view pairs of the five contexts and five targets. To quantify the motion of the rendered images, we run a frozen VGGT (Wang et al., 2025a) on the rendered and the ground-truth targets. Additionally, we report XFactor's proposed TPS metric through rotation accuracy, translation accuracy, and $SE(3)$ AUC.

4.2 Interpolation

Table 1 evaluates held-out targets sampled from the same trajectory as the contexts, following prior work (Jiang et al., 2025; Zhao et al., 2025). RayZer attains the strongest image quality (24.55 dB PSNR), yet its raw cameras reach 0.0% RPA@5 while its rendered motion remains coherent (79.3%), showing that the rendering objective alone does not require physical cameras. E-RayZer obtains the strongest raw pose accuracy (78.5% RPA@5). LaPro reaches 74.7% while retaining a fully latent renderer, and surpasses E-RayZer on all three image metrics (22.80 vs. 22.11 dB PSNR, 0.709 vs. 0.697 SSIM, 0.222 vs. 0.248 LPIPS).

4.3 Extrapolation

Table 2 evaluates targets on a second trajectory through the same scene. RayZer drops to 14.18 dB and 0.0% RPA@5. Among self-supervised methods, LaPro achieves the best PSNR (18.00 dB) and LPIPS (0.395), surpasses E-RayZer on all three image metrics, and obtains the strongest strict raw pose accuracy: 81.7% RTA@5, 81.3% RRA@5, and 76.4% RPA@5. LagerNVS, which relies on 3D-supervised VGGT, remains stronger in image quality. Figure 4 shows renders and predicted cameras for both settings.

Table 1: Interpolation on DL3DV ($W = 96$). RayDer-L-576 is evaluated zero-shot. Motion: RPA@5 estimated by VGGT from the rendered and ground-truth targets. Below the rule: LagerNVS uses a VGGT backbone and cameras trained with 3D supervision; not directly comparable. Full pose metrics appear in Appendix A.

	Method design				Image synthesis			Raw pose accuracy			Motion
Model	Self-sup.	Train	Latent	C/T	PSNR↑	SSIM↑	LPIPS↓	RTA@5↑	RRA@5↑	RPA@5↑	RPA@5↑
E-RayZer	✓	2D	✗	5/5	22.11	0.697	0.248	**82.9**	**84.9**	**78.5**	85.1
RayDer-L-576	✓	2D	✓	5/5	23.44	0.731	0.359	0.3	1.5	0.0	75.5
RayZer	✓	2D	✓	16/8	**24.55**	**0.765**	**0.197**	0.4	8.3	0.0	79.3
LaPro (ours)	✓	2D	✓	5/5	22.80	0.709	0.222	78.9	79.5	74.7	78.9
LagerNVS	✗	3D	✓	5/5	22.24	0.680	0.206	–	–	–	**92.7**

Table 2: Extrapolation on 122 annotated DL3DV scenes: contexts and targets lie on distinct trajectories that observe largely the same visible regions. Image metrics average the same five targets. RayDer-L-576 is evaluated zero-shot. Motion: RPA@5 estimated by VGGT from the rendered and ground-truth targets. Below the rule: LagerNVS uses a VGGT backbone and cameras trained with 3D supervision; not directly comparable. Full pose metrics appear in Appendix A.

	Method design				Image synthesis			Raw pose accuracy			Motion
Model	Self-sup.	Train	Latent	C/T	PSNR↑	SSIM↑	LPIPS↓	RTA@5↑	RRA@5↑	RPA@5↑	RPA@5↑
E-RayZer	✓	2D	✗	5/5	17.29	0.496	0.427	81.0	80.5	74.8	71.8
RayDer-L-576	✓	2D	✓	5/5	16.86	0.538	0.631	0.4	0.1	0.0	34.5
RayZer	✓	2D	✓	16/8	14.18	0.332	0.597	0.3	1.5	0.0	1.5
LaPro (ours)	✓	2D	✓	5/5	18.00	0.503	0.395	**81.7**	**81.3**	**76.4**	57.5
LagerNVS	✗	3D	✓	5/5	**19.20**	**0.572**	**0.320**	–	–	–	**90.0**

4.4 Cross-scene pose transferability

Figure 5 show the transferability metrics that showcase whether motion inferred from a source scene be transferred into a target scene. At the strict 10° threshold, LaPro nevertheless reaches 97.9% rotation accuracy, 84.4% translation accuracy, and 56.4% SE(3) AUC. Relative to XFactor, a method that applies an explicit pose-preserving augmentation mask, our results outperform the three metrics by 6.4, 26.9, and 29.6 percentage points, respectively; relative to RayZer, the gains are 45.8, 75.8, and 54.9 points. LaPro is therefore the strongest evaluated latent-rendering model under TPS. We refer the reader to Appendix B (Table 6) for full results.

4.5 Discussion

Photorealism does not imply physical cameras. On the input trajectory, RayZer renders best and a frozen VGGT recovers coherent motion from its renderings (79.3% RPA@5), yet its own cameras reach 0.0%. Its renderer and cameras are therefore mutually compatible but not physical: the camera acts as an interpolation cue that only its renderer can decode. Off the trajectory, this code fails. RayZer loses 10.4 dB and its rendered motion collapses to 1.5%; RayDer shows the same pattern ($75.5\% \rightarrow 34.5\%$).

Grounded cameras degrade like explicit geometry. LaPro loses 4.8 dB from interpolation to extrapolation, matching the explicit-geometry E-RayZer rather than the latent renderers, and its raw pose accuracy does not degrade ($74.7\% \rightarrow 76.4\%$ RPA@5), whereas E-RayZer's falls ($78.5\% \rightarrow 74.8\%$). A latent renderer trained with a fixed projective objective thus inherits the robustness of explicit geometry while surpassing E-RayZer on every image metric in both regimes. Without any 3D labels, LaPro also exceeds the PSNR of LagerNVS in interpolation, which relies on a 3D-supervised backbone and ground-truth query cameras.

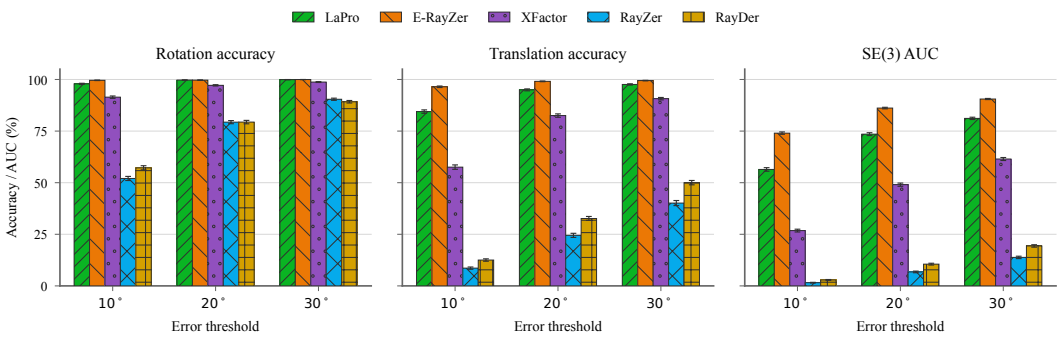

Figure 5: Cross-scene transferability across 4,000 ordered DL3DV pairs. Bars show the mean released TPS metric and error bars show deterministic ordered-pair bootstrap 95% confidence intervals. Each TPS estimate compares ten unordered relative motions between the donor and transferred T5 sequences. All methods use the same frozen pair manifest and VGGT scorer. RayDer-L-576 is evaluated zero-shot using its released checkpoint, without DL3DV training or fine-tuning.

Table 3: Geometry-source and NVS-only ablation. "Fixed external" denotes detached MapAnything depth. Initialisation and update count are reported explicitly because the available checkpoints are not compute-matched.

Variant	Initialisation	Updates	Depth for $\mathcal{L}_{\text{geom}}$	Geometry	Train depth	PSNR↑	LPIPS↓	RPA@5↑
LaPro (original renderer)	Stage-1 80k	85k	Learned	$\lambda_g = 2$	Yes	**21.54**	**0.298**	**74.0**
Fixed external depth	From scratch	90k	MapAnything (detached)	$\lambda_g = 2$	No	20.02	0.371	57.5
NVS only	From scratch	30k	None	None	No	17.65	0.488	0.0

4.6 Ablations

Geometry source. We isolate the role of the geometry objective with two variants trained from scratch with the same data, sampler, optimiser, and learning rate: one replaces the learned depth with fixed, detached MapAnything (Keetha et al., 2026) depth, and one removes depth and $\mathcal{L}_{\text{geom}}$ entirely. We stopped the NVS-only run at 30k updates, as its raw pose accuracy remained at 0.0% throughout training with no upward trend. Rendering alone reaches 17.65 dB and collapses raw pose accuracy to 0.0% RPA@5, consistent with RayZer. Adding the warping objective with fixed external depth raises these to 20.02 dB and 57.5%, showing that projective consistency itself grounds the cameras; the full model reaches 21.54 dB and 73.8%.

5 Conclusion

We introduced LaPro, a self-supervised feed-forward NVS model that uses dense projective geometry to ground a latent representations and predicted cameras in geometry without resorting to explicit structures. Our proposed method, LaPro, preserves competitive interpolation quality while recovering far more physical cameras than latent NVS trained from image reconstruction alone. Under the extrapolation setting, it obtains the best PSNR, LPIPS and strict raw pose accuracy among the self-supervised models. The remaining gap is appearance rather than camera controllability. E-RayZer retains stronger rendered-motion scores in extrapolation, and the qualitative results show that targets revealing unseen content remain difficult for a deterministic latent renderer. This separates two problems that ordinary interpolation conflates: learning a camera representation that generalises beyond the observed path, and synthesising plausible appearance where no context image provides evidence. LaPro substantially closes the first while leaving the second as the clearest direction for future work.

AI use statement

We used generative AI tools in this work. Parts of the model and evaluation code were implemented with AI under the authors' direction, and we used AI to provide feedback on the research methodology,

experimental design and evaluation protocols and to help articulate the conceptual arguments in the paper. The authors validated the implementation through end-to-end checks, including training behaviour and agreement of baseline results with independently reported values. All reported numbers come from our own runs. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Reproducibility statement

Section 3 describes the architecture and training objectives, and Section 4 the data, evaluation protocols, and metrics. Appendix E lists the hyperparameters and training schedules. All baselines use publicly released checkpoints. We will release our code, trained models, and the extrapolation trajectory annotations.

References

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

Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *IEEE International Conference on Image Processing (ICIP)*, volume 2, pp. 168–172, 1994.

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

Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth estimation. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

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 *International Conference on Learning Representations (ICLR)*, 2024.

Ranran Huang, Weixun Luo, Ye Mao, and Krystian Mikolajczyk. From none to all: Self-supervised 3D reconstruction via novel view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 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 *IEEE/CVF International Conference on Computer Vision (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 *International Conference on Learning Representations (ICLR)*, 2025.

Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, et al. Mapanything: Universal feed-forward metric 3d reconstruction; map-anything. github. io. In *2026 International Conference on 3D Vision (3DV)*, pp. 499–509. IEEE, 2026.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as relative positional encoding. *Advances in Neural Information Processing Systems*, 38:15984–16009, 2026.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 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 *European Conference on Computer Vision (ECCV)*, 2020.

Thomas W. Mitchel, Hyunwoo Ryu, and Vincent Sitzmann. True self-supervised novel view synthesis is transferable. In *International Conference on Learning Representations (ICLR)*, 2026.

Ulrich Prestel, Stefan Andreas Baumann, Nick Stracke, and Björn Ommer. RayDer: Scalable self-supervised novel view synthesis from real-world video. *arXiv preprint arXiv:2605.31535*, 2026.

Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lučić, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas Funkhouser, and Andrea Tagliasacchi. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Mehdi S. M. Sajjadi, Aravindh Mahendran, Thomas Kipf, Etienne Pot, Daniel Duckworth, Mario Lučić, and Klaus Greff. RUST: Latent neural scene representations from unposed imagery. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

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

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

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Ruoyu Wang, Yi Ma, and Shenghua Gao. Recollection from pensieve: Novel view synthesis via learning from uncalibrated videos. *arXiv preprint arXiv:2505.13440*, 2025b.

Frederik Warburg, Ethan Weber, Matthew Tancik, Aleksander Holynski, and Angjoo Kanazawa. Nerfbusters: Removing ghostly artifacts from casually captured NeRFs. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

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 *European Conference on Computer Vision (ECCV)*, 2024.

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.

Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

APPENDIX

A COMPLETE POSE RESULTS

Table 4: Interpolation pose accuracies on 140 scenes. Top: raw model-predicted C5+T5 cameras, evaluated without alignment over 45 pairs per scene. Bottom: VGGT-estimated motion from rendered and ground-truth T5 sequences, evaluated over ten pairs per scene. These are the saved mean accuracies, in percent; higher is better. RayDer-L-576 uses its released zero-shot checkpoint. † LagerNVS uses GT query extrinsics and GT context-based scale; raw camera accuracy is not applicable. Its rendered-motion scores use the same VGGT oracle. Bold denotes the highest available accuracy in each column, including the GT-query reference.

Raw pose (C5+T5)

	RTA↑			RRA↑			RPA↑		
Model	@5°	@15°	@30°	@5°	@15°	@30°	@5°	@15°	@30°
E-RayZer	**82.9**	**93.9**	**96.0**	**84.9**	**92.8**	**96.1**	**78.5**	**91.3**	**95.1**
RayDer-L-576	0.3	2.7	10.7	1.5	7.1	17.3	0.0	0.2	2.0
LagerNVS†	–	–	–	–	–	–	–	–	–
RayZer	0.4	3.8	13.7	8.3	30.3	56.9	0.0	1.2	9.0
LaPro (ours)	78.9	87.6	91.9	79.5	85.7	90.0	74.7	83.7	88.4

Rendered motion (VGGT, T5)

Model	RTA@5↑	RRA@5↑	RPA@5↑	RPA@15↑	RPA@30↑
E-RayZer	88.6	89.3	85.1	93.1	95.9
RayDer-L-576	78.2	85.1	75.5	89.1	93.4
LagerNVS†	**94.1**	**95.1**	**92.7**	**95.5**	96.5
RayZer	83.9	89.2	79.3	95.4	**98.1**
LaPro (ours)	82.7	87.8	78.9	92.4	95.3

Table 5: Extrapolation pose accuracies on 122 scenes. Top: raw model-predicted C5+T5 cameras, evaluated without alignment over 45 pairs per scene. Bottom: VGGT-estimated motion from rendered and ground-truth T5 sequences, evaluated over ten pairs per scene. These are the saved mean accuracies, in percent; higher is better. RayDer-L-576 uses its released zero-shot checkpoint. † LagerNVS uses GT query extrinsics and GT context-based scale; raw camera accuracy is not applicable. Its rendered-motion scores use the same VGGT oracle. Bold denotes the highest available accuracy in each column, including the GT-query reference.

Raw pose (C5+T5)

	RTA↑			RRA↑			RPA↑		
Model	@5°	@15°	@30°	@5°	@15°	@30°	@5°	@15°	@30°
E-RayZer	81.0	**92.8**	**96.2**	80.5	**92.3**	**95.4**	74.8	**90.5**	**94.5**
RayDer-L-576	0.4	3.0	12.5	0.1	1.4	8.0	0.0	0.1	1.0
LagerNVS†	–	–	–	–	–	–	–	–	–
RayZer	0.3	3.5	13.7	1.5	15.6	43.8	0.0	0.5	8.3
LaPro (ours)	**81.7**	91.0	93.6	**81.3**	90.7	93.5	**76.4**	88.7	92.0

Rendered motion (VGGT, T5)

Model	RTA@5↑	RRA@5↑	RPA@5↑	RPA@15↑	RPA@30↑
E-RayZer	77.5	80.7	71.8	91.3	96.2
RayDer-L-576	41.1	56.7	34.5	69.6	85.9
LagerNVS†	**92.3**	**94.6**	**90.0**	**97.1**	**99.3**
RayZer	7.5	7.0	1.5	11.4	30.5
LaPro (ours)	62.8	71.0	57.5	78.0	86.1

B CROSS-SCENE TRANSFERABILITY RESULTS

Table 6: Cross-scene transferability on DL3DV. We sample 4,000 unique ordered donor/recipient pairs from all 140 benchmark scenes at $W = 24$. Each transferred T5 sequence is compared with its donor T5 using the released XFactor/VGGT TPS metric over ten unordered target pairs. RayDer-L-576 is evaluated zero-shot using its released checkpoint, without DL3DV training or fine-tuning. Values are percentages; higher is better. Each cell reports mean (per-pair median). Best and second-best mean results are bold and underlined, respectively.

Metric	τ	**LaPro (ours)**	**E-RayZer**	**XFactor**	**RayZer**	**RayDer-L-576**
R. Acc.	10°	<u>97.9</u> (100.0)	**99.6** (100.0)	91.4 (100.0)	52.0 (50.0)	57.2 (60.0)
	20°	**99.7** (100.0)	**99.7** (100.0)	<u>97.2</u> (100.0)	79.3 (90.0)	79.4 (100.0)
	30°	<u>99.9</u> (100.0)	**100.0** (100.0)	98.7 (100.0)	90.4 (100.0)	89.2 (100.0)
T. Acc.	10°	<u>84.4</u> (100.0)	**96.5** (100.0)	57.5 (60.0)	8.6 (0.0)	12.5 (0.0)
	20°	<u>95.0</u> (100.0)	**99.1** (100.0)	82.5 (100.0)	24.5 (10.0)	32.7 (20.0)
	30°	<u>97.6</u> (100.0)	**99.5** (100.0)	90.7 (100.0)	40.1 (30.0)	50.0 (50.0)
AUC	10°	<u>56.4</u> (64.0)	**74.0** (79.0)	26.8 (23.0)	1.6 (0.0)	2.9 (0.0)
	20°	<u>73.5</u> (81.5)	**86.2** (89.5)	49.1 (52.0)	6.8 (0.0)	10.5 (4.0)
	30°	<u>81.1</u> (87.7)	**90.5** (93.0)	61.5 (67.3)	13.8 (5.7)	19.3 (12.3)

C LEARNED DEPTH

Figure 6 shows depth predicted by our self-supervised depth branch, alongside MapAnything pseudo-depth for reference.

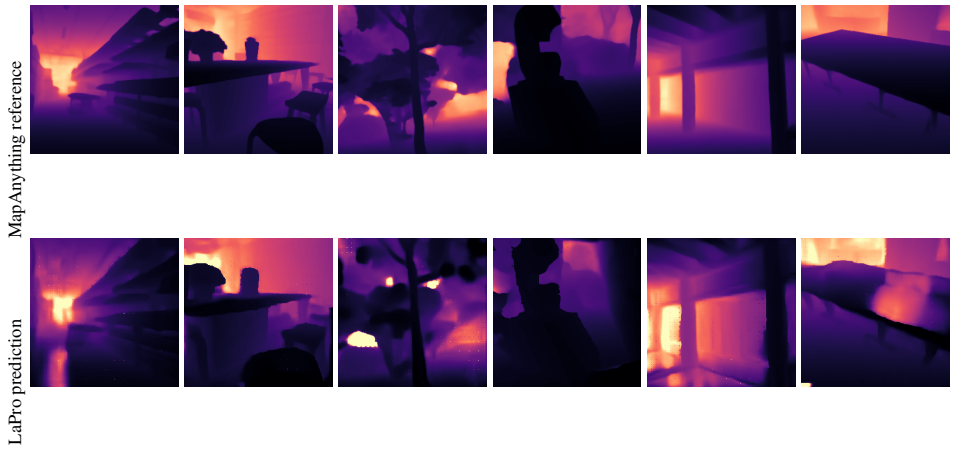

Figure 6: Qualitative depth predictions near the end of Stage 2. Top: MapAnything pseudo-depth, shown for reference only; it does not supervise our depth branch. Bottom: LaPro's self-supervised prediction. Colour scaling differs between rows, so colour indicates recovered structure rather than metric agreement.

D ADDITIONAL QUALITATIVE COMPARISONS

The following pages retain the remaining scene selections from the interpolation and extrapolation comparisons. Each page contains four blocks: interpolation on the left and extrapolation on the right, with two scenes per regime. The left and right blocks need not depict the same scene. Each block shows target ranks 1, 3, and 5 for every method; all five common context and five common target cameras are displayed. Cameras are direct model-predicted SE(3) poses, with the same context-only display alignment as Figure 4; no VGGT-estimated cameras are used. The reference grid aids depth perception and is not an estimated physical floor. RayDer-L-576 retains its native preprocessing, with images resized only for display.



Figure 7: Additional qualitative comparisons. Left: interpolation; right: extrapolation. Top: interpolation record 000014 and extrapolation record 000009. Bottom: interpolation record 000019 and extrapolation record 000054. Each block compares methods on the same three target frames. Camera conventions follow Figure 4. LagerNVS uses supplied GT query cameras and GT context-based scale; its raw-pose panel is not applicable.

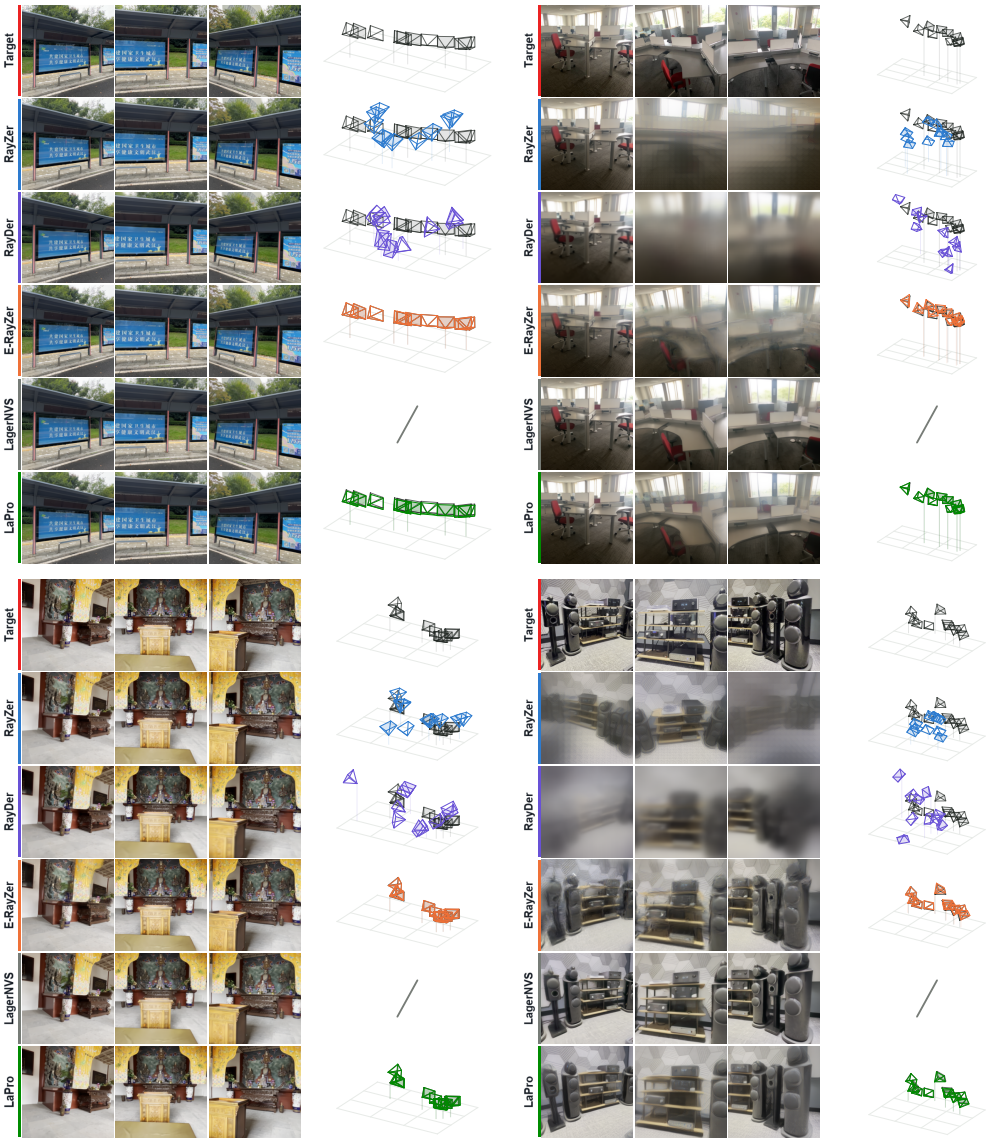

Figure 8: Additional qualitative comparisons. Left: interpolation; right: extrapolation. Top: interpolation record 000021 and extrapolation record 000058. Bottom: interpolation record 000018 and extrapolation record 000065. Each block compares methods on the same three target frames. Camera conventions follow Figure 4. LagerNVS uses supplied GT query cameras and GT context-based scale; its raw-pose panel is not applicable.



Figure 9: Additional qualitative comparisons. Left: interpolation; right: extrapolation. Top: interpolation record 000049 and extrapolation record 000076. Bottom: interpolation record 000065 and extrapolation record 000111. Each block compares methods on the same three target frames. Camera conventions follow Figure 4. LagerNVS uses supplied GT query cameras and GT context-based scale; its raw-pose panel is not applicable.

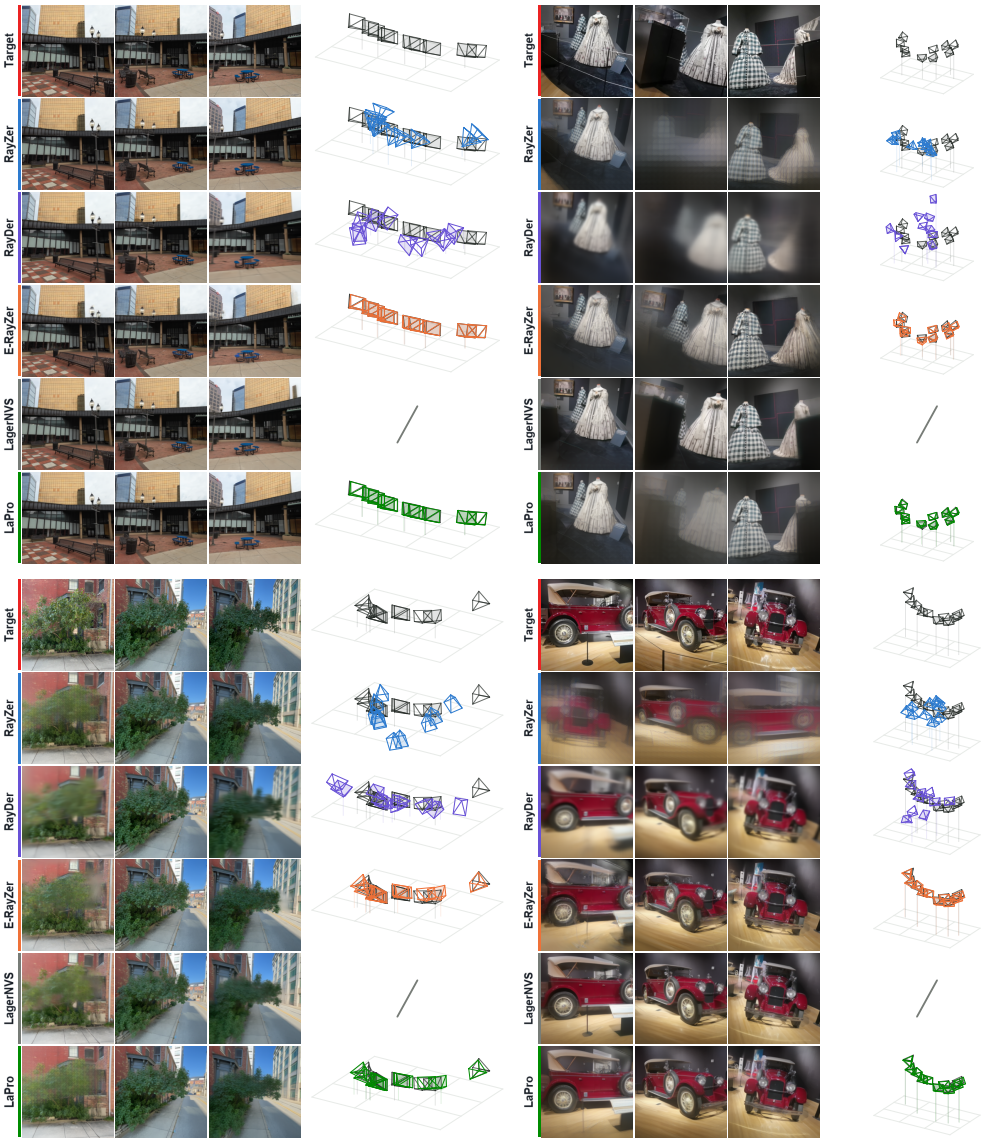

Figure 10: Additional qualitative comparisons. Left: interpolation; right: extrapolation. Top: interpolation record 000008 and extrapolation record 000116. Bottom: interpolation record 000091 and extrapolation record 000112. Each block compares methods on the same three target frames. Camera conventions follow Figure 4. LagerNVS uses supplied GT query cameras and GT context-based scale; its raw-pose panel is not applicable.

E METHOD AND IMPLEMENTATION DETAILS

This appendix expands the module-level equations in the main text. We retain batch size B, use $V_C = 5$ contexts and one target during training, and denote $N = 256$ patches per image and working width $d = 768$. Learned camera and register embeddings are denoted by Q and Z before refinement. The following details specify the camera encoder, depth and RGB decoders, geometric losses, and training schedules.

E.1 CAMERA ESTIMATOR

We square-crop images to $H = W = 256$ and extract final-layer patch features from a frozen DINOv3-L/16 backbone. For batch size B, $N = 16 \times 16 = 256$ patches, and backbone width $C_0 = 1024$, these features have shape $F \in \mathbb{R}^{B\times V\times N\times C_0}$. We append one learnable camera token and 16 learnable register tokens per view. A transformer built from VGGT-Omega blocks alternates within-view and inter-view attention over eight stages. Inter-view attention acts on camera and register tokens only at stages 3 and 6, and on all tokens at the remaining stages. Within-view attention uses spatial rotary positional encoding on the patch grid. We concatenate the final within-view and inter-view states and project their 2048 channels to $d = 768$:

$$(Q^\star, Z^\star, F^\star) = \mathrm{split}(W_h \mathcal{T}_{\mathrm{cam}}(\{F, Q, Z\}; \mathcal{M}_{\mathrm{attn}})), \qquad F^\star \in \mathbb{R}^{B\times V\times N\times d}, \tag{10}$$

Here $\mathcal{T}_{\mathrm{cam}}$ denotes the attention stack before output projection, and split selects the camera, register, and patch token groups; it introduces no learned operation. This is the internal realisation of the module $\mathcal{E}_{\mathrm{cam}}$ in Eq. 2. The projected sequence has shape $B \times V \times (N + 17) \times d$, with $Q^\star$ containing one camera token and $Z^\star$ containing 16 register tokens per view. Camera and register embeddings are shared across views; there are no temporal or view-index embeddings, so the encoder is permutation-equivariant.

Directed information flow. In every inter-view stage, context queries read only context views, including their own view. Each target reads the contexts and itself:

$$\mathcal{A}(i) = \begin{cases} \mathcal{I}_C, & I_i \in \mathcal{I}_C, \\ \mathcal{I}_C \cup \{I_i\}, & I_i \in \mathcal{I}_T. \end{cases} \tag{11}$$

The same rule applies to camera and register tokens, preventing an indirect route for target appearance into context features. For multi-target evaluation, target branches remain independent: I_t never attends to another target. Target images therefore inform their own camera estimates without providing visual features to the renderer.

Camera gauge and rays. During training, the sampler selects the middle frame of the sorted window as the canonical view c, tracks its index through the random view permutation, and always retains it in the context set. Evaluation similarly designates a context reference. A pairwise head maps $[q_c^\star; q_i^\star]$ to a 6D rotation representation and 3D translation, with $P_c = I_4$. The canonical token also predicts a positive focal length shared across views, with $f_x = f_y$ and the principal point fixed at the image centre. These choices fix the reference frame, without providing metric-scale supervision. Predicted poses and intrinsics define dense Plücker ray maps $R_i = [\mathbf{o}_i \times \mathbf{d}_i, \mathbf{d}_i] \in \mathbb{R}^{6\times H\times W}$.

E.2 DENSE DEPTH DECODER

The module $\mathcal{D}_{\mathrm{depth}}$ in Eq. 3 includes ray-conditioned attention, the DPT head, and the bounded-depth mapping specified below. A stride-16 convolution embeds each patch's dense rays into d channels. For view i, we add these ray tokens to its refined features and use three attention blocks to read the other views:

$$q_i^0 = F_i^\star + E_R^{\mathrm{depth}}(R_i), \qquad X_{-i} = \mathrm{concat}_{j\neq i}\big(F_j^\star + E_R^{\mathrm{depth}}(R_j)\big), \tag{12}$$

$$q_i^\ell = \mathcal{B}_{\mathrm{depth}}^\ell(q_i^{\ell-1}, X_{-i}), \qquad \ell = 1, 2, 3, \tag{13}$$

$$\bar{z}_i = \mathrm{DPT}(q_i^0, q_i^1, q_i^2, q_i^3). \tag{14}$$

Here $q_i^\ell \in \mathbb{R}^{B\times N\times d}$ and $X_{-i} \in \mathbb{R}^{B\times(V-1)N\times d}$. Each block applies query self-attention, cross-attention, and an MLP. The DPT head fuses the four query states into a full-resolution inverse-depth

logit residual. We subtract its mean over all views and pixels of each scene, add a fixed logit $b_{2\mathrm{m}}$ corresponding to an initial depth of 2, and map to bounded depth:

$$z_i = \bar{z}_i - \operatorname*{mean}_{j,\mathbf{u}} \bar{z}_j(\mathbf{u}) + b_{2\mathrm{m}}, \qquad D_i = \left[\sigma(z_i)\left(\frac{1}{d_{\min}} - \frac{1}{d_{\max}}\right) + \frac{1}{d_{\max}}\right]^{-1}, \tag{15}$$

with $[d_{\min}, d_{\max}] = [0.1, 20]$. Centering acts on the logit residual, not on the depth map, and limits global depth drift. This final mapping is computed in FP32.

E.3 Renderer

Context appearance and ray fusion. The renderer retains all context patch features rather than compressing them into a fixed scene-code bottleneck. To preserve appearance information, we augment $F_C^\star$ with context-only features $F_C^{(8)}$ from the output of DINO's eighth transformer block and a learned RGB patch embedding:

$$\widetilde{F}_C = F_C^\star + \gamma_8 \odot A_8(F_C^{(8)}) + \gamma_{\mathrm{rgb}} \odot E_{\mathrm{rgb}}(\mathcal{I}_C), \tag{16}$$

$$X_C = \mathrm{MLP}_{\mathrm{fuse}}\Big(\mathrm{LN}[\widetilde{F}_C; E_R^{\mathrm{src}}(R_C)]\Big), \qquad X_C \in \mathbb{R}^{B\times V_C\times N\times d}. \tag{17}$$

The adapter A_8 normalizes the frozen 1024-dimensional features and projects them to $d = 768$; $\gamma_8, \gamma_{\mathrm{rgb}} \in \mathbb{R}^d$ are learned channel-wise gates initialized to 0.1. RGB and dense source rays use separate stride-16 convolutions. The fusion MLP operates independently at each patch. The block-8 branch supplements the refined context features; the camera encoder continues to consume final-layer DINO features.

Source and target interaction. We flatten X_C across views and append the 16 refined registers from each context to obtain source tokens x^0. Target tokens r^0 consist of normalized, patch-embedded target rays and four learned decoder registers. With five contexts, $x^0 \in \mathbb{R}^{B\times 1360\times 768}$ and $r^0 \in \mathbb{R}^{B\times 260\times 768}$. Each of eleven bidirectional blocks first applies self-attention independently to the source and target streams, then exchanges information in both directions:

$$\bar{r}^\ell = r^{\ell-1} + \mathrm{SA}_t^\ell(r^{\ell-1}), \qquad \bar{x}^\ell = x^{\ell-1} + \mathrm{SA}_s^\ell(x^{\ell-1}), \tag{18}$$

$$r^\ell = \mathrm{FFN}_t^\ell\Big(\bar{r}^\ell + \mathrm{CA}_{t\leftarrow s}^\ell(\bar{r}^\ell, \bar{x}^\ell)\Big), \qquad x^\ell = \mathrm{FFN}_s^\ell\Big(\bar{x}^\ell + \mathrm{CA}_{s\leftarrow t}^\ell(\bar{x}^\ell, \bar{r}^\ell)\Big). \tag{19}$$

Attention operators include pre-normalization, and $\mathrm{FFN}(z) = z + \mathrm{MLP}(\mathrm{LN}(z))$. Source self-attention connects patches and registers across all context views; bidirectional cross-attention makes their subsequent refinement query-dependent. A twelfth block applies target self-attention, target-to-source cross-attention, and an MLP, updating only the target stream. No target image features enter these operations.

As in the main text, PRoPE is used in renderer self- and cross-attention with live predicted poses and intrinsics; these camera inputs are not detached.

Progressive RGB decoding. The head $\mathcal{H}_{\mathrm{rgb}}$ in Eq. 7 combines the following linear and progressive branches. After removing decoder registers and normalizing the final target patch features $r^\star \in \mathbb{R}^{B\times N\times 768}$, a linear head predicts patch RGB logits. A parallel convolutional branch reshapes $r^\star$ to a 16×16 feature grid and progressively upsamples to 32×32 with 384 channels and 64×64 with 192 channels. Each scale uses two residual convolutional blocks; a final pixel-shuffle projection produces a 256×256 RGB-logit correction:

$$\hat{I}_t = \sigma(\mathrm{Unpatch}(W_{\mathrm{rgb}} r^\star) + \mathcal{D}_{\mathrm{prog}}(r^\star)). \tag{20}$$

The correction's final projection is zero-initialized, preserving the linear head's initial output while allowing spatial refinement across patch boundaries. The resulting image is decoded entirely in the latent renderer, without rasterizing an explicit 3D representation.

E.4 Real-view differentiable cross-warping

For every ordered pair of observed views (i, j), $i \neq j$, we inverse-warp source image I_j into destination i. For homogeneous pixel coordinate $\mathbf{u}$ in view i,

$$\mathbf{X}_i = D_i(\mathbf{u}) K_i^{-1}\mathbf{u}, \qquad \tilde{\mathbf{u}}_j = K_j\left[P_j^{-1}P_i\begin{bmatrix}\mathbf{X}_i\\ 1\end{bmatrix}\right]_{1:3}. \tag{21}$$

Perspective division and bilinear sampling give $\mathcal{W}_{i\leftarrow j}$. We retain projections with positive depth inside the source image. Following differentiable view-synthesis supervision (Zhou et al., 2017) and minimum-reprojection auto-masking (Godard et al., 2017; 2019), we minimize over valid source views and retain a pixel only if its minimum warped error is below its minimum unwarped error. The photometric residual combines SSIM dissimilarity and a Charbonnier penalty ρ (Charbonnier et al., 1994):

$$\mathcal{L}_{\text{photo}} = \frac{1}{|\mathcal{M}|} \sum_{i,\mathbf{u}\in\mathcal{M}_i} \min_{j\in\mathcal{V}_i(\mathbf{u})} \left[\alpha \, \frac{1 - \text{SSIM}(I_i, \mathcal{W}_{i\leftarrow j})}{2} + (1-\alpha)\rho(I_i - \mathcal{W}_{i\leftarrow j}) \right], \tag{22}$$

where $\mathcal{V}_i(\mathbf{u})$ contains valid sources and $\mathcal{M}_i$ is the auto-mask. We average this loss equally over four scales $s \in \{1, 2, 4, 8\}$. We also warp frozen final-layer DINO features at patch resolution and minimize their cosine distance, $\mathcal{L}_{\text{feat}}$, and apply edge-aware inverse-depth smoothness $\mathcal{L}_{\text{smooth}}$:

$$\mathcal{L}_{\text{geom}} = \mathcal{L}_{\text{photo}} + \lambda_f \mathcal{L}_{\text{feat}} + \lambda_s \mathcal{L}_{\text{smooth}}. \tag{23}$$

The feature tensors are detached, while the sampling coordinates remain differentiable with respect to predicted cameras and depth. Geometry supervision uses all observed views, including the target; the directed mask restricts appearance flow into the renderer, not the views used in this loss.

E.5 Curriculum and two-stage optimisation

We first train the camera estimator and depth decoder with $\mathcal{L}_{\text{geom}}$. To establish geometry before exposing the model to large motion, each six-view sample is drawn from a temporal window whose maximum length grows linearly:

$$W_{\max}(k; T) = \left\lfloor 6 + 26 \min\left(\frac{k}{T}, 1\right) \right\rfloor, \qquad W \sim \mathcal{U}\{6, \ldots, W_{\max}(k; T)\}, \tag{24}$$

with $T = 20{,}000$ Stage-1 updates. Both window endpoints are included, four interior frames are sampled without replacement, and the six views are randomly permuted. The target is sampled from the non-canonical views, keeping the reference in the context set. Smoothness is warmed up as $\lambda_s(k) = 10^{-3} \min(k/2{,}000, 1)$.

Stage 2 loads the pretrained camera and depth modules and initializes a fresh renderer, including its appearance adapters and progressive decoder. All three modules are jointly optimized with

$$\mathcal{L}_{\text{joint}} = \underbrace{\text{MSE}(\hat{I}_t, I_t) + \lambda_p \mathcal{L}_{\text{VGG}}(\hat{I}_t, I_t)}_{\mathcal{L}_{\text{NVS}}} + \lambda_g \mathcal{L}_{\text{geom}}. \tag{25}$$

The current warm-start recipe uses a shorter window ramp with $T = 1{,}000$ Stage-2 updates, then samples the full range $W \in [6, 32]$. Smoothness starts at its full weight in Stage 2. Camera and depth modules use $0.1\times$ the renderer learning rate; DINOv3 and the perceptual network remain frozen. Rendering gradients reach the camera encoder through refined context tokens, registers, Plücker rays, and live PRoPE transforms, while the depth decoder is trained by the geometric objective. Joint training therefore retains the projective constraint as the renderer learns appearance synthesis.