FEEDFORWARD NOVEL VIEW SYNTHESIS USING LATENT REPRESENTATIONS FOR IN-THE-WILD IMAGES

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent feed-forward models have shown promise for novel view synthesis (NVS) from unconstrained photo collections. However, existing training setups offer limited coverage of real-world appearance variation, transient occlusion, or diverse source-view configurations, while differences in evaluation protocols complicate comparisons. We introduce the Wild-NVS dataset and benchmark, curated from internet photo collections, with a sampling strategy that constructs source–target view sets using bidirectional covisibility, joint target coverage, and viewpoint diversity. The benchmark provides scene-level splits and fixed evaluation protocols across multiple source-view counts. We also propose Wild-NVS, a feed-forward model that synthesizes target views from latent representations. It combines bottlenecked appearance conditioning with a learned transient prediction head that guides source-feature suppression before decoding. Experiments on Wild-NVS and PhotoTourism demonstrate state-of-the-art performance among deterministic feed-forward methods for in-the-wild NVS, with improved appearance modeling and reduced transient artifacts. Our models and dataset will be made public.

1 INTRODUCTION

Feed-forward novel view synthesis (NVS) enables rendering from sparse image collections without per-scene optimization. Recent methods either predict explicit 3D Gaussians (Lin et al., 2025; Jiang et al., 2025) or synthesize target views directly from latent representations (Jin et al., 2024; Szymanowicz et al., 2026). However, these models remain challenged by unconstrained photo collections, where images of the same scene can differ substantially in illumination, weather, exposure, and camera processing. Transient objects, such as pedestrians, vehicles, and temporary structures, further occlude the underlying scene. A model must therefore synthesize a coherent static scene from incomplete observations while accounting for image-dependent appearance.

Recent methods have begun to address these challenges within feed-forward Gaussian splatting. GenWildSplat (Gupta et al., 2026) augments existing NVS training data with synthetic lighting variations and transient objects, and uses segmentation masks to suppress transients. However, its synthetic augmentations capture a narrower range of appearance variations than the large-scale real-world photo collections in our proposed dataset. WildSplatter (Fujimura et al., 2026) trains on source-view pairs sampled from MegaScenes (Tung et al., 2024) and restricts supervision to regions with cross-view depth consistency. Its pair-based training does not expose the model to configurations that require combining complementary observations from more than two source views. Our evaluations nonetheless reveal persistent difficulties in suppressing transient artifacts in both methods. Furthermore, differences in evaluation datasets and protocols complicate comparisons, motivating a common benchmark with clearly specified protocols.

To address these limitations, we introduce the Wild-NVS dataset and benchmark for novel view synthesis from in-the-wild images. Built upon the scene reconstructions provided by MegaDepth-X (Li et al., 2026), Wild-NVS captures real variations in appearance and transient occlusion. Our sampling strategy constructs source–target view sets using bidirectional covisibility, joint target coverage, and viewpoint diversity, providing geometrically verified training examples with multiple source views. We establish scene-level training, validation, and test splits, together with explicit source-view selection and evaluation protocols for $\{4, 6, 8, 10, 12\}$ view inputs, providing a common benchmark for assessing feed-forward methods.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

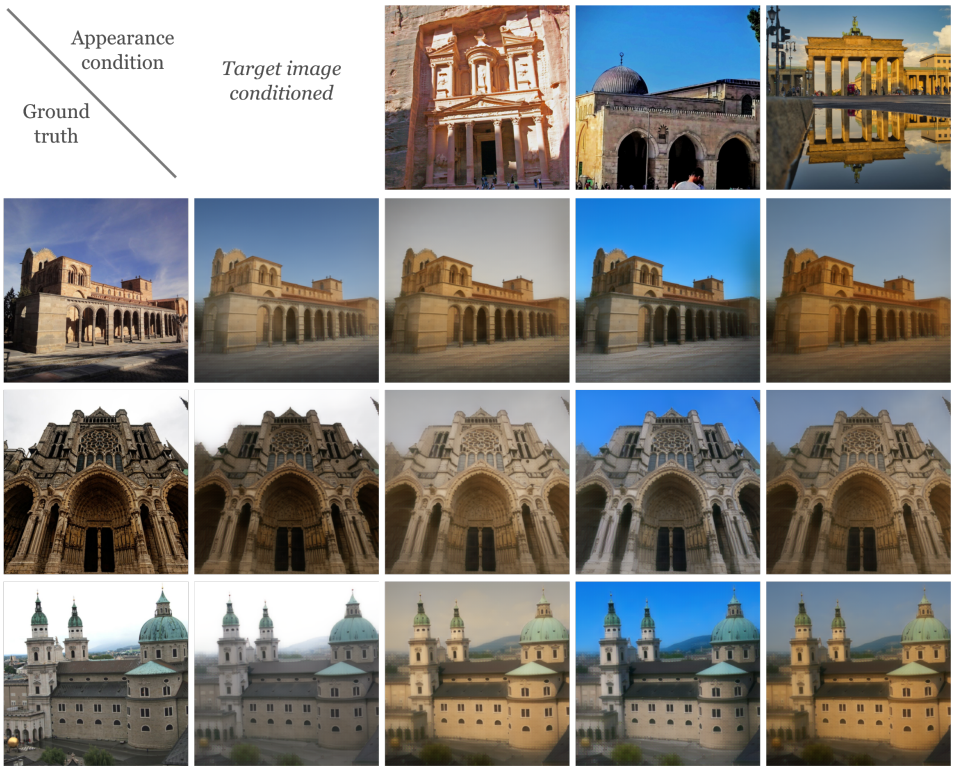

Figure 1: Appearance conditioning to target views using images from a different scene. Each row denotes a single target rendered using different appearance conditioning views (shown in row 1).

We also propose Wild-NVS, a feed-forward model that extends LagerNVS (Szymanowicz et al., 2026) with appearance conditioning and transient feature suppression. Unlike prior deterministic in-the-wild methods, which predict explicit 3D Gaussians, Wild-NVS encodes the source views into latent tokens and decodes the target view from them. An appearance encoder extracts a compact code from a conditioning image, allowing the decoder to render the desired appearance through a bottleneck designed to limit scene-specific structural information. Decoding from latent tokens allows us to leverage a transient prediction head to identify potentially transient content in the source tokens and guide the decoder to reduce the influence of transients in the synthesized view. Experiments on the Wild-NVS benchmark and PhotoTourism demonstrate the benefits of both dataset adaptation and our complete model. Wild-NVS achieves state-of-the-art performance among the evaluated deterministic feed-forward methods across PSNR, SSIM, and LPIPS on both datasets. Qualitative results further demonstrate appearance transfer from different scenes to synthesize targets (Figure 1) and improved transient artifact mitigation while preserving static scene details.

2 Related Work

Novel view synthesis from in-the-wild images: Unconstrained photo collections violate the static, photometrically consistent scene assumption of standard NVS, requiring separate treatment of image-dependent appearance and transient occluders. NeRF-W (Martin-Brualla et al., 2021) introduced per-image appearance embeddings together with a transient head with learned uncertainty. Subsequent Gaussian splatting methods follow the same decomposition. WildGaussians (Kulhanek et al., 2024) maps per-image and per-Gaussian embeddings to an affine transformation of Gaussian colors and uses uncertainty estimation to down-weight transients. GS-W (Zhang et al., 2024) and Wild-GS (Xu et al., 2024) extract local appearance from a reference image, via adaptive feature sampling and tri-plane features respectively, and both predict 2D visibility maps to handle transients. All of these methods require per-scene optimization.

Feed-forward novel view synthesis: Feed-forward models learn reconstruction priors from large multi-view datasets and either predict explicit Gaussians (Jiang et al., 2025; Lin et al., 2025) or de-

code target views directly from latent representations (Jin et al., 2024; Szymanowicz et al., 2026), but do not model image-dependent appearance or transients. WildCAT3D (Alper et al., 2025) trains a multi-view diffusion model on in-the-wild collections by conditioning on low-dimensional appearance embeddings from a feed-forward encoder, generating novel views from a single input image with controllable global appearance. Among deterministic methods, GenWildSplat (Gupta et al., 2026) and WildSplatter (Fujimura et al., 2026) extend feed-forward Gaussian splatting with a compact per-image appearance code that modulates Gaussian colors. GenWildSplat trains on outdoor DL3DV (Ling et al., 2024) scenes augmented with diffusion-based relighting and pasted COCO object segments, and relies on an external segmentation network to exclude transients. WildSplatter trains on MegaScenes (Tung et al., 2024) with two context views per target and restricts the loss to depth-consistent target regions; transients in the context views are not modeled explicitly. In contrast to these Gaussian-based approaches, our method builds on a latent NVS model (Szymanowicz et al., 2026) that decodes target views directly from source tokens, which allows a learned transient head to suppress transient content at the token level before decoding. We train it on real appearance variation and transient occlusion using geometrically verified multi-view source sets.

Evaluation of in-the-wild NVS: No common benchmark exists for multi-view feed-forward NVS from in-the-wild images. GenWildSplat evaluates on three PhotoTourism and 20 MegaScenes scenes without identifying the scenes or the source and target images, and WildSplatter evaluates on six NeRF-OSR (Rudnev et al., 2022) scenes without listing the selected images. Their results can therefore neither be reproduced nor compared. We establish the first such benchmark, with scene-level splits and fixed source-view selection and evaluation protocols for $\{4, 6, 8, 10, 12\}$ view inputs.

3 BACKGROUND

Our goal is to design a model that takes in a set of unconstrained source images $I_1, \ldots, I_V \in \mathbb{R}^{3\times H\times W}$ of a given 3D scene, the parameters of the target camera $\boldsymbol{c}$, and an appearance conditioning image $\boldsymbol{A}$ to synthesize the novel target view of the 3D scene. This can be formally written as $I = f(\boldsymbol{c}, \boldsymbol{A}|I_1, \boldsymbol{c}_1, \ldots, I_V, \boldsymbol{c}_V)$, where the source camera views $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_V$ parameters are optionally provided. We aim to achieve this novel view synthesis capability using a feed-forward model using latent representations from an encoder with 3D inductive bias, bypassing the need for an explicit 3D scene reconstruction. In the following section we describe LagerNVS (Szymanowicz et al., 2026), the method that we extend in Wild-NVS to achieve this goal.

LagerNVS: LagerNVS is a transformer based encoder-decoder architecture, where the source images are encoded independently of the target image into a latent representation $\mathbf{F}$ which are then decoded into the target image by the decoder. The NVS function f can thus be represented as a composition of an encoder $\texttt{Enc}$: $\mathbf{F} = \texttt{Enc}(I_1, \boldsymbol{c}_1, \ldots, I_V, \boldsymbol{c}_V)$, and a decoder $\texttt{Dec}$: $I = \texttt{Dec}(\mathbf{F}; \boldsymbol{c})$, where the encoder is built on top of the VGGT model (Wang et al., 2025). While effective in controlled settings, LagerNVS fails under severe photometric variations and transient occlusions typical of in-the-wild photography. We extend this foundation to handle such unconstrained conditions.

4 WILD-NVS BENCHMARK

Training feed-forward novel-view synthesis models on unconstrained imagery requires source views that jointly observe the target while spanning diverse viewpoints. We introduce Wild-NVS, a dataset and benchmark built on MegaDepth-X (Li et al., 2026) reconstructions. Our view-set construction balances depth-based target coverage with viewpoint diversity, providing a large training corpus alongside fixed validation and test benchmarks on held-out landmarks.

4.1 SOURCE–TARGET VIEW-SET CONSTRUCTION

Our MegaDepth-X corpus contains 1,868 reconstructions of 1,633 landmarks and approximately 440,000 images. Within each reconstruction, we construct *view sets*, each consisting of a target image and an ordered list of up to 64 source images. We first identify sources that observe parts of the target, then select and order them to balance joint target coverage with viewpoint diversity.

Covisibility. We measure how much of a target view i is also observed in a source view j using a depth-warping test (Fujimura et al., 2026). For each pixel $p \in \mathcal{V}_i$—where $\mathcal{V}_i$ denotes the target

pixels with valid MVS depth—we backproject it to 3D and project it into source view j, obtaining an image location q_p and a depth $z_j(p)$ in the source camera frame. The source is considered to observe this point if $z_j(p)$ agrees with the source MVS depth map D_j at q_p, within a log-depth tolerance τ. The covisibility score C_{ij} is the fraction of the target's valid-depth pixels that pass this test:

$$C_{ij} \;=\; \frac{1}{|\mathcal{V}_i|} \sum_{p \in \mathcal{V}_i} \mathbf{1}\Big[\, \big|\log z_j(p) - \log D_j(q_p)\big| \le \tau \,\Big]. \tag{1}$$

The indicator is zero if the projection lies outside the source image or behind the source camera, or if the source depth is invalid. *Joint target coverage* counts each valid-depth target pixel that passes the test in at least one source. Both scores are normalized by the number of valid-depth target pixels.

Candidate filtering. A source j is a candidate for target i if $C_{ij} \ge 0.20$ and $C_{ji} \ge 0.15$; the reverse score helps reject severe scale mismatches. We also suppress near-duplicate views using camera baseline, viewing direction, and relative focal length. A target is retained for source selection if it has sufficient valid-depth pixels and at least three candidates after this filtering. At most 64 candidates are kept, ranked by forward covisibility.

Source ordering. We initialize the source list with a seed pair selected for its joint target coverage and viewpoint spread. Spread measures separation in camera position and viewing direction. We then add sources that extend coverage while favoring viewpoints separated from those already selected. Among candidates that add a minimum amount of new coverage, we choose the one with the greatest spread. If no candidate meets this, we choose the candidate that adds the most coverage.

Once joint coverage reaches 80%, we select by viewpoint spread alone until the list contains eight sources or the candidates are exhausted. Any remaining candidates are appended using a score that combines additional coverage and viewpoint spread. We accept a view set if its first up to eight sources jointly cover at least 80% of the target's valid-depth pixels and meet a minimum separation criterion between source views. The resulting dataset contains 170,559 view sets across 1,805 reconstructions. Exact objectives, thresholds, and seed-selection fallbacks are given in Appendix E.

Training source sampling. The stored list supports training with different source counts and combinations. We use 2–8 sources per training sample. When drawing a subset, we first choose one of the stored seed pairs at random; each pair covers at least 55% of the target's valid-depth pixels. We fill the remaining slots by weighted sampling from the rest, favoring sources that contributed coverage or viewpoint spread during construction. Finally, we shuffle the selected sources.

4.2 Splits and benchmark

We split the corpus by landmark, keeping all reconstructions of each landmark in the same split. Training, validation, and test contain 1,500, 58, and 75 landmarks, respectively, with $\sim$150K, $\sim$8K, and $\sim$8K view sets. These are the full view-set pools; the fixed validation and test benchmarks use subsets selected from the held-out pools. The PhotoTourism (Jin et al., 2021) evaluation landmarks and their immediate surroundings are manually assigned to the test split to keep them out of training.

To construct the benchmark, we group views by reconstruction community, obtained by Louvain community detection (Blondel et al., 2008) on the graph of views sharing reconstructed 3D points. Retaining every view set with at least twelve sources and sampling up to three per community gives 1048 test cases each with a distinct target. Each case is evaluated at every source count $n \in \{4, 6, 8, 10, 12\}$ by serving its first n sources, so the target is fixed and the source sets are nested. We randomize presentation order while fixing one anchor source across counts.

5 Feedforward Wild-NVS

We present Wild-NVS, a feedforward novel view synthesis model that extends LagerNVS (Szymanowicz et al., 2026) with appearance conditioning and transient feature suppression for in-the-wild photo collections. Figure 2 provides an overview of the architecture.

Wild-NVS consists of four primary modules. The LagerNVS encoder `Enc` processes the source images $\{I_1, \ldots, I_V\}$ to extract latent features $\mathbf{F} = \{\mathbf{F}_1, \ldots, \mathbf{F}_V\}$. An appearance encoder $\mathcal{A}_\theta$ generates the global appearance embedding $\boldsymbol{a}$ which enables the decoder to model image-dependent illumination changes (Section 5.1). A transient prediction head Φ predicts a transient probability map

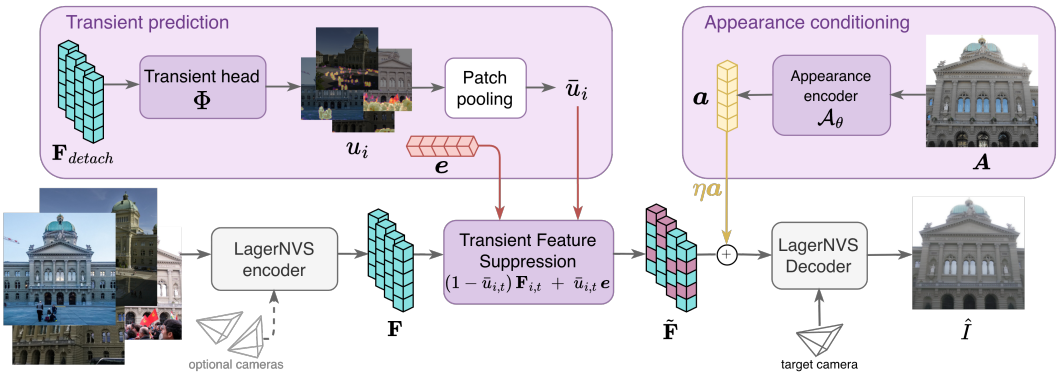

Figure 2: **Wild-NVS pipeline**. We adapt LagerNVS to model varying appearances in unconstrained images using the appearance encoder $\mathcal{A}_\theta$, and predict transients using the transient head Φ. A learnable transient embedding $\boldsymbol{e}$ facilitates transient feature suppression in the encoder features $\mathbf{F}$. The transient suppressed features are conditioned with appearance code $\boldsymbol{a}$ and passed to the decoder.

for each source view i from its latent features $\mathbf{F}_i$ which guide feature suppression before decoding (Section 5.2). Finally, the decoder `Dec` synthesizes the target view using the transient-suppressed, appearance-conditioned features and the target camera parameters.

5.1 Appearance Modeling

Existing methods such as WildGaussians (Kulhanek et al., 2024) and NeRF-W (Martin-Brualla et al., 2021) model appearance by learning a per-image appearance embedding. In the per-scene optimization domain this is feasible, since a new embedding for the target view can be optimized at test time prior to inference. We instead train a bottlenecked appearance encoder to predict an appearance code from a conditioning image $\boldsymbol{A}$, similar to recent feedforward approaches (Alper et al., 2025).

Our appearance encoder $\mathcal{A}_\theta$ is a lightweight convolutional neural network (CNN) that acts as an information bottleneck. The convolutional features are globally average-pooled into an 8-dimensional code, which a two-layer MLP expands to the final appearance embedding $\boldsymbol{a} \in \mathbb{R}^d$, matching the decoder embedding dimension d. The appearance condition is directly injected into the source features, preserving the internal decoder architecture. The embedding $\boldsymbol{a}$ is broadcast across all source views and added to every source token $\mathbf{F}_i$ via a learned scalar gate η:

$$\hat{\mathbf{F}}_i \;=\; \mathbf{F}_i + \eta\,\boldsymbol{a}, \qquad i = 1, \ldots, V. \tag{2}$$

By substituting $\hat{\mathbf{F}}$ for $\mathbf{F}$ in all subsequent operations, appearance information naturally propagates to the target tokens through the decoder layers. We initialize η to zero to prevent representational drift at the beginning, allowing the model to progressively learn the optimal injection strength.

5.2 Transient Modeling

To model the transient components, we train a transient prediction head Φ to output a per-pixel transient probability $u_i(p) \in [0, 1]$ for each source view. We leverage the multiview-aggregated features $\mathbf{F}_i$ from the encoder, corresponding to view i to predict the transient probability map:

$$u_i \;=\; \Phi\big(\mathrm{sg}[\mathbf{F}_i]\big) \in [0,1]^{h \times w}, \tag{3}$$

where $\mathrm{sg}[\cdot]$ is the stop-gradient operator, and $h \times w$ is the label resolution.

Transient labels: To supervise the transient prediction head, we leverage an instance segmentation network (Jocher et al., 2023) followed by a post-processing step that filters out false positives on static objects using the SfM reconstruction and MVS depth available in the Wild-NVS dataset. Every pixel of every image is assigned one of three labels: *transient*, *ignore*, or *static*.

We run the pre-trained segmentation network on every image and extract instances of the movable classes from the COCO class list as transient candidates. For a candidate m with instance mask $\mathcal{M}_m$, we measure its persistence in the scene using two statistics: the density ρ_m of observed SfM

keypoints, relative to the density over the whole image; and the median track length ℓ_m of those points, compared with the scene-wide median $\bar{\ell}$. A transient candidate is deemed persistent if:

$$|\mathcal{M}_m| \geq A_{\min} \quad \text{and} \quad \big(\rho_m \geq \rho_{\min} \ \text{ or } \ \ell_m - \bar{\ell} \geq \Delta\ell\big), \tag{4}$$

labeling the pixels *ignore*. This effectively filters out false positive detections on persistent objects like statues or parked vehicles. The remaining candidate pixels are labeled as *transient* ($\mathcal{M}_{\text{trans}}$).

Static labels are derived from a combination of geometric and semantic evidence. Let $\mathcal{P}$ denote the pixels of observed SfM points with a triangulated 3D point, dilated by a small radius; $\mathcal{V}$ the pixels with valid MVS depth; $\mathcal{S}$ the pixels classified as sky by the sky head of Depth Anything 3 (Lin et al., 2025); and $\mathcal{T}$ the union of all movable class instances. The mask of statically labeled pixels is then defined as $\mathcal{M}_{\text{static}} = \big(\mathcal{P} \cup \mathcal{V} \cup \mathcal{S}\big) \setminus \mathcal{T}$. We provide further details in Appendix D.

We train the transient head with binary cross-entropy averaged over pixels in $\mathcal{M}_{\text{trans}} \cup \mathcal{M}_{\text{static}}$, with labels 1 and 0, respectively. Ignored pixels do not contribute to the loss.

Transient feature suppression: To prevent transient objects in the source images from corrupting the target view, we leverage the predicted transient maps u_i to suppress dynamic objects in the source features before they reach the decoder. We introduce a learnable transient token $\boldsymbol{e} \in \mathbb{R}^d$ and interpolate it with the original source tokens based on the transient probability:

$$\tilde{\mathbf{F}}_{i,t} \ = \ (1 - \bar{u}_{i,t})\, \mathbf{F}_{i,t} \ + \ \bar{u}_{i,t}\, \boldsymbol{e}, \tag{5}$$

where $\bar{u}_{i,t}$ is the patch-level transient probability for token t, obtained by pooling the high-resolution transient map u_i. The appearance embedding from Eq. 2 is subsequently added to these suppressed tokens $\tilde{\mathbf{F}}_i$. Although the transient token $\boldsymbol{e}$ receives no explicit supervision, interpolating it directly via the supervised transient probabilities forces the model to implicitly learn a canonical representation for transient features, effectively mitigating the rendering process from dynamic occluders.

5.3 Model Training

Wild-NVS is trained using a photometric loss that combines a pixel-wise L2 loss and a perceptual loss, following LagerNVS (Szymanowicz et al., 2026). To prevent the model from penalizing itself for failing to reconstruct unpredictable transient objects, the photometric loss is applied exclusively to the static content pixels in the target view using a binary supervision mask $\mathcal{K}$:

$$\mathcal{L}_{\text{photo}} \ = \ \frac{1}{|\mathcal{K}|} \sum_{p \in \mathcal{K}} \big\| \hat{I}(p) - I(p) \big\|_2^2 \ + \ \lambda_{\text{p}}\, \mathcal{L}_{\text{perc}}\big(\hat{I} \odot \mathcal{K},\, I \odot \mathcal{K}\big). \tag{6}$$

The mask $\mathcal{K}$ is obtained by binarizing the predicted transient map of the target view using a threshold δ. Crucially, to prevent data leakage, these target-side transient probabilities are predicted via a separate forward pass where the target view is appended to the source view list without gradients. This ensures that the source tokens used for rendering never attend to the target image.

Training schedule: The network components are introduced in three distinct phases within a single continuous training run. In the first phase, the VGGT encoder, the decoder, and the appearance encoder are trained on the unmasked photometric loss alongside low-rank (LoRA) finetuning of the VGGT encoder. In the second phase, the transient prediction head is enabled and trained on features detached from the encoder. In the third and final phase, the feature suppression and target-side masking are activated, and all components are optimized jointly.

6 Experiments

Here we provide a detailed evaluation of our method Wild-NVS towards novel view synthesis from in-the-wild images. We present the datasets used for training, and evaluation, our baselines for comparisons, and specify implementation details of our method.

Datasets: Our method is trained using the train set of the proposed Wild-NVS dataset, consisting of 1500 landmarks from MegaDepth-X. This dataset provides real-world in-the-wild image collections that span varied camera settings, diverse transient occluders, and changing illumination. We evaluate our method on two datasets, the proposed Wild-NVS dataset, and the PhotoTourism dataset. We

Table 1: Quantitative results on the Wild-NVS dataset across four, six and eight source view inputs.

Method	4 Views			6 Views			8 Views		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
LagerNVS (CVPR′26)	12.99	0.448	0.654	13.18	0.451	0.641	13.31	0.452	0.633
LagerNVS-FT	14.61	0.490	0.517	14.85	0.499	0.497	15.03	0.505	0.484
GenWildSplat (CVPR′26)	12.59	0.377	0.630	13.14	0.387	0.621	13.51	0.396	0.616
WildSplatter	12.95	0.366	0.607	13.29	0.381	0.600	13.48	0.390	0.597
Wild-NVS	**17.37**	**0.523**	**0.463**	**17.65**	**0.532**	**0.445**	**17.81**	**0.537**	**0.434**

evaluate on the *Brandenburg Gate*, *Sacre Coeur* and the *Trevi Fountain scenes* for the PhotoTourism dataset using six source views as in GenWildSplat. We follow the steps detailed in Section 4 to choose the source views across comparisons of all methods.

Baselines: We compare against LagerNVS (Szymanowicz et al., 2026), a latent representation based feed-forward novel view synthesis method. We evaluate both its original pretrained model, denoted as *LagerNVS*, and a version finetuned on our Wild-NVS dataset, denoted as *LagerNVS-FT*. This comparison assesses the benefit of adapting the baseline to the proposed dataset. We additionally compare against GenWildSplat (Gupta et al., 2026) and WildSplatter (Fujimura et al., 2026), two feed-forward methods that explicitly reconstruct scenes using Gaussian splatting (Kerbl et al., 2023).

Implementation Details: We implement Wild-NVS in PyTorch, building on LagerNVS. We apply LoRA finetuning to the VGGT-based encoder and finetune the camera encoding MLP and the decoder, which consists of 12 transformer blocks (Vaswani et al., 2017). Training proceeds in three stages over 125K steps. During the first 45K steps, we jointly optimize these components and the appearance encoder using the unmasked photometric loss. From step 45K onwards, we additionally train the transient prediction head on detached encoder features. At step 60K, we activate transient feature suppression and photometric masking, continuing training until step 125K.

6.1 Results

Evaluation protocol: We report PSNR, SSIM, and LPIPS (Zhang et al., 2018) after center-cropping target images to squares and resizing them to each method's native rendering resolution: 448×448 for GenWildSplat, 504×504 for WildSplatter, and 512×512 for Wild-NVS. We follow GenWildSplat's original evaluation setup (Gupta et al., 2026), using model-predicted target poses and YOLO masks for source-image transient suppression. GenWildSplat does not accept source camera parameters, whereas WildSplatter and Wild-NVS use the provided parameters.

Quantitative evaluations: Table 1 reports results with 4-, 6-, and 8-views, with additional results in Appendix B.1. Finetuning LagerNVS on our dataset improves PSNR by 1.62/1.67/1.72 dB, respectively, surpassing GenWildSplat and WildSplatter across all metrics. Wild-NVS achieves further PSNR gains of 2.76/2.8/2.78 dB over LagerNVS-FT, alongside improvements in SSIM and LPIPS, demonstrating the benefits of our combined appearance and transient modeling. With more views, Gaussian splatting based methods surpass LagerNVS, with GenWildSplat surpassing WildSplatter at 8 views. We hypothesize this is due to the two-view training regime of WildSplatter. Wild-NVS achieves the best results; even with four views, it outperforms every baseline using more views.

Table 2 shows that Wild-NVS also achieves the best PSNR, SSIM, and LPIPS across all three PhotoTourism scenes. It improves PSNR over the strongest baseline for each scene by 5.16 dB on Brandenburg Gate (against LagerNVS-FT), 3.78 dB on Sacre Coeur, and 2.59 dB on Trevi Fountain (against WildSplatter), demonstrating that its gains extend beyond the Wild-NVS benchmark.

Table 2: Quantitative results on the PhotoTourism dataset.

Method	Brandenburg Gate			Sacre Coeur			Trevi Fountain		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
LagerNVS (CVPR′26)	12.29	0.599	0.575	13.24	0.476	0.545	13.23	0.341	0.626
LagerNVS-FT	16.39	0.702	0.389	15.50	0.615	0.350	16.81	0.527	0.379
GenWildSplat (CVPR′26)	14.56	0.566	0.550	15.71	0.497	0.497	15.70	0.376	0.521
WildSplatter	12.64	0.529	0.522	15.87	0.487	0.425	16.94	0.388	0.407
Wild-NVS	**21.55**	**0.743**	**0.310**	**19.65**	**0.644**	**0.307**	**19.53**	**0.541**	**0.360**



Figure 3: Qualitative results on the Wild-NVS dataset. Wild-NVS renders views preserving geometric structure with fine edges (row 1, row 3 dome, row 4 railing), models varying appearances (row 2), and removes transient objects (row 3 banners) while preserving static objects (row 4 sculpture).

Qualitative evaluations: Figure 3 compares Wild-NVS with existing methods for novel view synthesis from in-the-wild images. The first row illustrates the difficulty of reconstructing scenes with dense transient occlusions. In GenWildSplat, regions corresponding to YOLO-masked tourists bleed into the rendered view as dark blobs. WildSplatter produces thin, elongated artifacts across the courtyard and residual blobs near the base of the Brandenburg Gate. Although Wild-NVS does not fully recover the courtyard details, it reduces these artifacts and better preserves the sharp boundaries of the gate's columns. The blue boxes in the third and fourth scene examples further highlight its preservation of fine architectural structure, including the dome details and the railing.

The red box in the third scene example highlights differences in transient suppression. Wild-NVS removes the banners while retaining the grooves around the arch. WildSplatter partially suppresses the banners but also loses these architectural details, while GenWildSplat retains visible banner content and introduces surrounding distortions. In the final example, both baselines largely omit the sculpture highlighted in red, whereas Wild-NVS partially reconstructs its form. These examples



Figure 4: Interpolations of the appearance embedding $\boldsymbol{a}$ of two target images (far left, far right) used to render the target image at the interpolated camera pose using Wild-NVS.

illustrate the importance of suppressing transient content while retaining static objects and structural details. Additional results, together with their source views, are provided in the appendix.

Wild-NVS also better captures the warm appearance of the target in the second row of Figure 3, although it does not reproduce the detailed cloud structure or localized illumination exactly. Like prior methods, Wild-NVS uses an appearance conditioning image to control the rendered appearance. During training, the target image supplies this conditioning through a compact latent code, with a strong bottleneck designed to limit structural information leakage. Figure 1 demonstrates appearance transfer using conditioning images from multiple scenes. The renderings exhibit changes in color tone, brightness, and sky appearance while retaining the structure of the source scene. This provides qualitative evidence that the conditioning code can transfer appearance without requiring the conditioning image to depict the same scene. Finally, Figure 4 shows renderings obtained by jointly interpolating camera poses and appearance codes between two views of the same scene. The sequences transition between the endpoint appearances, including dusk-to-night conditions, while maintaining recognizable architectural structure and suppressing foreground pedestrians.

Ablations: Table 3 shows that appearance conditioning accounts for most of the quantitative gains, improving PSNR by 2.58 dB over LagerNVS-FT. Transient suppression further improves LPIPS and PSNR a little but slightly reduces SSIM, despite visibly mitigating transient artifacts (Figure 5). We hypothesize that reference-based metrics understate this benefit because target photographs themselves contain transients. Removing transients from commonly occurring places can reduce agreement with the reference. To assess transient artifacts directly, we use GPT-6 Astra to compare paired renderings from rows 1 and 2 on test examples. Neither rendering is flagged in 89.6% of pairs, while 8.8% contain flagged transient artifacts only in the appearance-only rendering, quantifying the evidence of improved transient suppression. Additional details in Appendix B.2.

Table 3: Component ablation at 100K steps.

Row	Method	PSNR ↑	SSIM ↑	LPIPS ↓
0	LagerNVS-FT	14.98	0.503	0.488
1	+ Appearance encoder	17.48	0.527	0.458
2	+ Transient suppression	17.49	0.518	0.450

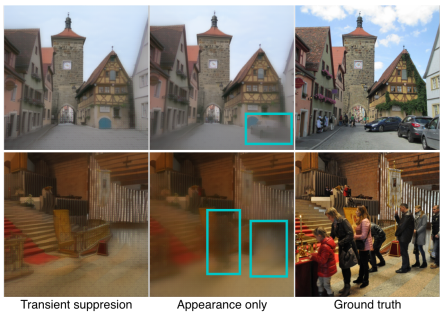

Figure 5: Transient mitigation examples.

7 Conclusion

We presented Wild-NVS, a feed-forward NVS model from latent representations, for in-the-wild images built on LagerNVS. A bottlenecked appearance encoder conditions the decoder on a compact code, and predicted transient probabilities modulate the source tokens before decoding to suppress transient artifacts. We also introduced the Wild-NVS dataset and benchmark, with geometrically verified source–target sets and fixed evaluation protocols. Evaluations on our benchmark and PhotoTourism, demonstrate qualitative and quantitative improvements compared to previous methods.

Limitations. As a deterministic model, Wild-NVS produces blurry renderings in regions unobserved in the source views, and the global appearance code cannot represent spatially varying lighting. Incorporating generative priors and spatially varying appearance codes are potential solutions.

AI USE STATEMENT

In this work, we used generative AI tools to design experiments, provide feedback on the research methodology, implement methods, reformat datasets, and support qualitative data analysis. We have not used generative AI tools to clean datasets, propose or refine hypotheses, or interpret results, and generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating or proving mathematical claims, and translation are not applicable to this work. Additionally, we used generative AI tools to create and edit code, create figures, and edit the paper for readability. We have reviewed all AI-assisted work: AI-generated code was reviewed and tested, and the manuscript was proofread by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Morris Alper, David Novotny, Filippos Kokkinos, Hadar Averbuch-Elor, and Tom Monnier. Wildcat3d: Appearance-aware multi-view diffusion in the wild, 2025. URL https://arxiv.org/abs/2506.13030.

Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008 (10):P10008, 2008.

Bardienus Pieter Duisterhof, Lojze Zust, Philippe Weinzaepfel, Vincent Leroy, Yohann Cabon, and Jerome Revaud. Mast3r-sfm: a fully-integrated solution for unconstrained structure-from-motion. In *2025 International Conference on 3D Vision (3DV)*, pp. 1–10. IEEE, 2025.

Yuki Fujimura, Takahiro Kushida, Kazuya Kitano, Takuya Funatomi, and Yasuhiro Mukaigawa. Wildsplatter: Feed-forward 3d gaussian splatting with appearance control from unconstrained images. *arXiv preprint arXiv:2604.21182*, 2026.

Vinayak Gupta, Chih-Hao Lin, Shenlong Wang, Anand Bhattad, and Jia-Bin Huang. Generalizable sparse-view 3d reconstruction from unconstrained images. *arXiv preprint arXiv:2604.28193*, 2026.

Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Transactions on Graphics (TOG)*, 44(6):1–16, 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. *arXiv preprint arXiv:2410.17242*, 2024.

Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *International Journal of Computer Vision*, 129(2):517–547, 2021.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolo. *Zenodo*, 2023.

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

Jonas Kulhanek, Songyou Peng, Zuzana Kukelova, Marc Pollefeys, and Torsten Sattler. Wildgaussians: 3d gaussian splatting in the wild. *arXiv preprint arXiv:2407.08447*, 2024.

Yuan Li, Yuanbo Xiangli, Hadar Averbuch-Elor, Noah Snavely, and Ruojin Cai. Long-tail internet photo reconstruction. *arXiv preprint arXiv:2604.22714*, 2026.

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

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

Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the wild: Neural radiance fields for unconstrained photo collections. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Viktor Rudnev, Mohamed Elgharib, William Smith, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. Nerf for outdoor scene relighting. In *European Conference on Computer Vision*, pp. 615–631. Springer, 2022.

Stanislaw Szymanowicz, Minghao Chen, Jianyuan Wang, Christian Rupprecht, and Andrea Vedaldi. Lagernvs: Latent geometry for fully neural real-time novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15443–15453, 2026.

Joseph Tung, Gene Chou, Ruojin Cai, Guandao Yang, Kai Zhang, Gordon Wetzstein, Bharath Hariharan, and Noah Snavely. Megascenes: Scene-level view synthesis at scale. In *European conference on computer vision*, pp. 197–214. Springer, 2024.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

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)*, 2025. arXiv:2503.11651.

Yuanbo Xiangli, Ruojin Cai, Hanyu Chen, Jeffrey Byrne, and Noah Snavely. Doppelgangers++: Improved visual disambiguation with geometric 3d features. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27166–27175. IEEE, 2025.

Jiacong Xu, Yiqun Mei, and Vishal M Patel. Wild-gs: Real-time novel view synthesis from unconstrained photo collections. *Advances in Neural Information Processing Systems*, 37:103334–103355, 2024.

Dongbin Zhang, Chuming Wang, Weitao Wang, Peihao Li, Minghan Qin, and Haoqian Wang. Gaussian in the wild: 3d gaussian splatting for unconstrained image collections. In *European Conference on Computer Vision*, pp. 341–359. Springer, 2024.

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

APPENDIX

A TRAINING DETAILS

Wild-NVS is initialized from the released LagerNVS General checkpoint and trained for 125k steps. The pretrained VGGT weights remain frozen, with LoRA adapters of rank 16 and scaling parameter $\alpha = 32$ applied to attention query/key/value and output projections and both MLP layers in every frame and global attention block. The decoder, encoder projection layers and camera MLP, appearance encoder, transient head, and transient token are trained fully. Table 4 lists the hyperparameters.

Table 4: Model training hyperparameters. The head and transient token begin their warm-ups in phases 2 and 3, respectively.

Setting	Value
Optimizer	AdamW, $\beta = (0.9, 0.95)$
Weight decay	0.05; zero for LoRA
Gradient-norm clipping	3.0
Learning rate, decoder and appearance encoder	10^{-5}
Learning rate, LoRA	10^{-4}
Learning rate, transient head and transient token	10^{-4}
Learning-rate schedule	3k-step warm-up, cosine decay
Head/token warm-up at activation	500 steps
Source views	2–8
Sampling weights for [2, 3, 4, 5, 6, 7, 8] sources	$16 : 4 : 16 : 4 : 8 : 2 : 8$
Target views per sample	2
Training image resolution	512×512
Source-camera conditioning withheld	Probability 0.2
Appearance embedding dropout	Probability 0.3
L2 / perceptual / transient loss weights	$1.0 \,/\, 0.5 \,/\, 1.0$
Perceptual network	VGG-19
Transient threshold δ for target masking	0.5
Target-mask dilation radius	2 pixels at 256×256
Minimum supervised fraction per target	5%

Each sample supervises the view-set target and one randomly selected source view. Sources are sampled without replacement around a stored seed pair as described in Section 4. If a requested source count exceeds the available candidates, all candidates are included and the remaining slots repeat uniformly sampled candidates. The resulting context is shuffled.

The transient head's last convolution is initialized with weights drawn from $\mathcal{N}(0, 0.01^2)$ and bias $\mathrm{logit}(0.027)$, so its initial predictions approximate the transient fraction among labeled pixels. From phase 2 onward, the objective is $\mathcal{L}_{\mathrm{photo}} + \lambda_{\mathrm{t}}\mathcal{L}_{\mathrm{t}}$, with $\lambda_{\mathrm{t}} = 1$. Target masking is activated only in phase 3.

For the photometric loss in equation 6, pixels with target transient probability at least δ are dilated and excluded. The resulting keep mask $\mathcal{K}$ is computed at 256×256 and upsampled by nearest-neighbor interpolation to the rendering resolution. A sample is excluded from the photometric loss if any target retains less than 5% of its pixels. The separate, no-gradient target prediction pass follows Section 5.3.

B ADDITIONAL RESULTS

B.1 PERFORMANCE ACROSS SOURCE-VIEW COUNTS

The fixed-target protocol in Appendix E.4 supports comparisons across 4, 6, 8, 10, and 12 source views on the same 1,048 test targets. The 10- and 12-view settings also evaluate source counts beyond the model's training range of 2–8 views. All methods show an increase in performance with increasing number of views as reported in Table 5. However, since the union-coverage of the target

Table 5: Quantitative results on the Wild-NVS dataset across varying source view counts.

Method	Metric	4 Views	6 Views	8 Views	10 Views	12 Views
LagerNVS (CVPR′26)	PSNR ↑	12.99	13.18	13.31	13.35	13.42
	SSIM ↑	0.448	0.451	0.452	0.453	0.454
	LPIPS ↓	0.654	0.641	0.633	0.628	0.623
LagerNVS-FT	PSNR ↑	14.61	14.85	15.03	15.17	15.28
	SSIM ↑	0.490	0.499	0.505	0.510	0.513
	LPIPS ↓	0.517	0.497	0.484	0.475	0.468
GenWildSplat (CVPR′25)	PSNR ↑	12.59	13.14	13.51	13.71	13.81
	SSIM ↑	0.377	0.387	0.396	0.402	0.406
	LPIPS ↓	0.630	0.621	0.616	0.615	0.614
WildSplatter	PSNR ↑	12.95	13.29	13.48	13.55	13.57
	SSIM ↑	0.366	0.381	0.390	0.393	0.396
	LPIPS ↓	0.607	0.600	0.597	0.596	0.595
Wild-NVS	PSNR ↑	**17.37**	**17.65**	**17.81**	**17.91**	**18.01**
	SSIM ↑	**0.523**	**0.532**	**0.537**	**0.541**	**0.543**
	LPIPS ↓	**0.463**	**0.445**	**0.434**	**0.426**	**0.420**

Table 6: Component ablation across varying source view counts. Reported at 100K steps due to compute resource constraints.

Row	Method	Metric	Avg.	4 Views	6 Views	8 Views	10 Views	12 Views
0	LagerNVS-FT	PSNR ↑	14.98	14.61	14.85	15.03	15.17	15.28
		SSIM ↑	0.503	0.490	0.499	0.505	0.510	0.513
		LPIPS ↓	0.488	0.517	0.497	0.484	0.475	0.468
1	+ Appearance encoder	PSNR ↑	17.48	**17.15**	17.40	**17.55**	17.63	17.71
		SSIM ↑	**0.527**	**0.518**	**0.525**	**0.529**	**0.531**	**0.533**
		LPIPS ↓	0.459	0.482	0.465	0.455	0.448	0.443
2	+ Transient suppression	PSNR ↑	**17.49**	**17.15**	**17.41**	**17.55**	**17.64**	**17.72**
		SSIM ↑	0.518	0.509	0.516	0.520	0.522	0.525
		LPIPS ↓	**0.450**	**0.474**	**0.457**	**0.447**	**0.440**	**0.434**

gets saturated, the performance also saturates. Figure 6 shows PSNR for all methods as the number of source views grows.

We also report the ablation on the components across different number of source views in Table 6 at 100K steps due to computational resource constraints. We notice that the benefit of the transient suppression mechanism (row 2) as compared to the appearance-only model (row 1) has the same trend throughout: a small gain in PSNR and LPIPS, and a minor drop in SSIM.

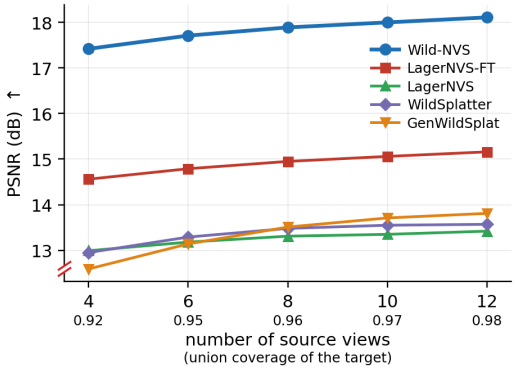

Figure 6: PSNR on the fixed-target Wild-NVS test benchmark as a function of the number of source views. The value below each view count is the mean joint coverage of the target by the served sources (Table 14).

B.2 TRANSIENT SUPPRESSION

To quantify the transient mitigation that is not reflected by the evaluated metrics, we perform a visual assessment as reported in the main paper. We pair the full-model and appearance-only predictions for the same target and source views. The reviewer receives only the predictions, with the full model presented as A and the appearance-only ablation as B. We query GPT-6 Astra with the following prompt:

```
Compare these two renders of the same viewpoint.  Treat the
first image as A and the second image as B. Look for any
transient objects rendered into the image in a ghost-like
fashion, such as translucent silhouettes, partial bodies of
people, or vehicle fragments.  Look for visible ghost-like
remnants, not ordinary blur, lighting/color differences,
shadows, reflections, intact solid objects, or a body merely
cropped by the image boundary.  Do not assume either image
is better.  Respond with '0' if neither has ghost-like
content, '1' if A has ghost-like content but not B, '2'
if B has ghost-like content but not A, and '3' if both have
ghost-like content.  Respond with one digit only.
```

Each reviewed pair receives one label: neither rendering contains transient artifact, the full model only, the appearance-only model only, or both. Outcome counts are recorded separately for each evaluation set, with the number of reviewed pairs as the denominator. We notice that nearly 8.8% of the pairs show a transient artifact when the transient suppression mechanism is removed as reported in Table 7.

We present examples of such transient artifacts in Figure 7 with the source views for reference and highlighting transient components that seep into the final rendering using a dashed pink oval. We also visualize additional examples of the renderings in Figure 8 for better viewing quality.

Table 7: Transient labels from GPT-6 Astra

Label	Percentage
0 (Artifacts in neither)	89.61%
1 (Artifacts only in full-model)	0.18%
2 (Artifacts only in appearance-only model)	8.79%
3 (Artifacts in both models)	1.41%

C APPEARANCE ENCODER ARCHITECTURE

Table 8 details the encoder $\mathcal{A}_\theta$ in Section 5.1. Each convolutional block contains a 3×3 convolution with stride 1, padding 1, and no bias, followed by BatchNorm, ReLU, and 2×2 max pooling with stride 2. Global average pooling reduces the final feature map to an eight-dimensional code; the MLP maps this code to the decoder dimension $d = 768$. Feature injection follows equation 2.

D TRANSIENT SUPERVISION AND PREDICTION

D.1 LABEL CONSTRUCTION SETTINGS

We use YOLOv8x-seg (Jocher et al., 2023) at an inference size of 1,024 pixels, with non-maximum-suppression IoU threshold 0.7. Transient candidate classes are *person, bicycle, car, motorcycle, bus, train, truck, backpack, umbrella, handbag, and suitcase*. Table 9 gives the thresholds used with the labeling rules in Section 5.2.

Persistence. Relative density ρ_m is the number of observed SfM points per pixel inside the eroded instance mask divided by the corresponding density over the full image. Only observations associated with a triangulated 3D point are counted. Track length is the number of images observing a 3D point; ℓ_m is the median over points observed within the eroded mask, and $\bar{\ell}$ is the median over all 3D points in the reconstruction. Candidates with no such observations remain transient.



Figure 7: Examples of transient suppression ablation including source views. Each pair of rows show the ground truth view, the render from the full-model, and the render from the appearance-only model followed by the source views in the second row.

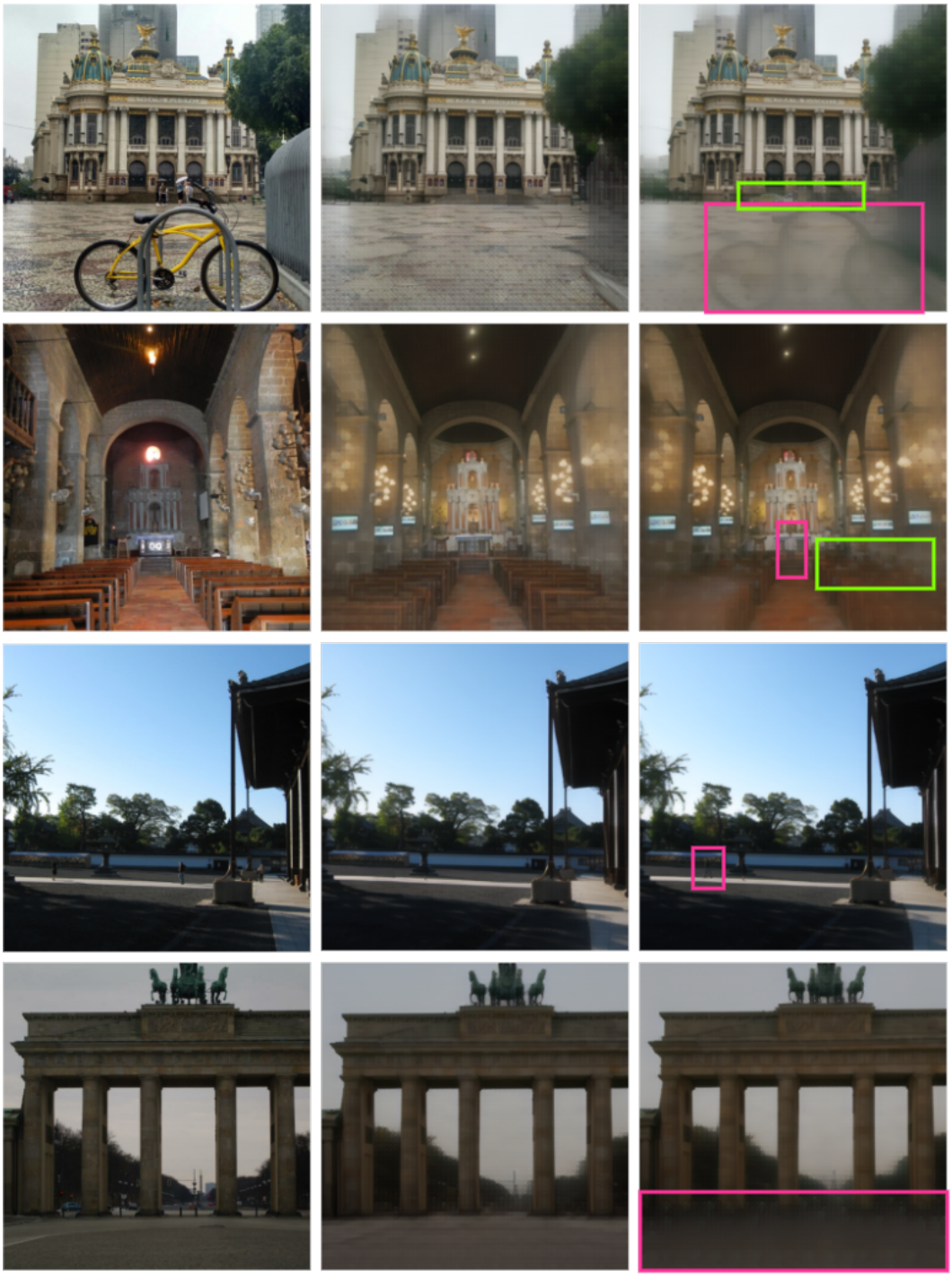

Figure 8: Examples of rendered views from the full-model and the appearance-only model. The appearance-only model renders transient objects such as vehicles, bikes, people, and severely blurs the courtyard due to the presence of transient people often covering this region.The highlighted regions denote transient artifacts in the rendering of the appearance-only model.

Table 8: Appearance encoder for a 512×512 conditioning image. Global average pooling also permits other input resolutions without changing the MLP.

Operation	Output channels	Spatial resolution
Conditioning image	3	512×512
Convolutional block 1	64	256×256
Convolutional block 2	64	128×128
Convolutional block 3	128	64×64
Convolutional block 4	64	32×32
Convolutional block 5	16	16×16
Convolutional block 6	8	8×8
Convolutional block 7	8	4×4
Global average pooling	8	1×1
Linear $8 \to 768$; ReLU	768	—
Linear $768 \to 768$; LayerNorm	768	—

The instance mask is eroded by four pixels to reduce contamination from background points along its boundary; erosion is reduced if it would empty the mask. The area gate $A_{\min}$ applies to the instance mask before erosion. Persistent-instance ignore labels take precedence where they overlap transient candidates.

Sky and static-label boundaries. Sky masks come from the sky head of Depth Anything 3's metric model, DA3METRIC-LARGE (Lin et al., 2025), with probability threshold 0.5. Inference uses the full image resized to a 504-pixel long side, with dimensions rounded to multiples of 14; predictions are mapped back to the image. For the static-label rule in the main paper, the SfM observations in $\mathcal{P}$ are dilated by five pixels, and the exclusion mask $\mathcal{T}$ contains all movable-class detections with confidence at least 0.10, dilated by eight pixels. These conservative exclusions prevent uncertain instance boundaries from being labeled static.

Label statistics. Across all 441,119 images, averaged over the corpus, 1.66% of pixels are labeled transient, 58.86% static, and 0.77% ignore, the latter being persistent candidates. The remaining $\sim 38.7\%$ also take the ignore label, which is excluded from the transient-head loss.

D.2 Transient prediction head

Table 10 specifies the head Φ of equation 3. Its input is the detached, multiview-aggregated VGGT patch feature map. The encoder processes a 518×518 image with 14×14 patches, producing a 37×37 grid. The head outputs transient probabilities at 256×256, matching the supervision labels.

Pooling to source tokens. To obtain $\bar{u}_{i,t}$ in equation 5, we map pixels of the probability map back to their corresponding encoder-patch footprints and average within each footprint. The mapping accounts for the three-pixel border trim of the unpadded convolutions and the factor-two downsampling. Applying this correspondence on both axes gives the pixel set averaged for each token.

Table 9: Label-construction settings. Pixel areas and radii refer to the original image resolution, before the training crop and resize. The notation $A_{\min}$, $\rho_{\min}$, and $\Delta\ell$ follows equation 4.

Setting	Value
Transient candidate confidence	≥ 0.25
Instance area gate $A_{\min}$, before erosion	300 pixels
Relative SfM density threshold $\rho_{\min}$	0.5
Track-length offset $\Delta\ell$	2 observations
Persistence-mask erosion radius	4 pixels
Sky probability threshold	0.5
SfM-point dilation radius for static labels	5 pixels

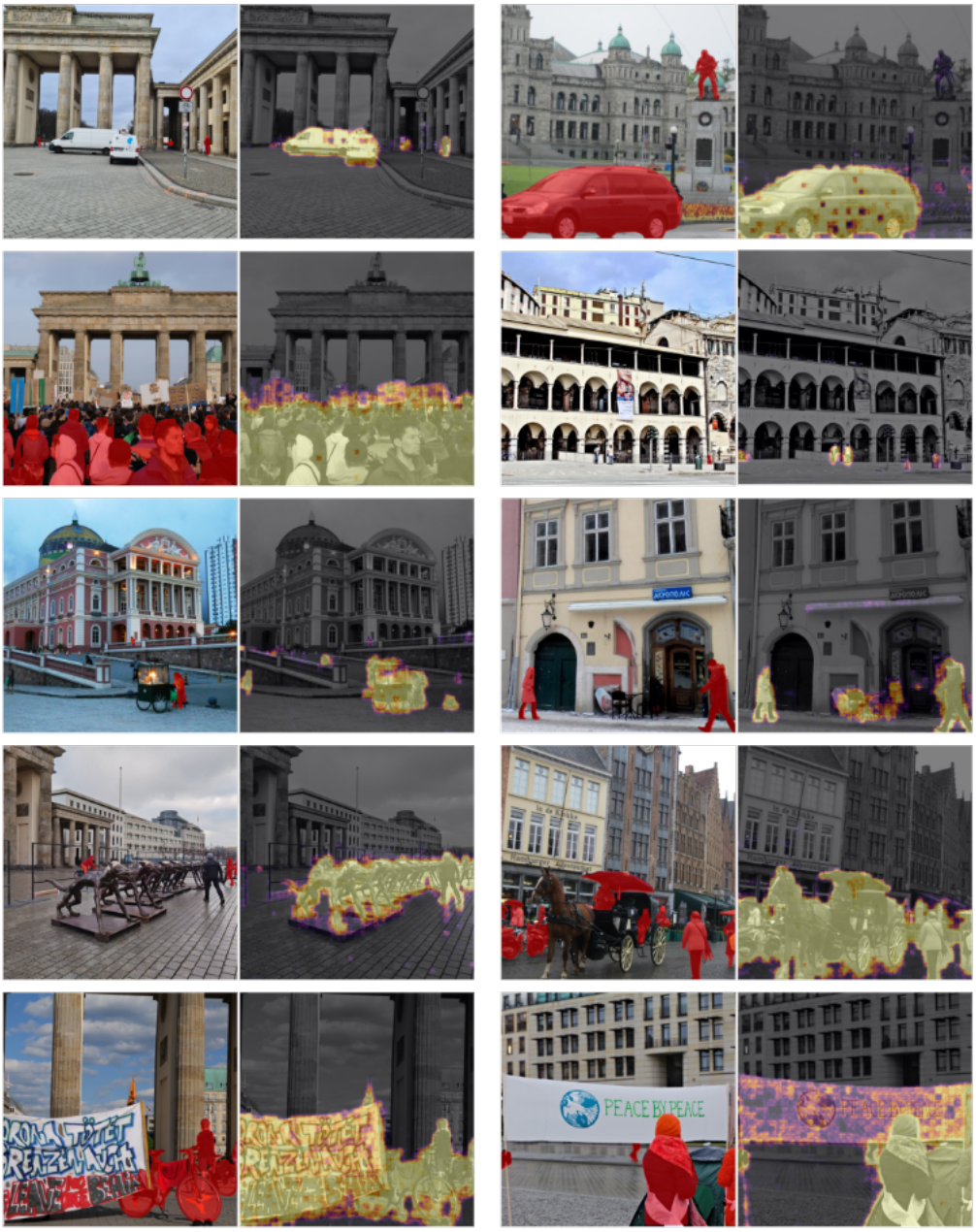

Figure 9: Comparison of the transient labels available with the output of the transient prediction head. The red masks on the left image of each pair denote the transient labels. The learned head better captures transients, overcoming missing or erroneous labels.

Supervision. For an example with V source views, the transient head is trained with the binary cross-entropy

$$\mathcal{L}_{\mathrm{t}} \;=\; -\frac{1}{|\Omega|}\sum_{i=1}^{V}\left(\sum_{p\in\mathcal{M}_{\mathrm{trans},i}}\log u_i(p) \;+\; \sum_{p\in\mathcal{M}_{\mathrm{static},i}}\log\big(1-u_i(p)\big)\right), \tag{7}$$

where $\Omega=\bigcup_{i=1}^{V}\left(\mathcal{M}_{\mathrm{trans},i}\cup\mathcal{M}_{\mathrm{static},i}\right)$ contains the labeled pixels of all source views. No gradient from this loss enters the encoder because the input features are detached. Optimization settings are given in Appendix A.

Table 10: Transient prediction head. Linear projection is applied independently to each patch token. The loss uses the pooled logits directly; the sigmoid produces probabilities for feature suppression.

Operation	Output channels	Spatial resolution
VGGT patch features	2,048	37×37
LayerNorm; linear projection	784	37×37
Pixel shuffle ($\times 14$); ReLU	4	518×518
3×3 convolution, no padding; ReLU	8	516×516
5×5 convolution, no padding	1	512×512
2×2 max pooling, stride 2	1	256×256
Sigmoid	1	256×256

We present examples of the transient prediction head probability maps in Figure 9. Compared to the labels used for supervision, the learned transient head overcomes detection failures and false positive detections (row 1, 2). Furthermore this also learns transientness based on view-aggregated features to detect classes that were not used to curate the transient labels used for the supervision of the head as observed in the correct identification of the cart, chairs placed outside for seating, the sculptures placed one after the other, the horse carriage, and the banners in the last row.

E WILD-NVS DATASET DETAILS

This section gives the settings and statistics accompanying the view-set construction in Section 4.

E.1 SOURCE DATA AND COVISIBILITY

MegaDepth-X (Li et al., 2026) provides registered internet photographs, camera parameters, sparse SfM geometry, and dense MVS depth maps. Its reconstruction pipeline uses MASt3R-SfM (Duisterhof et al., 2025) with Doppelgangers++ (Xiangli et al., 2025) to disambiguate visually similar structures, rejects faulty reconstructions through manual verification, and refines the MVS depth to reduce erroneous geometry and transient artifacts. We use these reconstructions and depth maps as inputs to our view-set construction. The release used here contains 441,119 images in 1,868 reconstructions from 1,633 landmarks; 439,705 images have MVS depth.

We evaluate the covisibility test in equation 1 with log-depth tolerance $\tau = 0.05$. Depth maps are resized by nearest-neighbor sampling to 256×256, with camera intrinsics scaled accordingly. Covisibility is computed across each reconstruction. Reconstruction communities are obtained with Louvain community detection (Blondel et al., 2008) on a graph whose edges link views sharing at least 50 reconstructed 3D points, weighted by the number of shared points; they are used only to distribute benchmark targets across groups of views.

E.2 GEOMETRIC CRITERIA AND SELECTION SETTINGS

Near-duplicate views. Let $\mathbf{c}_a$ denote the camera center of view a, $\mathbf{v}_a$ its unit optical-axis direction, and d_t the median valid depth of target t. We use the baseline angle $\theta_t(a, b)$, viewing-direction angle $\phi(a, b)$, and focal-length ratio $r_f(a, b)$:

$$\begin{aligned} \theta_t(a,b) &= \frac{\|\mathbf{c}_a - \mathbf{c}_b\|_2}{d_t}, \qquad \phi(a,b) = \arccos(\mathbf{v}_a^\top \mathbf{v}_b), \\ r_f(a,b) &= \frac{\max(\bar{f}_a, \bar{f}_b)}{\min(\bar{f}_a, \bar{f}_b)}, \qquad \bar{f}_a = \frac{f_{x,a}}{2x_{0,a}}. \end{aligned} \tag{8}$$

Here $f_{x,a}$ is the horizontal focal length and $x_{0,a}$ the horizontal principal-point coordinate, so $2x_{0,a}$ approximates image width. The baseline angle is a small-angle proxy for the triangulation angle, computed in radians and converted to degrees for the thresholds below. Two views are near-duplicates when all three conditions hold: $\theta_t < 1^\circ$, $\phi < 2^\circ$, and $r_f < 1.2$.

We remove candidates near-identical to the target, then suppress duplicates among the candidates, keeping the one with the highest forward covisibility. We also deduplicate accepted targets, keeping

the view set with the most sources, with ties broken by target name. For this last comparison, the baseline normalization uses the mean of the two targets' median depths. **Viewpoint spread and seeds.** The spread score used for source ordering in the main paper combines camera separation and viewing direction:

$$s_t(a,b) = \frac{\theta_t(a,b)}{0.35\,\mathrm{rad}} + 0.5\,\frac{\phi(a,b)}{30^\circ}. \tag{9}$$

The spread of a candidate relative to the selected set S is $\min_{u\in S} s_t(a,u)$. The seed search considers the 32 candidates with highest individual coverage and maximizes $\mathrm{cov}(\{a,b\}) + 0.55 s_t(a,b)$, where cov is the joint coverage defined in the main paper. It first requires pair coverage of at least 0.55 and both seeds to have $\theta_t(a,t), \theta_t(b,t) \geq 5^\circ$. If no pair qualifies, we relax the seed-to-target angle requirement; if no pair meets the coverage floor either, we select the pair with maximum coverage. Up to four qualifying seed pairs are stored for training sampling.

Table 11 specifies the remaining settings of the construction described in the main paper. The separation test is applied between the first $\min(8,m)$ sources of a list of length m, independently of the seed-to-target preference. Training sampling weights are the marginal coverage gain plus 0.55 times spread at insertion, with a floor of 10^{-6}; the first source uses its individual coverage.

E.3 Curation statistics

Table 12 gives the number of views removed at each stage, and Table 13 gives the final split sizes. Across the accepted corpus, the median joint coverage of the first two, four, and eight sources is 89.4%, 95.1%, and 98.1%, respectively. For lists shorter than the requested prefix, these corpus statistics use all available sources.

E.4 Fixed-target benchmark

Following Section 4, we retain test view sets with at least twelve sources and sample up to three distinct targets uniformly without replacement from each reconstruction community. Views that belong to no community, because they share fewer than 50 points with every other view, form one additional group per reconstruction. This produces 1,048 cases from 390 groups across 73 test landmarks. All five source counts $\{4,6,8,10,12\}$, use these same targets and the corresponding prefixes of the stored source lists.

For model evaluation, we choose one anchor from the first four sources and keep it as the first index at every source count. The remaining sources are shuffled for each case and source count. This preserves the reference camera while allowing the source sets to grow from 4 to 6, 8, 10, and 12 views. Table 14 reports their mean target coverage.

E.5 PhotoTourism

We use the official NeRF-W test images as targets and training images as candidate sources for Brandenburg Gate, Sacré-Cœur, and Trevi Fountain. Covisibility is computed from the COLMAP reconstructions and cleaned MVS depth maps distributed with the Image Matching Challenge release of PhotoTourism (Jin et al., 2021). The candidate filtering, deduplication, and source-ordering procedure then follows Section 4 unchanged. We serve six sources per target for comparison with the six-view setting of GenWildSplat (Gupta et al., 2026). The evaluation contains 10, 21, and 19 targets in the three scenes, respectively.

Table 11: Additional view-set construction settings. Coverage fractions use the target's valid-depth pixels as their denominator.

Setting	Value
Minimum target valid-depth fraction	> 0.15
Minimum new coverage during coverage expansion	2%
Coverage floor for the first up to eight sources	80%
Minimum pairwise baseline angle θ_t in that prefix	5°
Coverage weight / spread weight in combined scores	$1 \,/\, 0.55$

Table 12: Curation attrition. Forward eligibility requires at least three candidates with $C_{ij} \geq 0.20$ and target valid-depth fraction above 0.15. Each surviving target defines one view set.

Stage	Removed	Remaining
Input images	—	441,119
Require MVS depth	1,414	439,705
Forward-covisibility and valid-depth eligibility	206,095	233,610
Require enough candidates after reverse-covisibility filtering	3,875	229,735
Require enough candidates after source deduplication	1,044	228,691
Joint-coverage test	1,219	227,472
Source-separation test	31,485	195,987
Target deduplication	25,428	170,559

Table 13: Full view-set pools. Benchmark cases are selected from the held-out pools.

Split	Allocated landmarks	Reconstructions with view sets	View sets
Training	1,500	1,661	154,425
Validation	58	65	8,181
Test	75	79	7,953
Total	1,633	1,805	170,559

If strict construction rejects a test target, we retry with the target valid-depth-fraction threshold, seed-coverage floor, joint-coverage floor, and source-separation floor set to zero, and the minimum candidate count reduced to two. Forward and reverse covisibility thresholds and near-duplicate suppression are retained. Under this fallback, additions after the seed pair are selected by spread without the coverage-expansion phase. Only targets with at least six sources enter evaluation; positive target depth is still required to compute coverage.

Table 14: Coverage on the fixed-target test benchmark, measured over target pixels with valid MVS depth. Targets are identical across columns and source sets are nested.

Source views	4	6	8	10	12
Test targets	1,048	1,048	1,048	1,048	1,048
Mean joint coverage (%)	91.57	94.74	96.17	97.13	97.77