Feedforward Novel View Synthesis for Heterogeneous Cameras

Meng Wei[1] **Cheng Zhang**[1] **Boying Li**[1*] **Yihang Chen**[3,1]
Jianmin Zheng[2] **Hamid Rezatofighi**[1] **Jianfei Cai**[1]
[1]Monash University [2]Nanyang Technological University [3]Shanghai Jiaotong University
{meng.wei,cheng.zhang,boying.li}@monash.edu
{yhchendut}@gmail.com {ASJMZheng}@ntu.edu.sg
{hamid.rezatofighi,jianfei.cai}@monash.edu

Abstract

Feed-forward novel view synthesis has recently shown promising results from sparse posed images, but most existing methods assume that context and target views share a fixed camera family. This homogeneous-camera assumption breaks in practical multi-sensor systems, where perspective, fisheye, and panoramic cameras may coexist and where the target projection may be unseen during training. We study feed-forward NVS across heterogeneous central cameras and identify a key ambiguity introduced by tokenization: a visual token aggregates a projection-dependent bundle of pixel rays, while existing camera encodings mainly expose absolute rays or token-center relations. To address this, we combine token-center relative Camera Positional Encodings with our proposed local raymaps, a token-level representation that explicitly describes the intra-patch ray distribution summarized by each token. We further propose projection-aware 2D RoPE, which replaces raw image-grid coordinates with ray-induced angular coordinates so that relative positional reasoning is aligned across camera projections. Together, these components treat diverse cameras as calibrated samplings of a shared ray space rather than separate visual domains. On ScanNet++ under heterogeneous-camera settings, our method improves over camera-conditioned baselines under mixed-camera evaluation and demonstrates zero-shot generalization to panoramic views.

1 Introduction

Novel view synthesis (NVS) aims to synthesize unseen views from posed observations and remains a central topic in robotics, embodied perception, and world models. Recent feed-forward NVS methods have made substantial progress by predicting novel views from sparse posed images in a single forward pass [1–5]. Most of these methods, however, are primarily designed, trained, and evaluated under a homogeneous-camera setting, where the input and target views share a projection family, such as perspective [5] or panoramic cameras [6].

This homogeneous setting excludes many practical capture and rendering systems. Multi-sensor platforms often combine perspective, fisheye, and panoramic cameras, while downstream applications may require rendering into a camera geometry different from those of the input sensors. In a heterogeneous-camera system, NVS is not only a problem of interpolating views in 3D space, but also one of reasoning about visual evidence across different samplings of ray space. This raises our central question: *how should camera geometry be represented in a single feed-forward model to improve reasoning across projections, even for camera types unseen during training?*

*Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

A purely data-driven approach would expose the model to different intrinsics, distortions, and cross-camera capture conditions. However, this data-driven strategy is difficult to scale because camera variation is continuous and real multi-camera data introduce non-geometric factors such as exposure, white balance, and calibration noise. More importantly, heterogeneous inference requires the model to exhibit *few- or zero-shot generalizability* to camera types broader than training data. A reusable geometric principle for unseen projections is preferable to exhaustive memorization.

This motivates a local view of camera generalization. Although global projection functions can differ substantially, their local ray configurations recur across camera types. An unseen camera may define a new full-image projection, while many of its patch-level ray patterns remain geometrically similar to local patterns observed during training. We therefore formulate heterogeneous-camera generalization as learning over reusable local ray geometry, rather than over discrete camera categories or exhaustively sampled camera instances.

Existing camera encodings, in contrast, operate at three other granularities: absolute pixel-level rays in Plücker raymaps [5], relative camera geometry between views [7, 8], and relative token-center rays [9, 10]. Pixel-level raymaps contain the information needed to characterize intra-patch geometry, but leave the model to extract its local angular structure from world- or camera-frame ray representations. Token-center encodings describe cross-token relations; however, two tokens with the same center ray can still cover different ray distributions under varied projections. Explicitly exposing this patch-level structure can therefore complement existing camera encodings.

Based on these observations, we introduce local raymaps for feed-forward NVS across heterogeneous central cameras, complementary to token-center relative Camera Positional Encodings [10]. A local raymap represents the patch-level ray configuration associated with each visual token, explicitly exposing the intra-patch ray geometry to the feed-forward model. We further propose projection-aware 2D RoPE [11] in ray-induced angular coordinates to align relative positional reasoning across camera projections. Under this derivation, perspective, fisheye, and equirectangular views are treated as calibrated samplings of a shared ray space rather than as separate visual domains. We evaluate the proposed formulation on ScanNet++ v2 [12] by constructing heterogeneous-camera NVS tasks from undistorted, fisheye, and panoramic images, including mixed-camera evaluation and zero-shot target-camera evaluation.

In summary, our contributions are:

- We introduce heterogeneous-camera feed-forward NVS, a setting where context and target views may use different central projection models, and define mixed-camera and zero-shot target-camera evaluation protocols on ScanNet++.
- We combine token-center relative camera positional encoding with local raymaps, a token-level representation that exposes reusable intra-patch ray geometry without relying on discrete camera categories, to support heterogeneous-camera NVS and zero-shot generalization.
- We introduce projection-aware 2D RoPE in ray-induced angular coordinates, aligning relative positional reasoning across camera projections.
- We evaluate heterogeneous-camera NVS on ScanNet++ under mixed-camera and zero-shot target-camera protocols, showing consistent improvements over the evaluated camera-conditioned baselines.

2 Related Work

Wide-FoV Cameras Across 3D Tasks. Wide-FoV cameras, including fisheye and panoramic cameras, are widely used in embodied perception and AR/VR because they observe much larger scene extents than standard perspective cameras. This has motivated projection-aware modeling across several tasks. For reconstruction, panoramic and universal-camera depth methods estimate dense geometry from equirectangular, fisheye, or arbitrary camera inputs [13–15], while feed-forward 3D reconstruction models extend point-map or point-cloud prediction to panoramic and fisheye images [16, 17]. For NVS, omnidirectional and fisheye Gaussian-splatting methods adapt optimization and rasterization to non-perspective projection rays [18–21]. Wide-FoV images have also become a common interface for video generation and 3D world modeling through panorama diffusion and world-generation methods [22–28]. These works show that wide-FoV images require

geometry beyond ordinary image coordinates, and motivate asking how such projection-aware reasoning should be handled in a fundamental 3D task: novel view synthesis.

Feed-forward Novel View Synthesis. Novel view synthesis was popularized in recent learning-based 3D vision through per-scene radiance field optimization, including NeRF [29] and its successors [30, 31]. Feed-forward NVS [1–4, 32, 6, 5] instead synthesizes novel views, or predicts an intermediate 3D representation, directly from input images. Most methods are developed for standard perspective cameras: early generalizable radiance-field methods achieve neural rendering or predict 3D Gaussians using learned image features, while recent Transformer-based models reduce explicit 3D inductive bias and learn to directly synthesize target images, as in LVSM [5].

Recent work extends feed-forward NVS toward wide-FoV cameras, with PanSplat and Splatter-360 introducing specialized designs for feed-forward panorama synthesis [6, 33]. These camera-specific systems are effective, but both perspective and wide-FoV NVS methods usually assume a fixed camera family during training and evaluation. This leaves open a more general question: can a feed-forward NVS model reason when context and target views are produced by different projection models, possibly including target camera types unseen during training? We study this heterogeneous-camera setting by treating perspective, fisheye, and panoramic cameras as calibrated ray samplings of the same scene, and ask what information allows one model to share scene reasoning across them.

Camera Positional Encodings. Reasoning across cameras requires exposing camera geometry to the model. Transformers require positional information to make token interactions spatially meaningful [34–36]; for multi-view generation and reconstruction, this position is not only a 2D image coordinate but also the camera geometry that relates tokens across views. Existing methods therefore encode cameras either as token content or as attention positional encodings. LVSM provides absolute Plücker raymaps as token content, allowing context images and target cameras to be represented in a common 3D frame [5]. PRoPE formulates cameras as relative positional encodings, injecting camera pose and intrinsics into attention [7]. RayRoPE further studies ray-based positional encoding for multi-view attention [9], and UCPE expresses camera geometry through calibrated rays to support different projection models and lens distortions [10].

Despite these advances, camera positional encodings alone do not remove the ambiguity of token ray distributions, which is a key obstacle for reasoning across heterogeneous cameras. Attention-level encodings such as RayRoPE and UCPE explicitly describe geometric relations between token centers, but each patch token aggregates multiple pixel rays whose local distribution depends on camera projection. Existing pixel-level raymaps contain the information needed to characterize intra-patch geometry, but do not explicitly represent its local angular structure. World-space raymaps additionally depend on the global coordinate frame, leaving the model to learn the corresponding invariance [7]. Our local raymaps are therefore complementary: they provide token-level evidence about the intra-patch ray bundle, while existing camera positional encodings provide cross-token relations.

3 Method

We study feed-forward novel view synthesis for heterogeneous central cameras, where context and target views may come from pinhole, fisheye, or panoramic cameras. The key difficulty is not merely representing multiple camera types: calibrated rays already provide a common interface for token inputs and camera positional encodings, as reviewed in Sec. 3.1. Instead, heterogeneity changes the geometric meaning of Transformer tokens themselves. After patchification, each token summarizes a projection-dependent ray distribution, and the image lattice no longer provides a camera-invariant coordinate system for attention.

To support heterogeneous NVS, we first extend the token content with *local raymaps* (Sec. 3.2), allowing the model to reuse local ray patterns rather than infer camera categories from data coverage. We then replace raw image coordinates in 2D RoPE with ray-induced angular coordinates (Sec. 3.3), aligning relative positional phases across projection models. These components allow a single decoder-only Transformer to operate across mixed and even unseen camera geometries. Fig. 1 summarizes how our encoding exposes projection-consistent token geometry compared with priors.

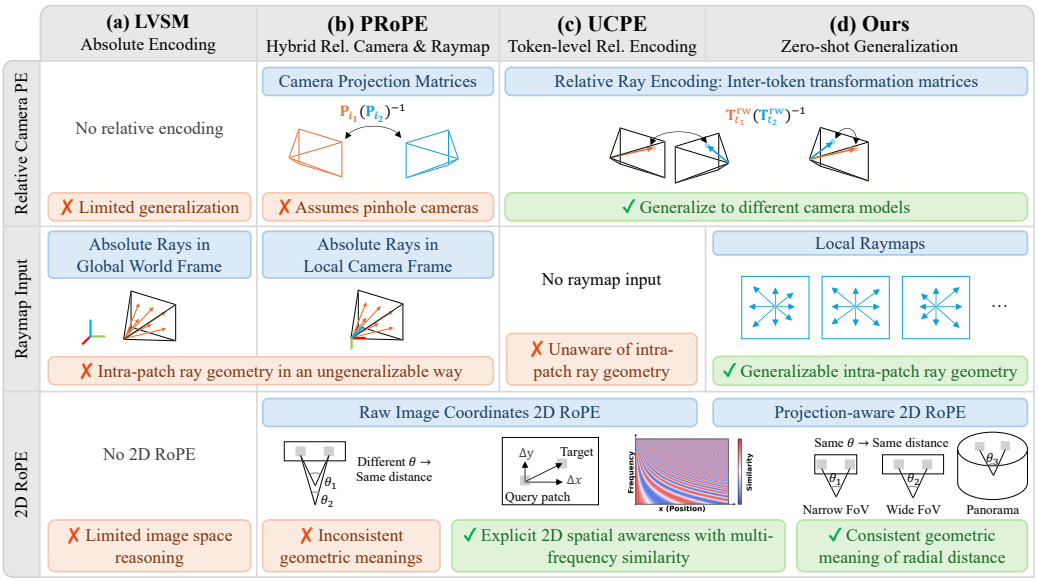


Figure 1: **Comparison of camera encoding methods.** Existing methods encode absolute rays, center-level relations, or raw image coordinates, leaving projection-dependent token ray statistics implicit. Our method combines relative camera encoding, local raymaps, and projection-aware 2D RoPE to explicitly expose consistent geometry across heterogeneous and unseen cameras.

3.1 Prerequisites

Problem setup. Let $\mathcal{C} = \{(I_i, \Pi_i, \mathbf{T}_i^{\text{cw}})\}_{i=1}^{N}$ denote context images I_i with known projection models Π_i and world-to-camera poses $\mathbf{T}_i^{\text{cw}}$. We define each projection model as a pixel-to-ray mapping $\Pi_i : \Omega_i \to \mathbb{S}^2$, which maps an image coordinate $\mathbf{u} \in \Omega_i$ to a unit ray direction $\Pi_i(\mathbf{u})$ in the local camera frame. Given a target camera $(\Pi_t, \mathbf{T}_t^{\text{cw}})$, feed-forward NVS predicts the target image $\tilde{I}_t$ in a single forward pass. We study the heterogeneous-camera setting, where the projection models Π_i and Π_t may differ across views and may even include camera types unseen during training. We assume central cameras, where all camera rays within one image pass through a single camera center.

Transformer-based NVS. Our backbone follows the decoder-only Transformer formulation of LVSM [5]. The model converts context observations and the target camera into a single sequence of patch tokens [35], applies self-attention, and decodes the output tokens at target positions into $\tilde{I}_t$.

In LVSM, camera geometry is provided as an absolute Plücker raymap R_i^{pl}, which stores the 3D ray associated with each image location in a common coordinate frame. For a context view, the RGB image and its raymap are tokenized jointly; for the target view, only the target raymap is tokenized:

$$\mathbf{z}_{i,p} = \theta_{\text{ctx}}\left(I_i, R_i^{\text{pl}}\right)_p, \qquad \mathbf{z}_{t,p} = \theta_{\text{tgt}}\left(R_t^{\text{pl}}\right)_p, \tag{1}$$

where p indexes image patches, and θ_{ctx} and θ_{tgt} denote the context and target tokenizers. The resulting context tokens $\{\mathbf{z}_{i,p}\}$ and target tokens $\{\mathbf{z}_{t,p}\}$ are concatenated before entering the Transformer. This formulation makes pixel-aligned camera rays available as token content, while geometric relations between tokens are still learned implicitly through attention.

Cameras as positional encodings. Camera positional encodings inject camera geometries directly into attention operations, complementary to input raymaps. Following RoPE-style formulations [11], each token p is assigned a camera-aware transformation matrix $\mathbf{D}_p$:

$$\mathbf{D}_p = \mathbf{I}_{d/4} \otimes \mathbf{T}_p, \tag{2}$$

where $\mathbf{I}$ is the identity matrix, d is the feature dimension, $\otimes$ denotes the Kronecker product, and $\mathbf{T}_p$ specifies the camera-dependent transform for token p. During attention, the query and key features

are transformed by $\mathbf{D}$ before computing attention scores:

$$O = \mathrm{Attn}\big(\mathbf{D}^\top Q, \mathbf{D}^{-1} K, V\big)\,. \tag{3}$$

For a query token q and a key token k, the attention logit is therefore modulated by the relative transform between their geometries:

$$\big(\mathbf{D}_q^\top Q_q\big)^\top \big(\mathbf{D}_k^{-1} K_k\big) = Q_q^\top \mathbf{D}_q \mathbf{D}_k^{-1} K_k, \qquad \mathbf{D}_q \mathbf{D}_k^{-1} = \mathbf{I}_{d/4} \otimes \mathbf{T}_q \mathbf{T}_k^{-1}. \tag{4}$$

The definition of $\mathbf{T}_p$ therefore determines which geometric relation is visible to attention. CaPE [37] uses the camera pose $\mathbf{T}^{\mathrm{cw}}_{c(p)}$, where $c(p)$ is the camera index associated with token p. Geometric Transform Attention (GTA) [8] additionally transforms the value matrix. Projective Positional Encoding (PRoPE) [7] incorporates camera intrinsics $\mathbf{K}$ and defines the camera-frame raymaps, named CamRay. RayRoPE [9] and Unified Camera Positional Encoding (UCPE) [10] define positional relations using calibrated ray geometry, extending attention-level encodings to diverse camera models and lens distortions. Our method thus builds on the ray-based UCPE formulation, which defines $\mathbf{T}_p$ as a function of the token-center ray direction, compatible with diverse camera models.

3.2 Extending Camera Positional Encodings for Heterogeneous Cameras

The prerequisites above describe two common ways to expose camera geometry to a Transformer: raymaps provide token content, and camera positional encodings modulate attention through token-center geometry. These mechanisms are often sufficient in a homogeneous-camera setting because the image lattice carries a stable geometric meaning. When all views follow the same projection family, a patch thus corresponds to a predictable local angular footprint, and a displacement on the token grid is also a stable proxy for a displacement in ray space.

Heterogeneous cameras break this structural shortcut, creating a representation ambiguity for current Transformer-based methods, where tokenized visual content doesn't share similar geometric meanings. Tokens covering the same image-space pixel range may aggregate different ray bundles under different camera projections: these tokens can cover a narrow perspective region, a stretched fisheye region, or a panoramic region at a different latitude. This mismatch can hinder the model from generalizing spatial relationships consistently across camera types.

To make this mismatch explicit, let Ω_p denote the pixel set of patch p, and let $\bar{\mathbf{u}}_p$ denote its center pixel. A context token is produced by jointly tokenizing the RGB patch and its camera representation,

$$\mathbf{z}_{i,p} = \theta_{\mathrm{ctx}}(I_i, R_i)_p\,, \tag{5}$$

where R_i denotes any camera ray representation used by the backbone, as in Fig. 1. Since Π_i maps pixels to camera-frame unit rays, the token-center ray is $\bar{\mathbf{r}}_{i,p} = \Pi_i(\bar{\mathbf{u}}_p)$, which is the primary geometric cue directly exposed to attention-level camera encodings.

However, the RGB evidence summarized by $\mathbf{z}_{i,p}$ is aggregated from the entire patch rather than a single pixel. Each token actually corresponds to a bundle of rays, whose support is defined as

$$\mathcal{B}_{i,p} = \{\Pi_i(\mathbf{u}) \mid \mathbf{u} \in \Omega_p\}\,. \tag{6}$$

Overall, $\bar{\mathbf{r}}_{i,p}$ only captures the center viewing direction of the token, whereas $\mathcal{B}_{i,p}$ represents the entire local ray distributions over the whole patch. In homogeneous-camera NVS, the former serves as a compact surrogate for the latter because the projection model consistently determines how the local ray bundle varies around the center ray. In heterogeneous-camera NVS, however, this surrogate relationship no longer holds reliably. For example, two key tokens, with indices k_1, k_2 and center-rays $\bar{\mathbf{r}}_{k_1}, \bar{\mathbf{r}}_{k_2}$, may share the same center-level geometric relation to a query token q with the center-ray $\bar{\mathbf{r}}_q$, while aggregating different local ray distributions $\mathcal{B}$:

$$g(\bar{\mathbf{r}}_q, \bar{\mathbf{r}}_{k_1}) = g(\bar{\mathbf{r}}_q, \bar{\mathbf{r}}_{k_2})\,, \quad \mathcal{B}_{k_1} \neq \mathcal{B}_{k_2}, \tag{7}$$

where $g(\cdot,\cdot)$ denotes the token-level relative ray relation visible to attention-level camera encodings. In this case, attention sees equivalent center-ray geometry even though the two keys aggregate different projection-induced ray bundles. The missing distinguishable variable is thus the local mapping from patch pixels to nearby rays.

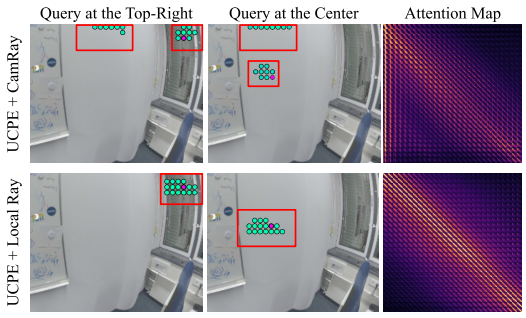


Figure 2: **Attention sink phenomenon** in visualizations of the "attended tokens with Top-16 weights" (green) and attention maps (target view only) in the 1st layer, with target-view tokens as queries (pink). Using CamRay representations makes tokens attend to learned **placeholder** tokens rather than to geometrically near tokens, while using local raymaps yields a cleaner locality.

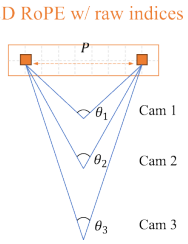


Figure 3: **2D RoPE with raw image indices.** Across heterogeneous cameras, relative positions in the raw image indices may have inconsistent geometric meanings.

From raymaps to local raymaps. Explicit raymaps provide a unified interface for heterogeneous cameras by specifying where sampled pixels look, potentially indicating the missing local ray statistics. However, world-space Plücker raymaps [5] and camera-frame raymaps (CamRay [7]) do not explicitly identify the local ray distributions hidden by patchification, and therefore leave these to models to infer. This implicit modelling could therefore lead to the attention-sink behavior illustrated in Fig. 2, where CamRay attention encourages tokens to first attend to some fixed model-learned tokens, even when their locations mismatch, whereas local raymaps show greater locality.

We address these issues by introducing *local raymaps*, which explicitly describe the local ray distribution within each token, easing the learning burden. The local raymaps shift the generalization unit from global projection types to reusable local ray patterns. While full-image projection models may differ widely, their patch-level ray configurations can recur across camera systems, allowing the representation to transfer to unseen projections without relying on camera-categorical encodings.

Construction of local raymaps. For each $P \times P$ image patch, we retain the corresponding $P \times P$ calibrated unit rays sampled at pixel centers. We encode each local ray bundle in a coordinate system normalized by the patch-center ray. Let $\bar{\mathbf{r}}_{i,p} = \Pi_i(\bar{\mathbf{u}}_p)$ be the center ray of patch p in view i. For each patch, we construct a rotation $\mathbf{Q}_{i,p} \in \mathrm{SO}(3)$ that maps camera-frame rays into a patch-local frame with $\mathbf{Q}_{i,p}\bar{\mathbf{r}}_{i,p} = \mathbf{e}_z$, where $\mathbf{e}_z = (0,0,1)^\top$. In other words, its local z-axis is the center ray and other in-plane axes are fixed by a deterministic reference axis. In practice, we use the up vector to determine the roll. When the up vector is nearly parallel to the center ray, we use a deterministic alternate reference axis.

For each pixel within the patch $\mathbf{u} \in \Omega_p$, the local ray is $\mathbf{r}^{\mathrm{local}}_{i,p}(\mathbf{u}) = \mathbf{Q}_{i,p}\Pi_i(\mathbf{u})$. This shared rotation preserves intra-patch angular relations and sampling patterns. We stack the rotated rays in pixel order to obtain the local raymap patch $\mathbf{R}^{\mathrm{local}}_{i,p} \in \mathbb{R}^{P\times P\times 3}$. Applying this normalization to all patches gives the local raymap R^{local}_i. We provide local raymaps as token-level geometric content together with the existing image:

$$\mathbf{z}_{i,p} = \theta_{\mathrm{ctx}}\big(I_i, R^{\mathrm{local}}_i\big)_p\,, \qquad \mathbf{z}_{t,p} = \theta_{\mathrm{tgt}}\big(R^{\mathrm{local}}_t\big)_p\,, \tag{8}$$

where R^{local}_i and R^{local}_t denote the local raymaps for context and target views respectively.

This design is compatible with the token-center relative camera positional encodings, UCPE specifically. Center-level encodings guide which tokens should interact geometrically, while local raymaps describe what ray footprint each interacting token summarizes. As illustrated by the attention visualization in Fig. 2, making the local footprint explicit reduces the burden on attention to recover patch-scale ray statistics from global or center-only ray cues, resulting in cleaner attention maps.

We do not explicitly model the angle between the sensor plane and the patch-center ray, but the calibrated samples retain the resulting local angular extent, anisotropy, and non-uniform spacing. Local raymaps also do not explicitly compute solid-angle footprints or projection Jacobians.

3.3 Projection-Aware 2D RoPE for Heterogeneous Cameras

Transformer-based feed-forward novel view synthesis requires a positional mechanism that makes token interactions aware of 2D spatial layout. Recent Camera Positional Encodings follow the GTA [8] formulation which defines one SE(3) transform for camera geometries and another SO(2) transform (2D RoPE [11]) for multi-frequency image-space positions, where 2D RoPE encodes relative positions on the raw image patch indices.

For homogeneous cameras, the patch indices have stable geometric meanings. A displacement of one patch horizontally or vertically matches a fixed displacement in the underlying image projection. Standard 2D RoPE is thus aligned with the camera model. Let $\mathbf{u}_p = (u_p, v_p)^\top$ denote the 2D patch coordinate of token p. 2D RoPE applies a coordinate-dependent rotation to queries and keys,

$$\tilde{\mathbf{q}}_q = \mathbf{D}_{\text{RoPE}}(\mathbf{u}_q)\mathbf{q}_q, \qquad \tilde{\mathbf{k}}_k = \mathbf{D}_{\text{RoPE}}(\mathbf{u}_k)\mathbf{k}_k, \tag{9}$$

where $\mathbf{D}_{\text{RoPE}}(\cdot)$ is a block-diagonal matrix of frequency-dependent rotations, which is applied to $u-$ and v-axis separately. The resulting attention logit depends on the relative lattice displacement,

$$\tilde{\mathbf{q}}_q^\top \tilde{\mathbf{k}}_k = \mathbf{q}_q^\top \mathbf{D}_{\text{RoPE}}(\mathbf{u}_k - \mathbf{u}_q)\mathbf{k}_k. \tag{10}$$

This formulation implicitly assumes that $\mathbf{u}_k - \mathbf{u}_q$ has a consistent geometric meaning across images. However, this assumption breaks under heterogeneous cameras. As illustrated by Fig. 3, the same lattice displacement can correspond to different angular displacements, since the projection topology could change the mapping from image coordinates to rays. Thus, the raw image indices are no longer a camera-invariant positional domain for 2D RoPE.

We therefore replace raw image patch indices with ray-induced angular coordinates. Since our input is already conditioned on camera raymaps, we directly compute the RoPE coordinates from the ray direction of each token. Let c_i denote the camera associated with token i, and let $\mathbf{d}_i^{c_i} = (d_{i,x}^{c_i}, d_{i,y}^{c_i}, d_{i,z}^{c_i})$ denote the unit ray direction of token i in that camera's local frame. We then map each ray to a longitude-latitude coordinate,

$$\boldsymbol{\eta}_i^{c_i} = h(\mathbf{d}_i^{c_i}) = (\lambda_i^{c_i}, \varphi_i^{c_i})^\top \in \mathbb{R}^2, \tag{11}$$

$$\lambda_i^{c_i} = \operatorname{atan2}(d_{i,x}^{c_i}, d_{i,z}^{c_i}), \qquad \varphi_i^{c_i} = \operatorname{atan2}\left(d_{i,y}^{c_i}, \sqrt{(d_{i,x}^{c_i})^2 + (d_{i,z}^{c_i})^2}\right). \tag{12}$$

Equivalently, since $\mathbf{d}_i^{c_i}$ is unit-normalized, $\varphi_i^{c_i} = \arcsin(d_{i,y}^{c_i})$. The same ray-to-angle mapping is used for all camera models in their respective camera frames, providing a common angular parameterization despite their different image lattices.

The RoPE rotations are then applied using the angular ray coordinate $\boldsymbol{\eta}_i^{c_i}$:

$$\tilde{\mathbf{q}}_q = \mathbf{D}_{\text{RoPE}}(\boldsymbol{\eta}_q^{c_q})\mathbf{q}_q, \qquad \tilde{\mathbf{k}}_k = \mathbf{D}_{\text{RoPE}}(\boldsymbol{\eta}_k^{c_k})\mathbf{k}_k. \tag{13}$$

The relative phase now depends on angular-coordinate differences. However, angular RoPE becomes unstable for edge patches near the panorama seam or undefined at the poles. We thus use it alongside ray-based UCPE, which supplies relative camera geometry without using longitude–latitude coordinates; this complements angular RoPE but does not remove its singularities.

4 Experiments

We evaluate the proposed method under heterogeneous-camera novel view synthesis, where accurate rendering requires both cross-view geometric reasoning and robustness to changes in camera projection. Our experiments study two complementary questions: whether the model improves reconstruction when camera types are mixed within the training distribution, and whether the learned representation transfers to a target camera type that is absent during training.

4.1 Experimental Setup

Dataset. We train and evaluate on ScanNet++ v2 [12], which captured indoor scenes with different cameras. We use fisheye (DSLR), undistorted, and panoramic images to construct heterogeneous-camera novel view synthesis pairs. For the zero-shot setting, panoramic views are excluded during training, allowing us to evaluate generalization to unseen, challenging camera geometry. We use the official training and validation splits, and use 224×336 resolution for perspective cameras and 256×512 for panorama cameras, at a similar scale to the training resolution used in PRoPE [7].

Table 1: **NVS comparisons on ScanNet++.** In-distribution data evaluation. Our methods achieve best results across three protocols.

Methods	Rel. Camera PE	Raymaps	2D RoPE	Undistorted			Fisheye			Mixed		
				PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
LVSM [5]	-	Plücker	Raw	24.716	0.781	0.174	24.290	0.763	0.175	24.104	0.766	0.183
PRoPE [7]	PRoPE	CamRay	Raw	23.510	0.760	0.173	22.780	0.733	0.181	21.614	0.712	0.207
UCPE-N	Rel. Ray	-	Raw	26.175	0.812	0.133	25.699	0.799	0.131	25.149	0.795	0.142
UCPE-P	Rel. Ray	Plücker	Raw	26.083	0.811	0.136	25.568	0.797	0.135	25.204	0.796	0.144
UCPE-C	Rel. Ray	CamRay	Raw	26.141	0.813	0.132	25.591	0.799	0.131	25.278	0.798	0.140
Ours (Raw)	Rel. Ray	Local Ray	Raw	26.308	0.815	0.131	25.784	0.800	0.130	25.347	0.799	0.139
Ours (Full)	Rel. Ray	Local Ray	Proj.	26.429	0.815	0.129	25.911	0.800	0.128	25.670	0.801	0.136

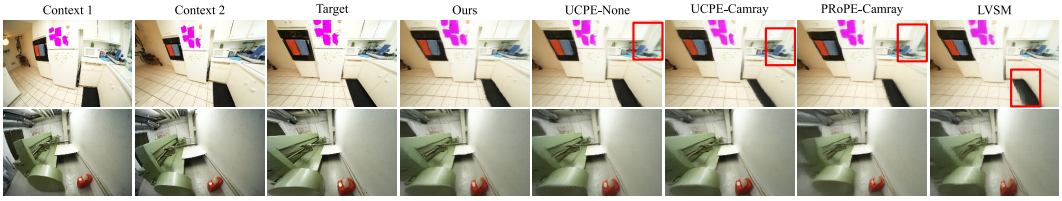


Figure 4: **Qualitative comparisons on mixed camera evaluation.** Both our method and the LVSM baseline have capabilities to infer a correct field of view across different camera types. Compared with others, our method produces sharper details in the shown examples.

Baselines. We compare our method with representative camera-conditioning strategies under heterogeneous camera settings. For token-level camera embeddings, we evaluate (1) *the Plücker raymaps* used in LVSM [5] and (2) *the CamRay raymaps* introduced in PRoPE [7]. These baselines expose camera geometry directly to visual tokens, but differ in their coordinate dependence: Plücker raymaps encode rays in a global frame, whereas CamRay encodes local camera-frame ray directions. For attention-level camera positional encodings, we compare against PRoPE [7], which injects relative projective camera geometry into self-attention, and UCPE [10], which provides a ray-based relative camera encoding for diverse cameras. Our variants use UCPE as the camera positional encoding and differ in the token-level raymap and the coordinate domain of 2D RoPE.

Training details. For the default LVSM setting, all methods are trained in the same codebase with identical hyperparameters. Following PRoPE [7], we adopt the lightweight LVSM [5] backbone with approximately 24.5M parameters to match a practical academic compute budget. Each training sample contains 2 context views and 1 target view, following the default PRoPE setting. Models are trained for 80k iterations on 2 NVIDIA A100 GPUs with a batch size of 8. We evaluate the performance of NVS using PSNR, SSIM [38], and LPIPS [39] metrics.

4.2 Evaluations

In-distribution heterogeneous evaluation. We first evaluate NVS performance on fisheye and undistorted camera types. The evaluation protocols include two homogeneous-camera cases, "fisheye → fisheye" and "undistorted → undistorted", and a mixed-camera case where context and target views are randomly drawn from the available camera types. We also apply the random focal length similar to PRoPE [7]. We use the Splatt3R benchmark [40] for view selection, which measures the overlapping ratio between the context views and the target view. While this overlapping ratio is measured on the undistorted images, we use the same camera poses for fisheye images as proximity.

Table 1 shows that relative ray-based camera positional encoding is an effective baseline for heterogeneous NVS, and that the proposed local raymaps further improve performance. The gains over UCPE baselines are consistent across these in-distribution protocols. Replacing raw 2D RoPE with projection-aware RoPE gives consistent additional gains over "Ours (Raw)", especially for fisheye and mixed-camera evaluation, where the image lattice is less geometrically stable. These results support the role of local ray distributions and projection-aware positional coordinates as complementary components. Fig. 4 provides qualitative comparisons under the heterogeneous-camera settings. Our method synthesizes target views with correct field of views and achieves sharper results, while other methods either struggle with correct camera relationship inference or generate blurred visual content.

Table 2: **Larger-backbone evaluation.** We evaluated variants trained with SVSM backbones under the Mixed protocol.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
UCPE-N	24.844	0.860	0.123
UCPE-P	24.934	0.864	0.121
UCPE-C	24.845	0.863	0.118
Ours	25.718	0.866	0.114

Table 3: **Zero-shot NVS comparisons.** Evaluated on panoramic targets. Raymaps: N/P/C/L=None/Plücker/CamRay/Local Ray.

Ray.	2D RoPE	Virtual Cams → Panorama			DSLR → Panorama		
		PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
N	Raw	22.426	0.731	0.167	22.659	0.765	0.237
P	Raw	22.853	0.740	0.170	22.760	0.762	0.257
C	Raw	Fail	Fail	Fail	Fail	Fail	Fail
L	Raw	23.139	0.755	0.141	22.999	0.774	0.226
L	Proj.	23.346	0.760	0.142	22.815	0.769	0.232



Figure 5: **Qualitative comparisons on zero-shot panoramic evaluation.** Left: virtual cameras to panorama images; Right: DSLR to panorama. Our methods exhibit fewer artifacts in the zero-shot generalization to panoramic targets.

Larger backbone. We also evaluate the 93.6M-parameter SVSM [41], with 8 encoder and 8 decoder layers, under the same ScanNet++ v2 Mixed evaluation protocol. All SVSM variants are trained for 200K steps with 4 target views per sample and a total batch size of 8. Table 2 shows consistent gains over other baselines, supporting transfer beyond the lightweight LVSM backbone.

Zero-shot generalizability. We next evaluate zero-shot generalization to panoramic target cameras, which are not used during training. We consider two protocols. In "Virtual Cams → Panorama", 6 context images are fisheye views synthesized from panoramas with random virtual camera models. In "DSLR → Panorama", context images come from real DSLR captures (fisheye) and targets are panoramas. To ensure enough coverage, we use 10 context images as inputs. The details of sampling method could be found in the supplementary materials. For all methods, we use the relative center-ray positional encoding (UCPE [10]). Since this setting involves cross-device appearance differences, we apply the same local histogram matching color correction module to all methods.

Table 3 shows that local raymaps improve transfer to unseen panorama geometry over the evaluated raymap baselines. As illustrated in Fig. 5, CamRay fails in this zero-shot setting, suggesting that its camera-frame ray inputs alone are not sufficient to cover the unseen projection regime. Our method produces sharper results compared with other baselines, indicating the zero-shot generalizability of the proposed components. The real-device setting also involves coverage and appearance differences that calibrated geometry alone does not resolve.

Table 4: **Ablations on raymap configurations.** We ablate the patch size P and ray sampling density s, with default values $P = 8$ and $s = 1$.

Setting	UCPE-C			Ours		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Default	25.278	0.798	0.140	**25.670**	**0.801**	**0.136**
P=16	23.871	0.765	0.171	**24.578**	**0.772**	**0.170**
s=0.5	25.231	0.797	0.140	**25.535**	**0.798**	**0.137**
s=2	25.297	0.798	0.140	**25.679**	**0.801**	**0.136**

Table 5: **RoPE ablations.** Raw vs. projection-aware RoPE on different variants, where C/L stands for CamRay/Local Ray respectively.

2D RoPE	Fisheye			Mixed		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
PRoPE(C)-Raw	22.780	0.733	0.181	21.614	0.712	0.207
PRoPE(C)-Proj.	25.049 +	0.781 +	0.145 -	24.856 +	0.783 +	0.153 -
UCPE(L)-Raw	25.784	0.800	0.130	25.347	0.799	0.139
UCPE(L)-Proj.	25.911 +	0.800 =	0.128 -	25.670 +	0.801 +	0.136 -

Table 6: **Inference time cost (ms).**

Method	Ray	Total
UCPE-N	0	23.3
UCPE-C	5.5	28.8
Ours (Raw)	6.5	29.8
Ours (Full)	6.5	34.8

Table 7: **Calibration sensitivity.** 200 Mixed samples. Clean: absolute scores; noise groups: changes from clean.

Method	Clean			Distortion noise			Pose noise		
	PSNR ↑	SSIM ↑	LPIPS ↓	ΔPSNR ↑	ΔSSIM ↑	ΔLPIPS ↓	ΔPSNR ↑	ΔSSIM ↑	ΔLPIPS ↓
UCPE-N	25.43	0.800	0.121	−0.086	−0.002	+0.0003	−1.468	−0.0323	+0.0082
UCPE-C	25.64	0.803	0.119	−0.077	−0.0013	+0.0003	−1.554	−0.0333	+0.0084
Ours	26.04	0.807	0.116	−0.095	−0.0014	+0.0004	−1.683	−0.0339	+0.0085

4.3 Ablations

There are two main components of our methods: the local raymap input and the projection-aware RoPE. Previous experiments have shown the isolated effects of these components, where "Ours (Raw)" does not use the projection-aware RoPE and has shown performance gain over other raymap inputs. To further verify our projection-aware RoPE, we replace the raw image indices RoPE with projection-aware RoPE in PRoPE [7]. As shown in Table 5, this modification improves overall performance, demonstrating that projection-aware RoPE is not specific to our overall architecture but can be applied to other camera-conditioning methods for heterogeneous-camera NVS.

Table 4 ablates the raymap configurations of varied patch sizes and ray-sampling densities under the Mixed protocol. Sampling rate s scales each axis of the ray grid, yielding $sH \times sW$ samples. Our default configuration uses patch size 8 and sampling density 1. Camera-input channel counts are matched by zero-padding where needed. Our full method improves on UCPE-C across these settings.

4.4 Robustness and efficiency

To test the robustness of the proposed method, we perturb calibration on 200 Mixed samples, keeping RGB inputs fixed and using identical perturbations across methods. Table 7 reports 5% relative distortion-coefficient errors and perturbations to both rotation and translation. Ours retains the best absolute scores after perturbation, although its PSNR drop under pose noise is larger.

Table 6 reports ray construction and complete inference time on an A100 GPU at 224×336, with unchanged parameter counts. Relative to UCPE-C, Ours (Raw) adds 1.0 ms (3.5%), while Ours (Full) adds 6.0 ms (20.8%). The angular RoPE variant therefore carries a larger runtime cost. Further camera-transition, resolution, multi-target, and cross-dataset evaluations are provided in Sec. B.

5 Conclusion

We studied feed-forward NVS in a heterogeneous-camera setting, where different projection families are handled by a single model rather than separate camera-specific systems. Since camera variation is continuous and difficult to cover exhaustively with calibrated heterogeneous-camera data, we argue that generalization should come from reusable geometric units rather than memorized camera types. By exposing token-local ray distributions and ray-domain positional coordinates, our method improves mixed-camera synthesis and exhibits zero-shot generalization to unseen target cameras.

Limitations. Heterogeneous-camera NVS also reflects a broader data challenge in the field. Our method assumes calibrated central cameras and is evaluated mainly on indoor ScanNet++ scenes.Scene depth and occlusion are inferred implicitly from RGB observations, while device-dependent appearance differences remain unaddressed. More diverse calibrated data and explicit treatment of these challenges are promising directions for improving heterogeneous-camera NVS.

Acknowledgments and Disclosure of Funding

Funding. This research used resources from the National Computational Infrastructure (NCI Australia), an NCRIS-enabled capability supported by the Australian Government. This research was also supported by Monash eResearch capabilities, including M3. We acknowledge Maincode for providing computational resources.

References

[1] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[2] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14124–14133, 2021.

[3] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[4] 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*. Springer, 2024.

[5] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. Lvsm: A large view synthesis model with minimal 3d inductive bias. In *The Thirteenth International Conference on Learning Representations*, 2025.

[6] Cheng Zhang, Haofei Xu, Qianyi Wu, Camilo Cruz Gambardella, Dinh Phung, and Jianfei Cai. Pansplat: 4k panorama synthesis with feed-forward gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[7] Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as relative positional encoding. *Advances in Neural Information Processing Systems*, 2025.

[8] Takeru Miyato, Bernhard Jaeger, Max Welling, and Andreas Geiger. Gta: A geometry-aware attention mechanism for multi-view transformers. In *International Conference on Learning Representations (ICLR)*, 2024.

[9] Yu Wu, Minsik Jeon, Jen-Hao Rick Chang, Oncel Tuzel, and Shubham Tulsiani. Rayrope: Projective ray positional encoding for multi-view attention. *arXiv preprint arXiv:2601.15275*, 2026.

[10] Cheng Zhang, Boying Li, Meng Wei, Yan-Pei Cao, Camilo Cruz Gambardella, Dinh Phung, and Jianfei Cai. Unified camera positional encoding for controlled video generation. *arXiv preprint arXiv:2512.07237*, 2025.

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

[12] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023.

[13] Zidong Cao, Jinjing Zhu, Weiming Zhang, Hao Ai, Haotian Bai, Hengshuang Zhao, and Lin Wang. Panda: Towards panoramic depth anything with unlabeled panoramas and mobius spatial augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 982–992, 2025.

[14] Yuliang Guo, Sparsh Garg, S Mahdi H Miangoleh, Xinyu Huang, and Liu Ren. Depth any camera: Zero-shot metric depth estimation from any camera. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26996–27006, 2025.

[15] Luigi Piccinelli, Christos Sakaridis, Mattia Segu, Yung-Hsu Yang, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unik3d: Universal camera monocular 3d estimation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1028–1039, 2025.

[16] Yijing Guo, Mengjun Chao, Luo Wang, Tianyang Zhao, Haizhao Dai, Yingliang Zhang, Jingyi Yu, and Yujiao Shi. Panovggt: Feed-forward 3d reconstruction from panoramic imagery. *arXiv preprint arXiv:2603.17571*, 2026.

[17] Ruxiao Duan, Erin Hong, Dongxu Zhao, Eric Turner, Alex Wong, and Yunwen Zhou. Fisheye3r: Adapting unified 3d feed-forward foundation models to fisheye lenses. *arXiv preprint arXiv:2603.28896*, 2026.

[18] Suyoung Lee, Jaeyoung Chung, Jaeyoo Huh, and Kyoung Mu Lee. Odgs: 3d scene reconstruction from omnidirectional images with 3d gaussian splattings. *Advances in Neural Information Processing Systems*, 37:57050–57075, 2024.

[19] Longwei Li, Huajian Huang, Sai-Kit Yeung, and Hui Cheng. Omnigs: Fast radiance field reconstruction using omnidirectional gaussian splatting. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2260–2268. IEEE, 2025.

[20] Zimu Liao, Siyan Chen, Rong Fu, Yi Wang, Zhongling Su, Hao Luo, Li Ma, Linning Xu, Bo Dai, Hengjie Li, et al. Fisheye-gs: Lightweight and extensible gaussian splatting module for fisheye cameras. *arXiv preprint arXiv:2409.04751*, 2024.

[21] Qi Wu, Janick Martinez Esturo, Ashkan Mirzaei, Nicolas Moenne-Loccoz, and Zan Gojcic. 3dgut: Enabling distorted cameras and secondary rays in gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26036–26046, 2025.

[22] Qian Wang, Weiqi Li, Chong Mou, Xinhua Cheng, and Jian Zhang. 360dvd: Controllable panorama video generation with 360-degree video diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6913–6923, 2024.

[23] Cheng Zhang, Hanwen Liang, Donny Y Chen, Qianyi Wu, Konstantinos N Plataniotis, Camilo Cruz Gambardella, and Jianfei Cai. Panflow: Decoupled motion control for panoramic video generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 12385–12393, 2026.

[24] Jing Tan, Shuai Yang, Tong Wu, Jingwen He, Yuwei Guo, Ziwei Liu, and Dahua Lin. Imagine360: Immersive 360 video generation from perspective anchor. *arXiv preprint arXiv:2412.03552*, 2024.

[25] Rundong Luo, Matthew Wallingford, Ali Farhadi, Noah Snavely, and Wei-Chiu Ma. Beyond the frame: Generating 360 panoramic videos from perspective videos. *arXiv e-prints*, pages arXiv–2504, 2025.

[26] Zhongqi Yang, Wenhang Ge, Yuqi Li, Jiaqi Chen, Haoyuan Li, Mengyin An, Fei Kang, Hua Xue, Baixin Xu, Yuyang Yin, et al. Matrix-3d: Omnidirectional explorable 3d world generation. *arXiv preprint arXiv:2508.08086*, 2025.

[27] Jiahao Wang, Luoxin Ye, TaiMing Lu, Junfei Xiao, Jiahan Zhang, Yuxiang Guo, Xijun Liu, Rama Chellappa, Cheng Peng, Alan Yuille, et al. Evoworld: Evolving panoramic world generation with explicit 3d memory. *arXiv preprint arXiv:2510.01183*, 2025.

[28] HunyuanWorld Team, Zhenwei Wang, Yuhao Liu, Junta Wu, Zixiao Gu, Haoyuan Wang, Xuhui Zuo, Tianyu Huang, Wenhuan Li, Sheng Zhang, et al. Hunyuanworld 1.0: Generating immersive, explorable, and interactive 3d worlds from words or pixels. *arXiv preprint arXiv:2507.21809*, 2025.

[29] 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. *Communications of the ACM*, 65 (1):99–106, 2021.

[30] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, August 2023. ISSN 0730-0301, 1557-7368. doi: 10.1145/3592433.

[31] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023.

[32] Yuedong Chen, Chuanxia Zheng, Haofei Xu, Bohan Zhuang, Andrea Vedaldi, Tat-Jen Cham, and Jianfei Cai. Mvsplat360: Feed-forward 360 scene synthesis from sparse views. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[33] Zheng Chen, Chenming Wu, Zhelun Shen, Chen Zhao, Weicai Ye, Haocheng Feng, Errui Ding, and Song-Hai Zhang. Splatter-360: Generalizable 360° gaussian splatting for wide-baseline panoramic images. *arXiv preprint arXiv:2412.06250*, 2024.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

[35] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[36] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[37] Kaixin Xiong, Shi Gong, Xiaoqing Ye, Xiao Tan, Ji Wan, Errui Ding, Jingdong Wang, and Xiang Bai. Cape: Camera view position embedding for multi-view 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[38] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

[39] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

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

[41] Evan Kim, Hyunwoo Ryu, Thomas W. Mitchel, and Vincent Sitzmann. Scaling view synthesis transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[42] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024.

A Implementation Details

A.1 Architecture

We use the lightweight decoder-only LVSM backbone [5] for our default experiments. Each model has 6 Transformer layers, 16 attention heads, a token dimension of 768, and patch size 8, resulting in approximately 24.5M parameters. Context views and the target camera are converted into patch tokens and concatenated into one sequence before self-attention.

Optimization. We follow the PRoPE training recipe [7]: for this backbone, all methods are trained for 80k iterations with a total batch size of 8 on two NVIDIA A100 GPUs. We use a cosine learning-rate scheduler with 2.5k warmup iterations. Unless otherwise specified in the experiment section, each training sample contains two context views and one target view, and all baselines use the same backbone, training schedule, image resolution, and view sampling protocol.

A.2 DSLR-to-Panorama sampling and evaluation pair construction

We evaluate zero-shot panorama prediction by constructing "DSLR→panorama" pairs in ScanNet++. For each scene, every panocam frame is treated as a target panorama p, and we select a fixed-size set of K DSLR context images,

$$\mathcal{C}(p) = \{i_1, \ldots, i_K\}. \tag{14}$$

Our sampling objective is twofold: (1) context views should be spatially close to the target panorama, and (2) their viewing directions should cover as much of the target panorama's spherical field of view as possible.

Overall procedure. For each target panorama p, we construct $\mathcal{C}(p)$ with the following geometry-guided pipeline. First, we keep only DSLR frames whose camera centers are within a distance threshold of the panorama center. Second, we transform each remaining DSLR viewing direction into the local coordinate frame of the target panorama. Third, we discretize the panorama sphere into angular bins and compute which bins each DSLR image covers. Finally, we greedily add the DSLR image that contributes the largest amount of previously uncovered solid angle. The greedy selection stops once K context images have been selected. During evaluation, we only keep target panoramas whose final coverage exceeds a prescribed threshold, so every retained pair has exactly K inputs and sufficient angular support.

Candidate filtering. Let $T_p = [R_p|t_p]$ denote the pose of target panorama p, where R_p is the camera-to-world rotation and t_p is the camera center. Let $T_i = [R_i|t_i]$ denote the pose of a DSLR frame i from the same scene. We first filter DSLR candidates by Euclidean distance between camera centers:

$$\mathcal{S}(p) = \Big\{ i : \ \|t_i - t_p\| \le d_{\max} \Big\}. \tag{15}$$

Viewing direction in the panorama frame. For each candidate $i \in \mathcal{S}(p)$, we use the DSLR camera forward axis as its viewing direction. With $e_z = [0, 0, 1]^\top$, the forward direction in world coordinates is

$$d_i^{\text{world}} = R_i e_z. \tag{16}$$

We then express this direction in the local coordinate frame of the target panorama:

$$d_i^p = R_p^\top d_i^{\text{world}}. \tag{17}$$

Writing $d_i^p = (x_i, y_i, z_i)^\top$, we convert it to azimuth and elevation as

$$\text{az}_i = \text{atan2}(x_i, z_i), \qquad \text{el}_i = \arcsin\left(\frac{y_i}{\|d_i^p\|_2} \right). \tag{18}$$

Angular coverage. We discretize the panorama domain into angular bins $\mathcal{B} = \{b\}$. Each bin b has a center $(\text{az}_b, \text{el}_b)$ and a non-negative weight w_b. We define a binary coverage indicator

$$\text{cov}(i, b) = \begin{cases} 1, & \text{if DSLR view } i \text{ covers angular bin } b, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

In practice, $\text{cov}(i, b) = 1$ when the bin center lies inside the DSLR camera field of view after projection from the panorama frame to the DSLR frame. Given a partial context set $\mathcal{C}$, the set of angular bins already covered by $\mathcal{C}$ is

$$\text{Covered}(\mathcal{C}) = \Big\{ b \in \mathcal{B} : \ \max_{i \in \mathcal{C}} \text{cov}(i, b) = 1 \Big\}. \tag{20}$$

The weighted coverage fraction is

$$\text{CovFrac}(\mathcal{C}) = \frac{\sum_{b \in \text{Covered}(\mathcal{C})} w_b}{\sum_{b \in \mathcal{B}} w_b}. \tag{21}$$

Greedy context selection. Starting from $\mathcal{C}_0 = \emptyset$, we add one DSLR image at a time. At step m, the next image is selected by maximizing the newly covered weighted area:

$$i_m^\star = \arg \max_{i \in \mathcal{S}(p) \setminus \mathcal{C}_{m-1}} \Delta(i \mid \mathcal{C}_{m-1}), \tag{22}$$

where the marginal gain is

$$\Delta(i \mid \mathcal{C}) = \sum_{b \in \mathcal{B}} w_b \, \mathbb{I}[\text{cov}(i, b) = 1 \ \wedge \ b \notin \text{Covered}(\mathcal{C})]. \tag{23}$$

We then update

$$\mathcal{C}_m = \mathcal{C}_{m-1} \cup \{ i_m^\star \}, \qquad m = 1, \ldots, K. \tag{24}$$

The resulting set $\mathcal{C}(p) = \mathcal{C}_K$ is used as the context input for target panorama p if it passes the final coverage threshold.

Solid-angle weighting. We use solid-angle weights so that uniformly spaced bins in $(\mathrm{az}, \mathrm{el})$ do not over-emphasize regions near the poles. For a bin b centered at elevation el_b, we therefore set

$$w_b = \max\big(0, \cos(\mathrm{el}_b)\big). \tag{25}$$

Since $\mathrm{CovFrac}$ in Eq. equation 21 normalizes by the total weight, the absolute scale of w_b does not affect the final coverage ratio.

Default hyper-parameters. The precomputed evaluation manifests use 5° bins in both azimuth and elevation. We set the panorama–DSLR center-distance threshold to $d_{\max} = 2.0\,\mathrm{m}$, select $K = 10$ context DSLR images per target panorama, and retain only targets whose final coverage satisfies $\mathrm{CovFrac}(\mathcal{C}(p)) > 0.8$. This filtering results in 36 valid scenes in total.

B Additional experiments

B.1 Harder camera transitions

We evaluate narrow-FoV perspective-to-panorama synthesis at 60° and 90° FoV. To provide sufficient coverage, the settings use 24 and 6 context views, respectively; methods share the same context images within each setting. Table 8 shows gains of 0.36 and 0.66 dB PSNR over the strongest baseline. The two settings differ in both FoV and context count, so their absolute scores do not isolate the effect of FoV.

Table 8: **Narrow-FoV perspective to panorama.** Context counts are matched across methods within each FoV.

FoV	Contexts	Method	PSNR ↑	SSIM ↑	LPIPS ↓
60°	24	UCPE-N	25.94	0.838	0.114
60°	24	UCPE-P	25.13	0.814	0.131
60°	24	Ours	**26.30**	**0.848**	**0.104**
90°	6	UCPE-N	23.79	0.782	0.151
90°	6	UCPE-P	23.66	0.769	0.171
90°	6	Ours	**24.45**	**0.798**	**0.140**

B.2 Higher-resolution synthesis

We further evaluated the performance of our method on higher-resolution images. We distinguish training at higher resolution from direct resolution transfer. With matched training at 448×672, the LVSM architecture, patch size 8, and 80K-step schedule are retained. Ours improves UCPE-C by 0.293 dB PSNR (Table 9). We also directly evaluate models trained at 224×336 at 448×672, without retraining or fine-tuning. Ours achieves the best PSNR and SSIM in this setting, while UCPE-C has the lowest LPIPS.

Table 9: **Higher-resolution Mixed evaluation protocols.** Both settings are evaluated at 448×672 with different training resolutions.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Training at 448×672			
UCPE-C	25.987	0.817	0.163
Ours	**26.280**	**0.819**	**0.157**
Direct transfer from 224×336 training			
UCPE-N	24.217	0.787	0.217
UCPE-P	24.100	0.786	0.228
UCPE-C	23.335	0.785	**0.214**
Ours	**24.499**	**0.788**	0.216

B.3 Multiple target views

We additionally evaluate the SVSM backbone with four target views per sample on 2,000 Mixed samples. Table 10 reports a 0.504 dB PSNR gain over CamRay, together with improved SSIM and LPIPS.

Table 10: **Four-target evaluation.** SVSM on 2,000 Mixed samples.

Raymap	PSNR ↑	SSIM ↑	LPIPS ↓
None	24.209	0.848	0.114
Plücker	23.963	0.845	0.118
CamRay	24.111	0.848	0.110
Ours	**24.615**	**0.852**	**0.104**

Table 11: **Cross-dataset evaluation.** Zero-shot transfer to 60 DL3DV cases.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
UCPE-N	16.266	0.4717	0.4762
UCPE-C	16.271	0.4707	**0.4731**
Ours	**16.500**	**0.4767**	0.4744

B.4 Cross-dataset evaluation

We directly evaluate models trained on ScanNet++ on 60 heterogeneous-camera cases from DL3DV [42], using synthetically resampled views under the Mixed protocol, without training or fine-tuning on DL3DV. Table 11 shows modest improvements in PSNR and SSIM, while UCPE-C has the lowest LPIPS. The low absolute scores reflect the difficulty of this cross-dataset setting.

C Failure cases

Figure 6 shows residual fine-structure errors on ScanNet++ and structural errors in two zero-shot DL3DV examples. Our method depends on the capabilities of the used backbones, therefore, even with calibrated camera rays, reconstruction still depends on the model's ability to infer scene depth and visibility from RGB images.

D More qualitative results

E Assets License

In Table 12, we list the licenses of all the existing assets including the code and data we have used in this work.

Asset	License / terms (linked)
LVSM [5]	CC BY-NC-SA 4.0
ScanNet++ v2 [12]	ScanNet++ Terms of Use (non-commercial research/education)
UCPE [10]	MIT
SVSM [41]	MIT
DL3DV [42]	CC BY-NC (as stated in the dataset license)

Table 12: **Licenses of existing assets we have used in this work.**

F Broader impacts

Heterogeneous-camera view synthesis may support scene visualization and robotic perception across different sensors. Synthesized views can contain incorrect details or be misused to misrepresent a scene. Outputs should thus be identified as synthesized and checked before consequential use; use and sharing of source data remain subject to their access and privacy restrictions.

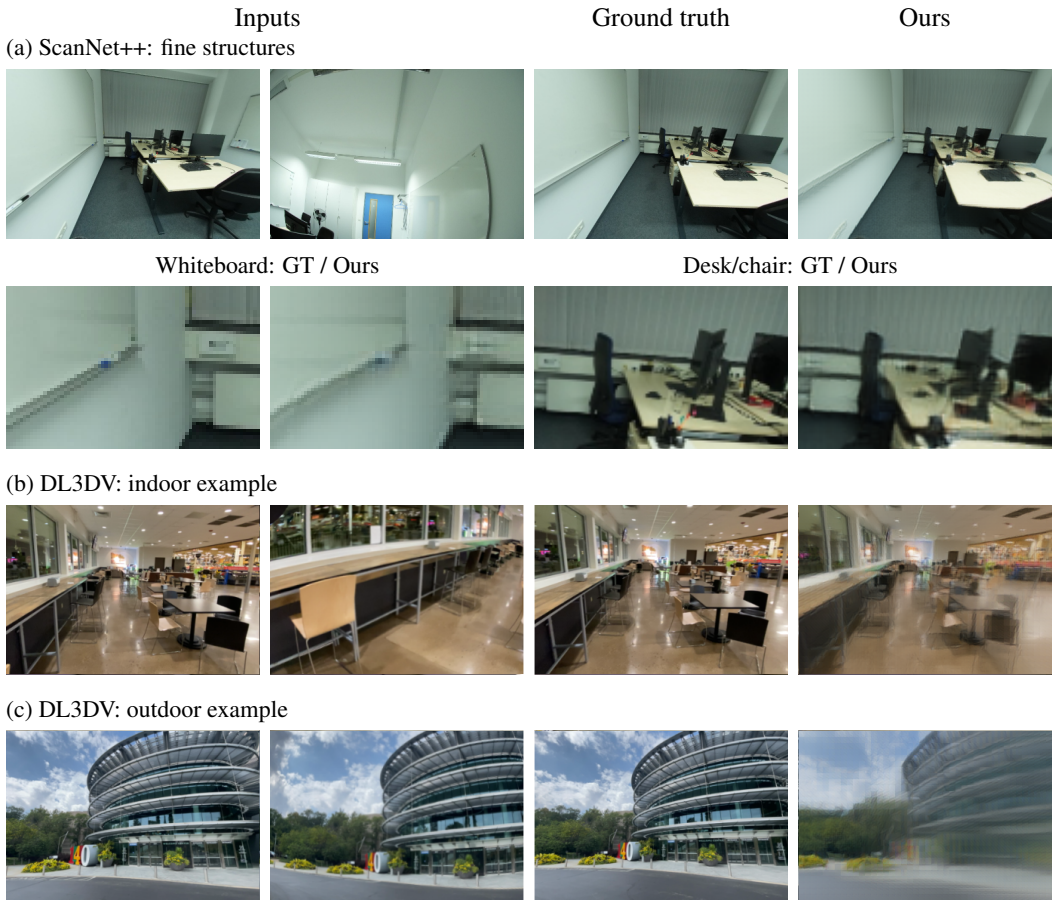


Figure 6: **Failure cases.** Two input context views, ground truth, and our predictions. (a) ScanNet++, with paired whiteboard and desk/chair crops below. (b,c) DL3DV, evaluated without retraining or fine-tuning.

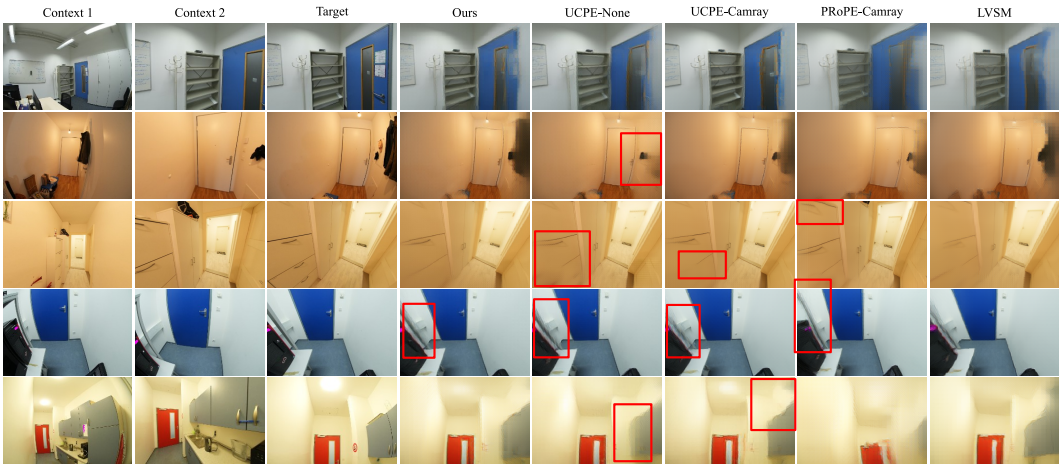


Figure 7: **More qualitative comparisons on mixed camera evaluation.**

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the heterogeneous-camera NVS setting, the local raymap and projection-aware RoPE contributions, and the claimed empirical gains. These claims are supported by the method and experiments in Sections 3 and 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in the conclusion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present theoretical results, but present emperical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 specifies the dataset and training protocals used in the experiments. More details about architectures and evaluation protocals are included in the supplementary.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The submission does not currently provide public codes. These will be released after publication. We use a publicly avaiable dataset.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The experimental setup and supplementary describe these details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The current experiments report metrics without error bars or statistical significance tests. The experiments are evaluated at sufficient samples of data.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports the compute setup, including two NVIDIA A100 GPUs, total batch size, and training iterations. Our methods aligned with academic resources and are not constrained by the CPU or memory.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research uses existing research datasets and evaluates view synthesis methods without human-subject experiments or unsafe data release. We are not aware of any deviation from the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk pretrained generative model or scraped dataset. The work evaluates a view-synthesis method on existing research data.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the datasets and prior methods used in the experiments and list license in the supplementary.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce or release a new dataset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The work does not involve crowdsourcing experiments or research with human subjects, so IRB approval or equivalent review is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not an important, original, or non-standard component of the core method or experiments.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.