IRIS: Implicit Rendering Matters for Pose-Free Novel View Synthesis

Wenyu Li[†], Sidun Liu[†], Peng Qiao[*], Yong Dou[*], and Tongrui Hu
National University of Defense Technology
Changsha, China

Abstract

Novel view synthesis from unposed multi-view images remains challenging, as the model must jointly learn scene representations and camera parameters without pose supervision. Existing approaches largely fall into two extremes: implicit latent-space rendering is flexible and easy to optimize, but often yields weakly grounded camera estimation; explicit 3D representations provide stronger geometric grounding, but introduce heavier parameterization and more fragile optimization. In this paper, we present IRIS, a fully self-supervised framework that provides a practical middle ground between these two paradigms. Instead of decoding free latent tokens or reconstructing fully explicit 3D primitives, IRIS represents the scene as a latent neural field and renders novel views by querying this field under self-predicted cameras. Specifically, projected features from reference views are aggregated at sampled 3D points to form point-wise latent features, which are then composed along target rays for rendering. This design preserves the flexibility and optimization stability of implicit modeling, while introducing stronger geometric structure than unconstrained latent rendering. Extensive experiments show that IRIS achieves strong novel view synthesis quality with competitive pose accuracy under fully self-supervised learning. Our project page: https://leo-frank.github.io/IRIS/.

CCS Concepts

• **Computing methodologies** → **Reconstruction**.

Keywords

Novel View Synthesis, 3D Reconstruction

ACM Reference Format:
Wenyu Li[†], Sidun Liu[†], Peng Qiao[*], Yong Dou[*], and Tongrui Hu. 2026. IRIS: Implicit Rendering Matters for Pose-Free Novel View Synthesis. In *Proceedings of the 34th ACM International Conference on Multimedia (MM '26), November 10–14, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3767308.3835537

1 Introduction

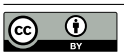
Novel view synthesis is a fundamental problem in computer vision and computer graphics, aiming to recover scene representation from limited observations and support high-quality rendering at target viewpoints. In recent years, this area has witnessed remarkable progress, ranging from radiance field representations such as NeRF[21], to explicit scene representations such as 3D Gaussian Splatting[16], and further to feed-forward models[7, 12, 13, 25, 32, 39] for view synthesis without scene-specific optimization.

Most of these methods rely on known or pre-estimated camera poses[4, 7, 13, 32], which directly affect the quality of target-view rendering. Their dependence on external camera acquisition still poses clear limitations. In practice, camera parameters are often obtained through Structure-from-Motion pipelines[27, 28], which not only increase computational cost and engineering complexity, but also become unstable in the presence of weak textures, heavy occlusions, or large viewpoint changes. Moreover, such dependence on external pose estimation limits the scalability of these methods to large collections of raw multi-view data. As a result, directly learning from unposed images, namely pose-free or self-calibrated novel view synthesis[11, 24], is becoming increasingly important.

Existing pose-free methods[9, 11, 41] have shown that it is feasible to jointly learn scene representations and camera parameters without pose supervision, but their rendering designs still largely fall into two extremes. The pioneering RayZer [11] adopts a flexible *implicit* latent-space renderer and is easier to optimize. However, its predicted cameras are more likely to degenerate into latent variables that merely support internal rendering compatibility, rather than corresponding to physically grounded geometry. Its follow-up, E-RayZer [41], obtains more physically-grounded camera spaces by representing the scene with *explicit* 3D Gaussians, but at the cost of lower rendering quality and more sensitive optimization. This raises a natural question: can one simultaneously retain the advantages of both sides, namely high rendering quality and more physically-grounded camera estimation?

Our key insight is that geometric constraints remain essential for self-supervised novel view synthesis. However, such constraints need not be introduced only through fully explicit 3D primitives. Instead, they can also arise from how an implicit scene representation is structured and rendered under self-predicted cameras. Based on this observation, we propose IRIS, which represents the scene as a latent neural field and renders novel views by querying this field under self-predicted cameras. Concretely, rather than decoding free latent tokens as in RayZer or reconstructing fully explicit 3D primitives as in E-RayZer, IRIS builds a latent neural field as the scene representation. Novel views are rendered by querying this field along target rays: at each sampled 3D point, projected features from reference views are aggregated into a point-wise latent feature, and these point-wise features are further composed along the ray to predict the target color. This design preserves the flexibility and trainability of implicit modeling, while introducing stronger geometric structure than unconstrained latent rendering.

[†] Equal contribution. [*] Corresponding authors.



MM '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2213-4/2026/11
https://doi.org/10.1145/3767308.3835537

Property	RayZer[11]	E-RayZer[41]	IRIS
Ordered-input NVS	●	◐	●
Unordered-input NVS	○	◐	●
Grounded camera	○	●	●
Trainability	●	○	●

Table 1: High-level comparison of RayZer, E-RayZer, and IRIS. ●, ◐, and ○ denote strong, moderate, and weak performance, respectively.

Experiments show that IRIS achieves strong novel view synthesis quality while maintaining competitive pose accuracy under the fully self-supervised setting. Compared with prior methods, IRIS is more robust to unordered inputs while avoiding the optimization difficulty of fully explicit 3D representations. Our main contributions can be summarized as follows: (1) We propose IRIS, a fully self-supervised framework for novel view synthesis from unposed multi-view images, based on a latent neural field representation and field querying under self-predicted cameras. (2) We present a field-based alternative to existing self-supervised rendering designs, which avoids both unconstrained latent decoding and fully explicit 3D primitives. (3) We show through experiments that IRIS achieves strong novel view synthesis quality and pose accuracy.

2 Related Work

In this section, we briefly review prior work on novel view synthesis (NVS). From the perspective of pose dependency, existing methods can be broadly categorized into three groups: **(1) Pose-Conditioned NVS.** These methods assume camera poses are available as input and synthesize novel views from posed images. **(2) Pose-Supervised NVS.** These methods do not require poses at inference time, but still rely on pose annotations or pose-related supervision during training. **(3) Fully Self-Supervised NVS.** These methods learn both camera pose estimation and novel view synthesis directly from unlabeled images, without using pose annotations in either training or inference.

Pose-Conditioned NVS. These methods assume camera poses are available at inference time and synthesize novel views from posed images. Representative examples include Neural Radiance Fields (NeRF [22]) and 3D Gaussian Splatting (3DGS [16]) However, both original methods require costly per-scene optimization, limiting practicality. To address this issue, later works developed generalizable pose-conditioned frameworks that amortize reconstruction and rendering across scenes. These methods can be broadly grouped into three categories (1) **Radiance-field-based methods** [4, 7, 8, 32, 34, 39] query radiance or latent properties at arbitrary 3D locations by combining spatial coordinates with image features from multiple source views, and perform ray-based rendering for target-view synthesis. (2) **3DGS-based methods** [3, 5, 6, 12, 40, 43] directly predict 3D Gaussian primitives as the scene representation and render novel views through differentiable splatting. (3) **Neural-network-only methods** [14, 26] remove explicit 3D inductive biases and instead learn a latent rendering function directly from data. Although these methods substantially reduce optimization cost and improve efficiency, their applicability is still limited by the requirement for accurate camera poses at test time. This limitation motivates later work that further relaxes pose dependency during training and inference.

Pose-Supervised NVS. A second line of research relaxes the reliance on camera poses at inference time, while still using pose supervision during training. Compared with pose-conditioned methods, these methods learn scene representations directly from 2D images instead of taking camera poses as input. Trained on large-scale posed image collections, such methods have shown promising performance in novel view synthesis. For example, LEAP [10] and PF-LRM [33] construct neural volumes in the canonical view coordinate system and perform neural rendering from the inferred representation. NoPoSplat [37] reconstructs scene Gaussians in the coordinate system of the first view directly from input images without requiring camera poses as input. Nevertheless, these methods still rely on ground-truth poses during training, which limits their scalability to fully unposed image collections.

Self-Supervised NVS. Fully self-supervised NVS aims to eliminate the reliance on camera poses during both training and inference, and instead nstead learns camera estimation and novel view synthesis. In practice, camera poses are often estimated using Structure-from-Motion (SfM) pipelines such as COLMAP [29], but these methods can be brittle under sparse-view or wide-baseline settings. Even when many views are available, incremental SfM may still discard difficult image regions by filtering feature correspondences that are inconsistent with the current reconstruction. With the differentiability of rendering methods, it is possible to optimize camera parameters jointly with scene representations. Early works such as NeRF– [35] show that camera poses can be estimated for forward-facing scenes by jointly optimizing poses and radiance fields from simple initialization, while BARF [19] improves optimization stability through a coarse-to-fine registration strategy. More recently, RayZer [11] removes explicit 3D inductive bias and adopts latent rendering, achieving strong synthesis quality for ordered image sequences. However, its learned pose space is only weakly grounded in physically interpretable 3D geometry, which limits its 3D awareness. E-RayZer [41] moves one step further by introducing explicit 3D Gaussian reconstruction into the self-supervised setting. While such explicit 3D representations offer stronger geometric grounding, they also make training substantially more difficult. To stabilize training, E-RayZer further adopts a fine-grained visual-overlap-based curriculum that starts from high-overlap samples and gradually expands to harder ones, while adaptively aligning heterogeneous data sources. Our method is also situated in the fully self-supervised setting, but differs from prior approaches by seeking a better balance between geometric grounding, rendering quality, and optimization stability: instead of relying on unconstrained latent rendering or explicit 3D Gaussian reconstruction alone, we learn a latent neural field to represent the scene.

3 Approach

In the fully self-supervised setting, our goal is to achieve strong novel view synthesis while learning camera parameters and scene representations directly from unlabeled multi-view images. We first revisit the two representative self-calibrated paradigms, namely RayZer[11] and E-RayZer[41], and clarify their key differences. We

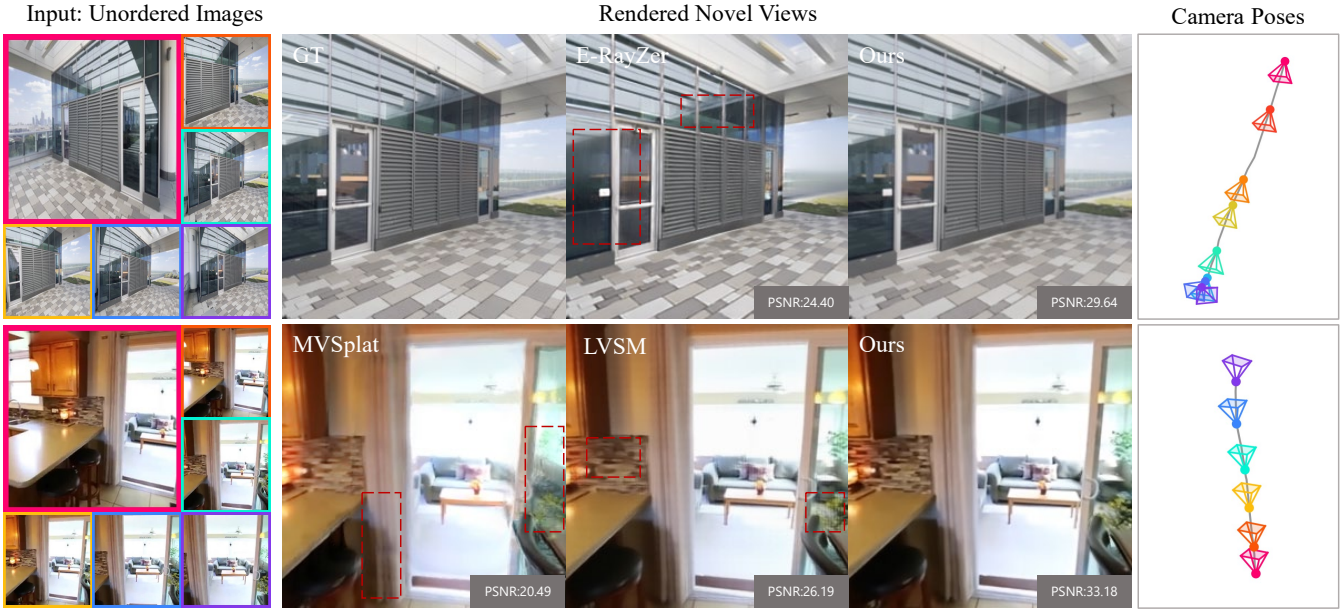


Figure 1: Given a set of unordered input images, our method predicts camera poses and synthesizes high-quality novel views, while being trained entirely on images without camera annotations. Compared with prior methods, our approach produces sharper and more faithful renderings. The rightmost panels visualize the predicted camera poses.

then present **IRIS**, which combines camera prediction with a latent field representation.

3.1 Preliminaries: Implicit or Explicit

Given a set of unposed multi-view images $\mathcal{I} = \{I_i\}_{i=1}^{V}$, following prior self-supervised learning frameworks, we split the input into two non-overlapping subsets: a reference set $\mathcal{I}_{\text{ref}}$ used for scene inference, and a target set $\mathcal{I}_{\text{tgt}}$ used for photometric supervision. Let $\mathcal{V}_{\text{ref}}$ and $\mathcal{V}_{\text{tgt}}$ denote the index sets of the reference and target views, respectively, with $V_{\text{ref}} = |\mathcal{V}_{\text{ref}}|$.

RayZer first patchifies all images into image tokens f, and introduces one learnable camera token per view, denoted by $p \in \mathbb{R}^{V\times d}$. A transformer-based camera estimator updates them jointly:

$$\{f^*, p^*\} = E_{\text{cam}}(\{f, p\}).$$

A canonical reference view c is selected, and the relative pose of each view i with respect to c is predicted as

$$\pi_i = \text{MLP}_{\text{pose}}([p_i^*, p_c^*]) \in \mathbb{R}^9, \qquad P_i = \Phi_{\text{SE(3)}}(\pi_i),$$

where π_i consists of a 6D rotation parameterization and a 3D translation and $P_i \in SE(3)$ denotes the camera pose. RayZer further predicts a shared focal length from the canonical camera token,

$$focal = \text{MLP}_{\text{focal}}(p_c^*),$$

which defines a shared intrinsic matrix K for all views, yielding the predicted camera parameters

$$\mathcal{P} = \{(P_i, K)\}_{i=1}^{V}.$$

Each predicted camera is converted into a pixel-aligned Plücker ray map

$$R_i^{\text{plk}} = \Pi_{\text{plk}}(P_i, K).$$

For the reference-view subset $\mathcal{I}_{\text{ref}}$, RayZer linearly tokenizes the predicted Plücker ray maps into ray tokens r_{ref}, fuses them with the corresponding image tokens f_{ref},

$$x_{\text{ref}} = \text{MLP}_{\text{fuse}}([f_{\text{ref}}, r_{\text{ref}}]),$$

and predicts the latent scene representation from learnable scene tokens z via

$$\{z^*, x_{\text{ref}}^*\} = E_{\text{scene}}(\{z, x_{\text{ref}}\}), \qquad z_{\text{scene}} = z^*.$$

Finally, the target views are rendered from the latent scene representation and the target-view ray tokens,

$$\hat{I}_{\text{tgt}} = f_\phi^{\text{rend}}(z_{\text{scene}}, r_{\text{tgt}}),$$

and the model is trained with photometric supervision on target views:

$$\mathcal{L} = \sum_{(I,\hat{I})\in(\mathcal{I}_{\text{tgt}}, \hat{\mathcal{I}}_{\text{tgt}})} \Big(\text{MSE}(\hat{I}, I) + \lambda_{\text{perc}}\,\text{Percep}(\hat{I}, I)\Big).$$

E-RayZer follows the same overall pose-first formulation as RayZer, but replaces RayZer's implicit latent scene representation with *explicit* 3D Gaussians [16]. Using the same fused image–ray tokens x_{ref} as in RayZer, E-RayZer predicts the Gaussian scene as

$$\mathcal{G}_{\text{ref}} = \big(f_\omega^{\text{gauss}} \circ E_{\text{scene}}\big)(x_{\text{ref}})$$

Here, $E_{\text{scene}}(x_{\text{ref}})$ denotes the scene encoding stage that performs multi-view aggregation and produces updated latent tokens, while $f_\omega^{\text{gauss}}(\cdot)$ is a lightweight Gaussian decoder that maps these tokens to per-pixel Gaussian parameters $\{g_i = (d_i, q_i, C_i, s_i, \alpha_i)\}_{i=1}^{V_{\text{ref}}HW}$ along the reference-view rays. Each Gaussian g_i is parameterized by distance along the ray d_i, orientation q_i, spherical-harmonic

color coefficients C_i, scale s_i, and opacity α_i. The target views are then rendered using the predicted target-view cameras

$$P_{\text{tgt}} = \{(K, P_i) \mid i \in \mathcal{V}_{\text{tgt}}\}, \qquad \hat{I}_{\text{tgt}} = \pi(\mathcal{G}_{\text{ref}}, P_{\text{tgt}})$$

where $\pi(\cdot)$ denotes differentiable Gaussian splatting.

Comparison and Motivation. RayZer benefits from flexible latent-space rendering and favorable optimization, but because camera prediction, scene inference, and rendering are jointly learned in *latent space*, the predicted cameras mainly need to remain *mutually compatible* with the internal renderer. As a result, the learned camera space is not necessarily physically grounded, and may partially degenerate into a latent variable that mainly serves rendering compatibility. By replacing latent rendering with *explicit* 3D Gaussians and differentiable splatting, E-RayZer achieves a more geometrically grounded camera space. This improvement, however, comes at the cost of a heavier representation, a more sensitive optimization process, and slightly weaker rendering quality. These two paradigms therefore expose a clear trade-off between optimization flexibility and geometric grounding. Our goal is to preserve the optimization advantages of latent scene modeling, while replacing unconstrained latent decoding with a rendering interface that is more strongly constrained by multi-view geometric consistency.

3.2 IRIS Model

Our Insight. The above comparison suggests that some form of geometric inductive bias remains essential for self-supervised novel view synthesis. Without it, camera prediction and scene representation may drift toward a latent solution that is internally compatible with the renderer, but not physically grounded. At the same time, simply adopting fully explicit 3D representations is not ideal either, since it introduces a heavier representation and a more fragile optimization process. Our key idea is therefore to introduce geometric structure in a lighter-weight manner: instead of relying on unconstrained latent decoding as in RayZer, or fully explicit 3D primitives as in E-RayZer, IRIS represents the scene as a *latent field*, which is queried and rendered under self-predicted cameras.

Shared Multi-View Encoder. Following RayZer and E-RayZer, IRIS adopts a *pose-first* formulation: camera parameters are predicted first and then used to define the scene for rendering. Unlike RayZer, however, IRIS does not use a dedicated camera estimator followed by a separate scene reconstructor conditioned on fused image–ray tokens. Instead, we use a *single shared multi-view encoder* to jointly produce per-view visual features and camera tokens, and directly retain the encoded view features for the subsequent latent-field representation. We instantiate the shared encoder with Muskie [18], since Muskie is a backbone designed for multi-view inputs that processes all input views jointly rather than encoding each frame independently. Given the input images $\mathcal{I} = \{I_i\}_{i=1}^{V}$ we tokenize each view into patch tokens and prepend one learnable camera token p_i to each view. The resulting multi-view tokens are processed jointly by the shared encoder:

$$\{f_i,\ p_i^*\}_{i=1}^{V} = E_\theta^{\text{enc}}(\mathcal{I}),$$

where $f_i \in \mathbb{R}^{hw \times d}$ denotes the encoded feature map of the i-th view, and $p_i^* \in \mathbb{R}^d$ denotes its updated camera token. A camera head then predicts camera parameters in the same way as RayZer: a canonical reference view c is selected, each view pose is regressed relative to c from the updated camera tokens, and a single shared intrinsic matrix K is predicted for all views from the canonical token. We denote the resulting camera set by $\mathcal{P} = \{(P_i, K)\}_{i=1}^{V}$.

Scene Representation. Unlike RayZer and E-RayZer, IRIS does *not* reconstruct a separate set of latent scene tokens or explicit 3D primitives. Instead, IRIS directly preserves the reference-view features produced by the shared encoder and uses them, together with the self-predicted cameras, as a latent scene memory. Formally, we denote this memory by

$$\mathcal{S} = \{(f_k, K, P_k)\}_{k \in \mathcal{V}_{\text{ref}}}$$

This scene memory induces a latent field, whose value at a 3D query point x is obtained by projecting x into all reference views under the predicted cameras, sampling the corresponding view-wise features, and aggregating them across views. Concretely, for each reference view k, we first obtain

$$v_k(x) = \Pi_k(x; f_k, P_k, K),$$

where $\Pi_k(\cdot)$ denotes projection followed by feature sampling from the k-th reference-view feature map. We then define the latent feature of x as

$$\tilde{f}(x) = F_{\mathcal{S}}(x) = \mathcal{T}_{\text{view}}\big(v_1(x), \ldots, v_{V_{\text{ref}}}(x)\big),$$

where $\mathcal{T}_{\text{view}}$ is a transformer that fuses the projected features from all reference views into a single point-wise latent feature. In this sense, $F_{\mathcal{S}}$ is a latent field induced on the fly from multi-view features under the self-predicted cameras.

Rendering. Given a target ray $r = (o_r, d_r)$, we sample M points along the ray,

$$x_m = o_r + z_m d_r, \qquad m = 1, \ldots, M,$$

and query the latent field at each sampled point:

$$\tilde{f}_m = F_{\mathcal{S}}(x_m).$$

We then compose the resulting point-wise features along the ray with a second transformer,

$$h_r = \mathcal{T}_{\text{ray}}(\tilde{f}_1, \ldots, \tilde{f}_M),$$

where $\mathcal{T}_{\text{ray}}$ is a transformer that aggregates the sampled point features into a ray-wise feature. Unlike classical volumetric rendering, which predicts explicit color and density for each sample and combines them with a fixed rendering rule[22], $\mathcal{T}_{\text{ray}}$ learns how the queried latent features should be composed to form the final ray representation. The rendered color is then predicted as

$$\hat{C}(r) = f_{\text{rgb}}(h_r).$$

We train IRIS with self-supervised photometric loss on target rays:

$$\mathcal{L} = \sum_{r \in \mathcal{R}_{\text{tgt}}} \|\hat{C}(r) - C(r)\|_2^2.$$

Since both latent-field querying and ray-wise rendering are conditioned on the self-predicted cameras, this objective jointly supervises camera estimation, scene representation learning, and rendering in a fully self-supervised manner.

Overall, IRIS can be viewed as a middle ground between RayZer and E-RayZer. Unlike RayZer, it does not rely on unconstrained latent decoding for rendering; unlike E-RayZer, it avoids fully explicit 3D primitives and their associated optimization difficulty. Instead, IRIS introduces geometric structure by constraining how the latent

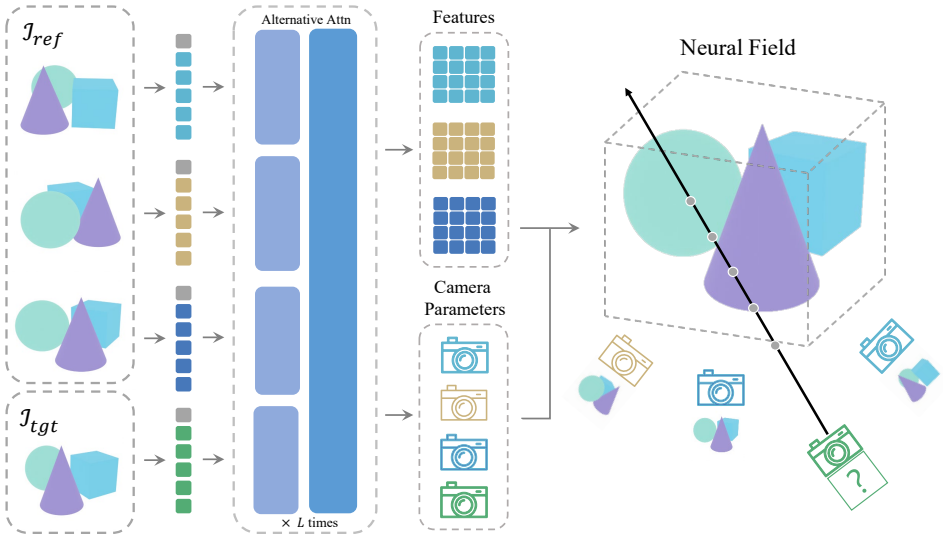


Figure 2: Overview of our method. Given unposed multi-view images, IRIS first predicts camera parameters and extracts latent reference features with a shared multi-view encoder, from which a latent neural field is constructed. Novel views are rendered by querying this field along target rays under the self-predicted cameras, where point-wise features are aggregated across reference views and then composed along each ray for color prediction.

scene representation is queried and rendered under self-predicted cameras.

4 Experiments

4.1 Experimental Setup

We report evaluation results for quality of novel view synthesis and pose estimation on several datasets to demonstrate the effectiveness of our method.

Implementation Details. We use Muskie[18] as the visual backbone and fine-tune it during training. For rendering, we sample 64 points within a fixed depth range along each ray. During training, we randomly sample a subset of rays for efficiency, while at inference time full-resolution images are rendered in chunks. We optimize the model with AdamW and a base learning rate of 4×10^{-5}, together with a warmup schedule. The encoder and pose predictor are trained with a 0.1× learning rate, while the renderer uses the full rate. Training is conducted on 8 A100 GPUs. In practice, training on DL3DV takes roughly 7 days, while training on the mixed-dataset setting takes about 14 days.

Training and Testing Data. We report results under both **single-dataset** and **multi-dataset** training settings. For single-dataset training, models are trained exclusively on Re10K [42] or DL3DV [20]. For multi-dataset training, we train on a mixture of Re10K [42], DL3DV [20], CO3Dv2 [23], ARKitScenes [2], and WildRGBD [36], covering a diverse set of indoor and outdoor scenes. This training setup is comparable to that of E-RayZer[41] in terms of both dataset scale and diversity. For evaluation, we mainly benchmark both pose estimation and novel view synthesis on the DL3DV [20] test set, NRGBD [1], 7Scenes [30], and ScanNet++ [38]. Each test sample contains 24 frames, with 16 used as reference and the remaining 8 as target views. We evaluate pose estimation over all frames, while NVS quality is reported only on the 8 target views. To better reflect real-world usage, our evaluation primarily uses randomly sampled input images that do not assume any fixed temporal or pose ordering; we denote this setting as **Rand**. In addition, since we find that RayZer is only applicable to ordered image sequences due to its internal image index embeddings, we also report results on ordered input sequences, denoted as **Seq**, to enable a more complete comparison with RayZer.

Baselines. We compare with baselines from three categories: pose-conditioned methods (MVSplat [5] and LVSM [13]), pose-supervised methods (NoPoSplat [37]), and self-supervised methods (SelfSplat [15], SPFSplat [9], RayZer [11], and E-RayZer [41]). We do not include comparisons with PF-LRM[33], since its official implementation is not publicly available. We note that NoPoSplat and SelfSplat are primarily designed for two-view input and show limited scalability to the multi-view setting. Moreover, among the self-supervised baselines, only RayZer and E-RayZer have been trained at relatively large scale—RayZer on DL3DV, and E-RayZer on an even broader mixture of datasets—whereas other methods (e.g., SPFSplat) are mainly pretrained on Re10K [42], whose camera trajectories are considerably less diverse. Therefore, we regard RayZer and E-RayZer as the primary baselines in our comparisons.

Evaluation Metrics. We evaluate novel view synthesis using the standard metrics PSNR, SSIM, and LPIPS. For pose estimation, we report relative pose accuracy (RPA) at thresholds of 5°, 15°, and 30°, where the relative pose error is defined as the maximum of the rotation error and the translation-direction error between the predicted and ground-truth relative poses.

4.2 Results

Main Results. We first evaluate our method under the **single-dataset** training setting, where the model is trained on a single

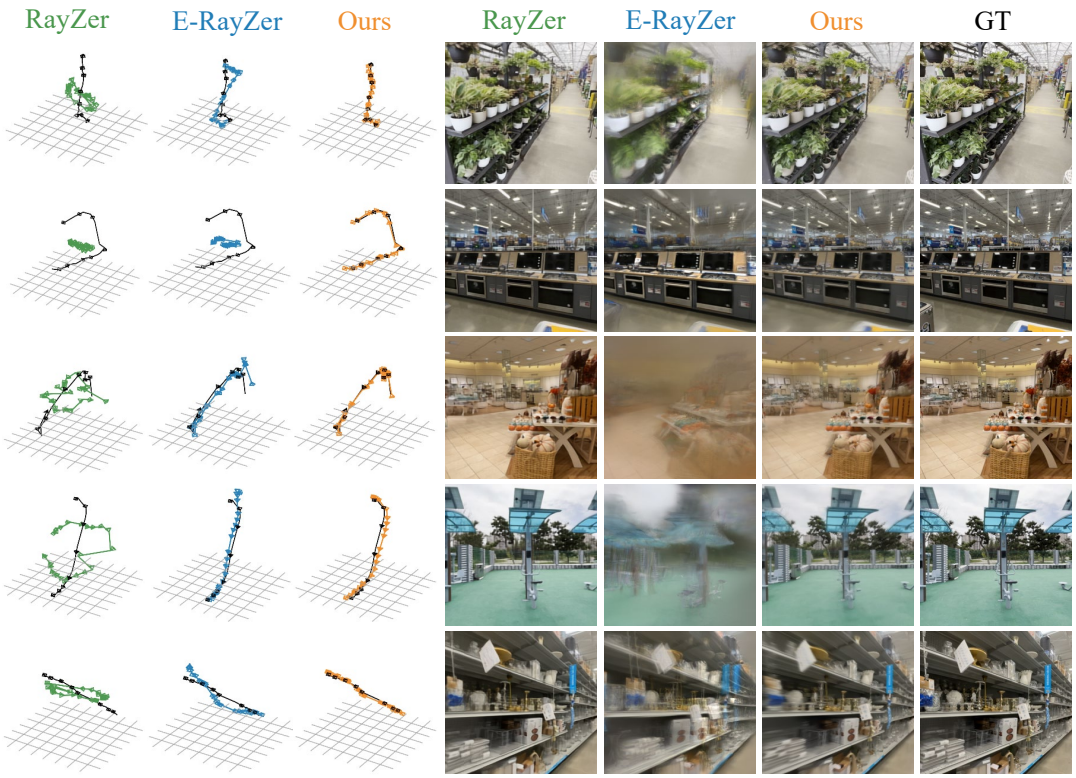


Figure 3: Qualitative comparisons of pose estimation and rendering quality. We visualize all poses of the predicted trajectory as colored frustums, while for the COLMAP trajectory we display only every third pose as a black frustum to reduce overlap.

Dataset	Method	Self-supervised?	Pose Accuracy			Novel View Synthesis		
			@5° ↑	@15° ↑	@30° ↑	PSNR↑	LPIPS↓	SSIM↑
Re10K	SPFSplat	✗(MASt3R[17])	0.673	0.928	0.961	24.418	0.148	0.788
	Ours	✓	0.590	0.884	0.997	31.272	0.117	0.920

Table 2: Comparison under the single-source training setting on Re10K[42].

Dataset	Method	Pose Accuracy			Novel View Synthesis		
		@5° ↑	@15° ↑	@30° ↑	PSNR↑	LPIPS↓	SSIM↑
DL3DV[20] (Seq.)	RayZer	0.000	0.010	0.089	26.286	0.169	0.814
	E-RayZer	0.658	0.824	0.902	21.187	0.272	0.701
	Ours	0.600	0.869	0.918	25.402	0.215	0.791
DL3DV[20] (Rand.)	RayZer	*Fail*					
	E-RayZer	0.656	0.811	0.895	21.124	0.271	0.701
	Ours	0.563	0.863	0.924	25.324	0.215	0.791
NRGBD[1]	E-RayZer	0.362	0.800	0.916	26.476	0.144	0.860
	Ours	0.437	0.861	0.979	31.404	0.128	0.921
ScanNet++[38]	E-RayZer	0.019	0.252	0.545	21.992	0.259	0.744
	Ours	0.036	0.333	0.627	22.938	0.276	0.763
7Scenes[30]	E-RayZer	0.312	0.729	0.882	26.727	0.175	0.870
	Ours	0.278	0.749	0.856	30.783	0.140	0.907

Table 3: Comparison under the single-source training setting on DL3DV[20]. RayZer fails when testing with random-order views.

dataset. We begin with Re10K, a widely used benchmark with relatively regular camera trajectories. As shown in Tab. 2 and Fig. 4, compared with SPFSplat, our method yields weaker pose accuracy but substantially better novel view synthesis quality. We attribute the pose gap to the strong MASt3R-based[17] initialization used by SPFSplat, which introduces additional geometric priors through correspondence-based supervision and therefore makes the comparison less directly aligned with a fully self-supervised setting. We do not include RayZer[11] and E-RayZer[41] in this comparison, since pretrained weights on Re10K are not publicly available for these methods.

We next evaluate on DL3DV [20], which provides a more challenging benchmark due to its greater scene diversity and more varied camera motion. As shown in the first two rows of Tab. 3, compared with E-RayZer, our method achieves overall comparable pose accuracy while improving novel view synthesis quality by a clear margin. This suggests that our learned representation is more effective for rendering under self-predicted cameras. We further provide qualitative comparisons in Fig. 3. We take E-RayZer as the primary comparison baseline on this benchmark, since it is the most relevant large-scale self-supervised method and is explicitly designed to address the geometric limitations of RayZer. It is also worth noting that RayZer shows a clear dependence on sequence regularity, as its image-index embeddings provide a strong cue for shortcut learning via frame interpolation. Consistent with this observation, RayZer performs reasonably well under the sequential setting, but fails when the views are randomly ordered.

Beyond the in-domain comparison on DL3DV, we further study whether the learned representation can generalize to unseen datasets.

Figure 4: Qualitative comparisons with SPFSplat[9] on Re10K[42]. Our method produces sharper and more structurally consistent renderings, achieving lower LPIPS.

The cross-dataset results in the remaining rows of Tab. 3 show that our method consistently demonstrates stronger generalization in novel view synthesis than E-RayZer, while maintaining broadly comparable pose accuracy. Notably, these evaluation image sets do not assume any fixed temporal or pose ordering, but are instead constructed to better reflect realistic multi-view captures. Since RayZer relies on ordered inputs and is therefore not suitable for this unordered cross-dataset setting, we do not treat it as a primary comparison method in the following evaluations.

Scaling to more training data. We further investigate whether enlarging the training distribution with mixed-source data can improve cross-dataset generalization. Comparing Tab. 4 with the DL3DV-only results in Tab. 3, we find that naively training on the mixed-source set does not consistently outperform single-source training for either E-RayZer or our method. Although our method still maintains clear advantages over E-RayZer in novel view synthesis on most benchmarks, the gains brought by mixed-source training are not stable, and in several cases the performance is even weaker than that of the DL3DV-only model. These observations suggest that simply increasing training data diversity does not automatically lead to better cross-dataset transfer for pose-free reconstruction and synthesis. A likely reason is that the mixed setting introduces substantially larger variations in scene statistics, camera motion patterns, and image distributions across datasets, making it harder for the model to learn a unified representation that generalizes well across domains. Overall, while our method remains competitive under mixed-source training, robust generalization across heterogeneous data sources likely requires more dedicated mechanisms than straightforward dataset mixing alone.

Comparison with supervised methods. We further compare our method with pose-supervised baselines on Re10K. As shown in Tab. 5, both LVSM [13] and MVSplat [5] rely on camera poses during both training and inference, whereas our method is trained without pose annotations and does not require camera poses at test time. Despite this weaker supervision, our method achieves the best PSNR and SSIM by a clear margin, outperforming both supervised

Dataset	Method	Pose Accuracy			Novel View Synthesis		
		@5° ↑	@15° ↑	@30° ↑	PSNR↑	LPIPS↓	SSIM↑
DL3DV[20]	E-RayZer	0.551	0.766	0.861	20.450	0.305	0.666
	Ours	0.514	0.824	0.893	24.042	0.237	0.763
NRGBD[1]	E-RayZer	0.280	0.782	0.913	25.837	0.173	0.842
	Ours	0.360	0.885	0.984	30.879	0.114	0.922
ScanNet++[38]	E-RayZer	0.011	0.262	0.545	21.950	0.267	0.743
	Ours	0.033	0.284	0.588	22.191	0.293	0.749
7Scenes[30]	E-RayZer	0.323	0.696	0.966	26.778	0.184	0.864
	Ours	0.296	0.717	0.791	28.767	0.157	0.898

Table 4: Comparison under the mixed-source training setting.

Figure 5: Qualitative comparison with supervised methods LVSM[13] and MVSplat[5].

Method	Training Supervision	Inference w. Camera Poses	PSNR↑	SSIM↑	LPIPS↓
LVSM[13]	2D + Camera	✓	27.603	0.866	0.110
MVSplat[5]	2D + Camera	✓	20.257	0.747	0.241
Ours	2D	✗	31.272	0.920	0.117

Table 5: Comparison with pose-supervised baselines on Re10K[42].

baselines in reconstruction fidelity. Compared with LVSM, our method still achieves the best PSNR/SSIM and competitive LPIPS, indicating that it can learn an effective scene representation for high-quality novel view synthesis without explicit pose supervision. These results suggest that strong multi-view rendering quality can emerge from self-supervised learning alone, without relying on posed inputs or camera labels.

4.3 Ablation studies

Backbone initialization. Tab. 6 shows that backbone initialization is critical for both pose estimation and novel view synthesis. Initializing from Muskie[18] yields substantially stronger performance than random initialization across all metrics, indicating that the gain comes from a stronger multi-view prior. This choice is motivated by the fact that Muskie is encouraged to discover cross-view correspondences and to produce view-consistent features with stronger geometric awareness. Such a property is particularly suitable for our setting: the backbone is expected not only to extract per-view appearance features, but also to provide a feature space in which information from different views can be reliably aligned and aggregated. This initialization plays an important practical role in stabilizing optimization and improving reconstruction quality.

Initialization	Frozen?	Pose Accuracy			Novel View Synthesis		
		@5° ↑	@15° ↑	@30° ↑	PSNR↑	LPIPS↓	SSIM↑
Random	✗	0.197	0.459	0.589	22.851	0.300	0.707
DINOv3[31]	✗	0.014	0.151	0.322	11.523	0.561	0.200
Muskie[18]	✓	0.002	0.038	0.167	15.060	0.563	0.284
Muskie[18]	✗	0.563	0.863	0.924	25.324	0.215	0.791

Table 6: Ablation on backbone initialization, evaluated on DL3DV [20].

Method	Pose Accuracy			Novel View Synthesis		
	@5° ↑	@15° ↑	@30° ↑	PSNR↑	LPIPS↓	SSIM↑
3DGS	*Fail*					
Volume Rendering	0.494	0.829	0.990	28.058	0.176	0.868
Ours	0.497	0.877	0.995	31.272	0.117	0.920

Table 7: Ablation on the renderer design, evaluated on Re10K[42].

Train Setting	Method	Self-supervised ?	Novel View Synthesis		
			PSNR↑	LPIPS↓	SSIM↑
Re10K (2-view)	MVSplat[5]	✗	22.720	0.168	0.823
	LVSM[13]	✗	27.580	0.120	0.895
	NoPoSplat[37]	✗	22.865	0.178	0.770
	SPFSplat[9]	✗	22.851	0.171	0.771
	SelfSplat[15]	✓	20.747	0.253	0.748
	Ours	✓	23.448	0.259	0.782
DL3DV (multi-view)	E-RayZer[41]	✓	14.899	0.287	0.703
	Ours	✓	23.708	0.251	0.794

Table 8: Ablation in the sparse-view case. Pose accuracy is omitted since it is not required in this setting. Methods such as MVSplat, LVSM, NoPoSplat, and SPFSplat rely on pose supervision or external supervised initialization.

Besides, freezing the Muskie backbone causes performance to collapse, showing that effective initialization alone is insufficient and that adapting the backbone to the target task is essential. We also find that DINOv3 performs poorly in this setting. A likely reason is that DINOv3 extracts features on a per-frame basis and lacks explicit cross-view interaction, making its features less suitable for regressing camera parameters in multi-view reconstruction.

Renderer. Tab. 7 studies the effect of renderer design. Replacing our implicit renderer with a standard volume-rendering formulation still yields reasonable pose accuracy, suggesting that the model can recover useful geometric cues under this design. However, the rendering quality drops consistently across all NVS metrics, indicating that our implicit renderer provides a more effective and flexible decoding interface for high-fidelity view synthesis. We further replace the scene representation with a 3DGS-style renderer, but observe training failure even with Muskie-based initialization. This observation is consistent with our findings on RayZer[11] and E-RayZer[41]: although careful data scheduling can partially alleviate the issue, optimizing an explicit 3DGS representation in a fully self-supervised setting remains highly unstable.

4.4 Depth Visualization

Although our renderer does not explicitly predict per-sample density as in classical volume rendering, we can extract a soft depth map

Figure 6: Rendered images and corresponding depth maps

from the ray aggregation module. For a target ray r, let $\{z_m\}_{m=1}^{M}$ denote the sampled depths and $\{w_m(r)\}_{m=1}^{M}$ the per-sample weights returned by the final ray-transformer block. These weights are obtained from the attention distribution of the last ray-aggregation layer and treated as a soft importance distribution. We compute

$$D(r) = \sum_{m=1}^{M} w_m(r) z_m.$$

Applying this computation to all target rays yields the relative depth map in Fig. 6. Since training is pose-free and uses a normalized ray-sampling range, this depth should be interpreted as relative soft depth rather than metric depth.

Sparse-view case. Tab. 8 reports results on Re10K with only two input views. In this highly sparse setting, methods that rely on pose supervision or external supervised initialization, such as LVSM[13], MVSplat[5], NoPoSplat[37], and SPFSplat[9], remain strong baselines, with LVSM achieving the best overall performance. Among the self-supervised methods trained directly in the 2-view regime, our method consistently outperforms SelfSplat[15] and achieves the strongest PSNR and SSIM, although there is still a gap to the best pose-supervised model. This suggests that the extreme two-view setting remains challenging for fully self-supervised reconstruction. More importantly, when trained with multi-view inputs and tested with only two views, our method clearly surpasses E-RayZer[41] by a large margin. This indicates that the scene representation learned from multi-view training transfers more effectively to sparse-view inference, and retains strong rendering capability even when the number of available input views is drastically reduced.

5 Conclusion

We presented IRIS, a fully self-supervised framework for novel view synthesis from unposed multi-view images. IRIS represents the scene as a latent neural field queried and rendered under self-predicted cameras, providing a practical middle ground between flexible latent modeling and geometrically grounded 3D reasoning. Experiments demonstrate strong rendering quality, competitive pose accuracy, and favorable generalization. We hope this work offers useful insights to the community on designing future self-supervised 3D vision models.

Method	Scene modeling	Rendering
RayZer	$z_{\text{scene}} = f_{\psi}^{\text{scene}}\left(z_0^{\text{scene}}, x_{\text{ref}}\right)$	$\hat{I}_{\text{tgt}} = f_{\phi}^{\text{rend}}\left(z_{\text{scene}}, \text{Linear}\left(R_{\text{tgt}}^{\text{plk}}\right)\right)$
E-RayZer	$\mathcal{G}_{\text{ref}} = f_{\omega}^{\text{gauss}} \circ f_{\psi'}^{\text{scene}}(x_{\text{ref}})$	$\hat{I}_{\text{tgt}} = \pi\left(\mathcal{G}_{\text{ref}}, P_{\text{tgt}}\right)$
IRIS	$\tilde{f}(x) = \mathcal{T}_{\text{view}}\left(v_1(x), \ldots, v_{V_{\text{ref}}}(x)\right)$	$\hat{C}(r) = f_{\text{rgb}} \circ \mathcal{T}_{\text{ray}}\left(\tilde{f}_1, \ldots, \tilde{f}_M\right)$

Table 9: Comparison of RayZer, E-RayZer, and IRIS in scene modeling and rendering.

6 Appendix

Definition of Pose Error. Given a set of predicted and ground-truth camera-to-world poses, $\{\hat{\mathbf{T}}_i^{\text{c2w}}\}_{i=1}^B$ and $\{\mathbf{T}_i^{\text{c2w}}\}_{i=1}^B$, we first convert them to world-to-camera poses:

$$\hat{\mathbf{T}}_i^{\text{w2c}} = (\hat{\mathbf{T}}_i^{\text{c2w}})^{-1}, \qquad \mathbf{T}_i^{\text{w2c}} = (\mathbf{T}_i^{\text{c2w}})^{-1}.$$

We then evaluate all unordered view pairs

$$\mathcal{P} = \{(i, j) \mid 1 \le i < j \le B\}.$$

For each pair $(i, j) \in \mathcal{P}$, the relative pose is defined as

$$\hat{\mathbf{T}}_{i\leftarrow j} = \hat{\mathbf{T}}_i^{\text{w2c}} \left(\hat{\mathbf{T}}_j^{\text{w2c}}\right)^{-1}, \qquad \mathbf{T}_{i\leftarrow j} = \mathbf{T}_i^{\text{w2c}} \left(\mathbf{T}_j^{\text{w2c}}\right)^{-1}.$$

Let

$$\hat{\mathbf{T}}_{i\leftarrow j} = \begin{bmatrix} \hat{\mathbf{R}}_{i\leftarrow j} & \hat{\mathbf{t}}_{i\leftarrow j} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \qquad \mathbf{T}_{i\leftarrow j} = \begin{bmatrix} \mathbf{R}_{i\leftarrow j} & \mathbf{t}_{i\leftarrow j} \\ \mathbf{0}^\top & 1 \end{bmatrix},$$

where $\hat{\mathbf{R}}_{i\leftarrow j}, \mathbf{R}_{i\leftarrow j} \in SO(3)$ and $\hat{\mathbf{t}}_{i\leftarrow j}, \mathbf{t}_{i\leftarrow j} \in \mathbb{R}^3$.

The rotation error is measured by the geodesic angle between the predicted and ground-truth relative rotations:

$$e_R(i, j) = \arccos\left(\frac{\text{Tr}\left(\mathbf{R}_{i\leftarrow j}^\top \hat{\mathbf{R}}_{i\leftarrow j}\right) - 1}{2}\right) \cdot \frac{180}{\pi}.$$

For translation, we only evaluate the direction and ignore the global scale. Let

$$c_t(i, j) = \frac{\mathbf{t}_{i\leftarrow j}^\top \hat{\mathbf{t}}_{i\leftarrow j}}{\|\mathbf{t}_{i\leftarrow j}\|_2 \, \|\hat{\mathbf{t}}_{i\leftarrow j}\|_2}.$$

Since the relative translation is treated as sign-ambiguous in our implementation, the translation-direction error is defined as

$$e_t(i, j) = \min\left(\arccos(c_t(i, j)), \, \pi - \arccos(c_t(i, j))\right) \cdot \frac{180}{\pi}.$$

Equivalently, this can be viewed as the unsigned angular error between the two translation directions.

Following the implementation, the **relative pose error** for each pair is defined as the maximum of the rotation and translation-direction errors:

$$e_{\text{pose}}(i, j) = \max\left(e_R(i, j), \, e_t(i, j)\right).$$

Given an angular threshold $\delta \in \{5^\circ, 15^\circ, 30^\circ\}$, the relative pose accuracy (RPA) is computed as

$$\text{RPA@}\delta = \frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \mathbf{1}\left[e_{\text{pose}}(i, j) \le \delta\right],$$

where $\mathbf{1}[\cdot]$ denotes the indicator function.

Compare with standard volume rendering. Both standard volume rendering and our method can be written under a unified ray-decoding form:

$$\hat{C}(r) = \mathcal{R}\left(\{g(x_m, r)\}_{m=1}^M\right), \qquad x_m = o_r + z_m d_r,$$

where $r = (o_r, d_r)$ is a target ray, $\{x_m\}_{m=1}^M$ are the sampled 3D points along the ray, $g(x_m, r)$ denotes the per-sample representation to be composed, and $\mathcal{R}(\cdot)$ denotes the ray-wise composition rule.

For standard volume rendering, the scene is parameterized by explicit radiance-field quantities:

$$g_{\text{vol}}(x_m, r) = (\sigma_m, c_m), \qquad (\sigma_m, c_m) = F_{\text{vol}}(x_m, d_r),$$

where σ_m and c_m denote the density and color at the m-th sample. The final pixel color is obtained by a fixed hand-crafted compositing rule:

$$\hat{C}_{\text{vol}}(r) = \sum_{m=1}^{M} T_m \alpha_m c_m,$$

with

$$\alpha_m = 1 - \exp(-\sigma_m \delta_m), \qquad T_m = \prod_{n=1}^{m-1} (1 - \alpha_n).$$

In contrast, our method does not predict explicit density and color at each sampled point. Instead, it constructs a latent field from multi-view features and uses the queried latent feature as the per-sample representation:

$$g_{\text{IRIS}}(x_m, r) = \tilde{f}_m, \qquad \tilde{f}_m = F_{\mathcal{S}}(x_m),$$

where

$$F_{\mathcal{S}}(x_m) = \mathcal{T}_{\text{view}}\left(v_1(x_m), \ldots, v_{V_{\text{ref}}}(x_m)\right).$$

These point-wise latent features are then composed along the ray by a learnable ray-wise aggregator:

$$h_r = \mathcal{T}_{\text{ray}}(\tilde{f}_1, \ldots, \tilde{f}_M), \qquad \hat{C}_{\text{IRIS}}(r) = f_{\text{rgb}}(h_r).$$

Therefore, the key difference is that standard volume rendering uses explicit physical quantities (σ, c) together with a fixed alpha-compositing rule, whereas our method uses latent point-wise features together with a learned ray-wise composition function.

Besides Table 7 in the main paper, Fig. 8 shows that replacing our learned implicit renderer with standard volume rendering produces visibly blurrier results and consistently worse quantitative performance. We attribute this gap to the fact that volume rendering uses a fixed hand-crafted composition rule, while our renderer learns to compose view-conditioned latent features adaptively along each ray, yielding a more expressive decoding interface for high-fidelity synthesis. This phenomenon has also been observed in prior work. For example, GNT[7, 32] showed that learned ray-wise rendering can better capture fine structures and appearance effects. Our

Algorithm 1 Computing point-wise latent features

Require: A 3D query point x; latent scene memory

$$\mathcal{S} = \{(f_k, K, P_k)\}_{k \in \mathcal{V}_{\text{ref}}}$$

Require: View-transformer blocks $\{\mathcal{T}_{\text{view}}^{(l)}\}_{l=1}^{L}$

for each reference view $k \in \mathcal{V}_{\text{ref}}$ **do**

  Project x into view k and sample the corresponding feature:

$$v_k(x) = \Pi_k(x; f_k, P_k, K)$$

  Compute the relative geometric cue $\Delta_k(x)$ between the target ray and view k

  Compute the visibility / validity mask $m_k(x)$

end for

Project sampled view-wise features to the model width:

$$u_k^{(0)}(x) = \phi_{\text{proj}}\big(v_k(x)\big), \qquad k = 1, \ldots, V_{\text{ref}}$$

Initialize by max-pooling across reference views:

$$q^{(0)}(x) = \max_{k \in \mathcal{V}_{\text{ref}}} u_k^{(0)}(x)$$

for $l = 1$ to L **do**

  Update the point token by aggregating multi-view features:

$$q^{(l)}(x) = \mathcal{T}_{\text{view}}^{(l)}\big(q^{(l-1)}(x),\ \{u_k^{(0)}(x)\}_{k=1}^{V_{\text{ref}}},\ \{\Delta_k(x)\}_{k=1}^{V_{\text{ref}}},\ \{m_k(x)\}_{k=1}^{V_{\text{ref}}}\big)$$

end for

Define the point-wise latent feature as

$$\tilde{f}(x) = F_{\mathcal{S}}(x) = q^{(L)}(x)$$

return $\tilde{f}(x)$

Algorithm 2 Rendering a target ray from point-wise latent features

Require: A target ray $r = (o_r, d_r)$

Require: Precomputed point-wise latent features $\{\tilde{f}_m^{(0)}\}_{m=1}^{M}$ and their sample locations $\{x_m\}_{m=1}^{M}$

Require: Ray-transformer blocks $\{(\psi^{(l)}, \mathcal{T}_{\text{ray}}^{(l)})\}_{l=1}^{L}$

Encode the sample locations and the target-ray direction:

$$e_m^{\text{pos}} = \gamma_{\text{pos}}(x_m), \qquad e^{\text{dir}} = \gamma_{\text{dir}}(d_r)$$

for $l = 1$ to L **do**

  Update each point feature with point/ray encodings:

$$\tilde{f}_m^{(l-\frac{1}{2})} = \psi^{(l)}\big(\tilde{f}_m^{(l-1)},\ e_m^{\text{pos}},\ e^{\text{dir}}\big), \qquad m = 1, \ldots, M$$

  Aggregate the point-wise features along the ray:

$$(\tilde{f}_1^{(l)}, \ldots, \tilde{f}_M^{(l)}) = \mathcal{T}_{\text{ray}}^{(l)}\big(\tilde{f}_1^{(l-\frac{1}{2})}, \ldots, \tilde{f}_M^{(l-\frac{1}{2})}\big)$$

end for

Normalize and pool the final point features:

$$h_r = \frac{1}{M} \sum_{m=1}^{M} \text{Norm}\big(\tilde{f}_m^{(L)}\big)$$

Predict the final ray color:

$$\hat{C}(r) = f_{\text{rgb}}(h_r)$$

return $\hat{C}(r)$

result extends this observation to a new regime: unlike prior evidence obtained in posed settings, we verify it in a fully pose-free setting, where camera parameters themselves are learned under self-supervision.

Formal comparison of scene modeling and rendering. Table 9 summarizes the main differences among RayZer, E-RayZer, and IRIS in scene modeling and rendering. RayZer represents the scene with latent tokens and renders target views using a learned decoder. E-RayZer instead predicts explicit 3D Gaussians and renders them by differentiable splatting. In contrast, IRIS induces a latent neural field from multi-view features and performs learnable ray-wise rendering. This comparison highlights that our method differs from prior self-supervised paradigms in both the scene representation and the rendering interface.

More Qualitative comparisons. We present more qualitative comparisons in Fig. 7. x

Implementation details. For ray sampling, we use a fixed normalized depth range of $[0.1, 1.0]$ in all experiments. We note that, in the fully self-supervised pose-free setting, the recovered camera translations are only defined up to a global scale, so the absolute scene depth is not directly meaningful. In this sense, the depth range mainly serves as a canonical sampling interval rather than a metric depth prior. We choose $[0.1, 1.0]$ empirically, and find it sufficient for stable training and rendering quality across datasets.

Our implementation is mainly based on the GNT-style aggregation design. However, unlike GNT[32], which alternates multiple view and ray transformer blocks, our renderer uses only a single view aggregation stage to compute point-wise latent features and a single ray aggregation stage to decode the final ray color. In practice, the point-wise latent feature computation starts by projecting a 3D query point x into each reference view and sampling the corresponding feature

$$v_k(x) = \Pi_k(x; f_k, P_k, K), \qquad k \in \mathcal{V}_{\text{ref}}.$$

Besides the sampled feature itself, we also compute a relative geometric cue $\Delta_k(x)$ and a binary validity mask $m_k(x)$. The cue $\Delta_k(x)$ provides geometry-aware information about the relation between the target ray and the k-th reference-view observation, while $m_k(x)$ indicates whether the projected point falls into a valid image region and should participate in aggregation. The sampled features are first projected to the model width and max-pooled across views for initialization, and are then fused by a single view transformer to produce the point-wise latent feature $\tilde{f}(x) = F_{\mathcal{S}}(x)$.

For ray aggregation, given sampled points $\{x_m\}_{m=1}^{M}$ on a target ray $r = (o_r, d_r)$, we first compute their point-wise latent features $\{\tilde{f}_m\}_{m=1}^{M}$ and then inject sinusoidal encodings of both the sample locations and the target-ray direction:

$$e_m^{\text{pos}} = \gamma_{\text{pos}}(x_m), \qquad e^{\text{dir}} = \gamma_{\text{dir}}(d_r).$$

These enriched point features are then composed by a single ray transformer into a ray-wise representation h_r, from which the final color is predicted by f_{rgb}.

Inference efficiency. To evaluate inference efficiency, we benchmark all methods on a fixed test scene and vary the numbers of

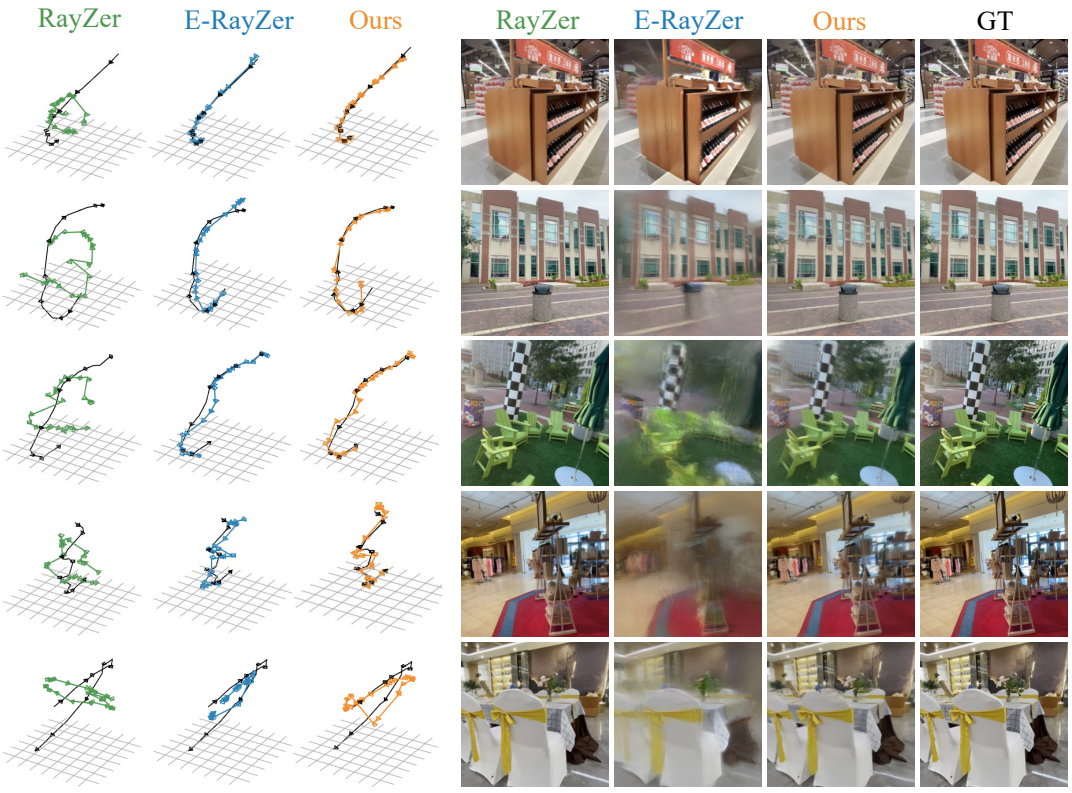


Figure 7: More qualitative comparisons of pose estimation and rendering quality. We visualize all poses of the predicted trajectory as colored frustums, while for the COLMAP trajectory we display only every third pose as a black frustum to reduce overlap.

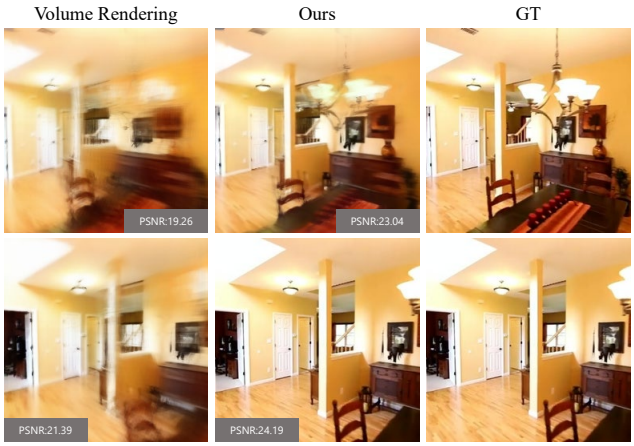


Figure 8: Compared with standard volume rendering, implicit rendering improves quality of novel view synthesis.

context and target views, denoted by M and N, respectively. For each (M, N) setting, we construct a single-scene batch by selecting M context frames and N target frames, move the batch to GPU, and measure the forward-pass latency using CUDA events. Following the benchmark script, each configuration is first warmed up for

Method	Self-supervised?	Mutli-view capable?	Remarks
MVSplat	✗	✗	
LVSM	✗	✓	
PF-LRM	✗	✓	Not publicly available.
NoPoSplat	✗	✗	
SelfSplat	✓	✗	
SPFSplat	✗(MASt3R init)	✓	Only trained on Re10K
RayZer	✓	✓	
E-RayZer	✓	✓	

Table 10: Taxonomy of the main baselines discussed in this paper, following our related-work categorization.

M	N	E-RayZer (ms)	RayZer (ms)	IRIS (s)
4	1	31.27	36.37	2.74
8	2	32.77	52.01	7.92
12	4	44.67	80.63	20.75
16	8	70.96	133.35	49.24

Table 11: Inference latency for baselines and ours.

two runs and then timed for four runs, and we report representative mean latency values in Table 11. By default, the benchmark measures the model forward pass itself and excludes additional method-specific input preparation overhead, so the reported numbers primarily reflect the rendering cost of each method under the corresponding (M, N) configuration. As shown in Table 11, E-RayZer is the fastest among the three methods at all representative operating points, while RayZer is moderately slower but remains in the millisecond regime. Our method is noticeably more expensive, and its latency increases more rapidly as both M and N grow. This trend is expected, since our renderer explicitly performs learned view aggregation and ray aggregation. As a result, the computational cost grows with both the number of target views and the amount of reference views used for each ray. At the same time, we stress that this additional cost is closely tied to the stronger rendering capability of our method. Our method adopts a finer-grained rendering process over sampled 3D points and multi-view features. In this sense, the higher latency mainly reflects a quality–efficiency trade-off.

References

[1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. 2022. Neural RGB-D Surface Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6290–6301.

[2] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, and Elad Shulman. 2021. ARKitScenes - A Diverse Real-World Dataset for 3D Indoor Scene Understanding Using Mobile RGB-D Data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. https://openreview.net/forum?id=tjZjv_qh_CE

[3] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. 2024. pixelSplat: 3D Gaussian Splats from Image Pairs for Scalable Generalizable 3D Reconstruction. In *Proc. CVPR*.

[4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. 2021. MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo. In *Proc. ICCV*.

[5] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. 2024. MVSplat: Efficient 3D Gaussian Splatting from Sparse Multi-View Images. *arXiv* 2403.14627 (2024).

[6] Yuedong Chen, Chuanxia Zheng, Haofei Xu, Bohan Zhuang, Andrea Vedaldi, Tat-Jen Cham, and Jianfei Cai. 2024. MVSplat360: Benchmarking 360 Generalizable 3D Novel View Synthesis from Sparse Views. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

[7] Wenyan Cong, Hanxue Liang, Peihao Wang, Zhiwen Fan, Tianlong Chen, Mukund Varma, Yi Wang, and Zhangyang Wang. 2023. Enhancing NeRF akin to Enhancing LLMs: Generalizable NeRF Transformer with Mixture-of-View-Experts. In *ICCV*.

[8] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. 2023. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400* (2023).

[9] Ranran Huang and Krystian Mikolajczyk. 2025. No Pose at All: Self-Supervised Pose-Free 3D Gaussian Splatting from Sparse Views. *arXiv preprint arXiv:2508.01171* (2025).

[10] Hanwen Jiang, Zhenyu Jiang, Yue Zhao, and Qixing Huang. 2023. LEAP: Liberate Sparse-view 3D Modeling from Camera Poses. *ArXiv* 2310.01410 (2023).

[11] Hanwen Jiang, Hao Tan, Peng Wang, Haian Jin, Yue Zhao, Sai Bi, Kai Zhang, Fujun Luan, Kalyan Sunkavalli, Qixing Huang, et al. 2025. RayZer: A Self-supervised Large View Synthesis Model. *arXiv preprint arXiv:2505.00702* (2025).

[12] Lihan Jiang, Yucheng Mao, Linning Xu, Tao Lu, Kerui Ren, Yichen Jin, Xudong Xu, Mulin Yu, Jiangmiao Pang, Feng Zhao, et al. 2025. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Transactions on Graphics (TOG)* 44, 6 (2025), 1–16.

[13] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. 2024. LVSM: A Large View Synthesis Model with Minimal 3D Inductive Bias. *arXiv* 2410.17242 (2024).

[14] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. 2025. LVSM: A Large View Synthesis Model with Minimal 3D Inductive Bias. In *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=QQBPWtvtcn

[15] Gyeongjin Kang, Jisang Yoo, Jihyeon Park, Seungtae Nam, Hyeonsoo Im, Sangheon Shin, Sangpil Kim, and Eunbyung Park. 2025. SelfSplat: Pose-free and 3D prior-free generalizable 3D Gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 22012–22022.

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (July 2023). https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[17] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. 2024. Grounding Image Matching in 3D with MASt3R. *arXiv preprint arXiv:2406.09756* (2024).

[18] Wenyu Li, Sidun Liu, Peng Qiao, Yong Dou, and Tongrui Hu. 2025. Muskie: Multi-view Masked Image Modeling for 3D Vision Pre-training. arXiv:2511.18115 [cs.CV] https://arxiv.org/abs/2511.18115

[19] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. 2021. BARF: Bundle-Adjusting Neural Radiance Fields. In *IEEE International Conference on Computer Vision (ICCV)*.

[20] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. 2024. 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*. 22160–22169.

[21] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Proc. ECCV*.

[22] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

[23] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. 2021. Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction. In *International Conference on Computer Vision*.

[24] Mehdi S. M. Sajjadi, Aravindh Mahendran, Thomas Kipf, Etienne Pot, Daniel Duckworth, Mario Lučić, and Klaus Greff. 2023. RUST: Latent Neural Scene Representations from Unposed Imagery. *CVPR* (2023).

[25] Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas Funkhouser, and Andrea Tagliasacchi. 2022. Scene Representation Transformer: Geometry-Free Novel View Synthesis Through Set-Latent Scene Representations. *CVPR* (2022). https://srt-paper.github.io/

[26] Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas A. Funkhouser, and Andrea Tagliasacchi. 2021. Scene Representation Transformer: Geometry-Free Novel View Synthesis Through Set-Latent Scene Representations. *CoRR* abs/2111.13152 (2021).

[27] Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *Proc. CVPR*.

[28] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. 2016. Pixelwise View Selection for Unstructured Multi-View Stereo. In *Proc. ECCV*.

[29] Thomas Schops, Johannes L Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. 2017. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3260–3269.

[30] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. 2013. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 2930–2937. doi:10.1109/CVPR.2013.377

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

[32] Mukund Varma T, Peihao Wang, Xuxi Chen, Tianlong Chen, Subhashini Venugopalan, and Zhangyang Wang. 2023. Is Attention All That NeRF Needs?. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=xE-LtsE-xx

[33] Peng Wang, Hao Tan, Sai Bi, Yinghao Xu, Fujun Luan, Kalyan Sunkavalli, Wenping Wang, Zexiang Xu, and Kai Zhang. 2024. PF-LRM: Pose-Free Large Reconstruction Model for Joint Pose and Shape Prediction. In *ICLR*.

[34] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P. Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas A. Funkhouser. 2021. IBRNet: Learning Multi-View Image-Based Rendering. In *Proc. CVPR*.

[35] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. 2021. NeRF−−: Neural Radiance Fields Without Known Camera Parameters. *arXiv preprint arXiv:2102.07064* (2021).

[36] Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. 2024. RGBD Objects in the Wild: Scaling Real-World 3D Object Learning from RGB-D Videos. arXiv:2401.12592 [cs.CV]

[37] Botao Ye, Sifei Liu, Haofei Xu, Li Xueting, Marc Pollefeys, Ming-Hsuan Yang, and Peng Songyou. 2024. No Pose, No Problem: Surprisingly Simple 3D Gaussian

Splats from Sparse Unposed Images. *arXiv preprint arXiv:2410.24207* (2024).
[38] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. 2023. ScanNet++: A High-Fidelity Dataset of 3D Indoor Scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*.
[39] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. 2021. PixelNeRF: Neural Radiance Fields from One or Few Images. In *Proc. CVPR*.
[40] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. 2024. GS-LRM: Large Reconstruction Model for 3D Gaussian Splatting. *European Conference on Computer Vision* (2024).
[41] Qitao Zhao, Hao Tan, Qianqian Wang, Sai Bi, Kai Zhang, Kalyan Sunkavalli, Shubham Tulsiani, and Hanwen Jiang. 2026. E-RayZer: Self-supervised 3D Reconstruction as Spatial Visual Pre-training. In *CVPR*.
[42] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. 2018. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817* (2018).
[43] Chen Ziwen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Li Fuxin, and Zexiang Xu. 2025. Long-LRM: Long-sequence Large Reconstruction Model for Wide-coverage Gaussian Splats. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.