StereoSplat: Metric-Scale Novel View Synthesis via Stereo-Grounded Gaussian Splatting

Vladimir Yugay[1]* **Denis Rozumny**[2] **Theo Gevers**[1]
Elias Vansteenkiste[2] **Martin R. Oswald**[1] **Diogo C. Luvizon**[2]
[1]University of Amsterdam [2]Meta
https://vladimiryugay.github.io/stereosplat

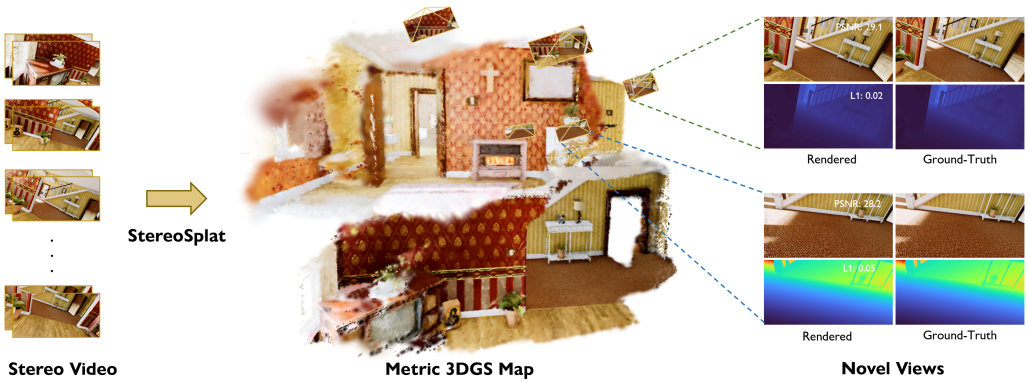

Figure 1: StereoSplat is a feed-forward framework to reconstruct a 3D Gaussian Splatting (3DGS) representation from calibrated stereo videos. Our novel stereo-native architecture leverages a stereo prior, an effective decoder, and a robust 3DGS aggregation mechanism to directly predict low-redundancy 3D Gaussians in world-grounded metric space from input views.

Abstract

We present StereoSplat, a feed-forward 3D Gaussian Splatting architecture designed explicitly for stereo videos to achieve reliable, metric-scale novel-view synthesis. Unlike monocular approaches that neglect the nature of binocular pairs, StereoSplat adopts a native stereo architecture that extracts dense multi-scale latent features and disparity from a foundation stereo backbone and feeds them into a dedicated Gaussian decoder. Our decoder utilizes cross-view feature fusion and gated depth refinement to directly translate these rich representations into per-pixel 3D Gaussians. To resolve the high redundancy of per-pixel predictions across overlapping views, we propose a learnable depth-adaptive aggregation mechanism that clusters and aggregates primitives in inverse-depth space while preserving fine-grained details. To support training and generalization, we introduce XRStereo, a large-scale synthetic dataset of stereo video featuring various rig configurations tailored to match the physical hardware of real-world platforms. Trained exclusively on synthetic data and evaluated across challenging benchmarks, StereoSplat yields substantial improvements in the depth accuracy of rendered novel views over established baselines while maintaining high-fidelity novel view synthesis. The model also transfers zero-shot to real-world stereo video, with applications in mixed reality, autonomous driving, and robotics.

*Work done during an internship at Meta.

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

1 Introduction

The introduction of 3D Gaussian Splatting (3DGS) [16] has established a new paradigm for real-time novel view synthesis, representing scenes as collections of anisotropic Gaussians that can be rendered at interactive frame rates. In parallel, scalable feed-forward architectures originating in natural language processing [27] and rapidly adopted in 2D [7] and 3D [30, 28] vision have enabled bypassing costly per-scene optimization entirely. Recently, works have begun to merge these two approaches, training large models that predict 3DGS representations directly from sparse input views in a single forward pass [4, 6, 14]. The resulting feed-forward 3DGS methods offer a compelling combination of real-time rendering, strong generalization across diverse scenes, and dramatically reduced reconstruction time compared to traditional multi-stage pipelines.

Despite this progress, current feed-forward 3DGS methods suffer from the fundamental limitation of lacking solid geometric grounding. Methods that rely on learned cost volumes or cross-attention to implicitly infer depth, such as pixelSplat [4] and MVSplat [6], produce noisy geometry that is only recoverable up to an arbitrary scale. Even recent approaches incorporating foundational depth priors [38] or 3D models [41, 14] remain anchored to scale-ambiguous, affine-invariant representations. While the synthesized novel views might look plausible in color, the underlying geometry is often inaccurate, lacks fine detail, and degrades with distance. This limitation, compounded by the lack of metric scale, renders these methods unsuitable for downstream tasks requiring high geometric accuracy. A key reason is that these models treat inputs as generic, unstructured images. In doing so, they discard the calibration information and epipolar constraints that structured camera setups, such as stereo rigs, readily provide.

Stereo vision offers a principled solution to this geometric grounding problem. Given a calibrated stereo rig with known baseline and intrinsics, which is a standard setup in XR devices and robotics, dense absolute metric depth can be recovered directly from disparity via triangulation. The recent emergence of stereo matching foundation models, such as FoundationStereo [35], has made this practical at scale by providing robust zero-shot disparity estimation across drastically different environments alongside rich multi-scale features that encode both local geometric detail and high-level scene semantics. However, to the best of our knowledge, no existing work bridges stereo foundation models with explicit 3DGS scene representations, and current architectures do not exploit stereo vision priors.

To address this gap, we present StereoSplat, the first feed-forward 3D Gaussian Splatting framework designed explicitly for stereo videos. Rather than relying on generic image-to-3D backbones that ignore the inherent geometric constraints of stereo cameras, we leverage a stereo disparity foundation model as our core feature extractor. We further propose a decoding module that processes stereo features and uses the metric stereo depth as a geometric anchor, so that scene reconstruction and novel view synthesis from multi-view inputs stay in metric scale. Given the high redundancy in per-pixel predictions across overlapping stereo views, we introduce a learnable spatial compression module that jointly models spatial and appearance information, reducing the memory and rendering cost of dense per-pixel 3DGS predictions without sacrificing quality. Finally, to enable robust generalization, we introduce a large-scale synthetic pre-training dataset comprising diverse and highly challenging stereo video sequences. Our contributions can be summarized as follows:

- StereoSplat, the first feed-forward 3DGS model natively designed for stereo videos, achieving generalizable metric-scale novel view synthesis.
- A stereo-aware Gaussian decoder that fuses cross-view features with a gated depth refinement mechanism to regress geometrically accurate 3D Gaussians.
- A novel scene aggregation mechanism that integrates geometric and appearance cues with learned confidences to efficiently fuse per-pixel primitives.
- A challenging, large-scale synthetic dataset for stereo video pre-training, alongside extensive evaluations demonstrating that StereoSplat yields a significant improvement in geometric accuracy over state-of-the-art feed-forward methods given the same stereo-video inputs.

2 Related Work

Foundational vision models. Since their introduction in machine translation [27], transformers have become the dominant architecture in natural language processing [23] and computer vision [11]. This trend has recently extended to 3D vision with the emergence of large-scale transformer models that jointly estimate camera poses and dense geometry [28, 29], or learn versatile representations for downstream 3D tasks [30]. DUSt3R [30] demonstrated that training transformer architectures on large-scale data using alternating self- and cross-attention enables the prediction of point maps in a shared coordinate system. VGGT [28] further extended this paradigm by scaling model capacity and predicting additional modalities beyond point maps, including tracks, depth, and camera poses. Despite their flexibility, these models are fundamentally constrained to unstructured, sparse-view, up-to-scale reconstructions and lack mechanisms for novel view synthesis. In contrast, our method operates directly on stereo sequences, enabling geometrically accurate, metric-scale reconstructions explicitly designed for high-fidelity novel view synthesis.

Feed-Forward Novel View Synthesis. While traditional NVS approaches require several optimization steps to reconstruct a scene via an explicit light field [17, 2] or an implicit neural field [21, 1, 22], generalizable methods train large-scale deep learning architectures to predict 3D scene representations in a single forward pass. Such approaches typically leverage neural rendering [43, 5, 8, 13], large reconstruction models [12, 39, 45, 47], and 3D Gaussian Splatting (3DGS) [33, 4, 6, 38]. Among these works, 3DGS models are widely adopted for their real-time rendering. However, they frequently struggle to reconstruct the scene geometry and incur large memory footprints due to per-pixel Gaussian predictions. To reduce this overhead, AnySplat [14] combined the VGGT backbone with differentiable voxelization to compress the representation. Although this improves spatial efficiency, the underlying geometric reconstruction remains limited because depth is only predicted up to scale, leading to incorrect geometry when combined with unreliable pose estimation. In contrast, our method is explicitly designed to operate on stereo videos and exploits the metric depth of calibrated stereo rigs, resulting in higher-fidelity color rendering. Furthermore, we achieve these results while effectively compressing the per-pixel primitives to maintain memory efficiency. Concurrently with our work, TokenGS [25] proposed compression by decoding splats from a fixed number of learnable tokens, but their framework is neither designed for stereo input nor compatible with dense depth priors.

Stereo matching. Deep stereo matching has progressed rapidly from early cost-volume architectures to large-scale foundation models. PSMNet [3] introduced spatial pyramid pooling combined with 3D convolutions to regularize cost volumes, while GwcNet [10] improved correlation efficiency through group-wise feature matching. A pivotal shift came with the adoption of recurrent refinement. RAFT [26], originally designed for optical flow, introduced an all-pairs correlation lookup with iterative GRU-based updates, and RAFT-Stereo [20] successfully adapted this paradigm to rectified stereo pairs. Subsequent methods further refined this iterative approach. For instance, IGEV-Stereo [36] combined geometry encoding volumes with iterative updates for improved accuracy, UniMatch [37] unified flow, stereo, and depth within a single framework, and Selective-Stereo [32] introduced adaptive frequency selection to handle varying scene content. Most recently, FoundationStereo [35] scaled this iterative paradigm to a foundation model trained on massive and diverse data, achieving robust zero-shot generalization across domains while providing rich multi-scale features alongside disparity. In our work, we leverage FoundationStereo as a backbone, repurposing its metric output and intermediate feature representations as the foundation for feed-forward 3DGS prediction.

3 Method

Given a video sequence of N rectified stereo image pairs $I = \{(I_i^L, I_i^R) \in \mathbb{R}^{H\times W\times 3}\}_{i=1}^N$, along with their corresponding camera extrinsics $T = \{(T_i^L, T_i^R) \in SE(3)\}_{i=1}^N$ and shared intrinsics K, our goal is to learn a mapping f_θ that directly predicts a set of 3D Gaussians $\mathcal{G}$ for novel view synthesis. Each Gaussian is parameterized by its mean in world coordinates $\mu \in \mathbb{R}^3$, appearance encoded via spherical harmonic coefficients $c \in \mathbb{R}^{3\times(k+1)^2}$ of degree k, anisotropic scale $s \in \mathbb{R}^3_{\geq 0}$, opacity $\sigma \in [0, 1]$, and rotation represented as a unit quaternion $q \in \mathbb{H}$.

We design our feed-forward reconstruction framework around a robust geometric prior, leveraging a frozen stereo foundation model to anchor the scene geometry. Specifically, we extract dense metric

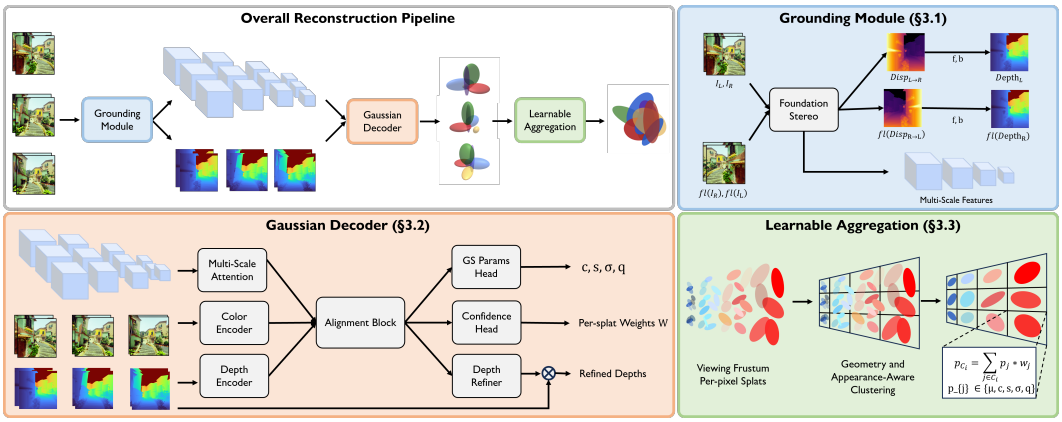

Figure 2: Overview of our framework. **Top-left**: StereoSplat consists of three key components: feature extraction and initial depth estimation, 3DGS decoding, and learnable primitive aggregation. **Top-right:** To extract per-view features and depth, the input stereo pair is flipped and reversed before passing through the backbone. Metric depth is then derived from the disparity maps using the camera focal length and stereo baseline. **Bottom-left:** Initial depth estimates and multi-scale features are aggregated in a shared feature space to regress 3DGS primitives and refine the depth. **Bottom-right:** The aggregation module clusters per-pixel predictions based on disparity and appearance features. Finally, learned per-splat weights are used to fuse primitives within the same voxel.

depth and multi-scale context features via bidirectional stereo matching (Sec. 3.1). These features are processed by the decoder that regresses per-pixel Gaussian attributes. During decoding, the stereo-derived depth is refined via a gated residual connection before the pixels are unprojected into an initial, dense set of 3D primitives $\tilde{\mathcal{G}}$ (Sec. 3.2). Finally, to ensure rendering and memory efficiency, we compress this redundant set into a compact representation $\mathcal{G}$ using distance-adaptive clustering and learned attribute aggregation (Sec. 3.3). The model is trained end-to-end via differentiable rendering using exclusively synthetic data, including the XRStereo dataset (Sec. 3.5). An overview of our framework is illustrated in Fig. 2.

3.1 Grounding module

For each stereo pair (I_i^L, I_i^R), we use FoundationStereo [35] as a frozen backbone $\mathcal{F}$ to extract initial depth and semantic features. Since $\mathcal{F}$ is designed to estimate disparity maps with respect to the left reference view ($d_i^L = \mathcal{F}(I_i^L, I_i^R)$), we exploit the symmetry of epipolar geometry to recover the corresponding right-view disparity. Specifically, we apply a horizontal reflection $\mathrm{flip}(\cdot)$ to the input pair to compute the right-view disparity:

$$d_i^R = \mathrm{flip}\Big(\mathcal{F}\big(\mathrm{flip}(I_i^R),\, \mathrm{flip}(I_i^L)\big)\Big) \tag{1}$$

These disparities are subsequently projected into metric depth maps $D_i^v = f \cdot b / d_i^v$ for each view $v \in \{L, R\}$ using the focal length f and baseline b of the calibrated rig, which set the metric scale of reconstruction. In addition to depth, this bidirectional inference provides a consistent multi-scale context feature pyramid $\{\mathbf{F}_i^{v,s}\}_{s=1}^{S}$ for both input images in a single forward pass, which serves as the foundation for our method.

3.2 Gaussian decoder

Using per-view images, depth maps, and feature pyramids, the decoder predicts a 3D Gaussian distribution for each pixel. This yields an initial set of primitives $\tilde{\mathcal{G}} = \{\tilde{g}_i^v\}$ of size $|\tilde{\mathcal{G}}| = 2NHW$.

Cross-view feature fusion. To inject multi-view context, we apply a lightweight cross-view transformer at every scale without sharing weights. At each scale s and spatial position p, we concatenate the left and right feature vectors across all views into a sequence of length $2N$, which is

processed by a stack of pre-norm transformer blocks:

$$\hat{\mathbf{F}}^{(s)}(p) = \text{Transformer}^{(s)}\Big(\Big[\mathbf{F}_1^{L,s}(p), \mathbf{F}_1^{R,s}(p), \ldots, \mathbf{F}_N^{L,s}(p), \mathbf{F}_N^{R,s}(p)\Big]\Big). \tag{2}$$

Each block comprises multi-head self-attention and a feed-forward network (FFN). Because spatial positions are processed independently, the attention matrix remains $2N \times 2N$, making the per-pixel computational complexity independent of the spatial dimensions $H_s \times W_s$. To ensure optimization stability, we zero-initialize the attention output projection and final FFN layer. This forces the module to initially act as an identity mapping, allowing it to smoothly adapt from the per-view baseline.

Gaussian Attribute Decoding. For brevity, we omit the pair index i and view index v, as the following operations are applied identically to each input image. The decoder fuses full-resolution image features, multi-scale depth features extracted from $\log D$, and the cross-view enhanced pyramid $\{\hat{\mathbf{F}}^{(s)}\}$ into a single full-resolution tensor via a series of upsampling and residual blocks. Because the initial depth estimate D can be inconsistent across views or time or because of flipping, we refine it by predicting a refined depth $\hat{D}$ and blending them using a learned scalar gate:

$$D' = (1-\alpha)\, D + \alpha\, \hat{D}, \qquad \alpha = \sigma(\alpha_0). \tag{3}$$

The learnable logit α_0 is initialized to strongly bias the network toward the geometrically stable stereo estimate D at the start of training. A linear head then maps each pixel feature to Gaussian attributes. Finally, using the refined depth D', camera intrinsics K, and poses T, we unproject each pixel into world coordinates to define the 3D mean μ, yielding the complete initial set of primitives $\tilde{\mathcal{G}}$.

3.3 Gaussian aggregation

The per-pixel set $\tilde{\mathcal{G}}$ is highly redundant. To reduce rendering costs, we compress $\tilde{\mathcal{G}}$ into a compact set $\mathcal{G}$ via hash-based aggregation. Because a Euclidean grid wastes capacity on distant geometry, we map Gaussian centers to inverse-depth coordinates $[x/z,\, y/z,\, 1/z]$ and quantize them on a grid of size δ. This naturally matches perspective projection. Lateral cell extent grows as δz and depth extent as δz^2, preserving detail on nearby surfaces while aggressively consolidating distant primitives. We augment the hash key with the quantized base color, given by the zeroth-order spherical harmonic coefficients, as well as scale magnitude and opacity, to prevent mixing across discontinuities. Within each cluster $\mathcal{C}$, we aggregate attributes using learned importance weights. A linear head predicts a scalar logit z_k for each primitive from its decoder features. We normalize these logits via a temperature-scaled softmax:

$$w_k = \frac{\exp(z_k/\tau)}{\sum_{j\in\mathcal{C}} \exp(z_j/\tau)}, \qquad \tau = \exp(\log \tau_0), \tag{4}$$

Here $\log \tau_0 \in \mathbb{R}$ is a learnable parameter controlling sharpness. By initializing $\log \tau_0 = 0$ to yield $\tau = 1$ and zero-initializing the weight head, all initial logits are zero. This reduces the softmax to a uniform distribution and exactly recovers standard `scatter_mean` aggregation at the start of training. Over time, the network learns to up-weight informative primitives. Although cluster assignments are discrete, weighted aggregation remains differentiable, allowing gradients to flow back to the decoder.

3.4 Training

We render the Gaussian scene $\mathcal{G}$ into target views using differentiable 3D Gaussian splatting [16] via the `gsplat` rasterizer [42]. Given a target camera with intrinsics $\mathbf{K}$ and pose $T \in SE_3$, the renderer projects and α-composites the primitives to produce a predicted image $\hat{I}$ alongside a depth map D_{rend}. We supervise the model against the ground-truth target image I and its corresponding metric depth D_{gt} using a joint objective. The photometric loss relies on the mean absolute error, structural similarity, and perceptual metrics.

$$\mathcal{L}_{\text{color}} = \lambda_1 \|\hat{I} - I\|_1 + \lambda_2 \big[1 - \text{SSIM}[\hat{I}, I]\big] + \lambda_3 \text{LPIPS}[\hat{I}, I] \tag{5}$$

Here the scalars $\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}_{\geq 0}$ weight the respective terms. We concurrently apply a depth loss using the mean absolute error.

$$\mathcal{L}_{\text{depth}} = \|D_{\text{rend}} - D_{\text{gt}}\|_1 \tag{6}$$

The complete training objective is $\mathcal{L} = \mathcal{L}_{\text{color}} + \lambda_d \mathcal{L}_{\text{depth}}$ with scalar weight $\lambda_d \in \mathbb{R}_{\geq 0}$. We optimize the decoder and aggregation modules end-to-end through the renderer.

3.5 XRStereo Dataset

The robust generalization of feed-forward stereo reconstruction is currently limited by the scarcity of high-quality stereo video datasets that capture static environments. To address this critical gap, we introduce XRStereo. This synthetic stereo video dataset comprises 137k high-quality frames split into 20k temporal sequences. These sequences feature moving cameras within static scenes across 9 indoor and 4 outdoor environments. To ensure visual diversity, each environment is populated with randomly placed objects sampled from a pool of 200 unique assets. All frames are rendered as rectified pinhole stereo pairs at 1K resolution with various rig configurations. These include horizontal fields of view ranging from 50 to 130 degrees and stereo baselines spanning 40 to 120 mm, reflecting the variability encountered in real VR/MR headsets and robotic platforms. Ground truth includes dense disparity maps, per-frame camera intrinsics, and extrinsics. A multi-stage quality assurance pipeline filters each frame to ensure sufficient non-dark pixels (>70%), minimal near-plane clipping (<2% of pixels below 11 cm depth), and low occlusion rates (>70% non-occluded pixels), with sequence-level QA further requiring all five frames to individually pass these checks. Combined with TartanAir [31] and Infinigen [15], XRStereo provides the geometric and appearance diversity needed to train our model for zero-shot transfer to real-world stereo video.

4 Experiments

Datasets. We train our method exclusively on synthetic data by combining complex samples from XRStereo with the extreme camera trajectories of TartanAir [31] and the diverse stereo sequences of Infinigen [15] training set. We evaluate our approach across three distinct settings. First, we report novel view synthesis performance within the same domain using the unseen test split of Infinigen. Second, we evaluate generalization following the evaluation protocol established by [24] on TartanAir v2. This benchmark introduces significant rendering difficulties due to extreme viewpoint shifts and severe illumination changes. Finally, we evaluate real-world generalization using the KITTI [9] dataset. This benchmark provides high-quality stereo videos and depth maps for testing our method outside the training distribution. Since our underlying 3D representation assumes a static scene, we manually exclude all video segments containing dynamic objects from both Infinigen and KITTI.

Evaluation Metrics. We evaluate novel views by first analyzing geometric depth reconstruction, followed by color synthesis. To measure geometric consistency, we use the Absolute Mean Relative Error and δ_1 accuracy following [14]. The Absolute Mean Relative Error calculates the average pixelwise normalized absolute depth difference, while the δ_1 metric finds the percentage of pixels where the predicted to true depth ratio stays within an acceptable bound. We additionally report the L1 error for absolute depth estimation following [44]. After validating geometric accuracy, we evaluate visual fidelity by computing PSNR, SSIM [34], and LPIPS [46] on full-resolution renders.

Baselines. We evaluate StereoSplat against state-of-the-art feed-forward novel-view synthesis methods spanning pose-aware and pose-free architectures. For pose-free baselines, we compare against AnySplat [14], VicaSplat [18], and the Gaussian head of Depth Anything 3 (DA3-Giant) [19]. We exclude NoPoSplat [41] because it strictly requires exactly two input images. Additionally, we benchmark against leading pose-aware models specifically DepthSplat [38], MVSplat [6], and PixelSplat [4]. At the time of submission, the checkpoints for the concurrent work, TokenGS [25], had not been released.

Evaluation Protocol. Following [41], we mask out regions unseen by the input cameras. We adopt the evaluation protocol of [40] for video sequences, dividing each video into local temporal windows of three stereo frames. We select this compact window to evaluate local temporal consistency, while extensive evaluations on larger context windows are provided in the supplementary material. Within each window, we select every other frame to evaluate novel view synthesis. All baselines receive the same input images as our method and are evaluated with their publicly released checkpoints, without retraining on stereo video. The pose-aware baselines MVSplat, PixelSplat, and DepthSplat use ground-truth poses. Because pose-free models inherently couple reconstruction and pose prediction in a normalized space, making ground-truth pose injection impossible, we follow the specific evaluation protocol introduced by [41, 14] for these methods. Since DA3 predicts depth only up to scale, we additionally align its rendered depth to the ground truth with an oracle median scaling before computing geometry metrics, which leaves its appearance metrics unchanged. Furthermore, to

evaluate each model within its intended operational limits, MVSplat, PixelSplat, and VicaSplat are tested at their natively supported 256×256 resolution, whereas StereoSplat operates at 320×640. To simulate real-world deployment, we evaluate our method across all experiments using trajectories from an off-the-shelf visual odometry system [24].

Our framework is not tied to a specific stereo backbone. We provide implementation details, a runtime and memory analysis, results with an alternative stereo backbone, an ablation of the gated depth refinement, evaluations with different numbers of input frames, additional qualitative results, and ablations on XRStereo in the supplementary material.

4.1 Novel View Synthesis

As shown in Table 1, StereoSplat produces highly accurate novel-view geometry together with high rendering quality. Conversely, baseline methods consistently struggle with scene geometry across datasets. For pose-free models like VicaSplat and AnySplat, this limitation arises because their predicted representations exist in a normalized, non-metric space. This issue is further exacerbated by treating stereo pairs as independent frames, which leads to accumulated camera pose errors and δ_1 scores below 10% on Infinigen. Pose-aware architectures encounter similar geometric challenges. Although they use large 3D or depth foundation models as backbones, explicit depth supervision, and ground-truth poses, methods such as pixelSplat, MVSplat, and DepthSplat do not produce accurate geometry from the same stereo frames. Notably, their geometric performance remains remarkably poor even when they produce visually plausible color reconstructions. In contrast, StereoSplat explicitly exploits the stereo structure and synthesizes novel views with highly accurate geometry. We analyze the contribution of each component in Sec. 4.2. Our approach consistently maintains accurate scene geometry across all tested environments, avoiding the severe degradation seen in baseline models (Fig. 5). StereoSplat also achieves the highest PSNR on all benchmarks, including the out-of-distribution real-world KITTI dataset.

4.2 Ablation Studies

Feature Selection. The stereo encoder generates a feature pyramid $\{\mathbf{F}^{(s)}\}_{s=1}^{S}$ at four spatial resolutions per view. To evaluate our design, we first test a naive baseline that processes all scales independently without cross-view fusion, yielding the lowest performance. We then ablate our proposed cross-attention model by selectively discarding scales. This reveals that omitting the high-dimensional fine spatial features $\mathbf{F}_{s=0}$ degrades performance more than losing the coarsest semantic context $\mathbf{F}_{s=4}$. Ultimately, applying cross-attention across the entire multi-scale pyramid yields the best overall rendering quality, as shown in Table 2.

Aggregation Mechanism. We evaluate several strategies for aggregating the primitives predicted at each pixel to improve spatial efficiency. The baseline retains all generated primitives without any reduction. We also compare against standard voxelization and the learnable variant introduced by Jiang et al. [14]. We then evaluate our proposed adaptive method, which dynamically scales voxel resolution using depth and appearance features. As demonstrated in Table 3, our approach achieves the highest rendering performance. This improvement arises from merging primitives in distant regions and clustering them using both geometric and visual cues. Furthermore, this strategy significantly increases the overall compression rate as visualized in Figure 3.

Compression Rate. Our approach provides a flexible trade-off between rendering quality and compression rate, defined as the ratio of retained primitives to the initial dense predictions. By adjusting the initial voxel size, we can significantly reduce the primitive count while incurring only a marginal decrease in reconstruction accuracy (Fig. 4). AnySplat [14] serves as the only baseline that explicitly compresses the 3D representation. We compare our method against this baseline on the Infinigen dataset. The camera motion between frames in these sequences remains relatively small across multiple input views. This minimal movement creates high visual redundancy, making the dataset an ideal environment for evaluating the effectiveness of compression mechanisms. Our approach demonstrates superior compression efficiency and scales effectively with the number of views, as shown in Fig. 3. This advantage stems directly from our adaptive voxelization strategy, which preserves critical structural details while discarding redundant primitives.

Table 1: **Novel view synthesis performance.** StereoSplat achieves the most accurate novel-view geometry and the highest PSNR across all benchmarks. Methods marked with † use ground truth camera poses. For methods marked with ‡, depth is aligned to the ground truth with oracle median scaling.

Method	Novel View Geometry			Novel View Appearance		
	AbsRel↓	δ_1↑	L1↓	PSNR↑	SSIM↑	LPIPS↓
Infinigen						
VicaSplat [18]	0.63	4.5	20.0	15.9	0.52	0.48
AnySplat [14]	0.53	0.5	15.9	25.2	0.88	0.18
DA3 [19]‡	0.41	42.2	9.5	21.6	0.73	0.33
MVSplat [6]†	2.64	20.7	33.5	27.1	0.90	0.15
pixelSplat [4]†	0.47	40.1	10.8	<u>29.0</u>	<u>0.95</u>	**0.07**
DepthSplat [38]†	<u>0.21</u>	<u>63.0</u>	<u>7.1</u>	26.8	0.91	0.13
Ours	**0.03**	**98.6**	**1.1**	**29.4**	**0.95**	<u>0.12</u>
TartanV2						
VicaSplat [18]	0.52	23.8	4.8	12.2	0.26	0.59
AnySplat [14]	0.22	61.9	2.8	18.6	0.66	0.26
DA3 [19]‡	0.47	40.8	3.6	18.3	0.59	0.32
MVSplat [6]†	1.35	8.4	8.1	20.2	0.85	0.18
pixelSplat [4]†	0.36	70.2	2.7	21.8	0.91	0.14
DepthSplat [38]†	<u>0.15</u>	<u>78.7</u>	<u>1.8</u>	<u>25.8</u>	<u>0.93</u>	**0.10**
Ours	**0.06**	**94.9**	**0.7**	**26.3**	**0.94**	**0.10**
KITTI						
VicaSplat [18]	0.25	49.5	5.3	16.5	0.64	0.28
AnySplat [14]	0.18	71.2	3.5	18.1	0.70	0.35
DA3 [19]‡	0.32	44.6	5.2	17.0	0.58	0.33
MVSplat [6]†	0.44	32.6	7.2	22.2	<u>0.92</u>	**0.09**
pixelSplat [4]†	0.37	43.6	6.9	<u>22.3</u>	0.91	0.11
DepthSplat [38]†	<u>0.13</u>	<u>79.8</u>	<u>2.4</u>	22.1	0.88	0.18
Ours	**0.04**	**96.0**	**0.9**	**25.2**	**0.94**	<u>0.10</u>

Table 2: Cross attention over multi-scale features across all input frames yields the best results.

Experiment	PSNR↑	L1↓
No fusion	22.8	1.4
Drop $\mathbf{F}_{s=0}$	22.9	1.1
Drop $\mathbf{F}_{s=4}$	23.1	1.1
Ours	**25.2**	**0.9**

Table 3: Adaptive aggregation enables high compression while improving rendering quality.

Strategy	PSNR↑	Comp. Rate↑
No Aggregation	23.3	1.0
Voxelization	23.2	1.1
Learnable Vox.	23.3	1.1
Ours	**25.2**	**1.6**

Scaling to Longer Videos. Our approach scales effectively to long video sequences. Thanks to the high geometric accuracy and pose-aware design of our framework, we perform inference independently across temporal chunks, concatenate the predictions, and refine the full aggregated scene with a small number of optimization iterations. As shown in Fig. 1, this approach successfully consolidates over 600 input frames into a globally consistent representation.

Limitations and Future Work. Although StereoSplat demonstrates strong NVS performance, it inherently relies on calibrated stereo pairs, precluding its use with uncalibrated or monocular captures. Furthermore, because our method assumes a static environment during scene aggregation, reconstruction quality degrades when processing highly dynamic videos. In addition, its end-to-end reconstruction speed is bounded by the stereo backbone, although our aggregation yields the lowest memory footprint and the fastest rendering among the compared methods (see supplementary material). An exciting direction for future work is to relax the calibration requirement and extend our framework to dynamic scenes.

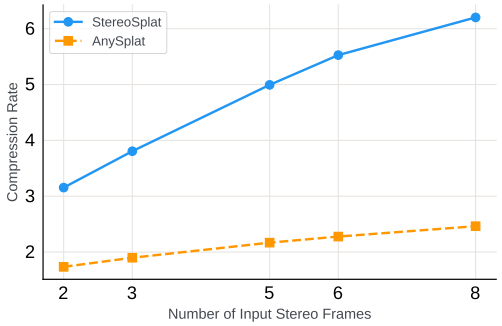

Figure 3: Our adaptive voxelization enables StereoSplat to achieve superior compression which scales as more stereo frames are provided.

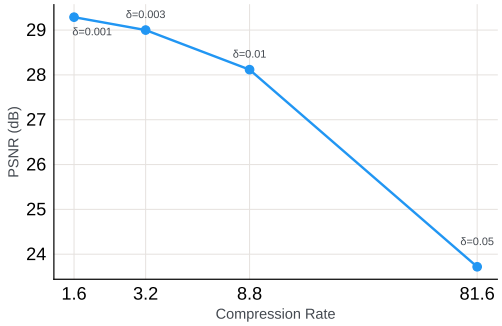

Figure 4: StereoSplat offers a flexible balance between compression rate and rendering quality by tuning the initial voxel size δ.

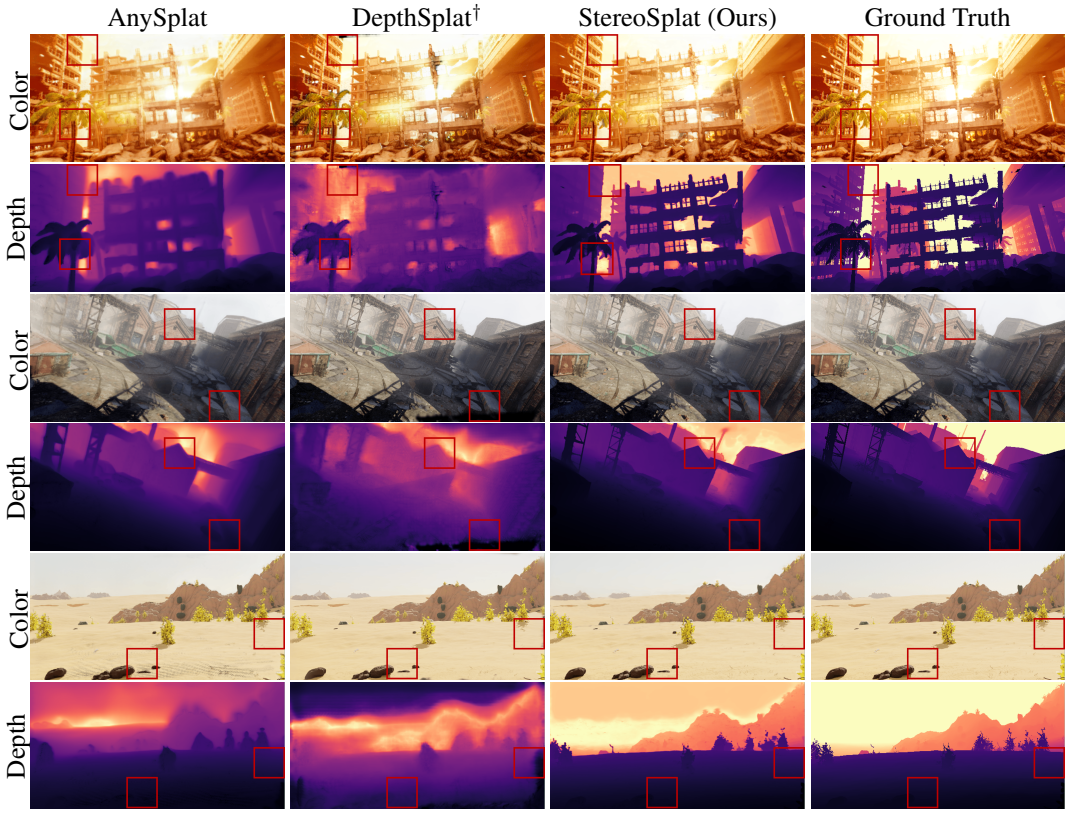

Figure 5: **Qualitative Results** on TartanAir and Infinigen. While the baselines can plausibly render the color of the novel views, they struggle to render accurate depth. StereoSplat with a stereo-native architecture is capable of capturing both accurate depth and color. Methods marked with † use ground-truth camera poses. Red squares highlight key regions. Best viewed zoomed in.

5 Conclusion

We presented StereoSplat, the first feed-forward 3D Gaussian Splatting framework designed for stereo videos. StereoSplat exploits the metric scale of calibrated stereo rigs and preserves it throughout the pipeline, overcoming the scale ambiguity of previous methods to predict accurate 3D Gaussians in metric space. Furthermore, our scene aggregation mechanism efficiently compresses per-pixel primitives, while pre-training on our large-scale synthetic XRStereo dataset improves generalization to real-world data. Extensive evaluations demonstrate that StereoSplat achieves state-of-the-art geometric accuracy and superior novel view synthesis across diverse synthetic and real-world scenes.

Acknowledgments and Disclosure of Funding

We thank Jaap Suter, Daniel Scharstein, Conor Oberlander, and David Fuhrer at Meta for creating the XRStereo dataset used in this work. Their work on the rendering pipeline, art design, and the variety of stereo rig configurations made the large-scale stereo video pre-training in this paper possible.

Funding. This work was funded by Meta and was carried out by Vladimir Yugay during an internship at Meta.

Competing interests. Denis Rozumny, Elias Vansteenkiste, and Diogo C. Luvizon are employees of Meta, and Vladimir Yugay was an intern at Meta while conducting this work. The authors declare no other competing interests.

References

[1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Computer Vision and Pattern Recognition*, 2022.

[2] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 425–432. ACM, 2001. doi: 10.1145/383259.383309.

[3] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

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

[5] 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. *arXiv preprint arXiv:2103.15595*, 2021.

[6] 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. *arXiv preprint arXiv:2403.14627*, 2024.

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

[8] John Flynn, Michael Broxton, Lukas Murmann, Lucy Chai, Matthew DuVall, Clément Godard, Kathryn Heal, Srinivas Kaza, Stephen Lombardi, Xuan Luo, Supreeth Achar, Kira Prabhu, Tiancheng Sun, Lynn Tsai, and Ryan Overbeck. Quark: Real-time, high-resolution, and general neural view synthesis. *ACM Trans. Graph.*, 43(6), November 2024. ISSN 0730-0301. doi: 10.1145/3687953. URL `https://doi.org/10.1145/3687953`.

[9] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[10] Xiaoyang Guo, Kai Yang, Wukui Yang, Xiaogang Wang, and Hongsheng Li. Group-wise correlation stereo network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3273–3282, 2019.

[11] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[12] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d, 2024. URL `https://arxiv.org/abs/2311.04400`.

[13] Hanwen Jiang, Hao Tan, Peng Wang, Haian Jin, Yue Zhao, Sai Bi, Kai Zhang, Fujun Luan, Kalyan Sunkavalli, Qixing Huang, et al. Rayzer: A self-supervised large view synthesis model. *arXiv preprint arXiv:2505.00702*, 2025.

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

[15] Junpeng Jing, Ye Mao, and Krystian Mikolajczyk. Match-stereo-videos: Bidirectional alignment for consistent dynamic stereo matching. In *European Conference on Computer Vision*, pages 415–432. Springer, 2024.

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023.

[17] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pages 31–42. ACM, 1996. doi: 10.1145/237170.237199.

[18] Zhiqi Li, Chengrui Dong, Yiming Chen, Zhangchi Huang, and Peidong Liu. Vicasplat: A single run is all you need for 3d gaussian splatting and camera estimation from unposed video frames. *arXiv preprint arXiv:2503.10286*, 2025.

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

[20] Lahav Lipson, Zachary Teed, and Jia Deng. Raft-stereo: Multilevel recurrent field transforms for stereo matching, 2021. URL `https://arxiv.org/abs/2109.07547`.

[21] 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.

[22] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4): 1–15, 2022.

[23] OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[24] Yuheng Qiu, Yutian Chen, Zihao Zhang, Wenshan Wang, and Sebastian Scherer. Mac-vo: Metrics-aware covariance for learning-based stereo visual odometry. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3814. IEEE, 2025.

[25] Jiawei Ren, Michal Tyszkiewicz, Jiahui Huang, and Zan Gojcic. Tokengs: Decoupling 3d gaussian prediction from pixels with learnable tokens. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[26] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow, 2020. URL `https://arxiv.org/abs/2003.12039`.

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

[28] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *CVPR*, 2025.

[29] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. *arXiv preprint arXiv:2501.12387*, 2025.

[30] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024.

[31] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam, 2020. URL https://arxiv.org/abs/2003.14338.

[32] Xianqi Wang, Gangwei Xu, Hao Jia, and Xin Yang. Selective-stereo: Adaptive frequency information selection for stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19701–19710, 2024.

[33] Yunsong Wang, Tianxin Huang, Hanlin Chen, and Gim Hee Lee. Freesplat: Generalizable 3d gaussian splatting towards free-view synthesis of indoor scenes. *arXiv preprint arXiv:2405.17958*, 2024.

[34] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4): 600–612, 2004.

[35] Bowen Wen, Matthew Trepte, Joseph Aribido, Jan Kautz, Orazio Gallo, and Stan Birchfield. Foundationstereo: Zero-shot stereo matching. *CVPR*, 2025.

[36] Gangwei Xu, Xianqi Wang, Xiaohuan Ding, and Xin Yang. Iterative geometry encoding volume for stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21919–21928, 2023.

[37] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, Fisher Yu, Dacheng Tao, and Andreas Geiger. Unifying flow, stereo and depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

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

[39] Yinghao Xu, Zifan Shi, Wang Yifan, Sida Peng, Ceyuan Yang, Yujun Shen, and Wetzstein Gordon. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arxiv: 2403.14621*, 2024.

[40] Zhen Xu, Zhengqin Li, Zhao Dong, Xiaowei Zhou, Richard Newcombe, and Zhaoyang Lv. 4DGT: Learning a 4D gaussian transformer using real-world monocular videos. In *Advances in Neural Information Processing Systems*, 2025.

[41] Botao Ye, Sifei Liu, Haofei Xu, Li Xueting, Marc Pollefeys, Ming-Hsuan Yang, and Peng Songyou. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. *arXiv preprint arXiv:2410.24207*, 2024.

[42] Vickie Ye, Ruilong Li, Justin Kerr, Matias Turkulainen, Brent Yi, Zhuoyang Pan, Otto Seiskari, Jianbo Ye, Jeffrey Hu, Matthew Tancik, and Angjoo Kanazawa. gsplat: An open-source library for gaussian splatting. *Journal of Machine Learning Research*, 26(34):1–17, 2025.

[43] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. Pixelnerf: Neural radiance fields from one or few images. In *Computer Vision and Pattern Recognition*, 2021.

[44] Vladimir Yugay, Theo Gevers, and Martin R. Oswald. Magic-slam: Multi-agent gaussian globally consistent slam, 2024. URL https://arxiv.org/abs/2411.16785.

[45] Kai Zhang, Sai Bi, Hao Tan, Yuanbo Xiangli, Nanxuan Zhao, Kalyan Sunkavalli, and Zexiang Xu. Gs-lrm: Large reconstruction model for 3d gaussian splatting. *European Conference on Computer Vision*, 2024.

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

[47] Chen Ziwen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Li Fuxin, and Zexiang Xu. Long-lrm: Long-sequence large reconstruction model for wide-coverage gaussian splats, 2025. URL https://arxiv.org/abs/2410.12781.

A Supplementary Material

A.1 Implementation Details

We train and evaluate our model on 320×640 resolution images with three input views per sample. The model is optimized using AdamW with a total effective batch size of 40. The learning rate follows a 10-epoch schedule, starting with a one-epoch linear warm-up to $\eta_{\max} = 10^{-3}$, followed by a cosine decay to $\eta_{\min} = 10^{-5}$ over the remaining budget. The total training objective is defined as:

$$\mathcal{L} = 0.8\,\mathcal{L}_{\ell_1}^{\text{rgb}} + 0.2\,\mathcal{L}_{\text{ssim}}^{\text{rgb}} + 0.05\,\mathcal{L}_{\text{lpips}}^{\text{rgb}} + 0.5\,\mathcal{L}_{\ell_1}^{\text{depth}}, \tag{7}$$

where the depth term $\mathcal{L}_{\ell_1}^{\text{depth}}$ is masked to include only pixels with valid ground-truth depth within the range $[0, 80]$ m. The loss weights are determined empirically based on validation performance. The framework is implemented in PyTorch and uses the original implementation from FoundationStereo as backbone, and the total training time is approximately 240 GPU-hours on NVIDIA H100.

A.2 Additional Quantitative Evaluation

To further evaluate the robustness of StereoSplat, we provide additional experiments across varying temporal window sizes $N \in \{2, 3, 4\}$, corresponding to total image counts of $4, 6$, and 8, respectively. This evaluation is conducted in a progressive manner, where for a window of size N, we utilize $N - 1$ intermediate frames for evaluation, which effectively increases the task difficulty by expanding the temporal baseline and increasing the number of 3DGS primitives. For improved readability, we categorize our results into geometric accuracy (Table 4) and appearance quality (Table 5). In both tables, we include a baseline comparison of the model's performance when trained without XRStereo to quantify the impact of our proposed data. While this variant remains competitive on the in-distribution Infinigen and TartanV2 benchmarks, the gap widens substantially on KITTI, indicating that XRStereo is particularly beneficial for generalization to unseen real-world data. While the reported experiments are constrained by standard evaluation protocols, our model is inherently scalable and supports an arbitrary number of input views, bounded only by the available GPU memory.

A.3 Additional Qualitative Comparisons

We provide extended qualitative comparisons to further evaluate the visual fidelity of our method. For these visualizations, we focus on the most competitive baselines, AnySplat and DepthSplat. We omit VicaSplat, PixelSplat, and MVSplat from this comparison as these methods are natively restricted to a 256×256 resolution, which precludes a fair high-resolution comparison with our 320×640 results. Furthermore, these models consistently underperform relative to more recent state-of-the-art approaches. As illustrated in Figure 6, AnySplat exhibits significant artifacts, particularly in rows 3 and 4, stemming from inaccuracies in camera pose estimation. While DepthSplat produces plausible color renderings, its underlying geometry is often inconsistent. This geometric inaccuracy manifests as photometric artifacts. For instance, the depth errors in row 5 directly degrade the clarity and structural integrity of the synthesized color view. In contrast, StereoSplat maintains both sharp geometric boundaries and high color fidelity.

Table 4: **Novel view geometry.** StereoSplat achieves the most accurate novel-view geometry across all benchmarks. Methods marked with † use ground truth camera poses.

Method	2 Stereo Frames			3 Stereo Frames			4 Stereo Frames		
	AbsRel↓	δ_1↑	L1↓	AbsRel↓	δ_1↑	L1↓	AbsRel↓	δ_1↑	L1↓
Infinigen									
VicaSplat	0.63	4.5	20.0	0.60	8.4	18.9	0.72	2.7	23.1
AnySplat	0.53	0.5	15.9	0.50	1.4	15.0	0.48	1.7	14.7
MVSplat†	2.64	20.7	33.5	2.54	20.5	32.4	2.63	20.0	33.6
pixelSplat†	0.47	40.1	10.8	0.48	39.7	10.8	0.48	39.7	10.8
DepthSplat†	0.21	63.0	7.1	0.26	52.5	10.2	0.29	49.6	10.6
Ours w/o XRStereo	0.04	97.5	**1.1**	0.04	97.5	1.1	0.04	97.7	1.1
Ours	**0.03**	**98.6**	**1.1**	**0.03**	**98.6**	**1.0**	**0.03**	**98.7**	**1.0**
TartanV2									
VicaSplat	0.52	23.8	4.8	0.55	19.4	5.1	0.55	28.7	4.5
AnySplat	0.22	61.9	2.8	0.21	66.4	2.7	0.20	67.0	2.6
MVSplat†	1.35	8.4	8.1	1.42	10.2	8.5	1.54	10.7	8.7
pixelSplat†	0.36	70.2	2.7	0.57	63.5	3.1	0.66	60.8	3.3
DepthSplat†	0.15	78.7	1.8	0.15	80.5	1.8	0.13	82.7	1.8
Ours w/o XRStereo	**0.06**	94.8	**0.7**	**0.08**	94.4	**0.7**	**0.06**	**94.8**	**0.7**
Ours	**0.06**	**94.9**	**0.7**	**0.08**	**94.5**	**0.7**	**0.06**	**94.8**	**0.7**
KITTI									
VicaSplat	0.25	49.5	5.3	0.35	22.6	7.5	0.34	43.4	5.4
AnySplat	0.18	71.2	3.5	0.13	82.4	2.8	0.14	79.1	3.0
MVSplat†	0.44	32.6	7.2	0.37	38.0	6.4	0.37	37.3	6.4
pixelSplat†	0.37	43.6	6.9	0.44	35.6	7.9	0.47	33.0	8.4
DepthSplat†	0.13	79.8	2.4	0.15	76.1	2.6	0.20	64.1	3.3
Ours w/o XRStereo	0.06	94.7	1.1	0.07	91.7	1.2	0.07	91.6	1.2
Ours	**0.04**	**96.0**	**0.9**	**0.05**	**95.7**	**1.0**	**0.04**	**95.7**	**1.0**

Table 5: **Novel view appearance.** Methods marked with † use ground truth camera poses.

Method	2 Stereo Frames			3 Stereo Frames			4 Stereo Frames		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Infinigen									
VicaSplat	15.9	0.52	0.48	16.4	0.52	0.50	17.1	0.53	0.48
AnySplat	25.2	0.88	0.18	24.7	0.86	0.19	25.4	0.88	0.19
MVSplat†	27.1	0.90	0.15	26.1	0.87	0.17	25.5	0.86	0.19
pixelSplat†	29.0	**0.95**	**0.07**	28.2	0.94	**0.09**	28.1	0.93	**0.09**
DepthSplat†	26.8	0.91	0.13	26.9	0.91	0.14	26.7	0.91	0.15
Ours w/o XRStereo	28.6	0.94	0.18	28.4	0.94	0.18	28.2	0.93	0.18
Ours	**29.4**	**0.95**	0.12	**29.0**	**0.95**	0.11	**28.7**	**0.95**	0.12
TartanV2									
VicaSplat	12.2	0.26	0.59	12.1	0.24	0.60	12.2	0.24	0.60
AnySplat	18.6	0.66	0.26	17.5	0.58	0.32	17.2	0.56	0.35
MVSplat†	20.2	0.85	0.18	17.7	0.72	0.31	17.0	0.67	0.35
pixelSplat†	21.8	0.91	0.14	19.2	0.80	0.26	18.4	0.76	0.30
DepthSplat†	25.8	0.93	**0.10**	25.0	0.91	**0.12**	**25.3**	0.92	**0.12**
Ours w/o XRStereo	**26.3**	**0.94**	0.11	25.0	0.92	0.13	24.9	0.92	**0.12**
Ours	**26.3**	**0.94**	**0.10**	**25.2**	**0.93**	0.13	25.0	**0.93**	**0.12**
KITTI									
VicaSplat	16.5	0.64	0.28	13.5	0.42	0.42	14.2	0.43	0.42
AnySplat	18.1	0.70	0.35	17.1	0.63	0.40	16.6	0.60	0.44
MVSplat†	22.2	0.92	**0.09**	18.5	0.82	0.18	17.4	0.77	0.23
pixelSplat†	22.3	0.91	0.11	19.4	0.80	0.21	18.5	0.75	0.26
DepthSplat†	22.1	0.88	0.18	21.3	0.84	0.21	19.4	0.78	0.27
Ours w/o XRStereo	24.4	0.93	0.12	22.4	0.89	0.18	21.3	0.88	0.20
Ours	**25.2**	**0.94**	0.10	**23.9**	**0.92**	**0.13**	**22.9**	**0.90**	**0.14**

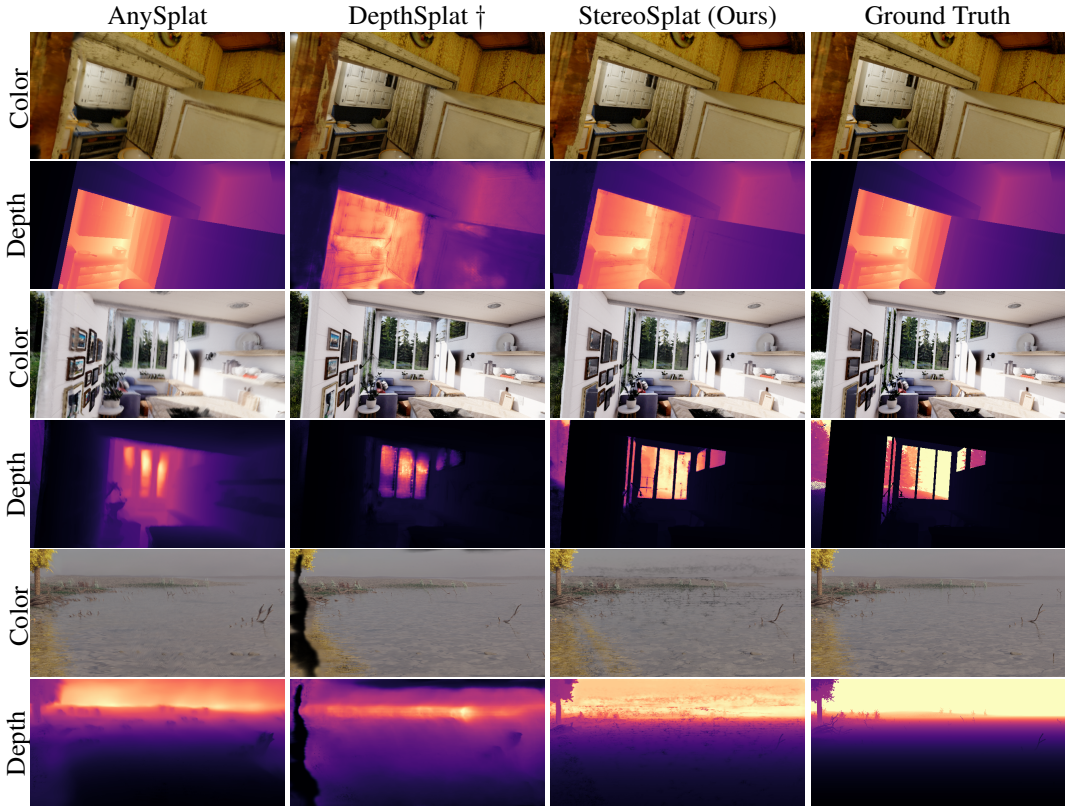

Figure 6: **Qualitative Results** on TartanAir and Infinigen. While the baselines can plausibly render the color of the novel views, they struggle to render accurate depth. StereoSplat with a stereo-native architecture is capable of capturing both accurate color and depth. Methods marked with † use ground-truth camera poses. Best viewed zoomed in.

A.4 XRStereo

We provide representative visual samples of XRStereo in Figure 7. To facilitate robust generalization for feed-forward stereo reconstruction, XRStereo comprises 137k high-quality 1K resolution frames across 20k temporal sequences. These sequences depict static scenes with dynamic camera trajectories across 13 diverse indoor and outdoor environments, populated with a large-scale library of 200 unique assets. A distinguishing feature of our dataset is the systematic variation of camera rig configurations, including fields of view ranging from 50° to 130° and stereo baselines from 40 to 120 mm. This variability ensures the model is resilient to the diverse hardware specifications found in consumer VR/MR headsets and robotic platforms. Furthermore, all samples have passed a multi-stage quality assurance pipeline that filters for sufficient illumination, minimal near-plane clipping, and low occlusion rates. By providing dense, high-fidelity disparity maps and precise camera metadata, XRStereo offers the geometric complexity necessary for training a robust model.

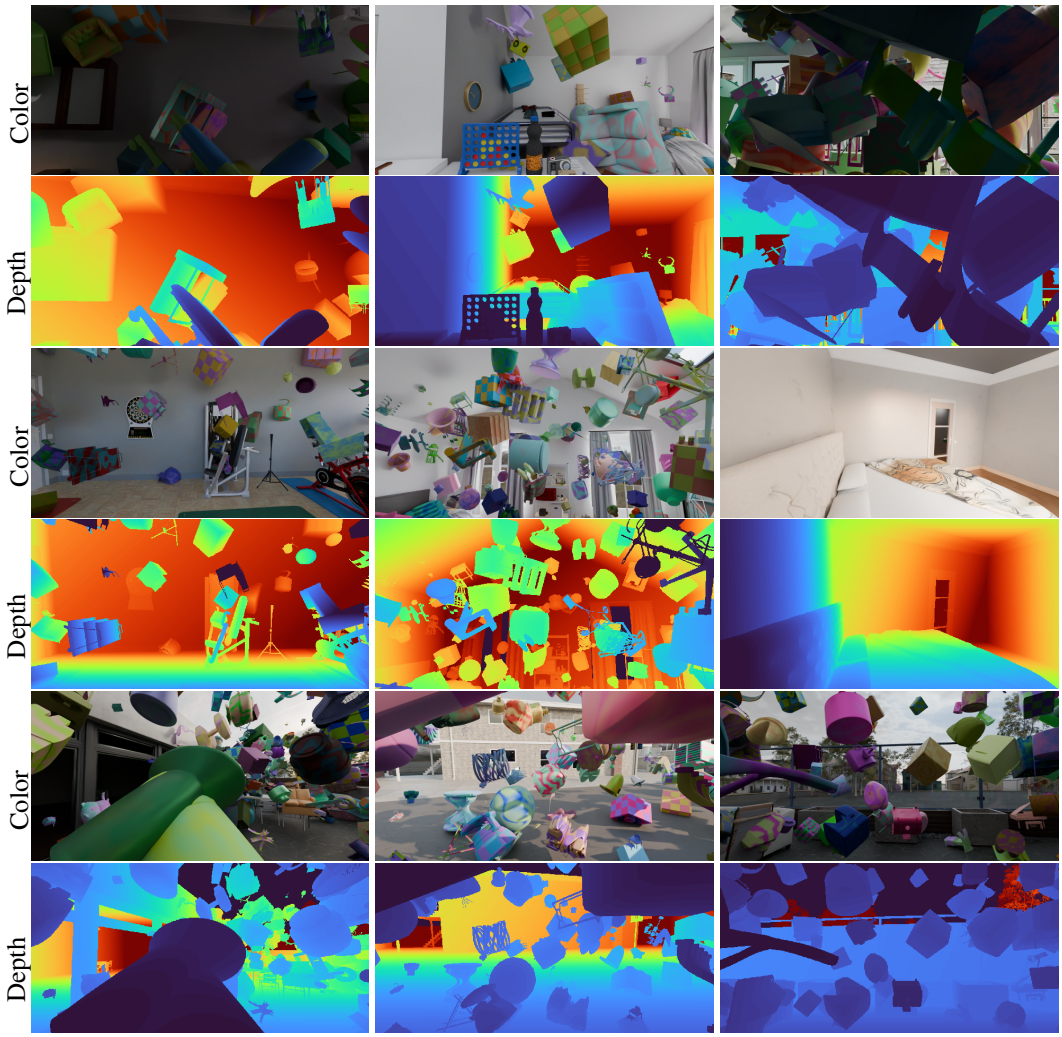

Figure 7: **Samples of XRStereo**. Representative left-view color frames and corresponding depth maps from our pretraining dataset. XRStereo comprises 137k high-resolution (1K) frames spanning 20k temporal sequences across 13 diverse indoor and outdoor environments populated with 200 unique assets yielding the geometric and photometric diversity needed to train a generalizable feed-forward stereo reconstruction model.

A.5 Runtime and Memory Analysis

We profile all methods on the Infinigen test split under the four-view protocol of Table 1, using CUDA-event timing (10 warm-up and 50 timed iterations, batch size 1) on a single NVIDIA A100-80GB GPU. As shown in Table 6, the end-to-end reconstruction of StereoSplat is currently slower than that of the baselines. However, our aggregation compresses 819k primitives to 132k ($6.2\times$), yielding the lowest peak memory and the highest rendering speed. Since reconstruction runs once per scene while rendering repeats for every target view, rendering throughput is the more relevant metric for interactive novel view synthesis. The efficiency of StereoSplat therefore lies in its compact representation and fast rendering rather than in end-to-end reconstruction latency.

Table 7 shows that this latency is almost entirely due to the frozen FoundationStereo [35] backbone. Our grounding module, decoder, and aggregation together add only 48.3 ms, and the aggregation alone, with 34 parameters, adds 4.4 ms. The backbone cost stems from its iterative GRU refinement and can be tuned without retraining: reducing the iterations from 12 to 8 yields a $1.14\times$ speedup at a 0.6% relative increase in AbsRel, and 6 iterations yield $1.23\times$ at 2.0%. Finally, per-pixel primitive generation is linear in the number of pixels and processed per view, and the hash-based aggregation is linear in the number of primitives, so long inputs can be processed in temporal chunks (Fig. 1).

Table 6: **Efficiency comparison** on Infinigen with four input views, measured on a single NVIDIA A100-80GB GPU. Our aggregation yields the most compact representation, the lowest peak memory, and the fastest rendering.

Method	Params	Peak Mem. [GB]↓	#Splats		FPS↑	
			Pre-Agg.	Post-Agg.↓	Total	Render
DepthSplat [38]	38.4M	6.14	1,376k	1,376k	**5.2**	59.5
AnySplat [14]	1.19B	5.73	776k	505k	1.9	157.2
Ours	65.0M	**2.03**	819k	**132k**	1.7	**236.9**

Table 7: **Runtime breakdown** of StereoSplat per reconstruction under the setting of Table 6. The frozen stereo backbone dominates the runtime, while our modules add only 48.3 ms.

Stage	Time [ms]	Share [%]	Params
FoundationStereo backbone (frozen)	558.5	91.4	62.3M
Grounding module + Gaussian decoder	43.9	7.2	2.71M
Gaussian aggregation	4.4	0.7	34
Rendering (post-agg.)	4.2	0.7	0
Total	610.9	100	65.0M

A.6 Alternative Stereo Backbone

Our framework is not tied to a specific stereo backbone. To demonstrate this, we replace FoundationStereo [35] with the lighter IGEV++ [36] and keep the rest of the architecture unchanged. As with FoundationStereo, we keep IGEV++ frozen with its publicly released SceneFlow weights and train only the remaining modules. As shown in Table 8, the IGEV++ variant still recovers considerably more accurate geometry than all baselines in Table 1. It trails our FoundationStereo model by 0.9–3.2 dB PSNR, with the largest gap on real-world KITTI, showing that a stronger stereo prior directly translates into higher rendering quality.

Table 8: **Alternative stereo backbone.** StereoSplat with the FoundationStereo backbone (numbers from Table 1) compared to StereoSplat with the IGEV++ backbone, evaluated under the same protocol. Both backbones are frozen.

Backbone	Novel View Geometry			Novel View Appearance		
	AbsRel↓	δ_1↑	L1↓	PSNR↑	SSIM↑	LPIPS↓
Infinigen						
IGEV++	0.07	90.9	2.2	27.1	0.85	0.29
FoundationStereo	0.03	98.6	1.1	29.4	0.95	0.12
TartanV2						
IGEV++	0.08	92.6	1.0	25.4	0.93	0.15
FoundationStereo	0.06	94.9	0.7	26.3	0.94	0.10
KITTI						
IGEV++	0.10	86.5	1.8	22.0	0.88	0.22
FoundationStereo	0.04	96.0	0.9	25.2	0.94	0.10

A.7 Gated Depth Refinement

To validate the gated depth refinement, we train variants with the gate α in Eq. (3) fixed. Setting $\alpha = 0$ removes the refinement and renders from the stereo depth D alone, while still using bidirectional inference, the decoder, and aggregation. Setting $\alpha = 1$ discards the stereo anchor and renders from the refined depth $\hat{D}$ alone. As shown in Table 9, the $\alpha = 0$ variant retains most of the geometric accuracy but loses 1.3 dB PSNR and roughly doubles LPIPS. Calibrated stereo thus provides a strong geometric anchor, while the learned refinement mainly improves rendering quality. Conversely, $\alpha = 1$ degrades geometry, and neither fixed setting matches the learned gate.

Table 9: **Gated depth refinement** on KITTI. Fixing the gate α in Eq. (3) to either extreme degrades performance compared to the learned gate.

Variant	AbsRel↓	δ_1↑	PSNR↑	LPIPS↓
$\alpha = 0$ (no refinement)	0.042	95.8	23.9	0.20
$\alpha = 1$ (no stereo anchor)	0.050	95.7	24.8	0.14
Ours (learned α)	**0.040**	**96.0**	**25.2**	**0.10**

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 main claims made in the abstract and introduction are clearly scoped and corroborated by the experimental evaluation in Experiments section.

   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 "Limitations and Future Work" paragraph at the end of Sec. 4 discusses the reliance on calibrated stereo input, the static-scene assumption, and the runtime of the stereo backbone, which is analyzed further in the Supplementary Material.

   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 include theoretical 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: We provide the information regarding to the experimental setup and datasets used in this paper in the Experiments section and Supplementary Material.

   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 code is not yet publicly available, and we are preparing its release on the project page. The paper and the Supplementary Material provide extensive implementation details to support reproducibility.

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: All of the training and test details are reported in the Experiments section and Supplementary Material.

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: Following common practice in feed-forward novel view synthesis, we report results without error bars because training each model multiple times is computationally expensive.

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 Supplementary Material reports the training compute (approximately 240 GPU-hours on NVIDIA H100) and a runtime and memory analysis on a single NVIDIA A100-80GB GPU.

   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: We have reviewed and followed 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 poses no such risks.

    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: We reported all public datasets and models used in this paper in the Experiments section.

    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: [Yes]

    Justification: New assets are described in the Method section and Supplementary Material.

    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 paper does not involve crowdsourcing nor 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 paper does not involve crowdsourcing nor research with human subjects.

    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: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    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.