pixelSplat: 3D Gaussian Splats from Image Pairs for Scalable Generalizable 3D Reconstruction

David Charatan[1] Sizhe Lester Li[1] Andrea Tagliasacchi[2] Vincent Sitzmann[1]
[1]Massachusetts Institute of Technology [2]Simon Fraser University, University of Toronto
{charatan, sizheli, sitzmann}@mit.edu andrea.tagliasacchi@sfu.ca

Abstract

We introduce pixelSplat, a feed-forward model that learns to reconstruct 3D radiance fields parameterized by 3D Gaussian primitives from pairs of images. Our model features real-time and memory-efficient rendering for scalable training as well as fast 3D reconstruction at inference time. To overcome local minima inherent to sparse and locally supported representations, we predict a dense probability distribution over 3D and sample Gaussian means from that probability distribution. We make this sampling operation differentiable via a reparameterization trick, allowing us to back-propagate gradients through the Gaussian splatting representation. We benchmark our method on wide-baseline novel view synthesis on the real-world RealEstate10k and ACID datasets, where we outperform state-of-the-art light field transformers and accelerate rendering by 2.5 orders of magnitude while reconstructing an interpretable and editable 3D radiance field. Additional materials can be found on the project website. [1]

1. Introduction

We investigate the problem of generalizable novel view synthesis from sparse image observations. This line of work has been revolutionized by differentiable rendering [29, 40, 41, 50] but has also inherited its key weakness: training, reconstruction, and rendering are notoriously memory- and time-intensive because differentiable rendering requires evaluating dozens or hundreds of points along each camera ray [58].

This has motivated light-field transformers [10, 37, 43, 47], where a ray is rendered by embedding it into a query token and a color is obtained via cross-attention over image tokens. While significantly faster than volume rendering, such methods are still far from real-time. Additionally, they do not reconstruct 3D scene representations that can be edited or exported for downstream tasks in vision and graphics.

[1]dcharatan.github.io/pixelsplat

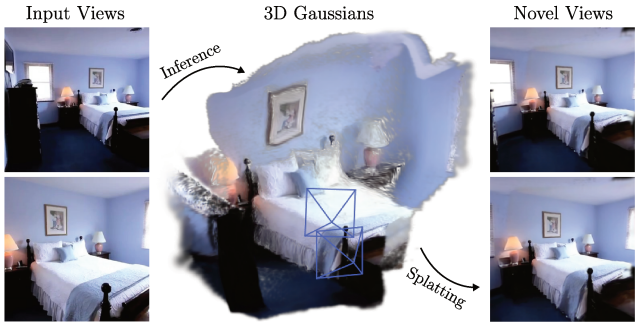

Figure 1. **Overview.** Given a pair of input images, pixelSplat reconstructs a 3D radiance field parameterized via 3D Gaussian primitives. This yields an explicit 3D representation that is renderable in real time, remains editable, and is cheap to train.

Meanwhile, recent work on single-scene novel view synthesis has shown that it is possible to use 3D Gaussian primitives to enable real-time rendering with little memory cost via rasterization-based volume rendering [19].

We present pixelSplat, which brings the benefits of a primitive-based 3D representation—fast and memory-efficient rendering as well as interpretable 3D structure—to generalizable view synthesis. This is no straightforward task. First, in real-world datasets, camera poses are only reconstructed up to an arbitrary scale factor. We address this by designing a multi-view epipolar transformer that reliably infers this per-scene scale factor. Next, optimizing primitive parameters directly via gradient descent suffers from local minima. In the single-scene case, this can be addressed via non-differentiable pruning and division heuristics [19]. In contrast, in the generalizable case, we need to back-propagate gradients through the representation and thus cannot rely on non-differentiable operations. We thus propose a method by which Gaussian primitives can implicitly be spawned or deleted during training, avoiding local minima, but which nevertheless maintains gradient flow. Specifically, we parameterize the positions (i.e., means) of Gaussians *implicitly* via dense probability distributions predicted by our encoder. In each forward pass, we sample

Gaussian primitive locations from this distribution. We make the sampling operation differentiable via a reparameterization trick that couples the density of a sampled Gaussian primitive to the probability of that location. When receiving a gradient that would increase the opacity of a Gaussian at a 3D location, our model increases the probability that the Gaussian will be sampled at that location again in the future.

We demonstrate the efficacy of our method by showcasing, for the first time, how a 3D Gaussian splatting representation can be predicted in a *single forward pass* from just a pair of images. In other words, we demonstrate how 3D Gaussians can be integrated in an end-to-end differentiable system. We significantly outperform previous black-box based light field transformers on the real-world ACID and RealEstate10k datasets while drastically reducing both training and rendering cost and generating explicit 3D scenes.

2. Related Work

Single-scene novel view synthesis. Advancements in neural rendering [50] and neural fields [29, 42, 57] have revolutionized 3D reconstruction and novel view synthesis from collections of posed images. Recent approaches generally create 3D scene representations by backpropagating image-space photometric error through differentiable renderers. Early methods employed voxel grids and learned rendering techniques [27, 31, 40], and more recently, neural fields [2, 28, 29, 57] and volume rendering [27, 29, 49] have become the de-facto standard. However, a key hurdle of these methods is their high computational demand, as rendering usually requires dozens of queries of the neural field per ray. Discrete data structures can accelerate rendering [6, 12, 25, 30] but fall short of real-time rendering at high resolutions. 3D Gaussian splatting [19] solves this problem by representing the radiance field using 3D Gaussians that can efficiently be rendered via rasterization. However, all single-scene optimization methods require dozens of images to achieve high-quality novel view synthesis. In this work, we train neural networks to estimate the parameters of a 3D Gaussian primitive scene representation from just two images in a single forward pass.

Prior-based 3D Reconstruction and View Synthesis. Generalizable novel view synthesis seeks to enable 3D reconstruction and novel view synthesis from only a handful of images per scene. If proxy geometry (e.g., depth maps) is available, machine learning can be combined with image-based rendering [1, 22, 36, 56] to produce convincing results. Neural networks can also be trained to directly regress multi-plane images for small-baseline novel view synthesis [45, 53, 60, 61]. Large-baseline novel view synthesis, however, requires full 3D representations. Early approaches based on neural fields [32, 41] encoded 3D scenes in individual latent codes and were thus limited to single-object scenes. Preserving end-to-end locality and shift equivariance between encoder and scene representation via pixel-aligned features [14, 23, 39, 52, 58] or via transformers [35, 54] has enabled generalization to unbounded scenes. Inspired by classical multi-view stereo, neural networks have also been combined with cost volumes to match features across views [5, 7, 18, 26]. While the above methods infer interpretable 3D representations in the form of signed distances or radiance fields, recent light field scene representations trade interpretability for faster rendering [10, 37, 43, 46, 47]. Our method presents the best of both worlds: it infers an interpretable 3D scene representation in the form of 3D Gaussians while accelerating rendering by three orders of magnitude compared to light field transformers.

Scale ambiguity in machine learning for multi-view geometry. Prior work has recognized the challenge of scene scale ambiguity. In monocular depth estimation, state-of-the-art models rely on sophisticated scale-invariant depth losses [11, 13, 33, 34]. In novel view synthesis, recent single-image 3D diffusion models trained on real-world data rescale 3D scenes according to heuristics on depth statistics and condition their encoders on scene scale [4, 38, 51]. In this work, we instead build a multi-view encoder that can infer the scale of the scene. We accomplish this using an epipolar transformer that finds cross-view pixel correspondences and associates them with positionally encoded depth values [16].

3. Background: 3D Gaussian Splatting

3D Gaussian Splatting [19], which we will refer to as 3D-GS, parameterizes a 3D scene as a set of 3D Gaussian primitives $\{\mathbf{g}_k{=}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \boldsymbol{\alpha}_k, \mathbf{S}_k)\}_k^K$ which each have a mean $\boldsymbol{\mu}_k$, a covariance $\boldsymbol{\Sigma}_k$, an opacity $\boldsymbol{\alpha}_k$, and spherical harmonics coefficients $\mathbf{S}_k$. These primitives parameterize the 3D radiance field of the underlying scene and can be rendered to produce novel views. However, unlike dense representations like neural fields [29] and voxel grids [12], Gaussian primitives can be rendered via an inexpensive rasterization operation [33]. Compared to the sampling-based approach used to render dense fields, this approach is significantly cheaper in terms of time and memory.

Local minima. A key challenge of function fitting with primitives is their susceptibility to local minima. The fitting of a 3D-GS model is closely related to the fitting of a Gaussian mixture model, where we seek the parameters of a set of Gaussians such that we maximize the likelihood of a set of samples. This problem is famously non-convex and generally solved with the Expectation-Maximization (EM) algorithm [8]. However, the EM algorithm still suffers from local minima [17] and is not applicable to inverse graphics, where only images of the 3D scene are provided and not ground-truth 3D volume density. In 3D-GS, local minima arise when Gaussian primitives initialized at random locations have to

move through space to arrive at their final location. Two issues prevent this: first, Gaussian primitives have local support, meaning that gradients vanish if the distance to the correct location exceeds more than a few standard deviations. Second, even if a Gaussian is close enough to a "correct" location to receive substantial gradients, there still needs to exist a path to its final location along which loss decreases monotonically. In the context of differentiable rendering, this is generally not the case, as Gaussians often have to traverse empty space where they may occlude background features. 3D-GS relies on non-differentiable pruning and splitting operations dubbed "Adaptive Density Control" (see Sec. 5 of [19]) to address this problem. However, these techniques are incompatible with the generalizable setting, where primitive parameters are predicted by a neural network that must receive gradients. In section 4.2, we thus propose a differentiable parameterization of Gaussian primitives that is not susceptible to local minima, allowing its use as a potential building block in larger end-to-end differentiable models.

4. Image-conditioned 3D Gaussian Inference

We present pixelSplat, a Gaussian-based generalizable novel view synthesis model. Given a pair of images and their associated camera parameters, our method infers a 3D Gaussian representation of the underlying scene, which can be rendered to produce images of unseen viewpoints. Our method consists of a two-view image encoder and a pixel-aligned Gaussian prediction module. Section 4.1 introduces our image encoding scheme. We show that our image encoding scheme addresses scale ambiguity, which is a key challenge during Gaussian inference from real-world images. Section 4.2 describes how our model predicts 3D Gaussian parameters, emphasizing how it overcomes the issue of local minima described in Section 3.

4.1. Resolving Scale Ambiguity

In an ideal world, novel-view-synthesis datasets would contain camera poses that are *metric*. If this were the case, each scene $\mathcal{C}_i^{\text{m}}$ would consist of a series of tuples $\mathcal{C}_i^{\text{m}} = \{(\mathbf{I}_j, \mathbf{T}_j^{\text{m}})\}_j$ containing images $\mathbf{I}_j$ and corresponding real-world-scale poses $\mathbf{T}_j^{\text{m}}$. However, in practice, such datasets provide poses that are computed using structure-from-motion (SfM) software. SfM reconstructs each scene *only up to scale*, meaning that different scenes $\mathcal{C}_i$ are scaled by individual, arbitrary scale factors s_i. A given scene $\mathcal{C}_i$ thus provides $\mathcal{C}_i = \{(\mathbf{I}_j, s_i\mathbf{T}_j^{\text{m}})\}_j$, where $s_i\mathbf{T}_j^{\text{m}}$ denotes a metric pose whose translation component is scaled by the unknown scalar $s_i \in \mathbb{R}^+$. Note that recovering s_i from a single image is *impossible* due to the principle of scale ambiguity. In practice, this means that a neural network making predictions about the geometry of a scene from a single image *cannot possibly* predict the depth that matches the poses reconstructed by structure-from-motion. In monocular depth estimation, this has been addressed via scale-invariant losses [11, 13, 33]. Our encoder similarly has to predict the geometry of the scene, chiefly via the position of each Gaussian primitive, which depends on the per-scene scale s_i. Refer to Figure 2 for an illustration of this challenge.

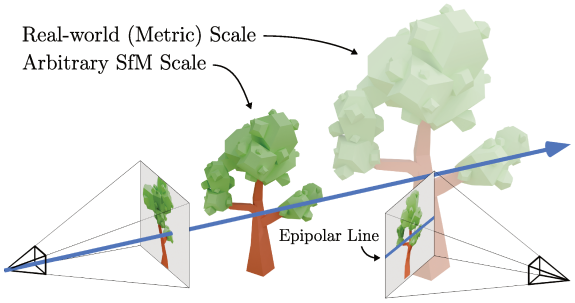

Figure 2. **Scale ambiguity.** SfM does not reconstruct camera poses in real-world, metric scale—poses are scaled by an arbitrary scale factor that is different for each scene. To render correct views, our model's 3D reconstruction needs to be consistent with this arbitrary scale. We illustrate how our epipolar encoder solves this problem. Features belonging to the ray's corresponding pixel on the left are compared with features sampled along the epipolar line on the right. Epipolar samples are augmented with their positionally-encoded depths along the ray, which allows our encoder to record correct depths. Recorded depths are later used for depth prediction.

We thus propose a two-view encoder to resolve scale ambiguity and demonstrate in our ablations (Table 2) that this is absolutely critical for 3D-structured novel view synthesis. Let us denote the two reference views as $\mathbf{I}$ and $\tilde{\mathbf{I}}$. Intuitively, for each pixel in $\mathbf{I}$, we will annotate points along its epipolar line in $\tilde{\mathbf{I}}$ with their corresponding depths in $\mathbf{I}$'s coordinate frame. Note that these depth values are computed from $\mathbf{I}$ and $\tilde{\mathbf{I}}$'s camera poses, and thus encode the scene's scale s_i. Our encoder then finds per-pixel correspondence via epipolar attention [16] and memorizes the corresponding depth for that pixel. The depths of pixels without correspondences in $\tilde{\mathbf{I}}$ are in-painted via per-image self-attention. In the following paragraphs, we discuss these steps in detail.

We first encode each view separately into feature volumes $\mathbf{F}$ and $\tilde{\mathbf{F}}$ via a per-image feature encoder. Let $\mathbf{u}$ be pixel coordinates from $\mathbf{I}$, and ℓ be the epipolar line induced by its ray in $\tilde{\mathbf{I}}$, i.e., the projection of $\mathbf{u}$'s camera ray onto the image plane of $\tilde{\mathbf{I}}$. Along ℓ, we now sample pixel coordinates $\{\tilde{\mathbf{u}}_l\} \sim \tilde{\mathbf{I}}$. For each epipolar line sample $\tilde{\mathbf{u}}_l$, we further compute its distance to $\mathbf{I}$'s camera origin $\tilde{d}_{\tilde{\mathbf{u}}_l}$ by triangulation of $\mathbf{u}$ and $\tilde{\mathbf{u}}_l$. We then compute queries, keys and values for epipolar attention as follows:

$$\mathbf{s} = \tilde{\mathbf{F}}[\tilde{\mathbf{u}}_l] \oplus \gamma(\tilde{d}_{\tilde{\mathbf{u}}_l}) \tag{1}$$

$$\mathbf{q} = \mathbf{Q} \cdot \mathbf{F}[\mathbf{u}], \quad \mathbf{k}_l = \mathbf{K} \cdot \mathbf{s}, \quad \mathbf{v}_l = \mathbf{V} \cdot \mathbf{s}, \tag{2}$$

where, $\oplus$ denotes concatenation, $\gamma(\cdot)$ denotes positional encoding, and $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ are query, key and value embedding matrices. We now perform epipolar cross-attention and up-

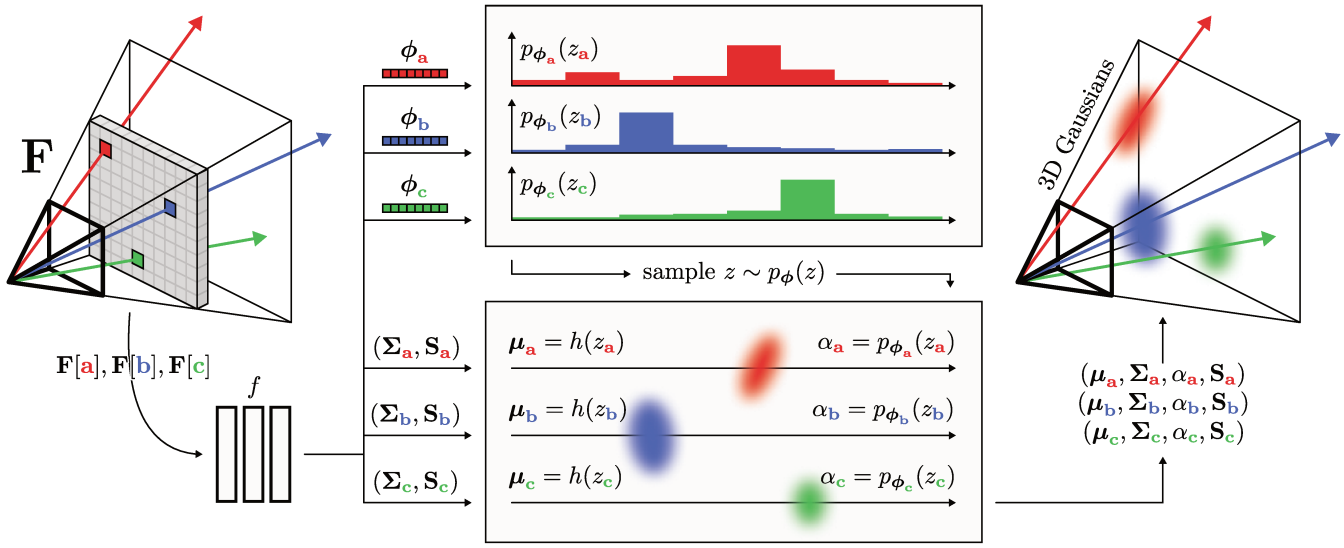

Figure 3. **Proposed probabilistic prediction of pixel-aligned Gaussians.** For every pixel feature $\mathbf{F}[\mathbf{u}]$ in the input feature map, a neural network f predicts Gaussian primitive parameters $\boldsymbol{\Sigma}$ and $\mathbf{S}$. Gaussian locations $\boldsymbol{\mu}$ and opacities $\boldsymbol{\alpha}$ are not predicted directly, which would lead to local minima. Instead, f predicts per-pixel discrete probability distributions over depths $p_{\boldsymbol{\phi}}(z)$, parameterized by $\boldsymbol{\phi}$. Sampling then yields the locations of Gaussian primitives. The opacity of each Gaussian is set to the probability of the sampled depth bucket. The final set of Gaussian primitives can then be rendered from novel views using the splatting algorithm proposed by Kerbl et al. [19]. Note that for brevity, we use h to represent the function that computes depths from bucket indices (see equations 6 and 7).

date per-pixel features $\mathbf{F}[\mathbf{u}]$ as:

$$\mathbf{F}[\mathbf{u}] \mathrel{+}= \text{Att}(\mathbf{q}, \{\mathbf{k}_l\}, \{\mathbf{v}_l\}), \tag{3}$$

where Att denotes softmax attention, and $+=$ denotes a skip connection. After this epipolar attention layer, each pixel feature $\mathbf{F}[\mathbf{u}]$ contains a weighted sum of the depth positional encodings, where we expect (and experimentally confirm in Sec. 5.3) that the correct correspondence gained the largest weight, and that thus, each pixel feature $\mathbf{F}[\mathbf{u}]$ now encodes the scaled depth that is consistent with the arbitrary scale factor s_i of the camera poses. This epipolar cross-attention layer is followed by a per-image self-attention layer,

$$\mathbf{F} \mathrel{+}= \text{SelfAttention}(\mathbf{F}). \tag{4}$$

This enables our encoder to propagate scaled depth estimates to parts of the image feature maps that may not have any epipolar correspondences in the opposite image.

Note that this mechanism can be extended to more than two input views. See the supplemental material for details.

4.2. Gaussian Parameter Prediction

In this step, we leverage our scale-aware feature maps to predict the parameters of a set of Gaussian primitives $\{\mathbf{g}_k = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \boldsymbol{\alpha}_k, \mathbf{S}_k)\}_k^K$ that parameterize the scene's radiance field. Every pixel in an image samples a point on a surface in the 3D scene. We thus choose to parameterize the scene via *pixel-aligned* Gaussians: for each pixel at coordinate $\mathbf{u}$, we take the corresponding feature $\mathbf{F}[\mathbf{u}]$ as input and predict the parameters of M Gaussian primitives. In the following, we discuss the case of $M{=}1$ primitives per pixel for simplicity; hence we aim to predict one tuple of values $(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\alpha}, \mathbf{S})$ per pixel. As we will see, the most consequential question is how to parameterize the position $\boldsymbol{\mu}$ of each Gaussian. Figure 3 illustrates the process by which we predict parameters of pixel-aligned Gaussians from a single reference image.

Baseline: predicting a point estimate of $\boldsymbol{\mu}$. We first consider directly regressing a Gaussian's center $\boldsymbol{\mu}$. This means implementing a neural network g that regresses the distance $d \in \mathbb{R}^+$ of the Gaussian's mean from the camera origin, and

Algorithm 1 Probabilistic Prediction of a Pixel-Aligned Gaussian.

Require: Depth buckets $\mathbf{b} \in \mathbb{R}^Z$, feature $\mathbf{F}[\mathbf{u}]$ at pixel coordinate $\mathbf{u}$, camera origin of reference view $\mathbf{o}$, ray direction $\mathbf{d}_\mathbf{u}$.

1: $(\boldsymbol{\phi}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{S}) = f(\mathbf{F}[\mathbf{u}])$ ▷ predict depth probabilities $\boldsymbol{\phi}$ and offsets $\boldsymbol{\delta}$, covariance $\boldsymbol{\Sigma}$, spherical harmonics coefficients $\mathbf{S}$
2: $z \sim p_{\boldsymbol{\phi}}(z)$ ▷ Sample depth bucket index z from discrete probability distribution parameterized by $\boldsymbol{\phi}$
3: $\boldsymbol{\mu} = \mathbf{o} + (\mathbf{b}_z + \boldsymbol{\delta}_z)\mathbf{d}_\mathbf{u}$ ▷ Compute Gaussian mean $\boldsymbol{\mu}$ by unprojecting with depth $\mathbf{b}_z$ adjusted by bucket offset $\boldsymbol{\delta}_z$
4: $\boldsymbol{\alpha} = \boldsymbol{\phi}_z$ ▷ Set Gaussian opacity $\boldsymbol{\alpha}$ according to probability of sampled depth (Sec. 4.2).
5: **return** $(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\alpha}, \mathbf{S})$

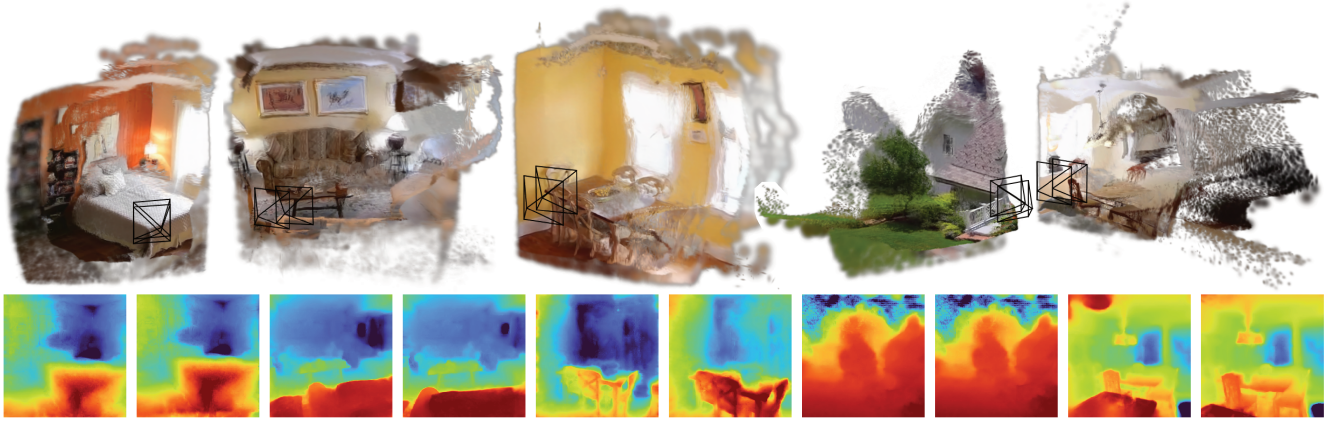

Figure 4. **3D Gaussians (top) and corresponding depth maps (bottom) predicted by our method.** In contrast to light field rendering methods like GPNR [47] and that of Du et al. [10], our method produces an *explicit* 3D representation. Here, we show zoomed-out views of the Gaussians our method produces along with rendered depth maps, as viewed from the two reference viewpoints.

then unprojecting it to 3D:

$$\boldsymbol{\mu} = \mathbf{o} + d_{\mathbf{u}}\,\mathbf{d},\;\; d = g(\mathbf{F}[\mathbf{u}]),\;\; \mathbf{d} = \mathbf{T}\mathbf{K}^{-1}[\mathbf{u},1]^T \quad (5)$$

where the ray direction $\mathbf{d}$ is computed from the camera extrinsics $\mathbf{T}$ and intrinsics $\mathbf{K}$. Unfortunately, directly optimizing Gaussian parameters is susceptible to local minima, as discussed in Section 3. Further, as we are back-propagating through the representation, we cannot rely on the spawning and pruning heuristics proposed in 3D-GS, as they are not differentiable. Instead, we propose a differentiable alternative that succeeds in overcoming local minima. In Table 2, we demonstrate that this leads to ≈1dB of PSNR performance boost.

Proposed: predicting a probability density of μ. Rather than predicting the depth d of a Gaussian, we predict the probability distribution of the likelihood that a Gaussian exists at a depth d along the ray $\mathbf{u}$. We implement this as a discrete probability density over a set of depth buckets. We set near and far planes d_{near} and d_{far}. Between these, we discretize depth into Z bins, represented by a vector $\mathbf{b} \in \mathbb{R}^Z$, where its z-th element $\mathbf{b}_z$ is defined in disparity space as

$$\mathbf{b}_z = \left(\left(1 - \frac{z}{Z}\right)\left(\frac{1}{d_{\text{near}}} - \frac{1}{d_{\text{far}}}\right) + \frac{1}{d_{\text{far}}}\right)^{-1}. \quad (6)$$

We then define a discrete probability distribution $p_{\boldsymbol{\phi}}(z)$ over an index variable z, parameterized by a vector of discrete probabilities $\boldsymbol{\phi}$, where its z-th element $\boldsymbol{\phi}_z$ is the probability that a surface exists in depth bucket $\mathbf{b}_z$. Probabilities $\boldsymbol{\phi}$ are predicted by a fully connected neural network f from per-pixel feature $\mathbf{F}[\mathbf{u}]$ at pixel coordinate $\mathbf{u}$ and normalized to sum to one via a softmax. Further, we predict a per-bucket center offset $\boldsymbol{\delta} \in [0,1]^Z$ that adjusts the depth of the pixel-aligned Gaussian between bucket boundaries:

$$\boldsymbol{\mu} = \mathbf{o} + (\mathbf{b}_z + \boldsymbol{\delta}_z)\,\mathbf{d}_{\mathbf{u}},\;\; z \sim p_{\boldsymbol{\phi}}(z),\;\; (\boldsymbol{\phi}, \boldsymbol{\delta}) = f(\mathbf{F}[\mathbf{u}]) \quad (7)$$

We note that, different from (5), during a forward pass a Gaussian location is *sampled* from the distribution $z{\sim}p_{\boldsymbol{\phi}}(z)$, and the network predicts the *probabilities* $\boldsymbol{\phi}$ instead of the depth d directly.

Making sampling differentiable by setting $\alpha = \phi_z$. To train our model, we need to backpropagate gradients into the probabilities of the depth buckets $\boldsymbol{\phi}$. This means we need to compute the derivatives $\nabla_{\boldsymbol{\phi}}\boldsymbol{\mu}$ of the mean $\boldsymbol{\mu}$ with respect to probabilities $\boldsymbol{\phi}$. Unfortunately, the sampling operation $z{\sim}p_{\boldsymbol{\phi}}(z)$ is not differentiable. Inspired by variational autoencoders [21], we overcome this challenge via a reparameterization trick. Accordingly, we set the opacity $\boldsymbol{\alpha}$ of a Gaussian to be equal to the probability of the bucket that it was sampled from. Consider that we sampled $z{\sim}p_{\boldsymbol{\phi}}(z)$. We then set $\boldsymbol{\alpha} = \boldsymbol{\phi}_z$, i.e., we set $\boldsymbol{\alpha}$ to the z-th entry of the vector of probabilities $\boldsymbol{\phi}$. This means that in each backward pass, we assign the gradients of the loss $\mathcal{L}$ with respect to the opacities $\boldsymbol{\alpha}$ to the gradients of the depth probability buckets $\boldsymbol{\phi}$, i.e., $\nabla_{\boldsymbol{\phi}}\mathcal{L} = \nabla_{\boldsymbol{\alpha}}\mathcal{L}$.

To understand this approach at an intuitive level, consider the case where sampling produces a correct depth. In this case, gradient descent *increases* the opacity of the Gaussian, leading it to be sampled more often. This eventually concentrates all probability mass in the correct bucket, creating a perfectly opaque surface. Now, consider the case of an incorrectly sampled depth. In this case, gradient descent *decreases* the opacity of the Gaussian, lowering the probability of further incorrect depth predictions.

Predicting $\boldsymbol{\Sigma}$ and $\mathbf{S}$. We predict a single covariance matrix and set of spherical harmonics coefficients per pixel by

	ACID			RealEstate10k			Inference Time (s)		Memory (GB)	
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Encode ↓	Render ↓	Training ↓	Inference ↓
Ours	**28.27**	**0.843**	**0.146**	**26.09**	**0.863**	**0.136**	0.102	**0.002**	**14.4**	**3.002**
Du et al. [10]	26.88	0.799	0.218	24.78	0.820	0.213	0.016	1.309	314.3	19.604
GPNR [46]	25.28	0.764	0.332	24.11	0.793	0.255	**N/A**	13.340	3789.9	19.441
pixelNeRF [58]	20.97	0.547	0.533	20.43	0.589	0.550	0.005	5.294	436.7	3.962

Table 1. **Quantitative comparisons.** We outperform all baseline methods in terms PSNR, LPIPS, and SSIM for novel view synthesis on the real-world RealEstate10k and ACID datasets. In addition, our method requires less memory during both inference and training and renders images about 650 times faster than the next-fastest baseline. In the memory column, we report memory usage for a single scene and 256×256 rays, extrapolating from the smaller number of rays per batch used to train the baselines where necessary. Note that we report GPNR's encoding time as N/A because it has no encoder. We **bold first-place** results and underline second-place results in each column.

extending the neural network f as

$$\phi, \delta, \Sigma, \mathbf{S} = f(\mathbf{F}[\mathbf{u}]). \tag{8}$$

Summary. Algorithm 1 provides a summary of the procedure that predicts the parameters $(\mu, \Sigma, \alpha, \mathbf{S})$ of a pixel-aligned Gaussian primitive from the corresponding pixel's feature $\mathbf{F}[\mathbf{u}]$.

5. Experiments

In this section, we describe our experimental setup, evaluate our method on wide-baseline novel view synthesis from image pairs, and perform ablations to validate our design.

5.1. Experimental Setup

We train and evaluate our method on RealEstate10k [61], a dataset of home walkthrough videos downloaded from YouTube, as well as ACID [24], a dataset of aerial landscape videos. Both datasets include camera poses computed by SfM software, necessitating the scale-aware design discussed in Section 4.1. We use the provided training and testing splits. Because the prior state-of-the-art wide-baseline novel view synthesis model by Du et al. [10] only supports a resolution of 256×256, we train and evaluate our model at this resolution. We evaluate our model on its ability to reconstruct video frames between two frames chosen as reference views.

Baselines. We compare our method against three novel-view-synthesis baselines. pixelNeRF [58] conditions neural radiance fields on 2D image features. Generalizable Patch-based Neural Rendering (GPNR) [46] is an image-based light field rendering method that computes novel views by aggregating transformer tokens sampled along epipolar lines. The unnamed method of Du et al. [10] also combines light field rendering with an epipolar transformer, but additionally uses a multi-view self-attention encoder and proposes a more efficient approach for sampling along epipolar lines. To present a fair comparison, we retrained these baselines by combining their publicly available codebases with our datasets and our method's data loaders. We train all methods, including ours, using the same training curriculum, where we gradually increase the inter-frame distance between reference views as training progresses. For further training details, consult the supplementary material.

Evaluation Metrics. To evaluate visual fidelity, we compare each method's rendered images to the corresponding ground-truth frames by computing a peak signal-to-noise ratio (PSNR), structural similarity index (SSIM) [55], and perceptual distance (LPIPS) [59]. We further evaluate each method's resource demands. In this comparison, we distinguish between the encoding time, which is incurred once per scene and amortized over rendered views, and decoding time, which is incurred once per frame.

Implementation details. Each reference image is passed through a ResNet-50 [15] and a ViT-B/8 vision transformer [9] that have both been pre-trained using a DINO objective [3]; we sum their pixel-wise outputs. We train our model to minimize a combination of MSE and LPIPS losses using the Adam optimizer [20]. For the "Plus Depth Regularization" ablation, we regularize depth maps by fine-tuning with 50,000 steps of edge-aware total variation regularization. Our encoder performs two rounds of epipolar cross-attention.

5.2. Results

We report quantitative results in Table 1. Our method outperforms the baselines on all metrics, with especially significant improvements in perceptual distance (LPIPS). We show qualitative results in Fig. 5. Compared to the baselines, our method is better at capturing fine details and correctly inferring 3D structure in portions of each scene that are only observed by one reference view.

Training and Inference cost. As shown in Table 1, our method is *significantly* less resource-intensive than the baselines. Compared to the next-fastest one, our method's cost to infer a single scene (encoding) and then render 100 images (decoding), the approximate number in a RealEstate10k or ACID sequence, is about 650 times less. Our method also uses significantly less memory per ray at training time.

Point cloud rendering. To qualitatively evaluate our method's ability to infer a structured 3D representation, we

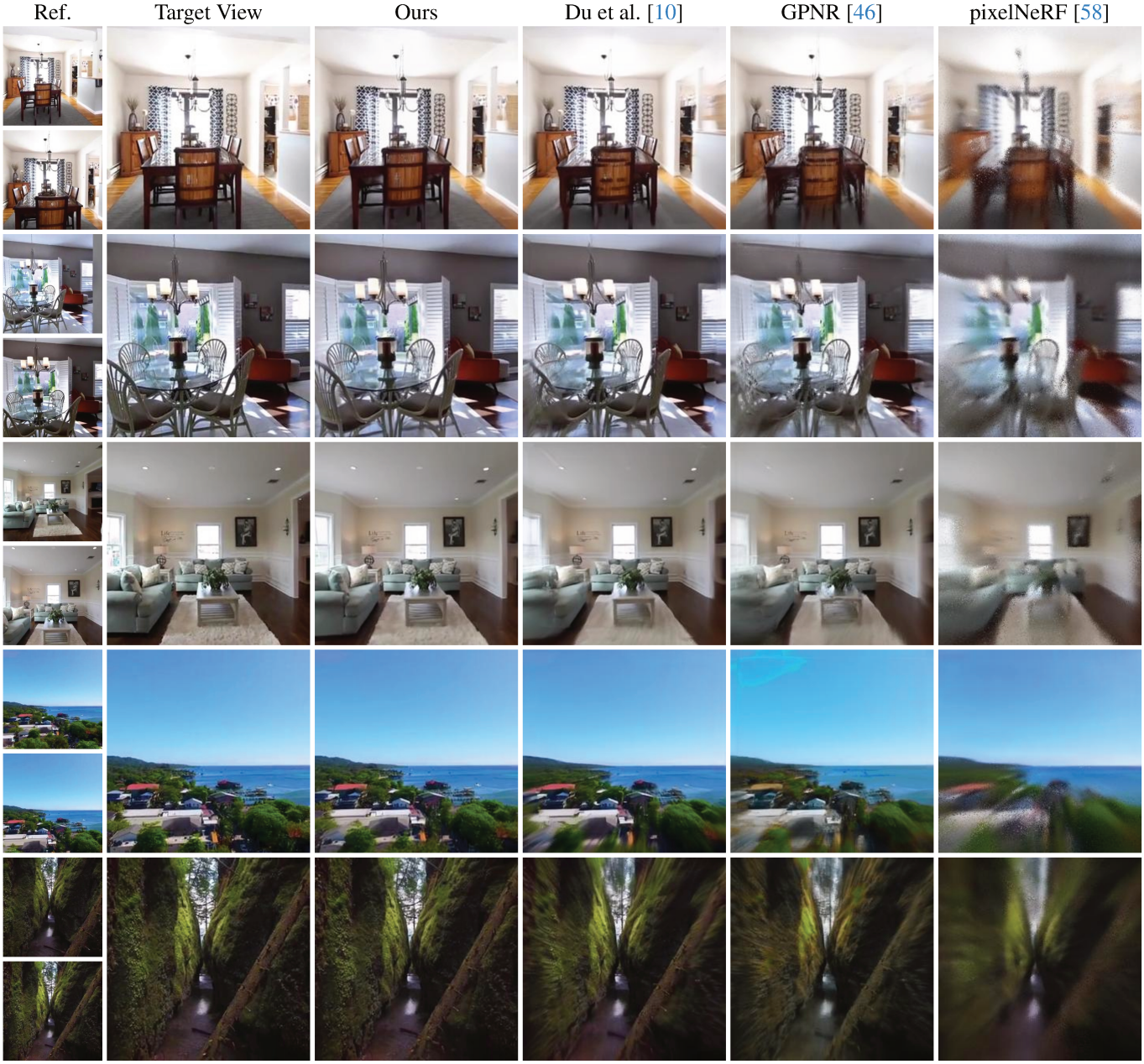

Figure 5. **Qualitative comparison of novel views on the RealEstate10k (top) and ACID (bottom) test sets.** Compared to the baselines, our approach not only produces more accurate and perceptually appealing images, but also generalizes better to out-of-distribution examples like the creek in the bottom row.

visualize the Gaussians it produces from views that are far outside the training distribution in Figure 4. We visualize point clouds using the version of our model that has been fine-tuned with a depth regularizer. Note that while the resulting Gaussians facilitate high-fidelity novel-view synthesis for in-distribution camera poses, they suffer from the same failure modes as 3D Gaussians optimized using the original 3D Gaussian splatting method [19]. Specifically, reflective surfaces are often transparent, and Gaussians appear billboard-like when viewed from out-of-distribution views.

5.3. Ablations and Analysis

We perform ablations on RealEstate10k to answer the following questions:

- Question 1a: Is our epipolar encoder responsible for our model's ability to handle scale ambiguity?
- Question 1b: If so, by what mechanism does our model handle scale ambiguity?

Figure 6. **Attention visualization.** We visualize the epipolar cross-attention weights between the rays on the left and the corresponding epipolar lines on the right to confirm that our model learns to find the correct correspondence along each ray.

- Question 2: Does our probabilistic primitive prediction alleviate local minima in primitive regression?

See quantitative results in Tab. 2 and qualitative ones in Fig. 7.

Ablating epipolar encoding (Q1a). To measure our epipolar encoding scheme's importance, we compare pixelSplat to a variant (No Epipolar Encoder) that eschews epipolar encoding. Here, features are generated by encoding each reference view independently. Qualitatively, this produces ghosting and motion blur artifacts that are evidence of incorrect depth predictions; quantitatively, performance drops significantly. In Figure 6, we visualize epipolar attention scores, demonstrating that our epipolar transformer successfully discovers cross-view correspondences.

Importance of depth for epipolar encoding (Q1b). We investigate whether the frequency-based positional encoding of depth in Eq. 2 is necessary for our proposed epipolar layer. We perform a study (No Depth Encoding) where only image features $\mathbf{F}$ are fed to the epipolar attention layer. This leads to a performance drop of $\approx$ 1dB PSNR. This highlights that beyond simply detecting correspondence, our encoder uses the scene-scale encoded depths it triangulates to resolve scale ambiguity.

Importance of probabilistic prediction of Gaussian depths (Q2). To investigate whether predicting the depth of a Gaussian probabilistically is necessary, we perform an ablation (No Probabilistic Prediction) which directly regresses the depth, and thus the mean μ, of a pixel-aligned Gaussian with a neural network. We observe a performance drop of $\approx$ 1.5dB in PSNR.

6. Conclusion

We have introduced pixelSplat, a method that reconstructs a primitive-based parameterization of the 3D radiance field of a scene from only two images. At inference time, our method is significantly faster than prior work on generalizable novel view synthesis while producing an explicit 3D scene representation. To solve the problem of local minima that arises in primitive-based function regression, we introduced a novel parameterization of primitive location via a dense probability distribution and introduced a novel reparameterization trick to backpropagate gradients into the parameters of this distribution. This framework is general, and we hope that our work inspires follow-up work on prior-based inference of primitive-based representations across applications. An exciting avenue for future work is to leverage our model for generative modeling by combining it with diffusion models [48, 51] or to remove the need for camera poses to enable large-scale training [44].

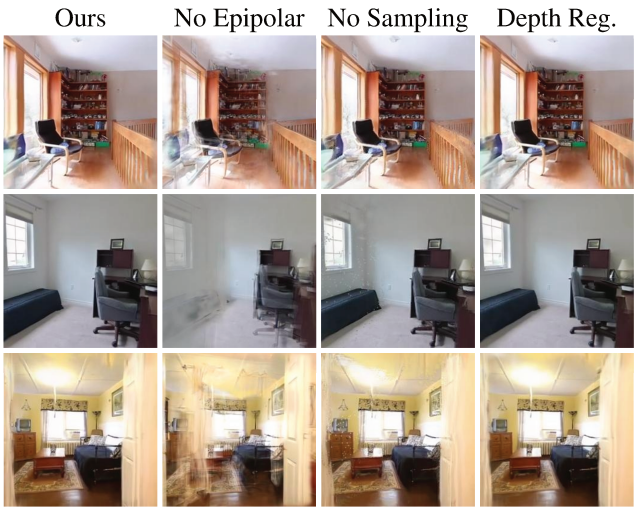

Figure 7. **Ablations.** Without the epipolar transformer, our model is unable to resolve scale ambiguity, leading to ghosting artifacts. Without our sampling approach, our model falls into local minima that manifest themselves as speckling artifacts. Regularizing our model's predicted depths minimally affects rendering quality.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Ours	**26.09**	**0.863**	**0.136**
No Epipolar Encoder	19.89	0.639	0.295
No Depth Encoding	24.97	0.830	0.158
No Probabilistic Sampling	24.62	0.828	0.171
Plus Depth Regularization	<u>25.28</u>	<u>0.847</u>	<u>0.152</u>

Table 2. **Ablations.** Both our epipolar encoder and our probabilistic sampling scheme are essential for high-quality novel view synthesis. Depth regularization slightly impacts rendering quality.

Acknowledgements. This work was supported by the National Science Foundation under Grant No. 2211259, by the Singapore DSTA under DST00OECI20300823 (New Representations for Vision), by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DOI/IBC) under 140D0423C0075, by the Amazon Science Hub, and by IBM. The Toyota Research Institute also partially supported this work. The views and conclusions contained herein reflect the opinions and conclusions of its authors and no other entity.

References

[1] Kara-Ali Aliev, Artem Sevastopolsky, Maria Kolos, Dmitry Ulyanov, and Victor Lempitsky. Neural point-based graphics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 696–712. Springer, 2020. 2

[2] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 2

[3] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 6

[4] Eric R Chan, Koki Nagano, Matthew A Chan, Alexander W Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. Generative novel view synthesis with 3d-aware diffusion models. *Proceedings of the International Conference on 3D Vision (3DV)*, 2023. 2

[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. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 14124–14133, 2021. 2

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. *arXiv preprint arXiv:2203.09517*, 2022. 2

[7] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2021. 2

[8] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society: series B (methodological)*, 39(1):1–22, 1977. 2

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. 6

[10] Yilun Du, Cameron Smith, Ayush Tewari, and Vincent Sitzmann. Learning to render novel views from wide-baseline stereo pairs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 5, 6, 7

[11] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 2, 3

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5501–5510, 2022. 2

[13] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3

[14] Pengsheng Guo, Miguel Angel Bautista, Alex Colburn, Liang Yang, Daniel Ulbricht, Joshua M Susskind, and Qi Shan. Fast and explicit neural view synthesis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3791–3800, 2022. 2

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 6

[16] Yihui He, Rui Yan, Katerina Fragkiadaki, and Shoou-I Yu. Epipolar transformers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 3

[17] Chi Jin, Yuchen Zhang, Sivaraman Balakrishnan, Martin J Wainwright, and Michael I Jordan. Local maxima in the likelihood of gaussian mixture models: Structural results and algorithmic consequences. *Advances in Neural Information Processing Systems (NeurIPS)*, 29, 2016. 2

[18] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18365–18375, 2022. 2

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4): 1–14, 2023. 1, 2, 3, 4, 7

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 6

[21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. 5

[22] Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. In *Computer Graphics Forum*, pages 29–43. Wiley Online Library, 2021. 2

[23] Kai-En Lin, Lin Yen-Chen, Wei-Sheng Lai, Tsung-Yi Lin, Yi-Chang Shih, and Ravi Ramamoorthi. Vision transformer for nerf-based view synthesis from a single input image. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2022. 2

[24] Andrew Liu, Richard Tucker, Varun Jampani, Ameesh Makadia, Noah Snavely, and Angjoo Kanazawa. Infinite nature: Perpetual view generation of natural scenes from a single image. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 6

[25] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[26] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang.

Neural rays for occlusion-aware image-based rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7824–7833, 2022. 2

[27] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 2019. 2

[28] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7210–7219, 2021. 2

[29] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–421, 2020. 1, 2

[30] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 2

[31] Thu H Nguyen-Phuoc, Chuan Li, Stephen Balaban, and Yongliang Yang. Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. *Advances in Neural Information Processing Systems (NeurIPS)*, 31, 2018. 2

[32] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3504–3515, 2020. 2

[33] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2, 3

[34] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021. 2

[35] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 10901–10911, 2021. 2

[36] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2

[37] Mehdi SM Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. *arXiv preprint arXiv:2111.13152*, 2021. 1, 2

[38] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, et al. Zeronvs: Zero-shot 360-degree view synthesis from a single real image. *arXiv preprint arXiv:2310.17994*, 2023. 2

[39] Prafull Sharma, Ayush Tewari, Yilun Du, Sergey Zakharov, Rares Andrei Ambrus, Adrien Gaidon, William T Freeman, Fredo Durand, Joshua B Tenenbaum, and Vincent Sitzmann. Neural groundplans: Persistent neural scene representations from a single image. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. 2

[40] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[41] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems (NeurIPS)*, 32, 2019. 1, 2

[42] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[43] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1, 2

[44] Cameron Smith, Yilun Du, Ayush Tewari, and Vincent Sitzmann. Flowcam: Training generalizable 3d radiance fields without camera poses via pixel-aligned scene flow. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 8

[45] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 175–184, 2019. 2

[46] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Generalizable patch-based neural rendering. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2022. 2, 6, 7

[47] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8269–8279, 2022. 1, 2, 5

[48] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Viewset diffusion:(0-) image-conditioned 3d generative models from 2d data. *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 8

[49] Andrea Tagliasacchi and Ben Mildenhall. Volume rendering digest (for nerf). *arXiv preprint arXiv:2209.02417*, 2022. 2

[50] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, W Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. In *Computer Graphics Forum*, pages 703–735. Wiley Online Library, 2022. 1, 2

[51] Ayush Tewari, Tianwei Yin, George Cazenavette, Semon Rezchikov, Joshua B Tenenbaum, Frédo Durand, William T Freeman, and Vincent Sitzmann. Diffusion with forward models: Solving stochastic inverse problems without direct supervision. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2, 8

[52] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 15182–15192, 2021. 2

[53] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 551–560, 2020. 2

[54] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[55] 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*, 2004. 6

[56] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7467–7477, 2020. 2

[57] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 2022. 2

[58] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2, 6, 7

[59] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep networks as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[60] Yunzhi Zhang and Jiajun Wu. Video extrapolation in space and time. *arXiv e-prints*, pages arXiv–2205, 2022. 2

[61] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37, 2018. 2, 6