latentSplat: Autoencoding Variational Gaussians for Fast Generalizable 3D Reconstruction

Christopher Wewer[1,2], Kevin Raj[1,2],
Eddy Ilg[2], Bernt Schiele[1,2], and Jan Eric Lenssen[1]

[1]Max Planck Institute for Informatics, Saarland Informatics Campus, Germany
[2]Saarland University, Saarland Informatics Campus, Germany
{cwewer, jlenssen}@mpi-inf.mpg.de

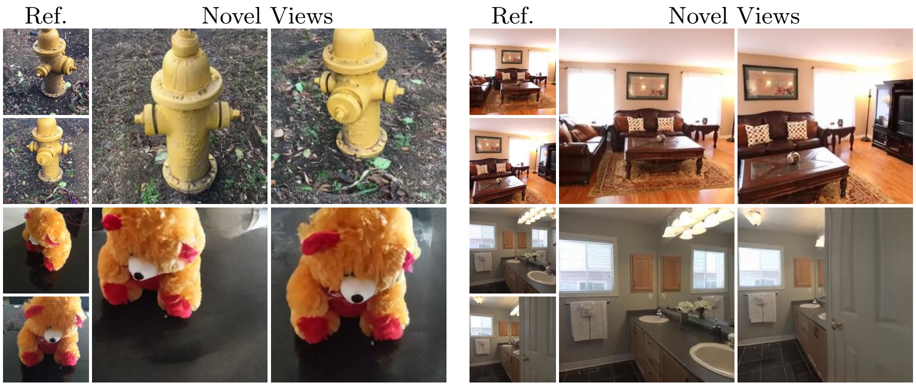

Fig. 1: We present **latentSplat**, a method for scalable generalizable 3D reconstruction from two reference views (left). We autoencode the views into a 3D latent representation consisting of variational feature Gaussians. From this representation, we can perform fast novel view synthesis (right), generalizing to interpolated and extrapolated views.

Abstract. We present latentSplat, a method to predict semantic Gaussians in a 3D latent space that can be splatted and decoded by a lightweight generative 2D architecture. Existing methods for generalizable 3D reconstruction either do not enable fast inference of high resolution novel views due to slow volume rendering, or are limited to interpolation of close input views, even in simpler settings with a single central object, where 360-degree generalization is possible. In this work, we combine a regression-based approach with a generative model, moving towards both of these capabilities within the same method, trained purely on readily available real video data. The core of our method are variational 3D Gaussians, a representation that efficiently encodes varying uncertainty within a latent space consisting of 3D feature Gaussians. From these Gaussians, specific instances can be sampled and rendered via efficient Gaussian splatting and a fast, generative decoder network. We show that latentSplat outperforms previous works in reconstruction quality and generalization, while being fast and scalable to high-resolution data. Our project website[1] provides additional results including videos.

Keywords: 3D Reconstruction · Novel View Synthesis · Feature Gaussian Splatting · Efficient 3D Representation Learning

[1] geometric-rl.mpi-inf.mpg.de/latentsplat/

1 Introduction

Performing 3D reconstruction from a single or a few images is a longstanding goal in computer vision, which went through many iterations of advancements, most recently driven by new techniques in the areas of generalizable radiance fields [10, 53] and foundational diffusion models [35, 40]. At the core, the task is to find an optimal 3D representation that fits a set of observations, a task that is highly underconstrained as there usually are an infinite amount of valid reconstructions that satisfy the given observations. Thus, a strong prior is needed to find a fitting solution - usually modeled by deep neural networks trained on a large amount of data. When designing methods to learn data priors for 3D reconstruction, efficiency is a crucial aspect to allow training on large datasets required for generalization. In this work, we present a method that is highly efficient, scales to a large amount of data and can be trained on real video data without 3D supervision. Real video data is readily available in vast quantities and a promising data type for large 3D models.

Recently, there have been two lines of solutions for the given task of generalizable reconstruction: *regression-based* approaches and *generative* approaches. Methods based on regression, such as pixelNeRF [53] or pixelSplat [9], are usually efficient but are only trained to predict the mean of all possible solutions. While they often succeed in predicting high quality reconstructions of regions that strongly correlate with the input observations, they struggle in regions of high uncertainty, collapsing to blurry reconstructions that lack high frequency details or fail entirely in unseen areas of larger scenes.

The high ambiguity of solutions in reconstruction from incomplete observations suggests to model it in a probabilistic fashion, that is, obtaining a distribution of possible reconstructions that allows to sample individual solutions. Generative approaches, such as Zero-1-to-3 [26] or GeNVS [8], follow this principle, allowing to obtain one possible, realistic reconstruction that might contain hallucinated details. Here, it is important to note that uncertainty in the case of 3D reconstruction varies heavily depending on 3D location. Some areas of the reconstructed scene are observed directly, maybe even by a lot of views, while others are fully occluded or subject to high ambiguity due to very sparse observation. Thus, a sophisticated reconstruction method that models uncertainty should account for varying amounts of uncertainty in 3D space, and needs a generative model to obtain high-quality reconstructions in uncertain areas.

Regression-based models have often been used as conditioning for generative models. A well-known, traditional example are Variational Autoencoders (VAEs) [23]. An encoder is used to parameterize a *variational distribution* in latent space, from which we can sample a specific latent vector that can be decoded into an element following the data distribution via a decoder. It is promising to bring this concept efficiently to 3 dimensions, where a regression model is used to estimate the uncertainty for different locations in 3D space individually, providing the desired locality in uncertainty modeling.

In this work, we approach the desired goals by introducing *latentSplat*, a fast method for generalizable 3D reconstruction that combines the strengths of regression-based and generative approaches. As core of latentSplat we introduce *variational 3D Gaussians*, a representation that models uncertainty explicitly by holding distributions of semantic features on predicted locations in 3D space. Variational Gaussians are obtained via an encoder from two images and model varying amounts of uncertainty depending on the location in 3D space. In observed locations, they can provide a regressed solution with low variance, while acknowledging uncertainty in unobserved areas. From a set of variational Gaussians in 3D space, we can sample a specific instance via the reparameterization trick, render it via efficient splatting to arbitrary views, and decode it with an efficient, generative decoder network in pixel space. We show that *latentSplat*:

– outperforms recent methods in two view reconstruction, achieving state-of-the-art quality both quantitatively and qualitatively, especially in challenging cases of wide-spread input views and view extrapolation,
– is fast and efficient in training and rendering, providing a more scalable solution than previous generative methods,
– is applicable to both, object-centric (with 360° NVS) and general scenes,
– is purely trained on real videos, which is a readily available data resource.

2 Related Work

We revisit recent methods in the area of generalizable novel view synthesis (NVS) and 3D reconstruction. Neural fields [5,29,33,43] have been the most dominant representation to store 3D information for single scenes and scene/object data distributions. Lately, more explicit representations live through a renaissance with 3D Gaussian Splatting [22]. Due to the challenging nature of the task, most methods that enable 360 degree generalization only perform reconstruction on the level of individual objects. Even with the recent rise of large-scale generative models, methods that perform extrapolating reconstruction of larger scenes are still a rarity, often due to missing scalability of the methods. In the following, we distinguish between regression-based approaches in Sec. 2.1 and generative approaches in Sec. 2.2.

2.1 Regression-based Generalizable NVS

Several regression-based models that perform reconstruction from a few views have been proposed in the recent years. A large line of work performs generalization over object categories [15,19,45,51] and are not capable of generalization on scenes. While some early methods in that direction are theoretically capable of generalization on a scene level they often fail to provide the necessary capacity or efficiency [12,24,31,53]. For larger scenes, several image-based rendering methods have been proposed [1,36,39,52]. They produce high quality results in view interpolation settings but cannot generalize to unseen areas. A related approach is to predict multi-plane images [46,49,55,56]. Since generalization is limited to multiple planes in 3D these methods do only allow small view-point variations. Multi-view stereo is also a popular way to provide geometry priors for novel-view synthesis with deep learning [10, 39], which has been modeled fully with deep learning architectures as well [20]. Several alternative representations have been introduced to perform novel view synthesis, such as neural rays [27], light fields [44, 48], and patches [47].

In contrast to all of the above, we are able to provide high-quality 360° reconstructions of object-centric scenes as well as view inter- and extrapolation on large scenes, given only 2 input views. Closest to our work is the very recent pixelSplat [9]. In contrast to their purely regression-based approach, we (1) introduce a semantic feature representation instead of purely explicit Gaussians and (2) model uncertainty explicitly in our framework and therefore enable correct generalization to out-of-context views. Thus, we are able to reconstruct full objects in high quality, even if they are only partially observed with two views.

2.2 Generative Models for NVS

Generative approaches succeed in situations with high uncertainty, i.e. when the signal coming from the conditioning is not sufficient to determine the full reconstruction, or if there is no conditioning at all. A large line of work performs 3D generation and reconstruction of objects with 3D-aware GAN architectures by sampling or inversion, respectively [6, 34]. While these methods are able to produce high-quality results, they are not applicable to scenes. Similarly, 2D diffusion models have been recently adapted to perform the task of 3D reconstruction of objects [25,26,50,57]. Another line of research trains diffusion models directly on 3D representations such as voxel grids [21, 30], triplanes [2, 11], or point clouds [28, 42], and can implement 3D reconstruction by guiding the diffusion process with gradients from reconstructing input views. However, these methods are limited to objects and do not scale to large scenes. Autoregressive transformers have been shown to be able to synthesize novel views that are consistent to some extent with the past sequence of views, but they fail to successfully leverage explicit 3D biases [38, 41].

One of the first works that applies 2D diffusion on larger scenes is GeNVS [8], which is also closest to our work. GeNVS [8] conditions an image space diffusion model with a pixelNeRF feature volume. In contrast to their approach, our approach is orders of magnitude faster and can scale easier to larger images, due to the efficient Gaussian representation instead of volume rendering, and a lightweight decoder instead of an expensive diffusion sampling procedure.

3 Autoencoding Variational Gaussians

In this section, we describe our method in detail, beginning with describing our reconstruction task in Sec. 3.1, before introducing the core of our framework, the semantic variational Gaussian representation, in Sec. 3.2. Then, Sec. 3.3 and Sec. 3.4 will introduce the encoder and decoder architectures, respectively. Last, training details and loss functions are given in Sec. 3.5.

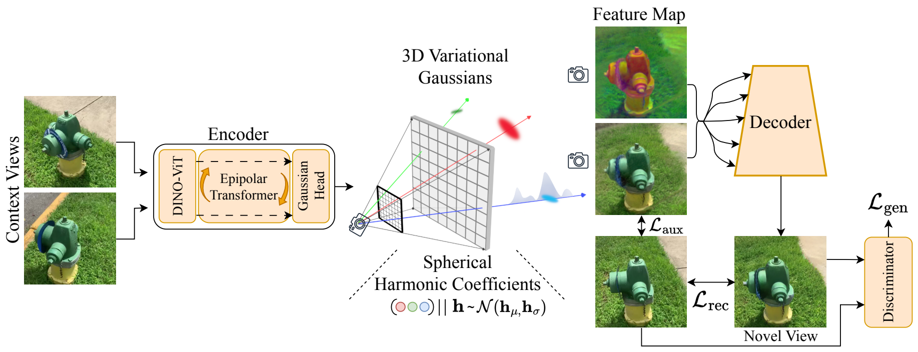
Context Views
Encoder
DINO-ViT
Epipolar Transformer
Gaussian Head
3D Variational Gaussians
Spherical Harmonic Coefficients
$(\circ\circ\circ) \,||\, \mathbf{h} \sim \mathcal{N}(\mathbf{h}_\mu, \mathbf{h}_\sigma)$
Feature Map
Decoder
$\mathcal{L}_{\text{aux}}$
$\mathcal{L}_{\text{rec}}$
$\mathcal{L}_{\text{gen}}$
Novel View
Discriminator

Fig. 2: We present latentSplat, a method for scalable generalizable 3D reconstruction from two views. The architecture follows an autoencoder structure. **(Left)** Two input reference views are encoded into a 3D variational Gaussian representation using an epipolar transformer and a Gaussian sampling head. **(Center)** Variational Gaussians allow sampling of spherical harmonics feature coefficients that determine a specific instance of semantic Gaussians. **(Right)** The sampled instance can be rendered efficiently via Gaussian splatting and a light-weight VAE-GAN decoder.

3.1 Overview and Assumptions

We aim to achieve novel view synthesis from two given video frames (reference views) as input. We assume a dataset of videos with camera poses for each frame such that we can build triplets of two reference views and a target view used for training of our model. As outlined in Fig. 2, our method consists of an encoder, encoding a pair of reference views into a 3D latent representation of Gaussians, the variational Gaussians themselves, and a decoder, rendering the Gaussians from arbitrary views. During training we optimize all parameters to reconstruct the ground-truth target view given the two input images, their camera poses, and the target pose. Once we trained the model, we can obtain variational Gaussians from two views and render them to synthesize novel views.

3.2 Variational 3D Gaussians

At the core of the presented method is a 3D representation that encodes the scene as a set of semantic 3D Gaussians, describing the scene appearance via attached view-dependent feature vectors. In addition, we model uncertainty for each semantic Gaussian individually by storing parameters μ and σ of a normal distribution of spherical harmonic coefficients instead of explicit feature vectors. In total, a scene is represented as a set of N *variational Gaussians* denoted as

$$\mathcal{G} = \{(\mathbf{x}, \mathbf{S}, \mathbf{R}, o, \mathbf{c}, \mathbf{h}_\mu, \mathbf{h}_\sigma)_i\}_{1 \leq i \leq N}, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^3$ is the three-dimensional location, diagonal matrix $\mathbf{S} \in \mathbb{R}^{3\times3}$ Gaussian scale, $\mathbf{R} \in \mathbb{R}^{3\times3}$ Gaussian orientation, $o \in [0,1]$ the opacity of the

Gaussian in 3D space, and $\mathbf{c} \in \mathbb{R}^K$ the spherical harmonics for view-dependent colors. Scale and rotation form the covariance of the 3D Gaussian in space, i.e. $\mathbf{C} = \mathbf{R}\mathbf{S}\mathbf{S}^\top\mathbf{R}^\top$ [22]. The variational distributions of Gaussian features are modeled in the coefficient space of spherical harmonics by parameters $\mathbf{h}_\mu, \mathbf{h}_\sigma$ defining normal distributions $\mathcal{N}(\mathbf{h}_\mu, \mathrm{diag}(\mathbf{h}_\sigma))$. The $\mathbf{h}_\sigma$ hold uncertainty information for individual locations in $\mathbb{R}^3$. We still optimize for explicit RGB coefficients $\mathbf{c}$ in addition to feature parameters $\mathbf{h}_\mu$ and $\mathbf{h}_\sigma \in \mathbb{R}^4$, as Gaussian shape parameters are best optimized on RGB signals (c.f. Sec. 3.4).

Sampling Semantic Gaussians. We distinguish between two states of our Gaussian representation, *variational Gaussians* and *semantic Gaussians*. The latter can be obtained from the former by sampling explicit spherical harmonic coefficients via the reparameterization trick for all Gaussians:

$$\mathbf{h} = \mathbf{h}_\mu + \epsilon \cdot \mathbf{h}_\sigma \qquad \epsilon \sim \mathcal{N}(0, \mathbf{1}), \tag{2}$$

allowing to backpropagate gradients from the sampled coefficients to the reference view encoder. Intuitively, *variational Gaussians* describe the distribution of all possible 3D reconstructions, conditioned on the given reference views. In contrast, *semantic Gaussians* represent a specific sample from this distribution, allowing consistent multi-view renderings of one possible reconstruction.

Rendering Semantic Gaussians. A set of semantic Gaussians can be rendered via the efficient Gaussian splatting renderer provided by Kerbl et al. [22]. We extended it to render feature vectors in addition to RGB colors. The spherical harmonic basis is used to decode our per-Gaussian coefficients $\mathbf{h}$ into view-dependent features before splatting them into pixel space. The encoder architecture presented in Sec. 3.3 predicts Gaussians in the field of view of two reference views. Thus, when rendering novel views, there might be regions in the rendered images that do not have any Gaussians, because they are outside of all reference view frustums. In order to provide a plausible reconstruction of these areas, we obtain our feature image $\mathbf{F}$ by sampling from the normal distribution $\mathcal{N}(\mathbf{F}^{\text{ren}}, 1 - \mathbf{O})$ with rendered features $\mathbf{F}^{\text{ren}}$ and opacity $\mathbf{O}$ via the reparameterization trick:

$$\mathbf{F} = \mathbf{F}^{\text{ren}} + \sqrt{1 - \mathbf{O}} \odot \epsilon\,, \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{1}). \tag{3}$$

The decoder can generate plausible details to fill these empty areas, since it is trained using a GAN formulation (c.f. Sec. 3.5).

3.3 Encoding Reference Views

The variational 3D Gaussian representation described in the previous section is obtained from two given reference views $\mathbf{I}_1$, $\mathbf{I}_2$ sampled from a video sequence. To this end, we adapt the epipolar transformer encoder from pixelSplat [9] to our setting of variational Gaussians by adding the capability of predicting means $\mathbf{h}_\sigma$ and variances $\mathbf{h}_\sigma$ of spherical harmonic coefficients for view-dependent features of each predicted 3D Gaussian (c.f. Fig 2, left). The encoder consists of three parts: (1) a vision transformer [32], (2) an epipolar transformer [16], and (3) a per-pixel sampling of 3D Gaussians from predicted depth distributions [9], which are shortly outlined in the following. For more details about the encoder architecture, we refer to the supplementals, Charatan et al. [9], and He et al. [16].

Vision transformer. The vision transformer is applied to both tokenized images $\mathbf{I}_1$ and $\mathbf{I}_2$ to obtain pixel-aligned feature maps. Compared to pixelSplat [9], we omit the ResNet and ony use a pre-trained DINO ViT-B/8 [4]. Each of the outputs is annotated with depth values from epipolar lines of the other view [16].

Epipolar transformer. Epipolar cross attention [16] is used to allow communication of features across corresponding pixels from both views. To this end, the attached depth values from corresponding epipolar lines are positionally encoded and concatenated to the individual feature maps. Then, keys, queries and values are computed for all locations before performing cross attention between each pixel and samples from its epipolar line in the other view. The communication between two views allows the encoder to resolve the scale ambiguity [9].

Gaussian sampling head. Last, each final feature map is used to predict a distribution over its rays, indicating the probability of a 3D Gaussian lying at the specific depth [9]. To that end, each ray is divided into a set of bins over which a discrete distribution is predicted. Further, for each bin, an offset is predicted to allow for obtaining accurate positions in 3D space. Multiple Gaussians can be sampled per ray. The probability of a sampled Gaussian is used as opacity o. For each sample, we also predict the remaining Gaussian properties of scale $\mathbf{S}$, rotation $\mathbf{R}$ (as quaternion), color $\mathbf{c}$, and variational parameters $(\mathbf{h}_\mu, \mathbf{h}_\sigma)$.

With the described encoder, uncertainty in reconstruction is modeled in two ways: First, the Gaussian locations are sampled from the predicted distributions over the rays, modeling uncertainty in 3D Gaussian location. Second, uncertainty in local appearance is modeled via distribution parameter prediction of the variational Gaussians.

3.4 Decoding

We render both, RGB colors and features into pixel space using the adapted 3D Gaussian rasterizer (c.f. Sec. 3.2). We found that also formulating a loss directly on an RGB output helps with optimizing the structural Gaussian parameters position $\mathbf{x}$, scale $\mathbf{S}$, rotation $\mathbf{R}$, and opacity o. To interpret the features, we use the pre-trained light-weight VAE decoder from LDM [40] (c.f. Fig. 2, right). It is a purely convolutional architecture with four upsample blocks, each consisting of two residual blocks. The decoder receives multi-scale feature images: first, we bilinearly down-sample the rendered feature image three times along the spatial dimensions. Then, we feed the different scales into the U-Net decoder at different stages of the architecture. The decoder is trained together with the remaining architecture using reconstruction and generative losses, as described in Sec. 3.5.

3.5 Training

The presented architecture consisting of encoder, variational Gaussians, and decoder is trained in an end-to-end fashion on video data. In each iteration, we sample a video from our training dataset, select two reference views and four target views. The selection criteria differ for different scene types (large scenes and object-centric scenes) and are detailed in the experimental setup in Sec. 4.1. The reference views are encoded into variational 3D Gaussians, rendered from the target camera perspectives, and decoded using the VAE decoder. We train all networks using the following losses.

Reconstruction Losses Similar to LDM [40], we use a combination of a standard L1 loss and LPIPS as a perceptual loss between the decoded feature image $\hat{\mathbf{T}}$ and the target image $\mathbf{T}$:

$$\mathcal{L}_{\text{rec}} = \lambda_1 \|\hat{\mathbf{T}} - \mathbf{T}\|_1 + \lambda_2 \text{ LPIPS}(\hat{\mathbf{T}}, \mathbf{T}) \tag{4}$$

Auxiliary Losses We further apply an auxiliary loss directly between the color renderings $\hat{\mathbf{T}}_{\text{aux}}$ and the target image $\mathbf{T}$ to provide better gradients to the structural parameters (c.f. Sec 3.2):

$$\mathcal{L}_{\text{aux}} = \lambda_3 \|\hat{\mathbf{T}}_{\text{aux}} - \mathbf{T}\|_2^2 + \lambda_4 \text{ LPIPS}(\hat{\mathbf{T}}_{\text{aux}}, \mathbf{T}) \tag{5}$$

Generative Loss To enable correct sampling and generation in uncertain regions, we further optimize our method using GAN losses, adapted from the LDM VAE-GAN decoder [40]. We add and further train a pre-trained discriminator network [18] D from LDM [40] predicting the likelihood of image patches being real. Therefore, our architecture is trained to directly maximize its output:

$$\mathcal{L}_{\text{gen}} = E_{\hat{\mathbf{T}}}[\log(1 - D(\hat{\mathbf{T}}))], \qquad \mathcal{L}_{\text{disc}} = E_{\mathbf{X}}[\log(D(\mathbf{X}))] + E_{\hat{\mathbf{T}}}[\log(1 - D(\hat{\mathbf{T}}))], \tag{6}$$

where $\mathbf{X}$ are real images from the training dataset. In summary, our autoencoder is optimized with $\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{aux}} + \mathcal{L}_{\text{gen}}$ and the discriminator with $\mathcal{L}_{\text{disc}}$.

4 Experiments

In this section, we detail our experiments made with latentSplat, provide comparisons with state-of-the-art baselines, and verify our architecture design in form of an ablation. Our experiments aim to support the following statements: (1) latentSplat improves on previous methods for two-view interpolation in terms of visual quality, (2) can also generalize better to extrapolation, i.e. producing novel views outside of given reference views, (3) avoids unnecessary hallucination but sticks to the identity of the observed scene, and (4) maintains the real-time rendering capabilities and memory efficiency of 3D Gaussian splatting. We provide additional results in the appendix.

Cat.	Method	Interpolation						Extrapolation					
		FID↓	KID↓	LPIPS↓	DISTS↓	PSNR↑	SSIM↑	FID↓	KID↓	LPIPS↓	DISTS↓	PSNR↑	SSIM↑
Hydrants	pixelNeRF [53]	183.24	0.104	0.566	0.345	18.39	0.411	238.41	0.156	0.658	0.429	**16.24**	0.360
	Du et al. [14]	154.20	0.090	0.471	0.276	**18.78**	**0.476**	275.61	0.208	0.599	0.382	15.79	**0.366**
	pixelSplat [9]	58.13	0.011	0.401	0.203	18.05	0.429	92.61	0.030	0.485	0.263	15.75	0.332
	Ours	**40.93**	**0.005**	**0.356**	**0.166**	18.01	0.413	**48.03**	**0.008**	**0.426**	**0.202**	15.78	0.306
Teddybears	pixelNeRF [53]	179.85	0.082	0.580	0.386	18.97	0.580	236.82	0.132	0.649	0.450	17.05	0.531
	Du et al. [14]	141.16	0.065	0.436	0.245	20.69	0.666	229.78	0.142	0.564	0.347	16.65	0.553
	pixelSplat [9]	74.82	0.014	0.369	0.200	20.73	**0.687**	123.33	0.047	0.473	0.260	17.51	**0.564**
	Ours	**53.47**	**0.004**	**0.338**	**0.173**	**20.83**	0.663	**71.12**	**0.010**	**0.434**	**0.219**	**17.71**	0.533

Table 1: 360° Novel view synthesis on CO3D. We outperform previous methods in terms of generative metrics and perceptual metrics. Due to the generative nature of our method, we are only on-par in traditional reconstruction metrics of PSNR and SSIM, as they strongly punish generation of details if they do not match the target.

4.1 Experimental Setup

Datasets We conduct two-view reconstruction experiments for an object-centric setting as well as for general video datasets capturing diverse scenes. For the former, we make use of the Common Objects in 3D (CO3Dv2) [37] dataset, which consists of video captures of real-world objects grouped into categories. Following related work [8], we choose cleaned subsets of hydrants and teddybears, and randomly split the scenes into 95% training and 5% test data. In all experiments, including baselines, we gradually increase the gap between the two reference views from initially 8-18 frames up to 25 frames roughly corresponding to 90° in the usually 102 frames long sequences. At the same time, we increasingly randomize the target view selection from pure interpolation until a uniform distribution over all views. For an evaluation on general scenes, we test latentSplat on RealEstate10k [56], a dataset of home walkthrough clips gathered from about 10000 YouTube videos. We use the provided splits and employ the same training curriculum as pixelSplat [9] except of increasing the sampling interval of target views up to 45 frames before and after both reference views to enable learning extrapolation.

Baselines We compare our approach against four baselines. pixelNeRF [53] conditions a single NeRF MLP by interpolating pixel-aligned features of reference views. Du et al. [14] is a light field rendering approach that uses multi-view self-attention and cross-attention of target rays to samples along its epipolar lines in the input images. pixelSplat [9] defines an encoder including an epipolar transformer to predict 3D Gaussians along the rays of two reference views that can be efficiently rendered via rasterization. Unlike the previous regression-based approaches, GeNVS [7] proposes a generative diffusion model with view-conditioning via pixelNeRF in a feature space.

Metrics We employ three groups of each two metrics: (1) FID [17] and KID [3] measure the similarity between the distributions of predicted novel views and the corresponding ground truth. They are the established metrics for image syn-

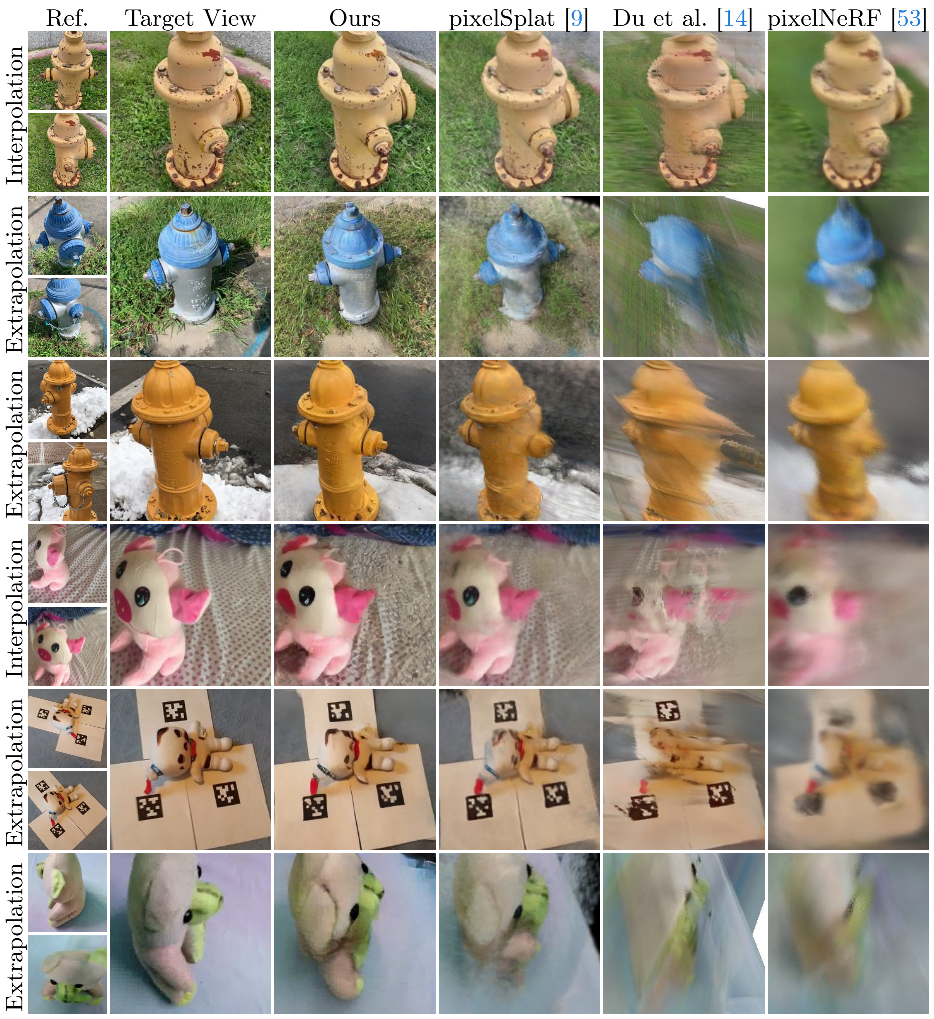

Fig. 3: Qualitative results on the CO3D dataset [37]. We evaluate on two categories. The top three rows depict the rendering of the target view on Hydrants, while the bottom three rows showcase the corresponding rendering of the target view on Teddybear, in comparison with the baselines.

thesis of generative models and reflect visual quality. (2) Perceptual metrics like LPIPS [54] and DISTS [13] leverage features of deep networks for the comparison of images w.r.t. structure and texture. (3) Classical reconstruction metrics such as PSNR and SSIM are still well-established for evaluation of dense-view 3D reconstruction. However, because incorporating plain pixel-wise similarities, these metrics prefer blur over realistic details and are therefore not well-suited for the evaluation of generative methods.
Further implementation details can be found in the supplementary materials and in our code, which we will make publicly available.

	Interpolation						Extrapolation					
Method	FID↓	KID↓	LPIPS↓	DISTS↓	PSNR↑	SSIM↑	FID↓	KID↓	LPIPS↓	DISTS↓	PSNR↑	SSIM↑
pixelNeRF [53]	152.16	0.132	0.550	0.359	20.51	0.592	160.77	0.141	0.567	0.371	20.05	0.575
Du et al. [14]	9.73	0.005	0.219	0.133	**24.55**	<u>0.812</u>	11.34	0.006	0.242	0.144	21.83	**0.790**
pixelSplat [9]	<u>4.41</u>	<u>0.002</u>	<u>0.174</u>	<u>0.107</u>	<u>24.32</u>	**0.822**	<u>5.78</u>	<u>0.003</u>	<u>0.216</u>	<u>0.130</u>	<u>21.84</u>	<u>0.777</u>
Ours	**2.22**	**0.001**	**0.164**	**0.094**	23.93	<u>0.812</u>	**2.79**	**0.001**	**0.196**	**0.109**	**22.62**	<u>0.777</u>

Table 2: Novel view synthesis on RE10k. We compare against recent methods on view interpolation and extrapolation on the large scale RealEstate10k dataset [56], showing that our method can handle large scale scenes and high resolution images. Same as in the CO3D setting, we outperform previous works in generative metrics and perceptual metrics, while being on par in traditional metrics. Qualitative results in Fig. 4 show that we produce higher quality reconstructions, especially for extrapolation.

4.2 Object-Centric 3D Reconstruction

Table 1 summarizes our quantitative results for object-centric 3D reconstruction on CO3D [37]. We outperform all baselines by a large margin in FID and KID while also significantly improving upon pixelSplat [9] in perceptual metrics. This indicates that we achieve a better visual quality and at the same time remain faithful w.r.t. to the observed scene. Despite being a generative approach, we also outperform all baselines in PSNR on teddybears, but fall short in SSIM. However, our deterministic ablation (c.f. Sec. 4.6) validates that these classical metrics prefer blurry reconstructions over perceptually good ones with generated details. The differentiation of interpolation and extrapolation reveals that our approach generalizes better to unseen areas of the scene, as shown by a smaller performance difference compared to the baselines. We show qualitative results in Fig. 3. latentSplat predicts sharp and detailed reconstructions fitting to the observations. Moreover, it succeeds in synthesizing completely unobserved areas allowing full 360° generalization. For the challenging case of modeling the unobserved backside of a hydrant (c.f. third row in Fig. 3), all regression-based approaches predict a blurry reconstruction indicating high uncertainty. latentSplat's ability of generating a realistic novel view in this case highlights the advantage of uncertainty and generation in the rendering process.

4.3 Scene-Level 3D Reconstruction

We report quantitative results for scene-level 3D reconstruction on RealEstate10k in Table 2. Again, latentSplat outperforms all baselines in FID and KID as well as LPIPS and DISTS. Although extrapolation on scene-level is a quite different task compared to learning a category-level prior for object-centric videos, we observe the same result that latentSplat generalizes better to extrapolation compared to all baselines, even achieving state-of-the-art in PSNR as well. This highlights the applicability of our method for various kinds of real-world videos. Looking at qualitative examples in Fig. 4, we can see that our approach produces clean and visually pleasing novel views with significantly less artifacts compared to the baselines.

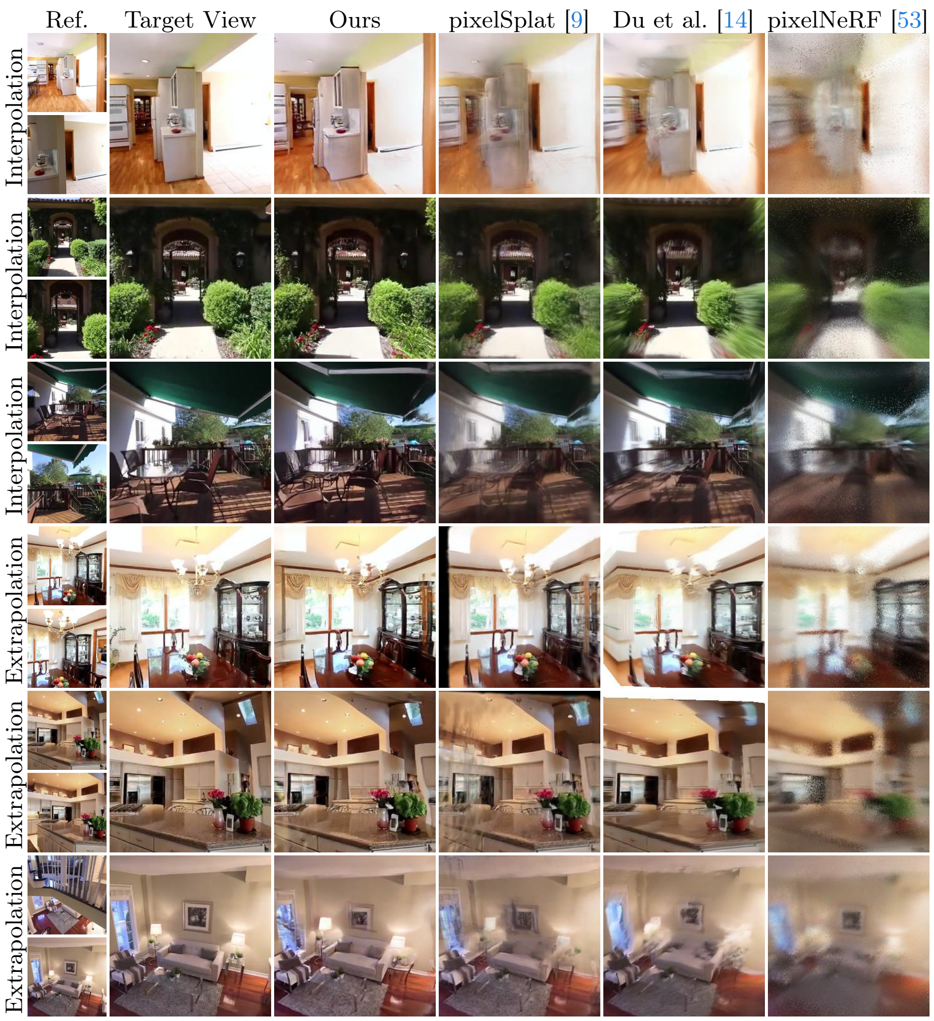

Fig. 4: Qualitative results on RealEstate10k [56]. We show that in many challenging cases latentSplat provides better reconstruction than previous approaches. This specifically holds when (1) reference views are far apart from each other or when (2) extrapolating outside of reference views.

4.4 Comparison with GeNVS

We provide an additional qualitative comparison with GeNVS [8] in Fig. 5. Their code is not available and we are not able to reproduce their results for a fair comparison. We still aim to do a comparison using figures from their paper. Note that there are major differences between the setups of GeNVS and latentSplat such as the number of input views and the version of the dataset (CO3Dv1 vs CO3Dv2). We match their setting as close as possible by creating a test split of all their evaluation examples, finding closest poses, and using the

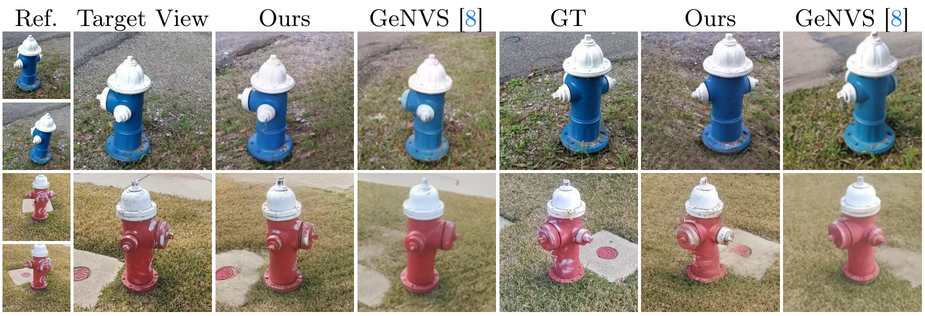

Fig. 5: Comparison with GeNVS [8] We qualitatively compare against GeNVS. Note that the setups of both methods differ strongly and that GeNVS code is not available for reproducing results. Thus, this comparison is **not fair**. We selected the same CO3D test examples and compared against renderings shown in their paper. Overall, both methods generate similar quality while ours is much faster (c.f. Sec. 4.5).

same input view together with a close second view. Although the comparison of single- and two-view 3D reconstruction is not fair, note that GeNVS uses scale normalization and depth cues (c.f. supplementary material C.3) to resolve the scale ambiguity from a single input view. The qualitative comparison shows similar image quality. However, our approach is much faster and efficient during both inference and training, as detailed in Sec. 4.5.

4.5 Efficiency

Table 4 shows our time and memory requirements for training and inference compared to the baselines. Compared to pixelSplat, we have a slightly faster encoder and only a 1ms slower rendering stemming to 68% from the convolutional decoder and the remaining 32% from splatting additional feature channels. Therefore, we maintain the real-time rendering capabilities of 3D Gaussian splatting despite introducing a generative model. Furthermore, we are memory efficient during both training and inference. Compared to GeNVS, we are orders of magnitude faster in inference and training.

4.6 Ablations

We conduct an ablation study for the extrapolation setting on CO3D hydrants with the results given in Table 3. Interestingly, while performing much worse in FID and KID with a deterministic version omitting the variational formulation as well as the GAN loss, we obtain state-of-the-art results for PSNR and SSIM. This shows that there is a trade-off between conservative and therefore blurry reconstructions favored by the classical metrics and realistic and detailed novel views rewarded by FID and KID. Without latent and/or RGB skip connections, we perform slightly worse in most metrics validating our design choice of leveraging the rendered features and colors at multiple resolutions. Overall, we find the best balance by using our generative approach together with latent

Method	FID↓	KID↓	LPIPS↓	DISTS↓	PSNR↑	SSIM↑
Deterministic	69.33	0.022	**0.406**	0.240	**16.50**	**0.340**
No skip con.	50.31	0.007	0.440	**0.204**	15.55	0.293
No RGB skip	**48.48**	**0.006**	0.438	0.206	15.48	0.288
Ours	48.88	**0.006**	0.436	0.205	15.61	0.299

Table 3: Ablation study. We ablate our architecture design for extrapolation on CO3D hydrants. *Deterministic* shows a version of our architecture without the variational formulation and without the GAN loss. *No skip con.* omits the multi-scale decoder input and *No RGB skip* omits the rendered RGB input to the decoder.

Method	Inference Time (s) Encode↓	Inference Time (s) Render↓	Training Time (GPU-h)↓	Inference Memory (GB)↓
pixelNeRF [53]	**0.003**	5.464	**96**	3.961
Du et al. [14]	0.011	1.337	288	19.604
pixelSplat [9]	0.113	**0.002**	192	**2.755**
GeNVS [8]	-	6.423[1]	2112	-
Ours	0.080	0.003	192	3.161

Table 4: Efficiency comparison. Our method renders roughly 2000 times faster and is much less expensive to train than the state-of-the-art generative model for novel view synthesis GeNVS [8]. Compared to the state-of-the-art regression-based approach, latentSplat entails a negligible increase in resource requirements in turn for significantly better image quality.

and RGB skip connections. We attribute that to the fact that the decoder has global and local context when interpreting features for image synthesis, which helps generating consistent images.

5 Conclusion

We presented latentSplat, a method that successfully combines the strengths of regression-based approaches with the power of a lightweight generative model to handle uncertainty. Our approach achieves state-of-the-art image quality in novel view synthesis while providing the highest perceptual similarity to the ground truth. Compared to previous generative approaches, latentSplat is much faster and more scalable, enabling real-time rendering in large resolutions.

Acknowledgements. This project was partially funded by the *Saarland/Intel Joint Program on the Future of Graphics and Media.* We also thank David Charatan and co-authors for the great pixelSplat codebase.

[1] GeNVS [8] does not provide code. We estimated the rendering time based on pixelNeRF [53] and StableDiffusion [40] denoising for 25 steps and resolution 256.

References

1. Aliev, K.A., Sevastopolsky, A., Kolos, M., Ulyanov, D., Lempitsky, V.: Neural point-based graphics. In: ECCV (2020) 3
2. Anciukevičius, T., Xu, Z., Fisher, M., Henderson, P., Bilen, H., Mitra, N.J., Guerrero, P.: Renderdiffusion: Image diffusion for 3d reconstruction, inpainting and generation. In: CVPR (2023) 4
3. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying MMD GANs. In: ICLR (2018) 9
4. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: CVPR (2021) 7, 18
5. Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local SDF priors for detailed 3D reconstruction. In: ECCV (2020) 3
6. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., Mello, S.D., Gallo, O., Guibas, L., Tremblay, J., Khamis, S., Karras, T., Wetzstein, G.: Efficient geometry-aware 3D generative adversarial networks. In: CVPR (2022) 4
7. Chan, E.R., Nagano, K., Chan, M.A., Bergman, A.W., Park, J.J., Levy, A., Aittala, M., De Mello, S., Karras, T., Wetzstein, G.: Genvs: Generative novel view synthesis with 3d-aware diffusion models (2023) 9
8. Chan, E.R., Nagano, K., Chan, M.A., Bergman, A.W., Park, J.J., Levy, A., Aittala, M., Mello, S.D., Karras, T., Wetzstein, G.: GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In: ICCV (2023) 2, 4, 9, 12, 13, 14
9. Charatan, D., Li, S., Tagliasacchi, A., Sitzmann, V.: pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In: arXiv (2023) 2, 4, 6, 7, 9, 10, 11, 12, 14, 18, 19, 23, 24, 25, 26, 27, 28
10. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: ICCV (2021) 2, 4
11. Chen, H., Gu, J., Chen, A., Tian, W., Tu, Z., Liu, L., Su, H.: Single-stage diffusion nerf: A unified approach to 3d generation and reconstruction. In: ICCV (2023) 4
12. Chibane, J., Bansal, A., Lazova, V., Pons-Moll, G.: Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In: CVPR (2021) 3
13. Ding, K., Ma, K., Wang, S., Simoncelli, E.P.: Image quality assessment: Unifying structure and texture similarity. IEEE Transactions on Pattern Analysis and Machine Intelligence (2022) 10
14. Du, Y., Smith, C., Tewari, A., Sitzmann, V.: Learning to render novel views from wide-baseline stereo pairs. In: CVPR (2023) 9, 10, 11, 12, 14, 19, 23, 24, 25, 26, 27, 28
15. Guo, P., Bautista, M.A., Colburn, A., Yang, L., Ulbricht, D., Susskind, J.M., Shan, Q.: Fast and explicit neural view synthesis. In: WACV (2022) 3
16. He, Y., Yan, R., Fragkiadaki, K., Yu, S.I.: Epipolar transformers. In: CVPR (2020) 7, 18
17. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) NeurIPS. Curran Associates, Inc. (2017) 9
18. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (July 2017) 8, 18

19. Jang, W., Agapito, L.: Codenerf: Disentangled neural radiance fields for object categories. In: ICCV (2021) 3
20. Johari, M.M., Lepoittevin, Y., Fleuret, F.: Geonerf: Generalizing nerf with geometry priors. CVPR (2022) 4
21. Karnewar, A., Vedaldi, A., Novotny, D., Mitra, N.: Holodiffusion: Training a 3D diffusion model using 2D images. In: CVPR (2023) 4
22. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (2023) 3, 6
23. Kingma, D.P., Welling, M.: Auto-Encoding Variational Bayes. In: ICLR (2014) 2
24. Lin, K.E., Yen-Chen, L., Lai, W.S., Lin, T.Y., Shih, Y.C., Ramamoorthi, R.: Vision transformer for nerf-based view synthesis from a single input image. In: WACV (2023) 3
25. Liu, M., Xu, C., Jin, H., Chen, L., Xu, Z., Su, H., et al.: One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. arXiv (2023) 4
26. Liu, R., Wu, R., Hoorick, B.V., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: ICCV (2023) 2, 4
27. Liu, Y., Peng, S., Liu, L., Wang, Q., Wang, P., Theobalt, C., Zhou, X., Wang, W.: Neural rays for occlusion-aware image-based rendering. In: CVPR (2022) 4
28. Melas-Kyriazi, L., Rupprecht, C., Vedaldi, A.: Pc2: Projection-conditioned point cloud diffusion for single-image 3d reconstruction. In: CVPR (2023) 4
29. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020) 3
30. Müller, N., Siddiqui, Y., Porzi, L., Bulò, S.R., Kontschieder, P., Nießner, M.: Diffrf: Rendering-guided 3d radiance field diffusion. In: CVPR (2023) 4
31. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: CVPR (2020) 3
32. Oquab, M., Darcet, et al.: Dinov2: Learning robust visual features without supervision (2023) 7
33. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: DeepSDF: Learning continuous signed distance functions for shape representation. In: CVPR (2019) 3
34. Pavllo, D., Tan, D.J., Rakotosaona, M.J., Tombari, F.: Shape, pose, and appearance from a single image via bootstrapped radiance field inversion. In: CVPR (2023) 4
35. Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv (2023) 2
36. Rakhimov, R., Ardelean, A.T., Lempitsky, V., Burnaev, E.: NPBG++: Accelerating neural point-based graphics. In: CVPR (2022) 3
37. Reizenstein, J., Shapovalov, R., Henzler, P., Sbordone, L., Labatut, P., Novotny, D.: Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In: ICCV (2021) 9, 10, 11, 19, 20, 21, 22, 23, 24, 25, 26
38. Ren, X., Wang, X.: Look outside the room: Synthesizing a consistent long-term 3d scene video from a single image. In: CVPR (2022) 4
39. Riegler, G., Koltun, V.: Free view synthesis. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) ECCV. Springer International Publishing, Cham (2020) 3, 4
40. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022) 2, 7, 8, 14, 18, 19

41. Rombach, R., Esser, P., Ommer, B.: Geometry-free view synthesis: Transformers and no 3d priors. In: ICCV (2021) 4
42. Schröppel, P., Wewer, C., Lenssen, J.E., Ilg, E., Brox, T.: Neural point cloud diffusion for disentangled 3d shape and appearance generation. In: CVPR (2024) 4
43. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. In: NeurIPS (2020) 3
44. Sitzmann, V., Rezchikov, S., Freeman, W.T., Tenenbaum, J.B., Durand, F.: Light field networks: Neural scene representations with single-evaluation rendering. In: NeurIPS (2021) 4
45. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3D-structure-aware neural scene representations. In: NeurIPS (2019) 3
46. Srinivasan, P.P., Tucker, R., Barron, J.T., Ramamoorthi, R., Ng, R., Snavely, N.: Pushing the boundaries of view extrapolation with multiplane images. In: CVPR (2019) 3
47. Suhail, M., Esteves, C., Sigal, L., Makadia, A.: Generalizable patch-based neural rendering. In: ECCV. Springer (2022) 4
48. Suhail, M., Esteves, C., Sigal, L., Makadia, A.: Light field neural rendering. In: CVPR (2022) 4
49. Tucker, R., Snavely, N.: Single-view view synthesis with multiplane images. In: CVPR (2020) 3
50. Watson, D., Chan, W., Brualla, R.M., Ho, J., Tagliasacchi, A., Norouzi, M.: Novel view synthesis with diffusion models. In: ICLR (2023) 4
51. Wewer, C., Ilg, E., Schiele, B., Lenssen, J.E.: SimNP: Learning self-similarity priors between neural points. In: ICCV (2023) 3
52. Wiles, O., Gkioxari, G., Szeliski, R., Johnson, J.: Synsin: End-to-end view synthesis from a single image. In: CVPR (2020) 3
53. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: CVPR (2021) 2, 3, 9, 10, 11, 12, 14, 19, 23, 24, 25, 26, 27, 28
54. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) 10
55. Zhang, Y., Wu, J.: Video extrapolation in space and time. In: ECCV (2022) 3
56. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: learning view synthesis using multiplane images. SIGGRAPH (2018) 3, 9, 11, 12, 19, 20, 27, 28
57. Zhou, Z., Tulsiani, S.: Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. In: CVPR (2023) 4

Appendix

In this supplementary material, we give details regarding architecture and evaluation in Sections A and B. Section C deals with the evaluation of 3D consistency using videos. We present visualizations of intermediate results as well as model uncertainty underlining the advantages of latentSplat as a fast generative approach in Section D. Finally, we provide additional qualitative results in Section E. Please also refer to the project website[2] for results in motion.

A Architecture Details

Our architecture is composed of an encoder, a decoder, and a discriminator.

Encoder We adapt the encoder from pixelSplat [9] to our setting. It consists of a pre-trained DINOv1 ViT-B/8 [4] backbone, an epipolar transformer [16], and a Gaussian sampling head. We upscale the local patch tokens from the backbone by repeating them for the original patch size and add the linearly projected global class token to obtain a feature map in the orginal image resolution. The epipolar transformer has two blocks with four-head attention [16] to 32 samples of each pixel's epipolar line. The token dimensionality is 128 and for computational feasibility we downscale the input feature maps by a factor of 4 in each dimension. To prevent a bottleneck with low spatial dimensions in form of the epipolar transformer, we add a skip connection in the original resolution. Finally, the Gaussian sampling head employs simple linear projections with activation functions specific to the individual parameters of the Gaussians like normalized quaternions for rotation and scaled and shifted sigmoid for scale. The resulting variational Gaussians exhibit spherical harmonic coefficients up to a degree of 4 for RGB and a Gaussian distribution of spherical harmonic coefficients up to a degree of 2 for features of dimensionality 4 (LDM VAE [40] feature size).

Decoder We adapt the pre-trained VAE decoder from LDM [40]. It is a fully convolutional architecture consisting of four upsample blocks, each consisting of two residual blocks. We add 1x1 convolutions as skip connections to feed in downscaled versions of the intermediate auxiliary RGB renderings and the feature maps to each upsample block. In order to avoid noisy gradients, we initialize the weights of these additional layers to zero.

Discriminator We finetune the pre-trained discriminator from LDM [40]. It is a very small CNN with three strided convolutions as hidden layers such that it does not have a global receptive field and only predicts the likelihood of patches [18].

Loss Our loss can be split into auxiliary losses, reconstruction losses, and a generative loss. We start the training with auxiliary losses only for the first 100k iterations to avoid fitting features to Gaussians in wrong positions. Furthermore, we delay the generative loss until 125k iterations to prevent an overly strong

[2] geometric-rl.mpi-inf.mpg.de/latentsplat/

discriminator. In order to account for scale and relevance, the individual terms have different weights. As reconstruction losses, the weights of L1 and LPIPS are $\lambda_1 = 1$ and $\lambda_2 = 1$, respectively. The mean squared error and LPIPS for the axiliary rendering are weighted by $\lambda_3 = 10$ and $\lambda_4 = 0.5$. Finally, we adopt the adaptive weighting of the generative loss from LDM [40] by leveraging the ratio of gradient norms from reconstruction and generative losses to the last layer of the decoder.

B Evaluation Details

View Selection. For the evaluation on a test split, we create examples out of two reference and three target views. For each scene of CO3D [37], we sample 20 reference view pairs within a frame distance of 16 to 25 while respecting the circular camera motion. Then we sample random target views within or outside of the reference views for interpolation or extrapolation, respectively. For RealEstate10k [56], we stick to the same evaluation setup of pixelSplat [9]. For each scene, we sample one reference view pair within a frame distance of 45 to 135 that fulfills the requirement of mutual overlap (coverage of epipolar lines) of at least 60%. For extrapolation, we sample target views up to 45 frames previous/after the first/second reference view.

Efficiency Comparison. We train all models including baselines until convergence, which translates to 200k iterations for latentSplat, pixelSplat [9], and pixelNeRF [53], and 300k for the method of Du et al. [14]. In wallclock time, this means we used two NVidia A40 GPUs for four days for both latentSplat and pixelSplat, one GPU for four days for pixelNeRF, and four GPUs for three days for the method of Du et al. [14]. For a fair comparison in terms of inference time and memory, we evaluate all methods on the same data (extrapolation on CO3D hydrants) and the same device (NVidia RTX 4090).

C 3D Consistency

We aim to assess the 3D consistency of novel view snythesis with latentSplat via a qualitative evaluation of videos. For this, please refer to the project website. Given just two reference views, we are able to synthesize full 360° novel views for CO3D [37] hydrants and teddybears without obvious geometric inconsistencies. Furthermore, the videos on RealEstate10k [56] appear realistic without flickering from pixel-level differences in generation between nearby frames. Note that we sample Gaussian features only once independent of the target views resulting in consistent renderings even in case of uncertainty. Only low-opacity regions outside of the reference view frustums are filled with independently sampled noisy features in the target image space. Hence, these especially difficult areas (usually background) are more prone to be inconsistent in different views.

D Additional Visualizations

We visualize the process of assembling the final prediction in terms of intermediate results in Fig. 6 for CO3D hydrants and Fig. 7 for CO3D teddybears.

Auxiliary. In order to provide better gradients to the structural parameters of the variational Gaussians, we apply an auxiliary reconstruction loss on directly rasterized RGB images skipping the VAE-GAN decoder. These renderings (4th column) suffer from blur in regions of high uncertainty as well as artifacts like floating Gaussians.

Features. Besides RGB, we render general feature maps that we visualize via employing PCA jointly over all pixel-aligned feature vectors of all frames (5th column). We observe that different parts of the objects are encoded by different latent features, while the background is clearly separated and filled with noise in areas of low density according to Eq. 3 in the main paper.

Uncertainty. Finally, we aim to illustrate the uncertainty of our variational Gaussians directly by rendering the standard deviation $\mathbf{h}_\sigma$ in Eq. 2 of the main paper, averaged over all feature channels. To deal with empty regions outside of the reference camera frustums, we set the background standard deviation to one, which is in line with our feature map sampling in Eq. 3. The resulting images (6th column) show generally higher uncertainty (dark) for the background, which is either completely invisible or only partly visible in the reference views, compared to the main object, for which the model learns a category-level prior. For the main object, the model is less certain about details like edges or the fur of teddybears than about plain uniform surfaces, which explains the advantage of the generative decoder w.r.t. a higher level of detail.

E Additional Results

We provide additional qualitative results for interpolation and extrapolation on CO3D [37] and RealEstate10k [56] in Fig. 8 to 13, and in the supplemental video.

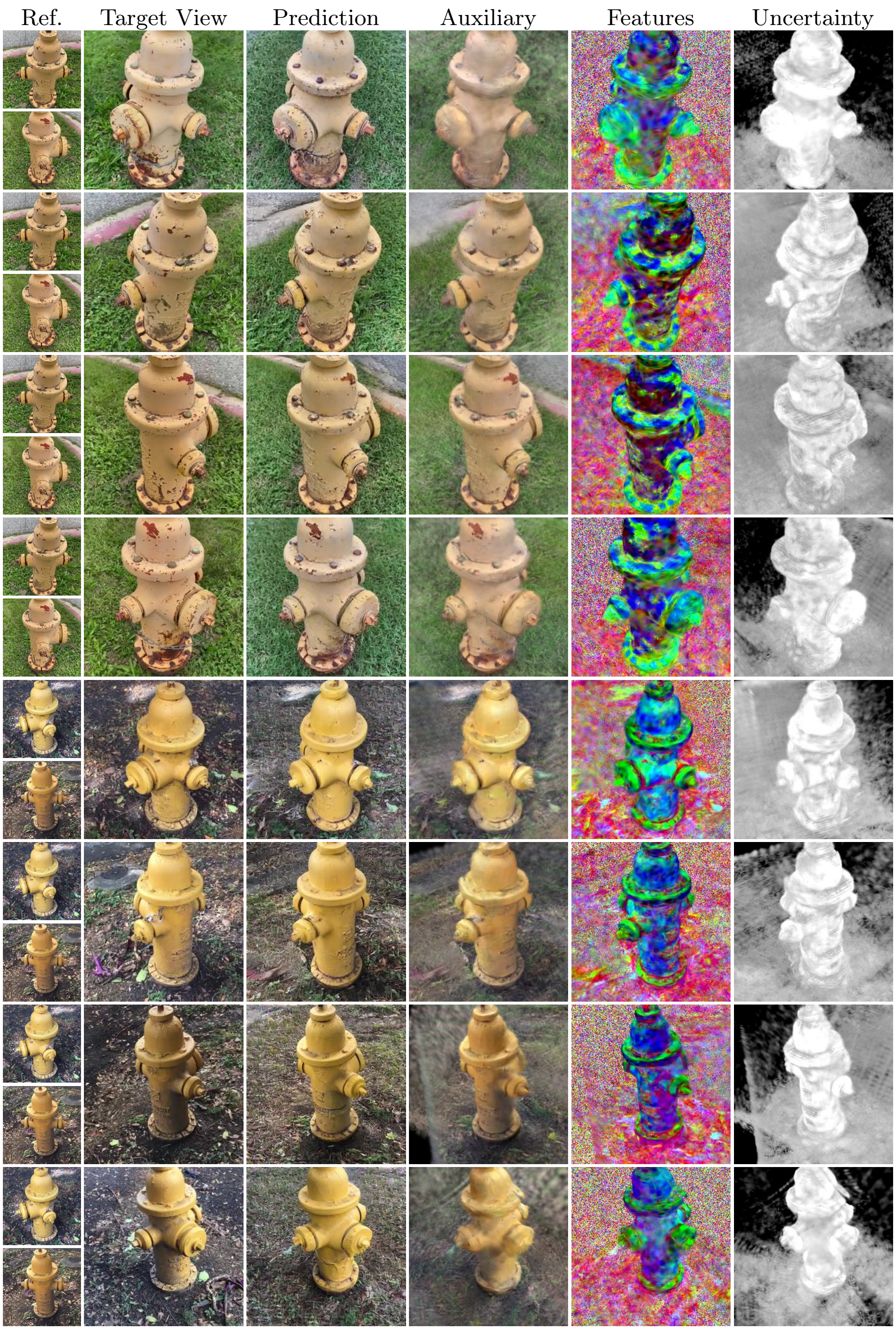

Fig. 6: Intermediate results for 360° novel view synthesis on CO3D hydrants [37]. For uncertainty on the right, darker regions correspond to higher uncertainty. Features are visualized with PCA dimensionality reduction to 3 dimensions.

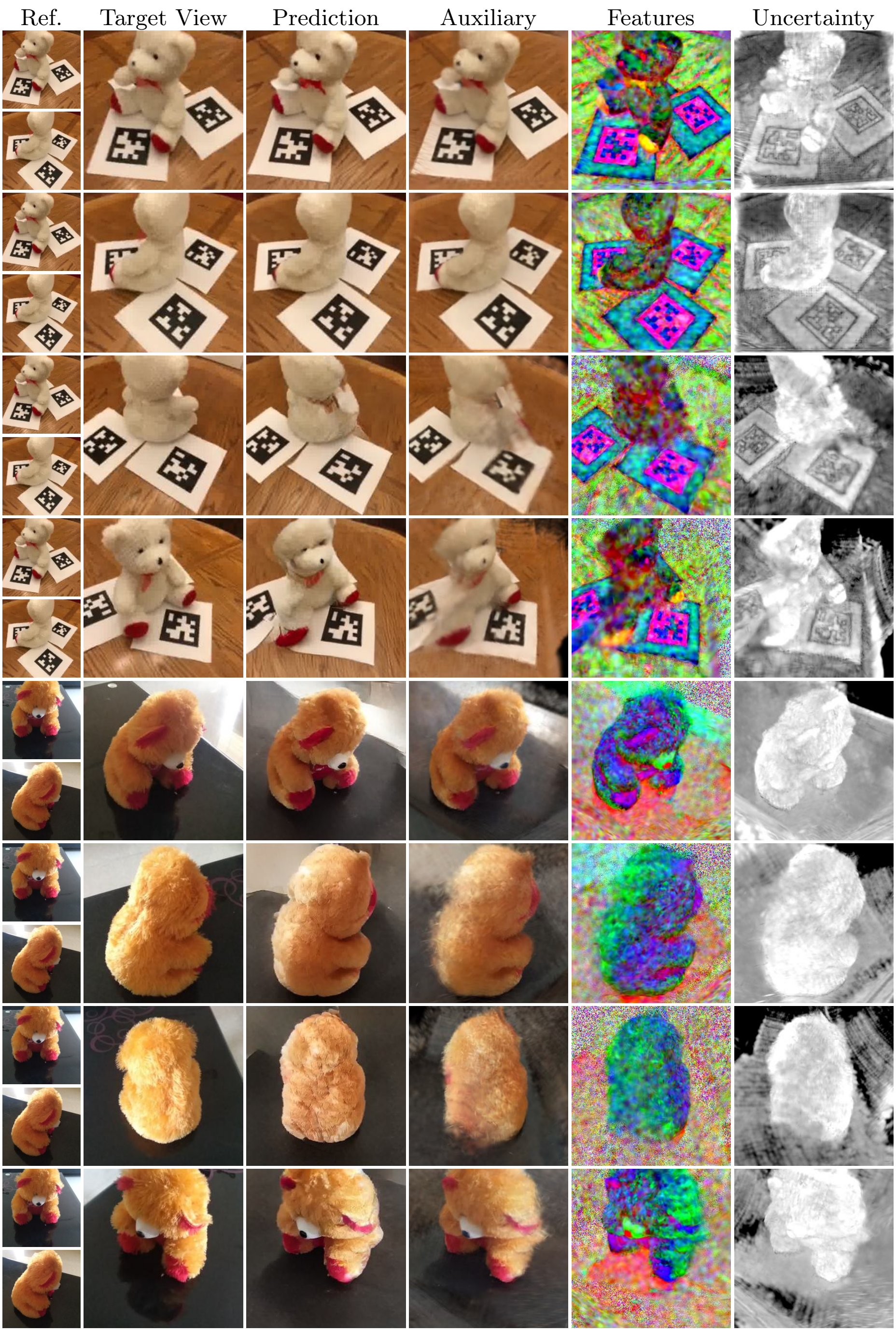

Fig. 7: Intermediate results for 360° novel view synthesis on CO3D teddy-bears [37]. For uncertainty on the right, darker regions correspond to higher uncertainty. Features are visualized with PCA dimensionality reduction to 3 dimensions.

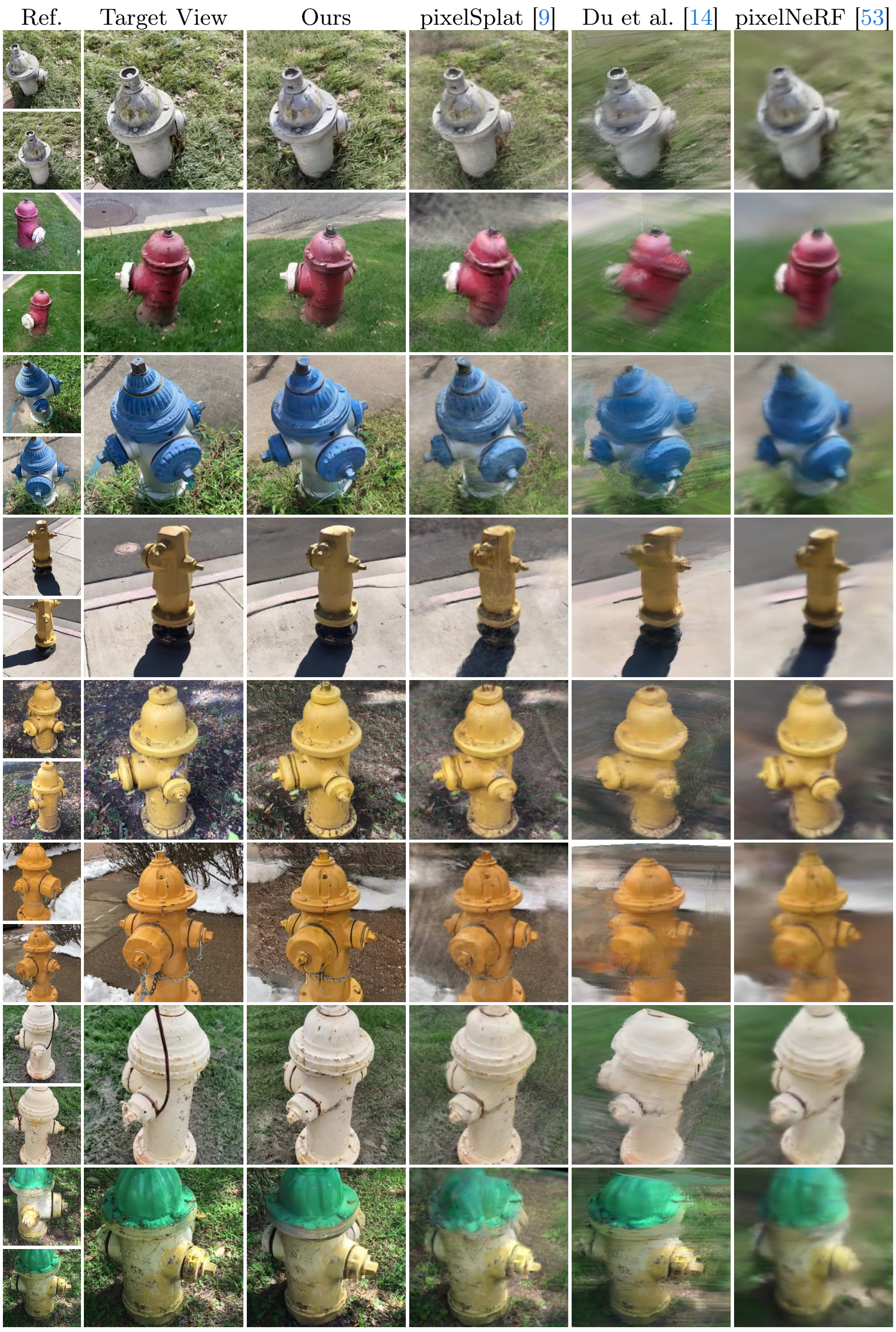

Fig. 8: Qualitative results for interpolation on CO3D hydrants [37].

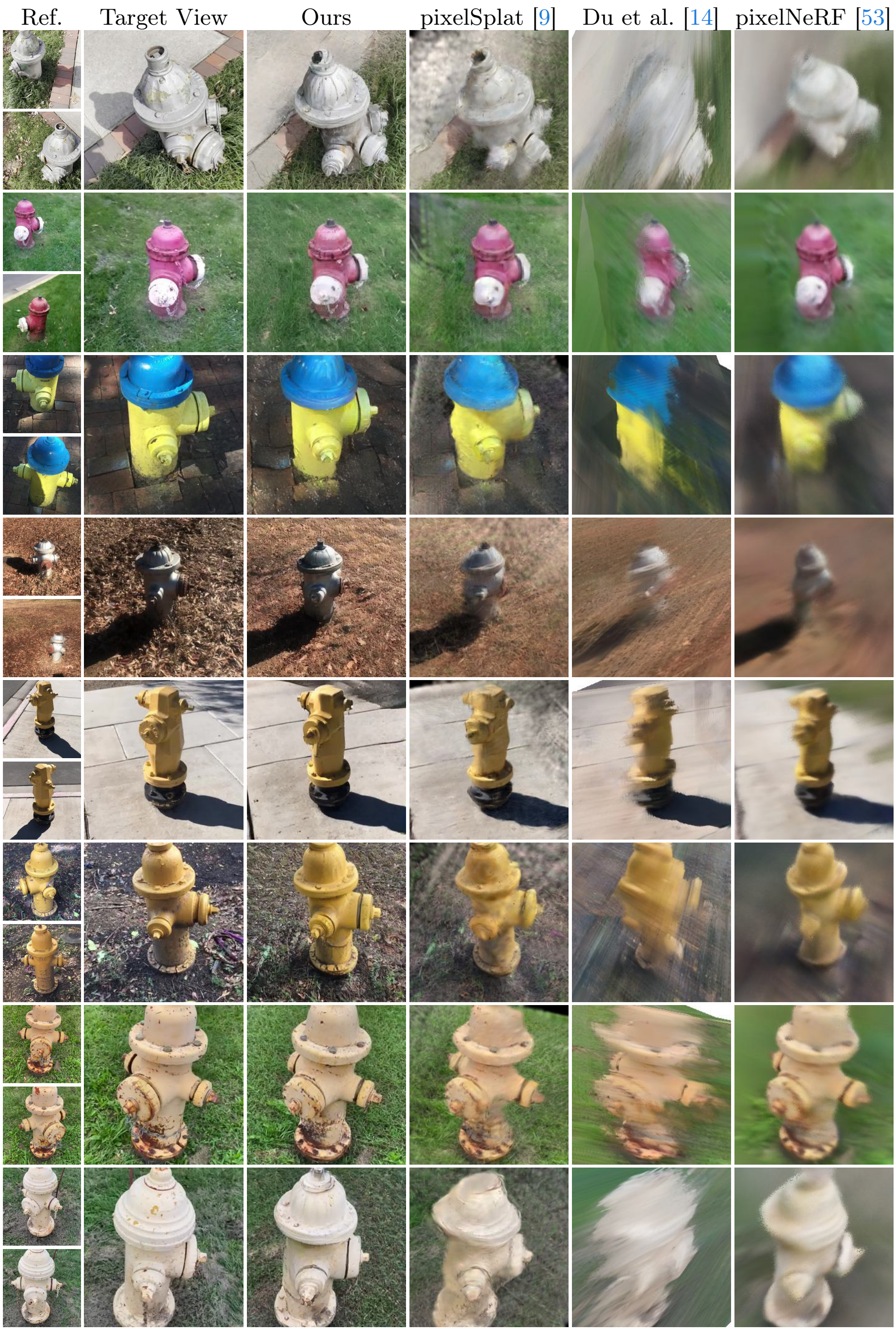

Fig. 9: Qualitative results for extrapolation on CO3D hydrants [37].

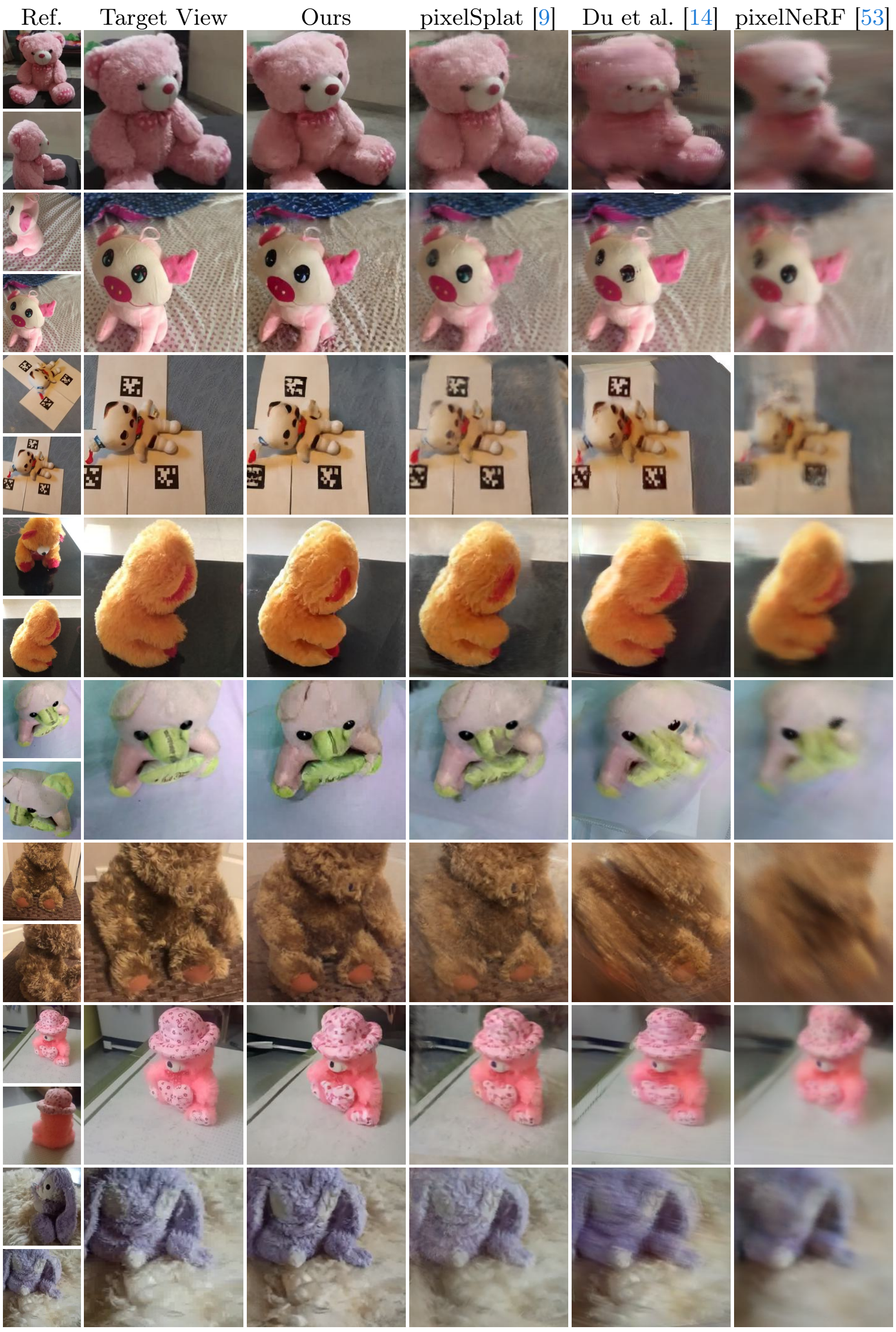

Fig. 10: Qualitative results for interpolation on CO3D teddybears [37].

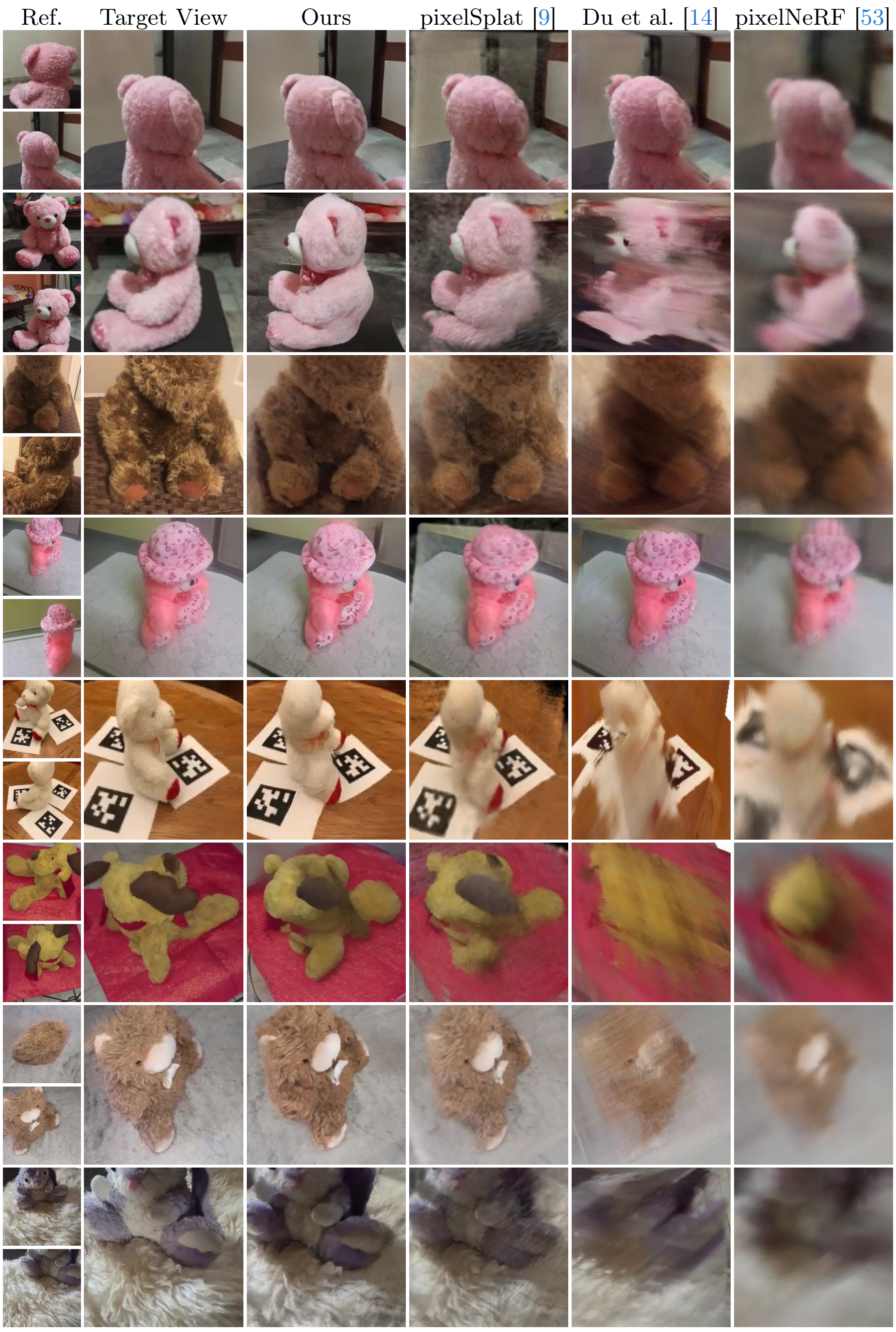

Fig. 11: Qualitative results for extrapolation on CO3D teddybears [37].

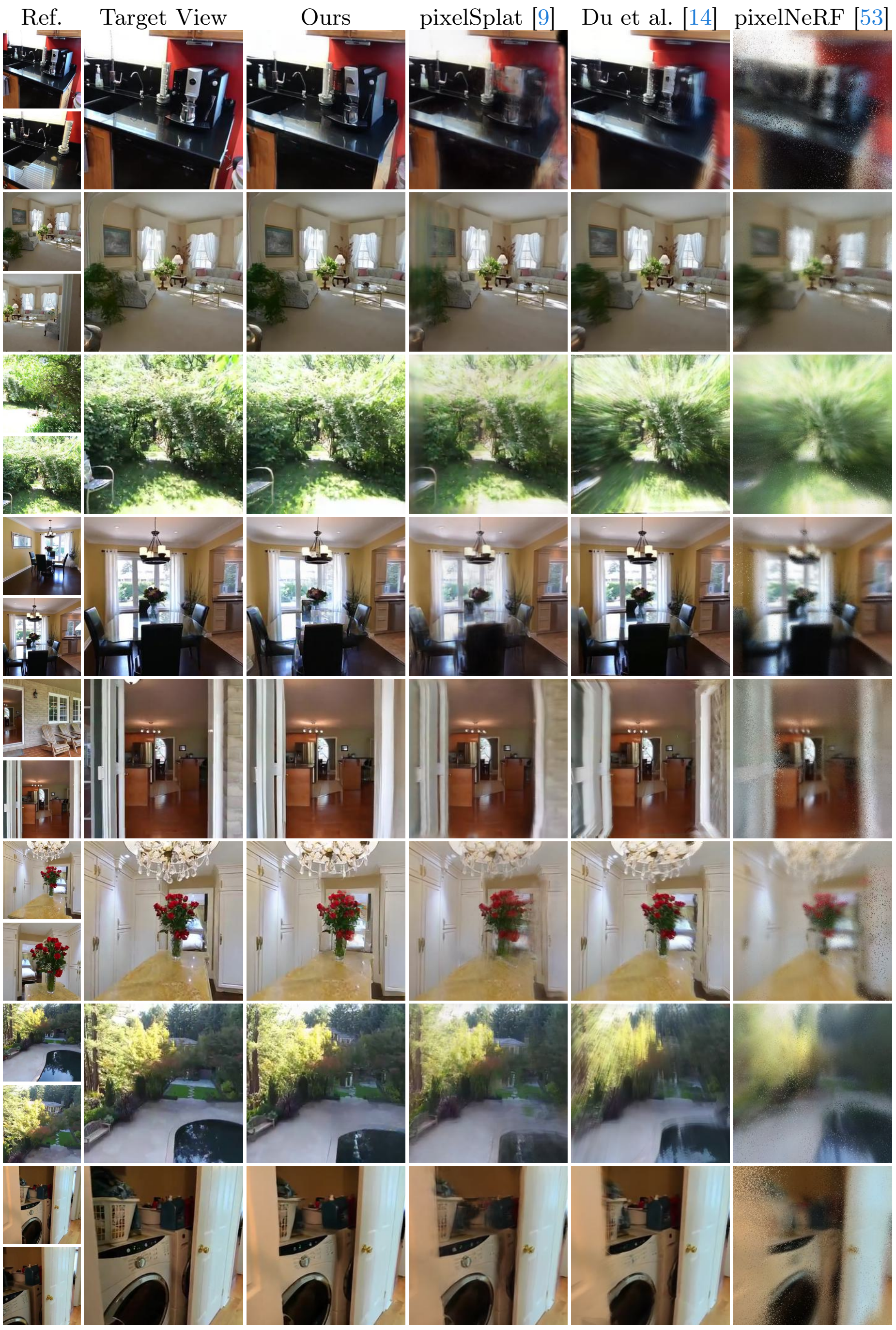

Fig. 12: Qualitative results for interpolation on RealEstate10k [56].

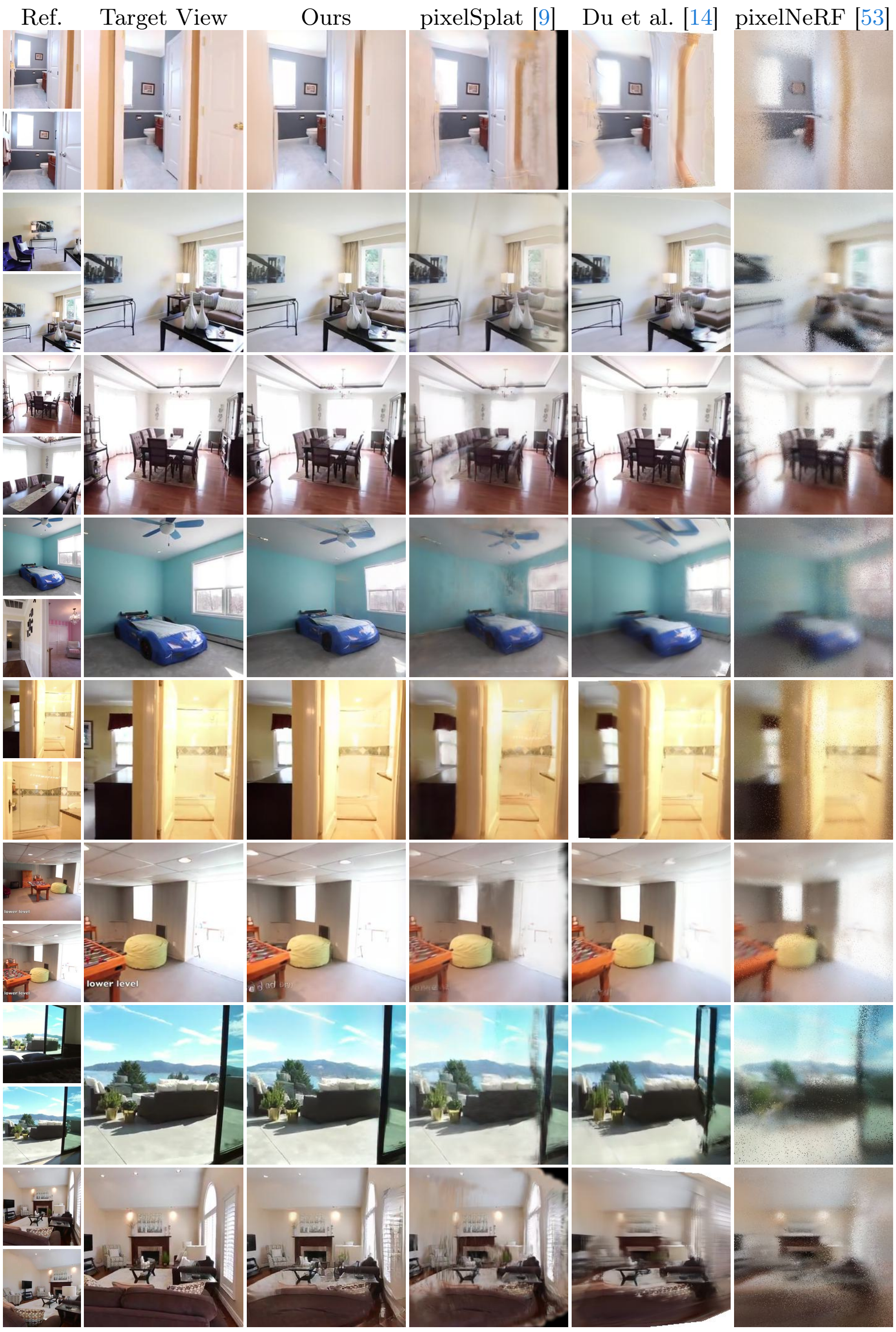

Fig. 13: Qualitative results for extrapolation on RealEstate10k [56].