SpotLessSplats: Ignoring Distractors in 3D Gaussian Splatting

Sara Sabour*[1,2], Lily Goli*[1,2], George Kopanas[1], Mark Matthews[1], Dmitry Lagun[1], Leonidas Guibas[1,3], Alec Jacobson[2], David J. Fleet[1,2], and Andrea Tagliasacchi[1,2,4]

[1] Google DeepMind
[2] University of Toronto
[3] Stanford University
[4] Simon Fraser University
https://spotlesssplats.github.io

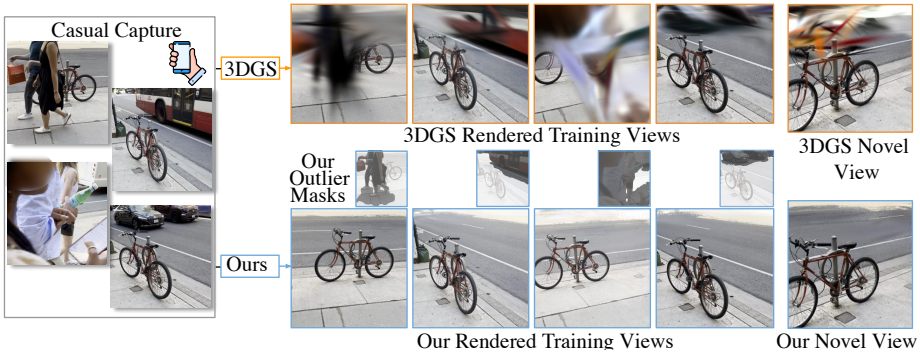


Fig. 1. SpotLessSplats cleanly reconstructs a scene with many transient occluders (**middle**), while avoiding artifacts (**bottom**). It correctly identifies and masks out all transients (**top**), even in captures with a large number of them (**left**).

Abstract. 3D Gaussian Splatting (3DGS) is a promising technique for 3D reconstruction, offering efficient training and rendering speeds, making it suitable for real-time applications. However, current methods require highly controlled environments—no moving people or wind-blown elements, and consistent lighting—to meet the inter-view consistency assumption of 3DGS. This makes reconstruction of real-world captures problematic. We present SpotLessSplats, an approach that leverages pre-trained and general-purpose features coupled with robust optimization to effectively ignore transient distractors. Our method achieves state-of-the-art reconstruction quality both visually and quantitatively, on casual captures.

1 Introduction

The reconstruction of 3D scenes from 2D images with neural radiance fields (NeRF) [27] and, more recently, with 3D Gaussian Splatting (3DGS) [17], has been the subject of intense focus in vision research. Most current methods assume that images are simultaneously captured, perfectly posed, and noise-free.

While these assumptions simplify 3D reconstruction, they rarely hold in real-world, where moving objects (e.g., people or pets), lighting variations, and other spurious photometric inconsistencies degrade performance, limiting widespread application.

In NeRF training, robustness to outliers has been incorporated by down-weighting or discarding inconsistent observations based on the magnitude of color residuals [6,25,39,50]. Similar methods adapted to 3DGS [10,19,48] address global appearance changes and single-frame transients seen in datasets like Phototourism [43]. Such captures include appearance changes occurring over weeks and different times of day, which are not common in most casual captures. For 3DGS in particular, the *adaptive densification* process itself introduces variance in color residuals, compromising detection of transients when directly applying existing ideas from robust NeRF frameworks.

In this paper we introduce SpotLessSplats (SLS), a framework for robust 3D scene reconstruction with 3DGS, via unsupervised detection of outliers in training images. Rather than detecting outliers in RGB space, we instead utilize a richer, *learned feature space* from text-to-image models. The meaningful semantic structure of this feature embedding allows one to more easily detect the spatial support of structured outliers associated, for example, with a single object. Rather than employing manually-specified robust kernels for outlier identification [39], we instead exploit adaptive methods in this feature space to detect outliers. To this end we consider two approaches within this framework. The first uses non-parametric clustering of local feature embeddings as a simple way to find image regions of structured outliers. The second uses an MLP, trained in an unsupervised fashion to predict the portion of the feature space that is likely to be associated with distractors. We further introduce a (complementary and general purpose) sparsification strategy, compatible with our robust optimization, that delivers similar reconstruction quality with two to four times fewer splats, even on distractor-free datasets, yielding significant savings in compute and memory. Through experiments on challenging benchmarks of casually captured scenes [36, 39], SLS is shown to consistently outperform competing methods in reconstruction accuracy.

2 Related work

Neural Radiance Fields (NeRF) [27], have gained widespread attention due to the high quality reconstruction and novel view synthesis of 3D scenes. NeRF represents the scene as a view dependent emissive volume. The volume is rendered using the absorption-emission part of the volume rendering equation [16]. Multiple enhancements have followed. Fast training and inference [8, 28, 45, 53], training with limited or single view(s) [15, 34, 54] and simultaneous pose inference [20,22,49] have brought radiance fields closer to practical applications. More recently, 3D Gaussian Splatting (3DGS) [17] was proposed as a primitive-based alternative to NeRFs with significantly faster rendering speed, while maintaining high quality. 3D Gaussians can be efficiently rasterized using alpha blending [58].

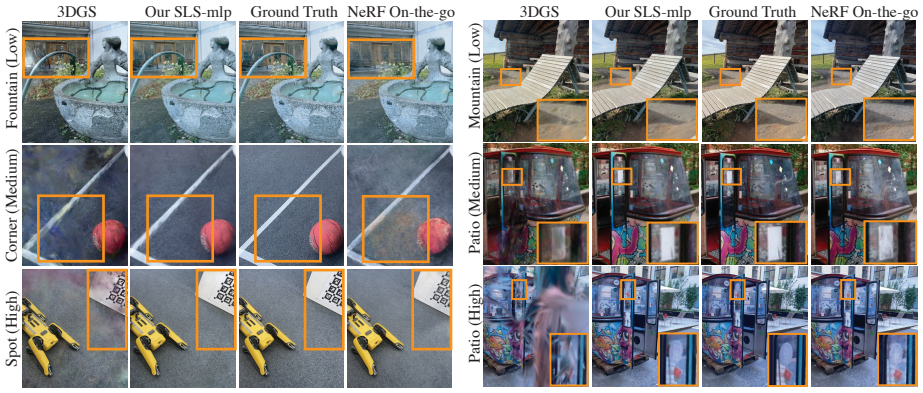
Fig. 2. Our outlier classification using clustered semantic features covers the distractor balloon fully, but an adapted robust mask from [39] misclassifies pixels with similar color to background, as inliers.

This simplified representation takes advantage of modern GPU hardware to facilitate real-time rendering. The efficiency and simplicity of 3DGS have prompted a shift in focus within the field, with many NeRF enhancements being quickly ported to 3DGS [5, 55].

Robustness in NeRF. The original NeRF paper made strong assumptions regarding the capture setup: the scene needs to be perfectly static, and the illumination should stay unchanged throughout the capture. More recently, NeRF has been extended to train on unstructured "in-the-wild" captured images that violate these constraints. Two influential works, NeRF-W [26] and RobustNeRF [39] addressed the problem of transient distractors, both using photometric error as guidance. NeRF-W [26] models a 3D uncertainty field rendered to 2D outlier masks that down-weight the loss at pixels with high-error, and a regularizer that prevents degenerate solutions. NeRF-W [26] also models global appearance via learned embeddings, which are useful for images captured over widely varying lighting and atmospheric conditions. Urban Radiance Fields (URF) [35] and Block-NeRF [46] similarly apply learned appearance embeddings to large-scale reconstruction. HA-NeRF [7] and Cross-Ray [52] model 2D outlier masks instead of 3D fields, leveraging CNNs or transformers for cross-ray correlations.

RobustNeRF [39], approached the problem from a robust estimator perspective, with binary weights determined by thresholded rendering error, and a blur kernel to reflect the assumption that pixels belonging to distractors are spatially correlated. However, both RobustNeRF and NeRF-W variants [7, 52] rely solely on RGB residual errors and because of this they often misclassify transients with colors similar to their background; see RobustMask in Figure 2. To avoid this, previous methods require careful tuning of hyper-parameters, i.e., the blur kernel size and thresholds in RobustNeRF and the regularizer weight in NeRF-W. On the contrary, our method uses the rich representation of text-to-image models for semantic outlier modeling. This avoids direct RGB error supervision, as it relies on feature-space similarities for clustering.

NeRF On-the-go [36] released a dataset of casually captured videos with transient occluders. Similar to our method, it uses semantic semantic features from DINOv2 [30] to predict outlier masks via a small MLP. However, it also relies on direct supervision from the structural rendering error, leading to potential over- or under-masking of outliers. This is illustrated in Figure 3, where over-masking has removed the hose ('Fountain') and has smoothed the carpet



	Fountain			Corner			Spot			Mountain			Patio			Patio High		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
MipNerf360	13.91	0.29	0.55	20.41	0.66	0.34	17.82	0.30	0.46	19.64	0.60	0.35	15.48	0.50	0.42	15.73	0.43	0.49
RobustNerf	17.20	0.41	0.54	20.21	0.70	0.35	16.40	0.38	0.69	18.07	0.49	0.49	17.55	0.53	0.45	12.99	0.35	0.61
NeRF On-the-go	20.11	0.61	0.31	24.22	0.80	0.19	23.33	0.79	0.19	20.15	0.64	0.26	20.78	0.75	0.22	21.41	0.72	0.24
3DGS	21.70	0.79	0.16	24.05	0.86	0.13	20.72	0.76	0.31	20.18	0.70	0.23	18.25	0.71	0.23	18.14	0.68	0.30
Our SLS-mlp	**22.81**	**0.80**	**0.15**	**26.43**	**0.90**	**0.10**	**25.76**	**0.90**	**0.12**	**22.53**	**0.77**	**0.18**	**22.24**	**0.86**	**0.10**	**22.84**	**0.83**	**0.16**

Fig. 3. Our method accurately reconstructs scenes with different levels of transient occlusion, avoiding leakage of transients or under-reconstruction evident by the quantitative and qualitative results on NeRF On-the-go [36] dataset.

('Spot'), while under-masking caused distractor leaks and foggy artifacts ('Corner', 'Spot'). NeRF-HuGS [6] combines heuristics from COLMAP's robust sparse point cloud [42], and off-the-shelf semantic segmentation to remove distractors. Both heuristics are shown to fail under heavy transient occlusions in [36].

Precomputed features. The use of precomputed vision features, such as DINO [4, 30] have demonstrated the ability to generalize to multiple vision tasks. Denoising Diffusion Probabalistic Models [13, 38, 44], known for their photorealistic image generation capabilities from text prompts [33, 37, 40], have been shown to have internal features similarly powerful in generalizing over many tasks e.g. segmentation and keypoint correspondence [1, 12, 24, 47, 57].

Robustness in 3DGS (concurrent works). Multiple concurrent works address 3DGS training on wild-captured data. SWAG [10] and GS-W [56] model appearance variation using learned global and local per-primitive appearance embeddings. Similarly, WE-GS [48] uses an image encoder to learn adaptations to the color parameters of each splat, per-image. Wild-GS [51] learns a spatial triplane field for appearance embeddings. All such methods [48, 51, 56] adopt an approach to outlier mask prediction like NeRF-W [26], with 2D outlier masks predicted to downweight high-error rendered pixels. SWAG [10] learns a per-image opacity for each Gaussian, and denotes primitives with high opacity variance as transients. Notable are SWAG [10] and GS-W [56] that show no or little improvement over the local/global appearance modeling, when additional learned transient masks are applied to Phototourism scenes [43]. SLS focuses on casual captures with longer duration transients and minimal appearance changes, common in video captures like those in the "NeRF on-the-go" dataset [36].

3 Background

We build our technique on top of 3D Gaussian Splatting [17], or 3DGS for brevity, which represents a 3D scene as a collection of 3D anisotropic Gaussians $\mathcal{G}=\{g_i\}$, henceforth referred to as splats. Given a set of posed images $\{\mathbf{I}_n\}_{n=1}^N$, $\mathbf{I}_i \in \mathbb{R}^{H\times W}$ of a casually captured scene, we aim to learn a 3DGS reconstruction $\mathcal{G}$ of the scene. Each splat g_i, is defined by a mean μ_i, a positive semi-definite covariance matrix $\boldsymbol{\Sigma}_i$, an opacity α_i, and view dependent color parameterized by spherical harmonics coefficients $\mathbf{c}_i$ [32].

The 3D scene representation is rendered to screen space by rasterization. The splat positions/means are rasterized to screen coordinates via classical projective geometry, while special care needs to be taken to rasterize the covariance matrix of each splat. In particular, if we denote with $\mathbf{W}$ the perspective transformation matrix, the projection of the 3D covariance to 2D screen space can be approximated following [58] as $\tilde{\boldsymbol{\Sigma}} = \mathbf{JW}\boldsymbol{\Sigma}\mathbf{W}^T\mathbf{J}^T$, where $\mathbf{J}$ is the Jacobian of the projection matrix, which provides a linear approximation to the non-linear projection process. To ensure $\boldsymbol{\Sigma}$ represents covariance throughout optimization (i.e., positive semi-definite), the covariance matrix is parameterized as $\boldsymbol{\Sigma} = \mathbf{RSS}^T\mathbf{R}^T$, where scale $\mathbf{S}$=diag($\mathbf{s}$) with $\mathbf{s}\in\mathbb{R}^3$, and rotation $\mathbf{R}$ is computed from a unit Quaternion q. Once splat positions and covariances in screen-spaces are computed, the image formation process executes volume rendering as alpha-blending, which in turn requires splat sorting along the view direction. Unlike NeRF, which renders *one pixel at a time*, 3DSG renders the *entire image* in a single forward pass.

3.1 Robust optimization of 3DGS

Unlike typical capture data for 3DGS [17], we do not assume the set of posed images $\{\mathbf{I}_n\}_{n=1}^N$ to be curated, but rather *casually captured*. That is, we *do not* require images to be depictions of a perfectly 3D consistent and static world. Following prior work, we (interchangeably) denote the portion of images that break these assumptions as *distractors* [39] or *transient effects* [25]. And unlike prior works [18, 25, 46], we do not make assumptions about the transient object class, appearance and/or shape.

We address this problem by taking inspiration from the pioneering work of [39] in RobustNeRF, which removes distractors by identifying the portion of input images that should be masked out in the optimization process. The problem reduces to predicting (without supervision) inlier/outlier masks $\{\mathbf{M}_n\}_{n=1}^N$ for each training image, and optimizing the model via a *masked* L1 loss:

$$\arg\min_{\mathcal{G}} \sum_{n=1}^{N} \mathbf{M}_n^{(t)} \odot \|\mathbf{I}_n - \hat{\mathbf{I}}_n^{(t)}\|_1. \tag{1}$$

where $\hat{\mathbf{I}}_n^{(t)}$ is a rendering of $\mathcal{G}$ at training iteration (t). As in RobustNeRF [39], transient effects can be detected by observing photometric inconsistencies during

Fig. 4. Lower and upper error residual labels provide a weak supervision for training an MLP classifier for detecting outlier distractors.

training; that is, image regions that are associated with a large loss value. By denoting with $\mathbf{R}_n^{(t)}=\|\mathbf{I}_n - \hat{\mathbf{I}}_n^{(t)}\|_1$ the image of residuals (with a slight abuse of notation, as the 1-norm is executed *pixel-wise*, along the color channel), the mask is computed as:

$$\mathbf{M}_n^{(t)}=\mathbb{1}\left\{\left(\mathbb{1}\{\mathbf{R}_n^{(t)}>\rho\} \circledast \mathbf{B}\right)>0.5\right\},\ P(\mathbf{R}_n^{(t)}>\rho)=\tau \tag{2}$$

where $\mathbb{1}$ is an indicator function returning 1 if the predicate is true and 0 otherwise, ρ is a generalized median with τ being a hyper-parameter controlling the cut-off percentile[5], and $\mathbf{B}$ is a (normalized) 3×3 box filter that performs a morphological dilation via convolution ($\circledast$). Intuitively, RobustNeRF [39], summarized by Eq. (2) above, extends a *trimmed robust estimator* [9] by assuming that inliers/outliers are *spatially correlated*. We found that directly applying ideas from [39] to 3DGS, even when not limited by cases of misleading color residual like those depicted in Figure 2, do not remove outliers effectively. Rather, several adaptations are necessary in order to accommodate differences in the representation and training process of 3DGS; see Section 4.2.

4 Method

The outlier mask in Eq. (2) is built solely based on photometric errors in the novel view synthesis process. Conversely, we propose to identify distractors based on their semantics, *recognizing* their re-occurrence during the training process. We consider semantics as *feature maps* computed from a self-supervised 2D foundation model (e.g. [47]). The process of removing distractors from training images then becomes one of identifying the sub-space of features that are likely to cause large photometric errors. As an example, consider a dog walking around in an otherwise perfectly static scene. We would like to design a system that either spatially in each image (Sec. 4.1) or more broadly, spatio-temporally in the dataset (Sec. 4.1), recognizes "dog" pixels as the likely cause of reconstruction problems, and automatically removes them from the optimization. Our method is designed to reduce reliance on local color residuals for outlier detection and over-fitting to color errors, and instead emphasizing reliance on semantic feature similarities between pixels. We thus refer to our methods as "clustering." In Section 4.1 we

[5] If τ=.5 then ρ=median($\mathbf{R}_n^{(t)}$)

detail how to achieve this objective. In Section 4.2 we then detail several key adjustments to adapt the ideas from RobustNeRF [39] to a 3DGS training regime; see Section 4.1.

4.1 Recognizing distractors

Given the input images $\{\mathbf{I}_n\}_{n=1}^N$, we pre-compute feature maps for each image using Stable Diffusion [38] as proposed by [47], resulting in feature maps $\{\mathbf{F}_n\}_{n=1}^N$. This pre-processing step is executed *once* before our training process starts. We then employ these feature maps to compute the inlier/outlier masks $\mathbf{M}^{(t)}$; we drop the image index n to simplify notation, as the training process involves one image per batch. We now detail two different ways to detect outliers.

Spatial clustering In the pre-processing stage, we additionally perform unsupervised clustering of image regions. Similar to super-pixel techniques [14, 21], we over-segment the image into a fixed cardinality collection of C spatially connected components; see 'Clustered Features' Fig. 2. In more detail, we execute agglomerative clustering [29] on the feature map $\mathbf{F}$, where each pixel is connected to its 8 surrounding pixels. We denote the clustering assignment of pixel p into cluster c as $\mathbf{C}[c,p]\in\{0,1\}$, and clustering is initialized with every pixel in its own cluster. Clusters are agglomerated greedily, collapsing those that cause the least amount of inter-cluster feature variance differential before/post collapse. Clustering terminates when C=100 clusters remain.

We can then calculate the probability of cluster c being an inlier from the percentage of its inlier pixels in Eq. (2):

$$P(c \in \mathbf{M}^{(t)}) = \Big(\sum_p \mathbf{C}[c,p] \cdot \mathbf{M}^{(t)}[p]\Big) \;/\; \sum_p \mathbf{C}[c,p], \tag{3}$$

and then propagate the cluster labels back to pixels as:

$$\mathbf{M}^{(t)}_{\text{agg}}(p) = \sum_c \mathbb{1}\{P(c \in \mathbf{M}^{(t)}) > 0.5\} \cdot \mathbf{C}[c,p] \tag{4}$$

We then use $\mathbf{M}^{(t)}_{\text{agg}}$, rather than $\mathbf{M}^{(t)}$, as inlier/outlier mask to train our 3DGS model in Eq. (1). We designate this model configuration as 'SLS-agg'.

Spatio-temporal clustering A second approach is to train a *classifier* that determines whether or not pixels should be included in the optimization Eq. (1), based on their associated features. To this end we use an MLP with parameters θ that predicts pixel-wise inlier probabilities from pixel features:

$$\mathbf{M}^{(t)}_{\text{mlp}} = \mathcal{H}(\mathbf{F}; \theta^{(t)}). \tag{5}$$

As the $\theta^{(t)}$ notation implies, the classifier parameters are updated *concurrently* with 3DGS optimization. $\mathcal{H}$ is implemented with 1×1 convolutions, and hence

acts in an i.i.d. fashion across pixels. We interleave the optimization of the MLP and the 3DGS model, such that the parameters of one are fixed while the other's are optimized, in a manner similar to alternating optimization.

The MLP classifier loss is given by

$$\mathcal{L}(\theta^{(\mathrm{t})}) = \mathcal{L}_{sup}(\theta^{(\mathrm{t})}) + \lambda \mathcal{L}_{reg}(\theta^{(\mathrm{t})}), \tag{6}$$

with λ=0.5, and where $\mathcal{L}_{sup}$ supervises the classifier:

$$\begin{aligned} \mathcal{L}_{sup}(\theta^{(\mathrm{t})}) = \max(\mathbf{U}^{(\mathrm{t})} - \mathcal{H}(\mathbf{F}; \theta^{(\mathrm{t})}), 0) \\ + \max(\mathcal{H}(\mathbf{F}; \theta^{(\mathrm{t})}) - \mathbf{L}^{(\mathrm{t})}, 0) \end{aligned} \tag{7}$$

and $\mathbf{U}$ and $\mathbf{L}$ are self-supervision labels computed from the mask of the current residuals:

$$\mathbf{U}^{(\mathrm{t})} = \mathbf{M}^{(\mathrm{t})} \text{ from Eq. (2) with } \tau = .5 \tag{8}$$

$$\mathbf{L}^{(\mathrm{t})} = \mathbf{M}^{(\mathrm{t})} \text{ from Eq. (2) with } \tau = .9 \tag{9}$$

In other words, we directly supervise the classifier only on pixels for which we can confidently determine the inlier status based on reconstruction residuals, and otherwise we heavily rely on semantic similarity in the feature space; see Figure 4. To further regularize $\mathcal{H}$ to map similar features to similar probabilities, we minimize its Lipschitz constant via $\mathcal{L}_{reg}$ as detailed in [23, Eq. (13)]. We then use $\mathbf{M}^{(\mathrm{t})}_{\mathrm{mlp}}$, instead of $\mathbf{M}^{(\mathrm{t})}$, as inlier/outlier mask to train 3DGS in Eq. (1). We designate this configuration as 'SLS-mlp'. As we are *co-training* our classifier together with the 3DGS model, additional care is needed in its implementation; see Section 4.2.

4.2 Adapting 3DGS to robust optimization

Directly applying any robust masking techniques to 3DGS can result in the robust mask overfitting to a premature 3DGS model (Sec. 4.2), with inlier estimator becoming skewed by image-based training (Sec. 4.2), or the densification tactics (Sec. 4.2) of 3DGS. We propose solutions to these issues in what follows.

Warm up with scheduled sampling We find it important to apply masks gradually, because the initial residuals are random. This is doubly true if we use the learned clustering for masking since the MLP will not have converged early in the optimization, and predicts random masks. Further, direct use of the outlier mask tends to quickly overcommit to outliers, preventing valuable error back-propagation and learning from those regions. We mitigate this by formulating our masking policy for each pixel as sampling from a Bernoulli distribution based on the masks:

$$\mathbf{M}^{(\mathrm{t})} \sim \mathcal{B}\left(\alpha \cdot 1 + (1-\alpha) \cdot \mathbf{M}^{(\mathrm{t})}_{*}\right) ; \tag{10}$$

where α is a staircase exponential scheduler, going from one to zero, providing a warm-up. This allows us to still sparsely sample gradients in areas we are not confident about, leading to better classification of outliers.

Trimmed estimators in image-based training As [39] implements a *trimmed* estimator, the underlying assumption is that each minibatch (on average) contains the same proportion of outliers. This assumption is broken in a 3DGS training run, where each minibatch is a *whole* image, rather than a random set of pixels drawn from the set of training images. This creates a challenge in the implementation of the generalized median of Eq. (2), as the distribution of outliers is skewed between images.

We overcome this by tracking residual magnitudes over multiple training batches. In particular, we discretize residual magnitudes into B histogram buckets of width equal to the lower bound of rendering error (10^{-3}). We update the likelihood of each bucket at each iteration via a discounted update to the bucket population, similar to fast median filtering approaches [31]. This maintains a moving estimate of residual distribution, with constant memory consumption, from which we can extract our generalized median value ρ as the τ quantile in the histogram population.

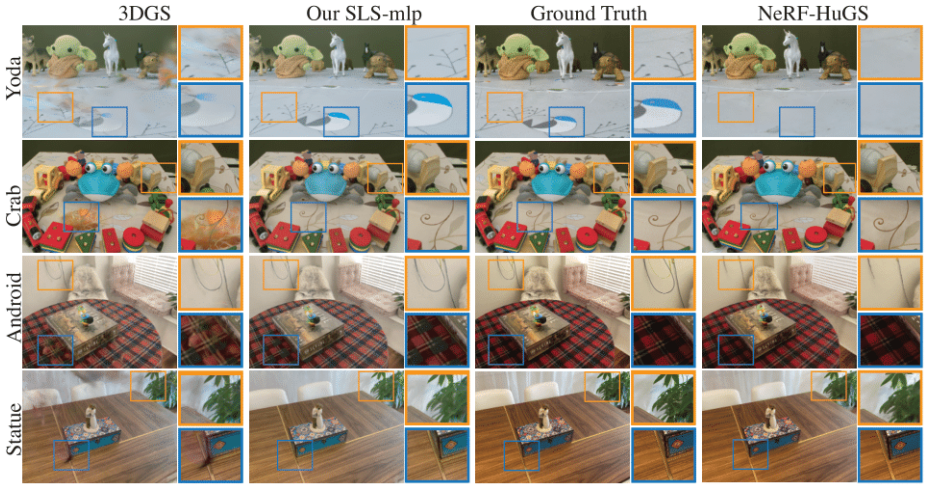
A friendly alternative to "opacity reset" [17] proposed to reset the opacity of all Gaussians every M iterations. This opacity reset is a mechanism that deals with two main problems. First, in challenging datasets the optimization has the tendency to accumulate Gaussians close to the cameras. These are often referred to as *floaters* in the literature. Floaters are hard to deal with because they force camera rays to saturate their transmittance early and as a result gradients do not have a chance to flow through the occluded parts of the scene. Opacity reset lowers the opacity of all Gaussians such that gradients can flow again along the whole ray. Second, opacity reset acts as a control mechanism for the number of Gaussians. Resetting opacity to a low value allows for Gaussians that never recover a higher opacity to be pruned by the density control mechanism [17].

However, opacity reset interferes with residual distribution tracking (Sec. 4.2), causing residuals to become artificially large in the iterations following opacity reset. Simply disabling does not work due to it's necessity to the optimization. Following [11], we instead propose utilization-based pruning (UBP). We track the gradient of the rendered colors with respect to the projected splat positions[6] x_g of each Gaussian g. Computing the derivative with respect to projected positions, as opposed to 3D positions, allows for a less memory-intensive GPU implementation, while providing a similar metric as in [11]. More concretely, we define the utilization as:

$$u_g = \sum_{t \in \mathcal{N}_T(t)} \mathbb{E}_{w,h} \left\| \mathbf{M}_{h,w}^{(t)} \cdot \frac{\partial \hat{\mathbf{I}}_{h,w}^{(t)}}{\partial x_g^{(t)}} \right\|_2^2 \tag{11}$$

We average this metric across the image ($W \times H$), computing it every T=100 steps accumulated across the previous set of $|\mathcal{N}_T(t)|$=100 images. We prune

[6] Please carefully note that this is the gradient of the rendered image with respect to Gaussian positions, and not the gradient of the loss.



	Statue			Android			Yoda			Crab (1)		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
MipNerf360	19.86	.69	.23	21.81	.69	.18	23.75	.77	.22	29.25	.92	.09
RobustNerf	20.60	.76	.15	23.28	.75	.13	29.78	.82	.15	32.22	.94	.06
NeRF On-the-go	21.58	.77	.24	23.50	.75	.21	29.96	.83	.24	-	-	-
NeRF-HuGS	21.00	.77	.18†	23.32	.76	.20†	30.70	.83	.22†	34.16	.96	**.07**†
3DGS	21.68	.83	.14	23.33	.80	.15	27.15	.92	.13	31.80	.96	.08
Our SLS-mlp	**22.69**	**.85**	**.12**	**25.15**	**.86**	**.09**	**33.60**	**.96**	**.10**	**35.85**	**.97**	.08
3DGS on clean	28.02	.95	.05	25.42	.87	.07	33.69	.94	.12	-	-	-
3DGS* on clean	28.63	.95	.04	25.38	.87	.07	36.34	.97	.07	-	-	-

Fig. 5. Quantitative and qualitative evaluation on RobustNeRF [39] datasets show that SLS outperforms baseline methods on 3DGS and NeRF, by preventing over- or under-masking. † denotes VGG LPIPS computed on NeRF-HuGS results rather than AlexNet LPIPS reported in NeRF-HuGS. 3DGS* denotes 3DGS with UBP.

Gaussians whenever $u_g < \kappa$, with $\kappa = 10^{-8}$. Replacing opacity reset with utilization-based pruning achieves both original goals of opacity reset while alleviating interference to our residual distribution tracking. Utilization-based pruning significantly compresses scene representation by using fewer primitives while achieving comparable reconstruction quality even in outlier-free scenes; see Section 5.2. It also effectively deals with floaters; see Figure 8. Floaters, naturally, have low utilization as they participate in the rendering of very few views. Furthermore, using masked derivatives as in Eq. (11) allows for the removal of any splat that has leaked through the robust mask in the warm-up stage.

Appearance modeling While [17] assumed that the images of a scene (up to distractors) are perfectly photometrically consistent, this is rarely the case for casual captures typically employing automatic exposure and white-balance. We address this by incorporating the solution from [35] adapted to the view-dependent colors represented as spherical harmonics from [17]. We co-optimize a latent $z_n \in \mathbb{R}^{64}$ per input camera view, and map this latent vector via an MLP to a linear transformation acting on the harmonics coefficients $\mathbf{c}$:

$$\hat{\mathbf{c}}_i = \mathbf{a} \odot \mathbf{c}_i + \mathbf{b}, \quad \mathbf{a}, \mathbf{b} = \mathcal{Q}(\mathbf{z}_n; \theta_{\mathcal{Q}}) \tag{12}$$

	Android	Statue	Crab (2)	Yoda	Mountain	Fountain	Corner	Patio	Spot	Patio-High	Average
3DGS	23.33 ± 0.13	21.68 ± 0.16	29.74 ± 0.37	27.15 ± 0.61	20.90 ± 0.18	21.85 ± 0.27	23.39 ± 0.43	18.33 ± 0.27	21.50 ± 0.85	18.06 ± 0.71	22.60
RobustFilter	24.50 ± 0.05	22.70 ± 0.06	31.34 ± 0.13	33.23 ± 0.13	22.29 ± 0.07	22.59 ± 0.07	25.20 ± 0.10	18.16 ± 0.19	25.54 ± 0.08	23.01 ± 0.18	24.85
SLS-agg	24.94 ± 0.08	**23.16 ± 0.08**	33.50 ± 0.14	35.01 ± 0.21	22.65 ± 0.14	23.03 ± 0.17	26.33 ± 0.10	22.31 ± 0.13	**26.34 ± 0.37**	**23.54 ± 0.15**	26.08
SLS-mlp w/o UBP	25.08 ± 0.04	22.75 ± 0.14	**34.43 ± 0.03**	**34.36 ± 0.24**	**22.93 ± 0.09**	**23.19 ± 0.13**	**26.74 ± 0.13**	**22.36 ± 0.07**	25.95 ± 0.47	23.27 ± 0.13	**26.11**
SLS-mlp w/ UBP	**25.15 ± 0.05**	22.69 ± 0.16	33.63 ± 0.27	33.60 ± 0.30	22.53 ± 0.11	22.81 ± 0.10	26.43 ± 0.08	22.24 ± 0.19	25.76 ± 0.15	22.84 ± 0.32	25.77

Fig. 6. We ablate our different robust masking methods on [39] and [36] datasets. The reconstruction metrics and qualitative masks illustrate the performance of SLS-agg Eq. (4) and SLS-mlp Eq. (5) over a basic RobustFilter Eq. (2) adapted from [39], and baseline vanilla 3DGS [17]. The final row enables Utility-Based Pruning (UBP) (Sec. 4.2). All methods use opacity reset disabled, the same scheduling in Eq. (10), and GLO Eq. (12) enabled on all runs including 3DGS. SLS-agg and SLS-mlp are mostly within 2σ of each other on all tasks. The σ is calculated from 5 independent runs each.

where $\odot$ is the Hadamard product, $\mathbf{b}$ models changes in brightness, and $\mathbf{a}$ provides the expressive power for white-balance. During optimization, the trainable parameters also include $\theta_{\mathcal{Q}}$ and $\{\mathbf{z}_n\}$. Such a reduced model can prevent $\mathbf{z}_n$ from explaining distractors as per-image adjustments, as would happen in a simpler GLO [25]; see [35] for an analysis.

5 Results

In what follows, we compare our proposed method on established datasets of casual distractor-filled captures (Sec. 5.1), comparing with other methods. We then investigate the effect of our proposed opacity reset alternative pruning (Sec. 5.2). Finally, we report a complete analysis of different variants of our clustering, along with an ablation study of our design choices (Sec. 5.3).

Datasets. We evaluate our method on the RobustNeRF [39] and NeRF on-the-go [36] datasets of *casual captures*. The RobustNeRF dataset includes four scenes with distractor-filled and distractor-free training splits, allowing us to compare a robust model with a 'clean' model trained on distractor-free images. All models are evaluated on a clean test set. The 'Crab' and 'Yoda' scenes feature variable distractors across images, not captured in a single casual video, but these exact robotic capture with twin distractor-free and distractor-filled images allow a fair comparison to the 'clean' model. Note the (originally released) Crab (1) scene had a test set with same set of views as those in the train set, which is fixed in Crab (2). We compare previous methods on Crab (1), and present full results on Crab (2) in Section 5.3. The NeRF on-the-go dataset has six scenes with three levels of transient distractor occlusion (low, medium, high) and a separate clean test set for quantitative comparison.

Baselines. Distractor-free reconstruction has yet to be widely addressed by 3D Gaussian Splatting methods. Existing methods mostly focus on global appearance changes such as brightness variation [10, 19, 48], and do not focus on

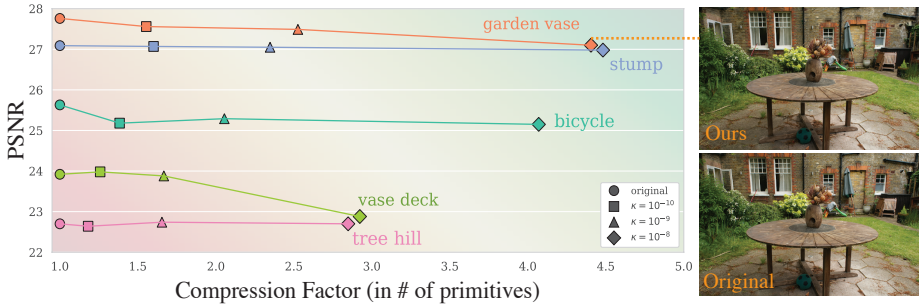


Fig. 7. Quantitative and qualitative results on MipNeRF360 [3] dataset shows gradient-based pruning can reduce the number of Gaussians up to 4.5× with only marginal degradation of image quality.

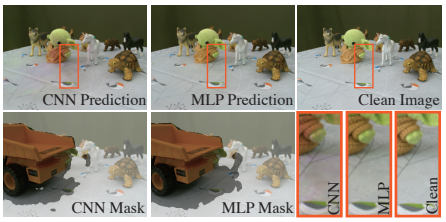
the distractor-filled datasets of casual captures curated for this task. We therefore compare against vanilla 3DGS and robust NeRF methods. We further add GLO to the vanilla 3DGS baseline to be comparable with MipNeRF360 results that have GLO enabled. We compare against state-of-the-art NeRF methods, NeRF on-the-go [36], NeRF-HuGS [6] and RobustNeRF [39]. We also include MipNeRF-360 [3] as a baseline for NeRF.

Metrics. We compute the commonly used image reconstruction metrics of PSNR, SSIM and LPIPS. We use normalized VGG features, as most do, when computing LPIPS metrics. NeRF-HuGS [6] reports LPIPS metrics from AlexNet features; for fair comparison, we compute and report VGG LPIPS metrics on their released renderings. Finally, note NeRF on-the-go does not evaluate on 'Crab', because of the aforementioned issue.

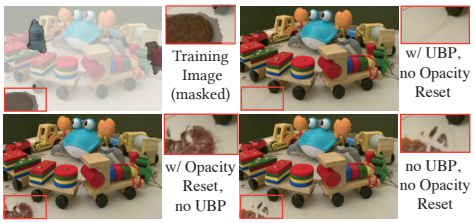
Implementation details. We train our 3DGS models with the same hyperparameters as in the officially released codebase. All models are trained for 30k iterations. We turn off the opacity-reset and only reset the non-diffuse spherical harmonic coefficients to 0.001 at the 8000th step. This ensures that any distractors leaked in the earlier stages of MLP training do not get modeled as view dependent effects. We run UBP every 100 steps, from the 500th to 15000th step. For MLP training, we use the Adam optimizer with a 0.001 learning rate. We compute image features from the 2nd upsampling layer of Stable diffusion v2.1, denoising time step of 261, and an empty prompt. [47] found this configuration most efficient for segmentation and keypoint correspondence tasks. We concatenate positional encoding of degree 20 to the features as input to the MLP.

5.1 Distractor-free 3D reconstruction

We evaluate our method by preforming 3D reconstruction on the RobustNeRF and NeRF on-the-go datasets. In Figure 5, we quantitatively show that SLS-mlp outperforms all the robust NeRF-based baselines on the RobustNeRF dataset. We improve significantly upon vanilla 3DGS, while having closer performance to the clean models, specifically on 'Yoda' and 'Android'. We further qualitatively



	SLS-mlp	CNN	SLS-mlp ($\lambda = 0.1$)	SLS-mlp ($\lambda = 1$)	SLS-agg	SLS-agg ($C = 10$)	SLS-agg ($C = 1000$)
PSNR ↑	28.72	27.91	28.71	28.61	28.91	27.71	29.00
SSIM ↑	0.90	0.89	0.90	0.89	0.90	0.89	0.90
LPIPS ↓	0.10	0.13	0.11	0.12	0.10	0.11	0.09



	SLS-mlp	No UBP	Opacity Reset	No GLO	No α	No $\mathcal{B}$
PSNR ↑	28.72	29.27	27.64	28.43	28.41	28.41
SSIM ↑	0.90	0.90	0.89	0.89	0.90	0.90
LPIPS ↓	0.10	0.09	0.11	0.11	0.11	0.11

Fig. 8. (**Left**) Ablation study on Section 4.1 on the design choices in SLS-mlp and SLS-agg. (**Right**) Ablation study on the different training dynamics.

compare with vanilla 3DGS and NeRF-HuGS. The qualitative results show that vanilla 3DGS tries to model distractors as noisy floater splats ('Yoda', 'Statue') or view-dependent effects ('Android') or a mixture of both ('Crab'). NeRF-HuGS [6] which uses segmentation-based masks shows signs of over-masking (removing static parts in all four scenes), or under-mask in challenging sparsely sampled views letting in transients ('Crab').

In Figure 3, we perform a similar analysis on the NeRF On-the-go [36] dataset. While we show superior quantitative results to both SOTA robust NeRF methods, we also achieve a significant performance boost compared to vanilla 3DGS. For low occlusion scenes the robust initialization of vanilla 3DGS from COLMAP [41] point clouds, specifically RANSAC's rejection of outliers, is enough to yield good reconstruction quality. However, as the distractor density increases, 3DGS reconstruction quality drops with qualitative results showing leakage of distractor transients. Additionally, qualitative results show that NeRF On-the-go fails to remove some of the distractors included in the early stages of training ('Patio', 'Corner', 'Mountain' and 'Spot'), showing further signs of overfitting to the rendering error. This is seen in the over-masking of fine details ('Patio High') or bigger structures ('Fountain') removed completely.

5.2 Effect of utilization-based pruning

In all our experiments, enabling our proposed utilization-based pruning (UBP) (Sec. 4.2), decreases the number of Gaussians from 4× to 6×. This compression translates to at least a 2× reduction in training time with UBP enabled and 3× during inference. Figure 8 shows that enabling UBP may degrade quantitative measurements slightly, but in practice the final renderings are cleaner with less floaters (e.g. bottom left of the image). Similar observations indicate that metrics such as PSNR and LPIPS may not completely reflect the presence of floaters as clearly as a rendered video. Given the substantial reduction in number of Gaussians, we propose UBP as a compression technique applicable to cluttered, *as well as clean*, datasets. Figure 7 shows that on clean MipNeRF360 [2] datasets, using UBP instead of opacity reset reduces the number of Gaussians from 2× to 4.5× while preserving rendering quality.

5.3 Ablation study

In Figure 6, we compare the performance of SLS with a progression of other robust masking techniques. The progression begins with a naive application of a robust filter (2), followed by the application of SLS-agg, and finally the use of an MLP in SLS-mlp. We demonstrate that both SLS-agg and SLS-mlp are capable of effectively removing distractors from the reconstructed scene, while maintaining maximal coverage of the scene. Further, in Figure 8 we ablate our choices in both architectural design, and the adaptations proposed in Section 4.2. Fig. 8 shows that using an MLP instead of a small CNN (both roughly having 30K parameters, and two non-linear activations) can adapt better to subtle transients like shadows. The choice of regularizer weight λ seems to have little effect. In agglomerative clustering, more clusters generally lead to better results, with gains diminishing after 100 clusters. Figure 8 further illustrates the effectiveness of UBP in removing leaked distractors. Our other adaptations, GLO, warm-up stage and Bernoulli sampling all show improvements.

6 Conclusion

We have presented SpotLessSplats, a method for transient distractor suppression for 3DGS. We established a class of masking strategies that exploit semantic features to effectively identify transient distractors without any explicit supervision. Specifically, we proposed a spatial clustering method 'SLS-agg' that is fast and does not require further training, simply assigning an inlier-outlier classification to each cluster. We then proposed a spatio-temporal learned clustering based on training a light-weight MLP simultaneously with the 3DGS model, 'SLS-mlp', that allows for higher precision grouping of semantically associated pixels, while marginally slower than clustering. Our methods leverage the semantic bias of Stable Diffusion features and robust techniques to achieve state of the art suppression of transient distractors. We also introduced a gradient-based pruning method that offers same reconstruction quality as vanilla 3DGS, while using significantly lower number of splats, and is compatible with our distractor suppression methods. We believe that our work is an important contribution necessary for widespread adoption of 3DGS to real-world in-the-wild applications.

Limitations. Our reliance on text-to-image features, although generally beneficial for robust detection of distractors, imposes some limitations. One limitation is that when distractor and non-distractors of the same semantic class are present and in close proximity, they may not be distinguished by our model. Further, the low-resolution features these models provide can miss thin structures such as the balloon string of Figure 6. Especially in the use of clustering, upsampling the features to image resolution results in imprecise edges. Our pruning strategy, is based on epistemic uncertainty computation per primitive which is effective in removing lesser utilized Gaussians. However if the uncertainty is thresholded too aggressively (e.g. 'vase deck' in Fig. 7), it can remove parts of the scene that are rarely viewed in the training data.

References

1. Amir, S., Gandelsman, Y., Bagon, S., Dekel, T.: Deep ViT Features as Dense Visual Descriptors. What is Motion For Workshop in ECCV (2022) 4
2. Barron, J., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.: Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. In: ICCV (October 2021), https://arxiv.org/abs/2103.13415 13
3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In: CVPR (2022) 12
4. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging Properties in Self-Supervised Vision Transformers. In: ICCV (2021) 4
5. Charatan, D., Li, S.L., Tagliasacchi, A., Sitzmann, V.: pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In: CVPR (2024) 3
6. Chen, J., Qin, Y., Liu, L., Lu, J., Li, G.: Nerf-hugs: Improved neural radiance fields in non-static scenes using heuristics-guided segmentation. CVPR (2024) 2, 4, 12, 13
7. Chen, X., Zhang, Q., Li, X., Chen, Y., Feng, Y., Wang, X., Wang, J.: Hallucinated Neural Radiance Fields in the Wild. In: CVPR (2022) 3
8. Chen, Z., Funkhouser, T., Hedman, P., Tagliasacchi, A.: Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In: CVPR (2023) 2
9. Chetverikov, D., Svirko, D., Stepanov, D., Krsek, P.: The trimmed iterative closest point algorithm. In: ICPR. vol. 3 (2002) 6
10. Dahmani, H., Bennehar, M., Piasco, N., Roldao, L., Tsishkou, D.: SWAG: Splatting in the Wild images with Appearance-conditioned Gaussians (2024) 2, 4, 11
11. Goli, L., Reading, C., Sellán, S., Jacobson, A., Tagliasacchi, A.: Bayes' Rays: Uncertainty quantification in neural radiance fields. In: CVPR (2024) 9
12. Hedlin, E., Sharma, G., Mahajan, S., He, X., Isack, H., Rhodin, A.K.H., Tagliasacchi, A., Yi, K.M.: Unsupervised Keypoints from Pretrained Diffusion Models. In: CVPR (2024) 4
13. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. NeurIPS (2020) 4
14. Ibrahim, A., El-kenawy, E.S.M.: Image segmentation methods based on superpixel techniques: A survey. Journal of Computer Science and Information Systems (2020) 7
15. Jain, A., Tancik, M., Abbeel, P.: Putting NeRF on a Diet: Semantically Consistent Few-Shot View Synthesis. In: ICCV (October 2021), https://arxiv.org/abs/2104.00677 2
16. Kajiya, J.T., Von Herzen, B.P.: Ray tracing volume densities. In: ACM TOG (1984) 2
17. Kerbl, B., Kopanas, G., Leimkuehler, T., Drettakis, G.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM TOG (Proc. SIGGRAPH) (2023) 1, 2, 5, 9, 10, 11
18. Kerbl, B., Meuleman, A., Kopanas, G., Wimmer, M., Lanvin, A., Drettakis, G.: A hierarchical 3d gaussian representation for real-time rendering of very large datasets. ACM TOG (Proc. SIGGRAPH) (2024) 5
19. Kulhanek, J., Peng, S., Kukelova, Z., Pollefeys, M., Sattler, T.: WildGaussians: 3D Gaussian Splatting In the Wild. arXiv (2024) 2, 11

20. Levy, A., Matthews, M., Sela, M., Wetzstein, G., Lagun, D.: MELON: NeRF with Unposed Images in SO(3). In: 3DV (2024) 2
21. Li, Z., Chen, J.: Superpixel segmentation using linear spectral clustering. In: CVPR (2015) 7
22. Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: BARF: Bundle-Adjusting Neural Radiance Fields. In: ICCV (October 2021), https://arxiv.org/abs/2104.06405 2
23. Liu, H.T.D., Williams, F., Jacobson, A., Fidler, S., Litany, O.: Learning smooth neural functions via lipschitz regularization. ACM TOG (2022) 8
24. Luo, G., Dunlap, L., Park, D.H., Holynski, A., Darrell, T.: Diffusion Hyperfeatures: Searching Through Time and Space for Semantic Correspondence. In: NeurIPS (2023) 4
25. Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J., Dosovitskiy, A., Duckworth, D.: NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In: CVPR (2021), https://arxiv.org/abs/2008.02268 2, 5, 11
26. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: CVPR (2021) 3, 4
27. Mildenhall, B., Srinivasan, P., Tancik, M., Barron, J., Ramamoorthi, R., Ng, R.: NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In: ECCV (2020) 1, 2
28. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. TOG (2022) 2
29. Müllner, D.: Modern hierarchical, agglomerative clustering algorithms. arXiv (2011) 7
30. Oquab, M., Darcet, T., Moutakanni, T., Vo, H.V., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: DINOv2: Learning Robust Visual Features without Supervision. TMLR (2023) 3, 4
31. Perreault, S., Hebert, P.: Median filtering in constant time. In: IEEE Transactions on Image Processing (2007) 9
32. Ramamoorthi, R., Hanrahan, P.: An efficient representation for irradiance environment maps. In: ACM TOG (2001) 5
33. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv (2022) 4
34. Rebain, D., Matthews, M., Yi, K.M., Lagun, D., Tagliasacchi, A.: LOLNerf: Learn from one look. In: CVPR (2022) 2
35. Rematas, K., Liu, A., Srinivasan, P.P., Barron, J.T., Tagliasacchi, A., Funkhouser, T., Ferrari, V.: Urban Radiance Fields. In: CVPR (2022) 3, 10, 11
36. Ren, W., Zhu, Z., Sun, B., Chen, J., Pollefeys, M., Peng, S.: NeRF On-the-go: Exploiting Uncertainty for Distractor-free NeRFs in the Wild. In: CVPR (2024) 2, 3, 4, 11, 12, 13
37. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. 2022 ieee. In: CVPR (2021) 4
38. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-Resolution Image Synthesis With Latent Diffusion Models. In: CVPR (2022) 4, 7
39. Sabour, S., Vora, S., Duckworth, D., Krasin, I., Fleet, D.J., Tagliasacchi, A.: RobustNeRF: Ignoring Distractors With Robust Losses. In: CVPR (2023) 2, 3, 5, 6, 7, 9, 10, 11, 12

40. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. NeurIPS (2022) 4
41. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016) 13
42. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR (2016) 4
43. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: Exploring photo collections in 3d. In: ACM TOG (2006) 2, 4
44. Song, Y., Ermon, S.: Generative modeling by estimating gradients of the data distribution. NeurIPS (2019) 4
45. Sun, C., Sun, M., Chen, H.T.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: CVPR (2022) 2
46. Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P.P., Barron, J.T., Kretzschmar, H.: Block-NeRF: Scalable Large Scene Neural View Synthesis. In: CVPR (2022) 3, 5
47. Tang, L., Jia, M., Wang, Q., Phoo, C.P., Hariharan, B.: Emergent correspondence from image diffusion. In: NeurIPS (2023) 4, 6, 7, 12
48. Wang, Y., Wang, J., Qi, Y.: WE-GS: An In-the-wild Efficient 3D Gaussian Representation for Unconstrained Photo Collections (2024) 2, 4, 11
49. Wang, Z., Wu, S., Xie, W., Chen, M., Prisacariu, V.A.: NeRF–: Neural Radiance Fields Without Known Camera Parameters (2021) 2
50. Wu, T.W., Zhong, F., Tagliasacchi, A., Cole, F., Oztireli, C.: D^2NeRF: Self-Supervised Decoupling of Dynamic and Static Objects from a Monocular Video. In: NeurIPS (2022) 2
51. Xu, J., Mei, Y., Patel, V.M.: Wild-gs: Real-time novel view synthesis from unconstrained photo collections (2024) 4
52. Yang, Y., Zhang, S., Huang, Z., Zhang, Y., Tan, M.: Cross-Ray Neural Radiance Fields for Novel-View Synthesis from Unconstrained Image Collections. In: ICCV (2023) 3
53. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: PlenOctrees for Real-Time Rendering of Neural Radiance Fields. In: ICCV (October 2021), https://arxiv.org/abs/2103.14024 2
54. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural Radiance Fields from One or Few Images. In: CVPR (2021), https://arxiv.org/abs/2012.02190 2
55. Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-splatting: Alias-free 3d gaussian splatting. In: CVPR (2024) 3
56. Zhang, D., Wang, C., Wang, W., Li, P., Qin, M., Wang, H.: Gaussian in the Wild: 3D Gaussian Splatting for Unconstrained Image Collections (2024) 4
57. Zhang, J., Herrmann, C., Hur, J., Cabrera, L.P., Jampani, V., Sun, D., Yang, M.H.: A Tale of Two Features: Stable Diffusion Complements DINO for Zero-Shot Semantic Correspondence. In: NeurIPS (2023) 4
58. Zwicker, M., Pfister, H., Van Baar, J., Gross, M.: Surface splatting. In: ACM TOG (2001) 2, 5