SWAG: Splatting in the Wild images with Appearance-conditioned Gaussians

Hiba Dahmani, Moussab Bennehar, Nathan Piasco, Luis Roldão, and Dzmitry Tsishkou

Noah's Ark, Huawei Paris Research Center, France

(a) (b) (c)

Fig. 1: Given *in-the-wild* captures (a), our model enables transient objects removal (b) and scene reconstruction with variable appearances (c).

Abstract. Implicit neural representation methods have shown impressive advancements in learning 3D scenes from unstructured *in-the-wild* photo collections but are still limited by the large computational cost of volumetric rendering. Recently, 3D Gaussian Splatting emerged as a much faster alternative with superior rendering quality and training efficiency, especially for small-scale and object-centric scenarios. Nevertheless, this technique suffers from poor performance on unstructured *in-the-wild* data. To tackle this, we extend over 3D Gaussian Splatting to handle unstructured image collections. We achieve this by modeling appearance to seize photometric variations in the rendered images. Additionally, we introduce a new mechanism to train transient Gaussians to handle the presence of scene occluders in an unsupervised manner. Experiments on diverse photo collection scenes and multi-pass acquisition of outdoor landmarks show the effectiveness of our method over prior works achieving state-of-the-art results with improved efficiency.

Keywords: 3D Gaussian Splatting · Unconstrained Photo Collection · Novel View Synthesis · Appearance Modeling · Real-time Rendering · Transient Object Removal

1 Introduction

Novel View Synthesis (NVS) and 3D scene reconstruction are long-standing, challenging tasks in the realms of computer vision and computer graphics. Over the past few years, Neural Radiance Fields (NeRF) [15] have demonstrated groundbreaking results in rendering photorealistic views from novel viewpoints using an implicit neural representation of the radiance and density fields in a scene. Although NeRFs are very effective in static scenes, their performance significantly degrades in dynamic scenarios (i.e. containing moving transient objects) or in the presence of changing conditions such as weather, exposure, and lighting (i.e. *in-the-wild* datasets). To tackle these challenges, NeRF-W [14] proposed an extension to NeRF enabling the reconstruction of outdoor landmarks from *in-the-wild* images. They achieve this by learning the changing appearance of the images through per-image embeddings. Furthermore, to handle the presence of occluders, the scene is decomposed into "static" and "transient" elements being modeled by separate radiance fields. These modifications significantly improved NeRF's performance when confronted with appearance changes and transient occluders. More recent works [3, 21] improve transient object handling by leveraging 2D visibility maps. Nonetheless, these methods still suffer from the inherent cost of volumetric rendering.

Recently, 3D Gaussian Splatting (3DGS) [8] attracted considerable interest thanks to its explicit representation resulting in faster training and rendering by leveraging GPU-based rasterization. 3DGS achieves fast training and matches visual quality obtained by the current SOTA NeRF methods [16]. This work represents the scene as a large number of anisotropic 3D Gaussians with color features and opacities. The Gaussians' set is then used in a differentiable splatting process to project the 3D Gaussians into a 2D image plane and blend them to get the rendered image. Similar to the original NeRF [15], 3DGS suffers from the same limitations regarding varying lighting conditions and the presence of occluders in the training images.

In this work, we present SWAG, the first *in-the-wild* extension for 3DGS. To achieve this, we propose to expand the capabilities of 3DGS and improve its robustness in these scenarios. To this aim, we propose to capture the varying appearance of each image using learned embeddings and leveraging a multilayer perceptron (MLP), significantly improving our model's capabilities. Second, we learn an image-dependent opacity variation to each Gaussian that enables a better handling of transient objects and a finer scene reconstruction as illustrated in Figure 1.

We conduct a variety of experiments on the Phototourism dataset [6] and NeRF-OSR [18] benchmark and show that not only do we improve 3DGS performance in these scenarios, but we also achieve SOTA rendering quality with significantly faster training and rendering speeds compared to previous works [3, 7, 14, 21]. To summarize, our main contributions are as follows:

- We introduce a new method for reconstructing scenes from unconstrained photo collections with varying appearances.

- We design a specific training scheme that captures occluders and enables transient object removal from a trained scene.
- We show that our method achieves SOTA performance in NVS *in-the-wild* scenarios.
- We demonstrate SWAG's abilities to generate new images with smooth visual transitions from learned appearances and to remove, in an unsupervised manner, transient objects from a captured scene.

2 Related Work

2.1 Neural Rendering *in-the-wild*

The abundance of *in-the-wild* unconstrained image data has encouraged adapting neural radiance fields to such cases. Early works like NeRF-W [14] have proposed a disentanglement between static and transient occluders using two appearance and transient per-image embeddings in addition to two radiance fields for each component of the scene (i.e. static and transient). Ha-NeRF [3] on the other hand uses a 2D image-dependent visibility map to eliminate occluders instead of using a decoupled radiance field since transient phenomena are only observed in individual 2D images. This simplification reduces the blurry artifacts that NeRF-W [14] suffered from in the attempt to reconstruct transient phenomena with a 3D transient field. Building upon previous methods, CR-NeRF [21] enhances their performance by exploiting the interaction information from multiple rays and fusing it into global information. Using a light-weight segmentation network, this method learns a visibility map without the supervision of ground truth segmentation masks to eliminate transient parts in 2D images. Another recent work, RefinedFields [7], leverages K-Planes and generative priors for *in-the-wild* scenarios. The learning is done through the alternation of two stages: a scene fitting to optimize the K-Planes [4] representation and a scene enriching phase that finetunes a pre-trained generative prior and infers a new K-Planes representation. As presented, Implicit-field representations have found diverse adaptations for *in-the-wild* scenarios. However, their time-consuming training and inference pose a challenge to achieving real-time rendering. This constrains their application in practical scenarios where fast rendering speed is crucial, especially in diverse interactive 3D applications.

2.2 Point-Based Rendering

Although NeRF demonstrated exceptional capabilities in generating photorealistic images, the demand for faster and more efficient rendering methods became increasingly evident. Point-based rendering has emerged as a prominent approach in computer graphics. It mainly revolves around the representation and rendering of the scene using discrete primitives. Starting from the most elementary form, Point Sample Rendering [5] authors proposed to sample a fixed-size formless set of points from a surface. Despite the developed strategies to adequately sample objects or surfaces, point sample rendering still suffers from

holes and discontinuity. To overcome the issue of 3D points sparsity, the following works used different geometric primitives like discs [2] and surfels [23] that have a larger extent compared to pixels. Differentiable point-based rendering is the groundwork of these methods since it enables end-to-end training of 3D primitives from images. Pulsar [10] introduced an efficient sphere-based differentiable renderer where the scene is represented by a set of spheres with learned positions, feature vectors, opacities, and radii. These parameters are optimized to minimize the photometric reconstruction loss using an efficient differentiable rendering. The rasterization is done using either a convolutional U-Net or a per-pixel one-by-one convolutional network applied to a 2D screen space feature map. The feature map is a blending function that combines each sphere channel information using weighting based on position, radius, and opacity. Recently 3DGS [8] used 3D Gaussians as primitives and introduced a fully differentiable fast tile-based rasterizer for Gaussian splats allowing real-time rendering while outperforming state-of-the-art visual quality. Despite the demonstrated capability of 3DGS, there is still room for refining this method to enhance the quality of rendered images. One observed issue with 3DGS [8] is the appearance of strong artifacts when changing the sampling rate. Additionally, the rendering quality declines significantly in low resolutions or far away from camera positions due to the aliasing caused by the pixel size compared to the screen size and the Nyquist frequency.

2.3 3DGS Rendering Improvement

To solve the aforementioned rendering issue, a multi-scale adaptation of 3DGS [20] has been introduced. This work explains the aliasing phenomena by the splitting of large amounts of Gaussians with a smaller extent than pixels in 3D regions featuring high-frequency details. To render at a certain scale, authors filter out too large or too small Gaussians. Additionally, small Gaussians are aggregated to form larger ones selected on bigger scales during training to prevent missing areas and low-frequency details. Motivated by Mip-NeRF's [1] rendering quality, Mip-Splatting [22] replaces the 2D dilation filter introduced in Surface splatting [23] using 2D Mip filters that replicate box filters to solve the aliasing issue and uses 3D smoothing filters with a maximal sampling rate obtained from training images. One other issue with 3DGS is its performance deterioration in scenes that may present variation in appearance. VastGaussian [12] introduced a decoupled appearance modeling using a pixel-wise appearance embedding and a CNN to predict a pixel-wise transformation for the rendered images. Hence these predicted transformations (multiplication, addition, and Gamma correction) work well for large scenes but they are insufficient to model more varying appearances that appear *in-the-wild* scenarios. Our work aims to extend *in-the-wild* 3DGS scene representation beyond closed-world setups.

3 Background

The scene in 3DGS [8] is represented by a set of 3D anisotropic Gaussians. Each Gaussian is parameterized by its centroid $\mathbf{x} \in \mathbb{R}^3$, scale $\mathbf{S} \in \mathbb{R}^3$, rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$, opacity $\alpha \in \mathbb{R}$ and color $\mathbf{c} \in \mathbb{R}^3$ encoded in spherical harmonic ($\mathbf{SH}$) coefficients. The 3D covariance matrix $\mathbf{\Sigma}$ of the 3D Gaussian is obtained using its rotation matrix $\mathbf{R}$ and scale $\mathbf{S}$:

$$\mathbf{\Sigma} = \mathbf{RSS}^T\mathbf{R}^T. \tag{1}$$

The 3D Gaussians are defined in world space following:

$$\mathbf{G}(\mathbf{y}) = e^{-\frac{1}{2}(\mathbf{y}-\mathbf{x})^T\mathbf{\Sigma}^{-1}(\mathbf{y}-\mathbf{x})}. \tag{2}$$

Given $\mathbf{J}$, the Jacobian of the affine projective transformation and $\mathbf{W}$, a viewing transformation, the covariance matrix $\mathbf{\Sigma}'$ in camera coordinates is as follows:

$$\mathbf{\Sigma}' = \mathbf{JW\Sigma W}^T\mathbf{J}^T. \tag{3}$$

These 3D Gaussians are projected to 2D splats and blended during a fast differentiable α-blending process to get 2D rendered images. Each pixel $\mathbf{y}$ color value $\mathbf{C}$ is calculated using N-ordered 2D splats using the formula:

$$\mathbf{C} = \sum_{i\in\mathbf{N}} \alpha_i'\mathbf{c}_i \prod_{j=1}^{i-1}(1-\alpha_j'), \tag{4}$$

α_i' is the final opacity of the Gaussian obtained by:

$$\alpha_i' = \alpha_i * e^{-\frac{1}{2}(\mathbf{y}'-\mathbf{x}')^T\mathbf{\Sigma}'^{-1}(\mathbf{y}'-\mathbf{x}')}. \tag{5}$$

where $\mathbf{x}'$ and $\mathbf{y}'$ are coordinates in the projected space.

Although 3D Gaussians can be arbitrarily initialized, a good prior is often required for optimal results. Typically, the 3D Gaussians centers are initialized using the sparse Structure from Motion (SfM) point cloud obtained from the set of images. To avoid leaving huge holes in the scene, their corresponding covariances are initialized to have isotropic Gaussians with initial radii equal to the mean distance to neighboring points. During training, the 3D Gaussian features are optimized such that the photometric difference between the rendered image $\mathbf{I_r}$ and the ground truth one $\mathbf{I_{gt}}$ is minimized with intervening controlling steps. To control the Gaussians' density and to address under- and over-distribution issues, authors use an adaptive densification step where Gaussians with large spatial gradients are split while transparent Gaussians that have very low opacity are pruned.

Although 3DGS [8] works well on object-centric and small scenes, it struggles with scenes presenting varying appearances and transient objects.

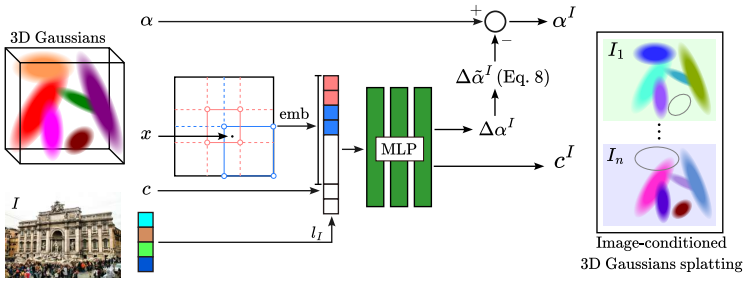

Fig. 2: SWAG model architecture – In addition to the typical Gaussians' features, we also optimize a Hash Grid encoding their centers $\mathbf{emb}(\mathbf{x})$, a per-image embedding vector l_I and an MLP. This MLP takes as inputs the Gaussians' colors c, the associated image embedding l_I, and their encoded centers $\mathbf{emb}(\mathbf{x})$ and outputs an image-dependent color $\mathbf{c^I}$ as well as an image-dependent opacity variation parameter $\Delta\alpha^I$. This parameter is set as the location variable of a concrete distribution which we sample to get the opacity variation $\Delta\tilde{\alpha}^\mathbf{I}$. Leveraging this opacity variation across diverse training images enables identifying and excluding transient Gaussians within the scene, as demonstrated by the grey Gaussians.

4 Method

In this section, we introduce SWAG, a novel 3DGS-based method for 3D scene reconstruction from *in-the-wild* photo collections. We propose to adapt the 3D Gaussians parameters to handle variable visual appearances and the presence of occluders typically found in such unconstrained image collections.

In Section 4.1, we explain how SWAG handles appearance variations using image-dependent embeddings injected into the Gaussian's colors. In Section 4.2, we explain how our proposal handles transient occluders by learning image-dependent Gaussians' opacities variations. Since our model's knowledge about the presence of occluders is embedded in the Gaussians opacities variation, it easily enables a clear disentanglement between static and transient elements of the scene. The overall architecture of our method is illustrated in Figure 2.

4.1 Appearance Variation Modeling

To adapt 3DGS to photometric variations, we associate, similarly to previous methods [14, 19], a trainable embedding vector l_I for each image. To inject the appearance information, one naive solution would be to calculate a global transformation per image to the 3D Gaussian colors or SH features. This strategy, however, might be insufficient since it cannot model photometric features at the local level in an image. To model local appearance variances in an image, one possible way would be to predict a local transformation or directly a specific color for each Gaussian on a per-image basis. However, such solution is intractable as the number of training parameters will grow according to the number of Gaussians times the number of images.

In SWAG, we propose a more effective solution where we use an MLP that takes as input a concatenation of the image embeddings and a positional encoding of the Gaussians' centers. By using a positional encoding conditioned on the center of the Gaussians, we are able to model local appearance variations that occur in the image while maintaining a tractable number of parameters to optimize. Initially, we considered estimating an affine color transformation similar to Urban Radiance Fields [17] to be applied to each Gaussian color. However, our initial results showed that affine color transformations alone cannot model all appearance changes. To solve this problem, we directly predict the Gaussian color from the MLP. We noticed that spatial embeddings coupled with an MLP produce a smoothing effect on the predicted colors. Hence, to render view-dependent specularities and high-frequency details and counterbalance this smoothing effect, we also add the Gaussian color obtained from the **SH** coefficients as input to the MLP.

Our final solution employs an MLP $\mathcal{F}_\theta$ that, for a given view, takes as input the color of the 3D Gaussian $\mathbf{c}$, the embedding vector of its center $\mathbf{emb}(\mathbf{x})$ and the image embedding l_I, and outputs the image-conditioned color $\mathbf{c^I}$. Accordingly, for a 3D Gaussian, the image-dependent color $\mathbf{c^I}$ is computed as follows:

$$\mathbf{c^I} = \mathcal{F}_\theta(\mathbf{c}, \mathbf{emb}(\mathbf{x}), l_I). \tag{6}$$

4.2 Transient Gaussians Modeling

Our initial results of applying 3DGS to *in-the-wild* scenarios showed that Gaussians representing transient objects produce blurred and elongated artifacts, due to their optimization across all images, despite transients appearing in only a few frames. Taking this observation into consideration, we introduce a learnable image-dependent opacity variation term $\Delta\tilde{\alpha}^{\mathbf{I}}$ to each Gaussian.

On the one hand, this opacity variation parameter allows Gaussians to reconstruct occluders present in some images. On the other hand, it enables those same Gaussians to be transparent (i.e. not involved in the image rendering process in other images) where these occluders are absent. During training, we want to encourage this variation parameter to either fully mask (i.e. in the case of occluders) or fully maintain (i.e. in the case of the static background) the 3D Gaussians' opacities. For this purpose, we sample $\Delta\tilde{\alpha}^{\mathbf{I}}$ using a Binary Concrete random variable [13]. This distribution is a continuous approximation of a Bernoulli distribution concentrating most of the mass on the boundaries of the interval $[0, 1]$. We return an additional output, $\Delta\alpha^{\mathbf{I}}$, of the previously introduced MLP $\mathcal{F}_\theta$ as the location parameter to the concrete function:

$$(\mathbf{c^I}, \Delta\alpha^{\mathbf{I}}) = \mathcal{F}_\theta(\mathbf{c}, \mathbf{emb}(\mathbf{x}), l_I), \tag{7}$$

$$\Delta\tilde{\alpha}^{\mathbf{I}} = \text{sigmoid}\left[\frac{1}{\mathbf{T}}\left(log\left(|\Delta\alpha^{\mathbf{I}}|\right) + log(\mathbf{U}) - log(1-\mathbf{U})\right)\right], \tag{8}$$

$$\mathbf{U} \sim \text{Uniform}(0, 1), \tag{9}$$

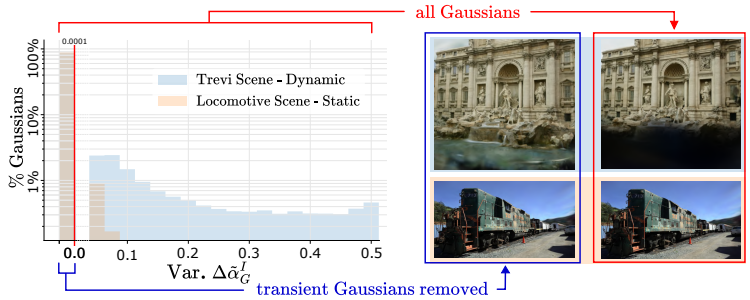

Fig. 3: Variance histogram analysis of the Gaussians' opacity variation $\Delta\tilde{\alpha}^{\mathbf{I}}$ w.r.t training images for a dynamic (i.e. containing transient objects) scene (top, *Trevi Fountain* from Phototourism [6]); and a static scene (bottom, *Locomotive* from Tanks & Temples [9]). The left column views are rendered using only static Gaussians (i.e having $\mathbf{Var}\left[\Delta\tilde{\alpha}^{\mathbf{I}}\right] = 0$), whereas right column views are rendered using all Gaussians.

where $\mathbf{T}$ is the temperature hyper-parameter. We do not use any annealing on the temperature parameter, rather we set $\mathbf{T} = 0.1$ to force, from the very beginning of the training, the opacity variation to match a binary distribution. During the evaluation, we fix $\mathbf{U}$ at 0.5. The final image-dependent 3D Gaussian opacity is formulated as:

$$\alpha^{\mathbf{I}} = \max\left(\alpha - \Delta\tilde{\alpha}^{\mathbf{I}}, 0\right). \tag{10}$$

With our formulation of the opacity, we can naturally disentangle between the transient and the static parts of the scene by defining transient Gaussians as the ones having non-zero variance of their associated parameter $\Delta\tilde{\alpha}^{\mathbf{I}}$ over training images: $\mathbf{Var}\left[\Delta\tilde{\alpha}^{\mathbf{I}}\right] \neq 0$.

Figure 3 shows the histogram of $\mathbf{Var}\left[\Delta\tilde{\alpha}^{\mathbf{I}}\right]$ on two scenes: one without transient content (*Locomotive*) and one with occluders (*Trevi*). We also show two rendered images of each scene: the right image using all the Gaussians and the left one using only Gaussians not detected as transient. The histogram shows that for a static scene, we detected less than 4% of the Gaussians as transient and around 96% of the Gaussians having $\mathbf{Var}\left[\Delta\tilde{\alpha}^{\mathbf{I}}\right] = 0$. Removing the 4% of transient Gaussians does not affect the rendered image quality. For *Trevi Fountain*, our model was able to represent transient objects using less than 20% of the Gaussians while maintaining the majority of the Gaussians to reconstruct static parts of the scene. Comparing the two rendered images shows that the transient Gaussians correspond to the mass of tourists present near the fountain, while removing the transient Gaussians resulted in an occluders-free rendering.

5 Experiments

5.1 Implementation details

Similar to 3DGS, we minimize the $\mathcal{L}_1$ loss combined with D-SSIM term to optimize the Gaussians parameters, the MLP $\mathcal{F}_\theta$ weights, the Hash Encoder pa-

Table 1: Quantitative results – Results on three real-world scenes from Phototourism [6] and efficiency comparison among SoTA methods for NVS *in-the-wild*. Given the significant gap between previous baselines [3,7,14,21] and 3DGS [8], we marginalize on the different GPU configurations.

	Bradenburg Gate			Sacre Coeur			Trevi Fountain			Mean Efficiency	
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Train time (h)	FPS
NeRF [15]	18.90	0.815	0.231	15.60	0.715	0.291	16.14	0.600	0.366	-	-
NeRF-W [14]	24.17	0.890	0.167	19.20	0.807	0.191	18.97	0.698	0.265	400†	<1†
Ha-NeRF [3]	24.04	0.877	0.139	20.02	0.801	0.171	20.18	0.690	0.222	452†	0.20†
CR-NeRF [21]	26.53	0.900	0.106	22.07	0.823	0.152	21.48	0.711	0.206	420†	0.25†
RefinedFields [7]	26.64	0.886	-	22.26	0.817	-	23.42	0.737	-	150†	<1†
3DGS [8]	19.99	0.889	0.180	17.57	0.831	0.219	18.47	0.761	0.234	0.50*	181*
SWAG (ours)	26.33	0.929	0.139	21.16	0.860	0.185	23.10	0.815	0.208	0.83*	15.29*

† Results reported on [7]. * Results computed using a high-tier GPU.

rameters, and the per-image embedding vectors. We fix the size of the latent embedding vectors to $n = 24$. For the Hash encoding parameters, we set the hash table size $T_h = 2^{19}$, the finest resolution $N_{max} = 2048$, the number of levels $L = 12$ and the coarsest resolution $N_{min} = 16$. The MLP we employ has 3 layers of 64 hidden units. Additional details on hyperparameters are provided in the supplemetary material.

5.2 Datasets

Similar to SoTA *in-the-wild* reconstruction methods, we evaluate SWAG's NVS capabilities using the Phototourism dataset [6] and report our results on three main touristic monuments: *Brandenburg Gate*, *Sacre Coeur*, and *Trevi Fountain*. We also evaluate our method on the benchmark introduced in NeRF-OSR [18] for outdoor scene relighting tasks. We report our performance using NeRF-MS's [11] data split on 4 sites: *stjohann (St. Johann, Saarbrücken)*, *lwp (Landwehrplatz, Saarbrücken)*, *st (Staatstheater, Saarbrücken)*, and *europa (Galerie Europa, Saarbrücken)* and the original NeRF-OSR [18]'s data split on 3 sites: *stjohann*, *lwp* and *lk2 (Ludwigskirche, Saarbrücken)*. More details about the datasets can be found in supplementary materials.

5.3 Evaluation

We provide visual comparisons with rendered images and report quantitative results based on common rendering metrics from the literature: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). Since only the embeddings associated with the training images are optimized during training, we follow NeRF-W's [14] evaluation approach: we optimize an embedding on the left half of each test image and report metrics on the right half.

Table 2: Quantitative results – Results on five sites from NeRF-OSR [18] outdoor scenes benchmark.

	stjohann			lwp			st			europa		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [15]	14.89	0.432	0.639	11.51	0.468	0.574	17.20	0.514	0.502	17.49	0.551	0.503
NeRF-W [14]	21.23	0.667	0.426	19.61	0.616	0.445	20.31	0.607	0.438	20.00	0.699	0.340
Ha-NeRF [3]	17.19	0.686	0.331	20.03	0.685	0.365	17.30	0.538	0.483	17.79	0.632	0.421
NeRF-MS [11]	22.84	0.793	0.235	21.90	0.719	0.336	20.68	0.630	0.402	21.03	0.721	0.294
3DGS [8]	16.77	0.741	0.268	11.76	0.609	0.414	17.16	0.629	0.406	20.18	0.782	0.252
SWAG (ours)	23.91	0.864	0.172	22.07	0.783	0.303	22.29	0.713	0.364	23.74	0.845	0.242

(a) Reported on NeRF-MS split [11]

	lk2		st		lwp	
	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑
NeRF-OSR [18]	19.86	0.626	15.83	0.556	17.38	0.576
3DGS [8]	15.91	0.687	12.53	0.585	13.72	0.659
SWAG (ours)	19.59	0.756	18.73	0.721	18.61	0.753

(b) Reported on original benchmark split from NeRF-OSR [18]

5.4 Results

Table 1 shows quantitative results on three Phototourism scenes. Unsurprisingly, optimizing 3DGS on *in-the-wild* photo collections showed poor results due to the lack of image appearance dependency. The 3D Gaussians colors converge to a mean color which makes the method unable to model visual changes caused by weather, lightning conditions and different camera specifications. The presence of transient objects also resulted in the persistence of artifacts. SWAG improves 3DGS performance and outperforms the baselines on SSIM across all datasets. In particular, our proposal improves 3DGS's PSNR by an average margin of 5.01 dB. SWAG outperforms NeRF-W across all datasets in terms of LPIPS, PSNR, and SSIM and shows competitive performance with CR-NeRF and RefinedFields despite the priors these two methods use and requiring significantly longer times to train.

Table 2 shows quantitative results on NeRF-OSR [18] benchmark. SWAG improves 3DGS performance by an average PSNR margin of 5 dB and outperformed all the other baselines regardless of the train/test split choices.

Figure 4 and Figure 5 show qualitative results of 3DGS and SWAG on Phototourism scenes and four of NeRF-OSR sites. In some parts of the scene, 3DGS presents artifacts and floaters either in under-observed part of the scene or in areas occluded by transient objects. Adding to that, the rendered color is shifted from the ground truth color as shown in Figure 4. SWAG produces visual appearances close to the ground truth thanks to our per-image conditioning. Compared to previous implicit methods [3,7,14,21], our proposal achieves one order of magnitude faster training and guarantees real-time rendering as shown in Table 1.

5.5 Controllable appearance

Modeling the image appearance with an embedding vector enabled us to learn a latent space that we can utilize to change the lighting and appearance of any

Fig. 4: Qualitative experimental results on three real-world scenes from Phototourism [6].s

viewpoint at inference time. On top of the training image embeddings shown in Figure 6, we can sample vectors from the appearance space using interpolation between the learned embeddings. These interpolations are smooth and natural as shown in Figure 7. We encourage the readers to explore the supplementary videos and appreciate more the naturalness of these interpolations.

5.6 Transient Objects Removal

As explained in Section 4.2, our method is able to localize 3D Gaussians that represent transient objects and, consequently, is able to manipulate the scenes omitting these objects. It should be noted that SWAG is thoughtfully designed to model transient objects without allowing the Gaussians to vary their opacity

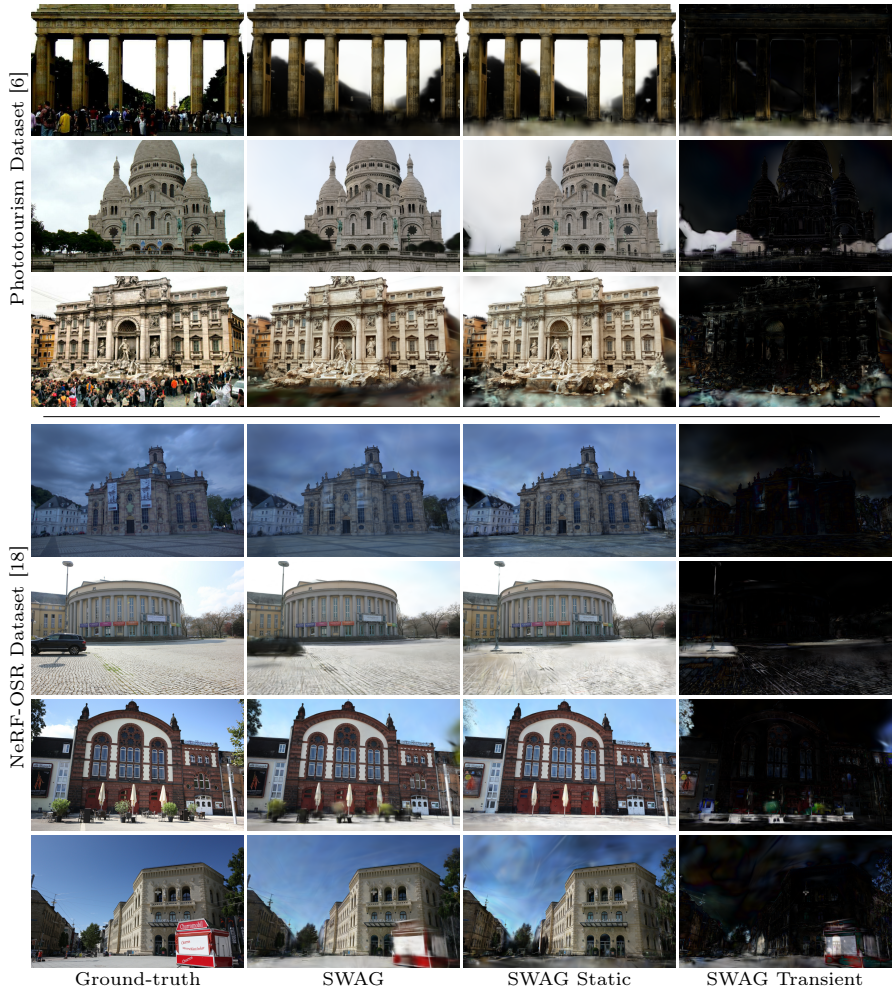

Fig. 5: Qualitative experimental results on three real-world scenes from Phototourism [6] and four NeRF-OSR [18] scenes. We demonstrate the capability of SWAG to disentangle between static and transient parts of the scene.

to compensate for appearance variation. Indeed, according to our experiments, it was observed that less than 20% (average on trained scenes) of the Gaussians capture transient objects. Consequently, SWAG achieves a better reconstruction of the static scene parts without having blurry elongated Gaussians that arise in view-dependent appearances with 3DGS. The capabilities of our model of disentangling static and dynamic Gaussians are clearly highlighted in Figure 5.

Fig. 6: Appearance transfer – SWAG is able to render a scene at any viewpoint with the visual appearance of any training image thanks to the learned appearance embeddings.

Fig. 7: Appearance interpolation – Interpolations between the appearance embeddings of two training images (left, right).

5.7 Ablations

In this section, we compare two variants of our method to analyze the contribution of each component of SWAG:

- **SWAG-A**, a variation wherein the transient detection part of our model is removed,
- **SWAG-T**, a variation wherein the image appearance color dependency is omitted.

We compare the results of our model variation on Phototourism scenes and we report the results in Table 3. SWAG-A enabled appearance modeling and

Table 3: Quantitative results – Results of two ablations of SWAG : SWAG-A and SWAG-T on three real-world scenes from Phototourism [6].

	Bradenburg Gate			Sacre Coeur			Trevi Fountain		
	PSNR ↑	SSIM↑	LPIPS ↓	PSNR ↑	SSIM↑	LPIPS ↓	PSNR ↑	SSIM↑	LPIPS ↓
SWAG	26.33	0.929	0.139	21.16	0.860	0.185	23.10	0.815	0.208
SWAG-A	26.44	0.928	0.141	21.02	0.855	0.183	22.35	0.801	0.223
SWAG-T	24.46	0.914	0.160	19.06	0.836	0.205	20.13	0.769	0.244

Fig. 8: Ablation study – Visual comparison of 3DGS, SWAG, and the two variants used in our ablation study: SWAG-T (no appearance) and SWAG-A (no transient).

closer colors to ground truth compared to 3DGS where colors shift away and converge to an average color for all training images. Transient objects' presence still causes floaters in the scene alike 3DGS as shown in Figure 8. In contrast, SWAG-T enables the elimination of these floaters and produces an occluders-free scene while suffering from color alterations compared to ground truth like 3DGS. Combining these two variations enables SWAG to model varying appearances with the ability to reduce scene floaters.

5.8 Discussion

SWAG leverages an MLP, a hash grid encoder, and image embedding to adapt 3DGS to *in-the-wild* scenarios. While this empowered 3DGS's reconstruction capabilities in these conditions as it was demonstrated in our experiments, it almost doubles the training time and has 10 times longer inference time per frame as shown Table 1 compared to 3DGS. Nonetheless, our method still requires significantly less training time than all previous *in-the-wild* baselines and achieves interactive rendering speed. Although SWAG succeeded in removing occluders using transient Gaussians, some parts of the scene where transient objects' presence is more frequent (eg. clouds in the sky and tourists on the roads) can suffer from being detected as transient. Eliminating all the transient Gaussians can leave some holes in these parts of the scene as seen in Figure 5 for the Trevi Fountain scene where some parts of the fountain contain black spots. One way to solve this is to remove Gaussians based on an adaptive threshold λ rather than considering as static only Gaussian with zero variance: $\mathbf{Var}\left[\Delta\tilde{\alpha}^{\mathbf{I}}\right] < \lambda$, but it requires manual tuning of this threshold λ.

6 Conclusion

In this paper, we presented SWAG, a method designed for tailoring 3DGS representations to *in-the-wild* scenarios. SWAG incorporates appearance modeling in the Gaussians' colors and employs an adaptive opacity modulation to handle the presence of transient objects. Extensive experiments demonstrate that SWAG achieves state-of-the-art results on two challenging benchmarks while exhibiting training times orders of magnitude faster than *in-the-wild* NVS baselines while enabling real-time rendering. As a first step in conditioning 3DGS for *in-the-wild* scene representations, this work suggests potential future research direction, such as extending SWAG to dynamic scenes.

References

1. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. ICCV (2021)
2. Botsch, M., Hornung, A., Zwicker, M., Kobbelt, L.: High-quality surface splatting on today's gpus. In: Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics (2005)
3. Chen, X., Zhang, Q., Li, X., Chen, Y., Feng, Y., Wang, X., Wang, J.: Hallucinated neural radiance fields in the wild. In: CVPR. pp. 12943–12952 (2022)
4. Fridovich-Keil, S., Meanti, G., Warburg, F.R., Recht, B., Kanazawa, A.: K-planes: Explicit radiance fields in space, time, and appearance. In: CVPR (2023)
5. Grossman, J., Sc, H., Dally, W.: Point sample rendering (1999)
6. Jin, Y., Mishkin, D., Mishchuk, A., Matas, J., Fua, P., Yi, K.M., Trulls, E.: Image matching across wide baselines: From paper to practice. International Journal of Computer Vision p. 517–547 (2020)
7. Kassab, K., Schnepf, A., Franceschi, J.Y., Caraffa, L., Mary, J., Gouet-Brunet, V.: Refinedfields: Radiance fields refinement for unconstrained scenes (2024)
8. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (2023)
9. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples. ACM Transactions on Graphics pp. 1–13 (2017)
10. Lassner, C.: Fast differentiable raycasting for neural rendering using sphere-based representations. CoRR (2020)
11. Li, P., Wang, S., Yang, C., Liu, B., Qiu, W., Wang, H.: Nerf-ms: Neural radiance fields with multi-sequence. In: ICCV. pp. 18591–18600 (2023)
12. Lin, J., Li, Z., Tang, X., Liu, J., Liu, S., Liu, J., Lu, Y., Wu, X., Xu, S., Yan, Y., Yang, W.: Vastgaussian: Vast 3d gaussians for large scene reconstruction (2024)
13. Maddison, C.J., Mnih, A., Teh, Y.W.: The concrete distribution: A continuous relaxation of discrete random variables. In: International Conference on Learning Representations (2017)
14. Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J.T., Dosovitskiy, A., Duckworth, D.: NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In: CVPR (2021)
15. 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)
16. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. CoRR (2022)
17. Rematas, K., Liu, A., Srinivasan, P.P., Barron, J.T., Tagliasacchi, A., Funkhouser, T., Ferrari, V.: Urban radiance fields. CVPR (2022)
18. Rudnev, V., Elgharib, M., Smith, W., Liu, L., Golyanik, V., Theobalt, C.: Nerf for outdoor scene relighting. In: ECCV (2022)
19. Turki, H., Ramanan, D., Satyanarayanan, M.: Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In: CVPR. pp. 12922–12931 (2022)
20. Yan, Z., Low, W.F., Chen, Y., Lee, G.H.: Multi-scale 3d gaussian splatting for anti-aliased rendering (2023)
21. 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)

22. Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-splatting: Alias-free 3d gaussian splatting (2023)
23. Zwicker, M., Pfister, H., van Baar, J., Gross, M.: Surface splatting. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques. p. 371–378 (2001)