Mipmap-GS: Let Gaussians Deform with Scale-specific Mipmap for Anti-aliasing Rendering

Jiameng Li[2,3*†]   Yue Shi[1,3*†‡]   Jiezhang Cao[3,4]   Bingbing Ni[1]   Wenjun Zhang[1]

Kai Zhang[5]   Luc Van Gool[3,6]

[1]Shanghai Jiaotong University   [2]KU Leuven   [3]ETH Zürich
[4]Harvard University   [5]Nanjing University   [6]INSAIT, Sofia University

Abstract

3D Gaussian Splatting (3DGS) has attracted great attention in novel view synthesis because of its superior rendering efficiency and high fidelity. However, the trained Gaussians suffer from severe zooming degradation due to non-adjustable representation derived from single-scale training. Though some methods attempt to tackle this problem via post-processing techniques such as selective rendering or filtering techniques towards primitives, the scale-specific information is not involved in Gaussians. In this paper, we propose a unified optimization method to make Gaussians adaptive for arbitrary scales by self-adjusting the primitive properties (e.g., color, shape and size) and distribution (e.g., position). Inspired by the mipmap technique, we design pseudo ground-truth for the target scale and propose a scale-consistency guidance loss to inject scale information into 3D Gaussians. Our method is a plug-in module, applicable for any 3DGS models to solve the zoom-in and zoom-out aliasing. Extensive experiments demonstrate the effectiveness of our method. Notably, our method outperforms 3DGS in PSNR by an average of 9.25 dB for zoom-in and 10.40 dB for zoom-out on NeRF Synthetic dataset. Our project website: https://github.com/renaissanceee/Mipmap-GS.

1. Introduction

With the development of implicit methods [2, 3, 41, 65], Novel View Synthesis (NVS) gains significant attention in virtual reality [11, 17, 80], augmented reality [49, 51, 60], and 3D generation [13, 18, 35, 36, 38, 43, 47, 56, 70, 72]. Recently, 3D Gaussian Splatting [29] shows state-of-the-art (SOTA) rendering quality with fast speed, due to the primitive-based representation and rasterization technique. However, 3DGS suffers from severe aliasing or blurriness
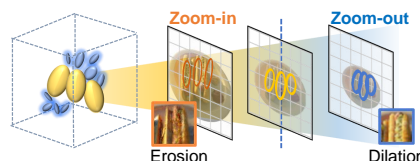

Figure 1. Aliasing at different observation distances.

when zooming in or out, leading to compromised user experience in interaction applications.

Typically trained on single-scale images, 3DGS is sensitive to sampling rates that deviate from training set [32, 66, 74]. This characteristic bears similarity to Neural Radiance Field (NeRF) [39], as discussed in Mip-NeRF [2]. When changing the observation distance, image resolution or camera focal length, Fig. 1 illustrates the zoom-out dilation and zoom-in erosion in 3DGS, **both of which called aliasing** in our paper following [74]. The quality degradation at unseen scales comes from mismatches between learned Gaussian fields and out-of-distribution sampling rates. Specifically, the 3D Gaussians are projected into the image plane and go through a 2D dilation filter before the rasterizer to smooth the shrinkage bias [74]. However, non-deformable Gaussians and constant dilation factor (which is set to be 0.3) do not seamlessly align with varying settings in Fig. 2. Zoom-in corresponds to finer pixel grids, where insufficiently smoothed projections in 3DGS pipeline [29] perform needle-like spikes appeared as artifacts or empty areas represented as structure missing. Conversely, zoom-out encounters excessive brightness and thickness, since too many Gaussians contribute to a single pixel.

To address the challenges of zooming in and out, most existing methods have unsatisfactory performance due to the lack of deformable representation. To accommodate the decreased sampling rate in zoom-out, [66] proposes a selective rendering method, which aggregates many small primitives for few large ones locally. However, the selection process inevitably introduces manually selected parameters and only applies in zoom-out. Inspires by [81], [74] introduces a 3D filter for frequency constraint, which works to smooth the zoom-in shrink, and replaces the original 2D di-
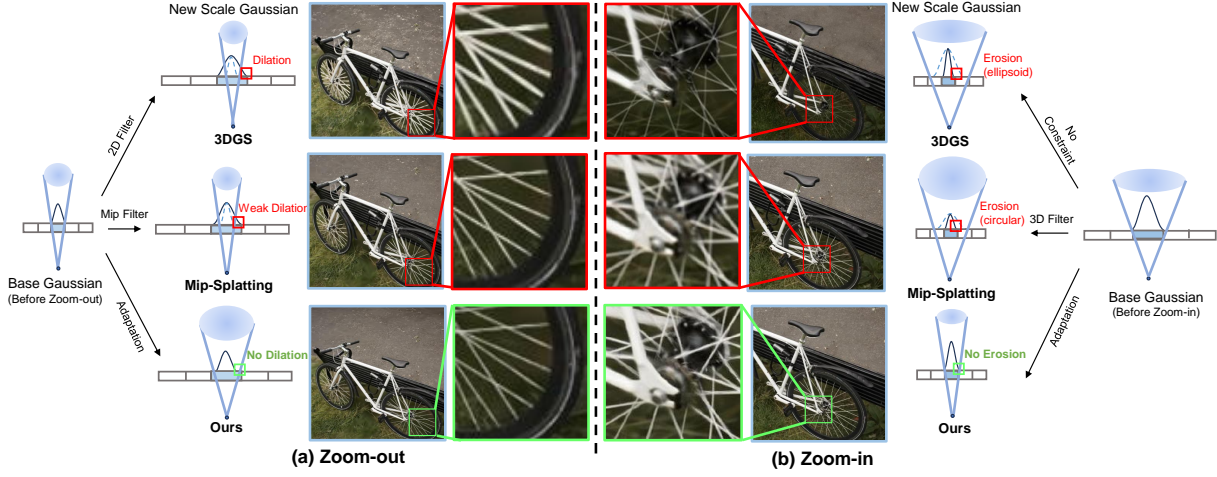
Figure 2. **Comparison of 3DGS, Mip-Splatting and ours in zoom-out and zoom-in scenarios.** 3DGS (top) lacks scale sensitivity and exhibits strong degradation. Mip-Splatting (middle) introduces a 2D Mip filter and 3D smooth filter to regulate scales heuristically while leaving 3D Gaussians unaware of the varying footprint in pixel space. Ours (bottom) adopts scale-adaptive Gaussians for explicit pixel coverage optimization. The projection in previous pixel coverage is illustrated in the dashed line.

lation filter with a Mip filter inspired by [81]. Although applicable for arbitrary scale rendering, the Mip filter in [74] lacks scale information and the 3D smooth filter suppresses the high-frequency components, which inevitably sacrifices some finer details [32]. **Thus, there is an urgent need to explore methods that adapt the source signal itself to essentially accommodate varying sampling rates.**

In this paper, we propose a novel scale-adaptive optimization method for 3DGS, addressing challenges of zoom-in and zoom-out rendering. Specifically, we construct mipmap-like pseudo ground-truth (pseudo-GT) to provide scale-specific information in a self-supervised way. A mipmap represents a signal (e.g. image or texture map) at a set of discrete downsampling scales and selects the appropriate scale for anti-aliasing rendering [2]. We change the once-created and precomputed mipmap to scale-specific mipmap-like pseudo-GT on novel views during test time. To deform Gaussians with the mipmap, we introduce a scale-aware guidance loss. By self-supervised optimization, our method fits for arbitrary-scale rendering. Our contributions are summarized as follows:

- We propose a scale-adaptive optimization approach with scale-aware guidance loss. Different from existing methods, our deformable Gaussians are self-adjusted for zoom-in and -out.
- Our method designs mipmap-like pseudo-GT on observation scales for test-time adaptation, which provides prior information for zooming deformation.
- Rather than training from scratch, our approach makes use of base Gaussians and converges within 1K iterations, offering computational advantages.
- Serving as a plug-in module, our method is applicable in any 3DGS models to address the zoom-in and -out issues, and improves the PSNR of 3DGS over 9dB.

2. Related Work

2.1. Novel View Synthesis

Novel View Synthesis aims to generate images from new viewpoints based on a set of source-view images. We will introduce two main branches according to different scene representations, *i.e.* explicit and implicit methods.

Explicit methods describe the scene in discrete spatial structures like point cloud [45, 59, 62], mesh [24, 52] and voxel grid [34]. Yet, explicit methods require large memory usage and perform not well on low-resolution scenes [20, 55]. As a pioneer of implicit representations, NeRF [39] utilizes Multi-Layer Perceptron (MLP) to store scene information implicitly, which maps view-related spatial positions and view directions to color and volume density values. To speed up the training or rendering of NeRF, [10, 19, 41, 65] represent the scene by grid-based feature, point-based feature and hash encoding. And [9, 28, 53, 72] explore generalizable NeRF. In multi-scale rendering works [2, 3, 23, 25, 42], Mip-NeRF [2] replaces pixel-wise ray sampling with conical frustums. Despite the significant performance of NeRFs, the volumetric ray marching necessitates high computation costs.

2.2. 3D Gaussian Splatting

Combining explicit and implicit methods for scene representation [11], 3DGS [29] learns a Gaussian field that holds discrete geometry centers and continuously optimized attributes. Due to its supervisor fidelity and speed, 3DGS [29] appears to be an appealing alternative to NeRF [39]. Unlike the MLP architecture and ray marching techniques [39], 3DGS employs primitive-based splatting and fast rasterizer, enjoying high parallelism in GPU [29, 71]. The learnable parameters of 3D Gaussians are directly optimized, including geometry information of mean and covariance matrix,

and color information including opacity and spherical harmonic (SH) coefficients. The growth and pruning of Gaussians are conducted by adaptive density control considering the accumulated gradients and opacity contributions.

3DGS has been applied in many tasks, such as sparseview reconstruction [8, 12, 14, 61, 67], 3D generation [57, 64, 76] and dynamic scenes [1, 21, 46, 68]. To improve the rendering quality, Pixel-GS [79] proposes a scaled gradient field optimized by pixel coverage weighted loss, while [77] introduces frequency spectrum as guidance. [22] put forward generalized exponential splatting to sharpen high-frequency details, and [26] simplifies 3D Gaussians to 2D Gaussians for surface reconstruction. Scaffold-GS [37] leverages anchor points and MLP to arrange view-adaptive Gaussians and reduce storage size. [5, 15, 69] revise the density control strategy during optimization. [75] propose Gaussian-Opacity-Field (GOF) for surface alignment and mesh extraction with a novel densification method. [63] leverages 3DGS to synthesize data factory under different focal lengths.

2.3. Zoom-out and Zoom-in

Despite the superior NVS capacity of NeRF and 3DGS, they face degradation when rendering at unseen scales, *i.e.* zoom-out and zoom-in.

When zooming out, the disparity between decreased sampling rate and high-frequency components causes aliasing. Mipmap and Level-of-Detail (LOD) techniques are leveraged in traditional computer graphics for anti-aliasing rendering. In NeRF-related works, [2–4] render conical frustums instead of rays and apply pre-filtering to the input positional encoding to resist aliasing. For 3DGS, [48, 66] adopt LoD representation for efficient and anti-aliasing rendering, while inevitably introducing hyperparameter tuning in level selection. Analytic-Splatting [32] treats each pixel as an area instead of separate points to deal with pixel footprint changes.

Zoom-in entails moving from a global view to local details. Previous NeRF-based methods leverage supersampling strategy with depth regularization [58], and postprocessing steps [27, 33] to address this issue. [50, 73] takes SR prior from images and video respectively for HR rendering. Mip-Splatting [74] introduces a 3D smooth filter to regulate high-frequency components of each primitive. [74] re-calculates and updates the scale of the 3D filter during training. In the testing stage, each primitive is subjected to the integrated 3D filter before being projected to screen space. Unlike previous methods, we propose scale-specific supervision to mimic mipmap. Rather than storing a precomputed data structure and interpolating between levels, we derive mipmap-like pseudo-GT on the fly and use it to refine the 3D model further.

3. Preliminaries

3.1. 3D Gaussian Splatting

3DGS [29, 81] represents scenes as a set of 3D Gaussian primitives $\{G_k \mid k = 1, \ldots, K\}$, with geometry center $\mu_k \in \mathbb{R}^{3 \times 1}$ and covariance matrix $\Sigma_k \in \mathbb{R}^{3 \times 3}$. Each 3D Gaussian is parameterized as:

$$G_k(x) = e^{-\frac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k)}, \tag{1}$$

where Σ_k is further expressed by orthogonal rotation matrix $R \in \mathbb{R}^{3 \times 3}$ and diagonal scale matrix $S \in \mathbb{R}^{3 \times 3}$: $\Sigma = RSS^\top R^\top$. During rendering, primitives overlapped at pixel x are sorted according to depth order $\{1, \ldots, K\}$. Then the view-dependent pixel color $c(x)$ modeled by spherical harmonics is calculated by :

$$c(x) = \sum_{k=1}^{K} c_k \alpha_k \prod_{i=1}^{k-1} (1 - \alpha_i), \tag{2}$$

where α_k is calculated by evaluating a projected 2D Gaussian G_k^{2D} multiplied with a learned opacity [71], and finally attends to the pixel color $c(x)$ with the primitive color c_k. Each primitive properties are optimized across all the training views coupled with density control intermittently.

Shrinkage Bias and 2D Dilation. In practice, 3D Gaussians are truncated to calculate pixel color with shrinkage bias [74]. The extreme shrink is expressed as δ impulse, which becomes needle-like spikes in renderings. In addition, the tiny Gaussians (smaller than 1 pixel) with trivial contributions are hard to optimize. Therefore, a 2D dilation operation [29] is designed before rasterization to undo the shrinkage bias:

$$G_k^{2D}(x) = e^{-\frac{1}{2}(x-\mu_k)^\top (\Sigma_k^{2D}+sI)^{-1}(x-\mu_k)}, \tag{3}$$

where s equals 0.3 to simulate a low-pass filter, achieving faithful rendering in Fig. 3a.
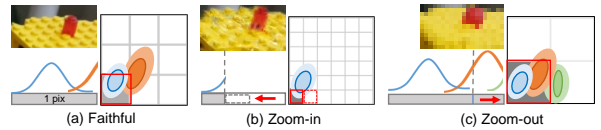

Figure 3. **A toy 3DGS model at multi-scale rendering.** The splatted Gaussians (dark) go through 2D dilation (light) for faithful rendering (a). However, the varying shrinkage bias and constant dilation cause zoom-in spikes (c) and zoom-out thickness (d).

3.2. Motivation

Despite the promising NVS results at basic scale (consistent with the training scale), zoom-in and -out rendering will bring two challenges for out-of-distribution generalization:

Varying Sampling Rates. 3DGS leverages a group of Gaussian ellipsoids for scene representation, akin to a Gaussian mixture model. During pixel shading, pixel grids are
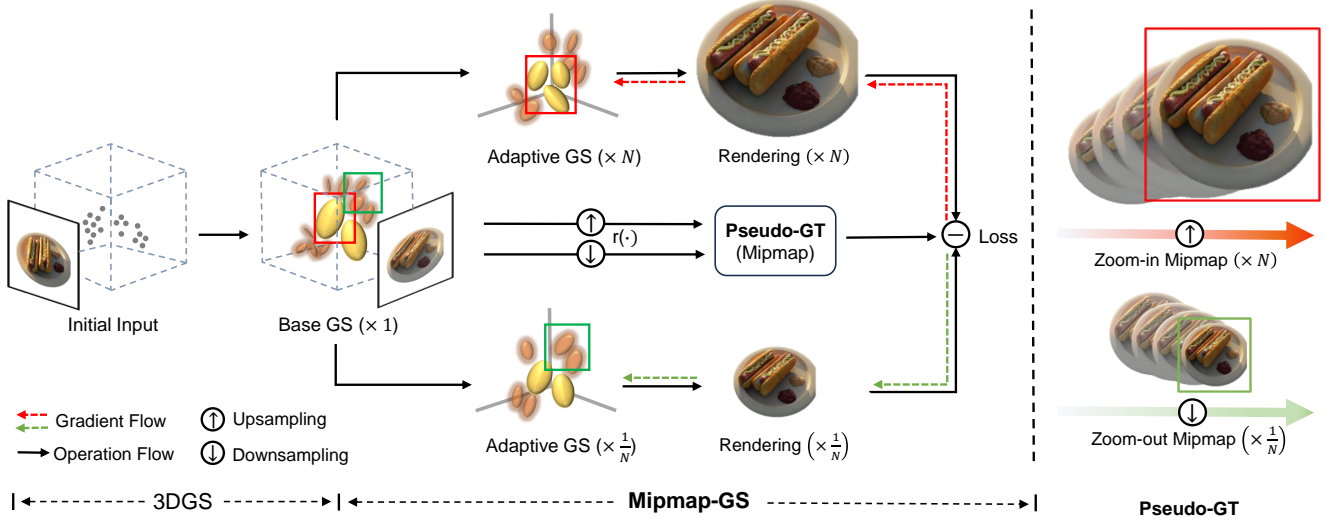
Figure 4. **Pipeline**. We adjust single-scale trained base Gaussians for arbitrary zoom factors supervised by mipmap-like pseudo-GT. The core component is the mipmap proposal module, which constructs scale-specific mipmap by upsampling or downsampling novel views.

allocated according to the current sampling rate. Then, Gaussians overlapped in one pixel are accumulated with shrinkage bias [74]. When the sampling rate changes, the varying shrinkage bias leads to confusion during accumulation. In Fig. 3, we simply illustrate different sampling rates and the shrunk Gaussian signals in 1D. The learned Gaussian field samples two signals at basic scale (Fig. 3a), while a zoom-in pixel (Fig. 3b) only captures a blue signal, and a zoom-out pixel (Fig. 3c) accumulates the orange signal for excessive contribution.

Constant Dilation. The pixel coverage during shading is decided jointly by projected Gaussians and dilation operation. The 2D dilation operation in 3DGS [29] is designed to amplify tiny Gaussians (marked as blue) smaller than one pixel (marked as grey) in Fig. 3a. However, the constant dilation doesn't fit for decreased pixel coverage in zoom-in, causing thin structures and empty areas with no splats (Fig. 3b). Conversely, maintaining the same dilation for zoom-out leads to excessively wide coverage and thick structures, which also undermines the rendering efficiency. In Fig. 3c, the green Gaussian was not meant to fall into the grey pixel, but due to inherited dilation, it gets included for pixel shading. The intuitive solution of super-sampling is sub-optimal due to the computational burden.

To mitigate the misalignment, [74] conducts both 3D and 2D smoothing using two smoothing filters, while [54] introduces a scale-adaptive 2D filter. These filters partially revise the scale of Gaussians to fit multi-scale rendering, yet the overall distribution doesn't actually change. [48, 66] arrange Gaussians hierarchically by LoD architecture, while the level selection is not accurate for pixel coverage and performs sensitive to hyper-parameters. Therefore, it's impossible to fit multi-scale rendering with fixed Gaussians. We propose Mipmap-GS to provide deformable Gaussians optimized via scale-specific mipmap.

4. Proposed Method

To solve the mismatched pixel coverage and Gaussians at new scales, we deform Gaussians interactively to fit for varying zoom factors. First, we construct mipmap-like pseudo-GT at the observation scale, see Sec. 4.1. Then, we introduce scale-adaptive Gaussians in Sec. 4.2, which are deformed from base Gaussians. Finally, we describe optimization details in Sec. 4.3, including scale-aware guidance loss and active pruning strategy.

4.1. Mipmap-like Pseudo-GT

Instead of conducting time-consuming multi-scale training, our method optimizes scale-adaptive Gaussians using mipmap-like pseudo-GT. Considering the NVS capacity of 3DGS, we first splat the set of base Gaussians into novel views at the basic scale ($\times 1$), which faces no degradation. Then, we construct new scale pseudo-GT with mipmap resizing function $r(x)$. For zoom-in adaptation, novel-view renderings are upsampled to $\times N$ using super-resolution methods like SwinIR [31]. Similarly, to create a scale-specific mipmap for zoom-out, the rendered images are downsampled to $\times 1/N$ of the original resolution. Compared with the zoom-in process which needs to generate details for extended pixels, the downsampling step involves less textural deterioration to mimic fading away. Therefore, we simply adopt bilinear interpolation as $r(x)$ to generate the LR mipmap. Note that mipmap is originally defined as a pre-calculated image sequence with progressively lower resolutions. Here, we use mipmap to refer to pseudo-GT at both lower and higher resolutions.

4.2. Scale-Adaptive 3D Gaussians

As introduced in Sec. 3.1, 3DGS [29] projects Gaussian ellipsoids into image space as a set of α-blended 2D Gaussians to accumulate the pixel color. The 3D shrinkage
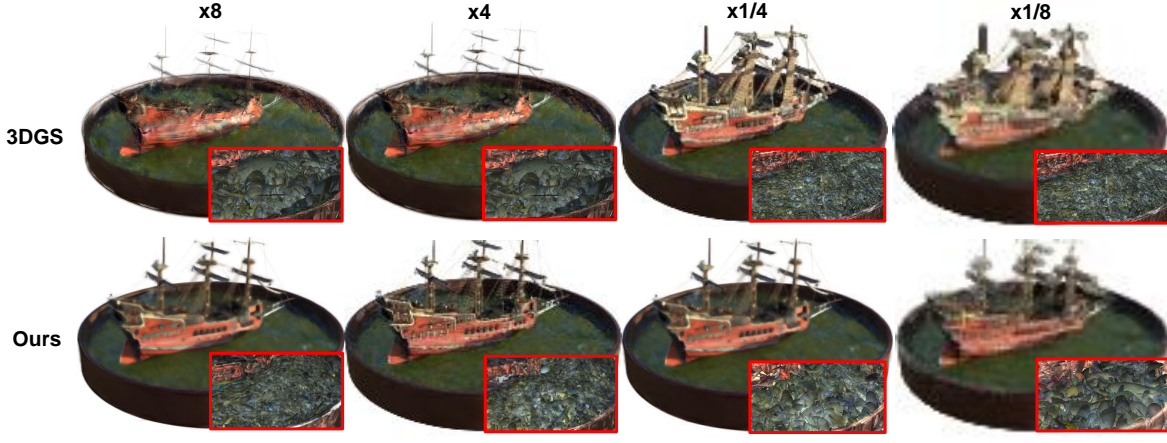
Figure 5. **Visualization of Gaussians.** 3DGS [29] uses a fixed set of Gaussians for multi-scale rendering, while our adaptive adjustment provides deformable Gaussians towards various zoom factors.

bias and 2D dilation restoration cooperate to fit the training scale. However, taking place in different spaces, these two operations work independently while not seamlessly aligned. As illustrated in Sec. 3.2, the existing space gap causes two challenges at multi-scale rendering, since dramatically varied sampling rates exceed the control range of the constant dilation, leading to aliasing results in Fig. 5.

To bridge the gap, we create the scale-specific mipmap $r(\hat{x})$ from base Gaussians $\mathbf{G}$. Then, we deform $\mathbf{G}$ into optimized Gaussians $\mathbf{G}^{\text{opt}}$ by photometric loss between the rendered image x and scale-specific mipmap $r(\hat{x})$:

$$\mathbf{G}^{\text{opt}} = \mathbf{G} - \beta\nabla L(x, r(\hat{x})), \qquad (4)$$

where β is the learning rate. Notably, our method solves the aliasing essentially since the scale prior directly informs of the new sampling rate to address two challenges in Sec. 3.2, instead of introducing filters to ease the issues caused by constant dilation. The scale-adaptive Gaussians leverage deformable attributes and distribution to fit any scale rendering. As a test-time adaptation method, our method keeps the scale consistency between 3D primitives and 2D pixel coverage for wide-coverage inference contexts.

4.3. Optimization

Illustrated in Alg. 1, we first generate the novel view $\hat{x}$ by base Gaussians $\mathbf{G}$. Then, $\hat{x}$ is upsampled or downsampled into $r(\hat{x})$ via mipmap function, serving as the pseudo-GT. We define the scale-aware guidance loss in adaptation as:

$$L(x, r(\hat{x})) = ||x - r(\hat{x})||^2. \qquad (5)$$

Here we use ℓ_2 loss and discuss other losses in Sec. 5.3. The deformation from $\mathbf{G}$ to $\mathbf{G}^{\text{opt}}$ is finished within 1K iterations.

Except for performance improvement, we also realize compact scene representation. In Alg. 1, the optimized primitive number K^{opt} of $\mathbf{G}^{\text{opt}}$ is smaller than the previous number K. The density control in 3DGS [29] stops halfway through the optimization process, leaving many low opacity Gaussians with trivial contribution [16, 30, 37, 40]. In contrast, we keep active pruning during the whole process for

a more compact representation, see more in supplementary materials. The storage size reduction by our active pruning is more obvious in zoom-out situations where the decreased spatial resolution requires fewer Gaussians to represent. What's more, our adjustment needs merely 3% iterations compared with training from scratch. Bringing neither interference towards the original pipeline nor new hyper-parameters, our method is compatible with subsequent works like Scaffold-GS [37], shown in Tab. 1.

Algorithm 1: Adaptive optimization of our method

Input: base Gaussians $\{\mathbf{G}_k \mid k = 1, \ldots, K\}$,
viewpoints $\{V_j \mid j = 1, \ldots, J\}$, scale N,
iteration S

Output: optimized Gaussians
$\{\mathbf{G}_k^{\text{opt}} \mid k = 1, \ldots, K^{\text{opt}}\}$

for $i = 0, ..., S$ **do**
    select $v_j \in V_j$ randomly
    render $\{\mathbf{G}_k\}$ to basic scale image $\hat{x}^j$, see Eqn.
    (2)
    construct mipmap $r(\hat{x}^j)$ by $\times N$ upsampling
    (zoom-in) or $\times \frac{1}{N}$ downsampling (zoom-out)
    render $\{\mathbf{G}_k^i\}$ to new scale image x^j, see Eqn.
    (2)
    optimize $\{\mathbf{G}_k^i\} = \arg\min_{\{\mathbf{G}_k^i\}} L(x^j, r(\hat{x}^j))$
    if $i \bmod 100 == 0$ **then**
       | densify or prune
    end
end
return $\{\mathbf{G}_k^{opt}\} = \{\mathbf{G}_k^{S+1}\}$

---

5. Experiments

5.1. Experimental Setup

Datasets. We conduct experiments on the NeRF Synthetic dataset [39] and the Mip-NeRF 360 dataset [3], measuring the effect of our continuous optimization approach. NeRF Synthetic dataset [39] has 8 synthetic objects without background. We adopt the common split of 100/100/200 for

Method	×1/2			×1/4			×1/8		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
NeRF [39]	0.962	32.43	0.041	0.964	30.29	0.044	0.951	26.70	0.067
Mip-NeRF [2]	0.970	33.31	0.031	0.969	30.91	0.036	0.961	27.97	0.052
Instant-NGP [41]	0.969	33.00	0.033	0.964	29.84	0.046	0.947	26.33	0.075
3DGS [29]	0.951	27.14	0.031	0.875	21.39	0.067	0.763	17.59	0.127
3DGS+EWA [29, 81]	0.971	31.66	0.024	0.959	27.82	0.033	0.940	24.62	0.047
Scaffold-GS [37]	0.953	27.48	0.030	0.886	21.83	0.061	0.781	18.03	0.116
Pixel-GS [79]	0.947	27.96	0.032	0.866	22.67	0.070	0.748	19.09	0.133
Octree-GS [48]	0.950	27.30	0.061	0.888	21.97	0.063	0.787	18.15	0.114
Analytic-Splatting [32]	0.977	34.21	0.019	0.977	31.49	0.021	0.969	28.42	0.031
Mip-Splatting [74]	0.977	34.00	0.019	0.978	31.85	0.019	0.973	28.67	0.026
Scaffold-Ours	0.971	32.71	0.026	0.975	32.46	0.023	0.976	30.81	0.028
3DGS-Ours	0.977	34.18	0.019	0.978	32.51	0.021	0.976	30.64	0.026

(a) Zoom-out.

Method	×2			×4			×8		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
NeRF [39]	0.921	27.54	0.100	0.881	25.56	0.170	-	-	-
Mip-NeRF [2]	0.944	29.36	0.057	0.876	25.47	0.159	0.832	23.47	0.207
NeRF-SR [58]	0.946	29.77	0.045	0.921	28.07	0.071	-	-	-
3DGS [29]	0.907	23.38	0.068	0.832	19.93	0.128	0.824	18.52	0.153
Scaffold-GS [37]	0.770	21.14	0.097	0.800	17.35	0.163	0.807	16.01	0.177
Pixel-GS [79]	0.897	24.95	0.070	0.823	21.26	0.133	0.819	19.69	0.161
Mip-Splatting [74]	0.960	30.08	0.051	0.917	27.12	0.105	0.886	25.71	0.136
Scaffold-Ours	0.968	30.79	0.043	0.936	28.20	0.078	0.911	26.59	0.153
3DGS-Ours	0.964	31.23	0.041	0.927	28.29	0.081	0.897	26.43	0.119

(b) Zoom-in.

Table 1. **Comparisons of zoom-out and zoom-in on NeRF Synthetic dataset** [39].

training/validation/test. The more challenging Mip-NeRF 360 dataset [3] consists of 9 real-world unbounded scenes. We follow the split factor 8 and report the average metrics over all scenes.

Evaluation Metrics. The rendering quality at each viewpoint is evaluated by Structural Similarity Index Measure (SSIM), Peak Signal-to-Noise Ratio (PSNR) and LPIPS [78]. Higher SSIM and PSNR values and lower LPIPS scores indicate better-rendering results.

Baselines. We take 3DGS [29], Scaffold-GS [37] and Pixel-GS [79] as baseline models. In addition, Mip-Splatting [74], Analytic-Splatting [32] and Octree-GS [48] are treated as SOTA anti-aliasing methods based on 3DGS. We also test the performance of GOF [75] and 3DGS + EWA [81]. GOF [75] inherits the anti-aliasing functions from [74], and 3DGS + EWA [81] replaces the original dilation with the EWA filter [81], introduced in [74]. Following the setting in their paper, [29, 32, 37, 74, 75, 79] are trained for 30K iterations, and [48] for 40K iterations.

Implementation Details. All the experiments are conducted on a single A100 GPU. The settings of density control, learning rate schedule and hyper-parameters are consistent with the original paper. After single-scale training, we regard rendering the set of Gaussians directly to different resolutions as the baseline setting following [74]. We

also test Scaffold-GS [37] with our adaptation to illustrate the plugin effects of our method.
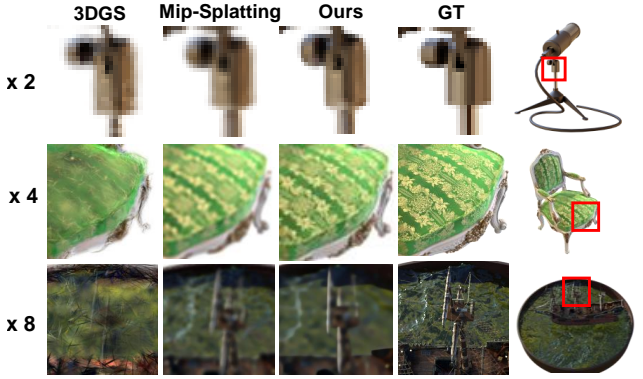

Figure 6. **Comparison of zoom-in results on NeRF Synthetic dataset** [39].

5.2. Comparisons with State-of-the-Art

Zoom-out. Following settings introduced by [74], baseline models are trained on full-resolution (×1) images and rendered at different resolutions (×1/2, ×1/4, ×1/8) to simulate the zoom-out effect. We derive renderings at the same resolution with respect to training, then downsample them into zoom-out pseudo-GT. Since downsampling from HR images involves minimal loss of information, this process often preserves most essential details of the original scene

Method	×1/2			×1/4			×1/8		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
Mip-NeRF 360 [3]	0.864	29.19	0.136	0.912	30.45	0.077	0.931	30.86	0.058
Instant-NGP [41]	0.712	25.23	0.251	0.809	26.84	0.142	0.877	28.42	0.092
zip-NeRF [4]	0.892	30.00	0.099	0.933	31.57	0.056	0.954	32.52	0.037
3DGS [29]	0.774	26.66	0.291	0.721	22.87	0.241	0.763	25.36	0.250
Octree-GS [48]	0.793	27.13	0.261	0.805	26.44	0.199	0.784	23.92	0.182
Analytic-Splatting [32]	0.713	25.70	0.350	0.791	27.04	0.216	0.880	29.44	0.106
Mip-Splatting [74]	0.812	27.44	0.233	0.799	27.16	0.209	0.744	27.03	0.326
GOF [75]	0.803	27.17	0.251	0.825	27.61	0.185	0.838	27.21	0.155
3DGS-Ours	0.854	28.57	0.156	0.828	28.34	0.199	0.854	28.57	0.154

(a) Zoom-out.

Method	×2			×4			×8		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
Mip-NeRF 360 [3]	0.727	25.18	0.260	0.670	24.16	0.370	0.706	24.10	0.428
Instant-NGP [41]	0.639	24.76	0.367	0.626	24.27	0.445	0.698	24.27	0.475
zip-NeRF [4]	0.696	23.27	0.257	0.565	20.87	0.421	0.559	20.27	0.494
3DGS [29]	0.740	23.49	0.243	0.619	20.69	0.394	0.603	19.21	0.477
3DGS+EWA [29, 81]	0.775	25.90	0.236	0.667	23.70	0.369	0.643	22.81	0.449
Mip-Splatting [74]	0.808	27.39	0.205	0.754	26.47	0.305	0.765	26.22	0.392
GOF [75]	0.802	27.05	0.206	0.744	26.17	0.313	0.758	25.90	0.411
3DGS-Ours	0.799	27.17	0.209	0.744	26.22	0.302	0.761	25.91	0.399

(b) Zoom-in.

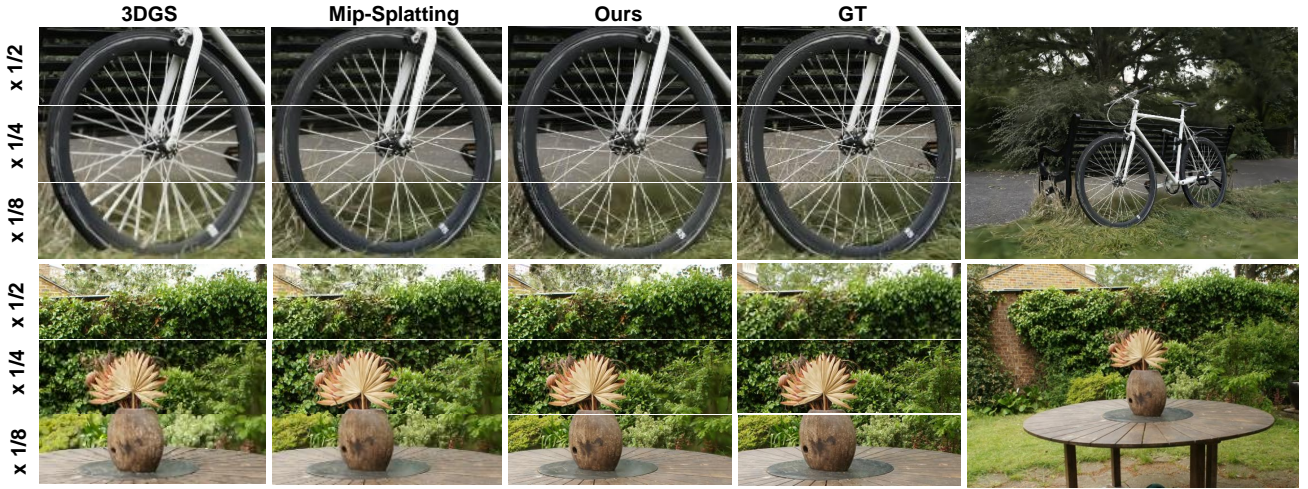Table 2. **Comparisons of zoom-out and zoom-in on Mip-NeRF 360 dataset** [3].

Figure 7. **Comparison of zoom-out results on Mip-NeRF 360 dataset [3].**

representation. Results in Tab. 1a and 2a show the improvement after our dynamic optimization. Although Scaffold-GS [37] and Pixel-GS [79] improve the full resolution rendering quality with a revised primitive growth strategy, the lack of multi-scale deformation compromises their robustness for unseen settings. And Fig. 7 shows zoom-out renderings, where the cropped wheel rendered by 3DGS [29] is dilated with thick spokes, and filters in Mip-Splatting [74] only mitigate the dilation to some extent. Our adaptation achieves the best visual effects and improves PSNR by 5dB compared with 3DGS [29]. Besides, our optimization for zoom-out converges faster than for zoom-in from Fig. 8a, and shows more remarkable improvement on NeRF Syn-

thetic dataset [39]. We ascribe the discrepancies in lifting effects to the quality of pseudo-GT. As [39] consists of isolated objects without background, the supervision from pseudo-GT is intensive and of high quality during the adaptation stage. In comparison, the pseudo-GT provided for Mip-NeRF 360 [3] has limited quality to cover the correct scene information and provides weaker guidance for adaptive optimization.

Zoom-in. The zoom-in baseline models are trained on ×1/8 downsampled images and rendered at ×2, ×4, ×8. As shown in Tab.2b and 1b, our method improves the zoom-in rendering quality of both 3DGS [29] and Schaffold-GS [37]. As one of the SOTA methods on super-resolution

[6, 7], SwinIR [31] provides HR pseudo-GT with abundant details for zoom-in deformation. In Fig. 6, zoom-in rendering of 3DGS exhibits needle-like aliasing due to intrinsic shrinkage bias, and Mip-Splatting [74] faces a high-frequency trade-off. Our method adjusts Gaussians into suitable pixel coverage for high-fidelity novel views.

5.3. Ablation Study

Training Iterations. Starting from trained Gaussians, our optimization needs much less training time to converge. As shown in Fig. 8a, our method converges within 0.5K iterations for zoom-out and 1K iterations for zoom-in, both on the order of seconds, see Tab. 3. In comparison, the filter update in Mip-Splatting [74] needs more time and is not compatible with 3DGS variations, e.g. [37].

Loss Functions. Originally, 3DGS takes SSIM and L1-loss weighted at 0.8 and 0.2 for training [29], while it is simplified to L2 loss in our optimization. We illustrate the metrics using different loss functions in Fig. 8b.
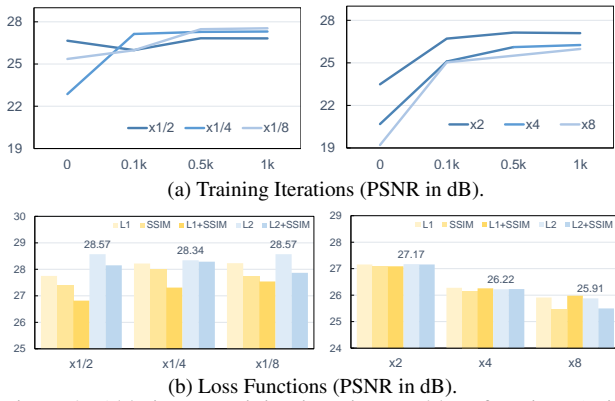

(a) Training Iterations (PSNR in dB).

(b) Loss Functions (PSNR in dB).

Figure 8. Ablation on training iterations and loss functions (Mip-NeRF 360 dataset [3]).

Method	Zoom-out	Zoom-in
3DGS [29]	2h30m	8m
3DGS-Ours	+40s	+75s
Scaffold-GS [37]	3h7m	12m
Mip-Splatting [74]	2h40m	8m

Table 3. Ablation on training time (Bicycle in [3]). Ours takes the average time over three scales.

PSNR (dB)	Ours+test	+train	+syn.	3DGS [29]
Zoom-out	30.84	29.80	30.24	22.60
Zoom-in	24.23	22.01	22.67	19.36

Table 4. Ablation on optimization views (Bicycle in [3]).

Optimization Views. As shown in Tab. 4, the performance drops when using training views (+train) because it focuses on known-view fitting instead of novel views. We choose test views to retain the original novel-view performance and further improve the zooming quality. As a test-time learning strategy, our method also works with synthetic views sam-

pled from the estimated camera trajectory. Here we report optimization results with 50 synthetic views (+syn.)

5.4. Further Study

Number of Primitives. We find that decreasing the number of splatted Gaussians in a certain range doesn't influence the performance. In Fig. 9, rendering merely high-opacity primitives of a trained 3DGS model [29] leads to comparable performance while up to 25% quantitative reductions. In Mip-Splatting [74], fewer primitives yield even slightly better results, which further proves the redundancy. Therfore, we keep active pruning throughout optimization.
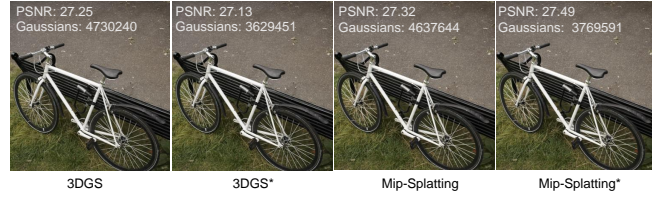

Figure 9. Comparison of selective rendering (Bicycle in [3]) at ×1/8. * selects Gaussians with opacity> 0.01.

View Consistency. Although we adopt SwinIR [31] to construct mipmap independently for each image without scene-level fine-tuning, the injected appearance is further refined within the 3D model for view-consistency, see Fig. 10.
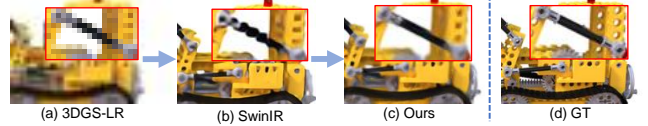

Figure 10. Comparison of view-consistency (Lego in [39]).

Application in Dynamic Scenes. Our method is applicable in dynamic 3DGS models for anti-aliasing rendering. Fig. 11 shows our plug-in effect in [68] with no disturbance on the deformation network.

Figure 11. Comparison of dynamic scenes (Jumpingjacks in [44]).

6. Conclusion

We present a unified method to fit scale-adaptive Gaussians for alias-free novel view synthesis. Given an unseen scale at test-time, the designed mipmap-like pseudo-GT allows adaptive adjustment of 3D Gaussians consistent with the zoom factor. Our method involves minimal modification towards the 3DGS pipeline and is applicable for any pre-trained Gaussian Splatting model as a plug-in module to mitigate aliasing for out-of-distribution generalization. Finally, our method converges on the order of seconds and helps to remove primitive redundancy, maintaining the fine scene representation without sacrificing real-time efficiency.

References

[1] Jeongmin Bae, Seoha Kim, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Per-gaussian embedding-based deformation for deformable 3d gaussian splatting. In *ECCV*, 2024. 3

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *CVPR*, pages 5855–5864, 2021. 1, 2, 3, 6

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, pages 5470–5479, 2022. 1, 2, 5, 6, 7, 8

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *ICCV*, 2023. 3, 7

[5] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in gaussian splatting. *arXiv preprint arXiv:2404.06109*, 2024. 3

[6] Jiezhang Cao, Qin Wang, Yongqin Xian, Yawei Li, Bingbing Ni, Zhiming Pi, Kai Zhang, Yulun Zhang, Radu Timofte, and Luc Van Gool. Ciaosr: Continuous implicit attention-in-attention network for arbitrary-scale image super-resolution. In *CVPR*, pages 1796–1807, 2023. 8

[7] Jiezhang Cao, Yue Shi, Kai Zhang, Yulun Zhang, Radu Timofte, and Luc Van Gool. Deep equilibrium diffusion restoration with parallel sampling. In *CVPR*, pages 2824–2834, 2024. 8

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

[9] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *ICCV*, pages 14124–14133, 2021. 2

[10] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, pages 333–350. Springer, 2022. 2

[11] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024. 1, 2

[12] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. In *ECCV*, 2024. 3

[13] Zilong Chen, Feng Wang, and Huaping Liu. Text-to-3d using gaussian splatting. *arXiv preprint arXiv:2309.16585*, 2023. 1

[14] Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-regularized optimization for 3d gaussian splatting in few-shot images. *arXiv preprint arXiv:2311.13398*, 2023. 3

[15] Lue Fan, Yuxue Yang, Minxing Li, Hongsheng Li, and Zhaoxiang Zhang. Trim 3d gaussian splatting for accurate geometry representation. *arXiv preprint arXiv:2406.07499*, 2024. 3

[16] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *arXiv preprint arXiv:2311.17245*, 2023. 5

[17] Ben Fei, Jingyi Xu, Rui Zhang, Qingyuan Zhou, Weidong Yang, and Ying He. 3d gaussian as a new vision era: A survey. *arXiv preprint arXiv:2402.07181*, 2024. 1

[18] Qijun Feng, Zhen Xing, Zuxuan Wu, and Yu-Gang Jiang. Fdgaussian: Fast gaussian splatting from single image via geometric-aware diffusion model. *arXiv preprint arXiv:2403.10242*, 2024. 1

[19] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, pages 5501–5510, 2022. 2

[20] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, pages 12479–12488, 2023. 2

[21] Zhiyang Guo, Wengang Zhou, Li Li, Min Wang, and Houqiang Li. Motion-aware 3d gaussian splatting for efficient dynamic scene reconstruction. *arXiv preprint arXiv:2403.11447*, 2024. 3

[22] Abdullah Hamdi, Luke Melas-Kyriazi, Guocheng Qian, Jinjie Mai, Ruoshi Liu, Carl Vondrick, Bernard Ghanem, and Andrea Vedaldi. Ges: Generalized exponential splatting for efficient radiance field rendering. In *CVPR*, 2024. 3

[23] Dongting Hu, Zhenkai Zhang, Tingbo Hou, Tongliang Liu, Huan Fu, and Mingming Gong. Multiscale representation for real-time anti-aliasing neural rendering. In *CVPR*, pages 17772–17783, 2023. 2

[24] Ronghang Hu, Nikhila Ravi, Alexander C Berg, and Deepak Pathak. Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In *ICCV*, pages 12528–12537, 2021. 2

[25] Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *ICCV*, pages 19774–19783, 2023. 2

[26] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM TOG*, pages 1–11, 2024. 3

[27] Xudong Huang, Wei Li, Jie Hu, Hanting Chen, and Yunhe Wang. Refsr-nerf: Towards high fidelity and super resolution view synthesis. In *CVPR*, pages 8244–8253, 2023. 3

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

[29] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. In *ACM TOG*, 2023. 1, 2, 3, 4, 5, 6, 7, 8

[30] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. *arXiv preprint arXiv:2311.13681*, 2023. 5

[31] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration us-

ing swin transformer. In *CVPR*, pages 1833–1844, 2021. 4, 8

[32] Zhihao Liang, Qi Zhang, Wenbo Hu, Ying Feng, Lei Zhu, and Kui Jia. Analytic-splatting: Anti-aliased 3d gaussian splatting via analytic integration. In *ECCV*, 2024. 1, 2, 3, 6, 7

[33] Chien-Yu Lin, Qichen Fu, Thomas Merth, Karren Yang, and Anurag Ranjan. Fastsr-nerf: Improving nerf efficiency on consumer devices with a simple super-resolution pipeline. In *WACV*, pages 6036–6045, 2024. 3

[34] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, pages 15651–15663, 2020. 2

[35] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *CVPR*, pages 9298–9309, 2023. 1

[36] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023. 1

[37] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *CVPR*, 2024. 3, 5, 6, 7, 8

[38] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *CVPR*, pages 8446–8455, 2023. 1

[39] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 5, 6, 7, 8

[40] Wieland Morgenstern, Florian Barthel, Anna Hilsmann, and Peter Eisert. Compact 3d scene representation via self-organizing gaussian grids. *arXiv preprint arXiv:2312.13299*, 2023. 5

[41] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. In *ACM TOG*, 2022. 1, 2, 6, 7

[42] Seungtae Nam, Daniel Rho, Jong Hwan Ko, and Eunbyung Park. Mip-grid: Anti-aliased grid representations for neural radiance fields. In *NeurIPS*, 2024. 2

[43] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *ICLR*, 2023. 1

[44] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, pages 10318–10327, 2021. 8

[45] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 2

[46] Zhiyin Qian, Shaofei Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. In *CVPR*, pages 5020–5030, 2024. 3

[47] Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023. 1

[48] Kerui Ren, Lihan Jiang, Tao Lu, Mulin Yu, Linning Xu, Zhangkai Ni, and Bo Dai. Octree-gs: Towards consistent real-time rendering with lod-structured 3d gaussians. *arXiv preprint arXiv:2403.17898*, 2024. 3, 4, 6, 7

[49] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *ECCV*. Springer, 2020. 1

[50] Yuan Shen, Duygu Ceylan, Paul Guerrero, Zexiang Xu, Niloy J Mitra, Shenlong Wang, and Anna Früstück. Supergaussian: Repurposing video models for 3d super resolution. In *ECCV*, 2024. 3

[51] Yujiao Shi, Hongdong Li, and Xin Yu. Self-supervised visibility learning for novel view synthesis. In *CVPR*, pages 9675–9684, 2021. 1

[52] Yue Shi, Bingbing Ni, Jinxian Liu, Dingyi Rong, Ye Qian, and Wenjun Zhang. Geometric granularity aware pixel-to-mesh. In *ICCV*, pages 13097–13106, 2021. 2

[53] Yue Shi, Dingyi Rong, Bingbing Ni, Chang Chen, and Wenjun Zhang. Garf: Geometry-aware generalized neural radiance field. *arXiv preprint arXiv:2212.02280*, 2022. 2

[54] Xiaowei Song, Jv Zheng, Shiran Yuan, Huan-ang Gao, Jingwei Zhao, Xiang He, Weihao Gu, and Hao Zhao. Sags: Scale-adaptive gaussian splatting for training-free anti-aliasing. *arXiv preprint arXiv:2403.19615*, 2024. 4

[55] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, pages 5459–5469, 2022. 2

[56] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. In *ICLR*, 2023. 1

[57] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *ECCV*, 2024. 3

[58] Chen Wang, Xian Wu, Yuan-Chen Guo, Song-Hai Zhang, Yu-Wing Tai, and Shi-Min Hu. Nerf-sr: High quality neural radiance fields using supersampling. In *ACM MM*, pages 6445–6454, 2022. 3, 6

[59] Jinglu Wang, Bo Sun, and Yan Lu. Mvpnet: Multi-view point regression networks for 3d object reconstruction from a single image. In *AAAI*, pages 8949–8956, 2019. 2

[60] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, pages 4690–4699, 2021. 1

[61] Christopher Wewer, Kevin Raj, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. *arXiv preprint arXiv:2403.16292*, 2024. 3

[62] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *CVPR*, pages 7467–7477, 2020. 2

[63] Renlong Wu, Zhilu Zhang, Yu Yang, and Wangmeng Zuo. Dual-camera smooth zoom on mobile phones. In *ECCV*, 2024. 3

[64] Dejia Xu, Ye Yuan, Morteza Mardani, Sifei Liu, Jiaming Song, Zhangyang Wang, and Arash Vahdat. Agg: Amortized generative 3d gaussians for single image to 3d. *arXiv preprint arXiv:2401.04099*, 2024. 3

[65] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *CVPR*, pages 5438–5448, 2022. 1, 2

[66] Zhiwen Yan, Weng Fei Low, Yu Chen, and Gim Hee Lee. Multi-scale 3d gaussian splatting for anti-aliased rendering. In *CVPR*, 2024. 1, 3, 4

[67] Chen Yang, Sikuang Li, Jiemin Fang, Ruofan Liang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian. Gaussianobject: Just taking four images to get a high-quality 3d object with gaussian splatting. *arXiv preprint arXiv:2402.10259*, 2024. 3

[68] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*, 2024. 3, 8

[69] Zongxin Ye, Wenyu Li, Sidun Liu, Peng Qiao, and Yong Dou. Absgs: Recovering fine details in 3d gaussian splatting. In *ACM MM*, 2024. 3

[70] Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529*, 2023. 1

[71] Wang Yifan, Shihao Wu, Hui Huang, Daniel Cohen-Or, and Olga Sorkine-Hornung. Patch-based progressive 3d point set upsampling. In *CVPR*, pages 5958–5967, 2019. 2, 3

[72] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, pages 4578–4587, 2021. 1, 2

[73] Xiqian Yu, Hanxin Zhu, Tianyu He, and Zhibo Chen. Gaussiansr: 3d gaussian super-resolution with 2d diffusion priors. *arXiv preprint arXiv:2406.10111*, 2024. 3

[74] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *CVPR*, 2024. 1, 2, 3, 4, 6, 7, 8

[75] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient and compact surface reconstruction in unbounded scenes. *arXiv preprint arXiv:2404.10772*, 2024. 3, 6, 7

[76] Bowen Zhang, Yiji Cheng, Jiaolong Yang, Chunyu Wang, Feng Zhao, Yansong Tang, Dong Chen, and Baining Guo. Gaussiancube: Structuring gaussian splatting using optimal transport for 3d generative modeling. *arXiv preprint arXiv:2403.19655*, 2024. 3

[77] Jiahui Zhang, Fangneng Zhan, Muyu Xu, Shijian Lu, and Eric Xing. Fregs: 3d gaussian splatting with progressive frequency regularization. In *CVPR*, pages 21424–21433, 2024. 3

[78] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 6

[79] Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-gs: Density control with pixel-aware gradient for 3d gaussian splatting. In *ECCV*, 2024. 3, 6, 7

[80] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, pages 286–301, 2016. 1

[81] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa volume splatting. In *VIS*, pages 29–538, 2001. 1, 2, 3, 6, 7