Spectral-GS: Taming 3D Gaussian Splatting with Spectral Entropy

Letian Huang   Jie Guo†   Jialin Dan   Ruoyu Fu   Shujie Wang   Yuanqi Li   Yanwen Guo
Nanjing University

{lthuang, danjialin, wangshujie}@smail.nju.edu.cn,   {guojie, fry, yuanqili, ywguo}@nju.edu.cn
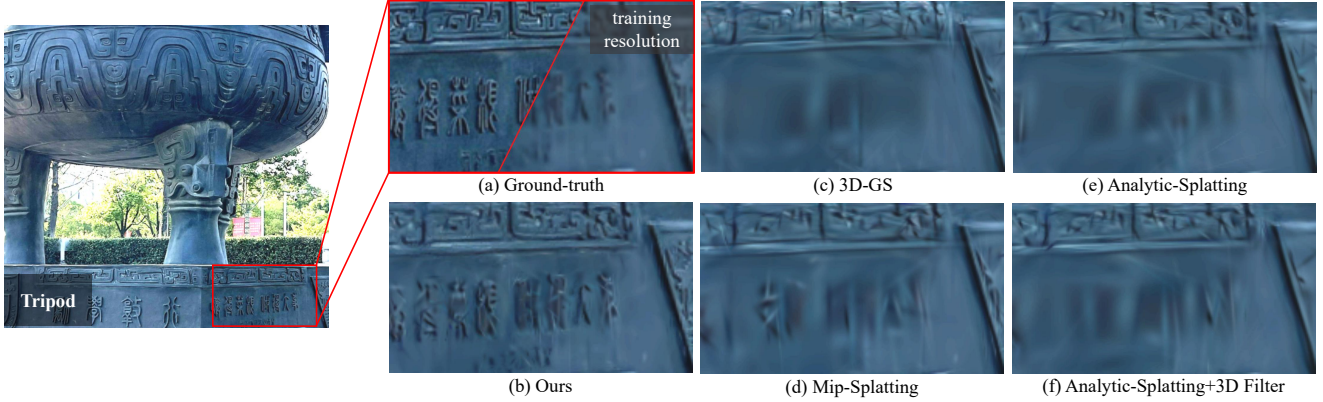
Figure 1. Despite its high efficiency in 3D reconstruction, 3D Gaussian Splatting (3D-GS) [19] suffers from needle-like artifacts **(c)** due to undersampling or view inconsistency. Recent works like Mip-Splatting [47] and Analytic-Splatting [23] try to eliminate these artifacts. Unfortunately, they still produce needles at high-frequency regions when zooming in, and will also cause over-blurriness **(d)(e)** since they lack shape awareness of 3D Gaussians. With spectral analysis of the variance matrix, we propose **Spectral-GS**, which imposes shape constraints on the 3D Gaussians and thus effectively addresses the above issues, generating high-quality photorealistic rendering **(b)**.

Abstract

Recently, 3D Gaussian Splatting (3D-GS) has achieved impressive results in novel view synthesis, demonstrating high fidelity and efficiency. However, it easily exhibits needle-like artifacts, especially when increasing the sampling rate. Mip-Splatting tries to remove these artifacts with a 3D smoothing filter for frequency constraints and a 2D Mip filter for approximated supersampling. Unfortunately, it tends to produce over-blurred results, and sometimes needle-like Gaussians still persist. Our spectral analysis of the covariance matrix during optimization and densification reveals that current 3D-GS lacks shape awareness, relying instead on spectral radius and view positional gradients to determine splitting. As a result, needle-like Gaussians with small positional gradients and low spectral entropy fail to split and overfit high-frequency details. Furthermore, both the filters used in 3D-GS and Mip-Splatting reduce the spectral entropy and increase the condition number during zooming in to synthesize novel view, causing view inconsistencies and more pronounced artifacts. Our Spectral-GS, based on spectral analysis, introduces 3D shape-aware splitting and 2D view-consistent filtering strategies, effec-

tively addressing these issues, enhancing 3D-GS's capability to represent high-frequency details without noticeable artifacts, and achieving high-quality photorealistic rendering.

1. Introduction

Reconstructing 3D scenes from 2D images and synthesizing novel views has been a critical task in computer vision and graphics. As the demand for real-time and photorealistic rendering continues to rise, 3D Gaussian Splatting (3D-GS) [19] has emerged as an efficient representation that can achieve high-speed rendering on the GPU. Unlike traditional implicit scene representations using MLPs [1, 2, 28, 48], 3D-GS adopts an explicit approach based on Gaussian functions. This bypasses the need for dense point sampling in volumetric rendering, thereby enabling real-time performance. However, 3D-GS [19] tends to optimize toward degraded needle-like Gaussians, resulting in unacceptable needle-like artifacts. Mip-Splatting [47] and Analytic-Splatting [23] attempt to address these issues by employing filtering or analytic integration to mitigate aliasing. Unfortunately, when representing high-frequency textures, these

---

† Corresponding author.

1

methods [23, 47] often lead to over-blurriness or still produce needle-like artifacts.

Needle-like Gaussians correspond to 3D Gaussians with low spectral entropy and high condition number. Existing 3D-GS [19] and its variants (including Mip-Splatting [47]) do not impose any restriction on the Gaussian's shape. The splitting strategies in 3D-GS lean towards generating degenerated elongated Gaussians, due to the lack of shape-awareness, relying instead on spectral radius and view positional gradients to guide the splitting. However, needle-like Gaussians with small positional gradients are hard to split. Even splitting, the condition number of the Gaussian remains ill-conditioned providing little help to alleviate needle-like artifacts. Worse still, both the EWA filter [51] in 3D-GS [19] and the 2D Mip filter in Mip-Splatting [47] reduce the spectral entropy and increase the condition number during zooming in to synthesize novel view. Due to the view-inconsistency in filtering, needle-like artifacts become more pronounced when zooming in or when the camera moves closer to the object.

Based on the above observations, we introduce *spectral analysis* to 3D Gaussian reconstruction. Specifically, we propose *3D shape-aware splitting* and *2D view-consistent filtering*, respectively, to address loss sensitivity and shape unawareness in densification, as well as view inconsistency in filtering. The splitting condition of our 3D shape-aware splitting is based on the spectral entropy of 3D Gaussians and our method ensures that the condition number after splitting reduces. The proposed 2D view-consistent filtering, combines a convolution that approximates super-sampling with a view-adaptive Gaussian blur that approximates interpolation to maintain the spectral entropy consistency. Our method Spectral-GS effectively enhances 3D-GS's capability to represent high-frequency details, mitigates needle-like artifacts, and achieves high-quality photorealistic rendering, as illustrated in Figure 1. Furthermore, our method is easily implemented, requiring only few changes to the original framework. In summary, we make the following contributions:

- We employ **spectral analysis** to examine 3D-GS, revealing issues such as loss sensitivity and shape unawareness in densification, as well as view inconsistency in filtering.
- We propose **3D shape-aware splitting** to regularize needle-like Gaussians, enhancing the high-frequency detail representation for 3D-GS and mitigating needle-like artifacts.
- We propose **2D view-consistent filtering** to resolve needle-like artifacts caused by view-inconsistency.

2. Related Work

Novel view synthesis is a longstanding challenge in computer vision and graphics. From traditional techniques [3, 5, 11, 13, 22, 36, 39], to neural network-based scene representations [10, 18, 21, 27, 31, 33, 38], various approaches have struggled to address the problem of synthesizing a new view from captured images.

2.1. Neural Radiance Fields

The Neural Radiance Field (NeRF) [28] stands out as a successful neural rendering method based on MLPs, primarily owing to its encoding of position and direction. This encoding allows for effective reconstruction of high-frequency information in scenes. Notable improvements on this encoding have been made by MipNeRF [1], NeRF-W [25], FreeNeRF [46], and Instant NGP [29]. These enhancements enable the handling of multi-resolution image inputs, multi-illumination with occlusion image inputs, sparse-view inputs, and achieve nearly real-time rendering capabilities, respectively. Barron *et al.* introduced MipNeRF360 [2] as an extension of MipNeRF [1] to address the issue of generating low-quality renderings for unbounded scenes in NeRF. Importantly, these methods thoroughly exploit the intrinsic capabilities of NeRF as an implicit scene representation without introducing additional model priors. However, due to the implicit representation of scenes and the dense sampling of points along rays, they still face challenges in achieving real-time performance.

2.2. 3D Gaussian Splatting

3D Gaussian Splatting (3D-GS) [19] has demonstrated notable advancements in rendering performance by departing from MLPs and ray sampling, opting instead for anisotropic Gaussians and projection. This paradigm shift has attracted considerable attention within the industry, leading to various studies upon this method [4, 6, 7, 24, 26].

Recent endeavors aim to enhance the robustness in sparse-view scenarios [44, 45], performance, storage efficiency [30] and mesh reconstruction [14] of 3D-GS. However, none of these methods specifically address the needle-like artifacts that are absent in NeRF but present in 3D-GS. Some methods [16, 47] are exploring the potential of 3D-GS. Nevertheless, the former [16] proposes the optimal projection to address artifacts caused by the increased projection error with a larger field of view. Mip-Splatting [47] proposes filtering to address aliasing caused by sampling. Analytic-Splatting [23] replaces Mip-Splatting's 2D Mip filter with a closed-form expression for the Gaussian integral within a pixel. These methods [23, 47] partially alleviate the needle-like artifacts. However, they still rely on the conventional density control solely based on the spectral radius and loss gradients without shape-awareness, leading to loss of high-frequency details.
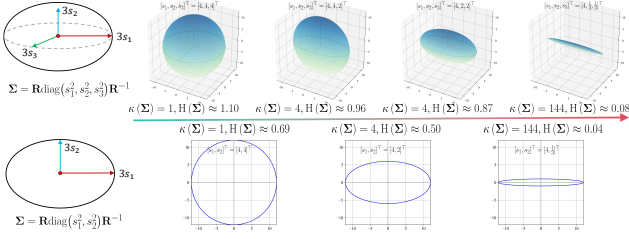
Figure 2. Visualization of Gaussians with the same spectral radius but different shapes. The spectrum of the 3D Gaussian is characterized by s_1, s_2, s_3 (top row), while the 2D Gaussian is characterized by s_1, s_2 (bottom row). From left to right, as the spectral entropy decreases and the condition number increases, the Gaussians transition from isotropic to anisotropic.

3. Preliminaries

Representation and Projection: 3D Gaussian Splatting [19] constructs a scene representation using volumetric primitives $\mathcal{G}^{3D}(\cdot)$, each characterized by position (also known as the mean) $\boldsymbol{\mu}$, a covariance matrix $\boldsymbol{\Sigma}$ (decomposed into scale $\mathbf{S} \in \mathbb{R}^{3 \times 3}$ and rotation $\mathbf{R} \in \mathrm{SO}(3)$, *i.e.*, $\boldsymbol{\Sigma} = \mathbf{R} \mathbf{S} \mathbf{S}^\top \mathbf{R}^\top$), opacity o, and spherical harmonics coefficients $\mathrm{SH}(\cdot)$. The 3D primitives are projected to 2D image space through a Jacobian matrix $\mathbf{J} \in \mathbb{R}^{2 \times 3}$ for the local affine approximation after being transformed to the camera space via a viewing transformation matrix $\mathbf{W}$. Then 2D Gaussians $\mathcal{G}^{2D}(\cdot)$, each characterized by the position $\boldsymbol{\mu}_{\mathrm{proj}}$ and a covariance matrix $\boldsymbol{\Sigma}_{\mathrm{proj}} = \mathbf{J} \mathbf{W} \boldsymbol{\Sigma} \mathbf{W}^\top \mathbf{J}^\top$, are rasterized using α-blending.

Optimization and Densification: 3D-GS [19] employs a loss function that combines $\mathcal{L}_1$ loss with a D-SSIM term:

$$\mathcal{L} = (1 - \lambda_1)\mathcal{L}_1 + \lambda_1 \mathcal{L}_{\text{D-SSIM}}. \quad (1)$$

For this loss can only optimize the parameters of Gaussian primitives but cannot change the number of primitives, phenomena such as "over-reconstruction" (regions where Gaussians cover large areas in the scene) and "under-reconstruction" (regions with missing geometric features) can occur. To address this, 3D-GS introduces an adaptive Gaussian densification scheme. For Gaussians with large view-space positional gradients $\nabla_{\boldsymbol{\mu}_{\mathrm{proj}}}\mathcal{L}$, the scheme chooses between clone and split strategies based on the scale of Gaussians. After splitting, the shape remains unchanged, with the scale being $\frac{1}{k}$ ($k = 1.6$ in all experiments) of the original, *i.e.*, $\boldsymbol{\Sigma}_{\mathrm{split}} = \frac{1}{k^2}\boldsymbol{\Sigma}$.

Filtering and Mip-Splatting [47]: To prevent projected Gaussians from becoming too small to cover an entire pixel, 3D-GS uses an EWA filter [51] for training stability:

$$\mathcal{G}_k^{2D}(\mathbf{u})_{\mathrm{EWA}} = o e^{-\frac{1}{2}(\mathbf{u}-\boldsymbol{\mu}_{\mathrm{proj}})^\top (\boldsymbol{\Sigma}_{\mathrm{proj}}+s\mathbf{I})^{-1}(\mathbf{u}-\boldsymbol{\mu}_{\mathrm{proj}})} \quad (2)$$
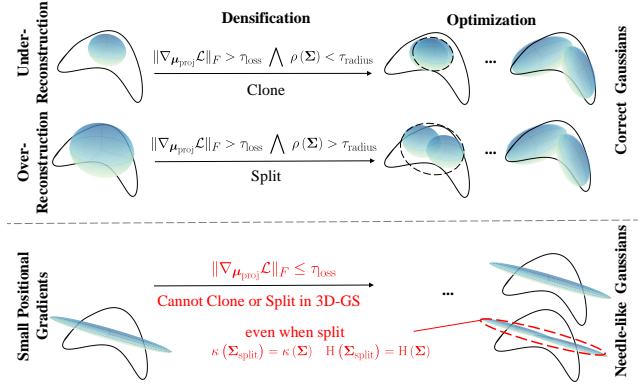

Figure 3. Illustrations of the optimization and densification of Gaussians in 3D-GS [19]. *Correct Gaussians:* When view-positional gradients $\nabla_{\boldsymbol{\mu}_{\mathrm{proj}}}\mathcal{L}$ exceed a certain threshold τ_{loss}, 3D-GS decides to clone or split based on the Gaussian's spectral radius $\rho(\boldsymbol{\Sigma})$. *Needle-like Gaussians:* However, 3D-GS does not split or clone Gaussians with low spectral entropy but small gradients.

where $\mathbf{u}$ is the pixel coordinate, $\mathbf{I}$ is a 2D identity matrix, s is a scalar hyperparameter to control the size of the filter, and $\mathcal{G}_k^{2D}(\cdot)_{\mathrm{EWA}}$ is the EWA filtered Gaussian.

To limit the maximal frequency for the 3D representation, Mip-Splatting [47] applies a 3D smoothing filter $\mathcal{G}_{\mathrm{low}}$ to the 3D Gaussians $\mathcal{G}^{3D}$, ensuring that the regularized Gaussians cover at least one pixel in all training views:

$$\mathcal{G}_k^{3D}(\mathbf{x})_{\mathrm{reg}} = (\mathcal{G}^{3D} \otimes \mathcal{G}_{\mathrm{low}})(\mathbf{x})$$

$$= o\sqrt{\frac{|\boldsymbol{\Sigma}|}{\left|\boldsymbol{\Sigma} + \frac{s}{\hat{\nu}_k} \cdot \mathbf{I}\right|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \left(\boldsymbol{\Sigma}+\frac{s}{\hat{\nu}_k}\cdot\mathbf{I}\right)^{-1}(\mathbf{x}-\boldsymbol{\mu})}. \quad (3)$$

Here, the scale $\frac{s}{\hat{\nu}}$ of the 3D filters for each primitive are different as they depend on the training views in which they are visible. In addition, the box filter of a function is equivalent to the integral over the corresponding region, while the convolution of Gaussians remains a Gaussian. Hense, Mip-Splatting [47] proposes a 2D Mip filter to approximate the integral or super-sampling within a pixel:

$$\mathcal{G}_k^{2D}(\mathbf{u})_{\mathrm{Mip}} = o\sqrt{\frac{|\boldsymbol{\Sigma}_{\mathrm{proj}}|}{|\boldsymbol{\Sigma}_{\mathrm{proj}}+s\mathbf{I}|}} e^{-\frac{1}{2}(\mathbf{u}-\boldsymbol{\mu}_{\mathrm{proj}})^\top (\boldsymbol{\Sigma}_{\mathrm{proj}}+s\mathbf{I})^{-1}(\mathbf{u}-\boldsymbol{\mu}_{\mathrm{proj}})}$$

$$(4)$$

where $\mathcal{G}_k^{2D}(\cdot)_{\mathrm{Mip}}$ is the 2D Mip filtered Gaussian. To compute the Gaussian integral within a pixel more analytically, Analytic-Splatting [23] derives the cumulative distribution function (CDF) of the Gaussians, replacing the 2D Mip filter used to approximate a box filter.

4. Spectral Analysis of 3D Gaussian Splatting

In this section, we first perform spectral analysis to 3D-GS and show loss-sensitivity and shape-unwareness in densifi-
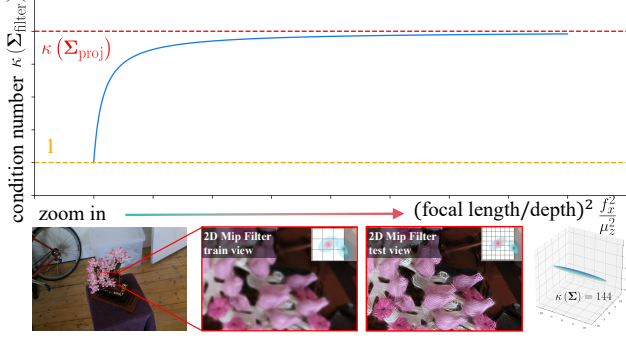
Figure 4. Illustrations of the condition number variation curve and rendering results when zooming in. We fix the condition number of 3D Gaussians $\kappa(\mathbf{\Sigma}) = 144$ during training. Due to view-inconsistency in filtering, the train view still produces satisfactory rendering results (2D Mip Filter train view), but the test view with higher $\frac{f_x^2}{\mu_z^2}$ shows needle-like artifacts (2D Mip Filter test view).

cation, and view-inconsistency in filtering.

Spectra of Gaussians: The covariance matrix $\mathbf{\Sigma}$ of a 3D/2D Gaussian is analogous to describing the configuration of an ellipsoid/ellipse. For example, the covariance matrix of 3D Gaussian can be eigendecomposed as follows:

$$\mathbf{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^\top\mathbf{R}^\top = \mathbf{R}\left(\mathbf{S}\mathbf{S}^\top\right)\mathbf{R}^{-1} = \mathbf{R}\mathrm{diag}\left(s_1^2, s_2^2, s_3^2\right)\mathbf{R}^{-1} \tag{5}$$

where the rotation matrix $\mathbf{R}$ is an orthogonal matrix ($\mathbf{R}^\top = \mathbf{R}^{-1}$), $3s_1, 3s_2, 3s_3$ represent the lengths of the ellipsoid's three axes (3σ rule) [19], and s_1^2, s_2^2, s_3^2 are the eigenvalues (spectrum) of $\mathbf{\Sigma}$. The spectral radius of the covariance matrix $\mathbf{\Sigma}$ can be derived as:

$$\rho(\mathbf{\Sigma}) = \max\left(s_1^2, s_2^2, s_3^2\right) \tag{6}$$

and can be used to measure the scale of the Gaussian. Additionally, the condition number and spectral entropy are respectively given by

$$\kappa(\mathbf{\Sigma}) = \frac{\rho(\mathbf{\Sigma})}{\rho_{\min}(\mathbf{\Sigma})} = \frac{\max\left(s_1^2, s_2^2, s_3^2\right)}{\min\left(s_1^2, s_2^2, s_3^2\right)}, \tag{7}$$

$$\mathrm{H}(\mathbf{\Sigma}) = \mathrm{tr}\left(-\frac{\mathbf{\Sigma}}{\mathrm{tr}(\mathbf{\Sigma})}\ln\frac{\mathbf{\Sigma}}{\mathrm{tr}(\mathbf{\Sigma})}\right) = -\sum_{i=1}^{3}\frac{s_i^2}{\mathrm{tr}(\mathbf{\Sigma})}\ln\frac{s_i^2}{\mathrm{tr}(\mathbf{\Sigma})}, \tag{8}$$

where $\mathrm{tr}(\mathbf{\Sigma}) = s_1^2 + s_2^2 + s_3^2$. These metrics can be used to measure the shape or degree of anisotropy of the Gaussian. In Figure 2, we visualize different 3D/2D Gaussians with the same spectral radius, i.e., $\rho(\mathbf{\Sigma}) = 16$.

It is easily proven that when $s_1 = s_2 = s_3$, the condition number is minimized and the spectral entropy is maximized. Furthermore, the eccentricity e of the ellipse described by $\mathbf{\Sigma}$ of the 2D Gaussian satisfies $e = \sqrt{1 - \frac{1}{\kappa(\mathbf{\Sigma})}}$,

which indicates that a higher condition number corresponds to a sharper shape of the Gaussian as shown in Figure 2. And the needle-like artifacts correspond to Gaussians with a low spectral entropy and a high condition number. Please refer to the detailed proofs in the supplementary materials.

Loss-Sensitivity and Shape-Unawareness in Densification: In Section 3, we have briefly introduced the optimization and densification in 3D-GS [19]. Here, we further analyze the spectra of the Gaussians involved in the densification. As shown in Figure 3, 3D-GS employs densification to grow the quantity of Gaussians when the loss gradient $\nabla_{\boldsymbol{\mu}_{\mathrm{proj}}}\mathcal{L}$ exceed a certain threshold τ_{loss}. While this approach effectively addresses issues of "over-reconstruction" and "under-reconstruction", it is highly sensitive to the design of the loss function $\mathcal{L}$ and the chosen threshold τ_{loss}. Specifically, when elongated Gaussians with low spectral entropy can fit high-frequency textures or geometry with small loss in the training views, the densification mechanism is not triggered. This can lead to needle-like artifacts or high-frequency over-blurriness. Furthermore, even when splitting, the spectral entropy and the condition number of the Gaussians remain consistent with that before the split and does not significantly alleviate needle-like artifacts, i.e., $\rho(\mathbf{\Sigma}_{\mathrm{split}}) = \frac{1}{k^2}\rho(\mathbf{\Sigma}), \kappa(\mathbf{\Sigma}_{\mathrm{split}}) = \kappa(\mathbf{\Sigma}), \mathrm{H}(\mathbf{\Sigma}_{\mathrm{split}}) = \mathrm{H}(\mathbf{\Sigma})$ (red text and lines in Figure 3).

View-Inconsistency in Filtering: In Section 3, we have provided a brief introduction to the EWA filter [51] in 3D-GS and the 2D Mip filter in Mip-Splatting [47]. From Equation 2 and 4, we observe that the two filters have identical covariance matrices $\mathbf{\Sigma}_{\mathrm{filter}} = \mathbf{\Sigma}_{\mathrm{proj}} + s\mathbf{I}$, differing only in the opacity term, i.e., $o \neq o\sqrt{\frac{|\mathbf{\Sigma}_{\mathrm{proj}}|}{|\mathbf{\Sigma}_{\mathrm{proj}} + s\mathbf{I}|}}$. However, when zooming in, the Jacobian matrix $\mathbf{J}$ changes, leading to a change in the projected covariance matrix $\mathbf{\Sigma}_{\mathrm{proj}}$ while the filter kernel $s\mathbf{I}$ remains constant. This causes variations in the condition number of 2D Gaussians after optimization:

$$\kappa(\mathbf{\Sigma}_{\mathrm{train}}) = \frac{\rho\left(\mathbf{J}_{\mathrm{train}}\mathbf{\Sigma}'\mathbf{J}_{\mathrm{train}}^\top\right) + s}{\rho_{\min}\left(\mathbf{J}_{\mathrm{train}}\mathbf{\Sigma}'\mathbf{J}_{\mathrm{train}}^\top\right) + s} \neq \kappa(\mathbf{\Sigma}_{\mathrm{test}}) \tag{9}$$

where $\mathbf{\Sigma}' = \mathbf{W}\mathbf{\Sigma}\mathbf{W}^\top$ is the covariance matrix in the camera space and $\mathbf{\Sigma}_{\mathrm{train}} = \mathbf{J}_{\mathrm{train}}\mathbf{\Sigma}'\mathbf{J}_{\mathrm{train}}^\top + s\mathbf{I}, \mathbf{\Sigma}_{\mathrm{test}} = \mathbf{J}_{\mathrm{test}}\mathbf{\Sigma}'\mathbf{J}_{\mathrm{test}}^\top + s\mathbf{I}$ denote the covariance matrices during training and testing, respectively. We have derived and visualized the variation curve of $\kappa(\mathbf{\Sigma}_{\mathrm{filter}})$ during the zoom-in process. As shown in the upper part of Figure 4, this function increases as the camera zooms in, i.e., as the $\frac{\text{focal length}}{\text{depth}}$ increases. Please refer to the supplementary materials for detailed derivations.

Due to the view-inconsistency in filtering, needle-like artifacts become more pronounced when zooming in or when the camera moves closer to the object, as illustrated by the rendering results in the lower portion of Figure 4.
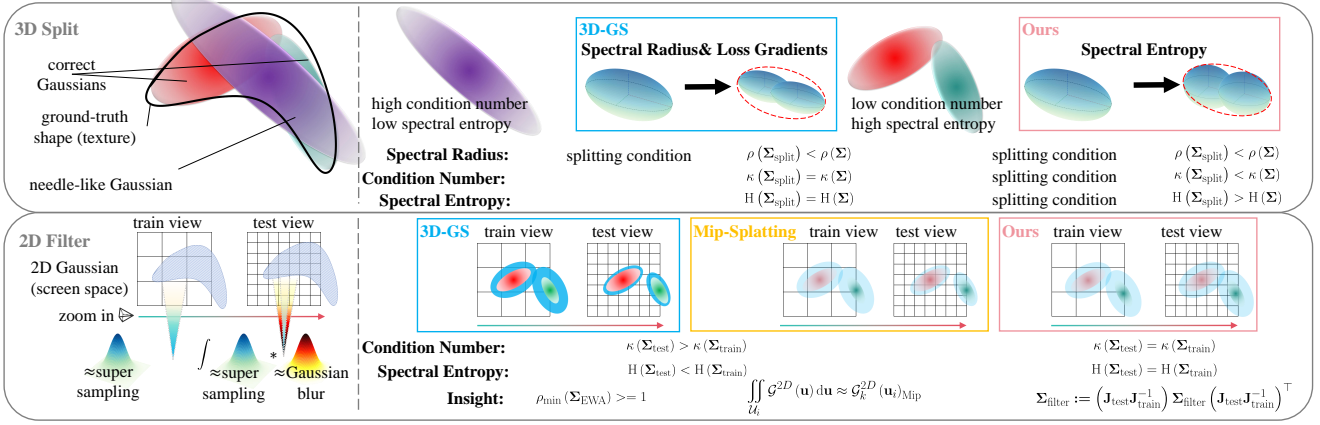
Figure 5. Overview of Spectral-GS. 3D Gaussian Splatting (3D-GS) [19] decides whether to split based on the positional gradients and the spectral radius of the covariance matrix without considering the shape of primitives. We propose the 3D shape-aware splitting strategy based on the spectral analysis (3D Split). In screen space, both the EWA filter [51] of 3D-GS which attempts to cover an entire pixel, and the Mip filter of Mip-Splatting [47] which approximates supersampling, result in a reduction of spectral entropy when zooming in to synthesize novel view. Our view-consistent filter's kernel is not constant to maintain the spectral entropy consistency (2D Filter).

5. Spectral-GS

Based on the spectral analysis for 3D-GS [19] and Mip-Splatting [47] in Section 4, we propose the 3D shape-aware splitting in Section 5.1 and the 2D view-consistent filtering in Section 5.2, respectively, to address loss sensitivity and shape unawareness in densification, as well as view inconsistency in filtering. The overview of our method Spectral-GS is illustrated in Figure 5.

5.1. 3D Shape-Aware Splitting

We propose the 3D shape-aware splitting to introduce shape-awareness into the optimization process. As shown in Ours of 3D Split in Figure 5, the splitting condition of our strategy is based on the spectral entropy of 3D Gaussians $H(\mathbf{\Sigma})$. When the spectral entropy exceeds a certain threshold τ_{spectral} and may exhibit visually needle-like artifacts, K points are sampled based on the probability density function (PDF) of the old Gaussian distribution $\mathcal{G}^{3D}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{\Sigma})$. The new Gaussian mixture distributions $\sum_K \mathcal{G}^{3D}(\mathbf{x}; \boldsymbol{\mu}_{\text{split}}, \mathbf{\Sigma}_{\text{split}})$ aim to fit the old Gaussian distribution $\mathcal{G}^{3D}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{\Sigma})$ as closely as possible, preserving high-frequency while increasing spectral entropy. Specifically, the reduction factor of the covariance matrix is not isotropic but anisotropic. The greedy algorithm reduces the spectral radius by the maximum extent:

$$\mathbf{\Sigma}_{\text{split}} = \mathbf{R}\text{diag}\left(\frac{1}{k_1^2}s_1^2, \frac{1}{k_2^2}s_2^2, \frac{1}{k_3^2}s_3^2\right)\mathbf{R}^\top, \quad (10)$$

$$k_i = k \cdot \mathbb{1}\left\{s_i^2 = \rho(\mathbf{\Sigma})\right\} + k_0, \quad (11)$$

where $\mathbb{1}\{\cdot\}$ is an indicator function and $k > 0, k_0 \geq 1$. Additionally, to ensure that the condition number after splitting does not exceed that before splitting, the following condition must be satisfied:

$$k < -k_0 + \frac{k_0 \rho^{\frac{3}{2}}(\mathbf{\Sigma})}{\sqrt{|\mathbf{\Sigma}|}}. \quad (12)$$

Please refer to the supplementary materials for more details.

5.2. 2D View-Consistent Filtering

For novel view synthesis, the level of scene detail is determined by the resolution of the training images. The finer details revealed by zooming in actually correspond to views that are not seen in the training set. Due to view-inconsistent filtering, 3D-GS [19] and Mip-Splatting [47] produce pronounced artifacts when zooming in. We propose the 2D view-consistent filtering, which combining a convolution that approximates supersampling with a Gaussian blur that approximates interpolation, as shown in Ours of 2D Filter in Figure 5. We first tend to approximate the integral of projected 2D Gaussians within each pixel window area with the 2D Gaussian filter:

$$\mathcal{G}_k^{2D}(\mathbf{u})_{\text{Box}} = \left(\mathcal{G}^{2D} \otimes \mathcal{B}\right)(\mathbf{u}) = \iint_{\mathcal{U}} \mathcal{G}^{2D}(\mathbf{u})\, d\mathbf{u}, \quad (13)$$

$$\mathcal{G}_k^{2D}(\mathbf{u})_{\text{Box}} \approx o\sqrt{\frac{|\mathbf{\Sigma}_{\text{proj}}|}{|\mathbf{\Sigma}_{\text{proj}} + s\mathbf{I}|}} e^{-\frac{1}{2}\left(\mathbf{u}-\boldsymbol{\mu}_{\text{proj}}\right)^\top\left(\mathbf{\Sigma}_{\text{proj}}+s\mathbf{I}\right)^{-1}\left(\mathbf{u}-\boldsymbol{\mu}_{\text{proj}}\right)}, \quad (14)$$

where $\mathcal{B}$ denotes the 2D box filter. Subsequently, we introduce the Gaussian blur to achieve view-consistency of the condition number, with the size of the convolution kernel given by:

$$\mathbf{\Sigma}_{\text{blur}} = \left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)s\mathbf{I}\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)^\top - s\mathbf{I}. \quad (15)$$
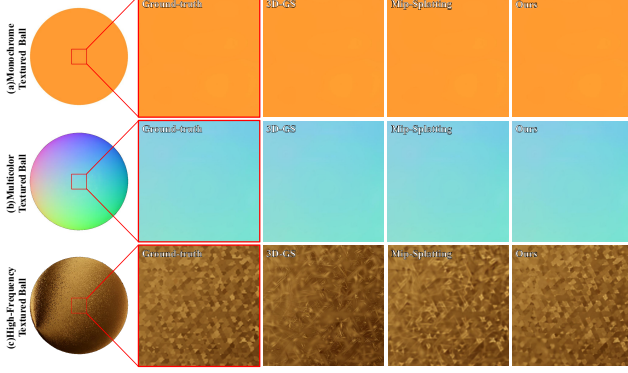
5

Figure 6. We conduct experiments on BALL with identical geometry but different textures. From top to bottom, they are the monochrome textured **(a)**, multicolor textured **(b)**, and high-frequency textured **(c)**.

Note that the matrix $\mathbf{J}_{\text{train}} \in \mathbb{R}^{2\times 3}$ is not a full-rank square matrix; $\mathbf{J}_{\text{train}}^{-1} \in \mathbb{R}^{3\times 2}$ represents a left/right inverse inverse of matrix $\mathbf{J}_{\text{train}}$. Since convolving two Gaussians with covariance matrices Σ_1, Σ_2 results in another Gaussian with variance $\Sigma_1 + \Sigma_2$, we obtain the result of combining the 2D box filter with Gaussian blur $\mathcal{G}^{blur}$ as follows:

$$\mathcal{G}_k^{2D}(\mathbf{u})_{\text{filter}} \approx \left(\mathcal{G}^{2D} \otimes \mathcal{B} \otimes \mathcal{G}^{blur}\right)(\mathbf{u})$$
$$= o_{\text{filter}} e^{-\frac{1}{2}\left(\mathbf{u}-\boldsymbol{\mu}_{\text{proj}}\right)^\top \Sigma_{\text{filter}}^{-1}\left(\mathbf{u}-\boldsymbol{\mu}_{\text{proj}}\right)}, \quad (16)$$

$$o_{\text{filter}} = o \sqrt{\frac{|\Sigma_{\text{proj}}|}{\left|\Sigma_{\text{proj}} + \left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right) s\mathbf{I}\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)^\top\right|}}, \quad (17)$$

$$\Sigma_{\text{filter}} = \Sigma_{\text{proj}} + \left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right) s\mathbf{I}\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)^\top. \quad (18)$$

Compared to the 2D Mip filter, our 2D view-consistent filter does not use a constant kernel but instead employs a view-adaptive kernel. We can also approximate Equation 18 using the filter kernel function $s(\text{focal length}, \text{depth}) = s_0 \frac{\text{focal length}^2}{\text{depth}^2}$, where s_0 is a constant. Please refer to the detailed proofs in the supplementary materials.

6. Experiments

To validate the effectiveness of Spectral-GS derived from spectral analysis, we conduct a series of experiments, comparing it with the original 3D-GS [19] and some current state-of-the-art methods. In these experiments, Analytic-Splatting+3D Filter refers to Analytic-Splatting [23] with the 3D smoothing filter from Mip-Splatting [47].

6.1. Implementation

We implement Spectral-GS based on the PyTorch framework in 3D-GS [19]. We use the default parameters of 3D-GS to maintain consistency with the original 3D-GS. For our approach, we empirically set the threshold τ_{spectral} in 3D

shape-aware splitting to 0.5, with $k = 0.6$, $k_0 = 1$ and $K = 2$, and s_0 in 2D view-consistent filtering to 0.1.

6.2. Datasets

We test our algorithm on a total of 12 scenes which are commonly used in NeRF [28] and 3D-GS [19].

Synthetic Scenes [28, 41]: In particular, we evaluate our approach on a total of six synthetic scenes. HOT-DOG, CHAIR, SHIP, LEGO, and MATERIALS are from the *Blender dataset* [28], while BALL is from the *Shiny Blender dataset* [41] but with modified textures.

Real Scenes [2, 15, 17, 20]: Additionally, we test our method on a total of six real scenes. TRUCK, PLAYROOM and FLOWERS are from the *Tanks Templates* [20], *Deep Blending* [15] and the *Mip-NeRF360* [2] datasets, respectively. TRIPOD, STONE and PILLOW are *High-frequency Spectrum* dataset captured by us.

6.3. Results

Quantitative comparisons: We adopt a train/test split for real datasets following the methodology proposed by 3D-GS [19]. Standard metrics such as PSNR, LPIPS [49], and SSIM are employed for evaluation. We report quantiative results in Table 1. Additionally, to verify the correlation between the spectral entropy of 3D Gaussians after optimization H (Σ) and the quality of novel view synthesis, we also provide the spectral entropy metric for each scene in Table 1. We employ the setting involving renderings at various focal lengths (*i.e.* $1\times$, $2\times$, $4\times$, $8\times$) to mimic zoom-in effects. Table 1 demonstrates that our spectral analysis-based method effectively increases the spectral entropy of scenes, thereby enhancing images' quality.

Qualitative comparisons: As illustrated in Figure 9, it can be observed that our method is capable of generating more realistic details, with fewer needle-like artifacts compared to 3D-GS [19], Mip-Splatting [47] and Analytic-Splatting [23]. And this is precisely the superiority brought about by our spectral analysis-based method, which results in higher spectral entropy.

6.4. Discussions

Ablation Study: We conduct ablation studies in the BALL scene, as shown in Figure 7. The observed needle-like artifacts are composed of two parts: those inherent to the 3D scene and those generated by the rendering algorithm. The artifacts inherent to the scene are addressed by regulating the 3D Gaussians' spectral entropy through our 3D splitting **(b)(e)**, while the artifacts caused by rendering are resolved by maintaining condition number consistency via our 2D filtering **(d)(e)**. Moreover, 3D splitting, which does not rely on loss gradients, enhances the representation of high-frequency details in the scene. Recall that 3D-GS [19] is

Table 1. Quantitative evaluation of our method compared to previous work across 12 scenes.

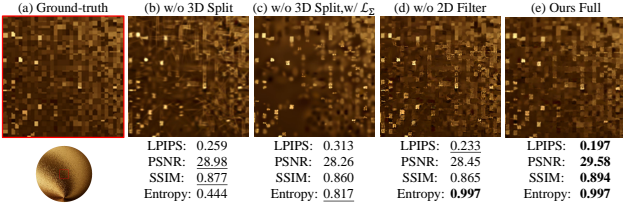| Metric	Method	Synthetic Dataset [28, 41]						Real Dataset [2, 15, 20]					
		Hotdog	Chair	Ship	Lego	Materials	Ball	Truck	Playroom	Flowers	Tripod	Stone	Pillow
LPIPS↓	3D-GS [19]	0.204	0.213	0.310	0.189	0.141	0.377	0.221	0.215	0.366	0.241	0.280	0.292
	Mip-Splatting [47]	0.126	0.114	0.216	0.111	0.113	0.285	0.166	0.201	0.315	0.222	0.230	0.275
	Analytic-Splatting [23]	0.165	0.173	0.280	0.150	0.141	0.293	0.176	0.205	0.342	0.228	0.281	0.289
	Analytic-Splatting+3D Filter [23]	0.133	0.112	0.219	0.119	0.115	0.294	0.176	0.205	0.342	0.226	0.274	0.287
	Ours	0.099	0.098	0.196	0.098	0.100	0.197	0.129	0.186	0.300	0.214	0.217	0.266
PSNR↑	3D-GS [19]	29.17	23.10	23.41	28.23	27.25	23.85	25.16	31.12	18.97	21.92	27.13	25.28
	Mip-Splatting [47]	33.85	30.75	28.44	31.86	29.18	29.03	26.59	31.85	21.90	23.49	29.48	25.94
	Analytic-Splatting [23]	32.99	29.27	26.93	31.44	28.75	29.08	26.57	31.92	21.64	23.15	29.65	25.90
	Analytic-Splatting+3D Filter [23]	33.73	30.92	28.46	31.47	29.09	29.10	26.53	31.77	21.64	23.14	29.65	25.96
	Ours	34.16	31.51	28.79	32.29	29.43	29.58	26.86	31.92	22.11	24.37	30.13	26.34
SSIM↑	3D-GS [19]	0.889	0.795	0.718	0.887	0.914	0.750	0.866	0.923	0.542	0.725	0.854	0.709
	Mip-Splatting [47]	0.952	0.943	0.864	0.944	0.936	0.873	0.893	0.928	0.637	0.767	0.892	0.720
	Analytic-Splatting [23]	0.931	0.864	0.796	0.929	0.925	0.875	0.891	0.927	0.620	0.754	0.784	0.718
	Analytic-Splatting+3D Filter [23]	0.952	0.934	0.865	0.940	0.936	0.875	0.891	0.928	0.620	0.756	0.919	0.719
	Ours	0.956	0.945	0.866	0.947	0.937	0.894	0.907	0.930	0.644	0.773	0.914	0.735
Entropy↑	3D-GS [19]	0.152	0.083	0.141	0.189	0.242	0.140	0.254	0.276	0.239	0.357	0.351	0.398
	Mip-Splatting [47]	0.339	0.397	0.338	0.361	0.397	0.444	0.544	0.349	0.664	0.442	0.532	0.489
	Analytic-Splatting [23]	0.234	0.222	0.210	0.241	0.285	0.362	0.377	0.292	0.492	0.416	0.430	0.467
	Analytic-Splatting+3D Filter [23]	0.332	0.390	0.334	0.354	0.397	0.496	0.518	0.321	0.682	0.425	0.478	0.479
	Ours	0.697	0.992	0.996	0.970	0.967	0.997	0.881	0.794	0.958	0.836	0.908	0.615

Figure 7. Illustrations of the ablation study on BALL. From left to right: ground truth (a), our method without 3D split (b), our method without 3D split but with $\mathcal{L}_{\boldsymbol{\Sigma}}$ (c), our method without 2D filter (d), and the full version of our method (e). The corresponding metrics are provided below the image.
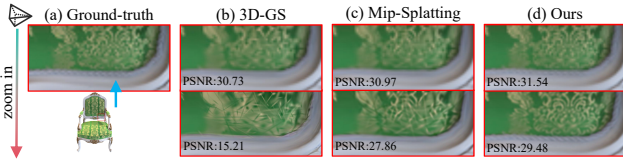

Figure 8. We present results from different methods at various sampling rates (focal lengths) on CHAIR. The images are ordered from top to bottom, corresponding to the transition from the training focal length to larger focal lengths. Each method includes a PSNR metric in the lower-left corner.

sensitive to loss and lacks shape-awareness. Another idea is to introduce a regularization term $\mathcal{L}_{\boldsymbol{\Sigma}}$ into the loss function that accounts for 3D Gaussians' shape:

$$\mathcal{L}_{\text{naive}} = (1-\lambda_1)\mathcal{L}_1 + \lambda_1 \mathcal{L}_{\text{D-SSIM}} + \lambda_2 \mathcal{L}_{\boldsymbol{\Sigma}}. \quad (19)$$

However, although this regularization term is able to effectively constrain and optimize the shape of Gaussians $(\mathrm{H}(\boldsymbol{\Sigma}) = 0.817 > 0.444)$, it does not affect densification

since the gradient with respect to $\boldsymbol{\mu}_{\text{proj}}$ is zero:

$$\|\nabla_{\boldsymbol{\mu}_{\text{proj}}} \mathcal{L}_{\boldsymbol{\Sigma}}\|_F = 0, \quad \|\nabla_{\boldsymbol{\Sigma}} \mathcal{L}_{\boldsymbol{\Sigma}}\|_F \neq 0, \quad (20)$$

$$\|\nabla_{\boldsymbol{\mu}_{\text{proj}}} \mathcal{L}_{\text{naive}}\|_F = \|\nabla_{\boldsymbol{\mu}_{\text{proj}}} \mathcal{L}\|_F, \quad (21)$$

which can lead to loss of high-frequency details (c)(e).

Relationship with Frequency: It is worth noting that Mip-Splatting [47] outperforms 3D-GS [19] in both spectral entropy and image quality metrics across all scenes, suggesting a relationship between the 3D smoothing filter based on frequency analysis and ours spectral analysis. Firstly, it is straightforward to prove that the 3D smoothing filter in Mip-Splatting increases the spectral entropy of the original Gaussian. Secondly, we know that Gaussian functions are closed under the Fourier transform (FT) [32] while the covariance in the frequency domain is the inverse in the spatial domain (scaled by a coefficient):

$$\mathcal{F}\{\mathcal{G}(\mathbf{x})\} = e^{-2\pi^2 \boldsymbol{\omega}^\top \boldsymbol{\Sigma} \boldsymbol{\omega}} \quad (22)$$

where $\mathcal{F}$ denotes the Fourier transform. The eigenvalues of the covariance matrix determine the bandwidth of the frequency spectrum. Larger eigenvalues result in a narrower spectrum in the corresponding direction, while smaller eigenvalues lead to a wider spectrum in that direction. Therefore, the spectral analysis of our covariance matrix is equivalent and unified with the frequency spectrum analysis of the Gaussian. We conduct experiments with fixed geometry on BALL with different textures to demonstrate that needle-like artifacts predominantly occur in high-frequency scenes, as shown in Figure 6. And our method effectively addresses the challenge of representing high-frequency details with Gaussians without artifacts.
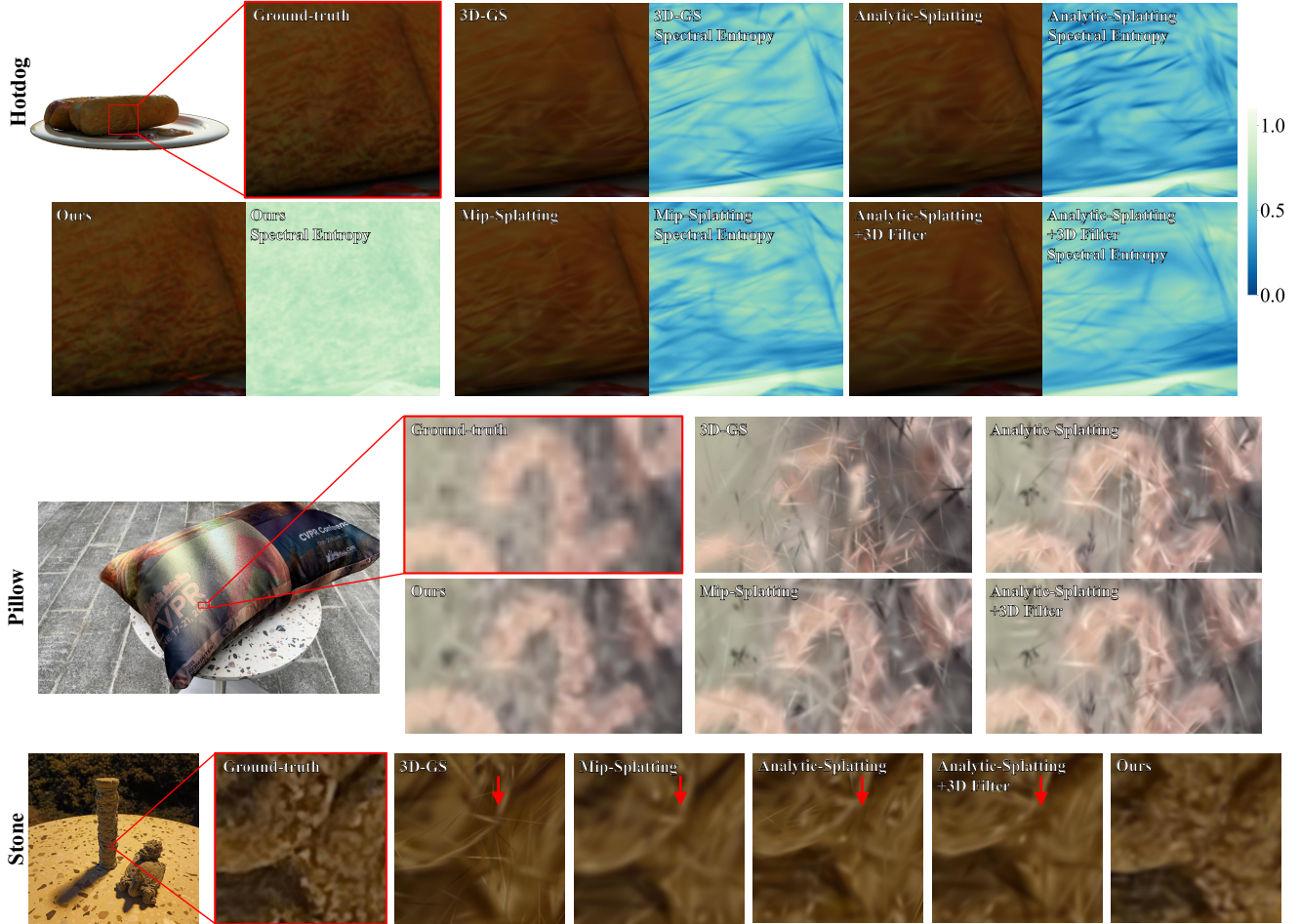
Figure 9. We show comparisons of our method to previous methods and the corresponding ground truth images from held-out test views. Additionally, We visualize the spectral entropy maps of 3D Gaussians after optimization. Bluer regions indicate lower spectral entropy, with more needle-like degraded Gaussians, while greener regions represent higher spectral entropy, without noticeable needle-like artifacts.

Impacts of Zooming in: To further validate the robustness of our method during the zoom-in process, we conduct a series of experiments. As shown in Figure 8, these methods synthesize novel views nearly identical to the ground-truth at the training view resolution, although textures from 3D-GS and Mip-Splatting are blurred at the blue arrow location due to their naive splitting algorithm. However, when increasing the focal length and sampling rate, 3D-GS exhibits pronounced needle-like artifacts that significantly degrade rendering quality and PSNR. While Mip-Splatting shows less degradation due to its use of a series of filters, Gaussian shapes still become sharper. In contrast, we maintain high quality similar to lower sampling rates, without needle-like Gaussians or loss of high-frequency details, through our 2D view-consistent filtering.

Limitation: Since our method does not introduce additional priors, such as image super-resolution networks [8], the resolution of novel views depends on the input images.

Additionally, our method includes hyperparameters similar to 3D-GS [19] and Mip-Splatting [47], so the results are influenced by these hyperparameters.

7. Conclusion

We propose Spectral-GS, a modification to 3D-GS, which introduces *3D scale-aware splitting* and *2D view-consistent filtering* strategies, based on our spectral analysis, to achieve needle-like-alias-free rendering at arbitrary close-up or zoomed-in view. Our splitting strategy effectively regularizes needle-like Gaussians and increase the spectral entropy, enhancing the high-frequency de- tails representation for 3D-GS and mitigating needle-like artifacts. And the 2D view-consistent filter combines a convolution that approximates supersampling with a Gaussian blur that approximates interpolation to resolve needle-like artifacts caused by view-inconsistency. Our experimental results validate accuracy of our analysis and effectiveness of our method.

References

[1] 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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5835–5844. IEEE, 2021. 1, 2

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 1, 2, 6, 7

[3] Chris Buehler, Michael Bosse, Leonard McMillan, Steven J. Gortler, and Michael F. Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001, Los Angeles, California, USA, August 12-17, 2001*, pages 425–432. ACM, 2001. 2

[4] Jiazhong Cen, Jiemin Fang, Chen Yang, Lingxi Xie, Xiaopeng Zhang, Wei Shen, and Qi Tian. Segment any 3d gaussians. *arXiv preprint arXiv:2312.00860*, 2023. 2

[5] Gaurav Chaurasia, Sylvain Duchêne, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM Trans. Graph.*, 32(3):30:1–30:12, 2013. 2

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

[7] Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. Gaussianeditor: Swift and controllable 3d editing with gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21476–21485, 2024. 2

[8] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015. 8

[9] David Eisenbud. *Commutative algebra: with a view toward algebraic geometry*. Springer Science & Business Media, 2013. 1

[10] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *CVPR*, 2020. 2

[11] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*, pages 1–8. IEEE Computer Society, 2007. 2

[12] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013. 1

[13] Steven J. Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen. The lumigraph. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, New Orleans, LA, USA, August 4-9, 1996*, pages 43–54. ACM, 1996. 2

[14] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024. 2

[15] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018. 6, 7, 3

[16] Letian Huang, Jiayang Bai, Jie Guo, Yuanqi Li, and Yanwen Guo. On the error analysis of 3d gaussian splatting and an optimal projection strategy. In *European conference on computer vision*, 2024. 2

[17] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 6

[18] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3d scenes. In *CVPR*, 2020. 2

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 1, 2, 3, 4, 5, 6, 7, 8

[20] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 6, 7

[21] Georgios Kopanas, Thomas Leimkühler, Gilles Rainer, Clément Jambon, and George Drettakis. Neural point catacaustics for novel-view synthesis of reflections. *ACM Trans. Graph.*, 41(6):201:1–201:15, 2022. 2

[22] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1996, New Orleans, LA, USA, August 4-9, 1996*, pages 31–42. ACM, 1996. 2

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

[24] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *2024 International Conference on 3D Vision (3DV)*, pages 800–809. IEEE, 2024. 2

[25] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 7210–7219. Computer Vision Foundation / IEEE, 2021. 2

[26] Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2

[27] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019. 2

[28] B Mildenhall, PP Srinivasan, M Tancik, JT Barron, R Ramamoorthi, and R Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, 2020. 1, 2, 6, 7

[29] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2

[30] Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3d gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10349–10358, 2024. 2

[31] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In *CVPR*, 2019. 2

[32] Henri J Nussbaumer and Henri J Nussbaumer. *The fast Fourier transform*. Springer, 1982. 7

[33] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2

[34] Gian-Carlo Rota and Gilbert Strang. A note on the joint spectral radius. *Indag. Math*, 22(4):379–381, 1960. 1

[35] Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *2007 15th European signal processing conference*, pages 606–610. IEEE, 2007. 1

[36] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 2

[37] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. 1

[38] Vincent Sitzmann, Michael Zollhoefer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *NeurIPS*, 2019. 2

[39] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3):835–846, 2006. 2

[40] Abraham Van der Sluis. Condition numbers and equilibration of matrices. *Numerische Mathematik*, 14(1):14–23, 1969. 1

[41] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 6, 7

[42] John Von Neumann. *Mathematical foundations of quantum mechanics: New edition*. Princeton university press, 2018. 1

[43] Lai Wei, Zhiquan Tan, Chenghai Li, Jindong Wang, and Weiran Huang. Large language model evaluation via matrix entropy. *arXiv preprint arXiv:2401.17139*, 2024. 1

[44] Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360° sparse view synthesis using gaussian splatting. *arXiv e-prints*, pages arXiv–2312, 2023. 2

[45] 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. 2

[46] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8254–8263, 2023. 2

[47] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 2, 3, 4, 5, 6, 7, 8

[48] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 1

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

[50] Dennis G Zill. *Advanced engineering mathematics*. Jones & Bartlett Learning, 2020. 1

[51] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Ewa splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, 2002. 2, 3, 4, 5

Spectral-GS: Taming 3D Gaussian Splatting with Spectral Entropy

Supplementary Material

A. Proofs in Spectral Analysis of Gaussians

A.1. Spectral Analysis of Matrices

Mathematically, the spectrum of a matrix refers to the set of its eigenvalues [9, 12, 50].

Eigenvalue (Spectrum): A matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ can be eigendecomposed as follows:

$$\mathbf{A} = \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^{-1} \tag{23}$$

where $\mathbf{\Lambda} = \mathrm{diag}\left(\lambda_1, \lambda_2, \ldots, \lambda_N\right)$ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues, $\mathbf{\Lambda}_{ii} = \lambda_i$. The trace of $\mathbf{A}$, denoted $\mathrm{tr}\left(\mathbf{A}\right)$, is the sum of all eigenvalues, *i.e.*, $\mathrm{tr}\left(\mathbf{A}\right) = \sum_{i=1}^{N} \lambda_i$. And the determinant of $\mathbf{A}$, denoted $\det\left(\mathbf{A}\right)$ or $|\mathbf{A}|$, is the product of all eigenvalues, *i.e.*, $\det\left(\mathbf{A}\right) = |\mathbf{A}| = \prod_{i=1}^{N} \lambda_i$.

Spectral Radius [34]: In mathematics, the spectral radius $\rho\left(\cdot\right)$ of a square matrix $\mathbf{A}$ is the maximum of the absolute values of its eigenvalues:

$$\rho\left(\mathbf{A}\right) = \max\left(|\lambda_1|, |\lambda_2|, \ldots, |\lambda_N|\right). \tag{24}$$

The eigenvector corresponding to the spectral radius of $\mathbf{A}$ is commonly referred to as the principal eigenvector.

Condition Number [40]: The condition number $\kappa\left(\cdot\right)$ of a function quantifies the sensitivity of the function's output to small perturbations in its input. When selecting the spectral radius as the matrix norm (spectral norm), the condition number of a normal matrix $\mathbf{A}$ is:

$$\kappa\left(\mathbf{A}\right) = \left\|\mathbf{A}^{-1}\right\| \|\mathbf{A}\| = \frac{\rho\left(\mathbf{A}\right)}{\rho_{\min}\left(\mathbf{A}\right)} \tag{25}$$

where $\rho_{\min}\left(\mathbf{A}\right) = \min\left(|\lambda_1|, |\lambda_2|, \ldots, |\lambda_N|\right)$ is the minimum of the absolute values of the eigenvalues.

Spectral Entropy [35, 37, 42, 43]: Let $\mathbf{A}$ be a positive semi-definite matrix ($\forall\, 0 \leq i \leq N,\ \lambda_i \geq 0$) and the trace of $\mathbf{A}$ be positive ($\mathrm{tr}\left(\mathbf{A}\right) > 0$). Then the matrix $\mathbf{K} = \frac{\mathbf{A}}{\mathrm{tr}(\mathbf{A})}$ satisfies $\mathrm{tr}\left(\mathbf{K}\right) = 1$. The spectral entropy $\mathrm{H}\left(\cdot\right)$ is:

$$\mathrm{H}\left(\mathbf{A}\right) = \mathrm{tr}\left(-\mathbf{K} \ln \mathbf{K}\right) = -\sum_{i=1}^{N} \frac{\lambda_i}{\mathrm{tr}\left(\mathbf{A}\right)} \ln \frac{\lambda_i}{\mathrm{tr}\left(\mathbf{A}\right)}. \tag{26}$$

A.2. Extrema in Spectral Analysis

This section will provide a detailed proof that the condition number is minimized and the spectral entropy is maximized when $s_1 = s_2 = s_3$.

Condition Number: It is evident that the spectrum of the Gaussian satisfies the following inequality:

$$0 < \min\left(s_1^2, s_2^2, s_3^2\right) \leq \max\left(s_1^2, s_2^2, s_3^2\right). \tag{27}$$

Consequently, the condition number satisfies

$$\kappa\left(\mathbf{\Sigma}\right) = \frac{\max\left(s_1^2, s_2^2, s_3^2\right)}{\min\left(s_1^2, s_2^2, s_3^2\right)} \geq 1, \tag{28}$$

with equality iff $\max\left(s_1^2, s_2^2, s_3^2\right) = \min\left(s_1^2, s_2^2, s_3^2\right)$. Therefore, we can conclude that the condition number of $\mathbf{\Sigma}$ is minimized when $s_1 = s_2 = s_3$.

Spectral Entropy: Using the 3D Gaussian as an example, we aim to find the maximum value of the spectral entropy in Equation 8 of the paper. Let t_i be $\frac{s_i^2}{\sum_{j=1}^{3} s_j^2}$. It is evident that this is a constrained optimization problem

$$\arg \max_{t_1, t_2, t_3} \mathrm{H}\left(\mathbf{\Sigma}\right) = -\sum_{i=1}^{3} t_i \ln t_i, \quad \text{s.t.} \sum_{i=1}^{3} t_i = 1 \tag{29}$$

which can be solved using Lagrange multipliers as below:

$$\arg \max_{t_1, t_2, t_3} \mathcal{F}\left(t_1, t_2, t_3, \lambda\right) = -\sum_{i=1}^{3} t_i \ln t_i + \lambda \left(\sum_{i=1}^{3} t_i - 1\right). \tag{30}$$

Now we can calculate the gradient:

$$\nabla_{t_1, t_2, t_3, \lambda} \mathcal{F}\left(t_1, t_2, t_3, \lambda\right) = \left(\frac{\partial \mathcal{F}}{t_1}, \frac{\partial \mathcal{F}}{t_2}, \frac{\partial \mathcal{F}}{t_3}, \frac{\partial \mathcal{F}}{\lambda}\right) \tag{31}$$

and therefore:

$$\nabla_{t_1, t_2, t_3, \lambda} \mathcal{F}\left(t_1, t_2, t_3, \lambda\right) = \mathbf{0} \iff \begin{cases} \lambda = \ln t_1 + 1 \\ \lambda = \ln t_2 + 1 \\ \lambda = \ln t_3 + 1 \\ \sum_{i=1}^{3} t_i = 1 \end{cases}. \tag{32}$$

In summary, we can conclude that the spectral entropy of $\mathbf{\Sigma}$ is maximized when $t_1 = t_2 = t_3$ ($s_1 = s_2 = s_3$).

A.3. Relationship Between $\kappa\left(\cdot\right)$ and $\mathrm{H}\left(\cdot\right)$

We can express the spectral entropy $\mathrm{H}\left(\mathbf{\Sigma}\right)$ as a function of the condition number $\kappa\left(\mathbf{\Sigma}\right)$ for the 2D Gaussian:

$$\mathrm{H}\left(\mathbf{\Sigma}\right) = -\frac{s_1^2}{s_1^2 + s_2^2} \ln \frac{s_1^2}{s_1^2 + s_2^2} - \frac{s_2^2}{s_1^2 + s_2^2} \ln \frac{s_2^2}{s_1^2 + s_2^2}$$
$$= \ln\left(\kappa\left(\mathbf{\Sigma}\right) + 1\right) - \frac{\kappa\left(\mathbf{\Sigma}\right) \ln \kappa\left(\mathbf{\Sigma}\right)}{\kappa\left(\mathbf{\Sigma}\right) + 1}. \tag{33}$$

Then we calculate the derivative:

$$\frac{\mathrm{dH}\left(\mathbf{\Sigma}\right)}{\mathrm{d}\kappa\left(\mathbf{\Sigma}\right)} = -\frac{\ln\kappa\left(\mathbf{\Sigma}\right)}{\left(\kappa\left(\mathbf{\Sigma}\right)+1\right)^2} \leq 0, \text{ where } \kappa\left(\mathbf{\Sigma}\right) \geq 1, \quad (34)$$

which indicates that the spectral entropy of a 2D Gaussian decreases as the condition number increases. We visualize this function in Figure 10a.

(a) $\mathrm{H}\left(\mathbf{\Sigma}\right) = f\left(\kappa\left(\mathbf{\Sigma}\right)\right)$ for the 2D Gaussian

(b) $\mathrm{H}\left(\mathbf{\Sigma}\right) = f\left(\kappa\left(\mathbf{\Sigma},\lambda\right)\right)$ for the 3D Gaussian

Figure 10. Visualization of the function $\mathrm{H}\left(\mathbf{\Sigma}\right) = f\left(\kappa\left(\mathbf{\Sigma}\right)\right)$ for the 2D Gaussian and $\mathrm{H}\left(\mathbf{\Sigma}\right) = f\left(\kappa\left(\mathbf{\Sigma},\lambda\right)\right)$ for the 3D Gaussian.

For the 3D Gaussian, assuming without loss of generality that $s_1 \leq s_2 \leq s_3$, we have $\kappa\left(\mathbf{\Sigma}\right) = \frac{s_3^2}{s_1^2}$ and let $\lambda = \frac{s_2^2}{s_1^2}$. We can similarly derive the function for the 3D Gaussian:

$$\mathrm{H}\left(\mathbf{\Sigma}\right) = \ln\left(\kappa\left(\mathbf{\Sigma}\right)+\lambda+1\right) - \frac{\kappa\left(\mathbf{\Sigma}\right)\ln\kappa\left(\mathbf{\Sigma}\right)+\lambda\ln\lambda}{\kappa\left(\mathbf{\Sigma}\right)+\lambda+1} \quad (35)$$

and visualize the function in Figure 10b.

A.4. View-Inconsistency in Filtering

The Jacobian matrix of the local affine approximation is as follows:

$$\mathbf{J} = \begin{bmatrix} \frac{f_x}{\mu_z} & 0 & -\frac{f_x\mu_x}{\mu_z^2} \\ 0 & \frac{f_y}{\mu_z} & -\frac{f_y\mu_y}{\mu_z^2} \end{bmatrix} = \begin{bmatrix} \frac{f_x}{\mu_z} & 0 & 0 \\ 0 & \frac{f_y}{\mu_z} & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & -\frac{\mu_x}{\mu_z} \\ 0 & 1 & -\frac{\mu_y}{\mu_z} \\ 0 & 0 & 1 \end{bmatrix} \quad (36)$$

where f_x, f_y denote the intrinsic parameters of the camera model and $\boldsymbol{\mu}' = \begin{bmatrix} \mu_x & \mu_y & \mu_z \end{bmatrix}^\top$ is the position of the 3D Gaussian in the camera space. We assume that the position of the Gaussian projected onto the $z = 1$ plane, *i.e.*, $\left[\frac{\mu_x}{\mu_z}, \frac{\mu_y}{\mu_z}\right]^\top$, remains unchanged during the camera zoom-in. Since the covariance matrix in the camera space $\mathbf{\Sigma}'$ also remains unchanged when zooming in, the following matrix is a constant matrix

$$\mathbf{\Sigma}'' = \begin{bmatrix} 1 & 0 & -\frac{\mu_x}{\mu_z} \\ 0 & 1 & -\frac{\mu_y}{\mu_z} \\ 0 & 0 & 1 \end{bmatrix} \mathbf{\Sigma}' \begin{bmatrix} 1 & 0 & -\frac{\mu_x}{\mu_z} \\ 0 & 1 & -\frac{\mu_y}{\mu_z} \\ 0 & 0 & 1 \end{bmatrix}^\top = \begin{bmatrix} a & b & c \\ b & d & e \\ c & e & f \end{bmatrix} \quad (37)$$

where $a, b, c, d, e, f \in [0, +\infty)$. Then we obtain:

$$\mathbf{\Sigma}_{\text{filter}} = \mathbf{J}\mathbf{\Sigma}'\mathbf{J}^\top + s\mathbf{I} \approx \begin{bmatrix} s + \frac{af_x^2}{\mu_z^2} & \frac{bf_xf_y}{\mu_z^2} \\ \frac{bf_xf_y}{\mu_z^2} & s + \frac{df_y^2}{\mu_z^2} \end{bmatrix}. \quad (38)$$

Then we can compute the condition number of the matrix:

$$\kappa\left(\mathbf{\Sigma}_{\text{filter}}\right) = \frac{\frac{1}{2}\mathrm{tr}\left(\mathbf{\Sigma}_{\text{filter}}\right) + \sqrt{\frac{1}{4}\mathrm{tr}^2\left(\mathbf{\Sigma}_{\text{filter}}\right) - \left|\mathbf{\Sigma}_{\text{filter}}\right|}}{\frac{1}{2}\mathrm{tr}\left(\mathbf{\Sigma}_{\text{filter}}\right) - \sqrt{\frac{1}{4}\mathrm{tr}^2\left(\mathbf{\Sigma}_{\text{filter}}\right) - \left|\mathbf{\Sigma}_{\text{filter}}\right|}}$$

$$= \frac{2s + \left(a + d\frac{f_y^2}{f_x^2} + \sqrt{\left(a - d\frac{f_y^2}{f_x^2}\right)^2 + 4b^2\frac{f_y^2}{f_x^2}}\right)\frac{f_x^2}{\mu_z^2}}{2s + \left(a + d\frac{f_y^2}{f_x^2} - \sqrt{\left(a - d\frac{f_y^2}{f_x^2}\right)^2 + 4b^2\frac{f_y^2}{f_x^2}}\right)\frac{f_x^2}{\mu_z^2}} \quad (39)$$

which is a function of $\frac{f_x^2}{\mu_z^2}$ due to the other values remaining constant during the camera zoom-in. And the derivative of the function is:

$$\frac{\mathrm{d}\kappa\left(\mathbf{\Sigma}_{\text{filter}}\right)}{\mathrm{d}\frac{f_x^2}{\mu_z^2}} = \frac{4ps}{\left(2s + \left(a + d\frac{f_y^2}{f_x^2} - p\right)\frac{f_x^2}{\mu_z^2}\right)^2} > 0 \quad (40)$$

where $p = \sqrt{\left(a - d\frac{f_y^2}{f_x^2}\right)^2 + 4b^2\frac{f_y^2}{f_x^2}}$. This indicates that the condition number of a 2D Gaussian increases when zooming in the camera (increasing the $\frac{f_x}{\mu_z}$). And according to the relationship between spectral entropy and the condition number derived in Section A.3, the spectral entropy $\mathrm{H}\left(\mathbf{\Sigma}_{\text{filter}}\right)$ also decreases when zooming in the camera. The curve of $\kappa\left(\mathbf{\Sigma}_{\text{filter}}\right)$ as a function of $\frac{f_x^2}{\mu_z^2}$ is visualized in Figure 4 of the paper.

A.5. k in 3D Shape-Aware Splitting

For the 3D Gaussian, assuming without loss of generality that $s_1 \leq s_2 \leq s_3$, we can derive the following:

$$\kappa\left(\boldsymbol{\Sigma}_{\text{split}}\right) = \frac{\max\left(\frac{s_3^2}{(k+k_0)^2}, \frac{s_2^2}{k_0^2}\right)}{\min\left(\frac{s_3^2}{(k+k_0)^2}, \frac{s_1^2}{k_0^2}\right)} = \begin{cases} \frac{k_0^2 s_3^2}{(k+k_0)^2 s_1^2} \\ \frac{s_2^2}{s_1^2} \\ \frac{(k+k_0)^2 s_2^2}{k_0^2 s_3^2} \end{cases} , \quad (41)$$

$$\kappa\left(\boldsymbol{\Sigma}\right) = \frac{s_3^2}{s_1^2}. \quad (42)$$

Clearly, we have:

$$\frac{k_0^2 s_3^2}{(k+k_0)^2 s_1^2} < \frac{s_3^2}{s_1^2} = \kappa\left(\boldsymbol{\Sigma}\right), \quad \frac{s_2^2}{s_2^2} \leq \frac{s_3^2}{s_1^2} = \kappa\left(\boldsymbol{\Sigma}\right). \quad (43)$$

And when Equation 12 in the paper is satisfied, we can derive the following:

$$\frac{k+k_0}{k_0} < \frac{\rho^{\frac{3}{2}}\left(\boldsymbol{\Sigma}\right)}{\sqrt{|\boldsymbol{\Sigma}|}} = \frac{s_3^2}{s_2 s_1}, \quad (44)$$

$$\frac{(k+k_0)^2 s_2^2}{k_0^2 s_3^2} < \frac{\left(\frac{s_3^2}{s_2 s_1}\right)^2 s_2^2}{s_3^2} = \frac{s_3^2}{s_1^2} = \kappa\left(\boldsymbol{\Sigma}\right). \quad (45)$$

In summary, we can conclude that the condition number after splitting does not exceed that before splitting, *i.e.*, $\kappa\left(\boldsymbol{\Sigma}_{\text{split}}\right) \leq \kappa\left(\boldsymbol{\Sigma}\right)$, when Equation 12 in the paper is satisfied.

A.6. 2D View-Consistent Filtering Kernel

We prove the view-consistency of our filter:

$$\begin{aligned}
\kappa\left(\boldsymbol{\Sigma}_{\text{train}}\right) &= \frac{\rho\left(\mathbf{J}_{\text{train}}\boldsymbol{\Sigma}'\mathbf{J}_{\text{train}}^{\top}\right) + \rho\left(s\mathbf{I}\right)}{\rho_{\min}\left(\mathbf{J}_{\text{train}}\boldsymbol{\Sigma}'\mathbf{J}_{\text{train}}^{\top}\right) + \rho_{\min}\left(s\mathbf{I}\right)} \\
&= \frac{\rho\left(\boldsymbol{\Sigma}'\right) + \rho\left(\mathbf{J}_{\text{train}}^{-1}s\mathbf{I}\left(\mathbf{J}_{\text{train}}^{-1}\right)^{\top}\right)}{\rho_{\min}\left(\boldsymbol{\Sigma}'\right) + \rho_{\min}\left(\mathbf{J}_{\text{train}}^{-1}s\mathbf{I}\left(\mathbf{J}_{\text{train}}^{-1}\right)^{\top}\right)} \\
&= \frac{\rho\left(\boldsymbol{\Sigma}_{\text{test}}\right) + \rho\left(\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)s\mathbf{I}\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)^{\top}\right)}{\rho_{\min}\left(\boldsymbol{\Sigma}_{\text{test}}\right) + \rho_{\min}\left(\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)s\mathbf{I}\left(\mathbf{J}_{\text{test}}\mathbf{J}_{\text{train}}^{-1}\right)^{\top}\right)} \\
&= \kappa\left(\boldsymbol{\Sigma}_{\text{test}}\right).
\end{aligned}$$

$$(46)$$

In Section A.4, when $s \propto \frac{f_x^2}{\mu_z^2}$, the function $\kappa\left(\boldsymbol{\Sigma}_{\text{filter}}\right)$ and $H\left(\boldsymbol{\Sigma}_{\text{filter}}\right)$ are constant. Therefore, we can approximate this operation using the filter kernel function $s\left(f_x, \mu_z\right) = s_0 \frac{f_x^2}{\mu_z^2}$, where s_0 is a constant.

B. Spectral-GS Algorithm

Our 3D shape-aware splitting and 2D view-consistent filtering algorithms are summarized in Algorithm 1.

Algorithm 1 Spectral-GS Algorithm

W, H: width and height of the training or testing images

$M, S, C, O \leftarrow$ Gaussians()   ▷ Pos, Covs, Colors, Opacs
if is testing **then**
   $V \leftarrow$ TestingView()
   $M_p, S_p \leftarrow$ SplatGaussian(W, H, M, S, V)
   $S_f, O_f \leftarrow$ SpectralBasedFilter(S_p, O, V)   ▷ Filter
   $I \leftarrow$ Rasterize($W, H, M_p, S_f, C, O_f, V$)
else
   $i \leftarrow 0$ ▷ Iteration Count
   while not converged **do**
      $V, \hat{I} \leftarrow$ TrainingView()   ▷ Camera and Image
      $M_p, S_p \leftarrow$ SplatGaussian(W, H, M, S, V)
      $S_f, O_f \leftarrow$ SpectralBasedFilter(S_p, O, V) ▷ Filter
      $I \leftarrow$ Rasterize($W, H, M_p, S_f, C, O_f, V$)
      $\mathcal{L} \leftarrow$ Loss($I, \hat{I}$)   ▷ Compute Loss
      $M, S, C, O \leftarrow$ Adam($\nabla\mathcal{L}$)   ▷ Backprop & Step
      if IsRefinementIteration(i) **then**
         for all $\mathcal{G}^{3D}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, c, o)$ **in** (M, S, C, O) **do**
            if $o < \epsilon_o$ or IsInvalidSpectrum($o, \boldsymbol{\Sigma}$) **then**
               SpectralBasedPruneGaussian()
            end if
            if $\|\nabla_{\boldsymbol{\mu}_{\text{proj}}}\mathcal{L}\|_F > \tau_{\text{loss}}$ **then**   ▷ Densify
               if $\rho\left(\boldsymbol{\Sigma}\right) > \tau_{\text{radius}}$ **then**   ▷ Split
                  LossBasedSplitGaussian($\boldsymbol{\mu}, \boldsymbol{\Sigma}, c, o$)
               else   ▷ Clone
                  LossBasedCloneGaussian($\boldsymbol{\mu}, \boldsymbol{\Sigma}, c, o$)
               end if
            end if
            if $H\left(\boldsymbol{\Sigma}\right) < \tau_{\text{spectral}}$ **then**   ▷ Densify
               SpectralBasedSplitGaussian($\boldsymbol{\mu}, \boldsymbol{\Sigma}, c, o$)
            end if
          end for
      end if
      $i \leftarrow i + 1$
   end while
end if

C. Additional Results

Figures 11 and 12 illustrate additional qualitative comparisons. The scenes in Figure 11 are BALL with modified textures, CHAIR, PLAYROOM and DRJOHNSON from the *Deep Blending* [15], and TRUCK. The scenes in Figure 12 are from the *High-frequency Spectrum* dataset captured by us. 3D-GS [19] and Analytic Splatting [23] exhibit pronounced needle-like artifacts. In contrast, methods with the 3D smoothing filter [47] show reduced needle-like Gaussian artifacts due to the 3D filter increasing the spectral entropy of the Gaussians. Our Spectral-GS eliminates the needle-like artifacts and enhances details, achieving more photorealistic rendering.
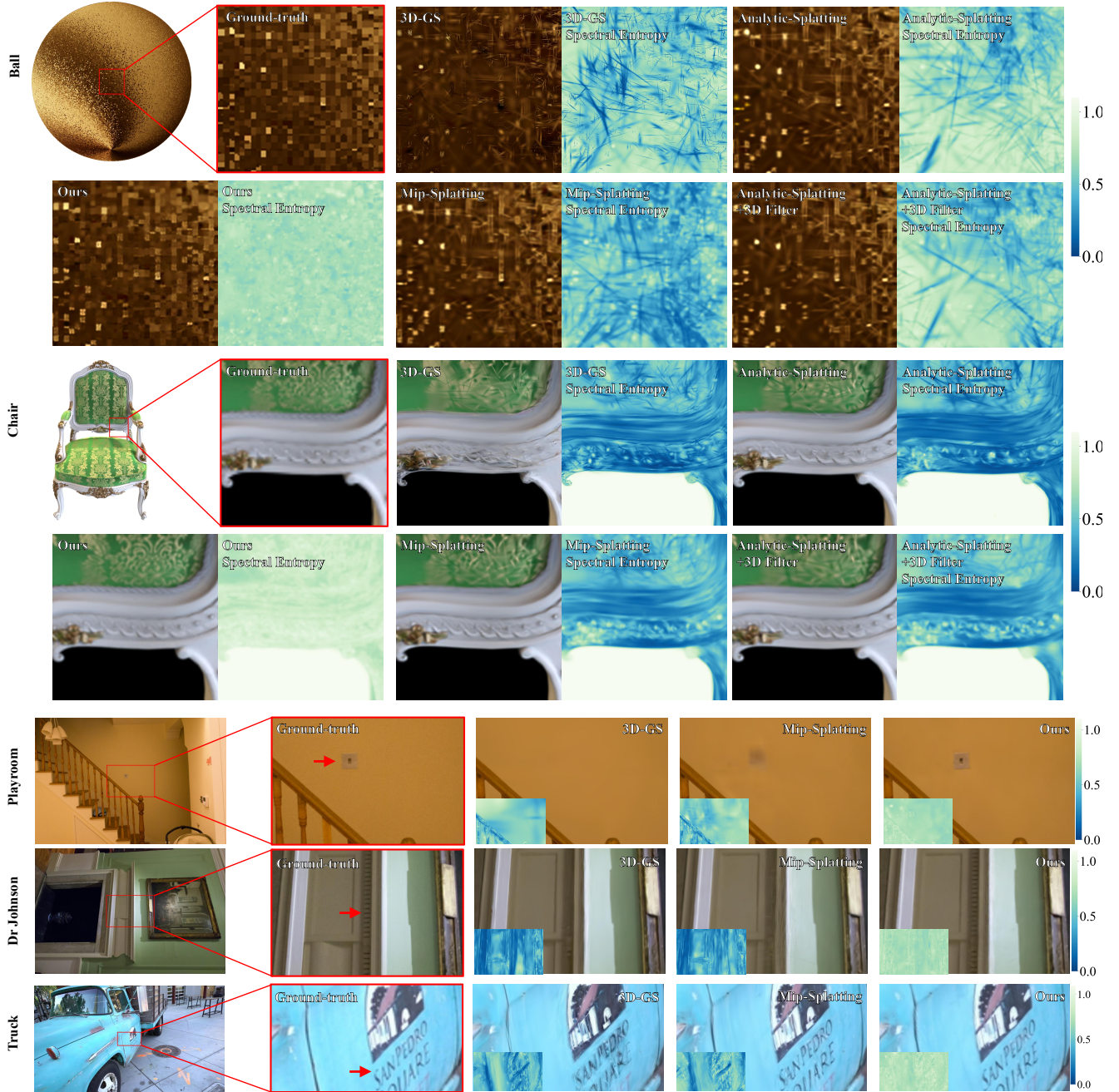
Figure 11. We show comparisons of our method to previous methods and the corresponding ground truth images from held-out test views. Additionally, We visualize the spectral entropy maps of 3D Gaussians after optimization. Bluer regions indicate lower spectral entropy, with more needle-like degraded Gaussians, while greener regions represent higher spectral entropy, without noticeable needle-like artifacts.
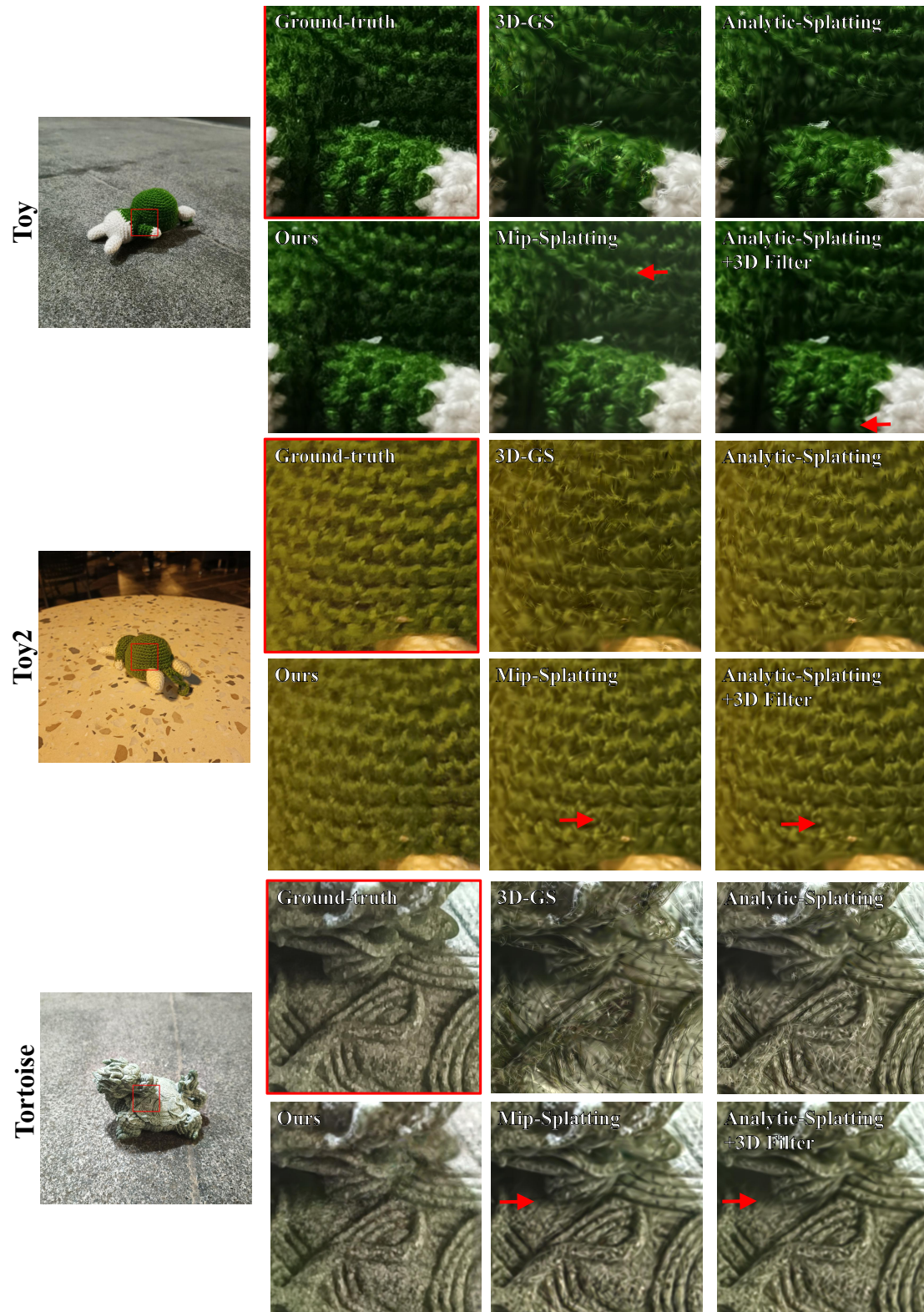
Figure 12. We show comparisons of our method to previous methods and the corresponding ground truth images from held-out test views. Differences in quality highlighted by arrows/insets.