Frequency-Aware Gaussian Splatting Decomposition

Yishai Lavi* yishailavi@mail.tau.ac.il
Leo Segre* leosegre@mail.tau.ac.il
Shai Avidan avidan@eng.tau.ac.il
Tel Aviv University
https://yishailavi.github.io/nerfstudio_lap/

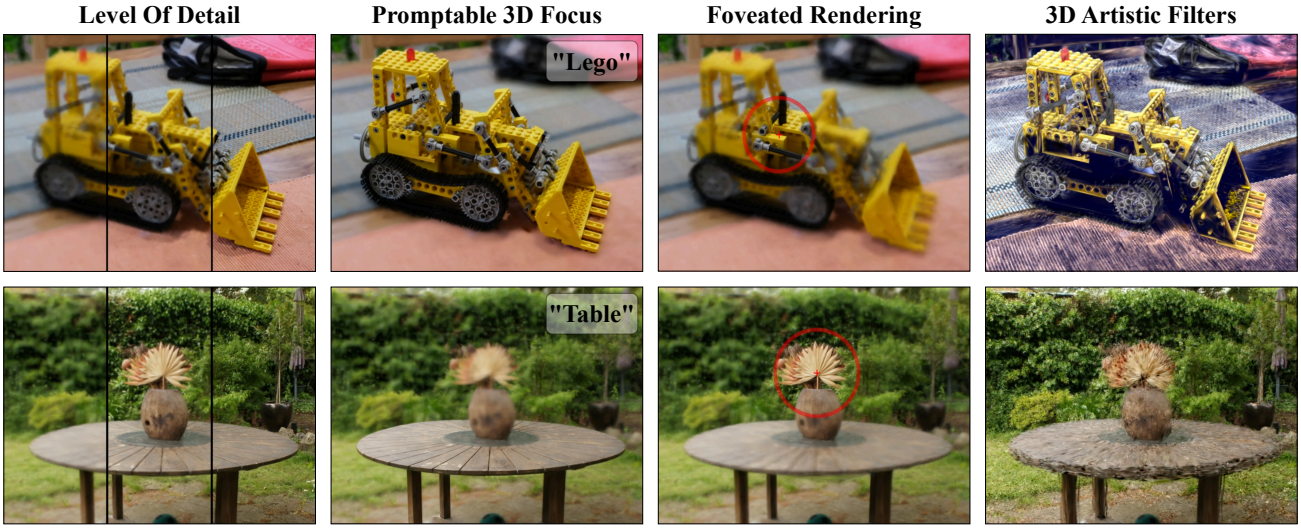

Figure 1. **Frequency-Aware Decomposition for 3D Gaussian Splatting:** We introduce a frequency-aware decomposition for 3D Gaussian Splatting (3D-GS). The decomposition assigns different Gaussians to different levels of the Laplacian Pyramid of the input images. This enables Level of Detail (LOD) rendering, Promptable 3D focus (the text prompt directs the model which object to keep in focus), Foveated rendering, and 3D artistic filters. Our method achieves the highest PSNR, SSIM, and rendering speed among all LOD-capable baselines.

Abstract

3D Gaussian Splatting (3D-GS) enables efficient novel view synthesis, but treats all frequencies uniformly, making it difficult to separate coarse structure from fine detail. Recent works have started to exploit frequency signals, but lack explicit frequency decomposition of the 3D representation itself. We propose a frequency-aware decomposition that organizes 3D Gaussians into groups corresponding to Laplacian-pyramid subbands of the input images. Each group is trained with spatial frequency regularization to confine it to its target frequency, while higher-frequency bands use signed residual colors to capture fine details that may be missed by lower-frequency reconstructions. A progressive coarse-to-fine training schedule stabilizes the decomposition.

Our method achieves state-of-the-art reconstruction quality and rendering speed among all LOD-capable methods. In addition to improved interpretability, our method enables dynamic level-of-detail rendering, progressive streaming, foveated rendering, promptable 3D focus, and artistic filtering. Our code will be made publicly available.

1. Introduction

3D Gaussian Splatting (3D-GS) [9] is an explicit, rasterization-based scene representation that delivers real-time, photorealistic view synthesis. But it lacks a notion of frequency: the millions of Gaussians are an unstructured pool, and beyond heuristics, there is no principled way to say which Gaussians carry low-frequency structure and which encode fine detail. Recent works have begun to exploit frequencies for improved reconstruction quality - for example, FreGS [36] uses progressive frequency regularization to address over-reconstruction issues during densifi-

*Equal contribution.

cation. However, these approaches lack explicit frequency decomposition of the 3D representation itself, making it difficult to perform tasks such as level-of-detail rendering, efficient streaming, and control over different frequency components. Our goal is precisely to introduce this missing structural organization, which enables the above applications and more, as demonstrated in Figure 1 - **Level of Detail**: render the scene at a chosen level of detail for progressive rendering and streaming. **Promptable 3D Focus**: keep selected objects sharp while blurring the rest based on a text prompt, in a 3D-consistent and interactive manner. **Foveated Rendering**: reduce detail in the peripheral area, increasing rendering speed. **3D Artistic Filters**: apply different artistic styles directly to the 3D scene.

We realize this structural organization by explicitly partitioning the Gaussians into a small number of *groups* that correspond to specific subbands in the input images, leveraging the 2D frequency bands of the training images to define these groups. We implement this through a progressive, coarse-to-fine training scheme. First, we optimize a base level of Gaussians to explain the low-frequency content of the images; we then introduce a new set of Gaussians for the next band and jointly optimize both levels, while constraining the earlier level to remain low pass using a frequency regularizer. This forces the newly added Gaussians to complement the reconstruction by capturing the missing higher-frequency residuals. Higher bands are added in the same manner. To model residuals faithfully, Gaussians in non-base levels use signed residual colors (RGB values can be positive or negative), allowing them to add or subtract color w.r.t. the accumulated rendering. The end result is a frequency-aware hierarchy in which each group renders a specific Laplacian-like subband, and any prefix of the hierarchy produces a valid, lower-detail reconstruction.

Using 2D image frequencies as a proxy is a deliberate design choice: it provides a stable and well-studied frequency basis, avoids the complexity of defining frequency for anisotropic splats that project differently across views, and, when enforced consistently across all views, yields a coherent segmentation of the underlying 3D content.

Our decomposition method achieves **state-of-the-art reconstruction metrics among level-of-detail and frequency-aware 3D-GS methods** on the standard benchmarks (*Mip-NeRF360*, *Tanks&Temples*, and *Deep Blending*) while preserving real-time rendering.

In short, we reframe level-of-detail not as mere hierarchical sparsification, but as *frequency segmentation of the representation itself*. Our contributions are: (i) a frequency-aware decomposition of 3D-GS driven by image-space Laplacian bands, (ii) a progressive training strategy with signed residual colors and band-separating regularization, and (iii) comprehensive experiments demonstrating SOTA quality among LOD/frequency-aware methods together with diverse, frequency-driven applications.

2. Related Work

2.1. Novel View Synthesis

Novel view synthesis aims to generate photorealistic images from previously unseen viewpoints, typically given only sparse reference views. Early approaches primarily relied on explicit geometry or depth-based warping, such as layered depth images [24], view interpolation [6], and light field modeling [13]. Later methods incorporated learned priors and convolutional neural networks to predict novel viewpoints directly from input images [21, 38]. A significant breakthrough came with Neural Radiance Fields (NeRF) [19], which represent scenes as continuous volumetric functions parameterized by multi-layer perceptrons, enabling high-quality view synthesis via volume rendering. NeRF has since inspired a series of high-profile successors [1, 3, 20] that extend the original framework. However, the implicit MLP representation common to these models still imposes a non-trivial computational load. Addressing efficiency from a different angle, 3D Gaussian Splatting (3D-GS) [9] replaces the network with explicit oriented Gaussians and a fast differentiable rasterizer, markedly reducing overhead while preserving high-quality rendering.

2.2. Frequency-Aware Representations

Many classical vision and graphics methods rely on frequency decompositions, such as Laplacian pyramids [5], discrete wavelet transforms [16], and Fourier-domain filtering, enabling multiscale analysis. [33] introduce Polynomial Neural Fields (PNFs) and demonstrate frequency-based manipulation of neural fields using Fourier PNFs. Neural rendering methods integrate similar concepts using Fourier-feature embeddings [28], progressive-frequency training [39], frequency-aware NeRFs for enhancing fine detail [37] and wavelet-based NeRF architectures [11, 31].

Within 3DGS, recent methods have started to exploit frequency signals. Mip-Splatting [34] enforces alias-free rendering via analytic Gaussian filtering based on sampling rate limits. GES [7] additionally considers frequency aware loss using DoG. [35] links Gaussian scale and density to underlying frequency content, and [26] extends this notion to dynamic scenes. Similarly to our approach, FreGS [36] utilizes a Fourier-based loss, but applies it to guide Gaussian densification in a coarse-to-fine manner to mitigate over-reconstruction. In contrast, we use it to group Gaussians into distinct frequency subbands of the scene.

2.3. Level of Detail in 3D Gaussian Splatting

Recent works have extended 3D Gaussian Splatting with level-of-detail mechanisms. Multi-Scale 3D-GS [32] introduces multiple Gaussian scales to reduce aliasing artifacts

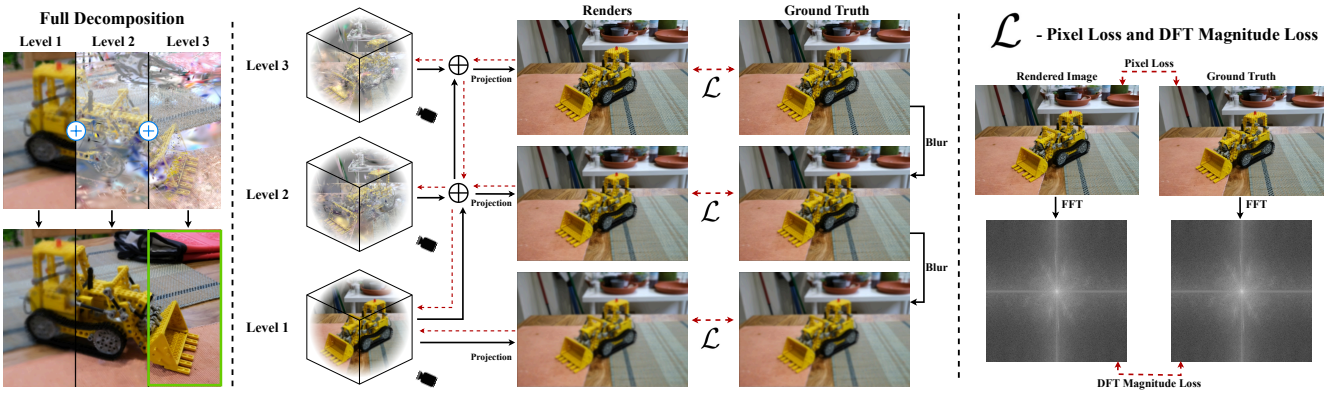

Figure 2. **Frequency-Aware 3D Gaussian Splatting Overview:** We decompose 3D Gaussians into frequency-specific groups to construct a multi-scale representation. (Left) Each Gaussian group targets a specific frequency band: Level 1 captures low-frequency structure, while Levels 2-3 add mid and high-frequency details through signed residual colors. (Middle) During training, each accumulated level (L1, L1+L2, L1+L2+L3) is supervised against appropriately blurred ground truth images, with red dashed lines showing gradient flow that constrains lower levels to remain frequency-limited while higher levels capture residual details. (Right) Our dual loss combines standard pixel loss with DFT magnitude loss in frequency space, ensuring each level contributes meaningful frequency content and preventing frequency leakage between bands.

low resolutions, while OCTree-GS [22] organizes Gaussians within an octree structure [17] to accelerate rendering across multiple scales and viewpoints. Other methods, including [10, 14, 15, 30], organize large-scale scenes using hierarchical data structures for efficient rendering, but focus primarily on scalability and massive scene handling rather than explicit frequency or detail separation.

LapisGS [27] addresses adaptive streaming by training Gaussian layers sequentially on progressively higher-resolution inputs, allowing opacity changes between layers to enforce a coarse-to-fine structure. FLoD [23] organizes Gaussians into discrete levels according to their spatial extent, enabling customized resolution control through sequential optimization. Both approaches freeze most of the parameters of previously optimized levels before introducing new ones, relying on heuristic measures: resolution or spatial extent, to define their level-of-detail separation.

In contrast, our method explicitly defines Gaussian groups based on the frequency content of input images, jointly training all frequency groups progressively rather than sequentially freezing lower levels. We employ dedicated frequency-domain regularization to enforce clear spectral boundaries between these groups. Additionally, higher-frequency Gaussians use signed residual colors to encode fine details efficiently by subtracting colors from coarser frequency levels, similar to a Laplacian pyramid, thereby reducing the total number of Gaussians required.

3. Method

Our method is based on the foundational framework of 3D Gaussian splatting [9]. By extending this method, we introduce a mechanism to group Gaussians based on their contribution to different frequency bands of the images. These groups, referred to as "levels", enable a hierarchical representation akin to the Image Laplacian Pyramid [5] in classical computer vision. Each frequency level captures progressively finer details, from diffuse colors and general structure to high-frequency variations, optimizing both the rendering process and frequency-aware modeling.

To render a scene at a chosen frequency level, all Gaussians up to that frequency level are rendered in a single render pass. In particular, rendering the full-resolution image requires simply rendering all Gaussians in one pass, just as in standard 3D Gaussian splatting.

3.1. Preliminary: 3D Gaussian Splatting

3D Gaussian Splatting (3D-GS) [9] represents a scene as a collection of anisotropic 3D Gaussians $\{G_i\}_{i=1}^{N}$, providing a structured, explicit alternative to neural-based representations.

The initialization of 3D-GS typically starts from a point cloud $\{x_i\}_{i=1}^{N}$ obtained via Structure-from-Motion (SfM). Each point x_i in this cloud is assigned a 3D Gaussian representation G_i, which is defined by a mean position μ_i and a covariance matrix Σ_i. The covariance is decomposed as:

$$\Sigma_i = \mathcal{R}\mathcal{S}\mathcal{S}^{\mathcal{T}}\mathcal{R}^{\mathcal{T}}, \tag{1}$$

where $\mathcal{S}$ represents a scaling matrix controlling the extent of the Gaussian, and $\mathcal{R}$ is a rotation matrix ensuring Σ_i remains positive semi-definite. Additionally, each Gaussian stores appearance information through spherical harmonics

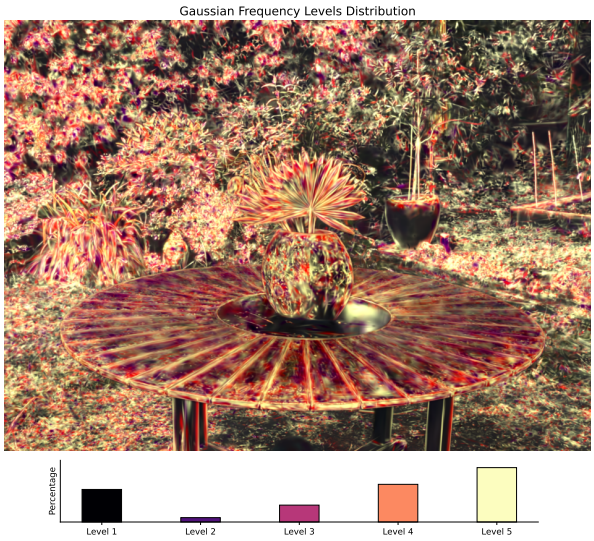

Figure 3. Visualization of the Gaussian frequency levels distribution in the scene. The colormap represents different frequency levels, where lower levels correspond to coarse scene structures and higher levels capture finer details.

and an opacity value α_i, which contributes to the blending process.

Rendering in 3D-GS involves projecting the 3D Gaussians into 2D space, converting them into 2D Gaussians G'_i. A differentiable rasterization process sorts these 2D Gaussians by depth and blends them according to their opacity. The final color $C(x)$ at a pixel position x is computed as:

$$C(x) = \sum_{j \in M} c_j \sigma_j \prod_{k=1}^{j-1} (1 - \sigma_k), \tag{2}$$

where $\sigma_j = \alpha_j G'_j(x)$, M represents the ordered set of Gaussians contributing to x, and c_j denotes the corresponding color. This differentiable rendering pipeline enables joint optimization of all Gaussian attributes during training, facilitating high-quality view synthesis.

3.2. Gaussian Level Grouping

Fig. 2 gives an overview of our method. We progressively group 3D Gaussians based on their frequency contribution, ensuring that the accumulated rendering of the first k groups matches a corresponding low-pass filtered version of the input. This guarantees that each individual level aligns with the respective frequency band in the Laplacian pyramid of the input images. To enforce frequency-aware consistency, we introduce a dedicated frequency regularization term in the loss function.

To hierarchically represent the scene by frequency levels, our method divides the Gaussians into groups, where each group corresponds to a specific band of frequencies. The scene is constructed progressively, starting from low frequencies and incorporating higher frequencies over time by adding additional groups. In practice, we modulate each Gaussian G_i with an integer, non-learnable parameter l_i, termed "level", which denotes the frequency group to which the Gaussian belongs. We initialize and update this parameter as follows:

Initialization: Gaussians of the first level are initialized using a Structure-from-Motion (SfM) point cloud, as in the original 3D Gaussian Splatting paper. All Gaussians are assigned an initial level of 1, representing the base frequency group.

Adaptive density control: During training, Gaussians can be split or cloned using the adaptive density control technique from the original method. The newly created Gaussians inherit the same level as their parent.

Progressive Level Introduction: Every K training steps, we introduce a new level by duplicating all Gaussians and assigning them the next frequency level; almost all duplicates are removed during training (see Supplementary for Gaussian count dynamics).

Fig. 3 shows the decomposition of 3D Gaussians to different groups that correspond to different levels of the Laplacian pyramid. As can be seen, the base level requires a considerable amount of Gaussians, while the intermediate low levels require far less, except for the last two levels, which correspond to high frequencies.

3.3. Residual Color Gaussians

Traditional Gaussian Splatting techniques rely on the accumulation of Gaussian colors to render scenes. However, in our method, higher-level Gaussians contribute details to lower-level Gaussians, such as textures and edges, which cannot be achieved efficiently by simply "adding color". To address this limitation, we introduce Residual Color Gaussians, allowing for both positive and negative color values. This approach compensates for lower-level contributions and facilitates the addition of high-frequency details, achieving a more accurate and detailed hierarchical representation.

Level 1 Gaussians retain colors in the range $[0, 1]$, consistent with the original 3D-GS paper, by using view-dependent spherical harmonic color generation; Given a Gaussian G_i and viewing direction d, the rendered color of the Gaussian is computed as:

$$C_i = SH(G_i, d) \tag{3}$$

Inspired by the Laplacian pyramid [5] in image processing, where residual levels can take both positive and negative values to represent higher-frequency sub-bands, we multiply the color range of all Gaussians at higher levels by 2 and shift down by 1, resulting in a range of $[-1, 1]$:

$$C_i = (SH(G_i, d) \cdot 2) - 1 \tag{4}$$

This simple adjustment is applied before alpha blending and enables subtraction of colors between positive and negative colored Gaussians. The final rendered color is clamped to the range $[0, 1]$. This approach effectively captures high-frequency details, analogously to the functionality of residual levels in the Laplacian pyramid.

3.4. Progressive Rendering of Frequency Bands

As stated above, at inference time, one can render all Gaussians in the scene at once to get the full resolution rendered image. Recall, in the image Laplacian Pyramid, one can reconstruct a Gaussian Pyramid by up-sampling and adding each Laplacian level with its successor, resulting in images with progressively finer details. Similarly, to regulate the different frequency levels in our method during the training stage, we render each level of the Gaussian Pyramid separately by combining all Gaussian groups up to the corresponding frequency level.

Importantly, all images are rendered at the same (full) resolution. This ensures that we can accurately regulate the frequencies of the Gaussians in each accumulated level without inadvertently removing valid high-frequency details due to subsampling. Subsampling would remove or alias frequency components above the Nyquist limit [25], making it impossible to determine whether Gaussians truly represent low-frequency information or if their details were simply lost during down-sampling. Rendering at full resolution allows us to preserve and regulate the true frequency content of each level accurately.

Specifically, for a pyramid with L levels, rendering the k-th frequency level, where $1 \leq k \leq L$, we render all Gaussians G_i that satisfy $l_i \leq k$, resulting in the following sequence of rendered images:

$$\{\hat{I}_k\}_{k=1}^{L}, \quad \hat{I}_k \in \mathbb{R}^{H \times W \times 3}, \quad 1 \leq k \leq L \tag{5}$$

3.5. Frequency Regulation and Losses Design

As mentioned earlier, we apply frequency regulation at full resolution. To achieve this, we utilize bilinear interpolation with a scale factor of 0.5 for downsampling, denoted as $\downarrow_2$, and bilinear interpolation with a scale factor of 2 for upsampling, denoted as $\uparrow_2$.

Hence, given a GT image $I \in \mathbb{R}^{H \times W \times 3}$, in order to regulate the frequency band of the k-th level $\hat{I}_k$, we construct the low-pass filtered image with:

$$I_k = ((I) \downarrow_2^{L-k}) \uparrow_2^{L-k} \tag{6}$$

This way, we obtain the frequency spectrum of the GT image as it was subsampled $L - k$ times, but we keep it in its original resolution.

To regulate the frequency spectrum, we introduce the frequency magnitude discrepancy term, which helps maintain the desired frequencies in $\hat{I}_k$. Given an image I_k, let $F_k(u, v)$ denote its Discrete Fourier Transform (DFT) at frequency coordinate (u, v). Using the frequency magnitudes $|F_k(u, v)|$, we define the frequency magnitude discrepancy term between two images as:

$$d_{\text{DFT}}(I_k, \hat{I}_k) = \frac{1}{UV} \sum_{u=0}^{U-1} \sum_{v=0}^{V-1} \left\| |F_k(u, v)| - |\hat{F}_k(u, v)| \right\| \tag{7}$$

where UV represents the total number of frequency components and $F_k(u, v)$ and $\hat{F}_k(u, v)$ denote the frequency coefficients of images I_k and $\hat{I}_k$, respectively.

Employing Eq. (7), we will define our frequency magnitude loss by applying the frequency discrepancy term on all levels, except for the last one (as we do not desire to limit its frequency band):

$$\mathcal{L}_{\text{DFT}} = \sum_{k=1}^{L-1} d_{\text{DFT}}(I_k, \hat{I}_k) \tag{8}$$

Furthermore, we adopt a spatial image discrepancy term similar to the one used in the original 3D-GS paper. Specifically, we regulate the k-th frequency level at its corresponding low resolution. To prevent aliasing, we employ the REDUCE operation, as originally introduced in the Laplacian Pyramid framework [5].

Given an image I and a 2D Gaussian kernel $\mathcal{G}$, the REDUCE operation is defined as a convolution with the kernel followed by subsampling every second row and column:

$$\text{REDUCE}(I) = (I * \mathcal{G})_{::2} \tag{9}$$

To regulate the k-th frequency level, we apply Eq. (9) $L - k$ times to both I_k and $\hat{I}_k$, yielding:

$$I'_k = \text{REDUCE}^{L-k}(I_k), \quad \hat{I}'_k = \text{REDUCE}^{L-k}(\hat{I}_k) \tag{10}$$

Using the processed images from Eq. (10), we define our spatial image discrepancy term as:

$$d_{\text{IM}}(I_k, \hat{I}_k) = (1 - \lambda_{\text{SSIM}})|I'_k - \hat{I}'_k| + \lambda_{\text{SSIM}}\, d_{\text{D-SSIM}}(I'_k, \hat{I}'_k) \tag{11}$$

where d_{IM} combines a pixel-wise difference term with a structural similarity (SSIM) discrepancy $d_{\text{D-SSIM}}$, weighted by λ_{SSIM}. Utilizing Eq. (11), the spatial image loss is computed by summing the spatial image discrepancy terms over the rendered and ground truth (GT) levels:

$$\mathcal{L}_{\text{IM}} = \sum_{k=1}^{L} \lambda_{d_k} d_{\text{IM}}(I_k, \hat{I}_k) \tag{12}$$

Finally, taking Eq. (12) and Eq. (8), our final loss function is defined as:

$$\mathcal{L} = \lambda_{\text{IM}} \mathcal{L}_{\text{IM}} + \lambda_{\text{DFT}} \mathcal{L}_{\text{DFT}} \tag{13}$$

Method	Mip-NeRF360					Tanks & Temples					Deep Blending				
	SSIM↑	PSNR↑	LPIPS↓	Gauss↓	FPS↑	SSIM↑	PSNR↑	LPIPS↓	Gauss↓	FPS↑	SSIM↑	PSNR↑	LPIPS↓	Gauss↓	FPS↑
3DGS	0.822	27.76	0.177	2673263	124.4	0.849	23.73	0.171	1571073	120.6	0.903	29.75	0.241	2465386	89.8
Lapis-GS**	0.753	26.32	0.248	6252724	62.7	0.811	23.27	0.215	2864263	63.70	0.892	28.94	0.222	3874530	48.1
FreGS*	0.826	27.85	0.209	–	–	0.849	23.96	0.178	–	–	0.904	29.93	0.240	–	–
FLOD	0.832	28.20	0.174	2320942	140.3	0.845	24.27	0.188	1314046	142.2	0.898	29.24	0.253	1783253	110.2
MSGS*	–	27.39	0.155	–	109.9	–	23.46	0.111	–	131.6	–	29.70	0.096	–	135.1
OCTree-GS	0.832	28.20	0.168	363001	176.3	0.822	23.79	0.224	131974	202.4	0.904	30.21	0.251	181506	174.4
Ours 5LOD	0.867	29.29	0.166	940257	190.6	0.878	24.21	0.111	898358	177.1	0.912	30.29	0.160	704216	181.0
Ours 3LOD	0.874	29.65	0.156	789182	224.6	0.879	24.40	0.120	493681	241.3	0.914	30.30	0.173	707819	186.0

Table 1. Comparison between our method and competing methods across three datasets. Cell colors indicate first (red), second (orange), and third (yellow) place for each metric. Our method ranks in the top 3 across all metrics and datasets, achieving the best SSIM, PSNR, and FPS in every case. * Indicates results taken from the original paper, not re-evaluated locally. ** Lapis-GS was trained with $\lambda_{\text{SSIM}} = 0.2$ following their suggestion to overcome GPU OOM (OOM on a 48GB Nvidia A6000).

where λ_{IM} and λ_{DFT} are weighting factors balancing the contributions of the spatial and frequency terms, respectively.

4. Results

We evaluate our method across a comprehensive set of metrics and benchmarks, aiming to validate both reconstruction quality and real-time rendering efficiency. Specifically, we report SSIM, PSNR, LPIPS, Gaussian count, and FPS, and compare against prior 3D Gaussian Splatting methods that support level-of-detail (LOD) rendering. For fair comparison, we re-trained and evaluated baselines using the same evaluation protocol. In addition, we provide per-level progressive rendering analysis and demonstrate several applications uniquely enabled by our frequency-aware representation, including streaming, foveated rendering, and promptable 3D focus.

Data and Evaluations: Following the 3D-GS dataset configuration, we evaluated our model across all nine scenes of the Mip-NeRF 360 dataset [2], the Truck and Train scenes from the Tanks & Temples dataset [12], and the Playroom and Dr. Johnson scenes from the Deep Blending dataset [8]. For evaluation, we adopt the 3D-GS protocol, selecting every eighth image as part of the test set.

4.1. Implementation Details

Our implementation is based on the NerfStudio SplatFacto model [29]. To ensure stable optimization, we begin training with low-resolution images and progressively double the training resolution each time a new frequency level is introduced. A new frequency level is added every $K = 2500$ steps. Upon adding a level, all optimizers and their schedulers are reinitialized, and Gaussian refinement is temporarily disabled for 300 steps. Throughout training, all levels are jointly optimized. Following 3D-GS, we set $\lambda_{\text{SSIM}} = 0.2$. Additionally, we empirically set $\lambda_{d_k} = 0.1$ for $k < L$, and $\lambda_{d_k} = 1$ for $k = L$. The final loss function is weighted as $\lambda_{IM} = 1$ and $\lambda_{DFT} = 0.001$.

All scenes are trained for 30,000 steps, with Gaussian refinement enabled in our model for the entire training process. Optimization is performed using SplatFacto's optimizers, and all experiments are conducted on a single NVIDIA RTX A5000 GPU. For our model, we incorporate the view-dependent anti-aliased opacity term proposed in Mip-Splatting [34].

4.2. Comparisons with the State-of-the-Art

We compare our method against recent 3D Gaussian Splatting techniques, including FLOD [23], OCTree-GS [22], MSGS [32], FreGS [36], and Lapis-GS [27]. While most of these methods support explicit level-of-detail (LOD) rendering, FreGS is included for its use of frequency-aware modeling, even though it does not offer progressive rendering. We also include 3DGS [9] as a baseline. Wherever possible, we re-trained or re-evaluated competing methods locally using the same evaluation protocol.

We evaluated two variants of our method: **Ours 3LOD**, which uses three frequency levels, and **Ours 5LOD**, which uses five. Both are trained with the same budget of 30,000 steps. These configurations allow fine-grained trade-offs between quality and performance, and demonstrate the flexibility of our frequency decomposition.

As shown in Table 1, both our 3LOD and 5LOD configurations consistently rank among the top three methods across all metrics and datasets. Notably, **Ours 3LOD achieves the highest SSIM, PSNR, and FPS in every dataset**, establishing it as the best performing method in terms of both reconstruction quality and rendering speed. Our models maintain highly competitive LPIPS scores despite using significantly fewer Gaussians than dense baselines like 3DGS and FLOD.

4.3. Progressive Rendering

Progressive rendering on resource-constrained devices allows users to render only low-frequency levels for high frame rates, then progressively add higher-frequency details as computational budget allows. For streaming appli-

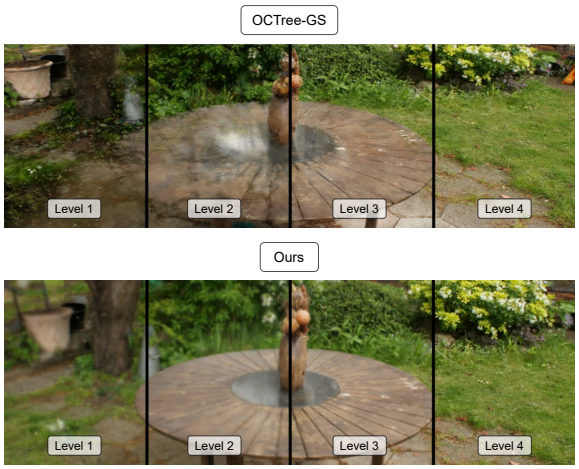

Figure 4. **Progressive Rendering Consistency:** Comparing between OCTree-GS (top) and our method (bottom) across different frequency levels (left to right). In OCTree-GS, lower levels suffer from missing geometric details, which are recovered as more levels are added. In contrast, our representation preserves the scene's geometry across all levels, with fine details progressively added, making it more suitable for frequency-aware applications.

cations, low-frequency Gaussians can be transmitted first for immediate rendering, with higher-frequency content streamed progressively to refine quality over time.

Our frequency-based decomposition naturally enables progressive rendering where each level provides a meaningful, artifact-free approximation of the scene. Unlike spatial LOD methods that often introduce noticeable artifacts (Figure 4), our frequency hierarchy ensures smooth quality transitions and coherent approximations at all levels.

As shown in Figure 5, our method consistently achieves the highest quality (PSNR) across all LODs while surpassing baselines in rendering speed from Level 3 onward.

4.4. Faster Geometry Interaction

One important application of our frequency decomposition method is fast geometry interaction. Since low-frequency Gaussians define the general structure of the scene, many geometric operations can be efficiently approximated by operating only on this subset. Figure 4 shows that we have structural integrity in lower levels.

We compare the performance of nearest-neighbor search on a randomly sampled 100K point cloud with a scale of 1, using either the full Gaussian set or only the low-frequency subset. Querying the KDTree [4] built from the full model required **1.95 seconds**, whereas querying the KDTree constructed from only the low-frequency Gaussians reduced this to **0.88 seconds**, achieving a $2.21\times$ speedup. Despite this simplification, the median discrepancy with respect to the KDTree operating on the full model was 0.0006, and the mean discrepancy was only 0.0018, corresponding to

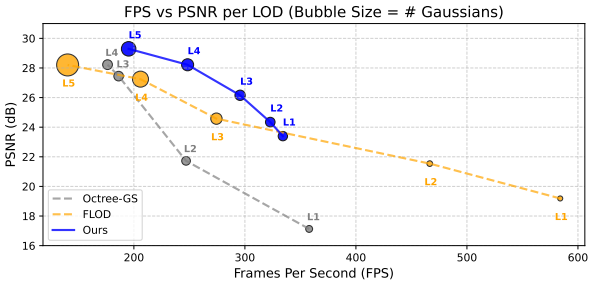

Figure 5. **Progressive Rendering and Streaming:** Comparison of our method, FLOD, and OCTree-GS across different levels of detail. The plot shows the trade-off between rendering speed (FPS) and reconstruction quality (PSNR), where each point corresponds to a specific LOD level and bubble size reflects the number of Gaussians. Results are averaged over all Mip-NeRF 360 scenes. Our method demonstrating superior performance in both quality and speed in the highest three LODs.

a 0.18% error relative to the point cloud scale. This result suggests that many geometry-based interactions, such as collision detection and spatial querying, can be significantly accelerated using only the low-frequency representation, with minimal approximation error. The point cloud and the measured scene can be seen in Fig. 6a.

4.5. Foveated Rendering

Our representation enables efficient foveated rendering by dynamically adjusting the contribution of different frequency levels based on eye-tracking input, allowing faster rendering of the region of interest (ROI) relative to the entire image. This is particularly beneficial for AR/VR applications, where optimizing computational resources without sacrificing perceptual quality is critical.

We modulate the contribution of 3D Gaussians at different frequency levels according to gaze-based weighting, discarding those whose effective opacity is below a threshold. As illustrated in Figure 6b, this strategy preserves high-frequency detail in the area where the user is looking while smoothly reducing detail in peripheral regions. Leveraging our multi-resolution representation, we achieve a **40% improvement in FPS** at 4K resolution, significantly reducing rendering costs without sacrificing perceptual fidelity. Further implementation details and additional examples are provided in the supplementary material.

4.6. Promptable 3D Focus

Figure 7 demonstrates our text-prompted frequency-aware 3D focus, where users specify objects via natural language to achieve object-focused rendering that preserves high-frequency detail for targets while blurring the background. This overcomes limitations of 2D mask-based methods, that suffer from artifacts and poor multi-view consistency.

(a) Efficient Geometry Interaction (b) Foveated Rendering (c) Artistic Visualization

Figure 6. **Applications of Frequency-Based Gaussian Rendering:** (a) Efficient geometry interaction using low-frequency levels for faster processing. Closer points are brighter. Using only low frequency Gaussians (left) is more than twice as fast as using all Gaussians (right) with negligible error. (b) Foveated rendering, improving FPS performance using eye-tracking. (c) Frequency-based artistic effects, demonstrating an X-ray-like visualization. Please zoom in for details.

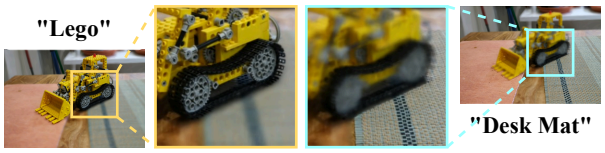

Figure 7. **Promptable 3D Focus:** Our method selectively renders only low-frequency Gaussians in the scene, except for Gaussians that belong to the prompted object. The left scene got the prompt "Lego", while the right scene got "Desk Mat".

We leverage Lang-Segment-Anything [18] for object identification. Given a text prompt, we generate masks across N training images, then identify corresponding Gaussians by projecting centers and computing mask overlap ratios. We preserve Gaussians that either (1) belong to the base frequency level or (2) exceed the mask threshold, producing 3D-consistent views with sharp objects and blurred backgrounds. This approach offers both visual and efficiency gains: fewer Gaussians are rendered, enabling compact object-focused representations.

4.7. Artistic 3D Filters

Our representation also enables consistent 3D artistic filtering by selectively modifying the attributes of Gaussians at different frequency levels. By altering specific groups of Gaussians, we can achieve artistic effects that remain coherent across different viewpoints. To demonstrate that, one can use our representation to create an X-ray-style filter, where low-frequency Gaussians are assigned a constant color of dark blue to simulate an underlying structure, while high-frequency Gaussians have their RGB values increased by 0.2, enhancing the visibility of fine details. As demonstrated in Figure 6c, this manipulation creates a 3D-consistent X-ray effect, preserving structural integrity across views. Additional artistic effects on different scenes are showcased in the supplementary material.

4.8. Ablation Studies

To better understand the contributions of different components in our approach, we conducted a series of ablation studies. These experiments assess the impact of key design choices on model efficiency and rendering quality. All ablation studies were performed on the Kitchen scene from the Mip-NeRF 360 dataset, allowing for consistent evaluation across different configurations. Specifically, we examined (i) *Residual Color Gaussians*, (ii) *Image Loss on Lower Frequency Levels*, and (iii) *Frequency Magnitude Loss*, revealing their respective influences on final rendering quality, memory requirements, and hierarchical decomposition. For a detailed discussion of each ablation, we refer the reader to the supplementary material.

5. Conclusions

We introduce frequency decomposition into 3D Gaussian Splatting, inspired by Laplacian pyramids in image processing, to provide structured control over frequency components in a scene. By organizing Gaussians into hierarchical subbands, incorporating Residual Color Gaussians, and leveraging progressive training, our method enhances interpretability and enables efficient level-of-detail management. Quantitatively, our method achieves state-of-the-art reconstruction quality and rendering speed among all level-of-detail extensions of 3D Gaussian Splatting, as demonstrated on standard benchmarks.

Additionally, this approach paves the way for applications such as promptable 3D focus and artistic filtering, allowing precise manipulation of frequency bands for creative effects and content-aware modifications. Furthermore, it supports practical scenarios like foveated rendering, progressive rendering, streaming, and fast geometry interaction, making real-time performance more adaptable across devices. These capabilities establish our method as both a versatile representation and a strong baseline for frequency-aware 3D scene modeling.

Acknowledgment: Part of this research was supported by ISF grant 2132/23.

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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5855–5864, 2021. 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. 6

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19697–19705, 2023. 2

[4] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975. 7

[5] Peter J Burt and Edward H Adelson. The laplacian pyramid as a compact image code. In *Readings in computer vision*, pages 671–679. Elsevier, 1987. 2, 3, 4, 5

[6] Shenchang Eric Chen and Lance Williams. *View Interpolation for Image Synthesis*. Association for Computing Machinery, New York, NY, USA, 1 edition, 2023. 2

[7] Abdullah Hamdi, Luke Melas-Kyriazi, Jinjie Mai, Guocheng Qian, Ruoshi Liu, Carl Vondrick, Bernard Ghanem, and Andrea Vedaldi. Ges: Generalized exponential splatting for efficient radiance field rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19812–19822, 2024. 2

[8] 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

[9] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 3, 6

[10] Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A hierarchical 3d gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 3

[11] Rajaei Khatib and Raja Giryes. Trinerflet: A wavelet based triplane nerf representation. In *European Conference on Computer Vision*, pages 358–374. Springer, 2024. 2

[12] 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

[13] Marc Levoy and Pat Hanrahan. Light field rendering. page 31–42, New York, NY, USA, 1996. Association for Computing Machinery. 2

[14] Yang Liu, Chuanchen Luo, Lue Fan, Naiyan Wang, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. In *European Conference on Computer Vision*, pages 265–282. Springer, 2024. 3

[15] Yang Liu, Chuanchen Luo, Zhongkai Mao, Junran Peng, and Zhaoxiang Zhang. Citygaussianv2: Efficient and geometrically accurate reconstruction for large-scale scenes. *arXiv preprint arXiv:2411.00771*, 2024. 3

[16] Stephane G Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE transactions on pattern analysis and machine intelligence*, 11(7):674–693, 1989. 2

[17] Donald Meagher. Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147, 1982. 3

[18] Luca Medeiros et al. Lang-segment-anything: Text-promptable segment anything model, 2023. Accessed: 2025-02-01. 8

[19] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[20] 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

[21] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 3500–3509, 2017. 2

[22] 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, 6

[23] Yunji Seo, Young Sun Choi, Hyun Seung Son, and Youngjung Uh. Flod: Integrating flexible level of detail into 3d gaussian splatting for customizable rendering, 2025. 3, 6

[24] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 231–242, 1998. 2

[25] Claude E Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949. 5

[26] Mingwen Shao, Yuanjian Qiao, Kai Zhang, and Lingzhuang Meng. Frequency-aware uncertainty gaussian splatting for dynamic scene reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 31(5):3558–3568, 2025. 2

[27] Yuang Shi, Géraldine Morin, Simone Gasparini, and Wei Tsang Ooi. Lapisgs: Layered progressive 3d gaussian splatting for adaptive streaming, 2025. 3, 6

[28] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020. 2

[29] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–12, 2023. 6

[30] Zipeng Wang and Dan Xu. Pygs: Large-scale scene representation with pyramidal 3d gaussian splatting. *arXiv preprint arXiv:2405.16829*, 2024. 3

[31] Muyu Xu, Fangneng Zhan, Jiahui Zhang, Yingchen Yu, Xiaoqin Zhang, Christian Theobalt, Ling Shao, and Shijian Lu. Wavenerf: Wavelet-based generalizable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18195–18204, 2023. 2

[32] Zhiwen Yan, Weng Fei Low, Yu Chen, and Gim Hee Lee. Multi-scale 3d gaussian splatting for anti-aliased rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20923–20931, 2024. 2, 6

[33] Guandao Yang, Sagie Benaim, Varun Jampani, Kyle Genova, Jonathan Barron, Thomas Funkhouser, Bharath Hariharan, and Serge Belongie. Polynomial neural fields for subband decomposition and manipulation. *Advances in Neural Information Processing Systems*, 35:4401–4415, 2022. 2

[34] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19447–19456, 2024. 2, 6

[35] Zhaojie Zeng, Yuesong Wang, Lili Ju, and Tao Guan. Frequency-aware density control via reparameterization for high-quality rendering of 3d gaussian splatting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9833–9841, 2025. 2

[36] Jiahui Zhang, Fangneng Zhan, Muyu Xu, Shijian Lu, and Eric Xing. Fregs: 3d gaussian splatting with progressive frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21424–21433, 2024. 1, 2, 6

[37] Xiaoyu Zhang, Weihong Pan, Chong Bao, Xiyu Zhang, Xiaojun Xiang, Hanqing Jiang, and Hujun Bao. Lookcloser: Frequency-aware radiance field for tiny-detail scene. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16122–16132, 2025. 2

[38] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European conference on computer vision*, pages 286–301. Springer, 2016. 2

[39] Junyu Zhu, Hao Zhu, Qi Zhang, Fang Zhu, Zhan Ma, and Xun Cao. Pyramid nerf: Frequency guided fast radiance field optimization. *International Journal of Computer Vision*, 131 (10):2649–2664, 2023. 2