KC-3DGS: KURTOSIS-CONSTRAINED GAUSSIAN SPLATTING FOR HIGH-FIDELITY VIEW SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

3D Gaussian Splatting (3DGS) enables real-time novel view synthesis, but with sparse training views it produces floaters, color casts and oversmoothed regions at novel viewpoints. Pixel-space losses (L1, SSIM) do not prevent this: they constrain the total reconstruction error, not how it is distributed across scales. We propose KC-3DGS, which regularizes 3DGS with wavelet-domain statistics of natural images. It combines three terms: (1) a scale-weighted alignment of wavelet coefficients, (2) a supervised kurtosis loss that matches the heavy-tailed statistics of each subband to the ground truth, and (3) a supervised cross-band covariance loss that matches how oriented subbands co-occur. A simple analysis shows why each term is needed. Pixel losses cannot distinguish errors at different scales. Oversmoothing and floaters lower subband kurtosis, which only a supervised kurtosis loss penalizes. Cross-band covariance captures joint structure, such as corners, that per-band statistics miss. Because these statistics pool over image positions, they can supervise renders from pseudo cameras near the training views, where sparse-view artifacts arise; all terms start after densification, leaving the Gaussian set unchanged. With 12 training views on five benchmarks (MipNeRF360, Tanks&Temples, Deep-Blending, WRIVA-ULTRRA, MVImgNet), KC-3DGS improves perceptual metrics (PSNR, SSIM, LPIPS, DreamSim) on scenes, and as a plug-and-play regularizer it also improves three existing sparse-view 3DGS methods by up to 28.8% reduction in DreamSim and 1.59 dB increase in PSNR.

1 INTRODUCTION

Novel-view synthesis (NVS) aims to render photorealistic images of a scene from unseen viewpoints, and underpins applications in VR/AR, robotics, and content creation Mildenhall et al. (2020); Barron et al. (2022). Recent explicit representations, in particular 3D Gaussian Splatting (3DGS), model scenes as sets of anisotropic Gaussians optimized to approximate a radiance field, enabling real-time, high-quality rendering via differentiable rasterization Kerbl et al. (2023). Compared to NeRF-based methods, 3DGS achieves competitive or superior image quality with much faster training and inference and has rapidly become a strong baseline for real-time NVS Mildenhall et al. (2020); Kerbl et al. (2023).

However, in practical settings only a small number of views are often available, making the reconstruction problem severely underconstrained Jain et al. (2021); Niemeyer et al. (2022); Wang et al. (2023). Under such sparse-view supervision, 3DGS suffers from unstable optimization and artifacts Xiong et al. (2025); Li et al. (2024); Zhu et al. (2024); Zhang et al. (2024). Adaptive density control (ADC) can misinterpret incomplete gradients, spawning Gaussians in regions unsupported by true geometry and pruning others prematurely, leading to floaters, oversmoothed blobs, and loss of fine details Kerbl et al. (2023); Li et al. (2024); Zhu et al. (2024). Standard photometric objectives (e.g., L1, L2, SSIM Wang et al. (2004)) penalize per-pixel errors but do not effectively constrain higher-order structure, so models can achieve strong PSNR/SSIM while still producing overly smooth, structurally inconsistent renderings Zhang et al. (2018).

Natural image statistics offer a complementary perspective. When images are decomposed into multi-scale, oriented band-pass components (e.g., via wavelets), the resulting coefficients exhibit sparse, heavy-tailed distributions with characteristic kurtosis patterns that are remarkably consistent across subbands Field (1987); Ruderman (1994); Simoncelli & Olshausen (2001); Mallat (1989);

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

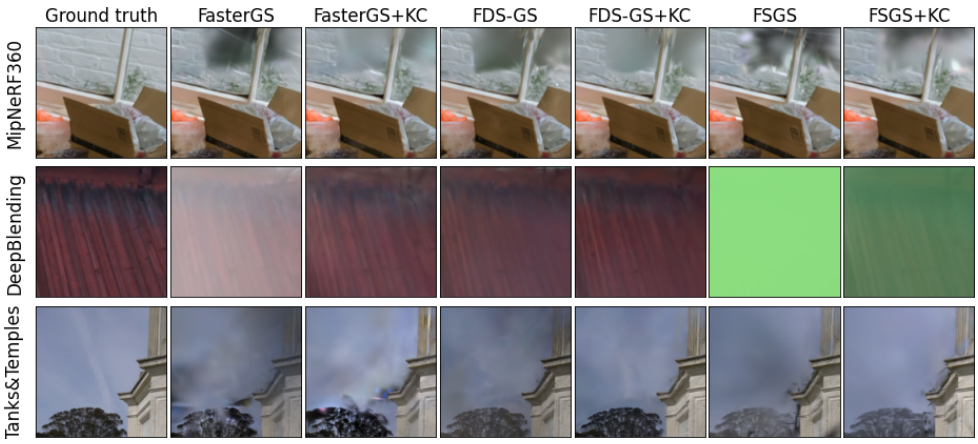

Figure 1: **Adding KC-3DGS (+KC) removes sparse-view floaters and color errors in three shown 3DGS methods.** Held-out crops at 12 training views (MipNeRF360 counter, DeepBlending Shed, Tanks and Temples Francis) from FasterGS (Hahlbohm et al., 2026), FDS-GS (Zeng et al., 2025) and FSGS (Zhu et al., 2024), each without and with +KC, hand-selected from candidates ranked by the share of error +KC removes.

Daubechies (1992). These kurtosis "concentration" properties have been widely exploited in classical image processing and, more recently, as priors for improving the perceptual quality of generative models Olshausen & Field (1996); Simoncelli & Adelson (1996); Chang et al. (2000); Portilla & Simoncelli (2000); Roy et al. (2025). Motivated by this, we ask: can we regularize explicit 3D radiance representations by aligning their wavelet-domain statistics with those of real images?

We propose KC-3DGS, a kurtosis-constrained Gaussian splatting framework for high-fidelity view synthesis. KC-3DGS augments the standard 3DGS objective with a lightweight, fully differentiable wavelet-based kurtosis concentration (KC) loss Kerbl et al. (2023); Roy et al. (2025). We apply a multi-scale Daubechies-3 wavelet transform to both rendered and ground-truth images, compute kurtosis per subband, and penalize discrepancies in both the global kurtosis range and the per-band values Mallat (1989); Daubechies (1992). This regularizer reduces pathological cross-band kurtosis imbalances while preserving the inherently heavy-tailed nature of natural images, encouraging sharper textures, cleaner edges, and more coherent structures—without modifying the 3DGS architecture or relying on additional geometric supervision Field (1987); Simoncelli & Olshausen (2001).

KC-3DGS is complementary to existing geometric, depth-based, or generative priors for robust NVS, and integrates seamlessly into existing 3DGS training pipelines Li et al. (2024); Zhu et al. (2024); Zhang et al. (2024); Wu et al. (2024). We evaluate our approach on MipNeRF360 Barron et al. (2022), Tanks and Temples Knapitsch et al. (2017), MVImgNet Yu et al. (2023), and a sparse-view iPad LTS dataset, focusing on challenging regimes with as few as 12 training views per scene. Across these benchmarks, KC-3DGS consistently improves standard metrics (PSNR, SSIM Wang et al. (2004), LPIPS Zhang et al. (2018), DreamSim Fu et al. (2023)), reduces floaters and oversmoothing, and yields more stable reconstructions. Analyzing the evolution of wavelet-band kurtosis during training further shows that KC regularization stabilizes 3DGS optimization and promotes more compact, faithful scene representations. Our contributions are threefold:

- We introduce KC-3DGS, a plug-and-play wavelet-domain kurtosis concentration loss for 3DGS that explicitly aligns higher-order statistics of rendered images with natural scenes.
- We propose a perceptually guided regularization strategy that operates on multi-scale wavelet coefficients rather than only pixel-space errors, improving structural coherence and reducing artifacts such as floaters and oversmoothed regions in sparse-view settings.
- We provide a comprehensive empirical and statistical analysis on several challenging NVS benchmarks, demonstrating that enforcing natural image statistics in the wavelet domain is an effective and general mechanism for enhancing the perceptual quality and robustness of explicit radiance representations.

2 Related Work

Sparse-view novel-view synthesis and 3DGS regularization. Sparse-view NVS is severely underconstrained because a small number of posed images does not fully determine scene geometry, visibility, or high-frequency appearance. NeRF-based approaches mitigate this ambiguity using semantic consistency (Jain et al., 2021), geometry and appearance regularization at unseen views (Niemeyer et al., 2022), frequency and occlusion regularization (Yang et al., 2023), or depth-ranking priors from imperfect depth estimates (Wang et al., 2023). Similar challenges arise in 3DGS: under sparse supervision, adaptive density control can overfit observed views, introduce floaters, and produce background collapse or oversmoothed structures. Existing sparse-view 3DGS methods therefore introduce depth priors, explicit consistency constraints, Gaussian unpooling, or coherent structured parameterizations to stabilize geometry (Xiong et al., 2025; Chung et al., 2024; Li et al., 2024; Zhu et al., 2024; Paliwal et al., 2024).

Natural Image Statistics and Wavelet Priors. Natural images exhibit well-characterized statistical regularities in the wavelet domain: sub-band coefficients follow sparse, heavy-tailed (high-kurtosis) distributions, and this structure is consistent across sub-bands and spatial frequencies (Field, 1987; Ruderman, 1994; Simoncelli & Olshausen, 2001). Wavelets provide a multi-resolution representation that separates image content across scale and orientation (Mallat, 1989; Daubechies, 1992). These properties underlie classical image processing techniques including sparse coding, compression, and denoising (Simoncelli & Adelson, 1996; Chang et al., 2000). The DiffNat (Roy et al., 2025) paper demonstrated that kurtosis concentration across wavelet sub-bands is a useful self-supervised signal for diffusion model artifact removal. We build on this observation but show that supervised alignment against ground-truth statistics is required when the target domain (3DGS) produces qualitatively different artifacts such as floaters, streaking, and oversmoothing. More related works are provided in Appendix A.1.

3 Method

3.1 Preliminaries: 3D Gaussian Splatting.

A 3DGS scene is parameterized by a set of N Gaussians, each defined by a mean position $\boldsymbol{\mu} \in \mathbb{R}^3$, a covariance matrix $\boldsymbol{\Sigma} = \mathbf{RSS}^\top \mathbf{R}^\top$ (decomposed into rotation $\mathbf{R}$ and scale $\mathbf{S}$), an opacity $\alpha \in [0, 1]$, and spherical harmonic coefficients encoding view-dependent color (Kerbl et al., 2023). Training minimizes a combination of L1 and SSIM losses against ground-truth images, with ADC applied periodically (Kerbl et al., 2023; Wang et al., 2004).

3.2 Kurtosis Concentration Loss.

While the cross-band covariance penalty reduces redundancy between frequency components, it does not regulate how kurtosis is distributed across wavelet bands. During 3DGS optimization, a few bands may become overly dominant while others remain weak, leading to unstable refinement and view-inconsistent structure. To mitigate this, we introduce a kurtosis concentration loss that penalizes excessive spread in band-wise kurtosis, encouraging balanced yet heavy-tailed frequency responses. Given flattened, standardized band responses $\hat{\mathbf{z}}_b(x) \in \mathbb{R}^P$, the kurtosis of band b is,

$$\kappa_b(x) = \frac{1}{P} \sum_{p=1}^{P} \left(\hat{\mathbf{z}}_b^{(p)}(x) \right)^4 - 3. \tag{1}$$

The minibatch kurtosis concentration loss is then defined as,

$$\mathcal{L}_{\text{kurt}} = \frac{1}{N} \sum_{n=1}^{N} \left[\max_b \kappa_b(x_n) - \min_b \kappa_b(x_n) \right]. \tag{2}$$

Minimizing this loss prevents any single wavelet band from dominating while preserving heavy-tailed spatial statistics, yielding more stable and structurally consistent 3DGS reconstructions.

3.3 Spatial Cross-Band Covariance Penalty.

In 3DGS, wavelet subbands should capture complementary structure: low-frequency bands model coarse geometry and illumination, while high-frequency bands encode textures and sharp details. However, redundant gradients across bands can cause cross-band interference, oversmoothing, texture leakage, and unstable multi-view refinement. We therefore introduce a spatial cross-band covariance regularizer that encourages frequency specialization by suppressing statistical dependency between subbands. Let $z_b(x) \in \mathbb{R}^{H \times W}$ be the b-th wavelet subband of a rendered image x, with B total

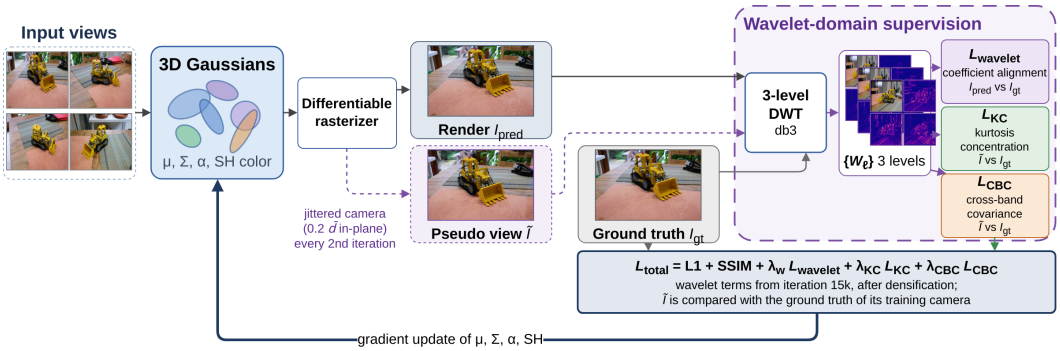

Figure 2: **KC-3DGS adds three wavelet-domain losses to 3DGS training, all starting after densification.** $\mathcal{L}_{wavelet}$ aligns the Daubechies-3 wavelet subbands of each training-view render with its ground truth, while $\mathcal{L}_{KC}$ matches subband kurtosis and $\mathcal{L}_{CBC}$ matches cross-subband covariance on renders from jittered pseudo cameras (Section 3.6). All three are added to the L_1 and SSIM losses of 3DGS, leaving the Gaussian representation and rasterizer unchanged.

bands and $P = HW$. We flatten and center each subband, stack the centered responses, and penalize off-diagonal covariance:

$$\mathbf{z}_b(x) \in \mathbb{R}^P, \qquad \tilde{\mathbf{z}}_b(x) = \mathbf{z}_b(x) - \frac{1}{P}\sum_{p=1}^{P} \mathbf{z}_b^{(p)}(x),$$
$$\tilde{Z}(x) = \left[\tilde{\mathbf{z}}_1(x), \ldots, \tilde{\mathbf{z}}_B(x)\right]^\top \in \mathbb{R}^{B\times P}, \qquad \Sigma_z(x) = \frac{1}{P-1}\tilde{Z}(x)\tilde{Z}(x)^\top, \tag{3}$$
$$\mathcal{L}_{\text{cov}} = \frac{1}{N}\sum_{n=1}^{N} \left\|\Sigma_z(x_n) - \text{Diag}(\text{diag}(\Sigma_z(x_n)))\right\|_F^2 .$$

This loss encourages wavelet bands to encode distinct structural cues, improving 3DGS optimization stability while preserving high-frequency detail and coherent low-frequency structure.

3.4 Wavelet Decomposition.

Given a rendered image $\mathbf{I}_{\text{pred}} \in \mathbb{R}^{H\times W\times 3}$ and corresponding ground truth $\mathbf{I}_{\text{gt}}$, we apply a J-level 2D discrete wavelet transform (DWT (Walnut, 2004)) using the Daubechies-3 (db3) wavelet via a differentiable PyTorch implementation (Cotter, 2019). As illustrated in Figure 3, the DWT recursively decomposes each channel into a low-frequency approximation subband (LL) and three detail subbands capturing horizontal (LH), vertical (HL), and diagonal (HH) edges. At each decomposition level, the approximation subband is further decomposed, yielding J (we use 3) sets of detail subbands $\{c_H^j, c_V^j, c_D^j\}_{j=1}^J$ at progressively coarser scales, plus a final approximation c_A. The resulting wavelet tree (Figure 3) reveals how edge information is distributed across scales: fine-scale subbands (bottom) capture high-frequency texture and sharp edges, while coarse-scale subbands (top-left quadrant) encode broader structural features.

We directly align the wavelet coefficients of the rendered image to those of the ground truth via an L1 loss on each detail subband, with a per-level weight that increases with decomposition depth (coarser levels, which have fewer coefficients, weighted more heavily):

$$\mathcal{L}_{\text{wav}} = \sum_{j=1}^{J} 2^j \left(\|c_H^{j,\text{pred}} - c_H^{j,\text{gt}}\|_1 + \|c_V^{j,\text{pred}} - c_V^{j,\text{gt}}\|_1 + \|c_D^{j,\text{pred}} - c_D^{j,\text{gt}}\|_1 \right), \tag{4}$$

where $j = 1$ corresponds to the finest scale. This weighting scheme addresses the energy decay inherent in natural images (Proposition 2), ensuring that fine-scale errors receive sufficient gradient signal to counteract the oversmoothing tendency of pixel-space losses.

3.5 Supervised Wavelet Statistics Losses.

While the KC and cross-band covariance (CBC) losses described above can be applied in an unsupervised manner to encourage desirable statistical properties in rendered images, we found that

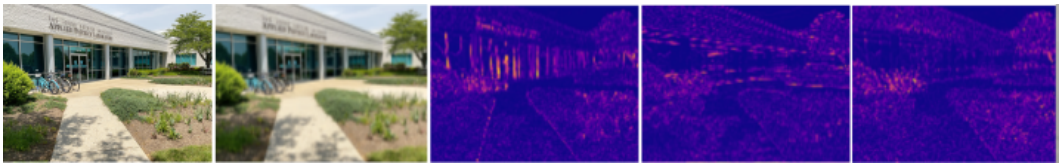

Figure 3: **Wavelet decomposition of a natural image.** Three-level discrete wavelet transform showing the ground truth, low-frequency approximation, and detail coefficients capturing vertical (HL), horizontal (LH), and diagonal (HH) edges at the coarsest resolution (in order left to right). We stack three resolution scales that get progressively finer. The kurtosis concentration loss exploits the statistical regularity of these band-pass subbands, where natural images exhibit characteristic heavy-tailed distributions that our method encourages in rendered outputs.

unsupervised application leads to suboptimal results for 3DGS. The KC loss, which minimizes the range of kurtosis values across wavelet subbands, can be trivially satisfied by producing uniformly blurry outputs where all subbands exhibit similarly low kurtosis. Similarly, the CBC penalty, which discourages correlation between oriented subbands (LH, HL, HH) at the same spatial locations, may inadvertently penalize sharp corners and edge intersections which are high-frequency geometric details we wish to preserve. Without grounding these statistics to reference values, the optimization has no target for what constitutes "correct" frequency behavior.

To address this, we reformulate both losses as supervised alignment objectives. Given a rendered image $\mathbf{I}_{\text{pred}}$ and ground truth $\mathbf{I}_{\text{gt}}$, we compute the KC and CBC statistics for both images and penalize their absolute difference:

$$\mathcal{L}_{\text{KC}}^{\text{sup}} = \left|L_{\text{kurt}}(\mathbf{I}_{\text{pred}}) - L_{\text{kurt}}(\mathbf{I}_{\text{gt}})\right| + \frac{1}{B}\sum_{b=1}^{B}\left|\kappa_b(\mathbf{I}_{\text{pred}}) - \kappa_b(\mathbf{I}_{\text{gt}})\right|, \tag{5}$$

$$\mathcal{L}_{\text{CBC}}^{\text{sup}} = \left|L_{\text{cov}}(\mathbf{I}_{\text{pred}}) - L_{\text{cov}}(\mathbf{I}_{\text{gt}})\right|. \tag{6}$$

The supervised KC loss encourages the prediction to match both the overall kurtosis spread and the per-subband kurtosis values of the ground truth, ensuring that the rendered image exhibits the same frequency-dependent peakedness characteristics as natural images. The supervised CBC loss ensures that the spatial correlation structure between oriented subbands in the prediction matches that of the ground truth—if the ground truth contains corners that induce cross-band correlation, the prediction is encouraged to reproduce them rather than smooth them away. Combined with the direct wavelet coefficient alignment loss $\mathcal{L}_{\text{wav}}$, these supervised statistics losses provide multi-scale frequency guidance that complements pixel-space reconstruction objectives, enabling sharper and more geometrically faithful 3DGS reconstructions.

3.6 Implementation details.

Here we provide implementation details of the training. For the first half of training ($t < T/2$), the objective is the 3DGS photometric loss alone, $\mathcal{L}_{\text{photo}} = 0.8\,L_1 + 0.2\,(1 - \text{SSIM})$, with densification ending at iteration 14,900. Any term active during densification changes the Gaussian set and collapses the reconstruction (Appendix A.3). From $t = T/2$, $\mathcal{L}_{\text{wav}}$ ($\lambda_{\text{wav}} = 9 \times 10^{-2}$) acts on the training view, where its target is pixel-aligned. On every second iteration, $\mathcal{L}_{\text{KC}}^{\text{sup}}$ ($\lambda_{\text{KC}} = 3 \times 10^{-3}$) and $\mathcal{L}_{\text{CBC}}$ compare a render $\tilde{\mathbf{I}}$ from a pseudo camera, translated in-plane by $0.2\,\bar{d}$, with the anchor ground truth $\mathbf{I}_{\text{gt}}$. Here $\bar{d}$ is the median nearest-neighbor distance between training cameras. On training views the photometric loss already matches the band statistics; sparse-view artifacts appear at novel viewpoints. Because the statistics pool over each subband, they tolerate the offset that would make a pixel-aligned loss supervise the wrong content. The weight of $\mathcal{L}_{\text{CBC}}$ is set adaptively to hold it at a fixed 0.1 share of the objective. Implementation details (weight schedule, kurtosis winsorization, and the normalized forms of $\mathcal{L}_{\text{CBC}}$ and $\mathcal{L}_{\text{wav}}$) are in Appendix A.2.

4 Theoretical Motivation

We formalize the intuition behind KC-3DGS through results that justify each component of our objective. Let $I_{\text{pred}}, I_{\text{gt}} \in \mathbb{R}^{H\times W}$ denote a single rendered channel and its ground truth, and let $\mathcal{W}$ denote the J-level orthogonal 2D DWT producing one approximation subband and $3J$ detail

Table 1: Sparse-view evaluation on FasterGS. Adding KC improves most metrics on all datasets. Parentheses: improvement over FasterGS (dB for PSNR, % for the others; positive is better).

Dataset	Method	PSNR ↑	SSIM ↑	LPIPS ↓	DreamSim ↓
MipNeRF360 (9)	FasterGS	18.36	0.5015	0.4189	0.1477
	FasterGS+KC	**18.61** (+0.25)	**0.5155** (+2.8%)	**0.4114** (+1.8%)	**0.1374** (+7.0%)
Tanks&Temples (16)	FasterGS	15.56	0.5573	0.4504	0.2427
	FasterGS+KC	**15.67** (+0.11)	**0.5693** (+2.1%)	**0.4430** (+1.6%)	**0.2303** (+5.1%)
DeepBlending (17)	FasterGS	18.96	0.6684	0.3771	0.2079
	FasterGS+KC	**19.25** (+0.29)	**0.6805** (+1.8%)	**0.3724** (+1.2%)	**0.2038** (+1.9%)
WRIVA-ULTRRA (8)	FasterGS	13.29	0.3210	**0.5921**	0.4111
	FasterGS+KC	**13.44** (+0.16)	**0.3339** (+4.0%)	0.5937 (−0.3%)	**0.3989** (+3.0%)
MVImgNet (7)	FasterGS	26.02	0.8774	0.1134	0.0191
	FasterGS+KC	**26.26** (+0.25)	**0.8788** (+0.2%)	**0.1122** (+1.1%)	**0.0174** (+8.7%)

subbands $\{c_H^j, c_V^j, c_D^j\}_{j=1}^J$. We assume an orthogonal wavelet basis (Daubechies in our experiments), treating boundary effects as negligible.

Standard 3DGS objectives admit a continuous family of distinct renderings with identical pixel-space loss energy.

Proposition 1 (Pixel-space invariance under wavelet redistribution)**.** *Let $\Delta = I_{pred} - I_{gt}$. Parseval's identity gives $\|\Delta\|_2 = \|\mathcal{W}\Delta\|_2$, so per-subband energies may be redistributed freely subject only to $\sum_j \|(\mathcal{W}\Delta)^j\|_2^2 = \|\Delta\|_2^2$, yielding a $(3J-1)$-dimensional family of perturbations indistinguishable under pixel-space L^2 supervision.*

Proofs are provided in Appendix B. This formalizes that L^2 (and, modulo norm equivalence, L^1 and SSIM) constrains only *aggregate* error energy, not its *distribution across scales*. Subband supervision closes this loophole.

Lemma 1 (Subband supervision strictly refines pixel supervision)**.** *For any orthogonal $\mathcal{W}$, $\sum_j \|(\mathcal{W}\Delta)^j\|_1 \geq \|\Delta\|_1$, with equality iff Δ has single-subband support. Hence $\mathcal{L}_{wav} = 0 \implies \mathcal{L}_1 = 0$, but not conversely.*

For natural images modeled as $1/f^\alpha$ processes with $\alpha \in [1, 2]$, the expected per-coefficient energy at level j scales as $2^{-j(\alpha-1)} \cdot 2^{-2j}$, accounting for the 4^j-fold reduction in coefficient count. An unweighted $\sum_j \|(\mathcal{W}\Delta)^j\|_1$ is therefore dominated by coarse-scale terms (precisely the scales 3DGS already handles well), while fine-scale errors that produce oversmoothing contribute negligibly. Re-weighting by 2^j inverts this falloff and approximately equalizes optimization influence across scales, giving $\mathcal{L}_{\text{wav}} = \sum_{j=1}^J 2^j \sum_{o \in \{H,V,D\}} \|c_o^{j,\text{pred}} - c_o^{j,\text{gt}}\|_1$.

Cross-band covariance as decorrelation. Modeling natural-image subbands as approximately factorized, $p(z_1, \ldots, z_B) = \prod_b p_b(z_b)$, motivates the third regularizer.

Lemma 2 (Frobenius decorrelation as Gaussian-factorization KL)**.** *Let Σ_z be the empirical cross-band covariance. The squared off-diagonal Frobenius norm $\|\Sigma_z - \text{diag}(\Sigma_z)\|_F^2$ equals, to leading order, the KL divergence between $\mathcal{N}(0, \Sigma_z)$ and the closest factorized Gaussian $\prod_b \mathcal{N}(0, \Sigma_{bb})$. With unit marginal variances, $\text{KL} = \frac{1}{2}\|\Sigma_z - \text{diag}(\Sigma_z)\|_F^2 + O(\|E\|^3)$, where E collects the off-diagonals.*

This grounds $\mathcal{L}_{\text{cov}}$ as a tractable likelihood-based decorrelation criterion that is genuinely *unsupervised*: it encodes a property of the target distribution (subband independence) rather than of any specific reference image.

5 Experiments

5.1 Datasets and Evaluation Protocol.

We evaluate on five benchmarks spanning structured academic scenes and realistic sparse-view site modeling. **MipNeRF360** (Barron et al., 2022) covers diverse indoor and outdoor scenes; **Tanks&Temples** (Knapitsch et al., 2017) tests large-scale geometry with wide-baseline views; and **DeepBlending** (Hedman et al., 2018) contains bounded indoor scenes with challenging lighting and

Table 2: KC-3DGS added to other sparse-view methods on MipNeRF360. Adding KC improves most metrics for every method. Parentheses as in Table 1.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	DreamSim ↓
FasterGS	18.36	0.5015	0.4189	0.1477
FasterGS+KC	**18.61** (+0.25)	**0.5155** (+2.8%)	**0.4114** (+1.8%)	**0.1374** (+7.0%)
FDS-GS	18.57	0.5263	0.3945	0.1290
FDS-GS+KC	**18.95** (+0.38)	**0.5413** (+2.8%)	**0.3881** (+1.6%)	**0.1174** (+9.0%)
FSGS	19.23	0.5406	**0.4284**	0.1376
FSGS+KC	**19.40** (+0.16)	**0.5525** (+2.2%)	0.4345 (−1.4%)	**0.1315** (+4.4%)
LAP	17.60	0.4810	0.4424	0.1847
LAP+KC	**18.21** (+0.61)	**0.5010** (+4.2%)	**0.4323** (+2.3%)	**0.1696** (+8.2%)

reflective surfaces. **MVImgNet** (Yu et al., 2023) is an object-centric multi-view benchmark of short hand-held captures.

Our most challenging setting is **WRIVA-ULTRRA** (Brown et al., 2024), whose sparse, irregular captures include transient objects, changing illumination and mixed camera models. We keep the 8 of 15 public scenes in which COLMAP registers at least 80% of the images, since we do not address camera registration, and test on the challenge's held-out reference views (3–41 per scene). On the other datasets, we hold out every 8th image for testing and train on 12 of the remaining views, one from each of 12 contiguous segments of the capture trajectory; DeepBlending Street-10 and Yellowhouse-12 are too small for this split, leaving 17 of its 19 scenes, and MVImgNet and 5 DeepBlending scenes have only 3–4 test views. All main results use FasterGS (Hahlbohm et al., 2026) for 30k iterations (seed 0, image scales in Appendix A.2), and KC-3DGS differs from it only in the protocol of Section 3.6, with the same trainer, split, seed and image scale in every scene. We report PSNR, SSIM, LPIPS (VGG) and DreamSim averaged over the scenes of each dataset, together with the number of scenes on which KC-3DGS improves each metric.

5.2 Main Results

Sparse-View Evaluation. Table 1 reports all 57 scenes at 12 views. KC-3DGS lowers DreamSim on 48 scenes and LPIPS on 46, and raises SSIM on 52 and PSNR on 40. Mean DreamSim improves by 1.9–8.7% per dataset, and mean PSNR rises by 0.11–0.29 dB. The one mean regression is LPIPS on WRIVA-ULTRRA (-0.3% improvement), and the largest single-scene PSNR loss is 0.93 dB (DeepBlending Bedroom).

KC-3DGS in other sparse-view methods. Because KC-3DGS only adds loss terms, it combines with other 3DGS methods (Table 2, MipNeRF360). It improves all four metrics on every scene for FDS-GS (Zeng et al., 2025) ($+0.38$ dB PSNR, $+9.0\%$ DreamSim improvement), raises PSNR and SSIM on every scene for LAP (Lavi et al., 2026) ($+0.61$ dB), and lowers DreamSim for FSGS (Zhu et al., 2024) on every scene, though FSGS's LPIPS worsens slightly (-1.4% improvement; other datasets in Appendix A.8). The gain requires per-Gaussian color: in DNGaussian (Li et al., 2024), whose color comes from a shared network, KC-3DGS worsens LPIPS and DreamSim (Appendix A.10).

Comparison with other perceptual losses. We also compare KC-3DGS with a perceptual loss: LPIPS (Zhang et al., 2018) applied to the same pseudo views, in place of our three terms and at a 20% loss share. KC-3DGS performs better: pseudo-view LPIPS recovers only about a third of its DreamSim reduction (-0.0037 vs. -0.0104), even though it optimizes an LPIPS network directly (Table 3). The two are complementary: combined, they improve further (PSNR $+0.34$ dB, DreamSim -0.0135).

Qualitative Analysis. Figure 4 shows adding KC improves image quality across multiple objects on MipNeRF360 datasets, with smoother details across grass, road, table, also improving PSNR. Figure 7 (Appendix A.11) shows six failures of 3DGS at 12 views, among them a green cast over the sky and statue (Tanks&Temples Francis), a floater that renders as a flat white patch over the ground (MipNeRF360 treehill), and dark streaked floaters in the sky (DeepBlending Boats). They occur where few training views constrain a region, so Gaussians fitted to one view occlude or tint it from another. KC-3DGS removes the first two and shrinks the third; streaks remain on Boats. Whole-view PSNR of the three views rises from 16.68 to 21.39, 8.95 to 14.81 and 12.00 to 13.89 dB. Figure 5 shows WRIVA-ULTRRA renders from a version of KC-3DGS without the pseudo views, in which

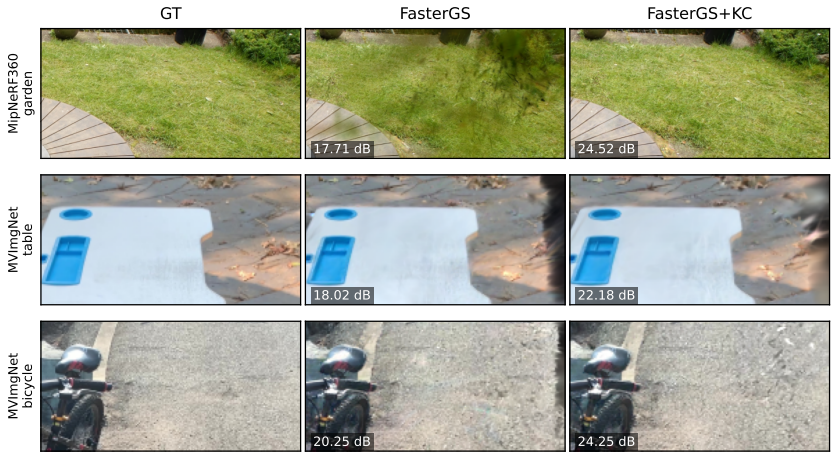

Figure 4: **FasterGS+KC removes floaters that FasterGS leaves in sparse-view renders.** Crops are held-out views from the 12 training views run showing the types of floater artifacts KC3DGS eliminates or suppresses.

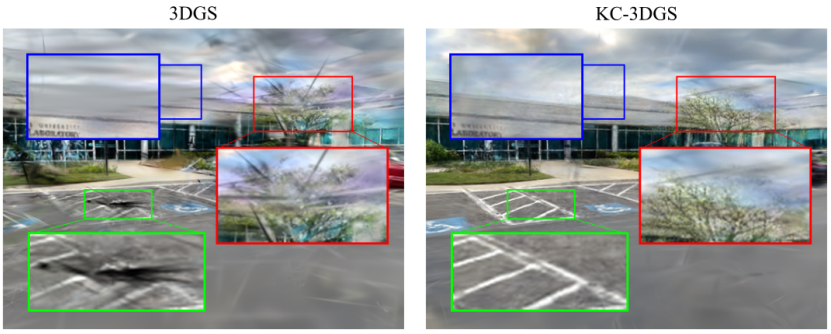

Figure 5: Qualitative comparison of FasterGS on the WRIVA-ULTRRA dataset, a challenging outdoor scene with sparse viewpoints with all three terms on the training views from the first iteration. **Blue**: It produces cleaner building facade reconstruction with reduced streak artifacts and improved window grid definition. **Red**: Foliage detail is better preserved, with sharper leaf structure and reduced color bleeding compared to the baseline's blurred appearance. **Green**: Asphalt texture and parking lot markings show improved clarity; the baseline exhibits characteristic over-smoothing on ground plane textures.

all three terms act on the training views only. We also perform human evaluation which is mentioned in the supplementary (Sec. A.12).

Interpretability Analysis. Figure 6 shows that wavelet-domain supervision emphasizes errors underweighted by pixel losses. Early in training, rendered LH/HL/HH subbands are weaker than the ground truth, indicating missing horizontal, vertical, and diagonal high-frequency structure. Wavelet L_1 errors concentrate on structural regions such as sidewalks, building edges, foliage, and parking markings rather than homogeneous sky. The normalized wavelet-to-pixel error ratio shifts from pixel-dominated coarse errors early in training to wavelet-dominated fine-structure errors later.

Ablation. On MipNeRF360, all 2^3 combinations of the three terms under the protocol of Section 3.6 (Table 4; all eight in Appendix A.4) show that pseudo-view $\mathcal{L}_{\text{KC}}$ carries the perceptual gain: alone it lowers DreamSim on 9 of 9 scenes, and adding $\mathcal{L}_{\text{CBC}}$ lowers it further and puts PSNR above 3DGS on every scene. The photometric loss on the same pseudo views matches KC-3DGS in PSNR but gives only a third of its DreamSim reduction, so the gain comes from the band statistics, not from supervising extra views. Over three training seeds, PSNR improves by $+0.26$ to $+0.27$ dB and DreamSim is lower on 9 in–9 of 9 scenes (Appendix A.6).

Number of training views. KC-3DGS targets sparse views, and its benefit fades as views are added (Table 11, MipNeRF360). At 12 and 24 views it improves every metric ($+0.26$ and $+0.35$ dB PSNR); at 50 views the differences are within run-to-run variation; with the full training set it is

Table 3: Comparison with other perceptual losses (LPIPS).

Method	ΔPSNR↑	ΔSSIM↑	ΔLPIPS↓	ΔDreamSim↓
FasterGS+KC	+0.23 (8/9)	+0.0128 (9/9)	−0.0076 (9/9)	−0.0104 (8/9)
FasterGS+pseudo-view LPIPS	+0.20 (9/9)	+0.0022 (8/9)	−0.0035 (9/9)	−0.0037 (7/9)
FasterGS+KC+pseudo-view LPIPS	+0.34 (8/9)	+0.0150 (9/9)	−0.0112 (9/9)	−0.0135 (8/9)
FasterGS+pseudo-view photometric†	+0.25 (7/9)	+0.0036 (7/9)	−0.0002 (7/9)	−0.0040 (6/9)

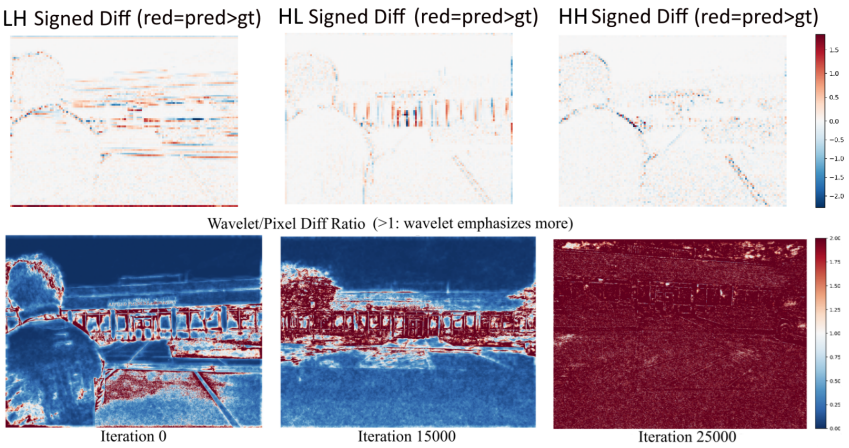

Figure 6: **L3 wavelet subbands over training.** First row: Signed differences between coarse-scale wavelet coefficients of predicted and ground-truth images across LH, HL, and HH detail subbands. Second row: Ratio of normalized wavelet detail differences to normalized pixel L_1 differences (red: wavelet-emphasized regions, blue: pixel-emphasized regions) on WRIVA-ULTRRA with 50 training images. Over training, wavelet-domain errors increasingly dominate pixel-space errors, especially in fine structural regions. Further training ablations and analysis are in Appendix Table 10, Figure 10, and Appendix A.2.

Table 4: Term ablations for MipNeRF360 with FasterGS on 12 training views.

Method	ΔPSNR↑	ΔSSIM↑	ΔLPIP S↓	ΔDreamSim↓
FasterGS+$\mathcal{L}_{\text{wav}}$	+0.05 (9/9)	+0.0024 (9/9)	−0.0005 (7/9)	+ 0.0006 (4/9)
FasterGS+$\mathcal{L}_{\text{KC}}$	+0.18 (7/9)	+0.0097 (9/9)	−0.0069 (9/9)	−0.0094 (9/9)
FasterGS+$\mathcal{L}_{\text{CBC}}$	−0.10 (3/9)	+0.0058 (7/9)	−0.0041 (9/9)	−0.0026 (5/9)
FasterGS+$\mathcal{L}_{\text{KC}}$+$\mathcal{L}_{\text{CBC}}$	+0.25 (9/9)	+0.0129 (9/ 9)	−0.0084 (9/9)	−0.0126 (9/9)
FasterGS+KC (all three)	+0.26 (9/9)	+0.0137 (9/9)	−0.0088 (9/9)	−0.0120 (8/9)

worse on every scene (0.77 dB PSNR on average). With dense views little is unconstrained, and the pseudo-view terms, whose target is offset from the pseudo camera, pull the reconstruction away from the photometric optimum.

Limitations. KC-3DGS is a sparse-view method. With dense views it lowers PSNR, and does not provide any substantial benefits. Its gains are mainly perceptual; PSNR improves only on 40 of 57 scenes. Because the terms act after densification, errors in the Gaussian set such as missing geometry remain. The terms also require per-Gaussian color control. Methods where color control is not explicitly controllable such as with a shared color network in DNGaussian hurts performance (Appendix A.10). Finally, the pseudo-view pass raises training time to 1.39–2.67× that of 3DGS (Appendix A.2).

6 Conclusion

We introduced KC-3DGS, a wavelet-domain regularizer for sparse-view 3DGS. Pixel-space losses constrain the total error of a render but not how it is distributed across scales, so they allow over-smoothed textures, floaters and color casts at novel viewpoints. KC-3DGS addresses this with scale-weighted wavelet alignment and supervised matching of subband kurtosis and cross-band correlation. The statistics are applied to pseudo views after densification. With 12 training views on 57 scenes from five benchmarks, KC-3DGS lowers DreamSim on 48 scenes and improves all four mean metrics on every dataset except WRIVA-ULTRRA (LPIPS). It also improves three existing sparse-view 3DGS methods as a plug-and-play addition. Qualitatively, it removes floaters and color casts in weakly observed regions. Its benefit is specific to sparse views, and extending it to methods with shared color networks is left for future work.

AI USE STATEMENT

In this work, we used generative AI tools to assist with writing code for running experiments, analyzing results, and to polish writing. We have not used generative AI tools in the proposed method: KC-3DGS. We have reviewed all AI-assisted work.

ETHICS STATEMENT

This work proposes a training-time regularizer for 3D Gaussian Splatting. Its only human-subject component is the preference study of Appendix A.12, an anonymous web task of about six minutes. Before starting, observers are told what is recorded: their choices, response times, screen size and browser type, with no account and no personal data beyond optional initials. Each observer receives a participant code with which their responses can be withdrawn. The images show building exteriors and scenes from public benchmarks. All experiments use publicly available benchmarks (MipNeRF360, Tanks&Temples, MVImgNet, DeepBlending) and the WRIVA-ULTRRA challenge release, used in accordance with their respective licenses. As with other 3D reconstruction methods, improved view synthesis could be misused to reconstruct scenes without consent; our method does not add capabilities beyond existing 3DGS pipelines in this respect.

REPRODUCIBILITY STATEMENT

The full loss formulation, including the wavelet coefficient alignment, kurtosis concentration, and cross-band covariance terms, is given in Section 3. Theoretical assumptions and complete proofs are provided in Section 4 and Appendix B. Datasets, view splits, pose sources, baseline implementations (Splatfacto and FasterGS), and evaluation metrics are described in Section 5; implementation details and compute (Appendix A.2), the training-protocol evidence (Appendix A.3), ablations (Appendix A.4) and seed spread (Appendix A.6) are in the appendix. Code will be provided upon acceptance.

REFERENCES

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 (CVPR)*, pp. 5460–5469, 2022.

Myron Brown, Michael Chan, and Michael Twardowski. WRIVA public data. IEEE DataPort, 2024. URL https://doi.org/10.21227/cjk5-gf33.

S. Grace Chang, Bin Yu, and Martin Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, 9(9):1532–1546, 2000. doi: 10.1109/83.862633.

Jaeyoung Chung, Jeongtaek Oh, and Kyoung Mu Lee. Depth-regularized optimization for 3D gaussian splatting in few-shot images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2024.

Fergal Cotter. *Uses of Complex Wavelets in Deep Convolutional Neural Networks*. PhD thesis, University of Cambridge, 2019. URL https://www.repository.cam.ac.uk/handle/1810/306661.

Ingrid Daubechies. *Ten Lectures on Wavelets*. SIAM, Philadelphia, PA, 1992. doi: 10.1137/1.9781611970104.

David J. Field. Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America A*, 4(12):2379–2394, 1987. doi: 10.1364/JOSAA.4.002379.

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5501–5510, 2022.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, volume 36, pp. 50742–50768, 2023.

Florian Hahlbohm, Linus Franke, Martin Eisemann, and Marcus Magnor. Faster-GS: Analyzing and improving gaussian splatting optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18946–18957, 2026.

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 (Proc. SIGGRAPH Asia)*, 37(6):257:1–257:15, 2018.

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024. doi: 10.1145/3641519.3657428.

Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting NeRF on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5885–5894, 2021.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, 2023. doi: 10.1145/3592433.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017.

Yishai Lavi, Leo Segre, and Shai Avidan. Frequency-aware gaussian splatting decomposition. In *2026 International Conference on 3D Vision (3DV)*, pp. 1133–1142. IEEE Computer Society, 2026. doi: 10.1109/3DV69130.2026.00112.

Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. DNGaussian: Optimizing sparse-view 3D gaussian radiance fields with global-local depth normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

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. doi: 10.1109/34.192463.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pp. 405–421. Springer, 2020.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):102:1–102:15, 2022. doi: 10.1145/3528223.3530127.

Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. RegNeRF: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5470–5480, 2022. doi: 10.1109/CVPR52688.2022.00540.

Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996. doi: 10.1038/381607a0.

Avinash Paliwal, Wei Ye, Jinhui Xiong, Dmytro Kotovenko, Rakesh Ranjan, Vikas Chandra, and Nima Khademi Kalantari. CoherentGS: Sparse novel view synthesis with coherent 3D gaussians. In *European Conference on Computer Vision (ECCV)*, pp. 19–37. Springer, 2024.

Javier Portilla and Eero P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–70, 2000. doi: 10.1023/A:1026553619983.

Aniket Roy, Maitreya Suin, Anshul Shah, Ketul Shah, Jiang Liu, and Rama Chellappa. Diffnat : Exploiting the kurtosis concentration property for image quality improvement. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=HdZQ7pMPRd.

Daniel L. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, 5 (4):517–548, 1994. doi: 10.1088/0954-898X_5_4_006.

Eero P. Simoncelli and Edward H. Adelson. Noise removal via bayesian wavelet coring. In *Proceedings of the 3rd IEEE International Conference on Image Processing (ICIP)*, volume 1, pp. 379–382, 1996.

Eero P. Simoncelli and Bruno A. Olshausen. Natural image statistics and neural representation. *Annual Review of Neuroscience*, 24:1193–1216, 2001. doi: 10.1146/annurev.neuro.24.1.1193.

David F. Walnut. *The Discrete Wavelet Transform*, pp. 215–248. Birkhäuser Boston, Boston, MA, 2004. ISBN 978-1-4612-0001-7. doi: 10.1007/978-1-4612-0001-7_8. URL https://doi.org/10.1007/978-1-4612-0001-7_8.

Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. SparseNeRF: Distilling depth ranking for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9065–9076, 2023.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/TIP.2003.819861.

Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P. Srinivasan, Dor Verbin, Jonathan T. Barron, Ben Poole, and Aleksander Holynski. Reconfusion: 3d reconstruction with diffusion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21551–21561, 2024. doi: 10.1109/CVPR52733.2024.02036.

Haolin Xiong, Sairisheek Muttukuru, Hanyuan Xiao, Rishi Upadhyay, Pradyumna Chari, Yajie Zhao, and Achuta Kadambi. SparseGS: Sparse view synthesis using 3D gaussian splatting. In *International Conference on 3D Vision (3DV)*, pp. 1032–1041, 2025. doi: 10.1109/3DV66043.2025.00100.

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 (CVPR)*, 2023.

Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, Guanying Chen, Shuguang Cui, and Xiaoguang Han. MVImgNet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9150–9161, 2023. doi: 10.1109/CVPR52729.2023.00883.

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 (CVPR)*, 2024.

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*, volume 39, pp. 9833–9841, 2025. doi: 10.1609/aaai.v39i9.33066.

Jiawei Zhang, Jiahe Li, Xiaohan Yu, Lei Huang, Lin Gu, Jin Zheng, and Xiao Bai. CoR-GS: Sparse-view 3D gaussian splatting via co-regularization. In *European Conference on Computer Vision (ECCV)*, pp. 335–352, 2024. doi: 10.1007/978-3-031-73232-4_19.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018.

Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. FSGS: Real-time few-shot view synthesis using gaussian splatting. In *European Conference on Computer Vision (ECCV)*, 2024.

A Supplementary Material

A.1 Related Work

Neural and explicit radiance-field representations. Novel-view synthesis has progressed from implicit neural radiance fields to explicit, optimizable scene representations. NeRF models a scene as a continuous volumetric radiance field and renders novel views through differentiable volume rendering (Mildenhall et al., 2020). Subsequent neural-field methods improve scalability, anti-aliasing, and efficiency for real-world or unbounded captures (Barron et al., 2022; Müller et al., 2022). Explicit radiance-field representations, such as Plenoxels, show that high-quality view synthesis can also be achieved without a large neural MLP by optimizing sparse volumetric primitives directly (Fridovich-Keil et al., 2022). 3D Gaussian Splatting (3DGS) further advances this line by representing scenes with anisotropic 3D Gaussians and rasterizing them efficiently, enabling real-time rendering with strong reconstruction quality (Kerbl et al., 2023). Recent Gaussian-splatting variants address aliasing, surface consistency, and geometric fidelity through footprint-aware filters or surface-aligned splats (Yu et al., 2024; Huang et al., 2024). These methods primarily modify the representation or rendering pipeline. KC-3DGS is complementary: it keeps the underlying 3DGS architecture unchanged and regularizes the rendered images through multi-scale wavelet statistics.

A.2 Implementation Details

Backbone. All FasterGS results use NerfICG's FasterGS (Hahlbohm et al., 2026) with its default configuration except a cap of 3,000,000 Gaussians (the default has none; it binds in 11 runs of Table 1 and on bicycle in Table 6): 30,000 iterations, spherical harmonics of degree 3, densification every 100 iterations from 600 to 14,900, opacity reset every 3,000 iterations during densification, and the photometric loss $0.8\,L_1 + 0.2\,(1 - \mathrm{SSIM})$ with no opacity or scale regularization. KC-3DGS and 3DGS runs of a scene differ only in the terms of Section 3.6. Training images are downscaled by 0.5 on DeepBlending, by 0.2–0.5 per scene on WRIVA-ULTRRA, and by 0.25 on the other datasets, except Tanks&Temples Truck, which its loader provides at native size.

KC-3DGS terms. The wavelet transform is a 3-level Daubechies-3 DWT with zero padding (`pytorch_wavelets` (Cotter, 2019)), applied per color channel to images clamped to $[0, 1]$. Table 5 lists every setting; none is tuned per dataset or scene. The weight of the pseudo-view $\mathcal{L}_{\text{CBC}}$ holds it at a fixed share $r = 0.1$ of the objective:

$$\lambda_{\text{CBC},t} = \min\left(\frac{r}{1-r}\,\frac{\bar{R}_t}{\bar{C}_t},\; 10\,\lambda_{\text{CBC},T/2}\right), \tag{7}$$

where $\bar{R}_t$ and $\bar{C}_t$ are exponential moving averages (rate 0.1) of the remaining objective and of the raw $\mathcal{L}_{\text{CBC}}$.

Table 5: KC-3DGS settings, identical for every dataset and scene.

Setting	Value	Where
Start iteration	15,000 ($T/2$), after densification ends at 14,900	all terms
λ_{wav}	9×10^{-2}, training view, every iteration	Eq. (4)
λ_{KC}	3×10^{-3} fixed, pseudo view	$\mathcal{L}_{\text{KC}}^{\text{sup}}$
$\lambda_{\text{CBC},t}$	share $r = 0.1$, EMA rate 0.1, cap $10\times$ initial	pseudo view
Pseudo-view frequency	every 2nd iteration (bounds cost; not tuned)	
Pseudo-camera offset	$0.2\,\bar{d}$ (median nearest-neighbor camera distance), in-plane, random direction	
Kurtosis winsorization	± 6 standard deviations	Eq. (1)
$\mathcal{L}_{\text{CBC}}$ form	squared difference of subband correlations	Sec. 3.6
$\mathcal{L}_{\text{wav}}$ pooling	level L_1 sums divided by the finest level's coefficient count	Sec. 3.6

Evaluation. Test views are rendered at training resolution and scored against the ground truth at the same resolution: PSNR, SSIM, LPIPS with the VGG backbone (Zhang et al., 2018), and DreamSim with its default ensemble (Fu et al., 2023). Per-scene values are means over test views; dataset values are means over scenes. Every paired comparison is checked to score the same images: test views

are matched across the two runs of a scene by ground-truth pixel content, and all 1,268 test views of Table 1 pair exactly. The methods of Table 2 run their released code through the same split files and image scales, and both variants of each pair use that method's loader, so they score the same test views. FSGS and DNGaussian receive monocular depth from their own pipelines.

Run-to-run variation. Repeating a run with the same seed and configuration does not reproduce it exactly (nondeterministic GPU accumulation; KC-3DGS also samples pseudo cameras). On three MipNeRF360 scenes (bicycle, counter, stump, 12 views), two runs of 3DGS differ by at most 0.07 dB PSNR, 0.002 LPIPS and 0.015 DreamSim, and two runs of KC-3DGS by at most 0.23 dB, 0.002 and 0.006. Per-scene differences smaller than this are not distinguishable from repetition; the dataset-level claims rest on how many scenes change in the same direction.

Compute. The 114 runs of Table 1 used NVIDIA A40 (58 runs), RTX A5000 (40) and RTX 5000 Ada (16) GPUs, some sharing a GPU with other runs, so the logged wall-clock times of the main runs are not comparable across runs. For cost, Table 6 runs 3DGS and KC-3DGS alone on one NVIDIA A40 each, on three MipNeRF360 scenes with 12 views and the full-view split. KC-3DGS takes 1.39–2.67$\times$ the training time of 3DGS; the overhead is largest on the cheapest scenes because the pseudo-view pass costs about the same on every scene. Mean step time covers all 30,000 iterations, including the first 15,000 where both methods are identical, so all of the added time falls in the second half. Peak memory rises by 0.25–0.76 GB (20–31%). KC-3DGS ends with 0.90–0.99$\times$ the Gaussians of 3DGS.

Table 6: Training cost, one run per NVIDIA A40, 30,000 iterations, MipNeRF360. 3DGS / KC-3DGS.

Scene	Split	Time (min)	Mean step (ms)	Peak memory (GB)	Gaussians (M)
bicycle	12 views	10.5 / 16.3	18.6 / 30.5	3.40 / 4.15	2.99 / 2.92
counter	12 views	3.9 / 10.4	6.9 / 15.2	0.82 / 1.08	0.59 / 0.58
stump	12 views	6.4 / 11.9	12.0 / 23.2	1.96 / 2.51	1.61 / 1.60
bicycle	full	14.7 / 20.5	19.7 / 31.2	3.43 / 4.19	2.96 / 2.74
counter	full	5.2 / 9.5	9.1 / 18.0	1.29 / 1.61	0.99 / 0.89
stump	full	12.1 / 17.7	16.6 / 29.0	3.31 / 3.96	2.98 / 2.85

A.3 Evidence for the Training Protocol

Real-weight terms during densification destroy the reconstruction. A kurtosis term involves a fourth power, so a coefficient at 6 standard deviations is pulled $6^3 = 216\times$ harder than one at 1. On training views and from the first iteration, we held $\mathcal{L}_{\text{KC}}$ at a fixed share X of the loss with the rule of Section 3.6 (no cap), everything else as in the training-view configuration ($\mathcal{L}_{\text{wav}}$, and $\mathcal{L}_{\text{KC}}$ and $\mathcal{L}_{\text{CBC}}$ on training views at 10^{-5} and 10^{-1} from the first iteration; the $\lambda_{\text{KC}} = 10^{-5}$ column of Table 7), on four DeepBlending scenes in the full-view setting (Table 7). Every scene collapses at every share, including the two where the low-weight configuration had improved on 3DGS. The Gaussian count reaches 1.27–2.92M against 0.54–1.13M for 3DGS, and renders fill with long, saturated needle-shaped Gaussians. The rasterizer accumulates the gradient of the whole objective into the view-space statistics that drive cloning and splitting, so the term changes which Gaussians exist. The same happens with $\mathcal{L}_{\text{CBC}}$ at a 10% share on training views together with pseudo-view $\mathcal{L}_{\text{KC}}$ at 20% (six sparse-view scenes): PSNR -1.3 to -6.2 dB, DreamSim $+0.04$ to $+0.38$, and 2.3–4.1$\times$ the Gaussians of 3DGS. Starting every term at iteration 15,000, after densification, leaves the count at 0.89–0.97$\times$ that of 3DGS.

Table 7: $\mathcal{L}_{\text{KC}}$ held at a fixed share X of the loss on training views from the first iteration (DeepBlending, full-view setting). PSNR / DreamSim.

Scene	3DGS	$\lambda_{\text{KC}} = 10^{-5}$	$X = 10\%$	$X = 20\%$	$X = 30\%$
Aquarium-20	23.00 / 0.027	23.76 / 0.025	17.59 / 0.117	11.67 / 0.607	10.14 / 0.702
Shed	28.90 / 0.022	29.51 / 0.020	22.66 / 0.072	13.05 / 0.574	10.70 / 0.710
Museum-1	24.85 / 0.017	23.96 / 0.021	15.92 / 0.142	10.58 / 0.632	9.68 / 0.794
Bridge	19.47 / 0.119	18.76 / 0.132	7.81 / 0.685	7.51 / 0.777	7.90 / 0.782

The harm runs through densification. Table 8 runs the same objectives from the first iteration and from iteration 15,000, on six sparse-view scenes. From the first iteration, the training-view configuration ends with 1.5–2.0× the Gaussians of 3DGS, and adding pseudo-view $\mathcal{L}_{\text{KC}}$ gives 1.3–2.0×. Started at iteration 15,000, after the last densification step, both end at 0.91–1.01×. With pseudo-view $\mathcal{L}_{\text{KC}}$, the late start also lowers DreamSim on 6 of 6 scenes, against 4 of 6 from the first iteration.

Table 8: The same objectives from the first iteration and from iteration 15,000 (12 training views). PSNR / DreamSim, final Gaussian count. *Training-view*: $\mathcal{L}_{\text{wav}}$ with $\mathcal{L}_{\text{KC}}$ and $\mathcal{L}_{\text{CBC}}$ on training views; + *pseudo*: with pseudo-view $\mathcal{L}_{\text{KC}}$ (3×10^{-3}) from iteration 15,000.

Scene	3DGS	Training-view, from 0	Training-view, from 15k	+ pseudo, from 0	+ pseudo, from 15k
DB Bridge	12.57 / 0.364, 1.12M	12.10 / 0.403, 1.73M	12.45 / 0.385, 1.14M	12.32 / 0.394, 1.50M	12.57 / 0.347, 1.02M
DB Shed	18.17 / 0.155, 0.33M	17.38 / 0.151, 0.60M	17.78 / 0.158, 0.32M	17.53 / 0.146, 0.56M	17.67 / 0.124, 0.33M
T&T Family	16.67 / 0.097, 0.89M	17.01 / 0.097, 1.66M	16.73 / 0.103, 0.90M	17.19 / 0.098, 1.59M	17.20 / 0.090, 0.87M
T&T Horse	16.15 / 0.160, 0.63M	16.26 / 0.160, 1.20M	16.20 / 0.161, 0.63M	16.51 / 0.153, 1.18M	15.86 / 0.159, 0.60M
T&T Lighthouse	13.31 / 0.413, 0.76M	13.39 / 0.391, 1.23M	13.54 / 0.395, 0.76M	13.29 / 0.404, 1.16M	13.69 / 0.378, 0.74M
T&T Truck	16.59 / 0.115, 0.97M	16.61 / 0.116, 1.97M	16.45 / 0.117, 0.97M	16.77 / 0.112, 1.90M	16.65 / 0.109, 0.95M

Training views leave almost nothing for the terms to correct. On a converged 3DGS model with its Gaussian set frozen (no densification), 10,000 further iterations with $\mathcal{L}_{\text{KC}}$ at a 10% share on training views, with either color and opacity or every attribute trainable, change PSNR by at most 0.19 dB and DreamSim by at most 0.002 against a photometric-only control of the same length, on the four DeepBlending scenes of Table 7. The collapse in Table 7 therefore needs densification, and without it the term is inert: the photometric loss has already matched the ground truth's band statistics on the views it sees.

Pseudo views need live geometry and a late start. Pseudo-view $\mathcal{L}_{\text{KC}}$ at 3×10^{-3} with only color and opacity trainable, from iteration 2,000, loses up to 1.9 dB (DB Bridge; 3.9 dB at 10^{-2}). With all parameters trainable and from iteration 15,000, it lowers DreamSim on 6 of 6 of the scenes of Table 8 (e.g. Lighthouse 0.413 to 0.359) with PSNR at most 0.2 dB below 3DGS. Detached from geometry at the same start, it loses DreamSim on Bridge, Shed and Truck.

A low weight is inert. At $\lambda_{\text{KC}} = 10^{-5}$ (the λ_{KC} column of Table 7), $\mathcal{L}_{\text{KC}}$ is at most 0.09% of the loss and $\mathcal{L}_{\text{CBC}}$ at 10^{-1} a median 0.007% over the 43 full-view runs of that configuration, so differences from 3DGS at that setting come from $\mathcal{L}_{\text{wav}}$. On training views the photometric loss already brings the rendered band kurtosis to the ground truth's, which leaves $\mathcal{L}_{\text{KC}}$ no residual to act on there; this motivates the pseudo views.

Realized loss shares. Over 59 runs (the 57 of the main table and DeepBlending Street-10 and Yellowhouse-12 trained on all but two images), pseudo-view $\mathcal{L}_{\text{KC}}$ at the fixed 3×10^{-3} realizes a median 10.8% of the loss on the iterations it runs (3.4% on MipNeRF360 flowers to 26.1% on MVImgNet suv), pseudo-view $\mathcal{L}_{\text{CBC}}$ 10.0% (lowest 8.4%, where the cap binds: it does so on 69% of the iterations of Tanks&Temples Lighthouse and 61% of MipNeRF360 bonsai), and $\mathcal{L}_{\text{wav}}$ 22–40% (median 32.4%). The pseudo-view shares estimate the objective of each pseudo iteration from the running average the $\mathcal{L}_{\text{CBC}}$ weight uses. Holding $\mathcal{L}_{\text{KC}}$ at a 10% share by the same rule as $\mathcal{L}_{\text{CBC}}$ requires λ_{KC} between 1.5×10^{-3} and 2.95×10^{-3} and is even with the fixed weight over six scenes and two seeds (PSNR better on 7 of 12, DreamSim on 5 of 12, medians within 0.07 dB and 0.001). At 20%, together with training-view $\mathcal{L}_{\text{CBC}}$ during densification, the reconstruction collapses as above; we did not run the 20% share alone.

Leave-one-out. Table 9 removes one term at a time on six sparse-view scenes (DeepBlending Bridge and Shed, Tanks&Temples Family, Horse, Lighthouse and Truck) at two training seeds, each seed scored against its own 3DGS run. We call a configuration safe when it lowers DreamSim on at least 5 of 6 scenes with a median PSNR difference of at least -0.21 dB and no scene below -1.03 dB, the largest PSNR spread between two same-seed runs with the three terms on training views (WRIVA-ULTRRA; the largest for 3DGS is 0.82 dB). Only the full configuration is safe on both seeds. Every removal produces a PSNR drop beyond -1.03 dB on a DeepBlending scene or loses DreamSim on more than one scene on at least one seed. Per-term DreamSim credit (full configuration better than the configuration without the term) is 10 of 12 scene-seeds for $\mathcal{L}_{\text{CBC}}$, 9 of 12 for $\mathcal{L}_{\text{KC}}$ and 7 of 12 for $\mathcal{L}_{\text{wav}}$; 3DGS itself moves by up to 0.030 DreamSim between seeds, so these counts do not rank the terms. The full 2^3 ablation is in Appendix A.4.

Table 9: Leave-one-out, six sparse-view scenes, two training seeds, against 3DGS of the same seed: scenes with lower DreamSim, median / worst PSNR difference (dB).

Configuration	Seed 0	Seed 1	Both (12)
KC-3DGS (all terms)	5/6, $+0.04$ / -0.21	5/6, $+0.12$ / -0.06	10/12, $+0.08$ / -0.21
without $\mathcal{L}_{\text{wav}}$	5/6, $+0.10$ / -1.56	5/6, $+0.08$ / -0.49	10/12, $+0.08$ / -1.56
without $\mathcal{L}_{\text{KC}}$	5/6, $+0.24$ / -1.31	4/6, $+0.15$ / -0.06	9/12, $+0.21$ / -1.31
without $\mathcal{L}_{\text{CBC}}$	3/6, -0.05 / -1.06	5/6, $+0.04$ / -1.04	8/12, $+0.04$ / -1.06

Photometric loss on pseudo views. Replacing the pseudo-view terms with the photometric loss at a comparable share matches KC-3DGS in PSNR but recovers a third of its DreamSim gain (Appendix A.4).

A.4 Ablation of the Three Terms

Table 10 runs all 2^3 combinations of $\mathcal{L}_{\text{wav}}$, $\mathcal{L}_{\text{KC}}$ and $\mathcal{L}_{\text{CBC}}$ under the protocol of Section 3.6 on the nine MipNeRF360 scenes (sparse, 12 training views, training seed 0). Every configuration resumes from the same 3DGS state at iteration 15,000, so rows differ only in the objective of the second half of training. A term that is off is absent from the objective; the remaining terms keep their weights and share targets. The last row replaces all three terms with the photometric loss $0.8\,L_1 + 0.2\,(1 - \text{SSIM})$ between the pseudo-view render and the anchor ground truth, held at a 20% share of the loss by the rule of Eq. (7), which is the share the two pseudo-view terms realize together. It tests whether pseudo views help through their extra supervision alone: the anchor image is offset from the pseudo camera by $0.2\,\bar{d}$, so a pixel-wise loss penalizes the parallax that a band statistic ignores.

Pseudo-view $\mathcal{L}_{\text{KC}}$ carries the perceptual gain: alone it lowers DreamSim on 9 of 9 scenes, and every configuration that contains it lowers DreamSim on 8 or 9 of 9. Pseudo-view $\mathcal{L}_{\text{CBC}}$ alone costs PSNR (0.10 dB on average, 0.65 dB at worst) for a small DreamSim gain on 5 of 9 scenes; added to $\mathcal{L}_{\text{KC}}$ it enlarges the DreamSim reduction from 0.0094 to 0.0126 and puts PSNR above 3DGS on all 9 scenes. $\mathcal{L}_{\text{wav}}$ alone raises PSNR slightly on every scene and leaves DreamSim unchanged; on MipNeRF360 it adds nothing measurable to the two pseudo-view terms, while on DeepBlending removing it causes a PSNR collapse (Table 9), so we keep it.

The photometric control matches the full configuration in PSNR ($+0.25$ dB) but recovers only a third of its DreamSim reduction (0.0040 on 6 of 9 scenes, against 0.0120 on 8 of 9), and improves SSIM and LPIPS on 7 of 9 scenes instead of 9. Pixel supervision on pseudo views thus improves reconstruction accuracy, while the perceptual gain requires the band statistics, which tolerate the offset between the pseudo camera and the anchor image. One seed and one dataset; the control's 20% share and the two pseudo-view terms' combined share (about 19%) are comparable, not identical.

Table 10: 2^3 ablation and the pseudo-view photometric control, MipNeRF360 (9 scenes), sparse 12 views, all rows warm-started from the same 3DGS state at iteration 15,000. Means over scenes; last column: scenes with lower DreamSim than 3DGS, and the worst per-scene PSNR difference (dB).

$\mathcal{L}_{\text{wav}}$	$\mathcal{L}_{\text{KC}}$	$\mathcal{L}_{\text{CBC}}$	PSNR ↑	SSIM ↑	LPIPS ↓	DreamSim ↓	DreamSim improved, worst ΔPSNR
			18.32	0.5007	0.4208	0.1491	–
✓			18.37	0.5031	0.4204	0.1497	4/9, +0.02
	✓		18.51	0.5104	0.4139	0.1397	9/9, −0.04
		✓	18.22	0.5065	0.4167	0.1466	5/9, −0.65
✓	✓		18.55	0.5105	0.4139	0.1390	9/9, +0.02
✓		✓	18.23	0.5085	0.4158	0.1456	6/9, −0.58
	✓	✓	18.57	0.5136	0.4124	0.1365	9/9, +0.04
✓	✓	✓	18.58	0.5144	0.4120	0.1371	8/9, +0.07
photometric on pseudo views			18.57	0.5043	0.4207	0.1452	6/9, −0.14

A.5 Number of Training Views

Table 11: KC-3DGS minus 3DGS on MipNeRF360 (9 scenes) by number of training views, same 246 test views in every row; *full*: every 8th image held out, the rest train. Every KC-3DGS run resumes from its 3DGS run's state at iteration 15,000. Means over scenes, with the number of scenes where KC-3DGS is better in parentheses; *Gaussians*: range over scenes of the final count relative to 3DGS.

Views	Scenes	ΔPSNR	ΔSSIM	ΔLPIPS	ΔDreamSim	worst ΔPSNR	Gaussians
12	9	+0.26 (9/9)	+0.0137 (9/9)	−0.0088 (9/9)	−0.0120 (8/9)	+0.07	0.95–0.99
24	9	+0.35 (9/9)	+0.0140 (9/9)	−0.0052 (9/9)	−0.0042 (8/9)	+0.15	0.93–0.99
50	9	+0.07 (6/9)	+0.0039 (4/9)	+0.0056 (1/9)	+0.0003 (3/9)	−0.35	0.91–0.98
full	9	−0.77 (0/9)	−0.0154 (0/9)	+0.0242 (0/9)	+0.0026 (0/9)	−1.45	0.90–0.96

A.6 Training-Seed Spread

Table 12 repeats the MipNeRF360 comparison of Table 1 at three training seeds. Each seed has its own 3DGS run, and KC-3DGS resumes from that run's state at iteration 15,000, so every row is a paired comparison at a fixed seed. The split (training and test views) is the same at every seed.

The difference from 3DGS is consistent across seeds: $+0.26$ to $+0.27$ dB PSNR, better on 8 or 9 of 9 scenes, with SSIM and LPIPS better on 9 of 9 at every seed and DreamSim better on 8, 8 and 9 of 9. It is larger than what 3DGS does on its own between seeds: per scene, 3DGS at seeds 1 and 2 differs by a median of 0.01 dB and 0.0026 DreamSim, while at those two seeds KC-3DGS differs from 3DGS by a median of 0.27 dB and 0.0078, with the same sign at both seeds on 8 of 9 scenes. The exception is kitchen, where KC-3DGS loses 0.07–0.12 dB PSNR at seeds 1 and 2 with DreamSim differences of $+0.0042$ and -0.0001, and gains 0.14 dB at seed 0. The seed-0 row is warm-started while Table 1 trains from scratch; the two KC-3DGS runs differ by a median of 0.13 dB and 0.0028 DreamSim per scene, within the variation between two 3DGS runs at the same seed (median 0.07 dB, up to 0.56 dB).

Table 12: KC-3DGS minus 3DGS on MipNeRF360 (sparse 12 views) at three training seeds, mean over scenes, with the number of scenes where KC-3DGS is better in parentheses.

Seed	Scenes	ΔPSNR	ΔSSIM	ΔLPIPS	ΔDreamSim
0	9	+0.26 (9/9)	+0.0137 (9/9)	−0.0088 (9/9)	−0.0120 (8/9)
1	9	+0.26 (8/9)	+0.0132 (9/9)	−0.0079 (9/9)	−0.0093 (8/9)
2	9	+0.27 (8/9)	+0.0137 (9/9)	−0.0078 (9/9)	−0.0126 (9/9)
mean ± sd		+0.26 ± 0.01	+0.0135 ± 0.0003	-0.0082 ± 0.0006	-0.0113 ± 0.0018

A.7 KC-3DGS Added to FDS-GS

KC-3DGS only adds loss terms, so it can be added to another 3DGS method without modifying its representation or schedule. We add it to FDS-GS (Zeng et al., 2025), a 3DGS method with per-Gaussian color that reparameterizes Gaussian scale and replaces density control with a frequency-aware one, on the nine MipNeRF360 scenes used for the ablation and the seed study (sparse, 12 training views, image scale 0.25). Both variants run the released FDS-GS code and objective for its 30,000 iterations through our split files. The KC-3DGS variant adds the three terms of Section 3.6 with the same weights, share targets and pseudo-view jitter as on FasterGS, starting at iteration 15,000, when FDS-GS's densification ends. The two variants are separate runs; FDS-GS fixes its random seed to 0, so they share the same initialization.

KC-3DGS improves FDS-GS on all four metrics on every scene (Table 13): PSNR by 0.38 dB (0.11–0.61 dB) and DreamSim by 0.0116 (9%). On FasterGS the same nine scenes gain 0.26 dB and 0.0120 DreamSim at seed 0 (Table 12). The realized $\mathcal{L}_{\text{KC}}$ share varies from 4% (flowers, garden) to 13% (room) under the fixed weight; the gain does not track it, and the two 4% scenes have among the largest relative DreamSim reductions. One seed; the run-to-run spread of FDS-GS was not measured, and the smallest per-scene differences (counter and kitchen DreamSim, 0.001) are below the spread we measured for KC-3DGS on FasterGS (0.006).

Table 13: FDS-GS without and with KC-3DGS (+KC), MipNeRF360 (9 scenes), sparse 12 views. Per-scene means over test views; bold: better of the pair. Last column: realized share of pseudo-view $\mathcal{L}_{\text{KC}}$ in the loss, averaged over the KC-3DGS phase.

		PSNR↑		SSIM↑		LPIPS↓		DreamSim↓		
Scene	Test views	FDS-GS	+KC	FDS-GS	+KC	FDS-GS	+KC	FDS-GS	+KC	$\mathcal{L}_{\text{KC}}$ share
bicycle	25	16.91	**17.28**	0.3136	**0.3324**	0.4883	**0.4826**	0.2123	**0.1950**	7%
bonsai	37	19.95	**20.56**	0.7366	**0.7446**	0.2932	**0.2876**	0.0835	**0.0774**	11%
counter	30	19.95	**20.07**	0.6618	**0.6721**	0.3362	**0.3302**	0.1191	**0.1178**	10%
flowers	22	13.93	**14.28**	0.2524	**0.2657**	0.5220	**0.5121**	0.1310	**0.1159**	4%
garden	24	20.22	**20.72**	0.5918	**0.6211**	0.2989	**0.2885**	0.0533	**0.0412**	4%
kitchen	35	21.37	**21.80**	0.7866	**0.7934**	0.2274	**0.2226**	0.0390	**0.0379**	11%
room	39	21.04	**21.40**	0.7519	**0.7621**	0.3208	**0.3162**	0.1166	**0.1103**	13%
stump	16	17.69	**18.23**	0.3054	**0.3269**	0.5294	**0.5217**	0.1772	**0.1621**	8%
treehill	18	16.07	**16.21**	0.3370	**0.3531**	0.5340	**0.5309**	0.2294	**0.1992**	7%
Mean		18.57	**18.95**	0.5263	**0.5413**	0.3945	**0.3881**	0.1290	**0.1174**	8%
Difference		+0.38 (9/9)		+0.0149 (9/9)		−0.0064 (9/9)		−0.0116 (9/9)		

A.8 KC-3DGS Added to FSGS

FSGS (Zhu et al., 2024) adds monocular depth supervision, its own pseudo views and Gaussian unpooling to 3DGS, and keeps per-Gaussian color. We add KC-3DGS to its released code on every scene of MipNeRF360, Tanks and Temples and DeepBlending (sparse, 12 training views, seed 0). FSGS's released schedule is 10,000 iterations with densification to its end; both variants run it unchanged and then train 10,000 more iterations without densification. The KC-3DGS variant adds the three terms of Section 3.6 from 10,000 on with the weights, share targets and pseudo-view jitter used on FasterGS, and the control trains the same length under FSGS's objective alone.

On MipNeRF360, KC-3DGS lowers DreamSim and raises SSIM on all nine scenes and raises PSNR by 0.16 dB (6 of 9; Table 15). The three PSNR losses, treehill (-0.21 dB), bonsai (-0.05 dB) and bicycle (-0.02 dB), are within the run-to-run variation we measured for KC-3DGS on FasterGS (0.23 dB). LPIPS is slightly worse, by 0.006, and improves on only 3 of 9 scenes. The same pattern holds on the other datasets (Table 14): DreamSim improves on 33 of 42 scenes and LPIPS on 14. On FasterGS and FDS-GS, LPIPS improves together with DreamSim and SSIM; on FSGS it does not. We have not isolated why; FSGS already supervises its own pseudo views, so the two sets of pseudo-view terms act together. One seed; the run-to-run spread of FSGS was not measured.

Table 14: FSGS without and with KC-3DGS (+KC), sparse 12 views. Means over scenes; bold: better of the pair; second row per dataset: scenes where +KC is better.

		PSNR↑		SSIM↑		LPIPS↓		DreamSim↓	
Dataset	Scenes	FSGS	+KC	FSGS	+KC	FSGS	+KC	FSGS	+KC
MipNeRF360	9	19.23	**19.40**	0.5406	**0.5525**	0.4284	0.4345	0.1376	**0.1315**
		6/9		9/9		3/9		9/9	
Tanks&Temples	16	16.58	**16.61**	0.5936	**0.6018**	0.4331	0.4356	0.1983	**0.1934**
		10/16		12/16		7/16		12/16	
DeepBlending	17	20.34	20.33	0.7077	**0.7087**	0.3945	0.3980	0.2004	**0.1925**
		8/17		8/17		4/17		12/17	
Scenes improved	42	24/42		29/42		14/42		33/42	

Table 15: FSGS without and with KC-3DGS (+KC), MipNeRF360 (9 scenes), sparse 12 views. Per-scene means over test views; bold: better of the pair.

		PSNR↑		SSIM↑		LPIPS↓		DreamSim↓	
Scene	Test views	FSGS	+KC	FSGS	+KC	FSGS	+KC	FSGS	+KC
bicycle	25	17.55	17.53	0.3451	**0.3565**	0.5189	0.5282	0.1752	**0.1742**
bonsai	37	21.75	21.69	0.7718	**0.7740**	0.2969	0.2977	0.0851	**0.0812**
counter	30	20.44	**20.51**	0.6827	**0.6922**	0.3292	**0.3286**	0.1125	**0.1117**
flowers	22	14.64	**15.06**	0.2615	**0.2724**	0.6099	0.6157	0.1990	**0.1810**
garden	24	20.50	**20.71**	0.5680	**0.5821**	0.3926	0.4132	0.0503	**0.0478**
kitchen	35	21.65	**21.81**	0.7877	**0.7933**	0.2288	**0.2251**	0.0392	**0.0370**
room	39	22.83	**23.01**	0.7873	**0.7974**	0.3070	**0.3040**	0.0960	**0.0914**
stump	16	17.40	**18.12**	0.3018	**0.3299**	0.5723	0.5809	0.2325	**0.2296**
treehill	18	16.35	16.14	0.3593	**0.3750**	0.5999	0.6176	0.2482	**0.2295**
Mean		19.23	**19.40**	0.5406	**0.5525**	0.4284	0.4345	0.1376	**0.1315**
Difference		+0.16 (6/9)		+0.0120 (9/9)		+0.0062 (3/9)		−0.0061 (9/9)	

A.9 KC-3DGS Added to LAP

LAP (Lavi et al., 2026) groups Gaussians into Laplacian-pyramid frequency levels and keeps per-Gaussian color. We run its released code through our split files. Its released recipe is 30k iterations with densification to its end; both variants run that recipe unchanged and then train 15,000 more iterations without densification. The KC-3DGS variant adds the three terms of Section 3.6 from iteration 30,000, after densification; the control continues under LAP's own objective. Results on MipNeRF360 are in Table 2.

A.10 KC-3DGS in a Baseline with a Shared Color Network

In FasterGS and FDS-GS every Gaussian carries its own color, so the gradient of a term on a rendered view reaches exactly the Gaussians that view sees. DNGaussian (Li et al., 2024) instead computes color from a hash-grid encoding and two MLPs shared by all Gaussians, evaluated at each Gaussian's position and viewing direction. The terms then act on the Gaussians only through that shared network: a gradient from one pseudo view changes the color of every Gaussian that uses the same grid cells and weights.

Protocol. We start from the released DNGaussian recipe trained for its 6,000 iterations with densification to the end (first row of Table 16), and continue for 3,000 iterations with the Gaussian set fixed (no densification or pruning) and every parameter trained, including the color network. The control continues under DNGaussian's own objective (photometric, depth and regularization terms). The KC-3DGS variant adds the three terms from the first continued iteration, with the weights, share targets and pseudo-view jitter of Section 3.6. MipNeRF360, 9 scenes, 12 training views, seed 0.

Table 16: DNGaussian continued for 3,000 iterations with a fixed Gaussian set, without (control) and with KC-3DGS. MipNeRF360 (9 scenes), sparse 12 views. Top: test metrics, means over scenes; in parentheses, scenes where the row is better than the one it is compared with. Bottom: the scene representation, means over scenes (details in the text).

	PSNR↑	SSIM↑	LPIPS↓	DreamSim↓
DNGaussian, 6k	16.36	0.368	0.670	0.418
+3k control	16.28	0.369	0.651	0.391
+3k KC-3DGS	16.25	0.359	0.670	0.425
KC-3DGS − control	−0.03 (5/9)	−0.009 (3/9)	+0.019 (0/9)	+0.034 (0/9)

	6k	+3k control	+3k KC-3DGS
Median Gaussian displacement (% of scene half-extent)	0	0.09	0.12
Gaussians with opacity < 0.05 (%)	3.0	5.5	9.2
Mean color shift from the 6k model	0	0.096	0.169
Color difference to 8 nearest neighbors	0.188	0.223	0.186
Color spread over the scene (std)	0.383	0.387	0.331
Training-view PSNR (dB)	22.68	23.43	21.48
Test-view finest-band detail energy / GT	0.246	0.273	0.235

Result. Continuing DNGaussian improves LPIPS and DreamSim on all 9 scenes; adding KC-3DGS cancels that gain and is worse than the control on LPIPS and DreamSim on all 9 (Table 16, top). PSNR is unchanged within 0.03 dB.

What changes in the scene. Both variants keep the same Gaussians in the same order, so we compare them Gaussian by Gaussian. Color is the network's output at each Gaussian's center, viewed from every test camera and averaged over cameras; neighbor differences use 20,000 Gaussians with opacity above 0.1 and the same neighbor sets in every variant; detail energy is the wavelet (db3) finest-level energy of the test renders divided by that of the ground truth. Geometry barely moves in either variant (0.1% of the scene half-extent), and near-transparent Gaussians rise from 5.5% to 9.2%. The difference is in color. KC-3DGS changes the shared color network 1.8× as much as the control, and the result is flatter: the color difference between neighboring Gaussians stays at the 6k level where the control raises it by 19%, the color spread over the scene falls 13% (lower than the control on 8 of 9 scenes), and the test renders carry less fine detail than the 6k model, while the control adds detail. The training-view PSNR falls 2.0 dB below the control's, so the terms' gradient through the shared network works against the fit to the training views rather than refining it. PSNR tolerates the flatter images; LPIPS and DreamSim do not.

Scope. With per-Gaussian color the same terms help: FDS-GS with KC-3DGS improves PSNR by 0.38 dB and DreamSim by 0.012 on all 9 MipNeRF360 scenes under the same weights (Appendix A.7). KC-3DGS therefore assumes that the rendered color is controlled per Gaussian; when it passes through a network shared across the scene, the terms have no per-Gaussian handle and should not be applied as-is. We tested one such method, one dataset and one seed, and did not attribute the effect to individual terms.

A.11 Sparse-View Failures that KC-3DGS Removes

Figure 7 shows the characteristic sparse-view failures of 3DGS on held-out views: large floaters that render as flat white or dark blobs (Tanks&Temples M60, MipNeRF360 treehill and stump, DeepBlending Boats), a color cast over a whole object (Francis), and a sky region filled with the wrong color (Lighthouse). They arise where the training views constrain a region weakly, so Gaussians fitted to one view occlude or tint it from another. KC-3DGS removes or shrinks each of them. The panels are selected, not random: per scene, the test view with the largest reduction in mean absolute error and, within it, the window with the largest local reduction. The all-scene numbers, including the scenes where KC-3DGS is worse, are in Table 1; the whole-view PSNR of the shown views rises by 0.9–5.9 dB (treehill 8.95 to 14.81 dB, Francis 16.68 to 21.39 dB).



Figure 7: **Sparse-view failures of 3DGS removed by KC-3DGS** (12 training views). Left: ground-truth test view with the crop outlined. Right: crops of the ground truth, 3DGS and KC-3DGS with crop PSNR. Per scene, the view and crop with the largest error reduction were selected; see Table 1 for all scenes.

A.12 Human Preference Study

DreamSim and LPIPS are learned proxies for perceived quality, so we also ask observers. Each trial shows a ground-truth crop between the KC-3DGS and 3DGS renders of the same held-out view and asks which render is closer to the reference photo: a full-reference forced choice, the judgment the metrics approximate. Crops are shown at native resolution (512 px square, 400 px on smaller views) at an integer number of screen pixels per image pixel, with the full view as a thumbnail; left and right are randomized per trial, and the answer buttons enable after 3 s.

The 44 primary pairs are one view per scene of MipNeRF360 (9), Tanks&Temples (16) and DeepBlending (19) at 12 training views; the 2 DeepBlending scenes too small for a 12-view split enter with their full-view runs. Views and crop positions were drawn by a seeded generator before any image was inspected. MVImgNet is left out because its views are smaller than the 400 px crop, and WRIVA-ULTRRA because the study was built before the scenes of Table 1. Each observer answers one of 22 forms of 30 trials: 20 primary pairs, 5 positive-control pairs of FSGS against 3DGS (16 Tanks&Temples pairs), and 5 attention pairs of 3DGS at 12 views against 3DGS trained on the full view set.

The analysis is pre-registered. Incomplete sessions and observers answering fewer than 4 of 5 attention pairs correctly are excluded. The statistic is the share of primary trials in which KC-3DGS is chosen, with a 95% interval from a bootstrap over observers and scenes jointly (10,000 resamples). An interval inside 40–60% means no perceptible difference, one above it that KC-3DGS is preferred, one below it that 3DGS is preferred, and anything else is inconclusive. The study counts as sensitive only if FSGS is chosen on at least 65% of control trials.

7 observers started, 1 did not finish and 0 failed the attention check, leaving 6, who together saw 40 of the 44 primary pairs (2–4 observers per pair). FSGS was chosen on 73.3% of 30 control trials and KC-3DGS on 59.2% of 120 primary trials (95% interval 45.2–72.9%). The outcome under the rule above is inconclusive: with 6 observers the interval is wide, and the study establishes no preference in either direction. We report it as collected.

A.13 Per-Scene Results

Figure 8 plots, for each scene of Table 1, the PSNR difference between KC-3DGS and 3DGS against the DreamSim difference. Over all 57 scenes, KC-3DGS improves both on 37, DreamSim alone on 11, PSNR alone on 3, and neither on 6; the largest PSNR loss is 0.93 dB (DeepBlending Bedroom). On most scenes the DreamSim gain comes without a PSNR loss. Table 17 lists every scene.

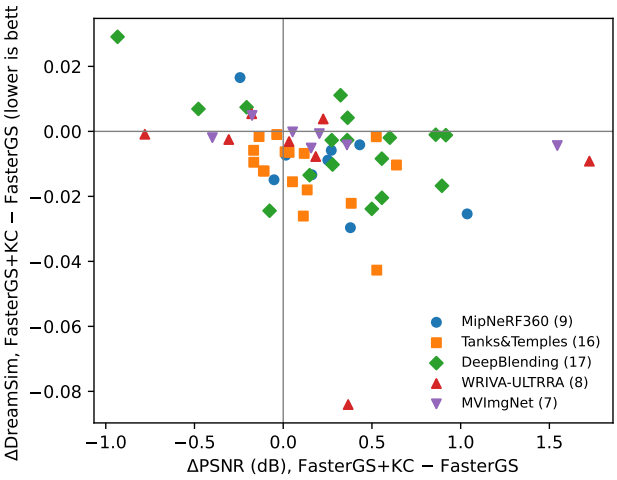

Figure 8: Per-scene difference, KC-3DGS minus 3DGS, 12 training views. Lower right is better on both metrics.

Table 17: Per-scene results, 12 training views: FasterGS / FasterGS+KC (difference, bold where FasterGS+KC is better). *Test*: number of test views; †: 3–4 test views, less reliable.

Scene	Test	PSNR ↑	SSIM ↑	LPIPS ↓	DreamSim ↓
MipNeRF360					
bicycle	25	17.30 / 17.24 (-0.05)	0.3069 / 0.3178 (**+0.0109**)	0.5001 / 0.4944 (**-0.0057**)	0.2065 / 0.1916 (**-0.0149**)
bonsai	37	19.86 / 19.61 (-0.24)	0.7134 / 0.7129 (-0.0006)	0.3348 / 0.3353 (+0.0005)	0.1058 / 0.1223 (+0.0166)
counter	30	19.66 / 19.91 (**+0.25**)	0.6450 / 0.6610 (**+0.0160**)	0.3535 / 0.3453 (**-0.0082**)	0.1219 / 0.1130 (**-0.0088**)
flowers	22	13.55 / 13.71 (**+0.16**)	0.2223 / 0.2336 (**+0.0112**)	0.5509 / 0.5408 (**-0.0101**)	0.1779 / 0.1645 (**-0.0133**)
garden	24	19.94 / 20.37 (**+0.43**)	0.5574 / 0.5820 (**+0.0246**)	0.3171 / 0.3057 (**-0.0115**)	0.0502 / 0.0461 (**-0.0041**)
kitchen	35	21.36 / 21.38 (**+0.01**)	0.7556 / 0.7647 (**+0.0091**)	0.2567 / 0.2460 (**-0.0106**)	0.0517 / 0.0444 (**-0.0073**)
room	39	21.44 / 21.71 (**+0.27**)	0.7371 / 0.7501 (**+0.0130**)	0.3378 / 0.3313 (**-0.0065**)	0.1123 / 0.1065 (**-0.0058**)
stump	16	17.55 / 17.93 (**+0.38**)	0.2771 / 0.2964 (**+0.0193**)	0.5550 / 0.5471 (**-0.0079**)	0.2257 / 0.1961 (**-0.0296**)
treehill	18	14.56 / 15.60 (**+1.04**)	0.2987 / 0.3214 (**+0.0226**)	0.5641 / 0.5569 (**-0.0071**)	0.2773 / 0.2520 (**-0.0254**)
Tanks&Temples					
Auditorium	56	17.82 / 17.65 (-0.17)	0.6863 / 0.6884 (**+0.0020**)	0.4392 / 0.4348 (**-0.0045**)	0.2380 / 0.2321 (**-0.0058**)
Ballroom	45	15.84 / 15.73 (-0.11)	0.4847 / 0.4890 (**+0.0043**)	0.4309 / 0.4191 (**-0.0118**)	0.2001 / 0.1879 (**-0.0122**)
Barn	52	16.37 / 16.26 (-0.11)	0.5514 / 0.5644 (**+0.0130**)	0.4506 / 0.4449 (**-0.0057**)	0.2703 / 0.2581 (**-0.0122**)
Church	81	14.39 / 14.44 (**+0.05**)	0.4753 / 0.4857 (**+0.0104**)	0.5016 / 0.4923 (**-0.0093**)	0.2933 / 0.2778 (**-0.0155**)
Courthouse	70	12.77 / 12.60 (-0.17)	0.4588 / 0.4704 (**+0.0116**)	0.5479 / 0.5396 (**-0.0084**)	0.4115 / 0.4020 (**-0.0095**)
Family	19	16.67 / 17.31 (**+0.64**)	0.6462 / 0.6643 (**+0.0181**)	0.3359 / 0.3249 (**-0.0110**)	0.0971 / 0.0868 (**-0.0103**)
Francis	33	16.93 / 17.46 (**+0.53**)	0.6946 / 0.7134 (**+0.0189**)	0.4620 / 0.4509 (**-0.0111**)	0.2223 / 0.1796 (**-0.0427**)
Horse	26	16.15 / 16.11 (-0.04)	0.6544 / 0.6619 (**+0.0075**)	0.3657 / 0.3632 (**-0.0025**)	0.1604 / 0.1594 (**-0.0010**)
Ignatius	33	14.88 / 14.89 (**+0.01**)	0.4584 / 0.4753 (**+0.0169**)	0.4314 / 0.4237 (**-0.0077**)	0.1519 / 0.1456 (**-0.0063**)
Lighthouse	35	13.31 / 13.43 (**+0.11**)	0.6128 / 0.6250 (**+0.0122**)	0.4698 / 0.4608 (**-0.0090**)	0.4129 / 0.3869 (**-0.0261**)
M60	24	15.12 / 15.51 (**+0.38**)	0.5521 / 0.5699 (**+0.0178**)	0.4697 / 0.4585 (**-0.0112**)	0.2926 / 0.2705 (**-0.0221**)
Meetingroom	47	15.30 / 15.16 (-0.14)	0.5286 / 0.5303 (**+0.0017**)	0.5090 / 0.5035 (**-0.0055**)	0.3245 / 0.3229 (**-0.0016**)
Panther	28	16.43 / 16.95 (**+0.52**)	0.6176 / 0.6271 (**+0.0095**)	0.4521 / 0.4617 (+0.0096)	0.2547 / 0.2531 (**-0.0016**)
Playground	32	16.54 / 16.67 (**+0.14**)	0.4447 / 0.4713 (**+0.0265**)	0.4716 / 0.4583 (**-0.0133**)	0.2063 / 0.1883 (**-0.0180**)
Train	53	13.83 / 13.86 (**+0.03**)	0.4563 / 0.4627 (**+0.0064**)	0.4932 / 0.4824 (**-0.0109**)	0.2330 / 0.2266 (**-0.0065**)
Truck	32	16.59 / 16.70 (**+0.12**)	0.5951 / 0.6093 (**+0.0142**)	0.3756 / 0.3696 (**-0.0060**)	0.1148 / 0.1080 (**-0.0068**)
DeepBlending					
Aquarium-20†	3	23.51 / 24.42 (**+0.92**)	0.8515 / 0.8580 (**+0.0065**)	0.1659 / 0.1611 (**-0.0048**)	0.0269 / 0.0258 (**-0.0012**)
Bedroom	25	17.73 / 16.79 (-0.93)	0.6539 / 0.6447 (-0.0092)	0.4645 / 0.4795 (+0.0150)	0.3285 / 0.3576 (+0.0291)
Boats	13	14.59 / 15.09 (**+0.50**)	0.6263 / 0.6424 (**+0.0160**)	0.4533 / 0.4383 (**-0.0151**)	0.3098 / 0.2859 (**-0.0238**)
Bridge	9	12.57 / 12.36 (-0.21)	0.4322 / 0.4338 (**+0.0016**)	0.5318 / 0.5565 (+0.0246)	0.3640 / 0.3715 (+0.0074)
CreepyAttic	31	18.22 / 18.54 (**+0.32**)	0.5243 / 0.5522 (**+0.0280**)	0.5156 / 0.5154 (**-0.0002**)	0.3835 / 0.3946 (+0.0111)
DrJohnson	33	16.54 / 16.81 (**+0.28**)	0.5175 / 0.5326 (**+0.0151**)	0.5644 / 0.5613 (**-0.0030**)	0.4737 / 0.4635 (**-0.0102**)
Hugo-1†	3	24.37 / 25.23 (**+0.86**)	0.8500 / 0.8569 (**+0.0069**)	0.1567 / 0.1530 (**-0.0037**)	0.0227 / 0.0217 (**-0.0010**)
Library	28	16.48 / 16.84 (**+0.36**)	0.5447 / 0.5640 (**+0.0193**)	0.5568 / 0.5512 (**-0.0056**)	0.3777 / 0.3750 (**-0.0027**)
Lumber	13	12.97 / 13.25 (**+0.27**)	0.4556 / 0.4708 (**+0.0152**)	0.4738 / 0.4625 (**-0.0113**)	0.1968 / 0.1941 (**-0.0027**)
Museum-1†	4	21.38 / 21.74 (**+0.36**)	0.7877 / 0.7910 (**+0.0033**)	0.1834 / 0.1835 (+0.0001)	0.0249 / 0.0291 (+0.0042)
Museum-2†	4	26.71 / 27.31 (**+0.60**)	0.8746 / 0.8876 (**+0.0129**)	0.1273 / 0.1180 (**-0.0094**)	0.0203 / 0.0184 (**-0.0019**)
NightSnow	12	24.76 / 24.28 (-0.48)	0.7453 / 0.7403 (-0.0051)	0.3487 / 0.3472 (**-0.0015**)	0.1153 / 0.1222 (+0.0069)
Playroom	28	17.22 / 17.78 (**+0.56**)	0.6491 / 0.6687 (**+0.0195**)	0.4810 / 0.4684 (**-0.0126**)	0.3108 / 0.3024 (**-0.0084**)
Ponche	7	19.58 / 20.14 (**+0.56**)	0.6929 / 0.7194 (**+0.0265**)	0.3479 / 0.3277 (**-0.0203**)	0.1159 / 0.0955 (**-0.0204**)
SaintAnne	15	17.11 / 17.26 (**+0.15**)	0.5687 / 0.5910 (**+0.0223**)	0.5098 / 0.5010 (**-0.0088**)	0.2657 / 0.2522 (**-0.0135**)
Shed	13	18.17 / 18.09 (-0.08)	0.7939 / 0.7993 (**+0.0054**)	0.3349 / 0.3329 (**-0.0020**)	0.1553 / 0.1309 (**-0.0244**)
Tree-18†	3	20.49 / 21.38 (**+0.89**)	0.7948 / 0.8166 (**+0.0219**)	0.1940 / 0.1740 (**-0.0200**)	0.0418 / 0.0251 (**-0.0167**)
WRIVA-ULTRRA					
ImageDensity A01 GoPro	7	15.48 / 15.67 (**+0.18**)	0.2881 / 0.3074 (**+0.0193**)	0.5540 / 0.5674 (+0.0135)	0.2847 / 0.2769 (**-0.0077**)
ImageDensity dev A01	5	12.20 / 12.02 (-0.18)	0.2826 / 0.2906 (**+0.0079**)	0.6026 / 0.6256 (+0.0229)	0.4243 / 0.4297 (+0.0054)
ImageDensity test A09†	4	14.20 / 15.92 (**+1.73**)	0.4095 / 0.4251 (**+0.0156**)	0.4649 / 0.4515 (**-0.0134**)	0.1927 / 0.1835 (**-0.0092**)
CameraModels dev A01	6	12.88 / 13.25 (**+0.37**)	0.3883 / 0.4244 (**+0.0361**)	0.5687 / 0.5530 (**-0.0156**)	0.4400 / 0.3559 (**-0.0841**)
CameraModels test A09†	3	11.91 / 12.13 (**+0.23**)	0.2176 / 0.2234 (**+0.0059**)	0.6601 / 0.6615 (+0.0014)	0.4980 / 0.5019 (+0.0038)
VaryingAltitudes test A09†	3	12.64 / 11.86 (-0.78)	0.2154 / 0.2220 (**+0.0066**)	0.6592 / 0.6519 (**-0.0073**)	0.4959 / 0.4950 (**-0.0009**)
PTZ A08	41	13.66 / 13.35 (-0.31)	0.4765 / 0.4843 (**+0.0078**)	0.6303 / 0.6445 (+0.0142)	0.5910 / 0.5885 (**-0.0025**)
PTZ A06 SingleZone	22	13.31 / 13.35 (**+0.03**)	0.2903 / 0.2944 (**+0.0041**)	0.5969 / 0.5937 (**-0.0033**)	0.3627 / 0.3595 (**-0.0031**)
MVImgNet					
bench†	3	27.17 / 27.38 (**+0.20**)	0.8790 / 0.8853 (**+0.0063**)	0.1034 / 0.0975 (**-0.0059**)	0.0077 / 0.0070 (**-0.0007**)
bicycle†	3	26.30 / 26.66 (**+0.36**)	0.8788 / 0.8801 (**+0.0013**)	0.1061 / 0.1010 (**-0.0051**)	0.0152 / 0.0110 (**-0.0042**)
car†	3	27.21 / 26.82 (-0.40)	0.9226 / 0.9179 (-0.0047)	0.1101 / 0.1125 (+0.0023)	0.0360 / 0.0340 (**-0.0019**)
chair†	3	28.02 / 28.17 (**+0.16**)	0.9174 / 0.9214 (**+0.0040**)	0.1145 / 0.1123 (**-0.0021**)	0.0150 / 0.0099 (**-0.0051**)
ladder†	3	21.25 / 21.30 (**+0.05**)	0.7326 / 0.7357 (**+0.0031**)	0.1687 / 0.1681 (**-0.0006**)	0.0112 / 0.0111 (**-0.0002**)
suv†	3	27.53 / 27.35 (-0.18)	0.9248 / 0.9192 (-0.0056)	0.0805 / 0.0865 (+0.0060)	0.0315 / 0.0364 (+0.0049)
table†	3	24.64 / 26.18 (**+1.54**)	0.8867 / 0.8919 (**+0.0052**)	0.1105 / 0.1073 (**-0.0032**)	0.0171 / 0.0128 (**-0.0044**)

A.14 Hyperparameter Sensitivity of the Training-View Configuration

Table 18 varies the three loss weights one at a time with all terms on the training views from the first iteration, including densification, rather than under the protocol of Section 3.6. Raising λ_{KC} or λ_{CBC} above the default degrades every metric, the collapse that motivates starting the terms after densification (Appendix A.3); the contribution of each term under the protocol is in Tables 9 and 10.

Table 18: Sensitivity to the loss weights with all terms on the training views from the first iteration (not the protocol of Section 3.6), FasterGS. Default in **bold**. Each block varies one weight with the others at the default (kc_lambda=10^{-5}, cbc_lambda=0.01, wavelet_lambda=0.001). Rows average three nested image pools of one WRIVA-ULTRRA site, a capture separate from the scenes of Table 1, except kc_lambda= 10^{-3}, for which only the 150-image pool has results.

kc_lambda	cbc_lambda	wavelet_lambda	DreamSim↓	PSNR↑	SSIM↑	LPIPS↓
Varying kc_lambda						
10^{-5}	0.01	0.001	**0.279**	**16.27**	**0.418**	**0.422**
10^{-4}	0.01	0.001	0.490	13.18	0.322	0.519
10^{-3}	0.01	0.001	0.767	10.43	0.260	0.677
Varying cbc_lambda						
10^{-5}	0.001	0.001	0.294	15.90	0.410	0.432
10^{-5}	0.01	0.001	**0.279**	**16.27**	**0.418**	**0.422**
10^{-5}	1.0	0.001	0.465	13.41	0.339	0.525
Varying wavelet_lambda						
10^{-5}	0.01	0.001	**0.279**	**16.27**	**0.418**	**0.422**
10^{-5}	0.01	0.01	0.293	15.93	0.408	0.434
10^{-5}	0.01	0.1	0.298	16.07	0.416	0.432
10^{-5}	0.01	1.0	0.338	15.61	0.397	0.452

A.15 Toy Example of Wavelets

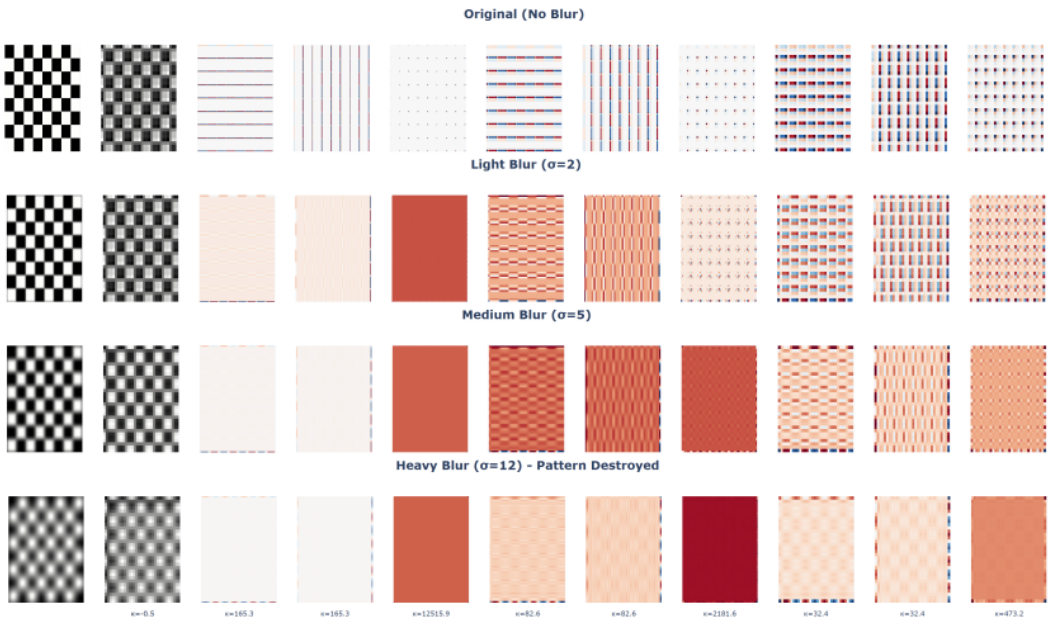

Figure 9: **Wavelet decomposition of a synthetic checkerboard under progressive blur.** A sharp checkerboard pattern (top row) is degraded with Gaussian blur of increasing strength (σ=2, 5, 12). For each blur level, we show the low-frequency approximation (LL) and detail sub-bands capturing horizontal (c_H), vertical (c_V), and diagonal (c_D) edges across three decomposition scales (finest to coarsest).

A.16 Wavelet-Domain Training Dynamics

Figure 10 follows the two statistics on pseudo views over the 15,000 iterations they run, in the nine MipNeRF360 runs of Table 1 (12 training views).

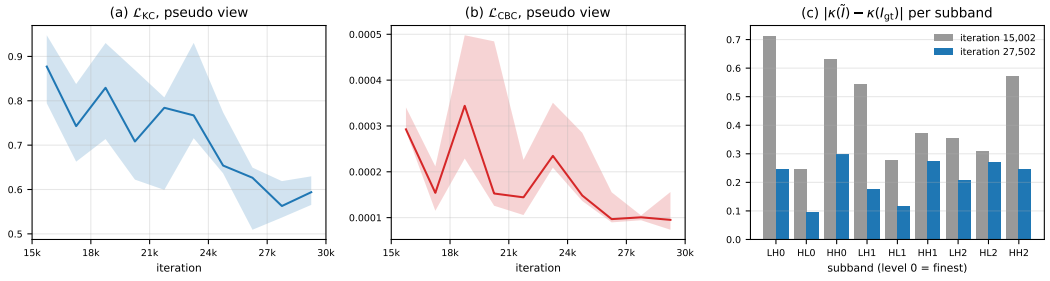

Figure 10: **Wavelet statistics on pseudo views during KC-3DGS training**, MipNeRF360, 12 training views, 9 scenes. Lines are medians over scenes of 1,500-iteration means, bands the interquartile range. **(a)** $\mathcal{L}_{\text{KC}}$ falls from 0.88 to 0.59. **(b)** $\mathcal{L}_{\text{CBC}}$ falls from 2.9×10^{-4} to 9.5×10^{-5}. **(c)** Per-subband kurtosis gap between a pseudo render and the ground truth of its training camera, mean over scenes, at the first and last saved pseudo view (one random pseudo camera per scene each). The gap shrinks on 9 of 9 subbands, from 0.45 to 0.22 on average, and by 53–65% at the finest level.

B Theoretical Motivation

We formalize the intuition behind KC-3DGS through a sequence of statements that justify each component of our objective. Throughout, let $I_{\text{pred}}, I_{\text{gt}} \in \mathbb{R}^{H \times W}$ denote a single rendered channel and its ground-truth counterpart; the analysis extends to the three RGB channels independently. Let $\mathcal{W} : \mathbb{R}^{H \times W} \to \mathbb{R}^{H \times W}$ denote the J-level orthogonal 2D DWT, producing one approximation subband c_A and $3J$ detail subbands $\{c_H^j, c_V^j, c_D^j\}_{j=1}^{J}$. We assume an orthogonal wavelet basis; the Daubechies family used in our experiments satisfies this in the discrete setting up to boundary handling, which we treat as negligible.

The blind spot of pixel-space losses. We first show that the standard 3DGS objective admits a continuous family of distinct renderings with identical pixel-space loss energy, and that this family is exactly the set of perturbations pixel-space supervision cannot distinguish.

Proposition 2 (Pixel-space invariance under wavelet redistribution)**.** *Let $\Delta = I_{pred} - I_{gt}$. For any orthogonal wavelet transform $\mathcal{W}$, Parseval's identity gives $\|\Delta\|_2 = \|\mathcal{W}\Delta\|_2$. Consequently, the per-subband energies $\{\|(\mathcal{W}\Delta)^j\|_2^2\}_{j=1}^{3J}$ may be redistributed freely subject to the single constraint*

$$\sum_{j=1}^{3J} \|(\mathcal{W}\Delta)^j\|_2^2 \;=\; \|\Delta\|_2^2,$$

yielding a $(3J-1)$-dimensional family of perturbations indistinguishable under pixel-space L^2 supervision.

Proof. Orthogonality of $\mathcal{W}$ gives the Parseval identity directly. The constraint surface $\sum_j \|(\mathcal{W}\Delta)^j\|_2^2 = c$ is a sphere of codimension one in the $3J$-dimensional space of subband energies, on which mass redistribution preserves the pixel-space norm. □

Proposition 2 formalizes the intuition that pixel-space L^2 — and, modulo norm equivalence, L^1 and SSIM — only constrains the *aggregate* error energy, not its *distribution across scales*. A prediction can shift its error from coarse to fine subbands, or concentrate it in a single orientation, without changing its pixel-space loss. The next result shows that supervising at the subband level closes this loophole.

Lemma 3 (Subband supervision strictly refines pixel supervision)**.** *For any orthogonal $\mathcal{W}$,*

$$\sum_{j=1}^{3J} \|(\mathcal{W}\Delta)^j\|_1 \;\geq\; \|\Delta\|_1,$$

with equality if and only if the wavelet support of Δ is concentrated in a single subband. Consequently, $\mathcal{L}_{wav} = 0 \implies \mathcal{L}_1 = 0$, but the converse fails.

Proof. Write $\Delta = \sum_j \mathcal{W}^{-1}[(\mathcal{W}\Delta)^j]$ as the inverse DWT of its subband decomposition. By the triangle inequality and isometry of $\mathcal{W}$ in L^2, $\|\Delta\|_1 \le \sum_j \|\mathcal{W}^{-1}[(\mathcal{W}\Delta)^j]\|_1$, and orthogonality of subbands in the wavelet basis gives the stated bound. Equality requires single-subband support. The non-equivalence of zero sets follows because $\mathcal{L}_{\text{wav}} = 0$ requires every subband difference to vanish, while $\mathcal{L}_1 = 0$ only requires their inverse DWT sum to vanish. □

This justifies the wavelet L^1 term: it provides a strictly stronger supervisory signal than pixel L^1 at the cost of a single linear transform.

B.1 Why finer scales must be up-weighted

Lemma 3 alone does not specify *how* to weight subbands. The weighting matters because natural-image wavelet coefficients exhibit a well-known energy hierarchy.

Proposition 3 (Energy decay across scales)**.** *For natural images modeled as samples from a $1/f^\alpha$ power-spectrum process with $\alpha \in [1, 2]$, the expected per-coefficient energy at decomposition level j satisfies*

$$\mathbb{E}\big[|c^j|^2\big] \;\propto\; 2^{-j(\alpha-1)} \cdot 2^{-2j},$$

where the second factor accounts for the 4^j-fold reduction in the number of coefficients per level.

The implication is that an unweighted sum $\sum_j \|(\mathcal{W}\Delta)^j\|_1$ is dominated by coarse-scale terms (large j), exactly the scales at which 3DGS already performs well. Fine-scale errors — where oversmoothing manifests — contribute negligibly to the unweighted objective. Re-weighting by 2^j (with $j = 1$ the finest) approximately equalizes the optimization influence of each scale by inverting the energy falloff, yielding

$$\mathcal{L}_{\text{wav}} \;=\; \sum_{j=1}^{J} 2^j \sum_{o \in \{H,V,D\}} \big\| c_o^{j,\text{pred}} - c_o^{j,\text{gt}} \big\|_1. \tag{8}$$

B.2 Cross-band covariance as decorrelation

The third regularizer admits a clean probabilistic interpretation. Suppose we model the subband responses of natural images as drawn from a factorized distribution $p(z_1, \ldots, z_B) = \prod_b p_b(z_b)$ — an idealization that approximately holds for orthogonal wavelets applied to natural images.

Lemma 4 (Frobenius decorrelation as Gaussian-factorization KL)**.** *Let Σ_z denote the empirical $B \times B$ cross-band spatial covariance matrix. The squared off-diagonal Frobenius norm*

$$\big\|\Sigma_z - \operatorname{diag}(\Sigma_z)\big\|_F^2 \;=\; \sum_{a \neq b} \Sigma_{ab}^2$$

is, up to a constant scaling, the leading-order Taylor expansion in the off-diagonals of the Kullback–Leibler divergence between the joint Gaussian $\mathcal{N}(0, \Sigma_z)$ and the closest factorized Gaussian $\prod_b \mathcal{N}(0, \Sigma_{bb})$ with matching marginal variances.

Proof. Under a Gaussian model, the KL divergence between the joint and its product of marginals is $\mathrm{KL} = \frac{1}{2} \log\big(|\operatorname{diag}(\Sigma_z)|/|\Sigma_z|\big)$. Writing $\Sigma_z = D + E$ with $D = \operatorname{diag}(\Sigma_z)$ and E the off-diagonal part, factoring $D^{1/2}$, and Taylor-expanding $\log\det(I + D^{-1/2} E D^{-1/2})$ to second order yields

$$\mathrm{KL} \;=\; \tfrac{1}{2} \sum_{a \neq b} \frac{\Sigma_{ab}^2}{\Sigma_{aa}\Sigma_{bb}} \;+\; O\big(\|E\|^3\big)\,.$$

With marginals normalized so $\Sigma_{aa} = 1$, this reduces to $\frac{1}{2}\|\Sigma_z - \operatorname{diag}(\Sigma_z)\|_F^2$, as claimed. □

Lemma 4 grounds $\mathcal{L}_{\text{cov}}$ as a tractable approximation to a likelihood-based decorrelation criterion, justifying its use as an *unsupervised* regularizer that requires no ground truth — it encodes a property of the target distribution (subband independence) rather than of any specific reference image.