

Article

Adaptive-Scale Color Offset and Error-Guided Gaussian Reallocation for Multi-Scale 3D Gaussian Splatting

Hyeonbin Park [†], Yoocho Lee [†], Dongho Lee  and Dongsan Jun ^{*}

Department of Computer Engineering, Dong-A University, Busan 49315, Republic of Korea; hbpark@donga-ispl.kr (H.P.); yhlee@donga-ispl.kr (Y.L.); dhlee@donga-ispl.kr (D.L.)

^{*} Correspondence: dsjun@dau.ac.kr

[†] These authors contributed equally to this work.

Abstract

Multi-scale rendering exposes scale-dependent reconstruction errors in 3D Gaussian splatting (3DGS) when the rendering resolution differs from the training resolution. Existing anti-aliasing and multi-scale optimization methods reduce these artifacts, but each Gaussian usually keeps a fixed color representation and a fixed spatial allocation after densification. This fixed representation is restrictive at coarse scales because one rendered pixel combines multiple Gaussian contributions into an average color that can differ from the color assigned to each Gaussian. In addition, Gaussians allocated by fine-scale reconstruction criteria may provide insufficient coverage for regions with persistent coarse-scale errors. This paper proposes adaptive-scale color offset and error-guided Gaussian reallocation for multi-scale 3DGS. The adaptive-scale color offset applies a scale-specific correction to the shared base color of each Gaussian while keeping its geometry unchanged. Error-guided Gaussian reallocation shifts low-contribution Gaussians toward high-error regions under a fixed primitive budget and preserves Gaussians that support coarse-scale rendering. Experimental results show that the proposed method improves multi-scale rendering quality across rendering scales on Mip-NeRF 360, Tanks and Temples, and Deep Blending, with larger gains at coarse scales. These results demonstrate that adapting Gaussian color and primitive allocation together improves scale-consistent 3DGS rendering without modifying the inference-stage rendering process.

Keywords: novel view synthesis; 3D Gaussian splatting; multi-scale rendering; anti-aliasing; scale-consistent rendering

MSC: 54H30; 65D18; 68U10



Academic Editor: David N. Olivieri

Received: 25 June 2026

Revised: 25 July 2026

Accepted: 26 July 2026

Published: 30 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Novel view synthesis (NVS) aims to synthesize images of a scene from viewpoints unseen during training. NVS supports practical applications such as virtual reality, digital twins, robotics, and 3D content creation. Neural radiance fields (NeRF) [1] represent a scene as a continuous volumetric function for novel view rendering. Although NeRF achieves high visual quality, dense ray sampling increases the rendering cost. 3D Gaussian splatting (3DGS) [2] reduces this cost by representing the scene with explicit anisotropic Gaussian primitives and rasterizing them directly. These primitives are optimized to reconstruct a single scene from training images at the training resolution.

In practical rendering scenarios, the same optimized 3DGS representation is often used at scales different from the training resolution. This scale mismatch becomes more

pronounced at coarse scales because each pixel covers a larger projected scene area. The effective sampling density therefore decreases, and aliasing artifacts become more severe. For this reason, a 3DGS model optimized at the training resolution can lose reconstruction accuracy at other rendering scales.

Existing multi-scale 3DGS methods address this scale sensitivity mainly through two strategies: scale-aware filtering [3,4] and multi-scale optimization [5,6]. Scale-aware filtering constrains Gaussian primitives according to the target sampling rate to suppress aliasing. The filtering changes the scale-dependent contribution of each Gaussian, while the Gaussian color remains shared across scales. A complementary strategy is multi-scale optimization, which trains one shared Gaussian set across multiple target scales. This optimization balances scale-dependent reconstruction losses, but the shared Gaussian set still defines per-Gaussian color and primitive allocation independently of coarse-scale errors. Figure 1 shows visible quality degradation in MS-GS under coarse-scale rendering. This example motivates a closer examination of Gaussian color and primitive allocation under coarse-scale rendering.

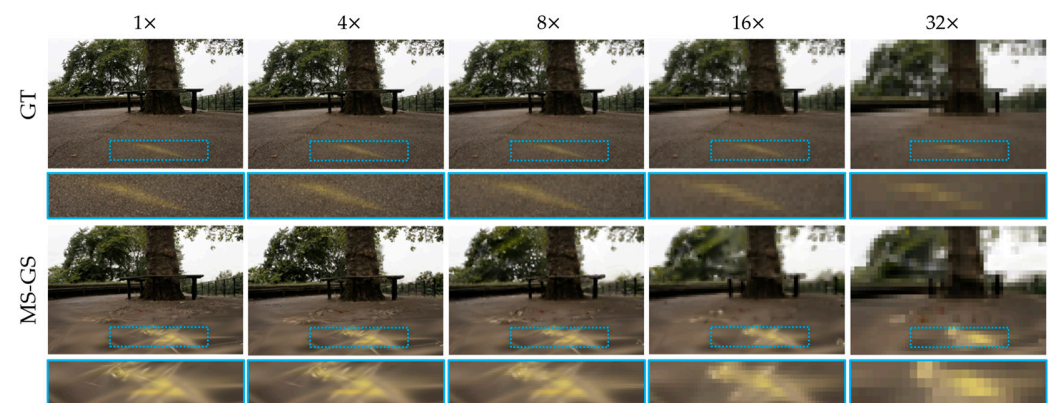


Figure 1. Coarse-scale failure cases of MS-GS on the treehill scene compared with the ground truth. The yellow streaks come from the color mismatch, and the structural blur comes from the Gaussian allocation imbalance. The blue boxes mark the magnified regions.

For Gaussian color, the key issue arises from the mismatch between fixed per-Gaussian color and the pixel average formed at each rendering scale. At a coarse scale, one pixel aggregates the projected contributions of multiple Gaussians into an average color. The average pixel color can differ from the color assigned to each contributing Gaussian. A single per-Gaussian color therefore only approximates these scale-dependent averages, which can produce coarse-scale color errors. For primitive allocation, the main issue arises during densification. Densification adds Gaussians in regions with large fine-scale reconstruction error, so fine-scale criteria largely determine primitive placement. After densification, the number of Gaussians is fixed, and later optimization primarily refines existing primitives. This allocation can concentrate redundant primitives in well-covered regions, whereas coarse-scale error regions can remain underrepresented.

To address the color mismatch and primitive allocation imbalance, we propose adaptive-scale color offset and error-guided Gaussian reallocation (ACO-EGR) for multi-scale 3DGS. ACO adds a scale-specific color offset to the shared base color of each Gaussian. At a target scale, the corresponding offset corrects the color used for rendering. This scheme provides scale-dependent color adaptation without changing Gaussian geometry. EGR redistributes low-contribution Gaussians toward high-error regions without increasing the primitive count. The contribution of each Gaussian is measured from its accumulated rendering weights across training views. Gaussians with large coarse-scale contributions are retained at their current positions. Low-contribution Gaussians outside this protected

set are then moved to underrepresented regions, which redirects primitive allocation toward persistent coarse-scale errors. The main contributions of this work are summarized as follows:

- We introduce ACO, which corrects scale-dependent color mismatch through a scale-specific offset for the spherical harmonics (SH) direct current (DC) coefficient while sharing the higher-order SH coefficients across rendering scales.
- We propose EGR, which redistributes a fixed Gaussian budget toward regions with persistent reconstruction errors while preserving Gaussians that support coarse-scale rendering.
- We integrate ACO and EGR into a unified framework that jointly addresses scale-dependent color mismatch and primitive-allocation imbalance. Experiments with MS-GS on three real-scene benchmarks demonstrate improved multi-scale rendering quality, particularly at coarse scales. Additional experiments on Mip-NeRF 360 demonstrate the applicability of ACO-EGR to three recent anti-aliasing 3DGS backbones.

The remainder of this paper is organized as follows: Section 2 reviews related work on scale-dependent rendering and multi-scale 3DGS. Section 3 describes how the proposed ACO-EGR addresses scale-dependent color mismatch and primitive allocation imbalance. Finally, experimental results and conclusions are presented in Sections 4 and 5, respectively.

2. Related Work

This section reviews prior studies relevant to scale-dependent rendering in NVS. The review first covers NeRF methods based on volumetric pre-filtering and scale-aware sampling. It then summarizes 3DGS methods that modify Gaussian filtering, primitive selection, or training across scales. Finally, it discusses Gaussian color and primitive allocation to clarify the motivation for the proposed method.

2.1. NeRF-Based Methods for Scale-Dependent Rendering

NeRF [1] represents a scene as a continuous radiance field parameterized by position and viewing direction. Novel views are synthesized by integrating sampled color and density values along camera rays. Although this rendering process achieves high view-synthesis quality, dense per-ray sampling increases computational cost. To reduce this cost, NeRF-based methods [7–10] use hash grids or explicit feature structures to accelerate rendering. These methods lower the sampling cost while preserving the ray-based sampling structure. When the output scale changes, point-wise ray samples can inadequately cover the scene area represented by each pixel and cause aliasing artifacts.

Anti-aliased NeRF methods address this issue by incorporating the pixel footprint into volumetric rendering. Mip-NeRF [11] approximates each pixel with a cone and encodes the corresponding conical frustum using integrated positional encoding. Mip-NeRF 360 [12] extends this design to unbounded scenes through a nonlinear parameterization that maps distant scene regions into a bounded domain. Zip-NeRF [13] applies anti-aliased rendering to grid-based fields for more efficient training. These methods indicate that scale consistency in NeRF depends on matching ray sampling to the pixel footprint.

2.2. 3DGS-Based Methods for Anti-Aliased Rendering

3DGS [2] renders anisotropic Gaussian primitives through rasterization for real-time novel view synthesis. A tile-based rasterizer projects these primitives onto the image plane and composites their color and opacity contributions through alpha blending. Because Gaussian parameters are optimized for the sampling pattern at the training resolution, rendering at a different output scale can introduce aliasing or blur.

Filtering-based methods address scale-dependent artifacts by controlling the projected footprint of each Gaussian. Mip-Splatting [3] limits high-frequency components near the training sampling rate with a 3D smoothing filter and a 2D approximation of the box filter. This band-limiting suppresses aliasing when the rendering sampling rate differs from the training setting. Analytic-Splatting [14] integrates the Gaussian response over each pixel footprint to compute an analytic anti-aliased value. Scale-adaptive Gaussian splatting [15] uses a 2D scale-adaptive filter at test time to adjust the effective Gaussian scale to the target rendering scale while keeping the trained parameters fixed. Anti-Aliased 2D Gaussian Splatting (AA-2DGS) [16] mitigates 2D Gaussian splatting (2DGS) aliasing via a world-space flat smoothing kernel while preserving geometric accuracy.

Beyond test-time filtering, multi-scale 3DGS methods construct a shared Gaussian set across multiple rendering scales. Multi-Scale 3DGS (MS-GS) [5] optimizes a single model with reconstruction losses evaluated at multiple output scales. MS-GS organizes fine-detail primitives into hierarchical Gaussian subsets and selects a scale-matched subset for each pixel to suppress aliasing. These methods share per-Gaussian color and primitive placement across scales. Coarse-scale color mismatch and allocation imbalance can therefore still occur.

2.3. Color Modeling and Primitive Allocation in 3DGS

Color modeling and primitive allocation have been studied in 3DGS from different perspectives. Spherical harmonics represent the view-dependent color of each Gaussian, and appearance embeddings compensate for image- or camera-specific appearance variations. Such color models describe angular or image-level color variation rather than scale-dependent color changes. NeRF-W [17] associates an appearance embedding with each training image and estimates an embedding for a novel view. VastGaussian [18] uses an appearance module during training and removes it at inference. ACO associates an explicit red-green-blue (RGB) offset with each rendering scale and applies the offset only to the SH DC coefficient of each Gaussian. The higher-order SH coefficients remain shared across scales. Since the target rendering scale is specified at inference, the corresponding offset can be selected directly.

LightGaussian [19] addresses model compression. It prunes low-significance Gaussians, distills the SH representation to a lower degree, and quantizes Gaussian attributes. Its objective differs from the scale-dependent color correction performed by ACO.

AbsGS [20] adds Gaussians guided by gradient-based criteria, whereas 3DGS as Markov Chain Monte Carlo (3DGS-MCMC) [21] relocates Gaussians while maintaining a fixed Gaussian count. Pixel-GS [22] refines the densification criterion with pixel-aware gradients, and Mini-Splatting [23] reorganizes the Gaussian density under a constrained budget. These allocation strategies are designed for single-scale training. Such reallocation can move Gaussians away from regions where they are useful for coarse-scale rendering. These single-scale allocation criteria indicate that shared 3DGS representations require primitive placement criteria that reflect coarse-scale rendering errors.

3. Proposed Method

This section describes the proposed ACO-EGR for multi-scale 3DGS. Tables 1–3 summarize the notation used in the framework overview, ACO, and EGR, respectively. ACO-EGR augments a multi-scale 3DGS backbone with two training-stage modules. ACO adapts per-Gaussian color across rendering scales through scale-specific DC color offsets. EGR refines the post-densification Gaussian allocation by moving low-contribution Gaussians to high-error regions within a fixed Gaussian set. Because both modules are optimized during training, the inference-stage rendering process of the backbone is preserved.

Table 1. Notation used in the overview of ACO-EGR.

Symbol	Description
i	Gaussian index
C	Rendered color of a pixel
c_i	Color of Gaussian i
α_i	Opacity of Gaussian i
w_i	Blending weight of Gaussian i

Table 2. Notation used in the Adaptive-Scale Color Offset formulation.

Symbol	Description
d	Viewing direction
Y_b	b -th SH basis function
$f_{i,b}$	SH coefficient of Gaussian i
B	Number of SH basis functions
$f_{i,0}$	DC coefficient of Gaussian i
S	Number of rendering scales
s	Rendering-scale index
Δ_i^s	DC color offset of Gaussian i at scale s

Note: SH and DC denote spherical harmonics and direct current, respectively.

Table 3. Notation used in the Error-Guided Gaussian Reallocation formulation.

Symbol	Description
N	Number of Gaussians after densification
v	View index
V	Number of training views
$w_i^{(v)}$	Blending weight of Gaussian i in view v
$\tilde{w}_i^{(v)}$	Normalized blending weight of Gaussian i in view v
C_i^{mean}	Mean blending contribution of Gaussian i over views
C_i^{max}	Maximum blending contribution of Gaussian i over views
C_i^{HE}	Error-weighted contribution of Gaussian i over views
G_i^{mean}	Mean normalized parameter-gradient magnitude of Gaussian i
s_i^{LC}, s_i^{HE}	Selection scores for an LC source and an HE target
p_i^{LC}, p_i^{HE}	Sampling probabilities for an LC source and an HE target
$M_{eff}^{LC}, M_{eff}^{HE}$	Effective sample sizes of the LC and HE distributions
M	Number of Gaussian pairs selected for relocation

Note: LC and HE denote low contribution and high error, respectively.

3.1. Overview of the ACO-EGR Framework

Figure 2 illustrates the overall flow of the proposed ACO-EGR. A multi-scale 3DGS backbone represents a scene with anisotropic 3D Gaussian primitives. Table 1 summarizes the notation used in the overview of ACO-EGR.

Each Gaussian contains a position, a covariance, an opacity, and a color represented by SH. Rendering follows front-to-back alpha compositing over the projected Gaussians. In this compositing process, each Gaussian contributes through its color and opacity. Front Gaussians attenuate the contribution of the Gaussians behind them. The rendered pixel value C is computed by Equation (1):

$$C = \sum_i c_i w_i, \quad \text{where } w_i = \alpha_i \prod_{j < i} (1 - \alpha_j), \quad (1)$$

where c_i , α_i , and w_i denote the color, opacity, and blending weight of the i -th Gaussian, respectively. Equation (1) separates the two quantities used by ACO-EGR. ACO modifies

the color term c_i through scale-specific DC color offsets, and EGR uses the blending weight w_i to estimate each Gaussian's rendering contribution. The two modules are added after backbone densification, once the Gaussian set has been fixed for subsequent training. During densification, split and clone operations expand the Gaussian set and change the coverage of individual primitives. After this stage, the fixed Gaussian set provides stable primitives for learning color offsets and performing reallocation.

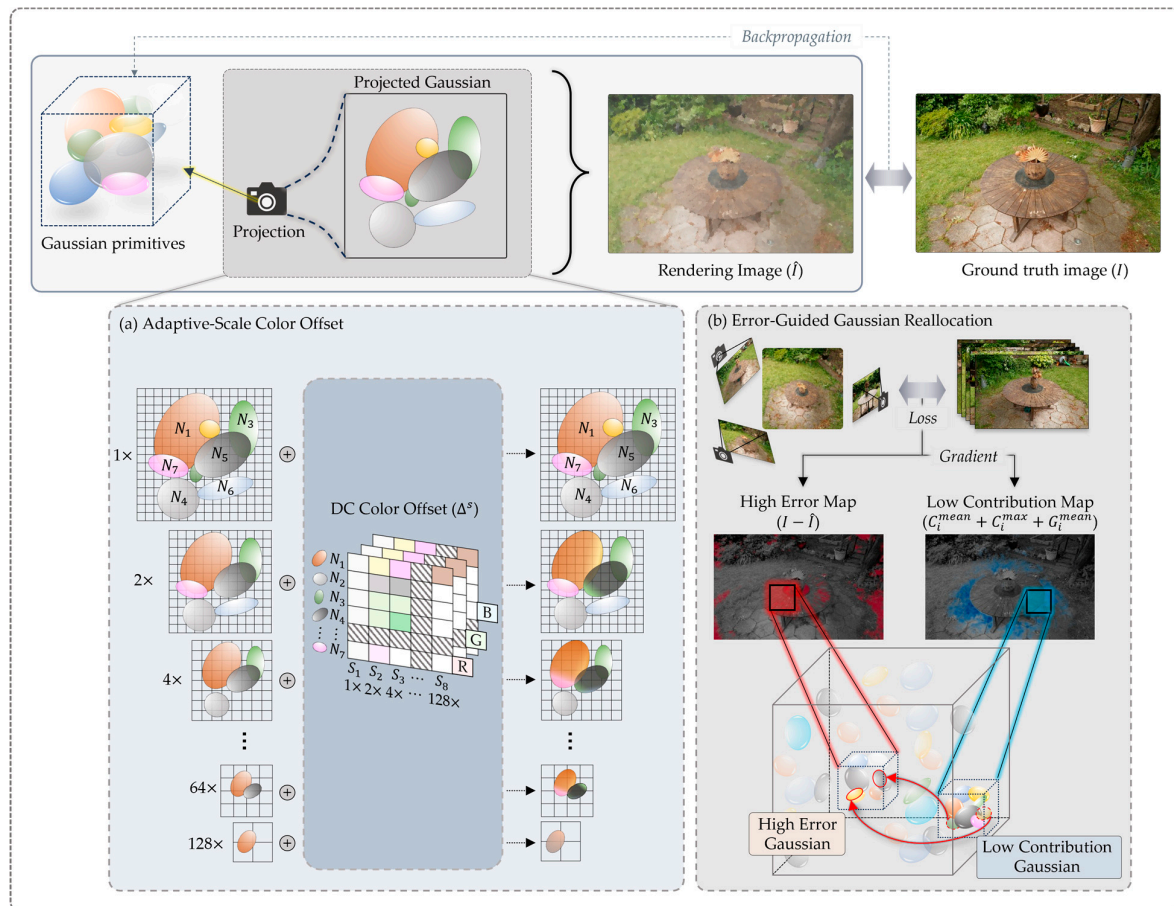


Figure 2. Overview of the proposed ACO-EGR for multi-scale 3DGS.

3.2. Adaptive-Scale Color Offset (ACO)

At coarse rendering scales, one pixel aggregates the projected contributions of more Gaussians into a scale-dependent average color. A fixed per-Gaussian color can deviate from this average because the same color representation is shared across rendering scales. ACO corrects this color mismatch within the SH color representation of 3DGS. Table 2 summarizes the notation used in the ACO formulation in Equations (2) and (3). For the i -th Gaussian, the color along the view direction d is represented by the SH expansion in Equation (2).

$$c_i(d) = \sum_{b=0}^{B-1} Y_b(d) f_{i,b}, \quad (2)$$

Here, $c_i(d)$, $Y_b(d)$, $f_{i,b}$, and B denote the color of Gaussian i along view direction d , the b -th SH basis function, the corresponding SH coefficient, and the number of SH basis functions, respectively. In Equation (2), $f_{i,0}$, corresponding to $b = 0$, denotes the DC coefficient. The DC coefficient defines the view-independent base color, and the higher-order coefficients represent view-dependent color variation.

The scale-dependent color mismatch is associated with the shared base color defined by the DC coefficient $f_{i,0}$ in Equation (2). ACO adds a scale-specific offset to this DC coefficient and shares the higher-order SH coefficients across scales. For each Gaussian i , ACO uses a learnable DC offset matrix $\Delta_i \in \mathbb{R}^{S \times 3}$. Each row $\Delta_i^s \in \mathbb{R}^3$ stores one RGB offset for scale s . The matrix is initialized as $\Delta_i = 0$, so the initial ACO color equals the backbone color. The scale-adapted color is defined in Equation (3).

$$c_i^s(d) = Y_0 f_{i,0}^s + \sum_{b=1}^{B-1} Y_b(d) f_{i,b}, \quad \text{where } f_{i,0}^s = f_{i,0} + \Delta_i^s, \quad (3)$$

Here, $\Delta_i^s \in \mathbb{R}^3$ denotes the DC color offset of Gaussian i at scale s , and $f_{i,0}^s$ denotes the adapted DC coefficient. Equation (3) changes the DC coefficient alone and keeps the view-dependent SH components $f_{i,b}$ for $b > 0$ shared across scales.

The DC offsets are optimized through the backbone reconstruction loss alone. At each rendering scale, the reconstruction loss between the rendered output and the ground truth back-propagates to the corresponding offset Δ_i^s , so the offset learns a scale-specific color correction. ACO modifies only the DC color term, so the module leaves the Gaussian geometry unchanged.

3.3. Error-Guided Gaussian Reallocation (EGR)

Densification allocates Gaussians mainly according to reconstruction error at the training scale, so regions with persistent coarse-scale errors can receive insufficient primitives. A selection criterion based only on training-scale contribution may relocate Gaussians with important coarse-scale support and degrade coarse-scale reconstruction quality. EGR reallocates low-contribution (LC) Gaussians toward high-error (HE) regions using contribution- and error-aware selection under a fixed Gaussian budget. It computes contribution, error, and gradient-based measures for each Gaussian, converts them into LC and HE scores, and samples pairs for relocation from the resulting probability distributions. Table 3 summarizes the notation used in the EGR formulation in Equations (4)–(12).

For each view, EGR normalizes the blending weight of each Gaussian by the total blending weight of all projected Gaussians. This produces a view-wise relative contribution comparable across views, as defined in Equation (4).

$$\tilde{w}_i(v) = \frac{w_i(v)}{\sum_{i'} w_{i'}(v) + \epsilon}, \quad (4)$$

Here, $w_i(v)$, i' , and ϵ denote the blending weight of Gaussian i in view v , the index over projected Gaussians in view v , and a small positive constant, respectively. The mean contribution C_i^{mean} measures the general contribution level over views, and the maximum contribution C_i^{max} measures strong contribution in a small subset of views. These two statistics are defined in Equation (5).

$$C_i^{mean} = \frac{1}{|V|} \sum_v \tilde{w}_i(v), \quad C_i^{max} = \max_v \tilde{w}_i(v), \quad (5)$$

where $|V|$ denotes the number of views used for contribution measurement. To build HE evidence, EGR combines reconstruction error with normalized Gaussian contribution. For each view, the absolute reconstruction differences are summed over all pixel positions and RGB channels. EGR multiplies this summed image error by the normalized contribution of each Gaussian and accumulates the resulting values over the training views. A Gaussian

receives a large error-weighted contribution when it is associated with large reconstruction error, as defined in Equation (6):

$$C_i^{HE} = \sum_v \tilde{w}_i(v) \sum_{h,w,\ell} |\hat{I}_{h,w,\ell}(v) - I_{h,w,\ell}(v)|, \quad (6)$$

Here, $\hat{I}_{h,w,\ell}(v)$ and $I_{h,w,\ell}(v)$ denote the rendered and the ground-truth (GT) images, respectively, at pixel position (h, w) and RGB channel ℓ in view v . The indices h and w denote the vertical and horizontal pixel positions, respectively, and $\ell \in \{1, 2, 3\}$ denotes the RGB channel index. The inner summation computes the summed absolute reconstruction error within each view, while the outer summation accumulates the normalized-contribution-weighted errors over the training views.

EGR forms LC and HE selection scores from the contribution and error measures defined above. Some Gaussians with small current contribution can receive large parameter updates during training and may later become useful. For this reason, EGR adds parameter-gradient magnitude in the LC scoring process. The gradient magnitude quantifies the update assigned to each Gaussian, and a small value corresponds to a weak update. EGR normalizes the gradient magnitude within each view and averages it over views, as expressed in Equation (7):

$$G_i^{mean} = \frac{1}{V} \sum_v \frac{\|\nabla_{\theta_i} L_2(v)\|}{\sum_{i'} \|\nabla_{\theta_{i'}} L_2(v)\| + \epsilon}, \quad \text{where } L_2(v) = \frac{1}{3H_v W_v} \sum_{h,w,\ell} (\hat{I}_{h,w,\ell}(v) - I_{h,w,\ell}(v))^2. \quad (7)$$

Here, $\nabla_{\theta_i} L_2(v)$ denotes the parameter gradient of Gaussian i computed from the mean squared reconstruction loss for view v . H_v and W_v denote the height and width of the image for view v , respectively. The summation in $L_2(v)$ accumulates the squared reconstruction differences over all pixel positions and RGB channels, while the factor $\frac{1}{(3H_v W_v)}$ converts the accumulated error into a mean value. The three measures have different numerical ranges. EGR applies robust z-score normalization before score computation. The robust z-score uses the median and median absolute deviation to reduce sensitivity to extreme values. This normalization stabilizes score computation when a small number of Gaussians have very large contribution values, as formulated in Equation (8).

$$z(x) = \frac{x - \text{median}(x)}{\text{median}(|x - \text{median}(x)|) + \epsilon}, \quad (8)$$

where $\text{median}(x)$ represents the median of measure x over all Gaussians. Because LC candidates have small mean contribution, small maximum contribution, and small gradient magnitude, EGR assigns a high score to Gaussians with low normalized values across these measures, as defined in Equation (9):

$$s_i^{LC} = \exp(-z(C_i^{mean}) - z(C_i^{max}) - z(G_i^{mean})). \quad (9)$$

EGR assigns a high HE score to a Gaussian with a large error-weighted contribution. To reduce the effect of scene-dependent error scales and extreme values, EGR uses a percentile score computed from the rank of C_i^{HE} among all Gaussians. A Gaussian with a larger C_i^{HE} receives a score closer to one, as defined in Equation (10):

$$s_i^{HE} = \frac{1}{N-1} \sum_{j=1}^N 1[C_j^{HE} < C_i^{HE}], \quad (10)$$

where $1[\cdot]$ is the indicator function and N is the number of Gaussians. The gate additionally evaluates the contribution of each Gaussian at scales of $4\times$ and coarser using the training

cameras. A Gaussian is excluded from the relocation candidates when its coarse-scale contribution exceeds its fine-scale contribution. This gate preserves Gaussians that provide stronger support for coarse-scale reconstruction. Using the LC and HE scores, EGR converts each LC or HE score into a sampling probability. For $q \in \{LC, HE\}$, EGR normalizes each score by the sum of scores in the corresponding candidate set, as expressed in Equation (11):

$$p_i^q = \frac{s_i^q}{\sum_k s_k^q}, \text{ where } q \in \{LC, HE\}, \quad (11)$$

Here, the denominator includes all candidate Gaussians for distribution q , and s_k^q is the score of candidate Gaussian k in that distribution. The resulting probability distributions determine both candidate sampling and the relocation count. EGR estimates this count with effective sample size. A concentrated distribution gives a smaller effective sample size, and a more uniform distribution gives a larger one. Because LC and HE candidates are paired for relocation, EGR sets M to the smaller effective sample size rounded down to an integer, as defined in Equation (12):

$$M = \left\lfloor \min(M_{eff}^{LC}, M_{eff}^{HE}) \right\rfloor, \text{ where } M_{eff} = \frac{1}{\sum_{j=1}^N p_j^2}, \quad (12)$$

where M_{eff}^{LC} and M_{eff}^{HE} refer to the effective sample sizes of the LC and the HE probability distributions, respectively. The value N follows the definition in Equation (10), and M is the resulting number of relocated Gaussian pairs.

For each of the M selected pairs, EGR relocates the LC Gaussian to the location of its paired HE Gaussian and copies the HE Gaussian's scale, orientation, and base color. The relocated Gaussian resets its higher-order SH coefficients and ACO offsets to zero and clears its optimizer state before subsequent training at the new location. After relocation, the paired Gaussians share the same location. EGR rescales their opacities to preserve the original transmittance according to the opacity split of 3DGS-MCMC [21].

The copied attributes serve only as an initialization at the high-error location. The paired Gaussians retain separate parameters and are independently optimized during subsequent training, allowing their appearance parameters to become distinct.

EGR operates after densification, once the Gaussian count has been fixed. Reallocation is performed at fixed intervals within a bounded window, followed by a standard optimization phase without further changes to the Gaussian allocation. Before every round, the LC and HE statistics are recomputed from the current model using all scale-1 training cameras. Each relocation therefore reflects the reconstruction state after the preceding optimization period.

EGR uses the adaptive relocation count, parameter reset, opacity rescaling, and continued gradient-based optimization as its stabilization strategy in place of an explicit rollback or separate re-correction module. The adaptive relocation count limits the perturbation introduced in each round, while the parameter reset and opacity rescaling provide a stable starting state for the relocated Gaussians. Gradient-based optimization resumes immediately after every round and updates the relocated Gaussians before the statistics for the next round are computed.

4. Experimental Results

This section evaluates the proposed ACO-EGR on three real-scene benchmarks with MS-GS [5] as the multi-scale 3DGS backbone. The evaluation first describes the experimental setup and then presents the multi-scale rendering performance of ACO-EGR compared with the retrained 3DGS and MS-GS models. The ablation studies quantify

module-level contributions while validating the ACO color-offset configuration and the EGR preservation strategy.

4.1. Experimental Setup

All experiments were conducted on a single NVIDIA RTX A6000 graphics processing unit (GPU). The proposed ACO-EGR was implemented on MS-GS [5], which served as the multi-scale 3DGS backbone. The single-scale 3DGS baseline and the MS-GS backbone were retrained from their public implementations under the official MS-GS training and testing protocol. The evaluation used three novel view synthesis benchmarks: Mip-NeRF 360 [12], Tanks and Temples [24], and Deep Blending [25]. For fair comparison, MS-GS and ACO-EGR were trained with identical experimental settings and densification thresholds. MS-GS and ACO-EGR were both trained for 40,000 iterations, with densification performed between iterations 500 and 15,000. EGR performs ten reallocation rounds at intervals of 1000 iterations from iteration 16,000 to iteration 25,000. Reallocation is then disabled, and standard optimization continues until iteration 40,000. Mip-Splatting, Analytic-Splatting, and AA-2DGS were trained for 30,000 iterations following their original training settings. Within each backbone comparison, the baseline and its ACO-EGR variant use the same iteration budget.

All methods use the same input resolutions, preprocessing, and training and evaluation camera splits. No additional data augmentation is used. Within each backbone comparison, the baseline and its ACO-EGR variant use the same multi-scale sampling schedule and loss configuration. Under this setting, performance differences between the two models can be attributed to the proposed modules rather than to differences in training conditions.

Rendering quality was evaluated using peak signal-to-noise ratio (PSNR) [26], structural similarity index measure (SSIM) [27], and learned perceptual image patch similarity (LPIPS) [28]. To assess computational overhead, inference-stage cost was measured by per-frame runtime and model memory usage.

4.2. Multi-Scale Rendering Performance

Multi-scale rendering performance is evaluated through scale-wise quality comparisons with retrained 3DGS and MS-GS, followed by inference-stage cost analysis. Table 4 summarizes the per-scale rendering quality under the MS-GS setting on three benchmark datasets. Each scale denotes the downsampling factor relative to the full-resolution image. Under the MS-GS evaluation protocol [5], Mip-NeRF 360 is evaluated from $1\times$ to $128\times$, whereas Tanks and Temples and Deep Blending are evaluated from $1\times$ to $64\times$. For each dataset, the retrained 3DGS baseline, the retrained MS-GS backbone, and ACO-EGR implemented on the MS-GS backbone (Ours) are compared using PSNR, SSIM, and LPIPS.

Before evaluating the proposed modules, Table 4 establishes why a multi-scale backbone is required. As the rendering scale becomes coarser, the retrained 3DGS baseline loses much of its full-resolution accuracy across all three datasets.

At the coarsest available scales, it obtains 14.48 dB at $128\times$ on Mip-NeRF 360, 13.77 dB at $64\times$ on Tanks and Temples, and 18.10 dB at $64\times$ on Deep Blending, whereas retrained MS-GS achieves 21.18, 20.22, and 26.10 dB at the same scales, respectively. This consistent coarse-scale degradation indicates that full-resolution training alone does not provide scale-consistent rendering. MS-GS therefore provides a more stable multi-scale backbone for evaluating the additional effects of the proposed ACO-EGR.

Table 4. Per-scale rendering quality under the MS-GS setting on three benchmark datasets.

Dataset	Method	PSNR	1× SSIM	LPIPS	PSNR	2× SSIM	LPIPS	PSNR	4× SSIM	LPIPS	PSNR	8× SSIM	LPIPS
Mip-NeRF 360	3DGS	27.52	0.8164	0.1352	25.23	0.8186	0.1166	21.35	0.7029	0.1419	18.83	0.5784	0.1594
	MS-GS	27.25	0.8069	0.1541	25.96	0.8012	0.1301	24.39	0.7605	0.1275	24.32	0.7817	0.1035
	Ours	27.70	0.8226	0.1264	26.16	0.8154	0.1208	26.35	0.8148	0.0965	25.58	0.8128	0.0856
Tanks and Temples	3DGS	23.84	0.8537	0.0913	22.48	0.8555	0.0758	19.46	0.7537	0.1048	17.09	0.6254	0.1179
	MS-GS	23.52	0.8446	0.1016	21.60	0.8269	0.0990	20.22	0.7716	0.1126	19.65	0.7497	0.1041
	Ours	23.85	0.8509	0.0895	22.02	0.8305	0.0927	21.38	0.8127	0.0937	21.00	0.7996	0.0827
Deep Blending	3DGS	29.75	0.9083	0.0907	29.51	0.9147	0.0608	27.56	0.8893	0.0611	24.77	0.8212	0.0722
	MS-GS	29.38	0.9028	0.0942	29.28	0.9083	0.0647	28.17	0.8917	0.0660	27.10	0.8716	0.0695
	Ours	29.49	0.9002	0.0914	29.26	0.9064	0.0644	28.63	0.8985	0.0574	27.90	0.8873	0.0557
Dataset	Method	PSNR	16× SSIM	LPIPS	PSNR	32× SSIM	LPIPS	PSNR	64× SSIM	LPIPS	PSNR	128× SSIM	LPIPS
Mip-NeRF 360	3DGS	17.12	0.4960	0.1430	15.86	0.4679	0.0698	14.97	0.4974	–	14.48	0.6180	–
	MS-GS	25.35	0.8433	0.0513	25.08	0.8787	0.0154	25.12	0.9120	–	21.18	0.8893	–
	Ours	26.35	0.8570	0.0478	25.92	0.8869	0.0200	26.05	0.9199	–	23.27	0.9068	–
Tanks and Temples	3DGS	15.42	0.5358	0.0685	14.33	0.5404	–	13.77	0.6776	–	–	–	–
	MS-GS	19.48	0.7689	0.0458	20.10	0.8410	–	20.22	0.9079	–	–	–	–
	Ours	20.55	0.8101	0.0370	20.95	0.8619	–	20.93	0.9191	–	–	–	–
Deep Blending	3DGS	22.25	0.7421	0.0659	20.04	0.6860	–	18.10	0.6801	–	–	–	–
	MS-GS	26.70	0.8821	0.0425	27.08	0.9203	–	26.10	0.9392	–	–	–	–
	Ours	27.16	0.8927	0.0381	27.35	0.9227	–	26.67	0.9432	–	–	–	–

Note: Entries marked with “–” indicate unavailable scales or metrics in the reference evaluation [5]. The best result at each scale is shown in bold.

Compared with the retrained MS-GS backbone, the proposed method provides larger gains mainly at coarser rendering scales. At 1× and 2×, PSNR changes remain small across the three datasets, with only a 0.02 dB decrease at 2× on Deep Blending. From 4× onward, ACO-EGR improves PSNR over MS-GS at all available scales. On Mip-NeRF 360, the PSNR gain reaches 1.96 dB at 4× and 2.09 dB at 128×. On Tanks and Temples and Deep Blending, the gains at the coarsest available scale reach 0.71 and 0.57 dB at 64×, respectively. These PSNR gains are accompanied by higher SSIM from 4× onward on all three datasets. The combined PSNR and SSIM improvements indicate stronger coarse-scale structural consistency rather than only higher pixel-wise fidelity.

LPIPS complements the PSNR and SSIM results by reflecting perceptual differences that are less directly represented by pixel-wise metrics. Across the evaluated LPIPS scales, the proposed method generally lowers perceptual error on all three datasets, especially at scales where local structures become difficult to distinguish. These LPIPS results indicate that ACO-EGR improves perceptual fidelity beyond the PSNR and SSIM gains, particularly in texture-rich regions and thin structures.

We assess the consistency of the PSNR gains using paired scene-level statistics on the nine Mip-NeRF 360 scenes. For each scene and rendering scale, the PSNR gain is calculated as the ACO-EGR result minus the corresponding MS-GS result. This paired difference removes the scene-dependent PSNR level shared by the two methods and directly measures the gain obtained by ACO-EGR.

Figure 3 presents the scene-level PSNR gains, together with their mean and 95% confidence interval at each rendering scale. The sampling unit is a Mip-NeRF 360 scene, and the PSNR for each scene is averaged over its held-out test views under the official MS-GS protocol. At each scale, the difference between the ACO-EGR and MS-GS PSNR values yields nine paired scene-level observations. The error bars represent the two-sided 95% confidence interval of the mean gain based on the t -distribution, calculated as $\Delta \pm t_{0.975,8} \cdot s / \sqrt{9}$, where s is the sample standard deviation and $t_{0.975,8} = 2.306$. No bootstrap procedure was used for Figure 3.

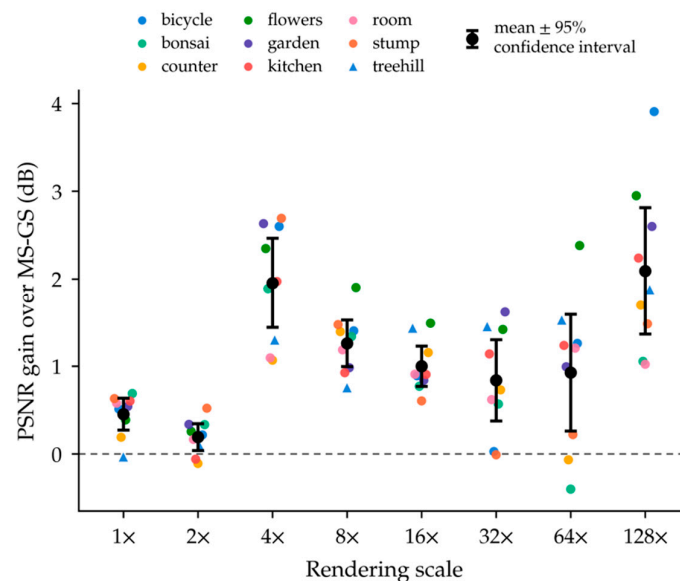


Figure 3. Per-scene PSNR gains of ACO-EGR over MS-GS, with error bars showing 95% confidence intervals.

To assess statistical significance, the exact two-sided Wilcoxon signed-rank test was applied to the same nine paired differences at each scale. The resulting p -values were adjusted across the eight rendering scales using the Benjamini–Hochberg procedure. The gain remained significant at every scale after adjustment, with no adjusted p -value exceeding 0.028.

Figure 4 illustrates this behavior on two Mip-NeRF 360 scenes. At coarse scales, each pixel aggregates multiple Gaussian contributions. This averaging can obscure local errors in flowers, grass boundaries, and bicycle spokes. In the highlighted regions, the proposed method shows sharper local patterns than MS-GS without changing the inference-stage rendering process.

Table 5 presents the per-scale rendering time and model memory size for the three benchmarks. The rendering time stays near the MS-GS baseline, especially in the coarse-scale ranges where Table 4 shows the largest quality gains. ACO requires additional model memory size to store scale-specific color offsets for Gaussian primitives. Overall, ACO-EGR converts the additional scale-specific color representation into coarse-scale quality gains with a marginal increase in inference-time cost.

Table 5. Per-scale rendering time and model memory size on three benchmark datasets.

Dataset	Method	Time (ms)								Memory Size (MB)
		1×	2×	4×	8×	16×	32×	64×	128×	
Mip-NeRF 360	3DGS	17.13	13.21	21.37	18.12	34.55	65.82	113.34	113.50	646.6
	MS-GS	10.32	7.47	6.53	6.48	5.81	5.69	5.70	5.80	920.9
	Ours	12.69	8.68	7.89	7.12	6.69	6.38	6.14	6.28	1044.4
Tanks and Temples	3DGS	16.05	12.61	24.82	22.27	42.84	71.34	83.07	42.86	372.2
	MS-GS	6.68	6.22	5.63	5.74	6.23	5.99	6.21	4.99	537.1
	Ours	8.83	7.40	6.47	5.98	5.72	5.94	5.67	4.86	619.9
Deep Blending	3DGS	18.05	7.70	11.73	12.20	21.02	35.91	61.07	66.79	587.1
	MS-GS	8.36	6.21	6.70	4.97	4.73	4.51	4.43	4.30	766.6
	Ours	9.40	6.66	5.87	5.34	5.14	5.09	4.96	4.82	889.0



Figure 4. Qualitative comparison of 3DGS, MS-GS, and Ours against ground truth (GT) on two Mip-NeRF 360 scenes from 1× to 32×. The blue boxes mark the magnified regions.

4.3. Ablation Studies

The ablation studies examine whether the gains in Table 4 arise from module complementarity or from the isolated effects of a single design. The analysis first separates the contributions of ACO and EGR, then validates the module-specific designs. In addition, the training behavior is examined to determine whether these design effects are reflected in the training-loss behavior after densification.

Table 6 evaluates how ACO and EGR contribute individually and jointly under the MS-GS setting. The results show a clear division of roles between the two modules. When ACO is removed, PSNR drops markedly from 4× onward, and the corresponding SSIM values also decrease in the same scale range. Compared with the configuration without

ACO, the full ACO-EGR model improves PSNR by 1.62 dB at $4\times$, 1.24 dB at $8\times$, 0.77 dB at $16\times$, and 0.73 dB at $32\times$. When EGR is removed, the PSNR reduction is smaller but consistent from $1\times$ to $16\times$, with values ranging from 0.19 to 0.22 dB.

Table 6. Module-wise ablation of ACO-EGR under the MS-GS setting on Mip-NeRF 360.

Config	PSNR	1× SSIM	LPIPS	PSNR	2× SSIM	LPIPS	PSNR	4× SSIM	LPIPS	PSNR	8× SSIM	LPIPS
Ours	27.70	0.8226	0.1264	26.16	0.8154	0.1208	26.35	0.8148	0.0965	25.58	0.8128	0.0856
w/o ACO	27.62	0.8221	0.1264	26.47	0.8224	0.1130	24.73	0.7716	0.1179	24.34	0.7731	0.1043
w/o EGR	27.48	0.8104	0.1476	25.97	0.8069	0.1356	26.16	0.8069	0.1094	25.39	0.8066	0.0954

Config	PSNR	16× SSIM	LPIPS	PSNR	32× SSIM	LPIPS	PSNR	64× SSIM	LPIPS	PSNR	128× SSIM	LPIPS
Ours	26.35	0.8570	0.0478	25.92	0.8869	0.0200	26.05	0.9199	–	23.27	0.9068	–
w/o ACO	25.58	0.8349	0.0565	25.19	0.8690	0.0240	25.35	0.9072	–	22.41	0.8910	–
w/o EGR	26.13	0.8531	0.0536	25.86	0.8861	0.0199	26.21	0.9223	–	23.27	0.9092	–

Note: The best result at each scale is shown in bold.

At the coarsest scale range, the configuration without EGR slightly exceeds the full model at $64\times$ and matches it in PSNR at $128\times$. This result indicates that EGR is most effective when local structures still contribute to reconstruction error.

At extremely coarse scales, spatial details are strongly compressed, so ACO alone can provide sufficient scale-specific color correction. LPIPS serves as a complementary perceptual check, not as the primary basis of interpretation. The full model obtains the lowest LPIPS from $4\times$ to $16\times$, where both modules contribute. ACO mainly corrects scale-dependent color errors at coarse scales. EGR refines Gaussian allocation in ranges where local structure still affects reconstruction quality. This separation confirms that the two modules target different error sources without duplicating the same correction.

Table 7 further examines the SH orders by applying the scale-specific ACO offset to DC only and then cumulatively extending it to all coefficients up to orders 1, 2, and 3. The number of offset coefficients increases from 1 for DC-only to 4, 9, and 16 for orders 0–1, 0–2, and 0–3, respectively. Storage and training-time GPU memory increase accordingly.

Table 7. Effect of higher-order SH offsets on rendering quality and memory cost across three benchmark datasets.

Dataset	Method	1×	2×	4×	PSNR					Avg.	Memory Size (MB)
Deep Blending	DC	29.49	29.26	28.63	27.90	27.16	27.35	26.67	–	28.07	889.0
	Order 0–1	29.06	28.71	28.10	27.48	26.80	26.88	25.98	–	27.57	1117.1
	Order 0–2	28.17	27.44	26.79	25.98	25.45	25.57	24.90	–	26.33	1464.3
	Order 0–3	27.11	25.12	24.72	24.24	23.93	23.73	23.20	–	24.58	4637.5
Tanks and Temples	DC	23.85	22.02	21.38	21.00	20.55	20.95	20.93	–	21.53	619.9
	Order 0–1	23.85	21.59	21.79	21.47	20.99	21.30	21.36	–	21.76	776.5
	Order 0–2	23.85	21.21	21.61	21.28	20.80	21.18	21.27	–	21.60	1022.9
	Order 0–3	23.77	20.18	20.99	20.76	20.45	20.70	20.52	–	21.05	3260.7
Mip-NeRF 360	DC	27.70	26.16	26.35	25.58	26.35	25.92	26.05	23.27	25.92	1044.4
	Order 0–1	27.59	25.35	26.60	25.84	26.53	26.04	26.28	23.92	26.02	1279.9
	Order 0–2	27.33	25.00	26.03	25.55	26.18	25.73	25.87	23.86	25.70	1645.6
	Order 0–3	–	–	–	–	–	–	–	–	–	–

Note: Entries marked with “–” indicate unavailable scales or metrics under the reference evaluation setting [5]. For Mip-NeRF 360 with Order 0–3, training could not be completed because the GPU memory limit was exceeded. The best result at each scale is shown in bold.

Higher-order offsets provide no consistent benefit across the three datasets. On Deep Blending, rendering quality decreases as the offset is extended to higher orders. Orders

0–1 yield only marginal gains on Tanks and Temples and Mip-NeRF 360, while extending the offset beyond order 1 brings no further improvement.

The order 0–3 configuration also exceeds the available GPU memory on Mip-NeRF 360. The marginal gains from orders 0–1 are dataset-specific and do not persist at higher orders, while the memory cost continues to increase with SH order. Based on this quality–memory trade-off, we select DC-only as the final ACO configuration. At the representation level, extending the scale-specific offset to the full SH representation may interfere with the shared view-dependent components and may not provide a consistent benefit. Consistent with this interpretation, the DC-term offset offers a practical design for ACO because it corrects scale-dependent color shifts while leaving the view-dependent SH components unchanged.

Figure 5 examines the training-loss behavior after densification to assess whether the module effects observed in the ablation studies lead to a lower training loss with limited fluctuations. During the early iterations, MS-GS and ACO-EGR exhibit similar L1 loss because the reallocated Gaussians require subsequent optimization updates before their contribution becomes visible. As training proceeds, ACO-EGR reaches a lower L1 loss than MS-GS and preserves this advantage through the final training phase. The difference between the two loss curves becomes largest near the final iterations. This observed behavior indicates a lower final training loss with limited fluctuations, providing empirical evidence of stable training behavior without establishing formal convergence.

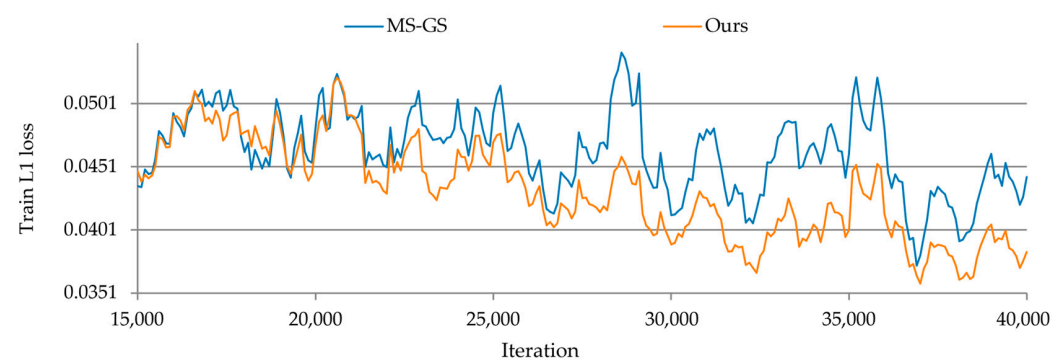


Figure 5. Training L1 loss of MS-GS and Ours on the Mip-NeRF 360 flowers scene after densification.

Rendering-scale variation affects image-plane spatial sampling, while SH order represents angular frequency with respect to the viewing direction. Because all rendering scales use the same camera poses, the SH evaluation direction of each Gaussian remains unchanged across scales.

At coarser scales, each pixel covers a larger spatial region and combines contributions from more Gaussians. This suggests that the resulting mismatch is more closely related to changes in aggregated pixel color than to changes in viewing direction. Based on this reasoning, ACO applies the scale-specific correction to the direction-independent DC coefficient, while the higher-order SH coefficients remain shared across scales.

To examine this design, Figure 6 reports the mean absolute magnitude of the SH coefficients across coefficient indices under the MS-GS setting on Deep Blending. The DC coefficient has a substantially larger magnitude than the higher-order SH coefficients. The SH coefficient analysis and higher-order offset ablation provide empirical support for using the DC coefficient as the primary target for scale-specific color correction. Thus, the DC-only design is motivated by this mechanism and supported by empirical results rather than by a formal analytical derivation.

Once densification is complete and the primitive count is fixed, EGR reallocates low-contribution Gaussians to regions with persistent reconstruction error. In the final

training phase, the reallocated primitives further reduce reconstruction error within the fixed Gaussian set.

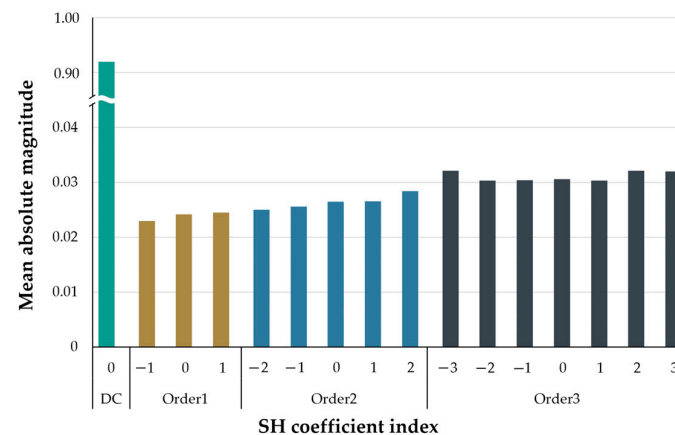


Figure 6. Mean absolute magnitude of the SH coefficients across coefficient indices under the MS-GS setting on Deep Blending.

Figure 7 evaluates the training-loss perturbation associated with EGR. The deviation is normalized by the standard deviation σ of the pre-anchor loss fluctuation. Across the 13 evaluation scenes, the median maximum deviation is 4.0σ with EGR and 3.8σ without EGR. The comparable deviations indicate that the relocation-induced perturbation remains within the intrinsic fluctuation of multi-scale training.

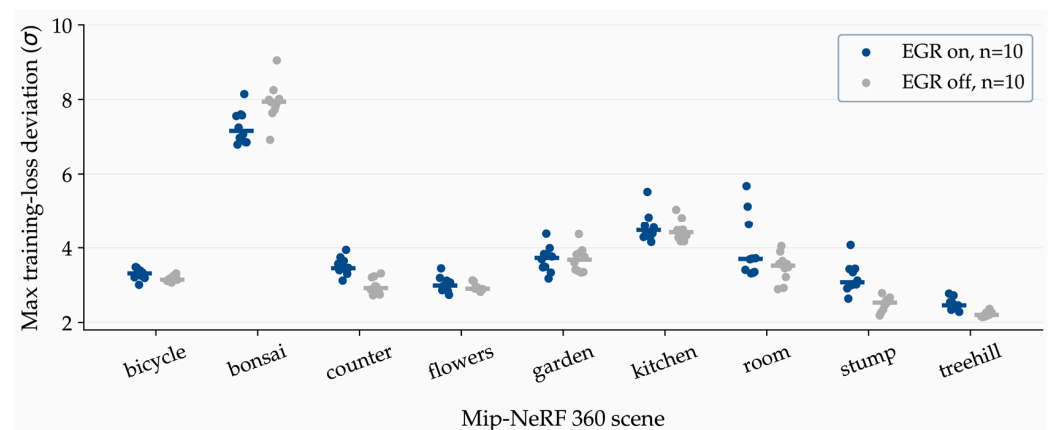


Figure 7. Maximum training-loss deviation after each EGR round relative to the EGR-disabled run.

Figure 8 summarizes the reallocation behavior of EGR across ten rounds on the three benchmarks. The relocation ratio decreases from 33–36% in the first round to 23–27% in the final round. About 35% of the Gaussians remain at their original positions. Among the relocated Gaussians, the average number of relocations is 4.2 across the ten rounds, while only 0.7% of all Gaussians are relocated in every round. These results show that the aggregate relocation activity decreases across the bounded window and that repeated relocation is confined to a small subset of Gaussians.

Figure 9 tracks the attribute differences between each relocated LC Gaussian and its paired HE Gaussian on the Tanks and Temples train scene.

The differences in the appearance and opacity attributes rapidly increase after relocation and reach 59–96% of the corresponding distances between unrelated Gaussian pairs within 1000 iterations. Although the paired Gaussians remain co-located by design, their appearance and opacity attributes become distinct during subsequent optimization. These results show that the shared initialization does not persist as identical optimized attributes.

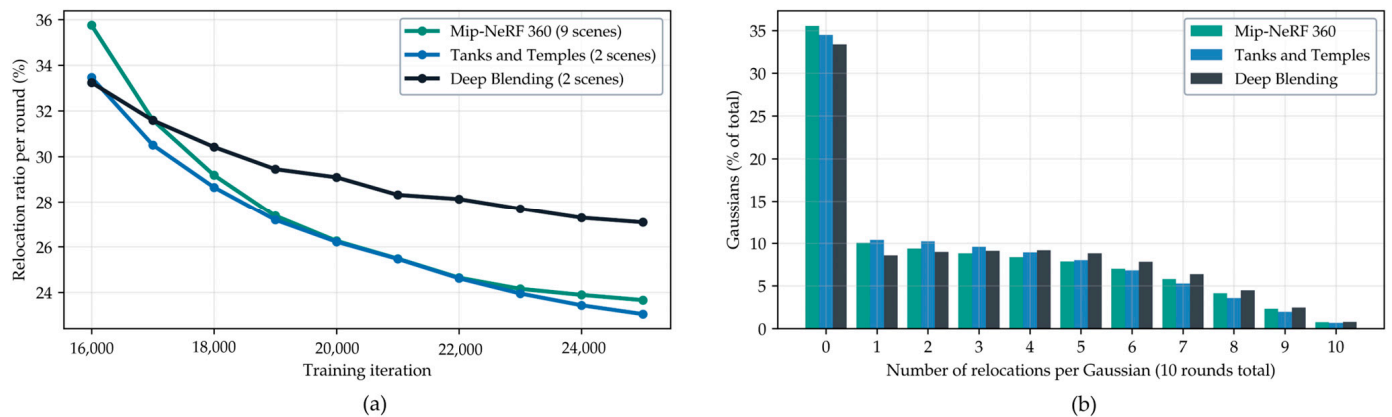


Figure 8. EGR reallocation statistics averaged over the scenes in each benchmark during iterations 16,000–25,000. (a) Relocation ratio at each round. (b) Distribution of the number of relocations per Gaussian over ten rounds.

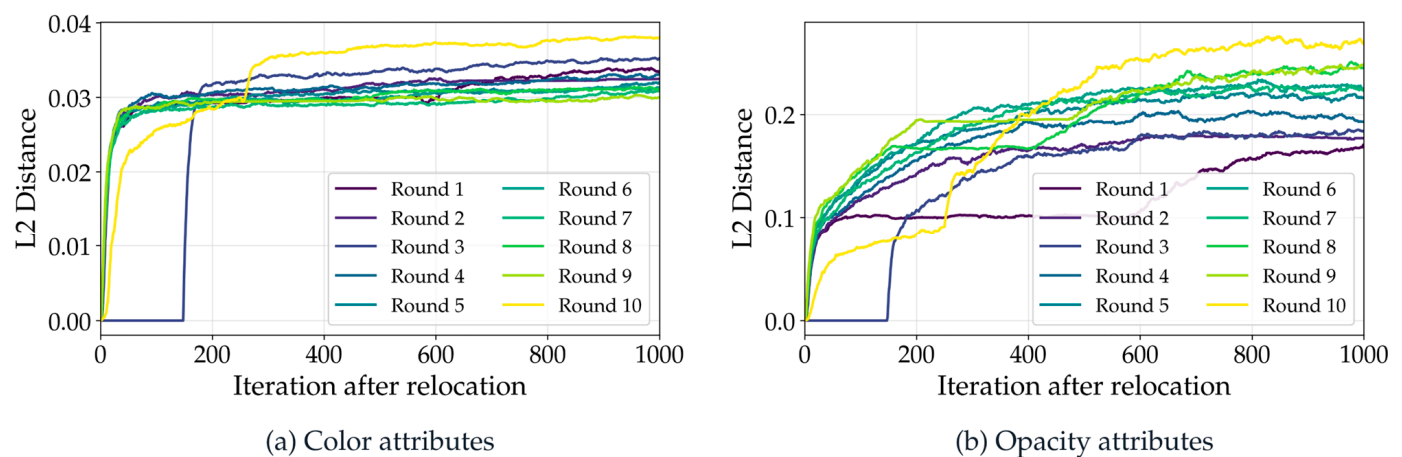


Figure 9. Differences between the relocated LC and HE Gaussians after EGR.

Table 8 evaluates the role of coarse-scale preservation in EGR on the Mip-NeRF 360 scenes. The configuration without preservation differs from the configuration without EGR in Table 6. It keeps the EGR reallocation process but removes the preservation component that protects Gaussians with stronger coarse-scale contribution. At $1\times$ and $2\times$, the two configurations produce similar results because fine-scale details dominate the reconstruction error. As the rendering scale becomes coarser, removing preservation increases the PSNR loss. The PSNR losses reach 0.79 dB at $16\times$, 1.09 dB at $64\times$, and 0.99 dB at $128\times$, which confirms the need for coarse-scale preservation. Without preservation, reallocation can reassign Gaussians that retain low-resolution structural information. The preservation gate protects Gaussians that remain important at coarse scales from being relocated.

The configuration without EGR in Table 6 skips reallocation entirely, so Gaussians supporting coarse-scale reconstruction remain in place. In contrast, the configuration without preservation in Table 8 retains reallocation while removing this safeguard, which explains its larger degradation at coarse scales.

At $64\times$, the configuration without EGR is only 0.16 dB higher than the full model, and the two configurations have the same PSNR at $128\times$. At these very coarse scales, extensive spatial aggregation limits the benefit of local primitive reallocation, while the perturbation introduced by relocation remains. The benefit of EGR is therefore concentrated in the scale range where local structural errors can be effectively reduced through primitive reallocation.

Table 8. Effect of the EGR coarse-scale preservation under the MS-GS setting on the Mip-NeRF 360 scenes.

Config	1×			2×			4×			8×		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
w/o preservation	27.67	0.8226	0.1260	26.19	0.8155	0.1192	26.15	0.8085	0.0996	25.19	0.7980	0.0913
Ours	27.70	0.8226	0.1264	26.16	0.8154	0.1208	26.35	0.8148	0.0965	25.58	0.8128	0.0856

Config	16×			32×			64×			128×		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
w/o preservation	25.56	0.8329	0.0570	25.16	0.8651	0.0247	24.96	0.9026	–	22.28	0.8878	–
Ours	26.35	0.8570	0.0478	25.92	0.8869	0.0200	26.05	0.9199	–	23.27	0.9068	–

Note: The best result at each scale is shown in bold.

Table 9 evaluates the composition of the LC score in Equation (9) through a leave-one-out ablation that removes one measure at a time. The highest-performing variant changes across datasets and rendering scales, and no ablated variant consistently outperforms the full LC score. The differences in average PSNR are small: the full score is within 0.09 dB of the best ablated variant on Mip-NeRF 360 and within 0.05 dB on Tanks and Temples, while it matches the best average result on Deep Blending.

Table 9. Leave-one-out ablation of the LC score.

Dataset	Method	PSNR								Avg.
		1×	2×	4×	8×	16×	32×	64×	128×	
Mip-NeRF 360	$C^{mean} + C^{max} + G^{mean}$ (Ours)	27.70	26.16	26.35	25.58	26.35	25.92	26.05	23.27	25.92
	$C^{mean} + C^{max}$	27.72	26.19	26.37	25.60	26.35	25.92	26.07	23.30	25.94
	$C^{mean} + G^{mean}$	27.67	26.14	26.30	25.56	26.31	25.93	26.18	23.42	25.94
	$C^{max} + G^{mean}$	27.66	26.34	26.36	25.64	26.35	25.93	26.11	23.65	26.01
Tanks and Temples	$C^{mean} + C^{max} + G^{mean}$ (Ours)	23.85	22.02	21.38	21.00	20.55	20.95	20.93	–	21.53
	$C^{mean} + C^{max}$	23.81	21.97	21.33	20.89	20.43	20.98	20.89	–	21.47
	$C^{mean} + G^{mean}$	23.84	22.05	21.47	20.99	20.59	21.04	21.05	–	21.58
	$C^{max} + G^{mean}$	23.80	21.95	21.37	20.89	20.30	20.79	20.61	–	21.39
Deep Blending	$C^{mean} + C^{max} + G^{mean}$ (Ours)	29.49	29.26	28.63	27.90	27.16	27.35	26.67	–	28.07
	$C^{mean} + C^{max}$	29.40	29.24	28.57	27.89	27.34	27.40	26.53	–	28.05
	$C^{mean} + G^{mean}$	29.50	29.27	28.64	27.89	27.32	27.36	26.53	–	28.07
	$C^{max} + G^{mean}$	29.39	29.17	28.55	27.82	27.26	27.37	26.44	–	28.00

Note: The best result at each scale is shown in bold.

Removing a measure makes the relocation criterion less conservative and may improve PSNR at an individual scale. However, the three measures screen different types of potentially useful Gaussians: C^{mean} identifies Gaussians with low overall contribution, C^{max} protects Gaussians that remain important in a small number of views, and G^{mean} protects Gaussians that are still receiving meaningful optimization updates. We retain all three measures to maintain a fixed and conservative relocation criterion across datasets and rendering scales.

Table 10 evaluates the sensitivity of the LC score to the weighting of the three measures. For this analysis, Equation (9) is generalized as $s^{LC} = \exp(-\lambda_1 z(C^{mean}) - \lambda_2 z(C^{max}) - \lambda_3 z(G^{mean}))$, and seven weight configurations are evaluated on Tanks and Temples and Deep Blending. The highest-performing configuration changes across rendering scales, and no single configuration consistently outperforms the others. On both datasets, the multi-scale average PSNR varies by no more than 0.10 dB across the tested configurations,

indicating that the LC score is largely insensitive to moderate weight changes. We retain the equal-weight configuration as a fixed default because it provides stable performance without dataset- or scale-specific weight tuning.

Table 10. Per-scale PSNR under different LC weight configurations.

Dataset	Config $\lambda_1, \lambda_2, \lambda_3$	PSNR							Avg.
		1×	2×	4×	8×	16×	32×	64×	
Tanks and Temples	(1, 1, 1) (Ours)	23.85	22.02	21.38	21.00	20.55	20.95	20.93	21.53
	(2, 1, 1)	23.79	21.95	21.34	20.96	20.47	21.04	21.04	21.51
	(1, 2, 1)	23.78	21.94	21.42	20.95	20.45	21.02	20.97	21.51
	(1, 1, 2)	23.88	22.05	21.41	20.91	20.46	20.98	20.90	21.51
	(0.5, 1, 1)	23.87	22.02	21.36	21.00	20.55	21.17	21.10	21.58
	(1, 0.5, 1)	23.82	21.99	21.40	21.04	20.60	21.03	21.04	21.56
	(1, 1, 0.5)	23.84	22.03	21.41	20.91	20.52	21.04	21.08	21.55
Deep Blending	(1, 1, 1) (Ours)	29.49	29.26	28.63	27.90	27.16	27.35	26.67	28.07
	(2, 1, 1)	29.45	29.21	28.58	27.85	27.29	27.41	26.42	28.03
	(1, 2, 1)	29.47	29.25	28.61	27.86	27.35	27.51	26.72	28.11
	(1, 1, 2)	29.44	29.18	28.58	27.81	27.24	27.35	26.52	28.02
	(0.5, 1, 1)	29.45	29.23	28.59	27.87	27.31	27.36	26.42	28.03
	(1, 0.5, 1)	29.46	29.22	28.62	27.81	27.26	27.32	26.41	28.01
	(1, 1, 0.5)	29.50	29.27	28.64	27.89	27.32	27.37	26.42	28.06

Note: The best result at each scale is shown in bold.

To evaluate ACO-EGR beyond the MS-GS backbone [5], we apply it to Mip-Splatting [3], Analytic-Splatting [14], and AA-2DGS [16]. These methods were originally evaluated over narrower scale ranges and use training settings different from our MS-GS-based multi-scale setting. For a consistent comparison, the three backbones and their ACO-EGR variants are evaluated under the same 1×–128× protocol.

As shown in Table 11, ACO-EGR increases the multi-scale average PSNR by 2.02 dB for Mip-Splatting, 1.44 dB for AA-2DGS, and 1.96 dB for Analytic-Splatting on Mip-NeRF 360. The gains are consistent across all three backbones and are larger at coarse scales. These results demonstrate that ACO-EGR remains effective beyond the MS-GS backbone.

At scales close to the training resolution, the three backbones already provide strong reconstruction quality. Under the fixed primitive budget, EGR reallocates some Gaussians toward persistent-error regions while preserving coarse-scale support. This may lead to small PSNR decreases at some fine scales while improving reconstruction quality at coarser scales.

Table 11. Generalization of ACO-EGR to three recent anti-aliasing 3DGS methods on Mip-NeRF 360.

Dataset	Method	PSNR							Avg.
		1×	2×	4×	8×	16×	32×	64×	
Mip-NeRF 360	Mip-Splatting	27.76	28.85	29.60	29.42	28.21	25.69	24.05	27.14
	Mip-Splatting + Ours	27.82	28.51	29.47	29.82	30.08	29.51	28.94	29.16
	AA-2DGS	26.54	27.57	28.53	28.89	28.57	27.36	25.39	27.04
	AA-2DGS + Ours	26.64	27.04	28.17	28.58	29.43	29.27	29.51	28.48
	Analytic-Splatting	27.34	28.56	29.16	28.96	28.22	26.16	23.45	26.68
	Analytic-Splatting + Ours	27.27	28.12	28.88	29.27	29.67	29.57	28.55	27.78

Note: The better result within each backbone pair is shown in bold.

5. Conclusions

This paper proposed ACO-EGR for scale-consistent multi-scale 3D Gaussian splatting. The method is motivated by fixed per-Gaussian color and post-densification primitive allocation, both of which become more evident at coarse rendering scales. ACO corrects scale-dependent color shifts through scale-specific DC-term offsets. EGR reallocates low-contribution Gaussians to regions with persistent reconstruction errors and preserves coarse-scale structural primitives. Experimental results show that ACO-EGR improves rendering quality primarily at coarse scales on Mip-NeRF 360, Tanks and Temples, and Deep Blending under the MS-GS setting. On Mip-NeRF 360, the PSNR gains over MS-GS reached 1.96 dB at $4\times$ and 2.09 dB at $128\times$, with a marginal increase in inference-time cost. The ablation studies confirmed that ACO provides the main scale-dependent color correction and that EGR refines Gaussian allocation in scale ranges where local structure still affects reconstruction quality. Further work will extend ACO-EGR toward broader scale-adaptive 3D scene representations with more flexible color adaptation and allocation strategies.

Author Contributions: Conceptualization, H.P., Y.L., D.L. and D.J.; methodology, H.P., Y.L., D.L. and D.J.; software, H.P., Y.L. and D.L.; validation, H.P., Y.L. and D.J.; formal analysis, D.J.; investigation, H.P., Y.L., D.L. and D.J.; resources, D.J.; data curation, H.P. and Y.L.; writing—original draft preparation, H.P. and Y.L.; writing—review and editing, D.J.; visualization, H.P., Y.L. and D.L.; supervision, D.J.; project administration, D.J.; funding acquisition, D.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (RS-2023-00246528).

Data Availability Statement: The data presented in this study are openly available. The Mip-NeRF 360 dataset is available at <https://jonbarron.info/mipnerf360/>, the Tanks and Temples dataset at <https://www.tanksandtemples.org/download/>, and the Deep Blending dataset at <http://visual.cs.ucl.ac.uk/pubs/deepblending/> (all accessed on 25 June 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Mildenhall, B.; Srinivasan, P.P.; Tancik, M.; Barron, J.T.; Ramamoorthi, R.; Ng, R. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *Commun. ACM* **2022**, *65*, 99–106. [CrossRef]
2. Kerbl, B.; Kopanas, G.; Leimkühler, T.; Drettakis, G. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.* **2023**, *42*, 139. [CrossRef]
3. Yu, Z.; Chen, A.; Huang, B.; Sattler, T.; Geiger, A. Mip-Splatting: Alias-Free 3D Gaussian Splatting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–22 June 2024; pp. 19447–19456.
4. Li, J.; Shi, Y.; Cao, J.; Ni, B.; Zhang, W.; Zhang, K.; Van Gool, L. Mipmap-GS: Let Gaussians Deform with Scale-Specific Mipmap for Anti-Aliasing Rendering. In Proceedings of the International Conference on 3D Vision (3DV), Singapore, 25–28 March 2025; pp. 935–945.
5. Yan, Z.; Low, W.F.; Chen, Y.; Lee, G.H. Multi-Scale 3D Gaussian Splatting for Anti-Aliased Rendering. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–22 June 2024; pp. 20923–20931.
6. Liu, J.; Kong, L.; Yan, J.; Chen, G. Mesh-Aligned 3D Gaussian Splatting for Multi-Resolution Anti-Aliasing Rendering. *IEEE Trans. Circuits Syst. Video Technol.* **2025**, *35*, 7368–7379. [CrossRef]
7. Gu, X.; Chang, Y.; Xia, J.; Yu, Y.; Tian, Z.; Chen, J. OptiNeRF: A Spatially Optimized Neural Rendering Framework for Complex Scene Reconstruction. *Mathematics* **2026**, *14*, 842. [CrossRef]
8. Müller, T.; Evans, A.; Schied, C.; Keller, A. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.* **2022**, *41*, 102. [CrossRef]
9. Fridovich-Keil, S.; Yu, A.; Tancik, M.; Chen, Q.; Recht, B.; Kanazawa, A. Plenoxels: Radiance Fields Without Neural Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 5501–5510.
10. Chen, A.; Xu, Z.; Geiger, A.; Yu, J.; Su, H. TensorRF: Tensorial Radiance Fields. In Proceedings of the European Conference on Computer Vision (ECCV), Tel Aviv, Israel, 23–27 October 2022; pp. 333–350.

11. Barron, J.T.; Mildenhall, B.; Tancik, M.; Hedman, P.; Martin-Brualla, R.; Srinivasan, P.P. Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 5855–5864.
12. Barron, J.T.; Mildenhall, B.; Verbin, D.; Srinivasan, P.P.; Hedman, P. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 5470–5479.
13. Barron, J.T.; Mildenhall, B.; Verbin, D.; Srinivasan, P.P.; Hedman, P. Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 1–6 October 2023; pp. 19697–19705.
14. Liang, Z.; Zhang, Q.; Hu, W.; Zhu, L.; Feng, Y.; Jia, K. Analytic-Splatting: Anti-Aliased 3D Gaussian Splatting via Analytic Integration. In Proceedings of the European Conference on Computer Vision (ECCV), Milan, Italy, 29 September–4 October 2024; pp. 281–297.
15. Song, X.; Zheng, J.; Yuan, S.; Gao, H.; Zhao, J.; He, X.; Gu, W.; Zhao, H. SA-GS: Scale-Adaptive Gaussian Splatting for Training-Free Anti-Aliasing. *arXiv* **2024**, arXiv:2403.19615.
16. Younes, M.; Boukhayma, A. Anti-Aliased 2D Gaussian Splatting. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), San Diego, CA, USA, 2–7 December 2025; pp. 9767–9792.
17. Martin-Brualla, R.; Radwan, N.; Sajjadi, M.S.M.; Barron, J.T.; Dosovitskiy, A.; Duckworth, D. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 19–25 June 2021; pp. 7210–7219.
18. Lin, J.; Li, Z.; Tang, X.; Liu, J.; Liu, S.; Liu, J.; Lu, Y.; Wu, X.; Xu, S.; Yan, Y.; et al. VastGaussian: Vast 3D Gaussians for Large Scene Reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 16–22 June 2024; pp. 5166–5175.
19. Fan, Z.; Wang, K.; Wen, K.; Zhu, Z.; Xu, D.; Wang, Z. LightGaussian: Unbounded 3D Gaussian Compression with 15x Reduction and 200+ FPS. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Vancouver, BC, Canada, 10–15 December 2024; pp. 140138–140158.
20. Ye, Z.; Li, W.; Liu, S.; Qiao, P.; Dou, Y. AbsGS: Recovering Fine Details in 3D Gaussian Splatting. In Proceedings of the 32nd ACM International Conference on Multimedia, Melbourne, Australia, 28 October–1 November 2024; pp. 1053–1061.
21. Kheradmand, S.; Rebain, D.; Sharma, G.; Sun, W.; Tseng, Y.-C.; Isack, H.; Kar, A.; Tagliasacchi, A.; Yi, K.M. 3D Gaussian Splatting as Markov Chain Monte Carlo. In Proceedings of the Advances in Neural Information Processing Systems (NeurIPS), Vancouver, BC, Canada, 10–15 December 2024; pp. 80965–80986.
22. Zhang, Z.; Hu, W.; Lao, Y.; He, T.; Zhao, H. Pixel-GS: Density Control with Pixel-Aware Gradient for 3D Gaussian Splatting. In Proceedings of the European Conference on Computer Vision (ECCV), Milan, Italy, 29 September–4 October 2024; pp. 326–342.
23. Fang, G.; Wang, B. Mini-Splatting: Representing Scenes with a Constrained Number of Gaussians. In Proceedings of the European Conference on Computer Vision (ECCV), Milan, Italy, 29 September–4 October 2024; pp. 165–181.
24. Knapitsch, A.; Park, J.; Zhou, Q.-Y.; Koltun, V. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM Trans. Graph.* **2017**, *36*, 78. [[CrossRef](#)]
25. Hedman, P.; Philip, J.; Price, T.; Frahm, J.-M.; Drettakis, G.; Brostow, G. Deep Blending for Free-Viewpoint Image-Based Rendering. *ACM Trans. Graph.* **2018**, *37*, 257. [[CrossRef](#)]
26. Huynh-Thu, Q.; Ghanbari, M. Scope of Validity of PSNR in Image/Video Quality Assessment. *Electron. Lett.* **2008**, *44*, 800–801. [[CrossRef](#)]
27. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [[CrossRef](#)] [[PubMed](#)]
28. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 586–595.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.