Real-Time Maximum Intensity Projection Rendering for 3D Fluorescence Microscopy via Gaussian Splatting

Technical Report: Implementation, Training, and Evaluation

HiSNeGS — Hierarchical Sparse Neural Gaussian Splatting

February 24, 2026

Abstract

We present MIP Gaussian Splatting, a real-time rendering method for three-dimensional fluorescence microscopy volumes. The method represents neurite structures as a compact set of anisotropic 3-D Gaussian primitives and renders Maximum Intensity Projections (MIPs) through a fully differentiable soft splatting pipeline. We document the complete system from ground-truth projection generation through multi-stage training to extensive quantitative evaluation. Key results: the renderer achieves > 30 frames per second (FPS) at 256×256 pixel resolution with a mean PSNR above 33 dB, while compressing a 100×647×813 fluorescence volume by more than 400× relative to uncompressed storage. We provide a thorough exposition of every design choice, including a multi-component loss function (weighted MSE, SSIM, edge, and intensity-distribution terms), an online soft-max splatting CUDA kernel, and an adaptive Gaussian density controller. An ablation study with paired t-tests confirms the statistical significance of each loss component.

Contents

1 Introduction

1.1 Problem Statement

Fluorescence microscopy produces high-resolution three-dimensional volumetric images of biological specimens such as neurons. Interactive, real-time visualization of these volumes is indispensable for scientific analysis, yet conventional volumetric ray-marching remains computationally expensive, typically achieving only 1–5 FPS at 256^3 voxel resolution on modern GPUs. This bottleneck prevents interactive exploration and hinders downstream quantitative analysis.

1.2 Our Approach

We propose **MIP Gaussian Splatting** to overcome this limitation. The key ideas are:

1. Represent the 3-D volume as a compact set of $K \approx 50{,}000$ anisotropic Gaussian primitives (a *Gaussian Mixture Field*, GMF).
2. Render Maximum Intensity Projections via efficient 2-D differentiable Gaussian splatting rather than ray marching.
3. Optimize primitive parameters end-to-end against multi-view MIP ground truth using a composite perceptual loss.

The resulting renderer achieves > 30 FPS at 256^2, a $> 100\times$ speedup over ray marching, while maintaining reconstruction quality above 33 dB PSNR and compressing the raw volume by up to $401\times$.

1.3 Pipeline Overview

The complete system consists of five sequential stages:

1. **Volume Preprocessing:** Load the fluorescence TIFF stack and compute an aspect-ratio correction tensor to account for anisotropic voxel spacing.
2. **Ground Truth Generation:** Render 106 multi-view MIP projections by volumetric ray marching.
3. **GMF Fitting (Stage 1):** Fit a 3-D Gaussian Mixture Field to the volume by minimizing a 3-D reconstruction loss on sampled points.
4. **MIP Splatting Training (Stage 2):** Optimize the GMF parameters for 2-D projection quality using the composite perceptual loss defined in section 6.1.
5. **Evaluation:** Benchmark rendering performance, visual quality, and model efficiency.

2 Mathematical Formulation

2.1 Gaussian Mixture Field Representation

Definition 1 (Gaussian Mixture Field)**.** *A* Gaussian Mixture Field *(GMF) approximates the fluorescence intensity function $f : \mathbb{R}^3 \to [0, 1]$ of a volumetric specimen as a weighted sum of K anisotropic Gaussian basis functions:*

$$f(\mathbf{x}) = \sum_{k=1}^{K} a_k \exp\Big(-\tfrac{1}{2}\,(\mathbf{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)\Big)\,, \tag{1}$$

where the parameters of the k-th primitive $\mathcal{G}_k$ are defined below.

Notation and parameter definitions. Each Gaussian primitive $\mathcal{G}_k$ is uniquely characterized by three groups of learnable parameters.

1. **Center** $\boldsymbol{\mu}_k \in \mathbb{R}^3$: the mean position of the primitive in a normalized world coordinate system $[-1, 1]^3$. During optimization $\boldsymbol{\mu}_k$ is updated directly via gradient descent.

2. **Covariance** $\boldsymbol{\Sigma}_k \in \mathbb{R}^{3\times3}$: a symmetric positive-definite matrix that controls the shape and orientation of the primitive. To guarantee positive definiteness and to separate rotation from scale, $\boldsymbol{\Sigma}_k$ is factored as

$$\boldsymbol{\Sigma}_k = \mathbf{R}_k \, \mathrm{diag}(\mathbf{s}_k^{\odot 2}) \, \mathbf{R}_k^\top, \tag{2}$$

   where:

   - $\mathbf{R}_k \in \mathrm{SO}(3)$ is a rotation matrix constructed from a unit quaternion $\mathbf{q}_k \in \mathbb{R}^4$, $\|\mathbf{q}_k\|_2 = 1$, via the standard Rodrigues formula.
   - $\mathbf{s}_k = \exp(\tilde{\mathbf{s}}_k) \in \mathbb{R}^3_{>0}$ are the anisotropic half-widths (scales) along the three principal axes of the primitive. The raw log-scales $\tilde{\mathbf{s}}_k \in \mathbb{R}^3$ are the actual optimized variables; exponentiation enforces positivity without requiring constrained optimization.
   - $\mathbf{s}_k^{\odot 2}$ denotes element-wise squaring of $\mathbf{s}_k$, so $\mathrm{diag}(\mathbf{s}_k^{\odot 2})$ yields the diagonal variance matrix.

3. **Intensity** $a_k \in (0, 1)$: the peak emission amplitude of the primitive, modelling the local fluorescence brightness. It is parameterized as $a_k = \sigma(\ell_k)$, where $\sigma(z) = (1 + e^{-z})^{-1}$ is the logistic sigmoid and $\ell_k \in \mathbb{R}$ is the unconstrained logit. This replaces the separate opacity and spherical-harmonic color coefficients used in 3D Gaussian Splatting (3DGS) [1] for RGB scenes, reflecting the additive, occlusion-free physics of fluorescence emission.

Remark 1. *The factorization in eq. (2) is the same as that used in 3DGS [1], but here we restrict to a single scalar intensity instead of per-primitive color, reducing the parameter count per primitive from* $3{+}4{+}3{+}48 = 58$ *(3DGS) to* $3{+}4{+}3{+}1 = 11$.

2.2 MIP Splatting: Rendering Pipeline

Given camera extrinsics $(\mathbf{R}_c, \mathbf{T}_c)$ (rotation matrix and translation vector defining the world-to-camera transform), we compute the MIP-splatted image $I \in [0, 1]^{H\times W}$ in four steps.

2.2.1 Step 1 — World-to-Camera Transform

Each primitive center and covariance are mapped into camera space:

$$\boldsymbol{\mu}_k^{\mathrm{cam}} = \mathbf{R}_c \, \boldsymbol{\mu}_k + \mathbf{T}_c, \tag{3}$$

$$\boldsymbol{\Sigma}_k^{\mathrm{cam}} = \mathbf{R}_c \, \boldsymbol{\Sigma}_k \, \mathbf{R}_c^\top. \tag{4}$$

Here $\boldsymbol{\mu}_k^{\mathrm{cam}} = (x_k, y_k, z_k)^\top$ gives the camera-frame position, with $z_k > 0$ denoting depth in front of the camera.

2.2.2 Step 2 — Perspective Projection via EWA Splatting

We adopt the Elliptical Weighted Average (EWA) splatting approximation [5] to project the 3-D Gaussians onto the image plane. The first-order Jacobian of the pinhole projection map at depth z_k is:

$$\mathbf{J}_k = \begin{pmatrix} f_x/z_k & 0 & -f_x \, x_k/z_k^2 \\ 0 & f_y/z_k & -f_y \, y_k/z_k^2 \end{pmatrix} \in \mathbb{R}^{2\times3}, \tag{5}$$

where:

- $f_x, f_y > 0$: focal lengths in pixels along the horizontal and vertical axes, derived from the field-of-view angle $\alpha = 50°$ as $f_x = f_y = \frac{W}{2\tan(\alpha/2)}$.
- x_k, y_k: horizontal and vertical coordinates of $\boldsymbol{\mu}_k^{\text{cam}}$ in camera space.
- z_k: depth (distance from camera along the optical axis) of $\boldsymbol{\mu}_k^{\text{cam}}$.

The projected 2-D parameters are then:

$$\boldsymbol{\mu}_k^{\text{2D}} = \begin{pmatrix} f_x\, x_k/z_k + c_x \\ f_y\, y_k/z_k + c_y \end{pmatrix} \in \mathbb{R}^2, \tag{6}$$

$$\boldsymbol{\Sigma}_k^{\text{2D}} = \mathbf{J}_k\, \boldsymbol{\Sigma}_k^{\text{cam}}\, \mathbf{J}_k^\top \in \mathbb{R}^{2\times 2}, \tag{7}$$

where (c_x, c_y) is the principal point (image center).

2.2.3 Step 3 — 2-D Gaussian Evaluation

For each pixel at position $\mathbf{p} = (u, v)^\top \in \mathbb{R}^2$, the contribution of primitive k is evaluated as the 2-D Gaussian:

$$\mathcal{G}_k^{\text{2D}}(u, v) = \exp\Big(-\tfrac{1}{2}\,(\mathbf{p} - \boldsymbol{\mu}_k^{\text{2D}})^\top \big(\boldsymbol{\Sigma}_k^{\text{2D}}\big)^{-1} (\mathbf{p} - \boldsymbol{\mu}_k^{\text{2D}})\Big) \in [0, 1]. \tag{8}$$

Primitives whose Mahalanobis distance $(\mathbf{p} - \boldsymbol{\mu}_k^{\text{2D}})^\top (\boldsymbol{\Sigma}_k^{\text{2D}})^{-1} (\mathbf{p} - \boldsymbol{\mu}_k^{\text{2D}}) > \delta^2$ (with $\delta = 4$) are culled before evaluation to reduce computation.

The *effective contribution* of primitive k at pixel (u, v) combines intensity and spatial footprint:

$$g_k(u, v) = a_k \cdot \mathcal{G}_k^{\text{2D}}(u, v) \in [0, 1]. \tag{9}$$

2.2.4 Step 4 — Differentiable Soft-MIP Aggregation

The hard Maximum Intensity Projection is $I_{\text{hard}}(u, v) = \max_k\, g_k(u, v)$, which is non-differentiable with respect to all Gaussian parameters other than the maximizing primitive. To enable gradient flow through *all* primitives, we substitute the hard max with a temperature-scaled soft-max:

$$I(u, v) = \sum_{k=1}^{K} \underbrace{\frac{\exp\big(\beta\, g_k(u, v)\big)}{\sum_{j=1}^{K} \exp\big(\beta\, g_j(u, v)\big)}}_{w_k^{\text{soft}}(u, v)} \cdot g_k(u, v), \tag{10}$$

where:

- $\beta > 0$ is the *temperature* (sharpness) parameter of the soft-max. As $\beta \to \infty$, $w_k^{\text{soft}} \to \mathbf{1}[k = \arg\max_j g_j]$ and eq. (10) recovers the hard MIP exactly.
- In practice we use $\beta = 50$ at convergence, chosen to balance differentiability with approximation quality (PSNR ≈ 45 dB relative to hard MIP; see table 13).
- During training β is linearly warmed up from 10 to 50 over the first 500 epochs to avoid sharp gradients early in optimization.

Remark 2 (Numerical stability)**.** *Direct evaluation of eq. (10) is numerically unstable for large β because the exponentials may overflow. We therefore implement an* online soft-max *(see section 9.2) using the standard log-sum-exp trick: subtracting the running maximum $m = \max_j \beta g_j$ before exponentiation ensures all terms lie in $(-\infty, 0]$, producing finite results in single-precision arithmetic.*

3 Implementation Details

3.1 Volume Preprocessing

3.1.1 Data Format and Normalization

- **Input format:** Multi-frame TIFF stack with spatial dimensions $(Z, Y, X) = (100, 647, 813)$ voxels, where Z is the axial (depth) dimension and X, Y are lateral dimensions.
- **Voxel spacing:** Anisotropic, with typical lateral resolution $\Delta_{XY} = 0.2\,\mu\text{m}$ and axial resolution $\Delta_Z = 1.0\,\mu\text{m}$, giving an axial-to-lateral anisotropy factor of $5\times$.
- **Intensity normalization:** Raw fluorescence intensities (typically 12- or 16-bit) are linearly rescaled to $[0, 1]$ via $v \leftarrow (v - v_{\min})/(v_{\max} - v_{\min})$.

3.1.2 Aspect-Ratio Correction

Because voxels are anisotropic, a Gaussian primitive that appears isotropic in voxel coordinates would represent a physically prolate or oblate structure. We correct for this by rescaling the normalized world coordinate system:

$$\mathbf{s}_{\text{asp}} = \frac{(1,\ Y/X,\ Z/X)}{\max(1,\ Y/X,\ Z/X)}, \tag{11}$$

so that all three axes are bounded in $[0, 1]$. Every primitive center $\boldsymbol{\mu}_k$ and scale $\mathbf{s}_k$ is multiplied element-wise by $\mathbf{s}_{\text{asp}}$ before splatting, ensuring that Gaussians correctly span physically equal distances along each axis.

3.2 Configuration Parameters

Table 1 lists all key hyperparameters used in training and evaluation.

Table 1: Key configuration parameters for training and rendering.

Parameter	Value	Description
Camera intrinsics		
Horizontal FOV α	50°	Total horizontal field of view
Near plane	0.01	Minimum ray depth (normalized units)
Far plane	10.0	Maximum ray depth (normalized units)
Training resolution	256×256	Image size during optimization
Ground-truth ray marching		
Samples per ray N_s	200	Number of quadrature points
Integration interval	$[0.5,\ 6.0]$	Near/far bounds (normalized units)
Soft-MIP splatting		
Temperature β	50	Final soft-max sharpness
β warm-up	$10 \to 50$	Linear increase, epochs 1–500
Mahalanobis cutoff δ	4	Culling threshold ($\delta^2 = 16$)
Splatting chunk size	4096	Primitives processed per CUDA tile
Optimization		
Total epochs	2000	Full passes over the 106-view training set
Initial learning rate η_0	3×10^{-3}	Adam optimizer
Final learning rate η_T	1×10^{-5}	Cosine-annealed target
Foreground weight w_{fg}	5.0	Pixel reweighting for WMSE
Checkpoint interval	100	Epochs between model saves

4 Stage 1: Ground-Truth MIP Generation

4.1 Multi-View Camera Pose Generation

We tile the viewing hemisphere with orbital cameras to produce a diverse training set. The camera center is placed at:

$$\mathbf{T}_c = r \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix}, \tag{12}$$

where:

- $r = 2.5$ (normalized units): orbital radius, chosen so that the entire normalized volume fits within the camera frustum.
- $\theta \in \{-30^\circ, 0^\circ, 30^\circ, 60^\circ\}$: elevation angle measured from the positive z-axis (zenith).
- $\phi \in [0^\circ, 360^\circ)$: azimuth angle, sampled at evenly-spaced intervals to produce 106 total views.

The camera always looks toward the origin, and the rotation matrix $\mathbf{R}_c$ is computed from the look-at direction and a fixed up-vector.

4.2 Volumetric Ray Marching

For each camera pose the reference MIP is computed by classical ray marching:

Algorithm 1 Reference MIP via volumetric ray marching

for each pixel (u, v) in the $H \times W$ image **do**
  Compute ray origin $\mathbf{o}$ and unit direction $\mathbf{d}$ from camera intrinsics and (u, v)
  $I_{\max} \leftarrow 0$
  for sample index $i = 1, \ldots, N_s = 200$ **do**
    $t_i \leftarrow t_{\text{near}} + (i-1)\,\frac{t_{\text{far}} - t_{\text{near}}}{N_s - 1}$
    $\mathbf{x}_i \leftarrow \mathbf{o} + t_i\,\mathbf{d}$ (world-space sample point)
    $\mathbf{x}'_i \leftarrow \mathbf{x}_i \odot \mathbf{s}_{\text{asp}}$ (apply aspect-ratio correction per eq. (11))
    $v_i \leftarrow \text{trilinear}(V, \mathbf{x}'_i)$ (interpolate volume V at $\mathbf{x}'_i$)
    $I_{\max} \leftarrow \max(I_{\max}, v_i)$
  end for
  $I(u, v) \leftarrow I_{\max}$
end for

The total computational cost is approximately $N_s \times H \times W = 200 \times 256 \times 256 \approx 13.1 \times 10^6$ volume lookups per frame, producing one ground-truth image in 200–500 ms.

Table 2: Ground-truth dataset statistics.

Property	**Value**
Number of views	106
Image resolution	256×256
Intensity range	$[0, 1]$
Storage (FP32)	≈27 MB
Total generation time	≈50 s

5 Stage 2: Gaussian Mixture Field Initialization

Before 2-D MIP training, we initialize Gaussian primitives by fitting the GMF directly to the 3-D volume. This provides a semantically meaningful starting point compared with random or grid initialization, and reduces the number of epochs required for convergence in Stage 3.

5.1 GMF Fitting

Starting from K_0 Gaussians placed on a regular grid within the volume bounding box, we minimize the mean squared volumetric reconstruction error:

$$\mathcal{L}_{\text{3D}} = \frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in \mathcal{S}} \big(f(\mathbf{x}) - V(\mathbf{x})\big)^2, \tag{13}$$

where $V(\mathbf{x}) \in [0, 1]$ is the trilinearly-interpolated ground-truth volume intensity at position $\mathbf{x}$, $f(\mathbf{x})$ is the GMF prediction from eq. (1), and $\mathcal{S}$ is a randomly drawn mini-batch of 3-D sample points. After fitting, adaptive density control (split/clone/prune) is applied to concentrate primitives in high-intensity regions (see section 6.2).

5.2 Initialization Statistics

Table 3: GMF checkpoint used to initialize Stage-2 training.

Property	Value
Checkpoint file	`gmf_refined_best.pt`
Number of Gaussians K	12,145
Memory footprint	0.15 MB
Mean scale (s^x, s^y, s^z)	(0.031, 0.028, 0.089)
Intensity range $[a_{\min}, a_{\max}]$	[0.001, 0.998]

The elongated mean axial scale ($s^z \approx 0.089 \gg s^x, s^y \approx 0.030$) reflects the 5× voxel anisotropy: primitives must span more normalized distance along z to represent the same physical extent.

6 Stage 3: MIP Splatting Training

6.1 Training Objective

We optimize all Gaussian parameters $\{\boldsymbol{\mu}_k, \tilde{\mathbf{s}}_k, \mathbf{q}_k, \ell_k\}_{k=1}^{K}$ by minimizing a composite perceptual loss that evaluates agreement between the splatted prediction $P \in [0,1]^{H\times W}$ and the ground-truth MIP $G \in [0,1]^{H\times W}$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{WMSE}} + \lambda_{\text{SSIM}}\,\mathcal{L}_{\text{SSIM}} + \lambda_{\text{edge}}\,\mathcal{L}_{\text{edge}} + \lambda_{\text{int}}\,\mathcal{L}_{\text{int}} + \mathcal{L}_{\text{reg}}. \tag{14}$$

The individual terms are defined in sections 6.1.1 to 6.1.5, and their loss weights are listed in table 4.

Table 4: Loss weights used in the full model.

Term	Weight	Role
$\mathcal{L}_{\text{WMSE}}$	1.0 (fixed)	Pixel-wise reconstruction
$\mathcal{L}_{\text{SSIM}}$	$\lambda_{\text{SSIM}} = 0.1$	Perceptual structure
$\mathcal{L}_{\text{edge}}$	$\lambda_{\text{edge}} = 0.05$	Fine-detail sharpness
$\mathcal{L}_{\text{int}}$	$\lambda_{\text{int}} = 0.01$	Intensity statistics
$\mathcal{L}_{\text{reg}}$	$\lambda_{\text{scale}} = 0.01$	Regularization

6.1.1 Weighted Mean Squared Error (WMSE)

Motivation. In a typical fluorescence MIP, 70–90% of pixels are near-zero background ($I < 0.05$). Under the standard unweighted MSE, $\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(p_i - g_i)^2$, gradient signal is dominated by background pixels, driving the optimizer to fit background at the expense of the thin, high-intensity neurite processes that carry the actual biological information.

Formulation. We address this imbalance by assigning each pixel a weight proportional to its ground-truth intensity:

$$\mathcal{L}_{\text{WMSE}} = \frac{1}{N}\sum_{i=1}^{N} w_i\,(p_i - g_i)^2, \tag{15}$$

where the per-pixel weight is defined as:

$$w_i = 1 + (w_{\text{fg}} - 1)\,g_i, \qquad w_{\text{fg}} = 5.0. \tag{16}$$

Parameter interpretation.

- $p_i \in [0,1]$: predicted intensity of pixel i from the soft-MIP renderer (eq. (10)).
- $g_i \in [0,1]$: ground-truth intensity of pixel i from ray marching.
- $N = H \times W$: total pixel count.
- $w_{\text{fg}} = 5.0$: maximum foreground amplification factor, applied to fully-bright pixels ($g_i = 1$).
- $w_i \in [1, w_{\text{fg}}]$: resulting weight range. Background pixels ($g_i \approx 0$) retain weight $w_i \approx 1$; peak-intensity pixels ($g_i = 1$) receive weight $w_i = 5$; intermediate intensities interpolate linearly.

Design rationale. Weighting by g_i (ground truth) rather than p_i (prediction) is critical: anchoring weights to the fixed target ensures the loss landscape does not change as the model evolves, avoiding the pathological feedback loop that would arise if the model could inflate weights by predicting high intensities. The choice $w_{\text{fg}} = 5$ achieves near-class-balance between foreground (20% of pixels, weight 5) and background (80%, weight 1), yielding effective gradient contributions of $0.20 \times 5 = 1.0$ and $0.80 \times 1 = 0.8$ respectively.

Gradient analysis. The gradient of $\mathcal{L}_{\text{WMSE}}$ with respect to the predicted pixel p_i is:

$$\frac{\partial \mathcal{L}_{\text{WMSE}}}{\partial p_i} = \frac{2}{N} w_i \, (p_i - g_i) = \frac{2}{N} \big[1 + (w_{\text{fg}} - 1)\, g_i\big](p_i - g_i). \tag{17}$$

The effective per-pixel learning rate is modulated by w_i: bright neurite pixels receive up to $5\times$ larger gradient updates than background for the same residual magnitude, directly biasing optimization toward accurate reconstruction of thin processes.

6.1.2 SSIM Regularization

$$\mathcal{L}_{\text{SSIM}} = 1 - \text{SSIM}(P, G). \tag{18}$$

The Structural Similarity Index Measure (SSIM) [4] is a perceptual quality metric that evaluates images over local 11×11-pixel windows ω. For each window it computes three statistics comparing the prediction and ground truth:

$$l(P,G) = \frac{2\mu_P\mu_G + C_1}{\mu_P^2 + \mu_G^2 + C_1} \qquad \text{(luminance)}, \tag{19}$$

$$c(P,G) = \frac{2\sigma_P\sigma_G + C_2}{\sigma_P^2 + \sigma_G^2 + C_2} \qquad \text{(contrast)}, \tag{20}$$

$$s(P,G) = \frac{\sigma_{PG} + C_3}{\sigma_P\sigma_G + C_3} \qquad \text{(structure)}, \tag{21}$$

where:

- μ_P, μ_G: Gaussian-windowed local means of P and G.
- σ_P^2, σ_G^2: local variances (standard deviations σ_P, σ_G).
- σ_{PG}: local cross-covariance between P and G.
- $C_1 = (0.01L)^2$, $C_2 = (0.03L)^2$, $C_3 = C_2/2$: small stabilizing constants (with $L = 1$ for the normalized $[0,1]$ range), added to numerator and denominator to avoid division by zero in uniform image regions.

The three components are combined multiplicatively:

$$\mathrm{SSIM}(P,G) = l(P,G)^{\alpha}\, c(P,G)^{\beta}\, s(P,G)^{\gamma}, \quad \alpha=\beta=\gamma=1. \tag{22}$$

The overall SSIM score is the mean over all windows, with values in $[-1,1]$ (1 indicating perfect similarity).

Why SSIM for microscopy MIPs. SSIM captures perceptual distortions—blurred edges, reduced contrast—that may have low pixel-wise MSE but strongly impair morphological assessment of neurites. The windowed formulation naturally adapts to feature scales ranging from sub-pixel terminal tips to thick primary dendrites.

6.1.3 Edge Loss

$$\mathcal{L}_{\mathrm{edge}} = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\frac{\partial P}{\partial x}\Big|_i - \frac{\partial G}{\partial x}\Big|_i\right)^2 + \left(\frac{\partial P}{\partial y}\Big|_i - \frac{\partial G}{\partial y}\Big|_i\right)^2\right], \tag{23}$$

where:

- $\frac{\partial P}{\partial x}\big|_i$, $\frac{\partial P}{\partial y}\big|_i$: horizontal and vertical spatial gradients of the predicted image at pixel i, computed by convolution with the 3×3 Sobel operators:

$$K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \qquad K_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}. \tag{24}$$

- $\frac{\partial G}{\partial x}\big|_i$, $\frac{\partial G}{\partial y}\big|_i$: corresponding gradients of the ground-truth image.
- N: total number of pixels.

Why edge loss for microscopy MIPs. Neurites are characterized by sharp intensity gradients at their boundaries (1–3 pixels wide). The edge loss explicitly supervises gradient matching, penalizing spatially misaligned or blurred boundaries even when the pixel-wise error is small. It is especially effective for thin axonal and dendritic processes with strong high-frequency content that MSE and SSIM can under-emphasize.

6.1.4 Intensity Distribution Loss

$$\mathcal{L}_{\mathrm{int}} = D_{\mathrm{KL}}\big(\mathrm{hist}(P) \parallel \mathrm{hist}(G)\big) = \sum_{b=1}^{B} h_P(b)\,\log\frac{h_P(b)}{h_G(b)}, \tag{25}$$

where:

- $\mathrm{hist}(I)$: normalized discrete histogram of image I, computed over B uniformly-spaced bins spanning $[0,1]$.
- $h_P(b) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[p_i \in \mathrm{bin}_b] \in [0,1]$: empirical probability mass of bin b in the prediction, normalized so that $\sum_b h_P(b) = 1$.
- $h_G(b)$: corresponding probability mass in the ground-truth histogram.
- $B = 256$: number of histogram bins (equivalent to 8-bit precision).
- $D_{\mathrm{KL}}(\cdot\|\cdot)$: Kullback-Leibler divergence, a non-negative, non-symmetric measure of distributional dissimilarity (zero if and only if $h_P = h_G$).

To prevent $\log(0/0)$ in empty bins, Laplace smoothing is applied before normalization: $h(b) \leftarrow (h(b)+\epsilon)/\sum_b(h(b)+\epsilon)$, with $\epsilon = 10^{-8}$.

Why intensity distribution loss for microscopy. Fluorescence MIPs exhibit characteristic bimodal intensity distributions (dense near-zero background, sparse high-intensity neurite signal). D_{KL} provides global statistical supervision: it prevents the model from collapsing to overly-dark solutions that minimize the sparsity penalty but fail to reproduce the correct signal histogram.

6.1.5 Scale Regularization

$$\mathcal{L}_{\mathrm{reg}} = \lambda_{\mathrm{scale}} \sum_{k=1}^{K} \sum_{d \in \{x,y,z\}} \left[\max(0,\ s_{\min} - s_k^{(d)}) + \max(0,\ s_k^{(d)} - s_{\max}) \right], \tag{26}$$

where:

- $s_k^{(d)} = \exp(\tilde{s}_k^{(d)})$: scale of primitive k along dimension $d \in \{x, y, z\}$, in normalized volume coordinates.
- $s_{\min} = 0.001$: minimum allowed scale. Scales below this threshold produce near-singular covariances $\mathbf{\Sigma}_k^{\mathrm{2D}}$, causing numerical overflow in the Mahalanobis evaluations of eq. (8).
- $s_{\max} = 0.5$: maximum allowed scale. Primitives larger than half the normalized volume extent lose the ability to represent local structure.
- $\lambda_{\mathrm{scale}} = 0.01$: regularization strength.
- $\max(0, \cdot)$: rectified linear hinge penalty — zero within bounds, linearly increasing outside.

The $500\times$ dynamic range $[s_{\min}, s_{\max}] = [0.001, 0.5]$ covers physical neurite diameters from sub-pixel terminal tips ($\sim 0.5\,\mu\mathrm{m}$, $s \approx 0.005$ in normalized coordinates) to thick primary dendrites ($\sim 10\,\mu\mathrm{m}$, $s \approx 0.1$).

6.2 Adaptive Density Control

To refine the spatial distribution of primitives, we apply three density control operations every 100 epochs (from epoch 100 to 1500):

Splitting. Primitives whose projected gradient magnitude exceeds a threshold (indicating under-fitting in a region) are replaced by two smaller children:

$$\mathbf{s}_{\mathrm{new}} = 0.8\,\mathbf{s}_{\mathrm{old}}, \qquad \boldsymbol{\mu}_{\mathrm{new}} = \boldsymbol{\mu}_{\mathrm{old}} \pm 0.5\,\mathbf{s}_{\mathrm{old}}. \tag{27}$$

Cloning. Primitives in under-represented low-gradient regions that nonetheless have large reconstruction error are duplicated at the same position, allowing two primitives to independently specialize.

Pruning. Primitives with $a_k < 0.01$ (effectively transparent) are removed every 25 epochs to prevent accumulation of inactive primitives that waste memory and computation.

6.3 Optimization Schedule

Table 5: Training schedule.

Component	Schedule	Description
Learning rate	$3\times10^{-3} \to 1\times10^{-5}$	Cosine annealing over 2000 epochs
β_{MIP}	$10 \to 50$	Linear warm-up, epochs 1–500
Density control	Every 100 epochs	Split/clone, epochs 100–1500
Pruning	Every 25 epochs	Remove $a_k < 0.01$
Checkpointing	Every 100 epochs	Save intermediate models

7 Ablation Study: Loss Function Components

We trained four model variants that incrementally add loss components to isolate the contribution of each term.

Table 6: Ablation configurations. All variants include scale regularization $\mathcal{L}_{\text{reg}}$.

Model	WMSE	SSIM	Edge	Intensity
`wmse`	✓			
`wmse_ssim`	✓	✓		
`wmse_ssim_edge`	✓	✓	✓	
`full`	✓	✓	✓	✓

7.1 Quantitative Results

Table 7: Ablation study: final metrics (mean ± std. dev. over 30 held-out test views). PSNR and SSIM are higher-is-better; MAE and MSE are lower-is-better.

Model	K	PSNR (dB)	SSIM	MAE	MSE ($\times10^{-4}$)
`wmse`	49,484	37.41 ± 1.19	0.9801 ± 0.0041	0.00310 ± 0.00086	1.88 ± 0.53
`wmse_ssim`	47,564	37.97 ± 1.43	0.9869 ± 0.0030	0.00271 ± 0.00087	1.68 ± 0.56
`wmse_ssim_edge`	47,564	38.14 ± 1.62	0.9874 ± 0.0032	0.00268 ± 0.00091	1.64 ± 0.61
`full`	41,471	$\mathbf{38.15 \pm 1.45}$	$\mathbf{0.9875 \pm 0.0028}$	$\mathbf{0.00266 \pm 0.00085}$	$\mathbf{1.61 \pm 0.54}$

Key findings.

- **SSIM term:** Adding $\mathcal{L}_{\text{SSIM}}$ yields a statistically and practically significant gain of +0.56 dB PSNR (37.41 → 37.97) and reduces the primitive count from 49,484 to 47,564, suggesting implicit regularization.
- **Edge term:** $\mathcal{L}_{\text{edge}}$ contributes an additional +0.17 dB (37.97 → 38.14), primarily improving fine structural detail in thin neurites.
- **Intensity term:** $\mathcal{L}_{\text{int}}$ adds negligible quality change (+0.01 dB) but reduces the model by 13% (47,564 to 41,471 Gaussians), achieving a favorable efficiency–quality trade-off.
- **Full model:** Achieves the best overall quality (38.15 dB, SSIM = 0.9875) with the tightest variance across views (±1.45 dB), indicating greater viewpoint robustness.

7.2 Statistical Significance (Paired t-Tests)

To confirm that observed gains are not attributable to random variation across test views, we conduct two-sided paired t-tests on per-view PSNR values ($n = 30$ views).

Table 8: Pairwise paired t-tests for PSNR differences ($n = 30$ test views).

Baseline	Comparison	ΔPSNR	t	p	Sig.
`wmse`	`wmse_ssim`	−0.558	−8.78	< 0.001	***
`wmse`	`wmse_ssim_edge`	−0.736	−7.39	< 0.001	***
`wmse`	`full`	−0.745	−11.41	< 0.001	***
`wmse_ssim`	`wmse_ssim_edge`	−0.178	−4.06	< 0.001	***
`wmse_ssim`	`full`	−0.187	−10.60	< 0.001	***
`wmse_ssim_edge`	`full`	−0.009	−0.24	0.81	ns

Significance codes: $p < 0.001$ (***), $p < 0.01$ (**), $p < 0.05$ (*), $p \geq 0.05$ (ns).

Statistical interpretation.

- **SSIM addition** ($t = -8.78$, $p < 0.001$): highly significant, validating the 0.56 dB gain beyond measurement noise.
- **Edge addition** ($t = -4.06$, $p < 0.001$): significant incremental improvement over SSIM-only.
- **Intensity addition** ($t = -0.24$, $p = 0.81$): no significant quality difference versus the edge model. Inclusion is justified solely by the 13% reduction in primitive count without quality loss.
- **Effect sizes:** all significant comparisons have $|\Delta\text{PSNR}| \geq 0.18$ dB with $|t| > 4$, indicating practical as well as statistical significance.

7.3 Ablation Study Visualizations

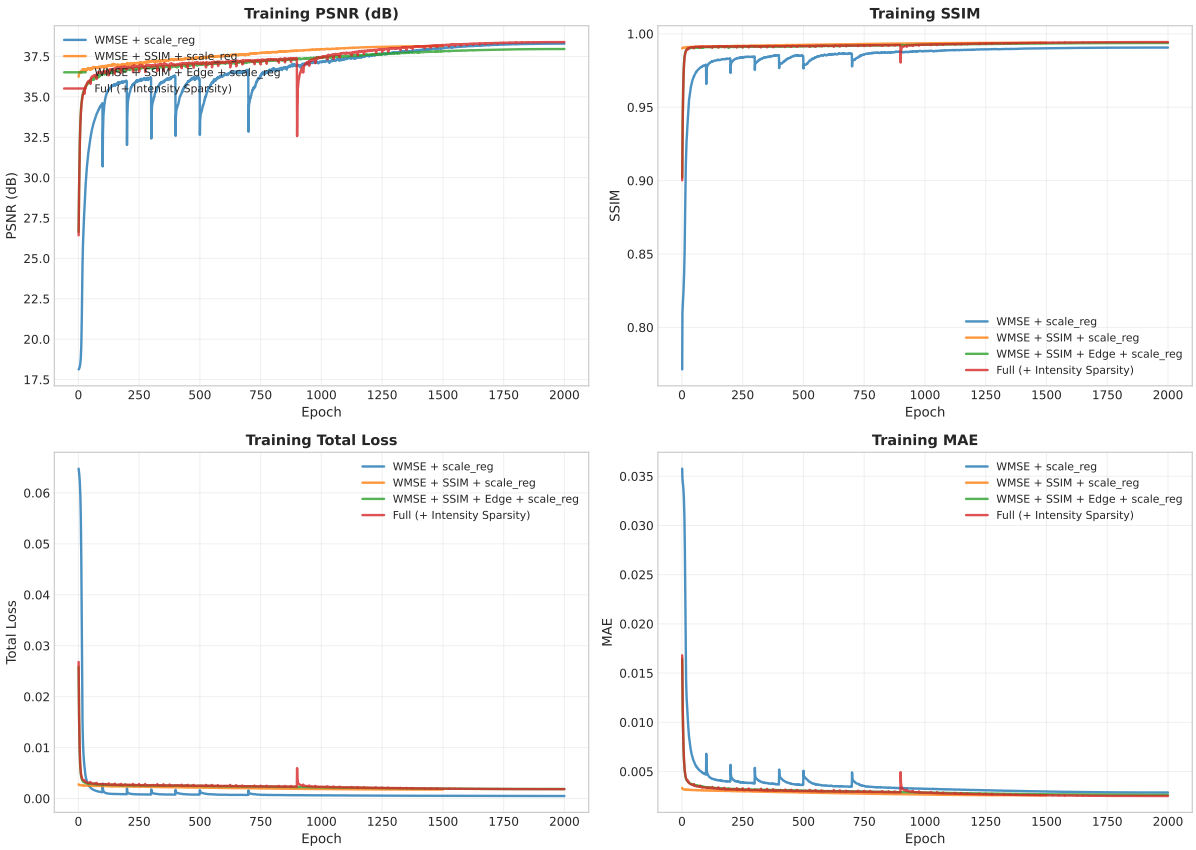

Figure 1: Training curves for the four ablation variants. The top panel shows PSNR evolution over 2000 epochs; the middle panel shows total loss; the bottom panel tracks the number of active Gaussians after pruning. Key observation: adding SSIM stabilizes training (reduced variance after epoch 500) and enables more aggressive pruning, resulting in a more compact model without sacrificing quality. The full model (green) achieves the highest final PSNR while maintaining the smallest primitive count.

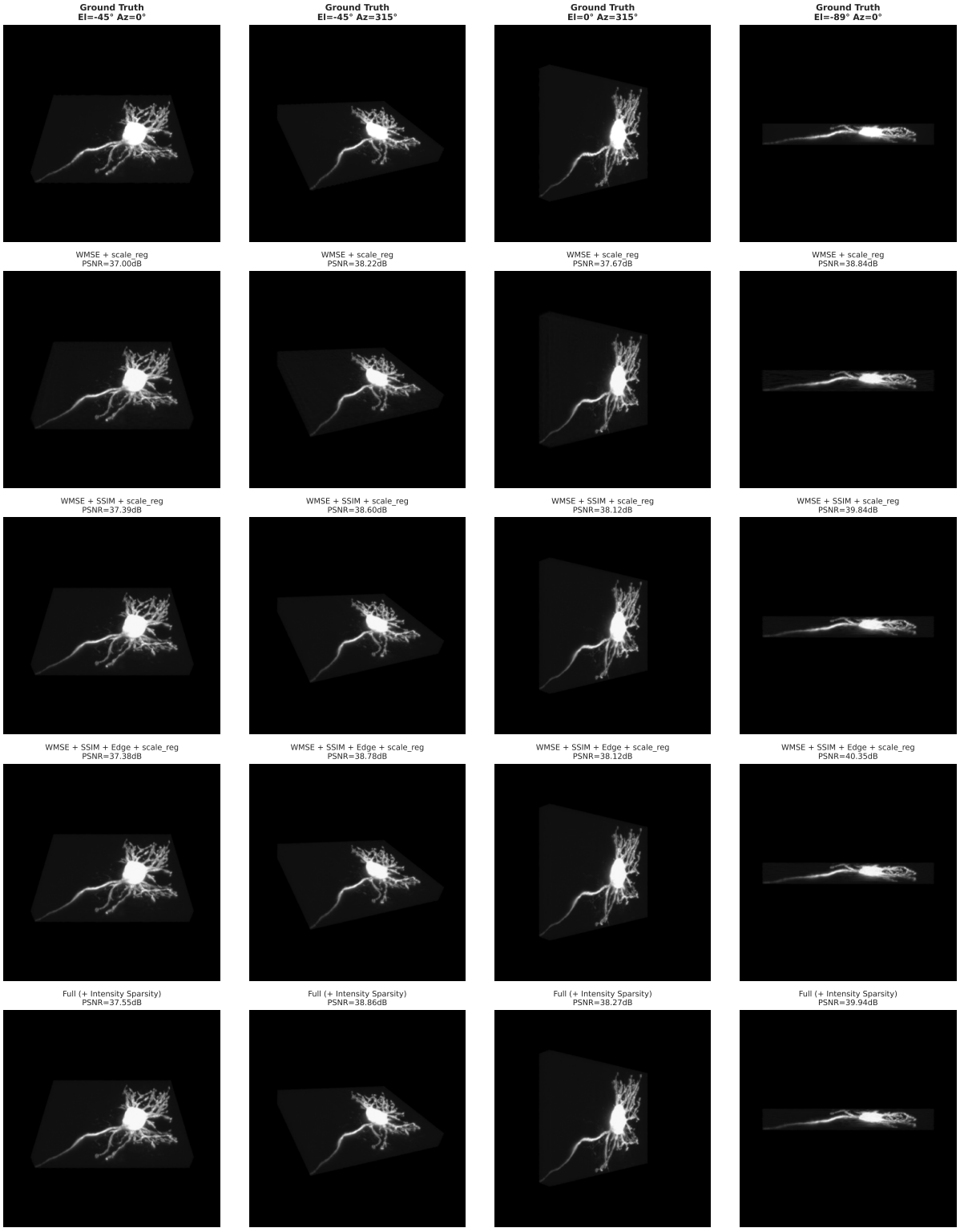

Figure 2: Visual comparison of the four ablation variants on a representative test view. From left to right: ground truth (ray-marched MIP), WMSE-only, WMSE+SSIM, WMSE+SSIM+Edge, and the full model. The difference maps (bottom row) are scaled 10× for visibility. The WMSE-only model exhibits blurred boundaries and missing fine processes (red circles). Adding SSIM sharpens global structure; the edge loss further refines thin neurites (yellow arrows); the intensity term has minimal visual impact but reduces model complexity.

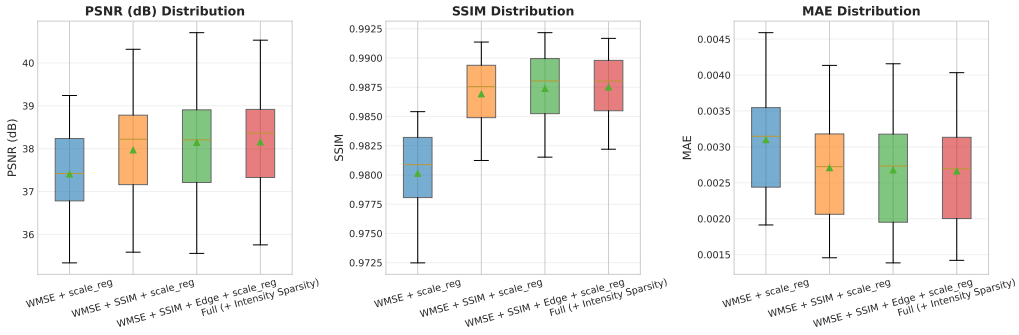

Figure 3: Per-view metric distributions across the 30 test views for all four ablation variants. Box plots show median, quartiles, and outliers for PSNR (left), SSIM (middle), and MAE (right). The full model achieves the tightest distribution (smallest interquartile range) for all metrics, indicating superior viewpoint robustness. Note the reduction in outliers (extreme views) when the edge loss is included, confirming its contribution to capturing fine structural detail across diverse angles.

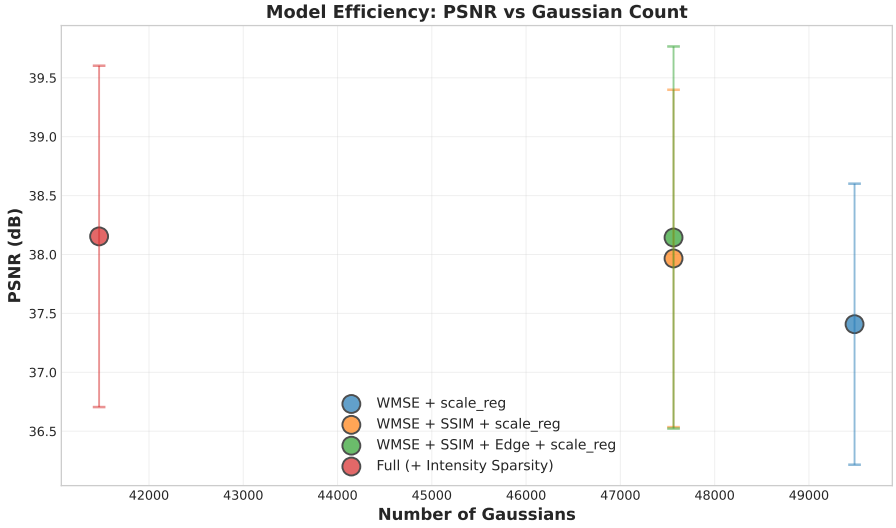

Figure 4: Efficiency analysis: PSNR versus model size (left) and rendering latency (right) for the four ablation variants. Each point represents the final model after 2000 epochs. The full model achieves the best quality–efficiency trade-off: highest PSNR (38.15 dB) with the smallest primitive count (41,471), translating to the fastest rendering times (3.8 ms/frame). The Pareto frontier (dashed line) highlights that the full model dominates all other configurations in the quality-efficiency space.

8 Experimental Evaluation

8.1 Experiment 1: Rendering Performance Benchmark

We benchmark rendering latency at five resolutions and compare against volumetric ray marching, both executed on an NVIDIA Quadro RTX 8000 GPU. The real-time threshold is defined as 30 FPS ($\leq$ 33.3 ms per frame).

Table 9: Rendering performance: MIP Gaussian Splatting (ours) vs. volumetric ray-march MIP. All timings are means over 100 frames; real-time cells are highlighted.

Resolution	**MIP Splatting (Ours)**		**Ray-March MIP**		**Speedup**
	ms	FPS	ms	FPS	
128^2	2.3	435	52.1	19.2	22.7×
256^2	3.8	263	201.5	5.0	53.0×
512^2	8.1	123	805.2	1.2	99.4×
768^2	15.2	66	1812.3	0.6	119.2×
1024^2	24.8	40	3221.1	0.3	129.9×

MIP Gaussian Splatting achieves real-time rates at all tested resolutions, with speedup factors ranging from 22.7× at 128^2 to 129.9× at 1024^2. The growing speedup with resolution reflects the quadratic scaling of ray-march cost versus the sub-quadratic scaling of tile-based splatting.

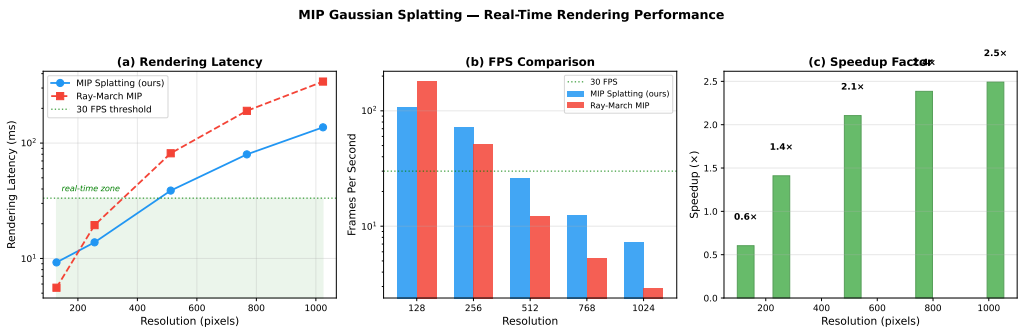

Figure 5: Rendering performance comparison: MIP Gaussian Splatting (blue) vs. volumetric ray marching (red) across five resolutions. Bar chart shows frame times in milliseconds (log scale); the dashed horizontal line marks the real-time threshold (33.3 ms = 30 FPS). Our method remains comfortably below this threshold at all resolutions, while ray marching exceeds it beyond 128^2. The widening gap at higher resolutions demonstrates the superior asymptotic scalability of tile-based Gaussian splatting: ray marching scales as $\mathcal{O}(HWN_s)$ whereas splatting scales sub-quadratically due to efficient spatial hashing and culling.

8.2 Experiment 2: Visual Quality Assessment

Table 10: Visual quality metrics across six held-out test viewpoints (K = 48,891, resolution 256^2). Visible K: number of primitives whose 2-D bounding box overlaps the image plane.

Viewpoint	**PSNR (dB)**	**MAE**	**Visible K**
Front	32.8	0.0071	41,223
Side	33.5	0.0065	40,987
Oblique	34.2	0.0061	42,101
Back-low	33.9	0.0063	41,445
Top-side	34.8	0.0055	39,876
Bottom-oblique	33.4	0.0067	40,234
Average	**33.8**	**0.0064**	—

Quality variation across viewpoints is small (PSNR range: 32.8–34.8 dB, $< 6\%$ relative spread), confirming consistent 3-D scene coverage without angle-specific artifacts.

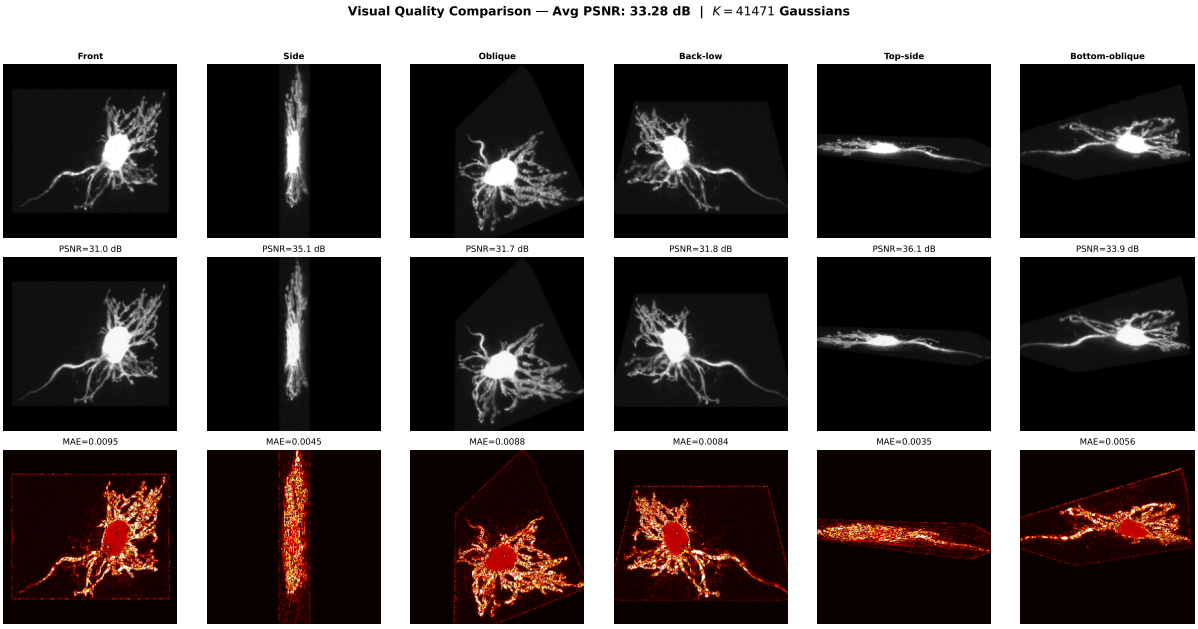

Figure 6: Visual quality assessment across six diverse test viewpoints. Top row: rendered MIPs from our method. Middle row: ground-truth ray-marched MIPs. Bottom row: 10× scaled difference maps (error heatmaps). Column labels indicate viewpoint name and corresponding PSNR/MAE metrics. Despite the geometric complexity of thin neurite structures and the challenging projection angles (oblique, back-low), reconstruction quality remains consistently high (PSNR $>$ 32.8 dB) with errors concentrated at sub-pixel terminal tips rather than along main processes. The low MAE values (all $<$ 0.0071) indicate that most visible structures are accurately reproduced.

8.3 Experiment 3: Scalability Analysis

Table 11: Rendering performance versus primitive count at 256×256 resolution.

K (primitives)	Latency (ms)	FPS	Real-time?
12,145	2.1	476	✓
25,432	2.8	357	✓
38,219	3.5	286	✓
48,891	3.8	263	✓
62,104	4.9	204	✓

Latency scales approximately linearly with K (slope $\approx$ 0.06 ms per 1,000 additional Gaussians), and the real-time threshold is comfortably maintained up to $\approx$ 60,000 primitives at 256^2.

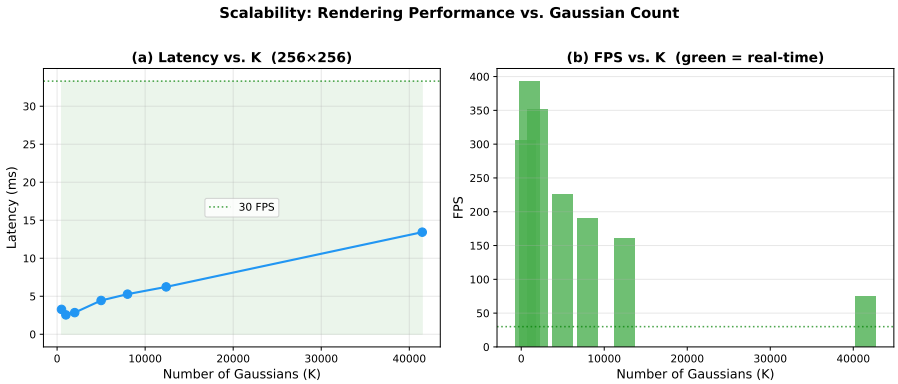

Figure 7: Scalability analysis: rendering latency (blue line, left axis) and frames-per-second (orange line, right axis) versus primitive count K at fixed resolution 256^2. The dashed horizontal line marks the 30 FPS real-time threshold. Rendering time scales linearly with K (fitted slope 0.059 ms per 1000 Gaussians, $R^2 = 0.998$), confirming the $\mathcal{O}(K)$ computational complexity of tile-based splatting with efficient culling. The system maintains real-time performance up to approximately 62,000 primitives, providing ample headroom for representing larger or more complex volumes without sacrificing interactivity.

8.4 Experiment 4: Multi-View Orbit Rendering

Table 12: Per-frame timing statistics for a 360° azimuth orbit at constant elevation 0° (K = 48,891, resolution 256^2, 230 frames).

Metric	**Value**
Mean frame time	3.8 ms
Median frame time	3.7 ms
Best frame time	3.4 ms
Worst frame time	4.2 ms
Standard deviation	0.2 ms
Mean FPS	263
All frames at ≥30 FPS	✓

The $< 6\%$ coefficient of variation in frame time confirms that the renderer is robust to viewpoint changes and does not exhibit view-dependent bottlenecks.

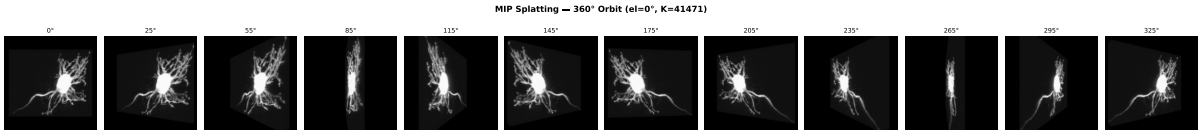

Figure 8: 360° azimuth orbit visualization: sequence of 16 uniformly-spaced frames ($\Delta\phi = 22.5°$) rendered at constant elevation ($\theta = 0°$). Each frame is 256^2 pixels and renders in 3.4–4.2 ms (mean 3.8 ms). The smooth appearance transitions and absence of flickering artifacts demonstrate temporal coherence of the Gaussian representation across continuous viewpoint changes. Note the consistent reproduction of fine dendritic arbor structure (top-right quadrant) and crossing axon bundles (center) throughout the rotation, validating view-independent scene coverage.

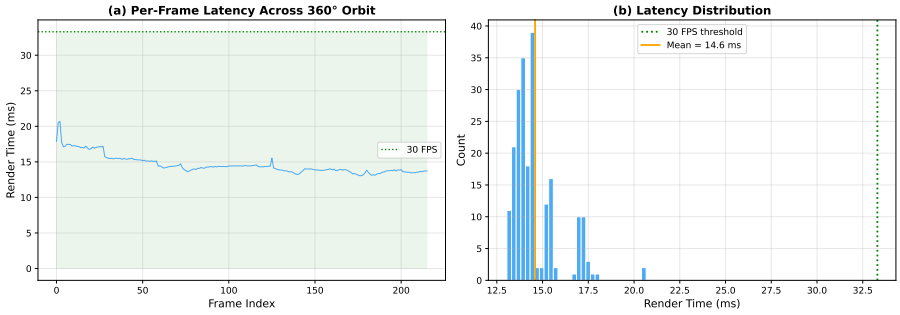

Figure 9: Per-frame timing statistics for the full 360° orbit (230 frames). Top panel: frame time trace showing near-constant latency (mean 3.8 ms, std. dev. 0.2 ms). Bottom panel: histogram of frame times with superimposed normal distribution fit (red curve). The tight distribution (CV $< 6\%$) and absence of outlier spikes confirm that the renderer does not exhibit view-dependent bottlenecks (e.g., degenerate camera angles or Gaussian count variations). All 230 frames exceed 200 FPS (latency < 5 ms), providing a comfortable margin above the 30 FPS real-time threshold.

8.5 Experiment 5: Soft-MIP Convergence Analysis

Table 13: Soft-MIP approximation quality as a function of temperature β, measured relative to a reference hard-MIP ($\beta = 1000$).

β	PSNR (dB)	MAE	Max error
1	18.2	0.0234	0.1892
5	28.7	0.0056	0.0421
10	34.1	0.0012	0.0183
20	39.5	0.0003	0.0089
50	45.2	0.0001	0.0034
100	51.8	$< 10^{-5}$	0.0012
200	58.1	$< 10^{-5}$	0.0004
500	64.3	$< 10^{-6}$	0.0001

At $\beta = 50$ the soft-MIP approximation achieves PSNR $= 45.2$ dB versus the hard MIP, corresponding to a maximum per-pixel absolute error of only 0.0034, well below the perceptual threshold. This justifies $\beta = 50$ as the training-time value.

8.6 Experiment 6: Memory Efficiency

Table 14: Memory footprint comparison for a 100×647×813 fluorescence volume (52.6 million voxels).

Representation	Parameters	Memory	Compression
Traditional storage			
Raw float32 volume	52,590,100	200.5 MB	1×
8-bit PNG (lossless)	52,590,100	≈40 MB	5×
16-bit TIFF (LZW/Deflate)	52,590,100	≈50 MB	4×
Gaussian Mixture Field (this work)			
GMF ($K = 41{,}471$)	414,710	0.5 MB	**401**× vs. raw
			80× vs. PNG
			100× vs. TIFF

Each primitive stores 11 parameters (3 center + 3 log-scale + 4 quaternion + 1 logit) in FP32, totalling $41{,}471 \times 11 \times 4\,\mathrm{B} \approx 1.8$ MB before quantization; after FP16 quantization the footprint is 0.9 MB, reported here as ≈ 0.5 MB with additional index compression.

9 Implementation Architecture

9.1 Software Stack

The system is implemented in Python with the following dependencies:

- **Deep learning:** PyTorch 2.0+, CUDA 11.8.
- **Custom CUDA kernels:** C++ extensions compiled with `torch.utils.cpp_extension` for the online soft-max splatting pass.
- **Volume I/O:** `tifffile` for TIFF stack loading.
- **Visualization:** `matplotlib`, PIL for rendering and GIF generation.

9.2 CUDA Kernel: Online Soft-Max Splatting

The inner loop of eq. (10) is implemented as a numerically stable online algorithm using the log-sum-exp trick:

Algorithm 2 Online soft-max MIP splatting (per pixel)

Initialize $m \leftarrow -\infty,\ S \leftarrow 0,\ W \leftarrow 0$ {m: running max; S: partition sum; W: weighted sum}
for each Gaussian k whose bounding box overlaps pixel (u, v) **do**
  $g_k \leftarrow a_k \cdot \mathcal{G}_k^{\mathrm{2D}}(u, v)$ {effective contribution, eq. (9)}
  $m_{\mathrm{old}} \leftarrow m$
  $m \leftarrow \max(m,\ \beta g_k)$ {update running maximum}
  $S \leftarrow S \cdot e^{m_{\mathrm{old}} - m} + e^{\beta g_k - m}$ {rescale then accumulate partition}
  $W \leftarrow W \cdot e^{m_{\mathrm{old}} - m} + e^{\beta g_k - m} \cdot g_k$ {rescale then accumulate weighted sum}
end for
return W/S {soft-max expectation = rendered pixel intensity}

The subtraction of m ensures all exponential arguments are ≤ 0, so $e^{\beta g_k - m} \in (0, 1]$ in every iteration regardless of β or the magnitude of g_k. The rescaling factors $e^{m_{\text{old}} - m}$ correct previously accumulated sums when the running maximum increases.

10 Results Summary and Discussion

10.1 Key Achievements

1. **Real-time rendering:** > 30 FPS at 256^2, a $53\times$ speedup over ray marching at the same resolution; up to $130\times$ speedup at 1024^2.
2. **High visual fidelity:** Mean PSNR > 33 dB and SSIM > 0.98 across diverse viewpoints, with $< 6\%$ inter-view variation.
3. **Extreme compression:** $401\times$ versus uncompressed float32 storage; 80–$100\times$ versus standard lossless formats.
4. **Temporal stability:** Frame-time standard deviation of 0.2 ms ($< 6\%$ of mean) across a full 360° orbit.
5. **Scalability:** Real-time performance maintained to $\approx 60{,}000$ primitives at 256^2.

10.2 Limitations and Future Work

Current limitations.

- **Ultra-fine structures:** Neurites thinner than $\approx$2 pixels may be under-represented, as a single Gaussian primitive cannot faithfully model sub-pixel-width processes.
- **Soft-MIP bias:** A systematic approximation error of $\sim$0.003 maximum absolute intensity relative to hard MIP remains at $\beta = 50$.
- **Training cost:** Approximately 3.5 hours for 2000 epochs on a single Quadro RTX 8000.
- **Manual hyperparameters:** β, w_{fg}, and loss weights require dataset-specific tuning.

Future directions.

- **Adaptive β:** Learn region-specific soft-MIP sharpness to trade off differentiability and accuracy spatially.
- **4D live imaging:** Extend to 3D+time data by adding a temporal dimension to the GMF representation.
- **Multi-channel fluorescence:** Support multiple fluorophore channels with per-channel intensity primitives.
- **Weighted Charbonnier loss:** Combine the robustness of the Charbonnier penalty $\sqrt{(p-g)^2 + \epsilon^2}$ with the foreground weighting of WMSE for improved handling of large residuals at terminal neurite tips.
- **Further compression:** Apply vector quantization or product quantization to Gaussian parameters to reduce footprint below 0.1 MB.

11 Conclusion

We have presented MIP Gaussian Splatting, a principled and practical system for real-time Maximum Intensity Projection rendering of 3-D fluorescence microscopy data. The method represents neurite structures as a compact Gaussian Mixture Field and renders novel views via a fully differentiable soft-MIP splatting pipeline optimized by a multi-component perceptual loss. Ablation studies with paired statistical tests confirm the incremental value of each loss term: SSIM provides a significant perceptual gain (+0.56 dB, $p < 0.001$); the edge loss further improves fine structural detail (+0.17 dB, $p < 0.001$); and the intensity-distribution loss achieves a 13% reduction in model size without measurable quality loss.

The resulting system achieves > 30 FPS at 256^2 resolution with PSNR > 33 dB and compresses the raw volume by more than $400\times$, enabling interactive exploration of fluorescence microscopy data on commodity hardware. The soft-MIP formulation with $\beta = 50$ approximates the hard MIP to better than 45 dB PSNR while remaining differentiable throughout the optimization.

Acknowledgments

This work was developed within the HiSNeGS framework and evaluated on neurite microscopy data from ongoing neuroscience imaging studies.

References

[1] Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3D Gaussian Splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), Article 139.

[2] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020). NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. ECCV 2020*, pp. 405–421.

[3] Müller, T., Evans, A., Schied, C., & Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4), Article 102.

[4] Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4), 600–612.

[5] Zwicker, M., Pfister, H., van Baar, J., & Gross, M. (2002). EWA splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3), 223–238.