# RadSplat: Radiance Field-Informed Gaussian Splatting for Robust Real-Time Rendering with 900+ FPS

Michael Niemeyer, Fabian Manhardt, Marie-Julie Rakotosaona, Michael Oechsle, Daniel Duckworth, Rama Gosula, Keisuke Tateno, John Bates, Dominik Kaeser, and Federico Tombari
Google
ECCV - Mar 2024
Presented by: Amir Alimohammadi

# Introduction

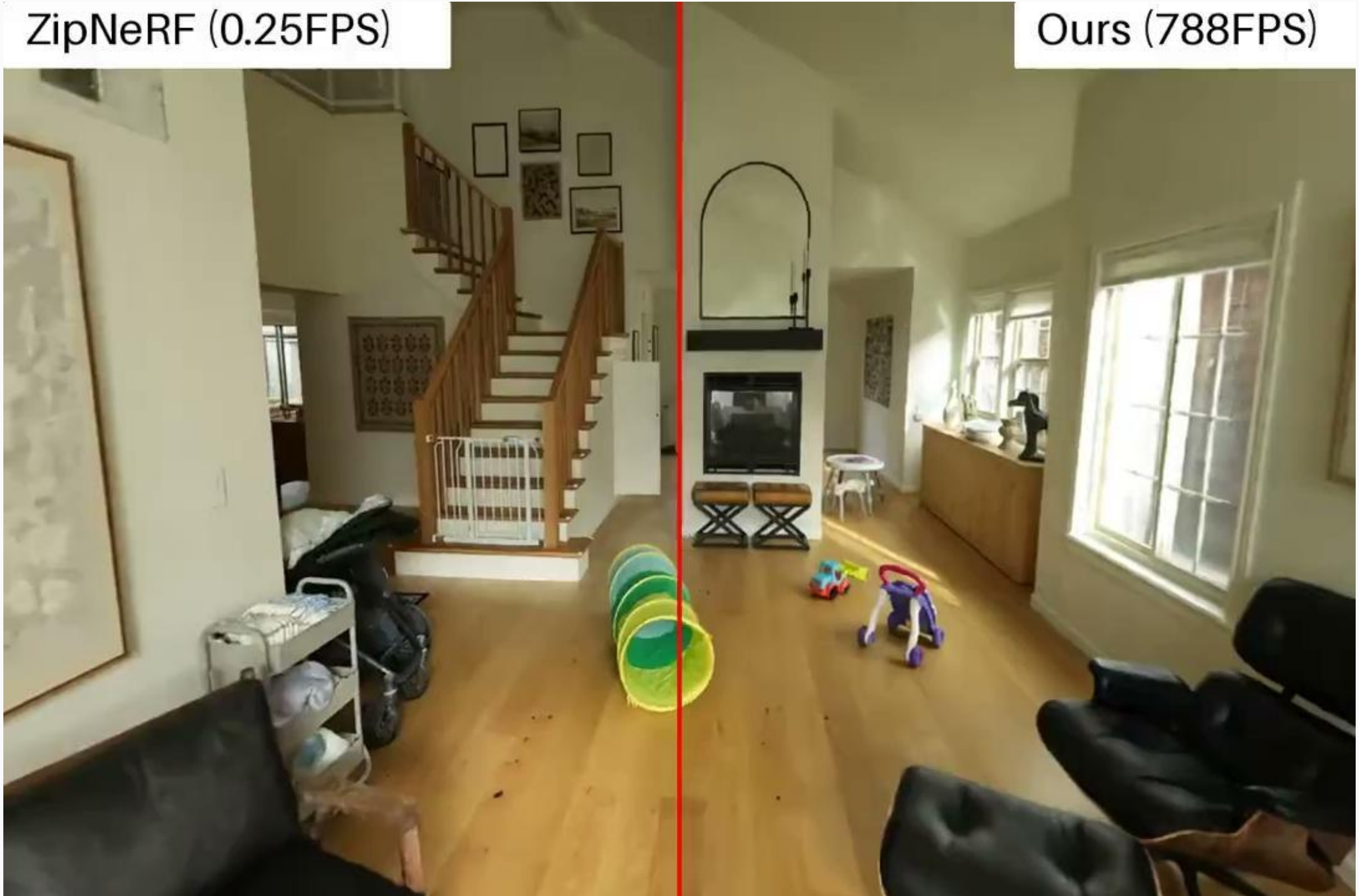


3D Gaussian Splatting


Zip-NeRF


Ours

RadSplat enables high-quality real-time rendering of complex large-scale scenes at 900+ FPS.


ZipNeRF (0.25FPS)
Ours (788FPS)

# Proposed Method

Three key ideas:

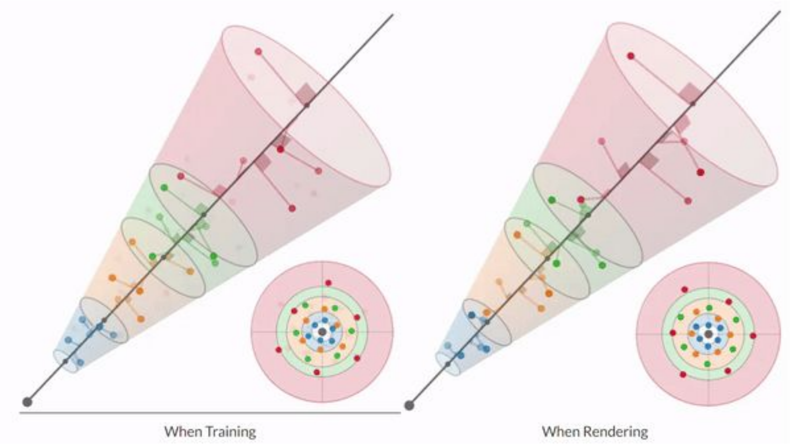
1. Neural Radiance Fields as a Prior

2. Radiance Field-Informed Gaussian Splatting

3. Post Processing

# Neural Radiance Fields as a Robust Prior

$$\mathcal{L}(\theta) = \sum_{\mathbf{r} \in \mathcal{R}_{\text{batch}}} \|\mathbf{c}^{\theta}_{\text{NeRF}}(\mathbf{r}) - \mathbf{c}_{\text{GT}}(\mathbf{r})\|_2^2$$



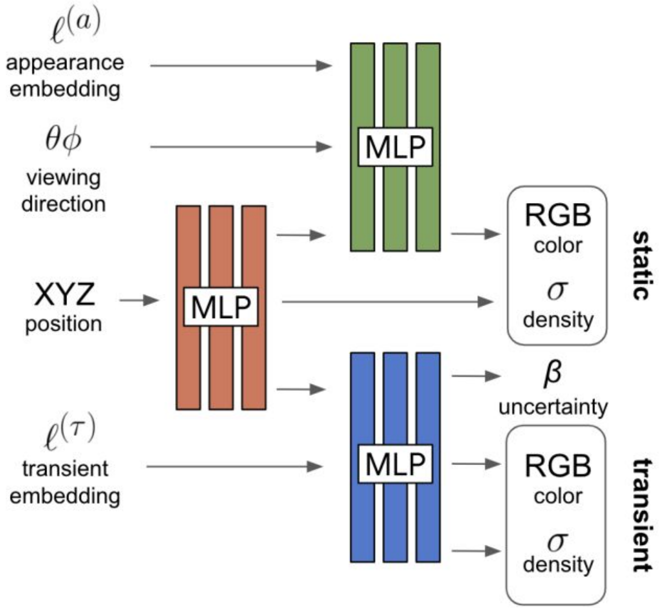

θ: network parameters

$r \in R_{batch}$: batches of rays sampled from the set of all pixels / rays R

Zip-NeRF (J. Barron, et al - ICCV 2023)

# Generative Latent Optimization (GLO)

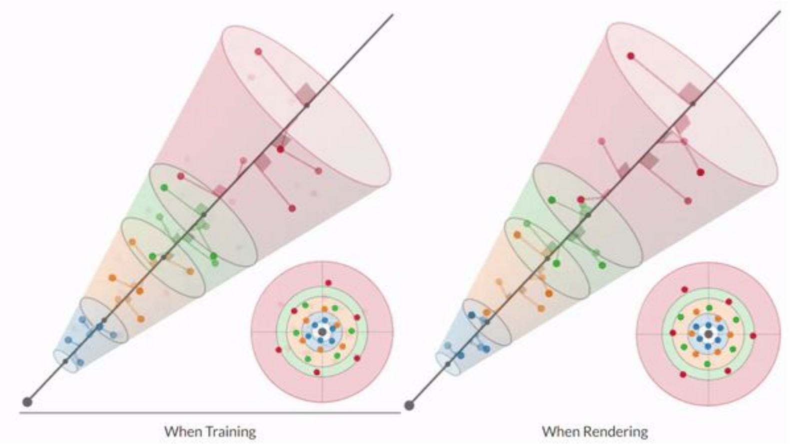
A per-image latent vector is optimized along with the neural field that enables explaining away the view-dependent effects.




NeRF in the Wild (R. Martin-Brualla, et al - CVPR 2021) - GLO (P. Bojanowsk, et al - ICML 2018)

# Robust Optimization on Real-World Data

$$\mathcal{L}(\theta, \{\mathbf{l}_i\}_{i=1}^{N}) = \sum_{\mathbf{r}_i \in \mathcal{R}_{\text{batch}}} \|\mathbf{c}_{\text{NeRF}}^{\theta, l_i}(\mathbf{r}_i) - \mathbf{c}_{\text{GT}}(\mathbf{r}_i)\|_2^2$$




θ: network parameters

$r \in R_{batch}$: batches of rays sampled from the set of all pixels / rays R

$l_i$: The set of GLO vectors

N: the number of input images

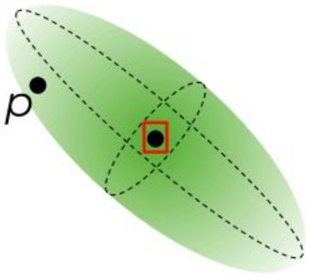
Zip-NeRF (J. Barron, et al - ICCV 2023)

# 3D Gaussian Splatting

3D Gaussian Splatting: Explicit point-based scene representation.

The impact of a 3D Gaussian i on an arbitrary 3D point p in 3D is defined as follows:

$$f_i(p) = \sigma(\alpha_i)\exp(-\frac{1}{2}(p - \boxed{\mu_i})\Sigma_i^{-1}(p - \boxed{\mu_i}))$$



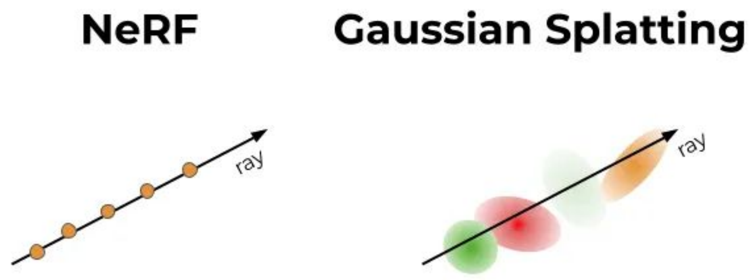

$$C(p) = \sum_{i \in N} c_i f_i^{2D}(p) \underbrace{\prod_{j=1}^{i-1}(1 - f_j^{2D}(p))}_{transmittance}$$

Where $f^{2D}(p)$ is a projection of f(p) onto an image plane of the camera that is being rendered.

# New parameters in the paper

Replace $f_j(p)$ with $\alpha_j$.

$$\mathbf{c}_{\mathrm{GS}} = \sum_{j=1}^{N_p} \mathbf{c}_j \alpha_j \tau_i \quad \text{where} \quad \tau_i = \prod_{i=1}^{j-1} (1 - \alpha_i)$$




NeRF equation from previous slides:

$$\mathbf{c}_{\mathrm{NeRF}} = \sum_{j=1}^{N_s} \tau_j \alpha_j \mathbf{c}_j \quad \text{where} \quad \tau_j = \prod_{k=1}^{j-1} (1 - \alpha_k), \quad \alpha_j = 1 - e^{-\sigma_j \delta_j}$$

# Radiance Field-Informed Gaussian Splatting

Volume rendering paradigm:

Allows for flexible manipulation of 3D space.

Density control:

Enables free initialization, removal, and modification of density in 3D.

Contrast to point-based representations:

Point-based methods are limited to gradient signals for existing geometry.

Initialization advantage:

Radiance fields offer easier initialization in the optimization process.

# How can NeRF priors be leveraged to improve initialization?

Rendering the median depth from our NeRF model for each ray:

$$z_{\text{median}} = \sum_{i=1}^{N_z} \alpha_i \|\mathbf{x}_i\|_2 \quad \text{where} \quad \tau_{N_z} \geq 0.5 \quad \text{and} \quad \tau_{N_z-1} < 0.5$$

Project all rays into 3D space to obtain our initial point set:

$$\mathcal{P}_{\text{init}} = \{\mathbf{p}_i\}_{i \in \mathcal{K}_{\text{random}}} \quad \text{with} \quad \mathbf{p}_i = \mathbf{r}_0(i) + \mathbf{d}_{r(i)} \cdot z_{\text{median}}(\mathbf{r}(i))$$

$K_{random}$ : A set of randomly selected indices from the entire set of rays.

$r_0(\cdot)$: The ray origin

$d_r(\cdot)$: The normalized ray direction

# Radiance Field-based Supervision

Radiance Field-based Methods:

Achieve state-of-the-art quality in view synthesis.

Handle challenging scenarios like in-the-wild captures and large-scale scenes.

Radiance Field-based Supervision

Generate cleaner supervision signals.

Rendering input images with a NeRF model and a constant zero GLO vector.

$$\mathcal{I}_f = \{I_f^j\}_{j=1}^N \quad \text{where} \quad I_f^j = \{\mathbf{c}_{\text{NeRF}}^{\theta, l_{\text{zero}}}(\mathbf{r}_j(i))\}_{i=1}^{H \times W}$$

# Training

$$\mathcal{I}_f = \{I_f^j\}_{j=1}^N \quad \text{where} \quad I_f^j = \{\mathbf{c}_{\text{NeRF}}^{\theta, l_{\text{zero}}}(\mathbf{r}_j(i))\}_{i=1}^{H \times W}$$

$l_{zero}$: Indicates the zero GLO vector

H & W: The height and the width images

$r_j$ (·): The rays belonging to the j-th image

**Loss Function:**

$$\mathcal{L}(\phi) = (1-\lambda)\|I_f^i - I_\phi^i\|_2^2 + \lambda\,\text{SSIM}(I_f^i, I_\phi^i) \quad \text{with} \quad i \sim \mathcal{U}(N)$$

U: The uniform distribution

# Ray Contribution-Based Pruning

All the techniques used by now have not helped us to achieve substantial speed improvements yet.

The number of points in a scene directly impacts rendering performance.

Proposed Solution:

- Develop a novel pruning technique to reduce the number of Gaussians.
- Maintain high-quality rendering output while accelerating performance across platforms.

# Pruning

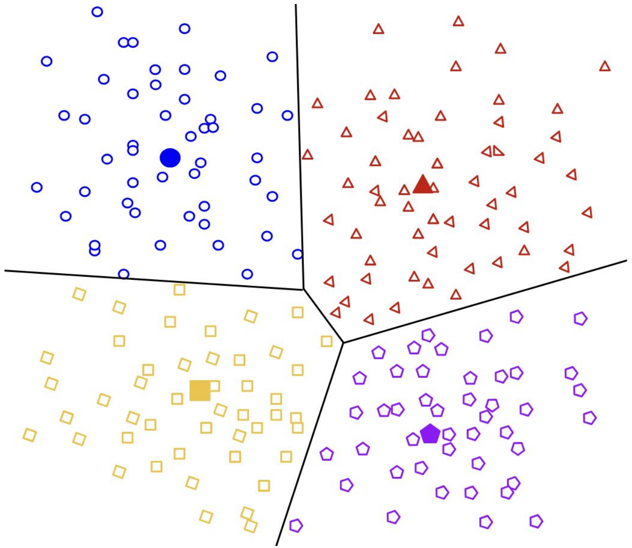
Defining an importance score by aggregating the ray contribution of Gaussian $p_i$ along all rays of all input images.

$$h(\mathbf{p}_i) = \max_{I_f \in \mathcal{I}_f, r \in I_f} \alpha_i^r \tau_i^r$$

Pruning step:

$$m_i = m(\mathbf{p}_i) = \mathbb{1}\left(h(\mathbf{p}_i) < t_{\text{prune}}\right) \quad \text{where} \quad t_{\text{prune}} \in [0, 1]$$

# Viewpoint-Based Visibility Filtering



## Viewpoint-Based Visibility Filtering

Handle larger, more complex scenes (houses, apartments).

Accelerate test-time rendering without quality loss.

## Input Camera Clustering

Group input cameras based on their locations to create a scene tessellation.

Employ k-means clustering on camera positions to identify cluster centers.

Assign cameras to corresponding cluster centers.

# Viewpoint-Based Visibility Filtering

Visibility Filtering:

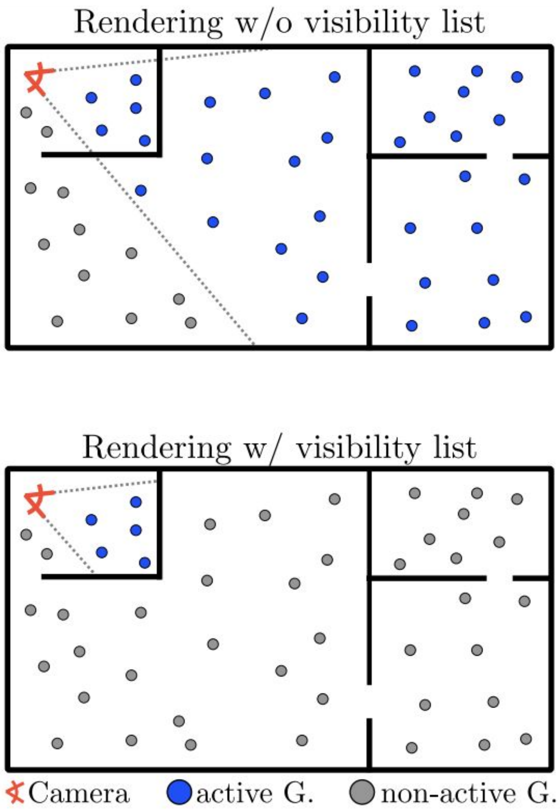
For each cluster center, select associated input cameras.

Render images from these camera viewpoints.

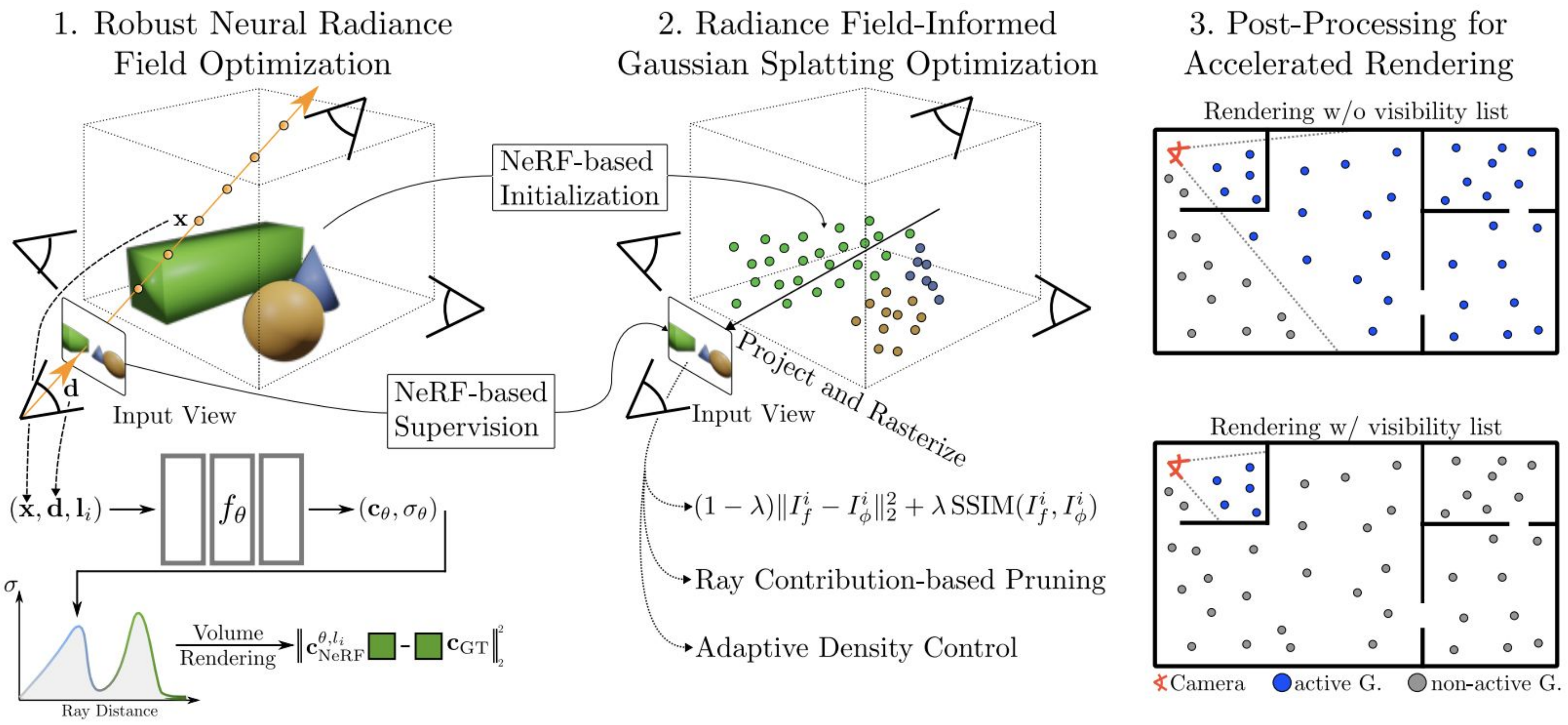
Compute importance scores and visibility indicator lists.

$$h_j^{\text{cluster}}(\mathbf{p}_i) = \max_{I \in \mathcal{I}_c^i, r \in I} \alpha_i^r \tau_i^r, \quad m_j^{\text{cluster}} = \mathbb{1}\left(h_j^{\text{cluster}}(\mathbf{p}_i) > t_{\text{cluster}}\right)$$

$I^i_c$: All images captured from cameras assigned to a cluster $X^i$
$t_{\text{cluster}}$: Controls the points that should be filtered out.



# Method Overview



# Quantitative Comparison

| | SSIM↑ | PSNR↑ | LPIPS↓ | FPS↑ | #G(M)↓ |
|---|---|---|---|---|---|
| INGP [35] | 0.705 | 25.68 | 0.302 | 9.26 | - |
| BakedSDF [73] | 0.697 | 24.51 | 0.309 | 539 | - |
| MERF [56] | 0.722 | 25.24 | 0.311 | 171 | - |
| SMERF [10] | 0.818 | 27.99 | 0.211 | 228 | - |
| CompactG [23] | 0.798 | 27.08 | 0.247 | 128 | 1.388 |
| LightG [11] | 0.799 | 26.99 | 0.25 | 209 | 1.046 |
| EAGLES [14] | 0.809 | 27.16 | 0.238 | 137 | 1.712 |
| 3DGS [21] | 0.815 | 27.20 | 0.214 | 251 | 3.161 |
| **Ours Light** | 0.826 | 27.56 | 0.213 | 907 | 0.370 |
| **Ours** | 0.843 | 28.14 | 0.171 | 410 | 1.924 |
| Zip-NeRF [2] | 0.836 | 28.54 | 0.177 | 0.25 | - |

**(a)** Mip-NeRF360 dataset [1]

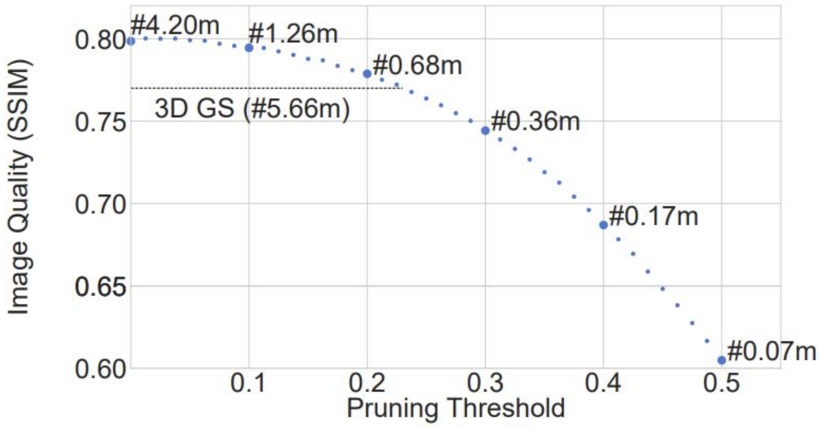
| | SSIM↑ | PSNR↑ | LPIPS↓ | FPS↑ |
|---|---|---|---|---|
| MERF [56] | 0.747 | 23.49 | 0.445 | 318 |
| SMERF [10] ($K=1$) | 0.776 | 25.44 | 0.412 | 356 |
| SMERF [10] ($K=5$) | 0.829 | 27.28 | 0.340 | 221 |
| 3DGS [21] | 0.809 | 25.50 | 0.369 | 470 |
| **Ours Light** | 0.838 | 26.11 | 0.368 | 748 |
| **Ours** | 0.839 | 26.17 | 0.364 | 630 |
| Zip-NeRF [2] | 0.836 | 27.37 | 0.305 | 0.25 |

**(b)** Zip-NeRF dataset [2]

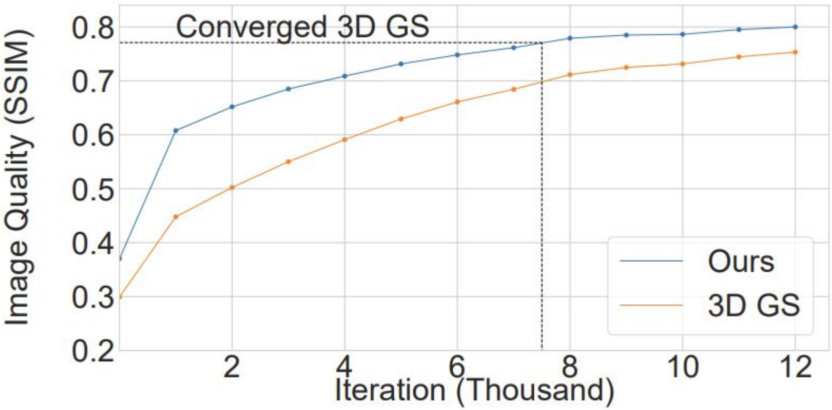
RadSplat reduces the point count by a factor of 10 compared to 3DGS [21] while enhancing image quality. (Light Model)

RadSplat surpasses Zip-NeRF in SSIM and LPIPS metrics, rendering 3,600 times faster. (Default Model)

# Pruning and Optimization Behavior

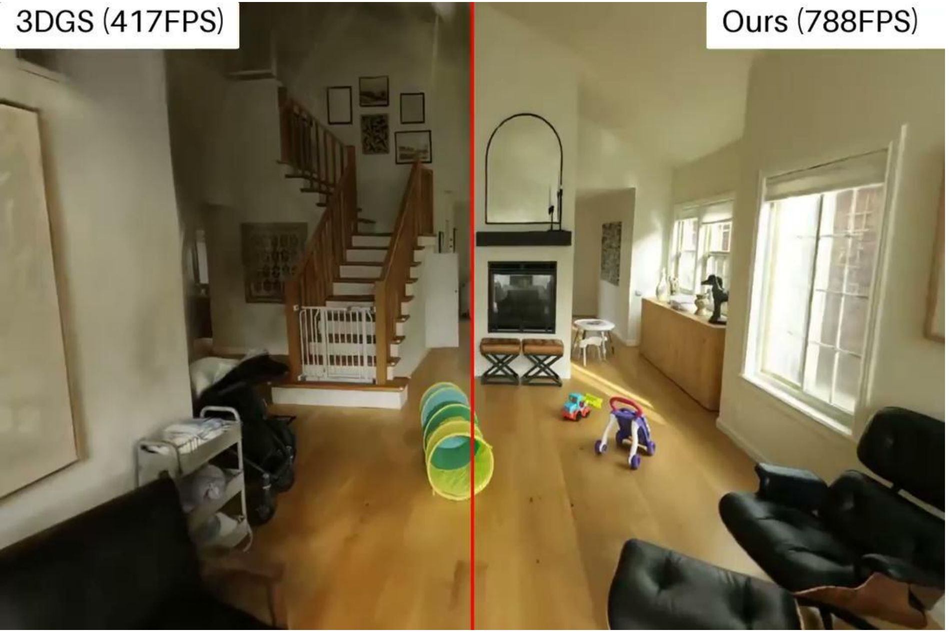


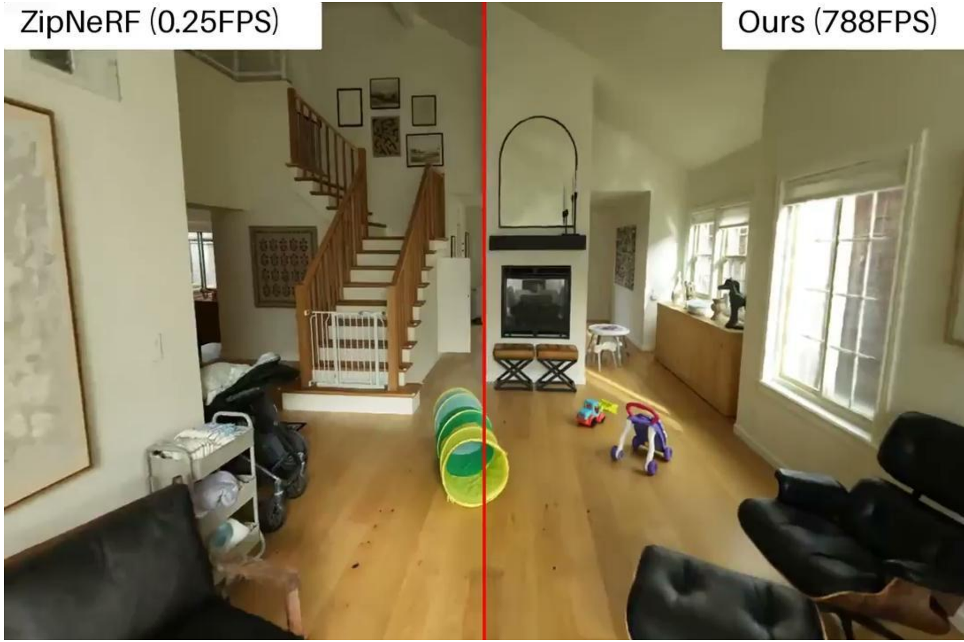

**(a)** SSIM against Pruning Threshold.




**(b)** SSIM against Iteration (in Thousand).

# Qualitative Result

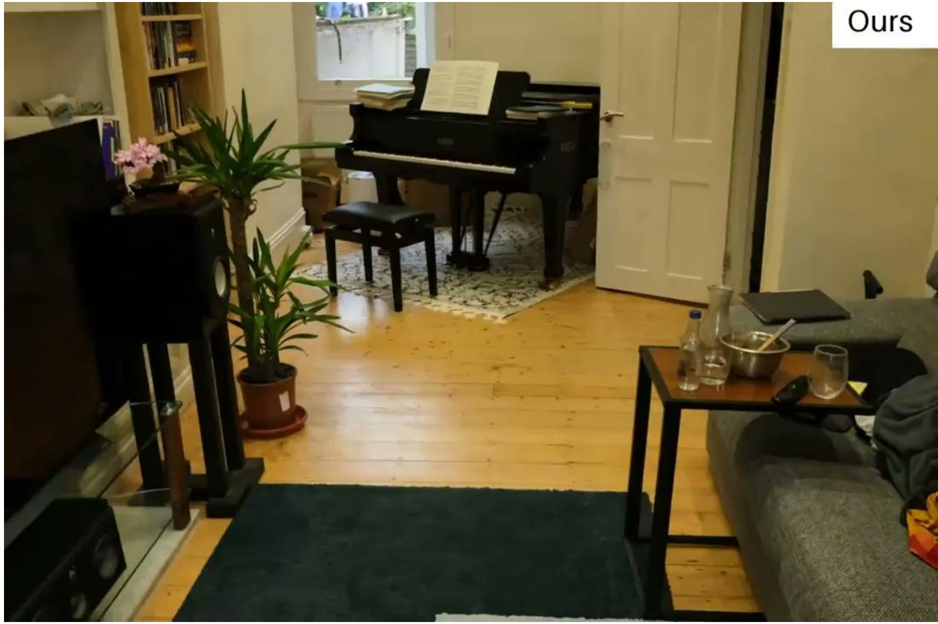


# Qualitative Result


# Qualitative Result


# Qualitative Result

# Thank you!

- Thank you for your attention!
- I appreciate your time and interest.
- If you have any questions, please feel free to ask.
- Contact information: alimohammadiamirhossein@gmail.com